UNIVERSIDAD NACIONAL AUTÓNOMA DE MÉXICO

FACULTAD DE ESTUDIOS SUPERIORES ARAGÓN

TRABAJO POR ESCRITO QUE

PRESENTA:

RICARDO NAVARRO LÓPEZ

TEMA DEL TRABAJO:

"TEORÍA Y CONCEPTOS DE LA ADMINISTRACIÓN DE BASES DE DATOS CONJUNTANDO SU APLICACIÓN BASADA EN SOFTWARE LIBRE DE UN SISTEMA DE SEGUIMIENTO Y CONTROL DE ERRORES DEL MANEJADOR DE BASES DE DATOS SYBASE"

EN LA MODALIDAD DE:

"SEMINARIOS Y CURSOS DE ACTUALIZACIÓN Y CAPACITACIÓN PROFESIONAL"

PARA OBTENER EL TÍTULO DE:

INGENIERO EN COMPUTACIÓN

ASESOR:

ING. VÍCTOR RAÚL VELASCO VEGA

FES Aragón

MÉXICO 2009

Agradecimientos

A la Universidad Nacional Autónoma de México

Agradezco la educación
que por gratuita que esta es
me dio una formación
para hacer mi vida como es

A la Facultad de Estudios Superiores Aragón

Me fue de mucho provecho
de la cercanía de esta Facultad contar
que gracias a ella permitió el hecho
de que el tiempo pudiera aprovechar

A los asesores

Ing. Víctor Raúl Velasco Vega
M. en E. Imelda de la Luz Flores Díaz
Ing. Blanca Estela Cruz Luévano
Ing. Narciso Acevedo Hernández
M. en C. Ricardo Arturo Gutiérrez Orozco

Por ofrecerme su tiempo y guía
en la revisión y corrección,
este trabajo otro sería
sin su valiosa colaboración

A mi familia

Le doy gracias a mi padre y madre
por darme un feliz hogar
ha sido ciertamente admirable
lo que con sus hijos han podido lograr

A mi esposa

Gracias bonita hermosa
por darme tu amor
has sido siempre mi musa
que me ha hecho soñador

Ricardo Navarro López

ÍNDICE

Introducción

La Administración de Bases de Datos es una especialidad dentro del área de la computación que debido a la gran diversidad, volumen e importancia tanto económica como estratégica, que tiene la información en cualquier organización hoy en día, ya sea ésta de una institución pública o privada, implica el uso de sistemas manejadores de bases de datos para garantizar su seguridad, consistencia, integridad, accesibilidad, entre otros factores.

Esto ha creado la inherente necesidad de contar con profesionales de las tecnologías de la información, que tengan los conocimientos y habilidades requeridas para administrar en forma eficaz y eficiente de tan valioso recurso.

Identificar y aplicar los ejemplos de diversos sistemas de gestión de bases de datos relacionales, desde el modelado para el almacenamiento de la información, su implementación, optimización, administración y mantenimiento en un servidor. El comprender la trascendencia de las normas ANSI SQL en cada una de las versiones disponibles, reconocer las principales tareas del administrador, así como establecer las políticas de seguridad y planes de contingencia para llevarlas a la práctica. El mostrar, capturar y explotar la información en un formato de salida como en un navegador de Internet, además de obtener el mayor provecho de la información son una gama de actividades que dan a una institución una solidez y para lograrla se necesita el obtener conocimientos diversos como los que se integran en este documento.

Capítulo 1. Sistemas de información y el modelo de datos relacional. Identificar todos los elementos que conforman una base de datos y diferenciarlos del manejo de archivos de datos; para ser capaz de conceptualizar hechos del mundo real como un modelo de datos, utilizando la teoría relacional para su representación.

Capítulo 2. Sistemas manejadores de bases de datos relacionales (RDBMS). Identificar la función, los componentes y las normas que debe de cumplir internacionalmente para su elección en el entorno de trabajo, así como su importancia en la actualidad. Analizando la función de las RDBMS, sus componentes, actualizaciones del SQL ANSI 89, 92 y 99, principales RDBMS, consideraciones de hardware, tipos de datos usados por los distintos RDBMS, tipos de usuarios, lenguajes transaccionales y la conectividad.

Capítulo 3. SQL (Structured Query Language). Conocer y aplicar el lenguaje SQL para la definición, control y consulta de la información necesaria en el desarrollo de aplicaciones. Utilizar el SQL para extraer la información necesaria a través de la elaboración de consultas que se puedan requerir y de la programación de los procedimientos a utilizar, así mismo, entender la ventaja de conocer las normas ANSI SQL y la sintaxis de las principales instrucciones del SQL dentro de la migración de consultas y procedimientos de un RDBMS a otro.

Capítulo 4. Acceso a datos a través de la programación de clientes. Utilizar distintos elementos que brindan los lenguajes de programación, para la consulta y manipulación de información en las bases de datos.

Capítulo 5. Fundamentos de sistemas operativos. Emplear los comandos y herramientas básicas de administración que intervienen en la configuración de un servidor en diferentes plataformas en las cuales puede residir el RDBMS basándose principalmente en sistemas operativos tipo UNIX para considerar sus elementos.

Capítulo 6. Habilidades directivas para administradores. Comprender la función del liderazgo y las formas para evitar la resistencia al cambio organizacional. Mejorar la capacidad para realizar presentaciones efectivas. Lograr una mejor administración del tiempo en relación con las juntas de trabajo. Son habilidades directivas necesarias en todo administrador de bases de datos.

Capítulo 7. Administración de bases de datos. La responsabilidad de procurar controlar los aspectos ambientales que tienen las bases de datos. En general estos incluyen la recuperación de las bases ante una falla o perdida (crear y probar respaldos). Verificar o ayudar a la verificación en la integridad de datos. En seguridad definir y/o implementar controles de acceso a los datos. Para la disponibilidad con asegurarse del mayor tiempo de funcionamiento. Con el desempeño el asegurarse de obtener el máximo trabajo incluso con las limitaciones. Para el desarrollo, soporte y pruebas con ayudar, controlar y orientar al programador a utilizar eficientemente la base de datos.

Capítulo 8. Buenas prácticas en la función de la administración. Fundamentos de la auditoria informática y controles, en el ámbito de la administración de bases de datos, así como las mejores prácticas a seguir en la administración de bases de datos. Estas buenas prácticas se fundamentan en las recomendaciones del COBIT Control OBjectives for Information and related Technology - Objetivos de control relacionadas a la tecnología de la información.

Capítulo 9. Seguridad en base de datos. Distinguir las principales vulnerabilidades en seguridad en el entorno de trabajo para reforzar, tomando las medidas necesarias para su corrección, el total aprovechamiento de los elementos y bondades ofrecidos por las RDBMS.

Capítulo 10. Performance and tunning. Entender la razón de reconfigurar una RDMS, usar herramientas usadas para reconfigurar, optimizar la asignación de memoria, ver como los parámetros relacionados con recursos afectan la asignación de memoria, entender las principales opciones de configuración. Estos conceptos a pesar de mostrarse básicamente para Sybase / SQL Server reflejan las características que se debieran optimizar en las diferentes RDBMS.

Capítulo 11. Modelo orientado a objetos. Principales características de la metodología orientada a objetos, como lo son: su modelado y su enfoque dentro de las empresas y organizaciones.

Capítulo 12. Tópicos avanzados data mining. ¿Cómo puede entenderse un fenómeno sobre la base de la interpretación de grandes volúmenes de datos? ¿De qué manera puede utilizarse la información para la toma de decisiones?, son algunos ejemplos de interrogantes comunes. La respuesta a estas preguntas es el objetivo del data mining, un conjunto de técnicas agrupadas con el fin de crear mecanismos adecuados de dirección, entre ellas puede citarse la estadística, el reconocimiento de patrones, la clasificación y la predicción. Aunque de manera teórica este apartado comenta estas inquietudes.

Capítulo 13. Para aterrizar los conocimientos obtenidos en los capítulos descritos anteriormente aplicándolos a un caso práctico se considera el acceso a datos a través de la programación de clientes, el uso de un lenguaje de programación de software libre PHP para implementar un sistema totalmente funcional que ayuda en uno de los aspectos de la administración de bases de datos para asegurar la disponibilidad, cerciorándose del mayor tiempo de funcionamiento de un RDBMS propietario como es Sybase apoyándose por software libre con PHP, PostgreSQL y Mozilla.

OBJETIVOS

1. Dejar asentados los conocimientos adquiridos y contenidos teóricos del diplomado de administración de bases de datos en un medio electrónico e impreso. Asimismo permitir su fácil aprovechamiento para que el administrador de bases de datos pueda usarlo como material de referencia en su trabajo cotidiano.

2. Aplicar los conocimientos adquiridos en el diplomado de administración de bases de datos para realizar una aplicación de tres capas (Capa de presentación *interfaz de usuario* – Capa de aplicación *lenguaje de programación* – Capa de datos *servidor de bases de datos*) que demuestre como opera la conjunción de:

    - Sistema operativo (CentOS 5).
    - Servidor Web (Apache).
    - Lenguaje de programación (PHP).
    - Base de datos (PostgreSQL).
    - Navegador Web como interfaz de usuario (Navegador Web Mozilla Firefox).

    Con el objetivo de ayudar, a un bajo costo, a disminuir el tiempo en que se lleva encontrar una solución al ocurrir un error en el Sistema Manejador de Bases de Datos Relacional Sybase para asegurar la disponibilidad de este RDBMS propietario de costo de licencia aproximado de US$32,0000 por CPU. Poder documentar en la aplicación el error, para darle seguimiento a éste y tenerlo como referencia para un caso similar en el futuro.

Capítulo 1

SISTEMAS DE INFORMACIÓN Y EL MODELO DE DATOS RELACIONAL

Identificar todos los elementos que conforman una base de datos y diferenciarlos del manejo de archivos de datos; para ser capaz de conceptualizar hechos del mundo real como un modelo de datos, utilizando la teoría relacional para su representación.

1.1. DEFINICIÓN DE DATO

Para conceptualizar el significado de dato se puede recurrir a las siguientes definiciones:

- Un dato es la unidad mínima de información, hechos sin valuar ó un valor sin significado.
- Hechos, ideas o conceptos que pueden ser reunidos y representados para dar lugar a una idea especifica.
- Representación simbólica (numérica, alfabética, etc.), de un atributo o característica de una entidad.
- Un dato, dentro de una Base de Datos, responde a la función: (Objeto, Atributo, Valor) Por ejemplo: Empleado, Edad, 35 años.
- De forma genérica se dice que un dato se puede definir como un hecho aislado y en bruto, que debe ser procesado por varias operaciones para obtener resultados relacionados con la evaluación e identificación de personas, eventos y objetos.

1.2. DEFINICIÓN DE LA INFORMACIÓN

Partiendo de las definiciones de lo que es el dato será la base para integrar definiciones de lo que es la información integrándolo en los siguientes conceptos:

- La información es un conjunto de datos interrelacionados entre sí, que tienen un significado del cual podemos obtener conocimientos para una futura toma de decisiones.
- La información se obtiene asociando los hechos en un contexto determinado, es decir, la adición o el procesamiento de los datos, proporcionan el conocimiento o entendimiento de ciertos factores.
- La información es un acontecimiento o una serie de acontecimientos, que llevan un mensaje y que al ser percibida por el receptor mediante alguno de sus sentidos, amplía sus conocimientos, en esta relación sólo el destinatario puede evaluar el significado y utilidad de la información recibida.

1.3. ANÁLISIS DE LA INFORMACIÓN.

El análisis de la información es un modelo de datos que consiste en la representación conceptual de la problemática que se desea resolver y cuya característica primordial es la claridad de su contenido.

La información es un recurso sumamente valioso para cualquier organización, sin embargo, la obtención de la información formal genera gastos y su valor sólo puede ser comparado con el valor que tendrá para el receptor final.

Algo importante de mencionar es que el costo de producción de la información es tangible y se puede medir gracias a los dispositivos y medios utilizados, pero la información es conceptual por naturaleza y no tiene características tangibles salvo representaciones simbólicas.

Características del valor de la información:

Accesible.	Es la facilidad y rapidez con que se obtiene la información resultante.
Clara.	Se refiere a la integridad y entendimiento de la información sin ambigüedades.
Precisa.	Que sea lo más exacta posible.
Propia.	Debe de haber relación entre el resultado y lo solicitado por el usuario.
Oportuna.	Menor duración del ciclo (entrada, procesamiento y entrega al usuario).
Flexible.	Adaptabilidad de la información a la toma de decisiones.
Verificable.	Que se pueda examinar la información.
Imparcial.	No se puede alterar o modificar la información (sólo por el dueño).
Cuantificable.	Todo dato procesado produce información.

1.4. DEFINICIÓN DE UNA BASE DE DATOS.

Una base de datos o banco de datos es un conjunto de datos que pertenecen al mismo contexto almacenados sistemáticamente para su posterior uso, siendo así un conjunto de datos relacionados entre sí con un objetivo común. La base de datos debe representar una colección integrada, estructurada y generalizada de datos. Debe de atender a las relaciones naturales de modo que suministre todos los caminos de acceso necesarios a cada unidad de datos con objeto de poder atender todas las necesidades de los diferentes usuarios. En este sentido, una biblioteca puede considerarse una base de datos compuesta en su mayoría por documentos y textos impresos en papel e indexados para su consulta.

En la actualidad, y gracias al desarrollo tecnológico de campos como la informática y la electrónica, la mayoría de las bases de datos tienen formato electrónico, que ofrece un amplio rango de soluciones al problema de almacenar datos.

Dentro de las Tecnologías de la Información existen sistemas gestores de bases de datos (SGBD[1]), que permiten almacenar y acceder a los datos de forma estructurada y rápida donde se deben de mantener los siguientes aspectos:

Colección de datos integrados, con redundancia controlada y con una estructura que refleje las interrelaciones y restricciones existentes en el mundo real; los datos que han de ser compartidos por diferentes usuarios y aplicaciones, deben mantenerse independientes de éstas, y su definición y descripción, únicas para cada tipo de datos, han de estar almacenadas junto con los mismos. Los procedimientos de actualización y recuperación han de ser comunes y bien determinados, habrán de ser capaces de conservar la integridad, seguridad y confidencialidad del conjunto de datos.

Tener una colección de tablas interrelacionadas. El contenido de una base de datos engloba a la información concerniente de una organización, de tal manera que los datos estén disponibles para los usuarios en tiempo real y son compatibles con usuarios concurrentes, una finalidad de la base de datos es eliminar la redundancia o al menos minimizarla.

Los tres componentes principales de un sistema de base de datos son el hardware, el software DBMS y los datos a manejar, así como el personal encargado del manejo del sistema.

Tipos de bases de datos.

Las bases de datos pueden clasificarse de varias maneras, de acuerdo al criterio elegido para su clasificación:

Según la variabilidad de los datos almacenados:

Bases de datos estáticas.	Son bases de datos de sólo lectura, utilizadas primordialmente para almacenar datos históricos que posteriormente se pueden utilizar para estudiar el comportamiento de un conjunto de datos a través del tiempo, para así realizar proyecciones y tomar decisiones.
Bases de datos dinámicas.	Son bases de datos donde la información almacenada se modifica con el tiempo, permitiendo operaciones como actualización y adición de datos, además de las operaciones fundamentales de consulta.

Según el contenido:

Bases de datos bibliográficas.	Solo contienen un representante de la fuente primaria, que permite localizarla. Un registro típico de una base de datos bibliográfica contiene información sobre el autor, fecha de publicación, editorial, título, edición, de una determinada publicación, etc. Puede contener un resumen o extracto de la publicación original, pero nunca el texto completo, porque sino se estaría en presencia de una base de datos a texto completo (o de fuentes primarias).

[1] SGBD. Sistemas Gestores de Bases de Datos más conocido en su acrónimo en inglés como DBMS.

Bases de datos numéricas.	Como su nombre lo indica, el contenido son cifras o números. Por ejemplo, una colección de resultados de análisis de laboratorio, entre otras.
Bases de datos de texto completo.	Almacenan las fuentes primarias, como por ejemplo, todo el contenido de todas las ediciones de una colección de revistas científicas.
Directorios.	Un ejemplo son las guías telefónicas en formato electrónico.
Bancos de datos de información Biológica.	Son bases de datos que almacenan diferentes tipos de información proveniente de las ciencias de la vida o médicas. Se pueden considerar en varios subtipos: • Aquellas que almacenan secuencias de nucleótidos o proteínas. • Las bases de datos de rutas metabólicas. • Bases de datos de estructura, comprende los registros de datos experimentales sobre estructuras 3D de biomoléculas. • Bases de datos clínicas. • Bases de datos bibliográficas (biológicas).
Bancos de imágenes, audio, video, multi-media.	Como se indica almacenan información gráfica, de audio o de carácter audiovisual.

Flat file database.

Hace unos años atrás, las bases de datos eran el resultado de una compleja programación y de complicados mecanismos de almacenamiento de archivos de texto planos conocidos también como *flat file database*[2].

[2] *Flat file database*. Una *flat file database* o base de datos en archivo plano es referido como un modelo de base de datos simple, donde la información es almacenada en archivos de texto.

Un *flat file database* es un archivo que contiene registros, generalmente un registro por línea. Los campos pueden estar separados con un carácter que se use de separador pudiendo ser un espacio en blanco, un tabulador, comas (CSV[3]), algún otro carácter o combinación de caracteres.

Para ejemplificar las *flat file databases* consideremos un listado de datos con los campos de **`ID`**, **`NOMBRE_COMPLETO`**, **`FECHA_NACIMIENTO`** que en el formato de una tabla se vería de la forma siguiente en la Tabla 1-1:

ID		NOMBRE_COMPLETO	FECHA_NACIMIENTO
1		Miguel Alejandro Palacios Redorta	06/03/1981
2		Efraín Velarde Calvillo	18/04/1986
3		Héctor Alfredo Moreno Herrera	17/01/1988
4		Darío Verón Maldonado	26/07/1979
5		Israel Castro Macías	20/12/1980
6		Jehu Beezye Chiapas Pérez	03/10/1985
7		Leandro Augusto Oldoni Stachelski	18/08/1977
8		Ariel Maximiliano López	05/04/1974
9		Reinaldo José da Silva	24/02/1980
10		Ignacio Martín Scocco	29/05/1985
11		José Luis López Monroy	19/10/1979
12		Sergio Arturo Bernal Hernández	09/02/1970
13		Antonio Sancho Sánchez	14/03/1976
14		Álex Diego Tejado	01/07/1985
15		Pablo Bonells Mendoza	09/09/1985
16		Fernando Espinosa Barrera	09/05/1983
17		José David Toledo Bósquez	18/04/1982
18		Pablo Edson Barrera Acosta	21/06/1987
19		Daniel Óscar Alanís García	02/03/1986
20		Ismael Íñiguez González	23/07/1981
21		Óscar Ariel González Mezzenasco	22/10/1974
22		Orlando Pineda Torres	15/02/1986
23		Marco Antonio Palacios Redorta	06/03/1981
24		Benjamín Jesús Mosco Méndez	09/02/1985
25		Rogelio Rodríguez Martínez	11/03/1976
26		Víctor Odín Patiño Bermúdez	24/08/1983
27		Fernando Morales Esquer	14/09/1985

Tabla 1-1. Listado del equipo de los Pumas Torneo Mexicano apertura 2006.

Mientras que el mismo listado en un formato CSV su forma sería la siguiente, mostrado en el código 1-1:

```
1,Miguel Alejandro Palacios Redorta,06/03/1981
2,Efraín Velarde Calvillo,18/04/1986
3,Héctor Alfredo Moreno Herrera,1988/01/17
4,Darío Verón Maldonado,26/07/1979
5,Israel Castro Macías,20/12/1980
6,Jehu Beezye Chiapas Pérez,03/10/1985
7,Leandro Augusto Oldoni Stachelski,18/08/1977
8,Ariel Maximiliano López,05/04/1974
```

[3] CSV. Comma-separated values. Archivo de texto separado por comas, formato de datos delimitando los campos por el carácter coma (,) y a cada registro separado por un carácter de nueva línea.

```
9,Reinaldo José da Silva,24/02/1980
10,Ignacio Martín Scocco,29/05/1985
11,José Luis López Monroy,19/10/1979
12,Sergio Arturo Bernal Hernández,09/02/1970
13,Antonio Sancho Sánchez,14/03/1976
14,Álex Diego Tejado,01/07/1985
15,Pablo Bonells Mendoza,09/09/1985
16,Fernando Espinosa Barrera,1983/05/09
17,José David Toledo Bósquez,18/04/1982
18,Pablo Edson Barrera Acosta,1987/06/21
19,Daniel Óscar Alanís García,02/03/1986
20,Ismael Íñiguez González,23/07/1981
21,Óscar Ariel González Mezzenasco,22/10/1974
22,Orlando Pineda Torres,1986/02/15
23,Marco Antonio Palacios Redorta,06/03/1981
24,Benjamín Jesús Mosco Méndez,09/02/1985
25,Rogelio Rodríguez Martínez,11/03/1976
26,Víctor Odín Patiño Bermúdez,24/08/1983
27,Fernando Morales Esquer,1985/09/14
```

Código 1-1. Listado del equipo de los pumas Torneo Mexicano Apertura 2006 en un formato flat file database con una conformación CSV.

Aunque también el contenido se puede separar por caracteres distintos como sería un guión para los campos y una barra para los registros como se ve en el Código 1-2:

```
1-Miguel Alejandro Palacios Redorta-06/03/1981|2-Efraín Velarde Calvillo-18/04/1986|3-Héctor Alfredo Moreno Herrera-1988/01/17|4-Darío Verón Maldonado-26/07/1979|5-Israel Castro Macías-20/12/1980|6-Jehu Beezye Chiapas Pérez-03/10/1985|7-Leandro Augusto Oldoni Stachelski-18/08/1977|8-Ariel Maximiliano López-05/04/1974|9-Reinaldo José da Silva-24/02/1980|10-Ignacio Martín Scocco-29/05/1985|11-José Luis López Monroy-19/10/1979|12-Sergio Arturo Bernal Hernández-09/02/1970|13-Antonio Sancho Sánchez-14/03/1976|14-Álex Diego Tejado-01/07/1985|15-Pablo Bonells Mendoza-09/09/1985|16-Fernando Espinosa Barrera-1983/05/09|17-José David Toledo Bósquez-18/04/1982|18-Pablo Edson Barrera Acosta-1987/06/21|19-Daniel Óscar Alanís García-02/03/1986|20-Ismael Íñiguez González-23/07/1981|21-Óscar Ariel González Mezzenasco-22/10/1974|22-Orlando Pineda Torres-1986/02/15|23-Marco Antonio Palacios Redorta-06/03/1981|24-Benjamín Jesús Mosco Méndez-09/02/1985|25-Rogelio Rodríguez Martínez-11/03/1976|26-Víctor Odín Patiño Bermúdez-24/08/1983|27-Fernando Morales Esquer-1985/09/14
```

Código 1-2. Listado del equipo de los pumas Torneo Mexicano Apertura 2006 un flat file database en un formato separado por el carácter guión (-) para separar los campos y el carácter barra (|) para separar los registros.

Tanto el código separado por comas y el separado por guiones representan a un archivo en texto plano con el contenido de información idéntica almacenado en dos diferentes formatos. Esta forma de almacenamiento era una de las primeras utilizadas que tenía diversas limitantes, pero con la popularización de la informática y la aparición de aplicaciones específicas también trajo con ella la disponibilidad de herramientas de gestión de datos que dieron lugar a los denominados Sistemas de Gestión de Bases de Datos.

1.5. ARQUITECTURA PARA LAS BASES DE DATOS.

De esta manera, la gestión de las bases de datos pudo liberarse de las grandes computadoras centrales para distribuirse según los intereses de los usuarios dotando de autonomía en la gestión de información a muchas entidades.

Los DBMS permitieron a todo tipo de usuarios crear y mantener sus bases de datos, dotándolos así de una herramienta que fuera capaz de transformar el nivel lógico y estructural de sus diseños de un conjunto de datos, representaciones gráficas y relaciones entre tablas traducidos al nivel físico correspondiente.

Para dotar a los usuarios de cierta seguridad en el intercambio de datos entre diferentes sistemas, en el diseño de archivos y creación de bases de datos, fue necesario normalizar los esquemas que guiaran la gestión de las mismas.

Un sistema de gestión de bases de datos se encuentra dividido en módulos, cada uno de los cuales controla una parte de la responsabilidad total del sistema. En la mayoría de los casos, el sistema operativo proporciona únicamente los servicios básicos y el DBMS debe partir de esa base y controlar además el manejo correcto de los datos.

El diseño de un sistema de gestión de base de datos debe incluir la interfaz entre el sistema de base de datos y el sistema operativo.

Como se ilustra en la Ilustración 1-1 los componentes funcionales de un sistema de gestión de base de datos, son los siguientes:

Gestor de archivos:	Administra la asignación de espacio en la memoria del disco y de las estructuras de datos usadas para representar la información.
Manejador de base de datos:	Se utiliza como una interfaz entre los datos y los programas de aplicación.
Procesador de consultas:	Traduce las proposiciones en lenguajes de consulta a instrucciones de bajo nivel. Además convierte la solicitud del usuario en una forma más eficiente.
Compilador de DDL:	Convierte las proposiciones DDL en un conjunto de tablas que contienen meta datos, estas se almacenan en el diccionario de datos.
Archivo de datos:	En él se encuentran almacenados físicamente los datos de una organización.
Diccionario de datos:	Contiene la información referente a la estructura de la base de datos.
Índices:	Permiten un rápido acceso a registros que contienen valores específicos.

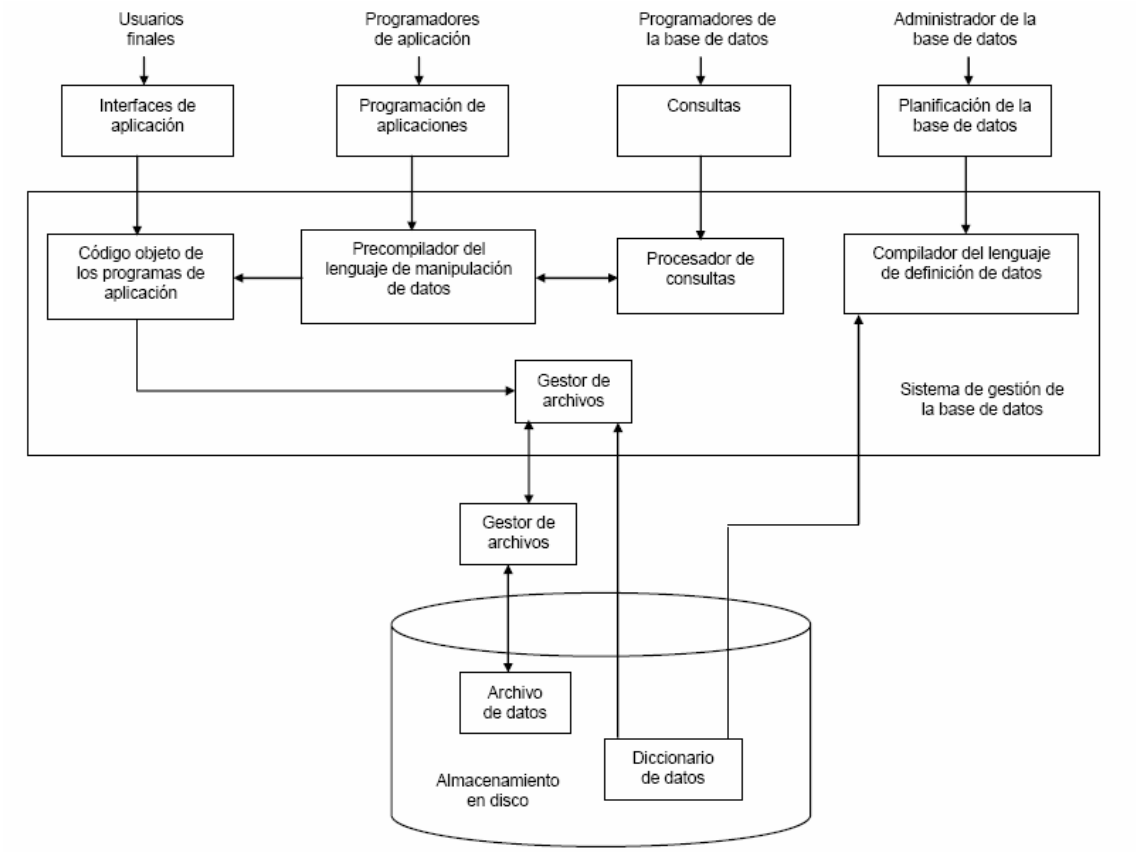


Ilustración 1-1. Integración de los componentes funcionales de un sistema de un DBMS.

Los DBMS deben respetar la arquitectura de tres niveles definidos, para cualquier tipo de base de datos, los cuales fueron asignados por el grupo ANSI/SPARC.

Como se muestra en la Ilustración 1-2 en esta arquitectura muestra que la base de datos se divide en niveles externo, conceptual e interno.

Nivel interno:	Es el nivel más bajo de abstracción y define cómo se almacenan los datos en el soporte físico, así como los métodos de acceso.
Nivel conceptual:	Es el nivel medio de abstracción, se trata de la representación de los datos realizada por la organización que recoge las vistas parciales de los requerimientos de los diferentes usuarios y las aplicaciones posibles, se configura como visión organizativa total e incluye la definición de datos y la relación entre ellos.

Nivel externo: Es el nivel de mayor abstracción, a este nivel le corresponden las diferentes vistas parciales que tienen de la base de datos los diferentes usuarios, en cierto modo, es la parte del modelo conceptual a la que tienen acceso.

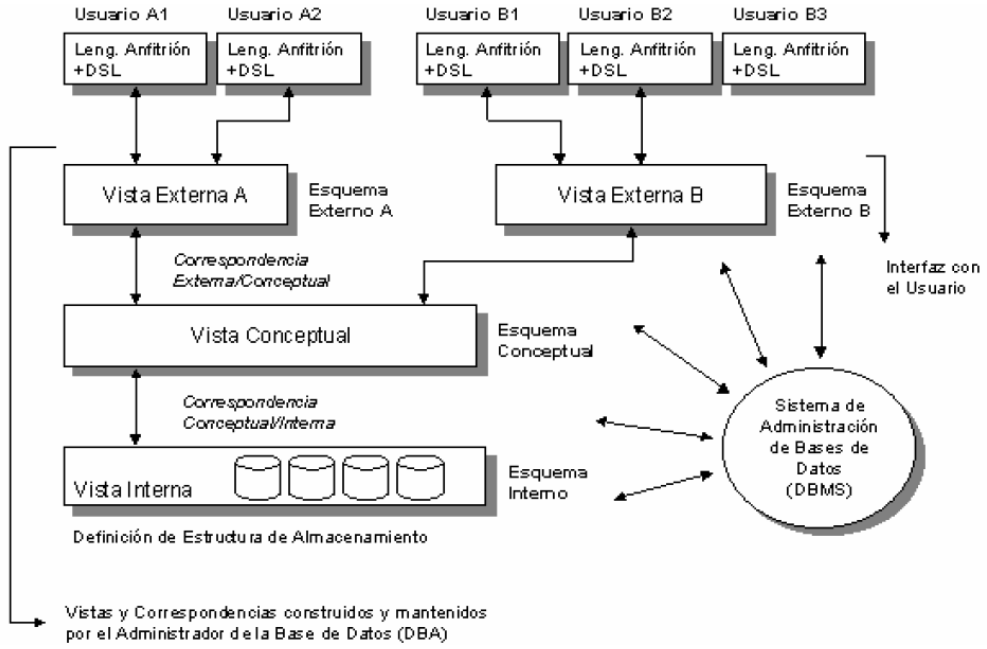


Ilustración 1-2. División de los niveles (interno, conceptual y externo) de un DBMS.

Una *vista externa* es una visión particular de un usuario o un grupo de usuarios de la base de datos.

La *vista conceptual* pretende ser la representación total y abstracta de los datos que componen la base.

Por último, la *vista interna* es de un nivel muy bajo y corresponde al almacenamiento físico de los datos de la base.

Las correspondencias se pueden definir como una asociación de distintas representaciones para un mismo dato.

Un DSL (Data Sub Language) es un sub-lenguaje de datos, es una combinación de dos lenguajes: un Lenguaje de Definición de Datos (DDL) y un Lenguaje de Manipulación de Datos (DML). Este lenguaje representa un nexo entre el sistema de base de datos y algún lenguaje anfitrión, por ejemplo, C, Java y PHP, el DSL provee diferentes herramientas a los lenguajes tradicionales para que se integren al Sistema Gestión de Base de Datos.

El DBMS es el software que maneja todos los accesos a la base de datos, cada solicitud de acceso de un usuario al sistema es interpretada e inspeccionada generando a continuación una respuesta coherente a las necesidades de la pregunta. La interfaz con el usuario es el límite de acceso que tiene un usuario común a la base, todo lo que está bajo este límite es transparente o desconocido para él.

El administrador de bases de datos (DBA) es la persona o grupo de personas encargadas del control general del sistema; sus responsabilidades o funciones incluyen:

Decidir el contenido de la base de datos: Comprende la identificación de entidades de interés para la organización y los datos a registrar de éstas entidades.

Decidir la estructura de almacenamiento y la estrategia de acceso: Esto es decidir como deben representarse los datos en forma interna y hacer la correspondencia entre estos y el modelo definido.

Vincularse con los usuarios: Comprende toda una labor de prestación de servicios que busca garantizar la existencia, en la base, de los datos necesarios.

Definir los controles de autorización y procedimientos de validación: Involucra la definición de restricciones de seguridad y protección para la conservación de la integridad de los Datos.

Definir una estrategia de respaldo y recuperación: Esto se basa en un esquema de seguridad más amplio cuyo objetivo es la operación exitosa del sistema.

Controles de desempeño y cambios de requerimiento: La idea es lograr un desempeño aceptable, según expectativas del sistema, esto mediante mecanismos de control.

Un esquema más genérico que nos permite ilustrar los niveles existentes dentro de la arquitectura de una base de datos es como el que se muestra en la Ilustración 1-3.

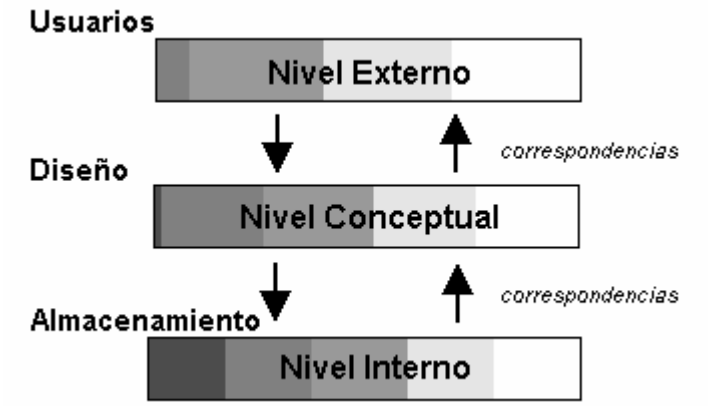


Ilustración 1-3. Esquema de los niveles en la arquitectura de bases de datos.

La arquitectura de tres niveles es útil para explicar el concepto de independencia de datos que podemos definir como la capacidad para modificar el esquema en un nivel del sistema sin tener que modificar el esquema del nivel inmediato superior. Se pueden definir dos tipos de independencia de datos:

Independencia lógica: es la capacidad de modificar el esquema conceptual sin tener que alterar los esquemas externos ni los programas de aplicación. Se puede modificar el esquema conceptual para ampliar la base de datos o reducirla. Por ejemplo, si se reduce la base de datos eliminando una entidad, los esquemas externos que no se refieran a ella no deberían verse afectados.

Independencia física: es la capacidad de modificar el esquema interno sin tener que alterar el esquema conceptual (o los externos).Por ejemplo, puede ser necesario reorganizar ciertos ficheros físicos con el fin de mejorar el rendimiento de las operaciones de consulta o de actualización de datos. Dado que la

independencia física se refiere sólo a la separación entre las aplicaciones y las estructuras físicas de almacenamiento, es más fácil de conseguir que la lógica.

1.6. CARACTERÍSTICAS DE LAS BASES DE DATOS

Redundancia

La redundancia de datos se refiere, a la existencia de información repetida o duplicada en diferentes tablas dentro de una base de datos.

La redundancia conduce a muchos problemas que tienen que ver con la integridad y consistencia de los datos. La redundancia de los datos requiere múltiples procedimientos de entrada y actualización de los mismos.

Dentro de una base de datos relacional la redundancia debe ser mínima y controlada. En ocasiones existirán motivos válidos de negocios o técnicos para mantener varias copias de los mismos datos almacenados.

Consistencia

Frecuentemente los problemas de consistencia de datos se deben a la redundancia de éstos.

Es muy probable que surjan incongruencias al almacenar la misma información en más de un lugar; ya que al modificar, eliminar o agregar un dato, en esas condiciones, debe realizarse en cada una de las instancias del mismo con el riesgo de no realizarlo en su totalidad, generando en este caso datos inconsistentes.

Integridad

La integridad de una base de datos se refiere no sólo a que los datos sean consistentes dentro de la base, sino además, que los valores que posean los datos sean válidos de acuerdo a las dependencias funcionales entre tablas y de acuerdo a las políticas de negocio.

La inconsistencia entre dos entradas que representan al mismo "hecho" es un ejemplo de falta de integridad que, por supuesto, sólo ocurre si existe redundancia en los datos almacenados.

La integridad de la base de datos se puede lograr mediante:

- El mantenimiento de una redundancia mínima y controlada.
- El establecimiento de llaves primarias o índices primarios.
- La validación de las dependencias entre tablas relacionadas.
- La creación de reglas de validación durante la inserción y edición de datos.

Seguridad

Hoy en día se considera a la información de una empresa como uno de los activos más valiosos e importantes, por lo que la seguridad de la misma es muy importante.

La seguridad implica asegurar que los usuarios están autorizados para llevar a cabo lo que tratan de hacer.

La seguridad de una base de datos se refiere principalmente al control de acceso, modificación y definición, tanto de los datos como de la estructura de la base de datos por parte de los diferentes usuarios a la misma.

Algunos sistemas operativos proporcionan algún nivel de seguridad en el control de acceso a usuarios, sin embargo ésta debe radicar principalmente en el DBMS o en la aplicación que maneje la base de datos, sobre todo para evitar la dependencia de entidades externas.

Por otro lado, una base de datos debe cumplir con las siguientes condiciones:

Los datos han de estar almacenadas juntos.

Tanto los usuarios finales como los programas de aplicación no necesitan conocer los detalles de las estructuras de almacenamiento ya que lo importante para estos usuarios es la información contenida.

Los datos son compartidos por diferentes usuarios y programas de aplicación; existe un mecanismo común para inserción, actualización, borrado y consulta de los datos.

Los procedimientos de actualización y recuperación, comunes, y bien determinados, habrán de ser capaces de conservar la integridad, seguridad y confidencialidad del conjunto de datos.

Tanto datos como procedimientos pueden ser transportables conceptualmente a través de diferentes DBMS.

1.7. MODELO DE DATOS.

Para empezar hay que definir que es un modelo.

Modelo: Es una representación de la realidad que contiene las características generales de algo que se va a realizar. En base de datos, esta representación la elaboramos de forma gráfica.

Modelo de datos: Es una colección de herramientas conceptuales para describir los datos, las relaciones que existen entre ellos, semántica asociada a los datos y restricciones de consistencia.

Los modelos de datos se dividen en tres grupos:

- Modelos lógicos basados en objetos.
- Modelos lógicos basados en registros.
- Modelos físicos de datos.

Modelos lógicos basados en objetos

Se usan para describir datos en los niveles conceptual y de visión, es decir, con este modelo representamos los datos de tal forma como nosotros los captamos en el mundo real, tienen una capacidad de estructuración bastante flexible y permiten especificar restricciones de datos explícitamente. Existen diferentes modelos de este tipo, pero el más utilizado por su sencillez y eficiencia es el modelo Entidad-Relación.

Modelo Entidad-Relación. Denominado por sus siglas como E-R; este modelo representa a la realidad mediante entidades que son objetos que existen y que se distinguen de otros por sus características, por ejemplo, un alumno se distingue de otro por sus características particulares como lo es el nombre o el número de control asignado al entrar a una institución educativa, para entender mejor esto, veamos un ejemplo:

Consideremos una empresa que requiere controlar a los vendedores y las ventas que ellos realizan; de este problema determinamos que los objetos o entidades principales a estudiar son el empleado (vendedor) y el artículo (producto), por lo tanto la relación entre ambas entidades la podemos establecer como venta; las características que identifican a cada entidad se muestran en la Ilustración 1-4.

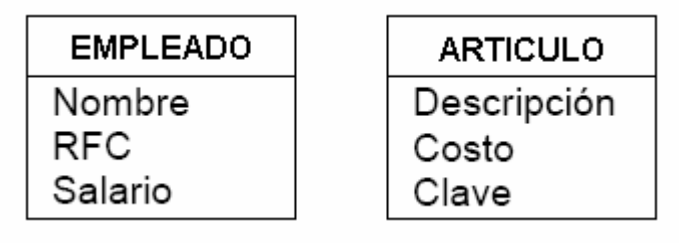


Ilustración 1-4. Ejemplo básico de Entidades en el Modelo Entidad – Relación.

Modelos lógicos basados en registros

Se utilizan para describir datos en los niveles conceptual y físico. Estos modelos utilizan registros e instancias para representar la realidad, así como las relaciones que existen entre estos registros (ligas) o apuntadores. A diferencia de los modelos de datos basados en objetos, se usan para especificar la estructura lógica global de la base de datos y para proporcionar una descripción a nivel más alto de la implementación.

Los cuatro modelos de datos más ampliamente aceptados son:

- Modelo jerárquico.
- Modelo de red.
- Modelo relacional.
- Modelo orientado a objetos

Modelo jerárquico

Es similar al modelo de red en cuanto a las relaciones y datos, ya que estos se representan por medio de registros y sus ligas. La diferencia radica en que están organizados por conjuntos de árboles en lugar de gráficas arbitrarias.

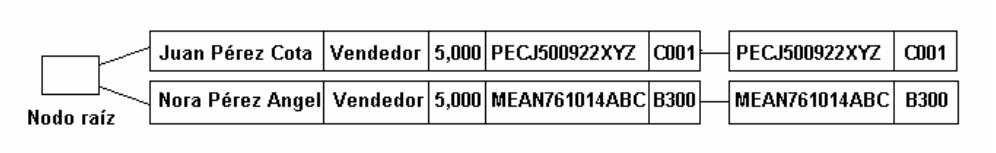


Ilustración 1-5. Representación gráfica del modelo jerárquico.

En este tipo de modelos la organización se establece en forma de árbol, donde la raíz es un nodo ficticio. Así tenemos que, una base de datos jerárquica es una colección de árboles de este tipo.

Modelo de red

Este modelo representa los datos mediante colecciones de registros, sus relaciones se representan por medio de ligas o enlaces, los cuales pueden verse como punteros.

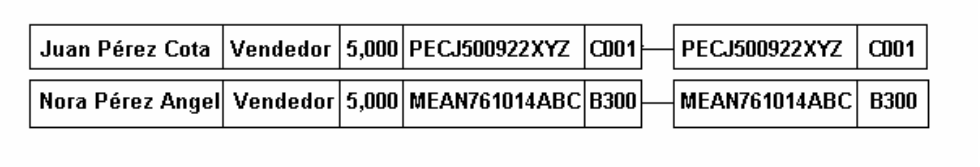


Ilustración 1-6. Representación gráfica del modelo de red.

Una base de datos de red como su nombre lo índica, esta formado por una colección de registros, los cuales están conectados entre sí por medio de enlaces. El registro es similar a una entidad como las empleadas en el modelo entidad-relación.

Un registro es una colección de campos (atributos), cada uno de los cuales contiene solamente almacenado un valor, el enlace es la asociación entre dos registros exclusivamente, así se puede ver como una relación estrictamente binaria.

Modelo relacional

En este modelo se representan los datos y las relaciones entre estos, a través de una colección de tablas, en las cuales los renglones (tuplas[4]) equivalen a cada uno de los registros que contendrá la base de datos y las columnas corresponden a las características (atributos) de cada registro localizado en la tupla, por ejemplo, para las tablas empleado y articulo, se tendría:

[4] Tupla. Es una hilera o fila en una tabla o del resultado de una consulta. Tuplas se refiere al conjunto de más de una tupla.

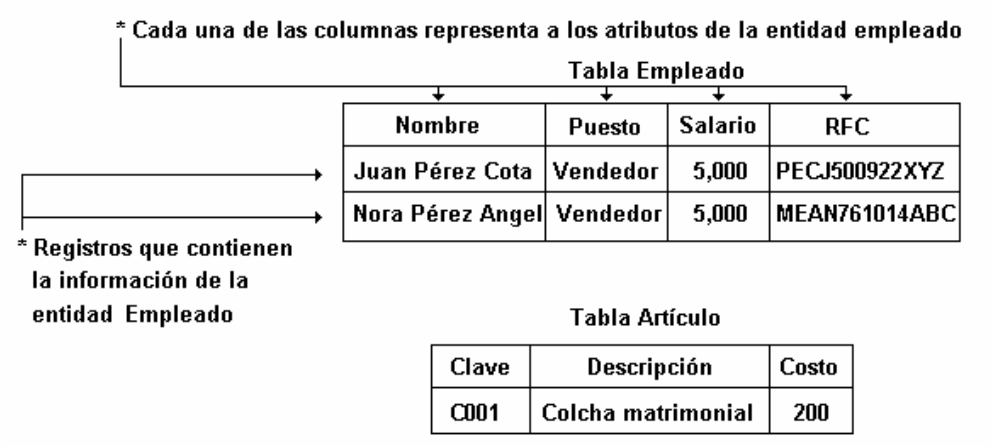


Ilustración 1-7. Representación de atributos y registros en el modelo entidad – relación.

Existen dos formas de representar las relaciones; sin embargo, es necesario definir el concepto de llave primaria, que no es más que un campo que definimos como atributo principal y que de una manera u otra es una forma única para identificar a una entidad.

Para ejemplificar esto diríamos que el RFC de un empleado se distingue de otro por que los RFC – según las reglas del negocio – no pueden ser iguales.

Las formas de representar las relaciones en este modelo son:

Haciendo una tabla que contenga cada una de las llaves primarias de las entidades involucradas en la relación, por ejemplo, tomando en cuenta que la llave primaria del empleado es su RFC, y la llave primaria del artículo es la clave.

La relación resulta de la siguiente forma:

RFC	Clave
PECJ500922XYZ	C001
MEAN761014ABC	B300

Ilustración 1-8. Representación de una relación entre dos tablas usando una tabla transitiva.

Incluyendo en alguna de las tablas de las entidades involucradas, la llave de la otra tabla.

Incrustamos la llave primaria de la tabla Artículo en la tabla Empleado:

Nombre	Puesto	Salario	RFC	Clave
Juan Pérez Cota	Vendedor	5,000	PECJ500922XYZ	C001
Nora Pérez Angel	Vendedor	5,000	MEAN761014ABC	B300

Ilustración 1-9. Forma de relacionar tablas incluyendo un campo extra.

Modelo orientado a objetos

El modelo de bases de datos orientado a objetos es una adaptación a los sistemas de bases de datos. Se basa en el concepto de encapsulamiento de datos y código que opera sobre estos en un objeto. Los objetos estructurados se agrupan en clases.

El propósito de los sistemas de bases de datos es la gestión de grandes cantidades de información. Las primeras bases de datos surgieron del desarrollo de los sistemas de gestión de archivos. Estos sistemas primero evolucionaron en bases de datos de red o en bases de datos jerárquicas y, más tarde, en bases de datos relacionales.

La interfaz entre un objeto y el resto del sistema se define mediante un conjunto de mensajes. Un método, es un trozo de código para implementar cada mensaje. Un método devuelve un valor como respuesta al mensaje.

Modelos físicos de datos

Se usan para describir a los datos en el nivel más bajo, aunque existen muy pocos modelos de este tipo, básicamente capturan aspectos de la implementación de los sistemas de base de datos. Existen dos clasificaciones de este tipo que son:

- Modelo unificador.
- Memoria de elementos.

1.8. ARQUITECTURA CLIENTE / SERVIDOR

La arquitectura de Cliente-Servidor divide el procesamiento entre un cliente y un servidor para optimizar la capacidad de cada sistema. El cliente generalmente es una PC o una estación de trabajo con una interfaz GUI. El servidor de la base de datos es otra PC o una estación de trabajo; comúnmente los clientes se conectan al servidor a través de una red.

La arquitectura Cliente-Servidor permite a los usuarios elegir la configuración de sistema más adecuada para su ambiente de computación, independientemente del sistema operativo, el hardware y el software de aplicación. Esto proporciona una estructura imprescindible para integrar ambientes heterogéneos.

Podemos entender el término Cliente-Servidor como un sistema en el que una máquina Cliente solicita a una segunda máquina llamada Servidor que ejecute una tarea específica, el Cliente suele ser una PC conectada a una red LAN y el Servidor, como un servidor de archivos PC o un servidor de archivos UNIX.

Por lo tanto el verdadero poder de los sistemas Cliente-Servidor radica en la gran variedad de aplicaciones cliente y software de desarrollo. En este ambiente, el cliente se encarga de la interfase con el usuario (pantallas, reportes, etc.) y la presentación lógica de los datos, mientras que el servidor se encarga de ejecutar las consultas SQL a la base de datos, de la integridad de la base de datos, así como del almacenamiento físico de los mismos.

El modelo Cliente-Servidor se define como la tecnología que proporciona al usuario final el acceso transparente a las aplicaciones, datos, servicios de cómputo o a cualquier otro recurso dentro del grupo de trabajo y/o a través de la empresa en diferentes plataformas.

Esta arquitectura se conforma de dos tipos de componentes que se comunican a través de una red: La red proporciona la conexión entre los clientes y los servidores.

- Back-end: Servidor
- Front-end: Cliente

El Servidor procesa las peticiones que hacen los clientes, y cuando es posible regresa el resultado. Mantiene la integridad lógica y el acceso de los datos.

El Cliente envía peticiones al servidor y manipula las respuestas. Este puede: desplegar y manejar el ambiente de trabajo de la aplicación y la interfaz de usuario, llevar a cabo la validación de datos, desplegar reportes y representar datos gráficamente.

La secuencia de eventos cuando un usuario accede al servidor de bases de datos se puede generalizar en los siguientes pasos:

- El usuario crea su consulta sobre los datos.
- El cliente formatea la consulta en lenguaje SQL y la envía a través de la red.
- El servidor de base de datos verifica los permisos sobre los datos a consultar.
- El servidor de base de datos procesa la consulta y regresa los resultados.
- El cliente recibe la respuesta y la presenta al usuario.
- El usuario visualiza y manipula los datos y reinicia el proceso.

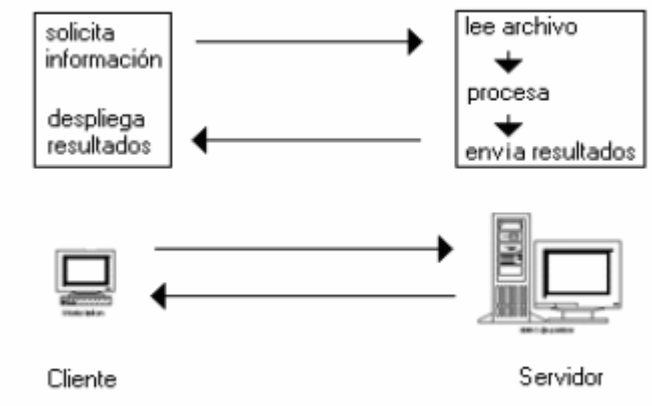


Ilustración 1-10. Diagrama de la arquitectura Cliente – Servidor.

Evolución del modelo relacional

La siguiente tabla hace una síntesis de la evolución del Modelo Relacional, desde su surgimiento a fines de la década de los sesenta hasta la actualidad.

Años	Sucesos
1968	Surge el Modelo Relacional (Codd).
1970	Desarrollos teóricos, álgebra relacional (Codd, 1972).
1973	Prototipos (Ingres, Sistema R, etc.).
1979	Oracle.
1981	SQL.
1982	Sybase, Informix.
1984	SQL /ANS.
1986	SQL ISO.
1990	Modelo Relacional versión 2 (RM/V2) Codd.
1992	SQL2 estándar.
1994	SQL3 Aún no estandarizado BDOO.

Tabla 1-2. Evolución del modelo relacional.

Objetivos del modelo relacional

El trabajo publicado por Codd en ACM (1970) presentaba un nuevo modelo de datos que perseguía una serie de objetivos, que se resumen en las siguientes líneas:

Independencia física. El modo en el que se almacenan los datos no influye en su manipulación lógica y por tanto, los usuarios que acceden a esos datos no tienen que modificar sus programas por cambios en el almacenamiento físico.

Independencia lógica. El añadir, eliminar o modificar objetos de la base de datos no repercute en los programas y/o usuarios que están accediendo a subconjuntos parciales de los mismos (vistas).

Flexibilidad. En el sentido de poder presentar a cada usuario los datos de la forma en que éste prefiera.

Uniformidad. Las estructuras lógicas de los datos presentan un aspecto uniforme, lo que facilita la concepción y manipulación de la base de datos por parte de los usuarios.

Sencillez. Las características anteriores, así como unos lenguajes de usuario muy sencillos, producen como resultado que el modelo de datos relacional sea fácil de comprender y de utilizar por parte del usuario final.

El modelo Relacional se divide en 3 partes: estructura de los datos, integridad de los datos, y manipulación de los datos.

Estructura del Modelo Relacional

La relación es el elemento básico del modelo relacional y se representa por una tabla. Informalmente, los términos y sus equivalentes son:

Relación	Tabla
Tupla	Fila
Atributo	Columna
Número de tuplas	Cardinalidad
Número de atributos	Grado
Dominio	Colección de valores

Clave primaria	Identificador único para la tabla

Tabla 1-3. Equivalencia de términos en modelo relacional

Es importante señalar que la tabla es plana en el sentido de que el cruce de una fila y una columna solo puede dar un valor, es decir, no se admiten atributos multivaluados.

Dominio y Atributo

Dominio: Un Dominio es un conjunto finito de valores homogéneos y atómicos caracterizados por un nombre. Homogéneo significa que los valores son todos del mismo tipo y atómicos significa que son indivisibles, es decir, si se descomponen se perdería la semántica del dominio, por ejemplo:

Dominio de Nacionalidades: Chilena, Francesa, Norteamericana

Todo dominio tiene un nombre y un tipo de datos, en el ejemplo anterior, el tipo de datos es un conjunto de caracteres de longitud máxima de 14. Se pueden asociar unidades de medida, como metros y kilos u otras restricciones. La importancia de los dominios es que restringen las comparaciones, es decir, sólo se pueden comparar atributos definidos sobre el mismo dominio.

Atributo: Un atributo es aquel que participa en la descripción de las entidades y que como tal constituye una pieza específica de información para un determinado dominio.

En el caso de que sean varios los atributos de una misma tabla, definidos por el mismo dominio, habrá que darles nombres distintos, ya que una tabla no puede tener dos atributos con el mismo nombre, sin embargo, existen los dominios compuestos los cuales están conformados por otros dominios, además de tener un nombre y permitir aplicar restricciones, por ejemplo:

Un usuario podría manejar además de los tres dominios Día, Mes y Año un dominio compuesto llamado Fecha que sería la combinación de los tres y al que podríamos aplicar restricciones de integridad a fin de que no aparecieran valores no válidos para la fecha.

Relación

Matemáticamente una relación definida sobre los n dominios D_1, D_2,…, D_n, es un subconjunto del producto cartesiano de estos dominios donde cada elemento de la relación (tupla), es una serie de n valores ordenados.

Claves

El término clave en una relación es un conjunto no vacío de atributos que identifican unívoca y mínimamente cada tupla. Toda relación siempre tendrá una clave candidata, estas claves pueden ser clasificadas en tres principales grupos:

Clave primaria: Es aquella clave que permite identificar las tuplas de la relación de forma única.

Clave alternativas: Son aquellas claves que no han sido escogidas como claves primarias, pero que también podrían identificar de manera única a una tupla.

Clave foránea: Es un conjunto no vacío de atributos cuyos valores han de coincidir con los valores de la clave primaria de una relación.

Restricciones

Las restricciones son estructuras no permitidas. Hay dos tipos: inherentes y del usuario. Las inherentes son aquellas propias al modelo, por ejemplo, no tener tuplas repetidas y las del usuario son aquellas que validan las instancias de la relaciones.

Justificaciones

En el modelo relacional se manejan algunas justificaciones, tal es el caso de las siguientes:

Las entidades del mundo real son identificables y distinguibles.

Si una entidad es lo bastante importante en el mundo real como para requerir una representación explícita en la base de datos, tal entidad deberá ser susceptible de identificarla sin ambigüedad, pues de lo contrario seria imposible hablar de ella, es por ello que se dice que en una base de datos, nunca se registrara información acerca de algo que no se pueda identificar.

Reglas de Codd

En la década de los 80's comenzaron a aparecer numerosos DBMS que se anunciaban como relacionales; sin embargo estos sistemas carecían de muchas características que se consideraban importantes en un sistema relacional, perdiendo muchas ventajas del modelo relacional. Como ejemplo externo de esto "sistemas relacionales" eran simplemente sistemas que utilizaban tablas para almacenar la información, no disponiendo de elementos como claves primarias, etc. En 1984 Codd publicó 12 reglas que un verdadero sistema relacional debería cumplir.

Cualquier BDMS que proclame ser relacional, deberá manejar, completamente, las bases de datos por medio de sus capacidades relacionales.

Regla de información. Toda la información dentro de una base de datos relacional se representa de manera explícita a nivel lógico y exactamente de una sola manera, como valores en una tabla.

Regla del acceso garantizado. Se garantiza que todos y cada uno de los datos (valor atómico) en una base de datos relacional pueden ser leídos recurriendo a una combinación del nombre de la tabla, valor de la llave primaria y nombre de la columna.

El manejo sistemático de los valores nulos. En un DBMS totalmente relacional se soportan los valores nulos (que son distintos de una cadena de caracteres vacía o de una cadena con caracteres en blanco o de cero o cualquier otro número), para representar información faltante o no aplicable de una forma consistente, independientemente del tipo de dato.

Catálogo dinámico en línea basado en un modelo relacional. La descripción de la base de datos se representa en el nivel lógico de la misma forma que los datos ordinarios, de tal suerte que los usuarios autorizados puedan aplicar el mismo lenguaje relacional para consultarla, que aquél que emplean para con sus datos habituales.

Regla del sub-lenguaje de dato completo. Se debe contar con un sub-lenguaje que contemple la definición de datos, la definición de vistas, la manipulación de datos, las restricciones de integridad, la autorización, el inicio y fin de una transacción.

Regla de actualización de vistas. Todas las vistas que teóricamente sean actualizables deberán ser actualizadas por medio del sistema.

Inserción, actualización y eliminación de alto nivel. La posibilidad de manejar una relación base o una relación derivada como un sólo operador se aplica a la lectura, inserción, modificación y eliminación de datos.

Independencia física de los datos. Los programas de aplicación y la actividad en terminales no deberán ser afectados por cambios en el almacenamiento físico de los datos o en el método de acceso.

Independencia lógica de los datos. Los programas de aplicación y la actividad en terminales no deberán ser afectados por cambios de cualquier tipo que preserven la información y que teóricamente permitan la no afectación en las tablas base.

Independencia de la integridad. Las restricciones de integridad de una base de datos deberán poder definirse en el mismo sub-lenguaje de datos relacional y deberán almacenarse en el catálogo, no en los programas de aplicación.

Independencia de la distribución. Un DBMS relacional tiene independencia de distribución.

Regla de la no subversión. Si un sistema relacional tiene un lenguaje de bajo nivel (un sólo registro cada vez), ese bajo nivel no puede ser utilizado para suprimir las reglas de integridad y las restricciones expresadas en el lenguaje relacional de nivel superior (múltiples registros a la vez).

Bases de datos relacionales

El término base de datos fue acuñado por primera vez en 1963, en un simposio celebrado en California. Se dice que una base de datos no es más que un conjunto de información relacionada, que esta agrupada o estructurada.

El archivo por sí mismo, no constituye una base de datos, sino más bien la forma en que está organizada ya esta información es la que da origen a la base de datos. Las bases de datos manuales, pueden ser difíciles de gestionar y modificar, por ejemplo, en una guía de teléfonos no es posible encontrar el número de un individuo si no sabemos su apellido, aunque conozcamos su domicilio.

Del mismo modo, en un archivo de pacientes en el que la información esté desordenada por el nombre de los mismos, será una tarea bastante engorrosa encontrar todos los pacientes que viven en una zona determinada, esta clase de problemas se pueden resolver creando una base de datos informatizada.

Desde el punto de vista informático, una base de datos es un sistema formado por un conjunto de datos almacenados en discos que permiten el acceso directo a ellos y un conjunto de programas que manipulan ese conjunto de datos.

Desde el punto de vista más formal, podríamos definir una base de datos como un conjunto de datos estructurados, fiables y homogéneos, organizados independientemente en una máquina, accesibles a tiempo real, compartidas por usuarios concurrentes que necesitan información diferente y no predecible en el tiempo.

La idea general es que estamos tratando con una colección de datos que cumplen las siguientes propiedades:

- Son independientes de las aplicaciones y del soporte de almacenamiento.
- Presentan la menor redundancia posible.
- Son compartidos por varios usuarios y/o aplicaciones.

Los sistemas relacionales son importantes porque ofrecen muchos tipos de procesos de datos, como: simplicidad y generalidad, facilidad de uso para el usuario final, períodos cortos de aprendizaje y las consultas de información se especifican de forma sencilla. Las tablas son un medio para representar la información de una forma más compacta y al mismo tiempo acceder a información contenida en dos o más tablas.

Las bases de datos relacionales están constituidas por una o más tablas que contienen la información ordenada de una forma organizada y que además cumplen con las siguientes leyes básicas:

- Generalmente, contendrán muchas tablas.
- Una tabla sólo contiene un número fijo de campos.
- El nombre de los campos de una tabla es distinto.
- Cada registro de la tabla es único.
- El orden de los registros y de los campos no está determinado.
- Para cada campo existe un conjunto de valores posible.

Entonces, una base de datos relacional es una base de datos en donde todos los datos visibles al usuario están organizados estrictamente como tablas de valores, y en donde todas las operaciones de la base de datos operan sobre estas tablas. Estas bases de datos son percibidas por lo usuarios como una colección de relaciones normalizadas de diversos grados que varían con el tiempo.

Diseño de las bases de datos relacionales

El primer paso para crear una base de datos, es planificar el tipo de información que se quiere almacenar en la misma, teniendo en cuenta dos aspectos: la información disponible y la información que necesitamos. La planificación de la estructura de la base de datos, en particular de las tablas, es vital para la gestión efectiva de la misma.

El diseño de la estructura de una tabla consiste en una descripción de cada uno de los campos que componen el registro y los valores o datos que contendrá cada uno de esos campos.

Los campos son los distintos tipos de datos que componen la tabla, por ejemplo: nombre, apellido, domicilio. La definición que un campo requiere principalmente es el nombre del campo, el tipo de campo y la longitud del mismo.

Los registros constituyen la información que va contenida en los campos de la tabla, por ejemplo, en la base de datos para un hospital sería el nombre del paciente, el apellido del paciente y la dirección del mismo.

Generalmente los diferentes tipos de campos que su pueden almacenar son los siguientes: Texto (caracteres), Numérico (números), Fecha / Hora, Lógico (informaciones lógicas si/no, verdadero/falso, etc.), imágenes.

En resumen, el principal aspecto a tener en cuenta durante el diseño de una tabla es determinar claramente los campos necesarios, definirlos en forma adecuada con un nombre especificando su tipo y su longitud.

El lenguaje de consulta relacional

El SQL es una herramienta para organizar, gestionar y recuperar datos almacenados en una base de datos informática. El nombre "SQL" es una abreviatura de Structured Query Language (Lenguaje de consultas estructurado). Como su propio nombre indica, SQL es un lenguaje informático que se puede utilizar para interaccionar con una base de datos y más concretamente con un tipo especifico llamado base de datos relacional.

El álgebra relacional consiste en una colección de operaciones sobre relaciones donde cada operación toma una o más relaciones como su operando y produce otra relación como su resultado. Dado que el resultado de una operación del álgebra relacional es una relación, ésta a su vez puede ser sujeta de posteriores operaciones algebraicas.

El álgebra relacional se basa en la teoría de conjuntos, relaciones y en el álgebra de conjuntos

Adicionalmente al conjunto básico de operadores como: unión, diferencia, producto cartesiano e intersección; incorpora operadores específicos de base de datos tales como proyección, selección y join.

Unión

Teniendo dos conjuntos, la unión nos representaría los elementos que se encuentran en ambos conjuntos, por ejemplo, la siguiente operación nos da una idea de lo antes mencionado:

{1,4,5,10} U {1,4,3,9} = {1,3,4,5,9,10}

En términos de tablas, hay que considerar que la unión sea compatible, es decir, el número de atributos debe ser el mismo y del mismo tipo de datos.

CLAVE	RFC	NOMBRE	DEPTO	SUELDO
125	LOPA650303	PEDRO LOPEZ	1	$ 250.00
236	PERM640506	MARIA PEREZ	1	$ 350.00
584	GOMA700808	ALBERTO GOMEZ	2	$ 350.00

CLAVE	RFC	NOMBRE	DEPTO	SUELDO
698	ALLO601212	OSCAR ALVAREZ	3	$ 300.00
741	SUOP701231	PABLO SUAREZ	2	$ 500.00
254	TEMM680409	MONICA TELLEZ	3	$ 350.00

CLAVE	RFC	NOMBRE	DEPTO	SUELDO
125	LOPA650303	PEDRO LOPEZ	1	$ 250.00
236	PERM640506	MARIA PEREZ	1	$ 350.00
584	GOMA700808	ALBERTO GOMEZ	2	$ 350.00
698	ALLO601212	OSCAR ALVAREZ	3	$ 300.00
741	SUOP701231	PABLO SUAREZ	2	$ 500.00
254	TEMM680409	MONICA TELLEZ	3	$ 350.00

Ilustración 1-11. Unión de dos relaciones.

Diferencia

Teniendo dos conjuntos, la diferencia nos representaría los elementos que están en uno de los conjuntos, pero que no están en el otro, por ejemplo, veamos la siguiente operación:

{1,4,5,10} - {1,4,3,9} = {5,10}

Producto Cartesiano

El producto cartesiano es el producto cruz entre 2 tablas, es decir, el resultado es la unión de cada renglón de una tabla con cada renglón de la otra tabla. El producto es una yuxtaposición donde los elementos son combinados o concatenados, por ejemplo:

{1,4} X {1,4} = {1,4,4,16}

X	A
X	B

X

1
2
3

=

X	A	1
X	B	1
X	A	2
X	B	2
X	A	3
X	B	3

Ilustración 1-12. Producto cartesiano de dos relaciones.

Intersección

Teniendo dos conjuntos, la intersección nos representaría los elementos que están en uno de los conjuntos y que además existen en el otro, por ejemplo, la siguiente operación:

{1,4,5,10} ∩ {1,4,3,9} = {1,4}

Proyección

La proyección selecciona y genera un subconjunto con los atributos indicados de una tabla. También es conocida como operación vertical.

CLAVE	RFC	NOMBRE	DEPTO	SUELDO
125	LOPA650303	PEDRO LOPEZ	1	$ 250.00
236	PERM640506	MARIA PEREZ	1	$ 350.00
584	GOMA700808	ALBERTO GOMEZ	2	$ 350.00
698	ALLO601212	OSCAR ALVAREZ	3	$ 300.00
741	SUOP701231	PABLO SUAREZ	2	$ 500.00
254	TEMM680409	MONICA TELLEZ	3	$ 350.00

CLAVE	RFC	NOMBRE	DEPTO	SUELDO
125	LOPA650303	PEDRO LOPEZ	1	$ 250.00
236	PERM640506	MARIA PEREZ	1	$ 350.00
584	GOMA700808	ALBERTO GOMEZ	2	$ 350.00
698	ALLO60.212	OSCAR ALVAREZ	3	$ 300.00
741	SUOP701231	PABLO SUAREZ	2	$ 500.00
254	TEMM680409	MONICA TELLEZ	3	$ 350.00

Ilustración 1-13. Proyección de una relación.

Selección

La selección toma y genera un subconjunto con los renglones indicados de una tabla. También es conocida como operación horizontal.

CLAVE	RFC	NOMBRE	DEPTO	SUELDO
125	LOPA650303	PEDRO LOPEZ	1	$ 250.00
236	PERM640506	MARIA PEREZ	1	$ 350.00
584	GOMA700808	ALBERTO GOMEZ	2	$ 350.00
698	ALLO601212	OSCAR ALVAREZ	3	$ 300.00
741	SUOP701231	PABLO SUAREZ	2	$ 500.00
254	TEMM680409	MONICA TELLEZ	3	$ 350.00

CLAVE	RFC	NOMBRE	DEPTO	SUELDO
125	LOPA650303	PEDRO LOPEZ	1	$ 250.00
236	PERM640506	MARIA PEREZ	1	$ 350.00
584	GOMA700808	ALBERTO GOMEZ	2	$ 350.00

Ilustración 1-14. Selección de una relación.

Join

La operación join es en esencia un producto cartesiano, donde se seleccionan las columnas que satisfagan las condiciones indicadas. Es la operación más común en las bases de datos relacionales.

CLAVE	NOMBRE	DEPTO
125	JUAN MENDEZ	1
365	MARIA NAJERA	2
698	JESUS MORALES	1

DEPTO	NOMBRE DEPTO
1	GERENCIA
2	CONTABILIDAD

CLAVE	NOMBRE	DEPTO	NOMBRE DEPTO
125	JUAN MENDEZ	1	GERENCIA
365	MARIA NAJERA	2	CONTABILIDAD
698	JESUS MORALES	1	GERENCIA

Ilustración 1-15. Join de una relación.

1.10. LA INDEPENDENCIA DE DATOS Y LA INTEGRIDAD REFERENCIAL

La independencia de datos

Una de las principales ventajas que provee una base de datos es la independencia entre los datos y los tratamientos que se hacen de ellos ya que en los sistemas orientados a procesos los datos eran sumamente dependientes de los programas.

Ocurre que históricamente la tasa de variación de los procesos es mayor que la de los datos que maneja, más aún cualquier actualización de los datos que maneja un proceso determina que éste sea necesariamente actualizado, lo que no siempre ocurre en el sentido contrario.

Lo anterior es asimilable a los cambios que sufren las organizaciones, ya que generalmente existen cambios visibles y otros no tanto, por ejemplo, un banco que cambia su imagen corporativa, podría implementar o eliminar funciones de atención al público así como medidas para la reducción de personal.

Los cambios de fondo no son habituales en las organizaciones, ya que ello tiene que ver con cambios en la misión de la organización y en los objetivos. Si un banco cambia de imagen corporativa (logo, eslogan, mercado), estos cambios son de forma y tienen que ver básicamente con variaciones en los procesos y eventualmente con algunos datos; si existe un cambio muy severo, entonces la situación se

torna más compleja, la misión organizacional cambia así como el objetivo, el cambio es más radical afectando con ello a cada uno de los elementos que componen la organización.

El concepto de base de datos rescata aquella dependencia que tienen los procesos de los datos y la radicaliza priorizando la independencia de estos últimos, determinando mecanismos de definición y de descripción que no requieren de procesos. Con lo anterior se crea un sustrato de datos que permiten modelar toda la organización y que además establecen a los procesos que los utilizan.

Como tal, la independencia de los datos se refiere a la protección contra los programas de aplicación que puedan originar modificaciones cuando se altera la organización física o lógica de la base de datos. Existen 2 niveles de independencia de datos:

Independencia física de datos: Es la capacidad de modificar el esquema físico sin provocar que se vuelvan a escribir los programas de aplicación.

Independencia lógica de datos: Capacidad de modificar el esquema conceptual sin provocar que se vuelvan a escribir los programas de aplicación.

La integridad referencial

El término de integridad referencial se enmarca en la segunda regla de integridad y se aplica a las claves foráneas:

"Si en una relación hay alguna clave foránea, sus valores deben coincidir con valores de la clave primaria a la que hace referencia, o bien, deben ser completamente nulos".

Lo que en realidad trata de decir el texto anterior es que las claves foráneas no pueden dejar de tener correspondencia con la clave primaria de la tabla externa; las tuplas que contienen claves foráneas que no tienen una clave candidata, se denominan entidades huérfanas. Existen además otros tipos de claves:

Clave primaria: Es aquel atributo que identifica de manera única a un registro. Esto es, no debe haber dos tuplas que tengan el mismo valor, por lo tanto, con sólo conocer el valor de la clave primaria para una determinada tupla será suficiente para identificarlo de manera única.

Clave candidata: Es el atributo o conjunto de atributos que podrían servir como llaves primarias.

Clave secundaria: Son aquellas claves candidatas que no se eligieron como llave primaria, es decir, tienen todas las características para ser claves primarias, pero que por alguna razón no fueron tomadas como tal debido quizás a que hubo otra que cumplía mejor con ese objetivo.

Clave foránea: Es una clave primaria en otra relación, estas representan las asociaciones entre las diferentes entidades, es decir, son claves que están siendo compartidas por dos tablas para formar una relación entre ellas.

1.11. MODELO ENTIDAD - RELACIÓN

El modelo Entidad - Relación, es una técnica de diseño de bases de datos gráfica, que incorpora información relativa a los datos y la relación existente entre ellos, para poder así plasmar una visión del mundo real sobre un soporte informático y que se caracteriza fundamentalmente por:

- Sólo reflejar la existencia de los datos sin expresar lo que se hace con ellos.
- La independencia de la base de datos y de los sistemas operativos.

La inclusión de todos los datos sin considerar las aplicaciones que se tendrán.

Entidades

Se puede definir como entidad a cualquier objeto, real o abstracto, que existe en un contexto determinado o que puede llegar a existir y del cual deseamos guardar información, por ejemplo, un profesor, un alumno o bien una materia. Las entidades las podemos clasificar en:

Regulares: Son aquellas entidades que existen por sí mismas, es decir, la existencia de un ejemplar de la entidad no depende de la existencia de otros ejemplares en otra entidad, por ejemplo, la entidad "**PROFESOR**".

Ilustración 1-16. Representación gráfica de una entidad regular.

Débiles: Son aquellas entidades en las que su existencia depende de la existencia de ejemplares en otras entidades, por ejemplo, la existencia de la entidad "**PROFESOR**" depende de la existencia de la entidad "**ESCUELA**".

Ilustración 1-17. Representación gráfica de una entidad débil.

Atributos

Las entidades se componen de atributos que son cada una de las propiedades o características que tienen las entidades. Cada ejemplar de una misma entidad posee los mismos atributos, tanto en nombre como en número, diferenciándose cada uno de los ejemplares por los valores que toman dichos atributos. Si consideramos la entidad "**PROFESOR**" y definimos los atributos **NOMBRE**, **CURSOS**, **TELÉFONOS** y **EDAD**, podríamos obtener los ejemplares mostrados en la Ilustración 1-18.

PROFESOR			
NOMBRE	CURSOS	TELÉFONOS	EDAD
Luís García	Algebra	555-55-55	80.500
Juan Antonio Álvarez	Diseño	666-66-66	92.479
Marta López	Base de Datos	777-77-77	85.396

Ilustración 1-18. Registros basados en la sentencia de los atributos.

Existen cuatro tipos de atributos:

Obligatorios: Aquellos que forzosamente deben tomar un valor.

Opcional: Aquellos atributos que pueden o no tener valores.

Monoevaluado: Aquel atributo que sólo puede tener un único valor.

Multievaluado: Aquellos atributos que pueden tener varios valores.

Dentro del diagrama, la entidad "**PROFESOR**" y sus atributos quedarían de la siguiente forma:

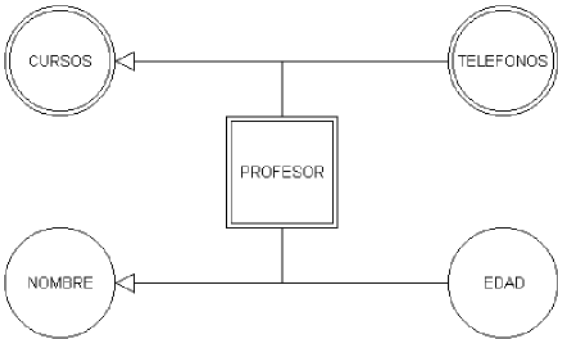


Ilustración 1-19. Diagrama atributos de una entidad.

Existen atributos, llamados derivados, cuyo valor se obtiene a partir de los valores de otros atributos, por ejemplo, supongamos que la entidad "**PROFESOR**" tiene los atributos "**NOMBRE**", "**FECHA DE NACIMIENTO**" *y* "**EDAD**" *;* el atributo "**EDAD**" es un atributo derivado por que se calcula a partir del valor del atributo "**FECHA DE NACIMIENTO**". Su representación gráfica es la siguiente:

Ilustración 1-20. Representación gráfica de un atributo derivado.

En determinadas ocasiones es necesaria la descomposición de un atributo para definirlos en más de un dominio, podría ser el caso del atributo "**TELEFONO**" que toma valores del dominio "**PREFIJO**" y del dominio "**NUMERO**". Estos atributos se representan de la siguiente forma:

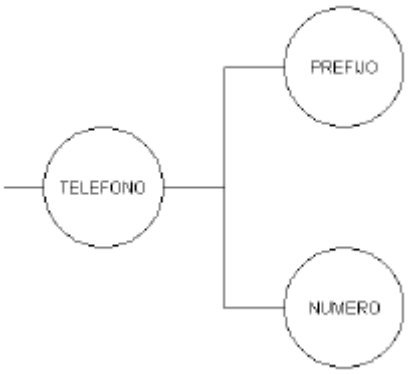


Ilustración 1-21. Descomposición de un atributo.

Dominio

Se define dominio como un conjunto de valores que puede tomar un determinado atributo dentro de una entidad. Por ejemplo:

Atributo	Dominio
Fecha de Alta	Calendario Gregoriano
Teléfono	Conjunto de números de teléfonos
Cobro de Incentivos	SI / NO
Edad	16 - 65

Tabla 1-4. Ejemplo del dominio de los atributos.

De forma casi inherente al término dominio aparece el concepto restricción para un atributo. Cada atributo puede adoptar una serie de valores de un dominio restringiendo determinados valores. El atributo "**EDAD**" toma sus valores del dominio N (números naturales) pero se puede poner como restricción aquellos que estén en el intervalo **(0-120)**, pero dentro de la entidad "**PROFESOR**" se podría restringir aun más el intervalo, puesto que la edad mínima para trabajar es de 16 años y la máxima de 65, por lo tanto el intervalo sería **(16-65)**.

Claves

El modelo Entidad - Relación exige que cada entidad tenga un identificador, se trata de un atributo o conjunto de atributos que identifican de forma única a cada uno de los ejemplares de la entidad.

Un ejemplo de identificador es el atributo "**RFC**", que en la entidad "**PROFESOR**", identifica de forma única a cada uno de los profesores. Estos identificadores reciben el nombre de Clave Primaria o Primary Key (PK). Como ya se había mencionado antes, puede ser que existan más identificadores, a estos atributos se les conoce como Identificadores Candidatos (IC).

Los atributos identificadores de una entidad se representan en los diagramas de la siguiente forma:

Ilustración 1-22. Atributos identificadores de una entidad

Interrelaciones

Se entiende por interrelación a la asociación, vinculación o correspondencia entre entidades. Por ejemplo, entre la entidad "**PROFESOR**" y la entidad "**CURSO**" podemos establecer la relación "**IMPARTE**" por que el profesor imparte cursos.

Al igual que las entidades, las interrelaciones se pueden clasificar en regulares y débiles, esto de acuerdo al tipo de entidad que estén asociando, entidades regulares o entidades débiles, con otra de cualquier tipo. Las interrelaciones débiles se subdividen en dos grupos:

En existencia: Cuando los ejemplares de la entidad débil no pueden existir si desaparece el ejemplar de la entidad regular del cual dependen.

En identificación: Cuando además de ser una relación en existencia, los ejemplares de la entidad débil no se pueden identificar por sí mismos y exigen añadir el identificador principal de la entidad regular del cual dependen para ser identificados.

Las interrelaciones, dentro de los diagramas, se representan de la siguiente forma:

Regulares

Ilustración 1-23. Representación de una interrelación regular

Débiles

Ilustración 1-24. Representación de una interrelación débil

Interrelación regular

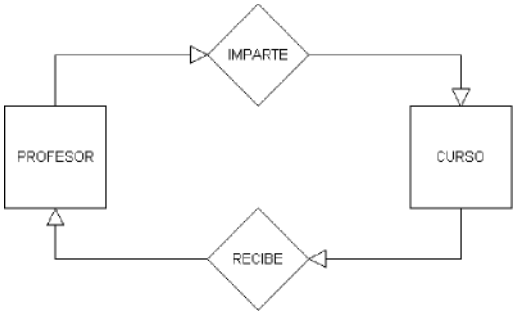


Ilustración 1-25. Ejemplo de una interrelación regular

Interrelación en identidad

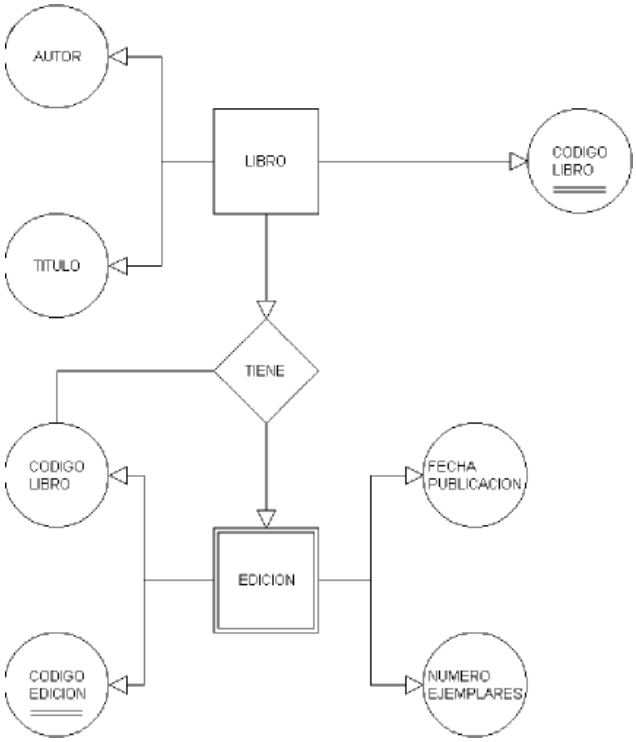


Ilustración 1-26. Interrelación de entidades

Interrelación en existencia

Ilustración 1-27. Interrelación de elementos en cascada

En cada interrelación se debe establecer el número máximo y mínimo de ejemplares de un tipo de entidad que pueden estar asociadas, mediante una determinada relación con un ejemplar de la otra entidad. Este valor máximo y mínimo se conoce como *cardinalidad* y según corresponda, se representa de la siguiente forma: **(0,N), (N,0), (1,N), (N,1), (0,1), (1,0)** ó **(N,N)**. La cardinalidad se representa de la siguiente forma:

Ilustración 1-28. Cardinalidad

En el diagrama anterior la cardinalidad "**CLIENTE**" – "**PEDIDO**" es **1:1** por que al formularnos la pregunta ¿cuántos clientes se pueden relacionar con un pedido? la respuesta es, uno como mínimo y uno como máximo, ya que un pedido es realizado por un único cliente y no cabe la posibilidad que el mismo pedido esté formulado por dos clientes distintos. La cardinalidad "**PEDIDO**" – "**CLIENTE**" es **1:N** por que al formularnos la pregunta ¿cuántos pedidos se pueden relacionar con un cliente? la respuesta es, como mínimo un pedido pertenece a un cliente, pero varios pedidos pueden estar relacionados con el mismo cliente.

Existen ocasiones concretas en que las relaciones tienen atributos, es el caso del diagrama siguiente en donde los alumnos reciben cursos, y la interrelación posee los atributos de fecha de comienzo, fecha de finalización y calificación.

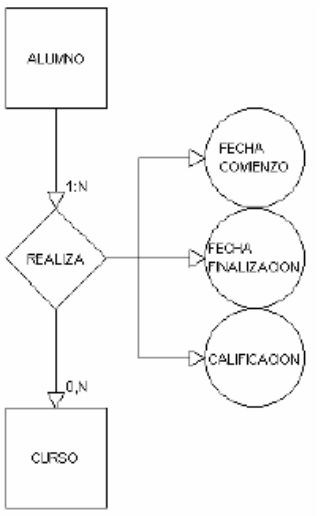


Ilustración 1-29. Atributos de una relación

A medida que se van estableciendo las interrelaciones hay que prestar especial atención a las interrelaciones cíclicas o redundantes, que son aquellas cuya eliminación no implica la pérdida de información. Pongamos como ejemplo en siguiente modelo entidad - relación:

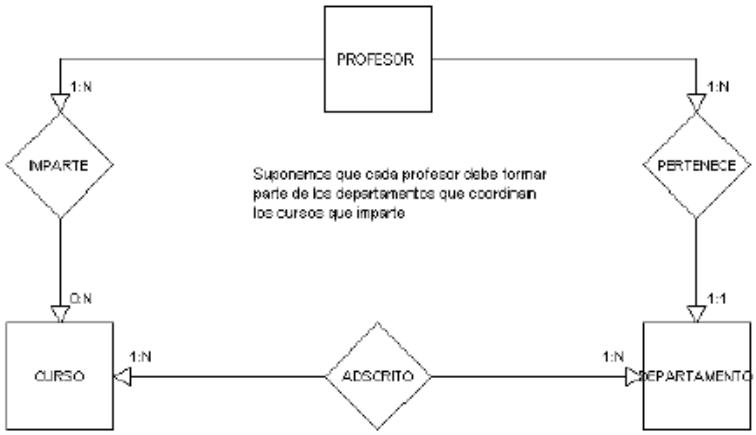


Ilustración 1-30. Interrelaciones cíclicas

Según se plantea el esquema la relación "**PERTENECE**" se puede suprimir por que para saber a qué departamentos pertenece un profesor basta con saber que cursos imparte y conociendo los cursos averiguamos que departamentos están asociados a los cursos. En este caso se dice que: "**PERTENECE**" **=** "**IMPARTE**" **+** "**ADSCRITO**".

Restricciones en las interrelaciones

Restricción de Exclusividad: Esta restricción se da cuando cada ejemplar de la entidad presente sólo puede combinarse con ejemplares de una sola de las entidades restantes. Por ejemplo:

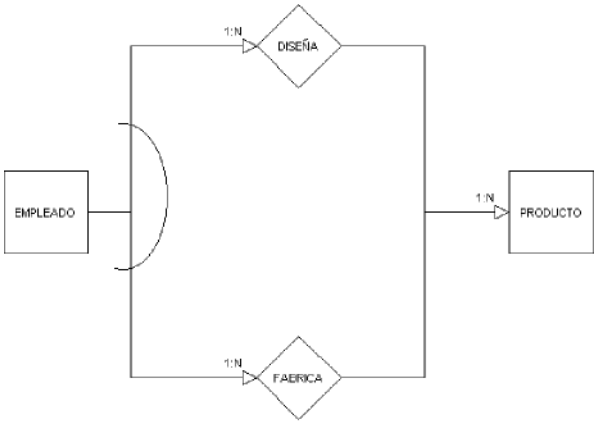


Ilustración 1-31. Restricción de exclusividad

Los empleados, en función de sus capacidades, o son diseñadores de productos o son operarios y los fabrican, no es posible que ningún empleado sea diseñador y fabricante a la misma vez.

Restricción de Exclusión: Se produce una restricción de exclusión cuando los ejemplares de las entidades sólo pueden combinarse utilizando una interrelación, este es el caso del siguiente ejemplo:

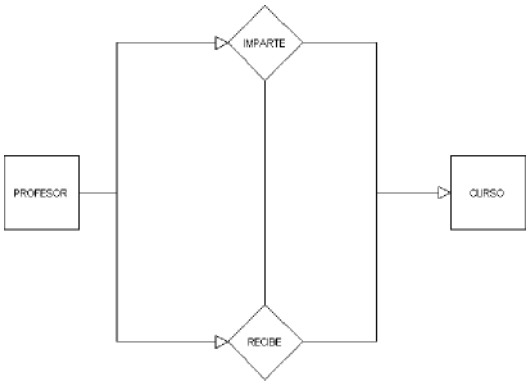


Ilustración 1-32. Restricción de exclusión

Un profesor no puede recibir e impartir el mismo curso.

Restricción de Inclusividad: Se dice que una relación es de inclusividad cuando todo ejemplar de una entidad participa en ambas interrelaciones, por ejemplo:

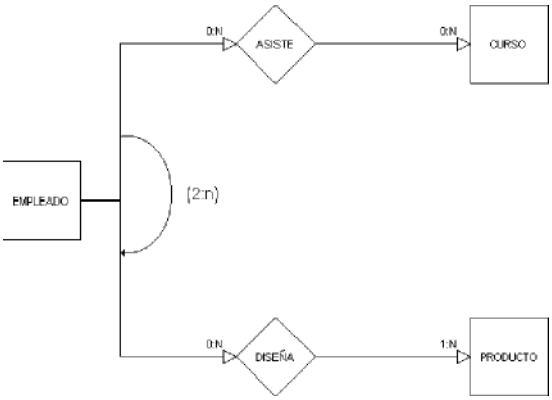


Ilustración 1-33. Restricción de inclusividad

Para que un empleado pueda trabajar como diseñador de productos deber haber asistido, al menos, a dos cursos.

Restricción de Inclusión: Se establece cuando la asociación entre entidades esta condicionada con la existencia de una segunda interrelación, por ejemplo:

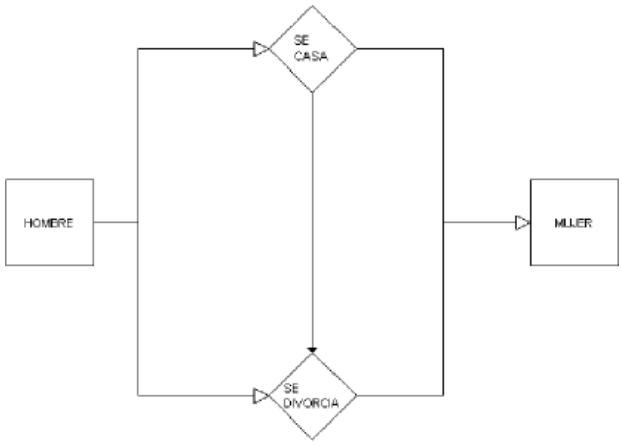


Ilustración 1-34. Restricción de inclusión

Para que un hombre se divorcie de una mujer deben esta casados.

1.12. NORMALIZACIÓN

El proceso de cristalización de las entidades y sus relaciones en formatos de tabla usando los conceptos relacionales se llama proceso de normalización y consiste en agrupar a los campos de datos en un conjunto de relaciones o tablas que representan a las entidades, sus características y sus relaciones de forma adecuada.

La razón de la normalización es asegurar que el modelo conceptual de la base de datos funcionará. Esto no significa que una estructura no normalizada no funcionará, sino que puede causar algunos problemas cuando los programadores de aplicación traten de modificar la base de datos para insertar, actualizar o eliminar datos.

Las formas de normalización fueron propuestas originalmente por Codd, entre 1971 y 1972. Posteriormente varios investigadores continuaron trabajando en esta teoría y a lo largo del tiempo han surgido varias formas de normalización que complementan y refuerzan a las enunciadas por Codd.

Las *formas normales* son una serie de restricciones que se definen sobre las estructuras relacionales para evitar anomalías al efectuar adiciones, eliminaciones o actualizaciones de tuplas. Con el fin de conseguir que una relación cumpla con una forma normal se efectúa un proceso de descomposición. Ésta implica dividir los atributos de una relación en dos subconjuntos (posiblemente con una intersección no vacía) sin que por ello se pierda alguna información contenida en la relación original.

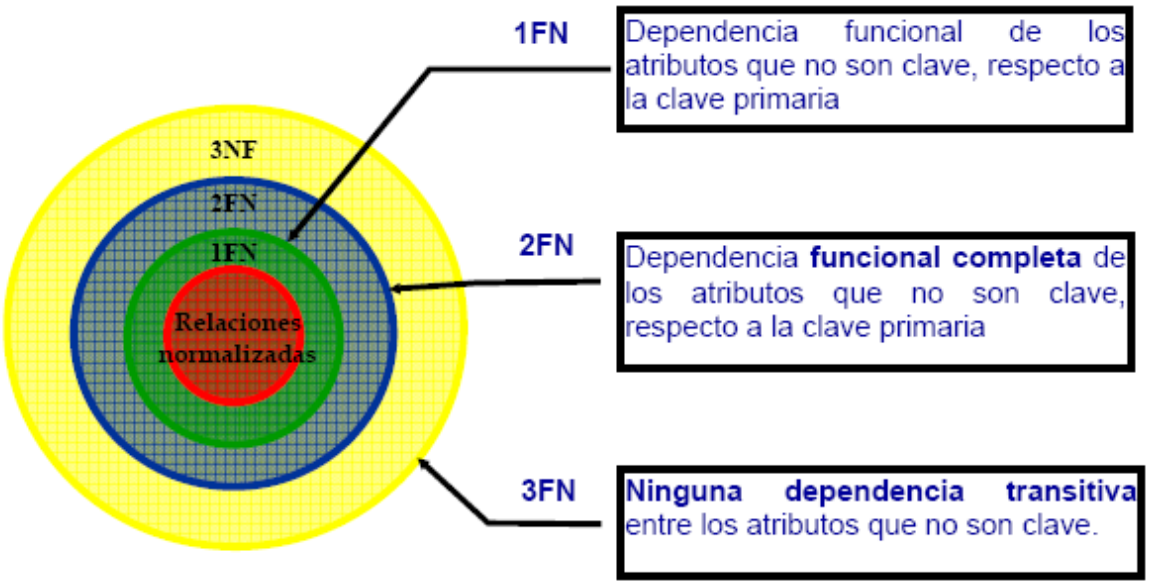


Ilustración 1-35. Restricciones de las formas formales

Las ventajas de la normalización son las siguientes:

- Evita anomalías en inserciones, modificaciones y borrados.
- Mejora la independencia de datos.
- No establece restricciones artificiales en la estructura de los datos.

Uno de los conceptos fundamentales en la normalización es el de dependencia funcional.

La dependencia funcional es una noción semántica donde cada dependencia es una clase especial de regla de integridad y representa una relación de uno a muchos.

La normalización se lleva a cabo en una serie pasos, cada paso corresponde a una forma normal que tiene ciertas propiedades. Conforme se va avanzando en la normalización, las relaciones tienen un formato más estricto y más fuerte y por lo tanto, son menos vulnerables a las anomalías de actualización. Las tres formas normales son las siguientes:

Primera Forma Normal (1FN): Una relación está en primera forma normal si, y sólo si, todos los dominios de la misma contienen valores atómicos, es decir, no hay grupos repetitivos. Si se ve la

relación gráficamente como una tabla, estará en 1 FN si tiene un solo valor en la intersección de cada fila con cada columna.

Segunda Forma Normal (2FN): Una relación está en segunda forma normal si, y sólo si, está en 1 FN y, además, cada atributo que no está en la clave primaria es completamente dependiente de la clave primaria.

Tercera Forma Normal (3FN): Una relación está en tercera forma normal si, y sólo si, está en 2FN y, además, cada atributo que no está en la clave primaria no depende transitivamente de la clave primaria. La dependencia es transitiva si existen las dependencias siendo atributos o conjuntos de atributos de una misma relación.

Básicamente, las reglas de normalización están encaminadas a eliminar redundancias e inconsistencias de dependencia en el diseño de las tablas.

1.13. HERRAMIENTAS CASE

Las Herramientas CASE[5] es la mejor base para el proceso de análisis y desarrollo de software, el avance tecnológico de las computadoras junto al dramático decremento en tamaño y costo han dado lugar a una gran variedad de aplicaciones que permiten resolver necesidades humanas muy especificas.

Desde el inicio de la creación de software ha existido la necesidad de crear herramientas automatizadas que permitan incrementar la productividad de los diseñadores de software, en un inicio, los esfuerzos se direccionaron hacia programas traductores, recopiladores, ensambladores, procesadores de macros, montadores y cargadores.

Al ver los beneficios de este conjunto de aplicaciones se genero una gran demanda por nuevo software con características similares. El significado de las siglas CASE viene de su acrónimo en ingles Computer Aided Assisted Automated Software Systems Engineering.

Ahora, las herramientas CASE se pueden definir como un conjunto de programas y ayudas que dan asistencia a los analistas, ingenieros de software y desarrolladores, durante las diferentes etapas del ciclo de desarrollo del Software: Investigación Preliminar, Análisis, Diseño, Implementación e Instalación.

Historia de las herramientas CASE

Las Herramientas CASE se iniciaron con un procesador de palabras que fue usado para crear y manipular documentación. Los 70's vieron la introducción de técnicas gráficas y diagramas de flujo de datos. Sobre este punto, el diseño y especificaciones en forma pictórica han sido extremadamente complejos y consumían mucho tiempo para realizar cambios.

Pronto se reemplazaron los paquetes gráficos por paquetes especializados que habilitan la edición, actualización e impresión en múltiples versiones de diseño. A diario, las herramientas gráficas integradas con diccionarios de base de datos para producir poderosos diseños y desarrollar

[5] CASE (Computer Aided Software Engineering, Ingeniería de Software Asistida por Computadora).

herramientas, podrían sostener ciclos completos de diseño de documentos. Como un paso final, la verificación de errores y generadores de casos de pruebas fueron incluidos para validar el diseño del software. Todos estos procesos pueden saberse integrados en una simple herramienta CASE que soporta todo el ciclo de desarrollo. La primera herramienta comercial se remonta a 1982, aunque algunos especialistas indican que algunos ejemplos de herramientas para diagramación ya existían.

No fue sino hasta 1985 cuando las herramientas CASE se volvieron realmente importantes en el proceso de desarrollo de software. Los proveedores prometieron a la Industria que muchas actividades serían beneficiadas por la ayuda de las CASE.

El objetivo en 1985 para muchos vendedores era producir software más rápidamente y las herramientas CASE fueron una familia de métodos que favorecieron la planeación, el análisis y el diseño además de que también brindaron beneficios para la mejora en la calidad, la fiabilidad, la utilidad y el rendimiento.

Clasificación de las Herramientas CASE

No existe una única clasificación de herramientas CASE y, en ocasiones, es difícil incluirlas en una clase en común. Podrían clasificarse así:

- Las plataformas que soportan.
- Las fases del ciclo de vida del desarrollo de sistemas que abarca.
- La arquitectura de las aplicaciones que produce.

Las herramientas CASE, en función de las fases del ciclo de vida que cubre, se pueden agrupar de la forma siguiente:

1. Herramientas integradas, I-CASE (Integrated CASE): Abarcan todas las fases del ciclo de vida del desarrollo de sistemas son llamadas CASE workbench.

2. Herramientas de alto nivel, U-CASE (Upper CASE): Orientadas a la automatización y soporte de las actividades desarrolladas durante las primeras fases del desarrollo, análisis y diseño.

3. Herramientas de bajo nivel, L-CASE (Lower CASE): Dirigidas a las últimas fases del desarrollo, construcción e implantación.

4. Juegos de herramientas, (Tools CASE): Son el tipo más simple de Herramientas CASE, automatizan una fase dentro del ciclo de vida. Dentro de este grupo se encontrarían las herramientas de reingeniería, orientadas a la fase de mantenimiento.

Componentes y funcionalidades de una herramienta CASE

Repositorio: Este se puede definir como la base de datos central de una herramienta CASE. El repositorio amplía el concepto de diccionario de datos para incluir toda la información que se va generando a lo largo del ciclo de vida del sistema, por ejemplo: componentes de análisis y diseño.

Módulos de diagramación y modelación: Algunos de los diagramas y modelos utilizados con mayor frecuencia son: diagrama de flujo de datos, modelo E-R, y técnicas matriciales.

Herramienta de prototipazo: El objetivo principal de esta herramienta es poder mostrar al usuario, desde los momentos iniciales del diseño, el aspecto que tendrá la aplicación una vez desarrollada. Ello facilitará la aplicación de los cambios que se consideren necesarios, todavía en la fase de diseño.

Generador de código: Normalmente se suele utilizar sobre ordenadores personales o estaciones de trabajo, por lo que el paso posterior del código al host puede traer problemas, al tener que compilar en ambos entornos.

Módulo generador de documentación: El módulo generador de la documentación se alimenta del repositorio para transcribir las especificaciones allí contenidas.

Herramientas CASE más utilizadas

ERwin: Es una herramienta para el diseño de base de datos, que Brinda productividad en su diseño, generación y mantenimiento de aplicaciones. Desde un modelo lógico de los requerimientos de información, hasta el modelo físico perfeccionado para las características específicas de la base de datos diseñada, además ERwin permite visualizar la estructura, los elementos importantes, y optimizar el diseño de la base de datos. Genera automáticamente las tablas y miles de líneas de *stored procedure y triggers* para los principales tipos de base de datos.

EasyCASE: Es un producto para la generación de esquemas de base de datos e ingeniería reversa, esta herramienta permite automatizar las fases de análisis y diseño dentro del desarrollo de una aplicación, para poder crear las aplicaciones eficazmente desde el procesamiento de transacciones a la aplicación de bases de datos de cliente/servidor, así como sistemas de tiempo real, es una herramienta multi-usuario.

Oracle Designer: Oracle Designer es un conjunto de herramientas para guardar las definiciones que necesita el usuario y automatizar la construcción rápida de aplicaciones cliente/servidor gráficas. Este esta integrado con Oracle Developer, lo cual provee una solución para desarrollar sistemas empresariales de segunda generación. Todos los datos ingresados por cualquier herramienta de Oracle Designer, en cualquier fase de desarrollo, se guardan en un repositorio central.

System Architect: Herramienta que posee un repositorio único que integra todas las herramientas, y metodologías usadas. En la elaboración de los diagramas, el System Architect conecta directamente al diccionario de datos, los elementos asociados, comentarios, reglas de validaciones, normalización.

DBDesigner: Producto destacable por su sencillez, permite modelar sobre MySQL y dispone de la capacidad de generar documentación e incluso pantallas de administración sobre PHP.

TableDesigner: Herramienta que permite la creación de bases de datos en Access y SQL Server de Microsoft.

1.14. MODELADO DE DATOS. CASO PRÁCTICO

Cartelera de cine.

Una cadena de cines desea presentar su cartelera por Internet mostrando al público los complejos que posee, su ubicación y las películas que exhiben.

La información de las películas que le gustaría consultar a la gente es: nombre de la película, reparto (actores principales y secundarios), duración, clasificación, género, compañía productora, director y alguna foto de promoción de la película.

Sobre los actores, hay veces en que a la gente le gusta saber más sobre su trayectoria o datos personales como: nombre real, nacionalidad, país de origen, fotografía, películas en las que ha participado y premios obtenidos.

Los géneros de las películas son: drama, acción, terror, romance, épica, infantil, comedia, etc.

Las clasificaciones son:

A: todo público B: adolescentes y adultos C: sólo adultos

Hay ocasiones en que la gente no tiene por cual decidirse y le gustaría saber la opinión de los expertos. Por lo que los críticos dan una calificación a las películas, promediándose estas calificaciones por película; una estrellita representa 2 puntos y 5 es la calificación más alta.

Los complejos pueden ofrecer varios servicios como son: cafetería, dulcería, juegos, guardarropa, valet parking, etc., los cuales pueden ser ofrecidos en uno o más complejos y al mismo horario.

Cada complejo tiene un número determinado de salas, las cuales pueden tener sonido: dolby, sound round, digital, etc.; así mismo pueden ser tipo estadio, a desnivel o escénicas.

Una sala exhibe solamente una película a la vez, repartida en varios horarios en los diferentes complejos.

Existen las veces en que un actor puede participar en 1 o más películas a la vez y en ciertas ocasiones hasta dirigirlas.

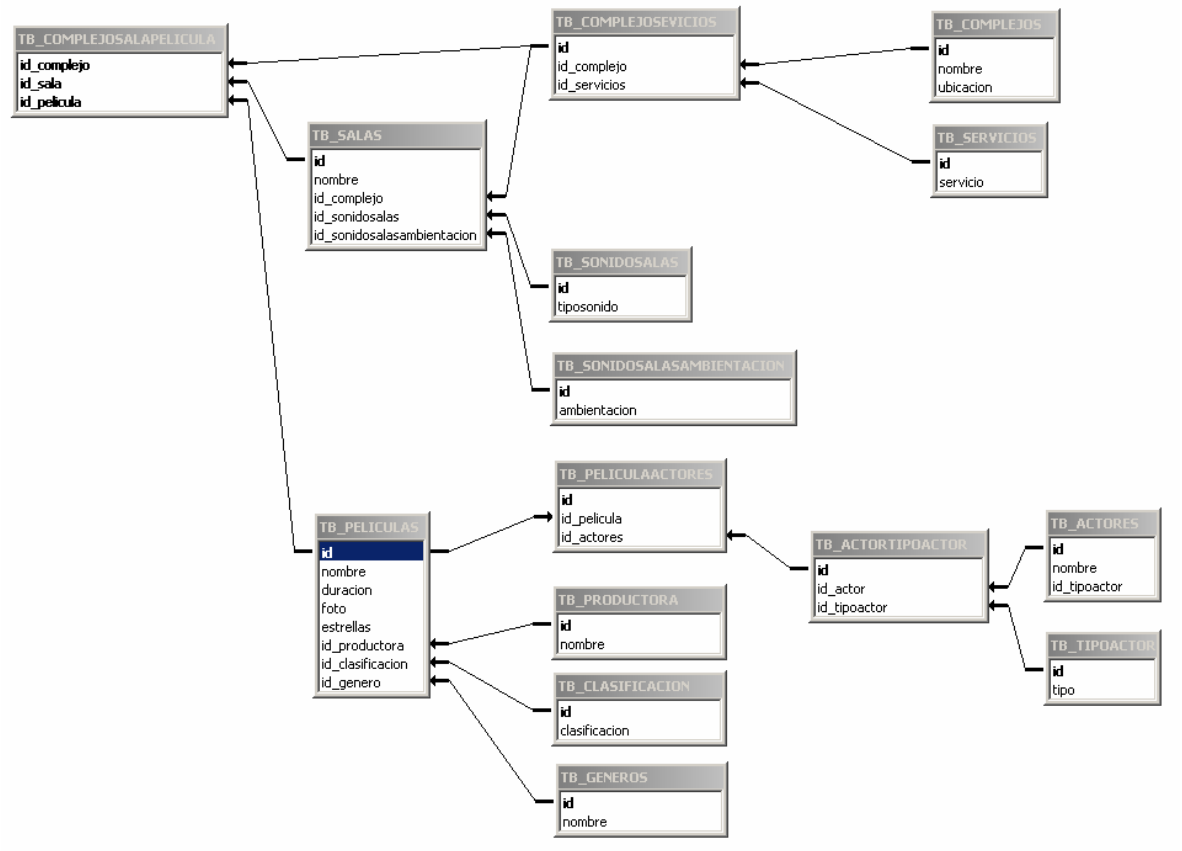


Ilustración 1-36. Uso de herramientas CASE para el modelado de datos.

Capítulo 2

SISTEMAS MANEJADORES DE BASES DE DATOS RELACIONALES (RDBMS)

Identificar la función, los componentes de un RDBMS y las normas que debe de cumplir internacionalmente para su elección en el entorno de trabajo, así como su importancia en la actualidad. Analizando ¿qué es un RDBMS? ó función de las RDBMS, componentes de las RDBMS, SQL ANSI 89, 92 y 99, principales RDBMS, consideraciones de hardware, tipos de datos usados por las RDBMS, tipos de usuarios, lenguajes transaccionales, conectividad.

2.1 ¿QUÉ ES UN RDBMS?

Entre la base de datos física (es decir, los datos tal y como están almacenados en la realidad) y los usuarios del sistema, existe un nivel de programas, denominado, manejador de bases de datos (MBD) o, en la mayoría de los casos, el sistema administrador de bases de datos DBMS (Data Base Management System).

Un RDBMS es el conjunto de programas que permiten la definición, manipulación y control de acceso para una o varias bases de datos.

Algunas características de los RDBMS son:

- Facilitan la integridad, seguridad y acceso de los datos.
- Los datos se almacenan como mínima redundancia.
- Las aplicaciones son independientes del almacenamiento físico de los datos.

2.2 FUNCIÓN DEL RDBMS

Un RDBMS debe permitir las siguientes condiciones en una base de datos:

- Los datos han de estar almacenados juntos.
- Tanto los usuarios finales como los programas de aplicación no necesitan conocer los detalles de las estructuras de almacenamiento.
- Los datos son compartidos por diferentes usuarios y programas de aplicación; existe un mecanismo común para la inserción, actualización, borrado y consulta de los datos.
- Los procedimientos de actualización y recuperación, comunes, y bien determinados, habrán de ser capaces de conservar la integridad, seguridad y confidencialidad del conjunto de datos.
- Tanto datos como procedimientos pueden ser transportables conceptualmente a través de diferentes RDBMS.

Conceptualmente lo que sucede en un RDBMS cuando un usuario realiza alguna petición, se presenta lo siguiente:

- El usuario solicita alguna petición a la base de datos empleando algún sublenguaje de datos determinado (SQL).
- El RDBMS interpreta esa solicitud y la analiza.
- El RDBMS inspecciona en orden el esquema externo de ese usuario, la correspondencia externa/conceptual asociada, el esquema conceptual, la correspondencia conceptual/interna y la definición de la estructura de almacenamiento.
- El RDBMS ejecuta las operaciones necesarias sobre la base de datos almacenada y devuelve una respuesta al usuario.

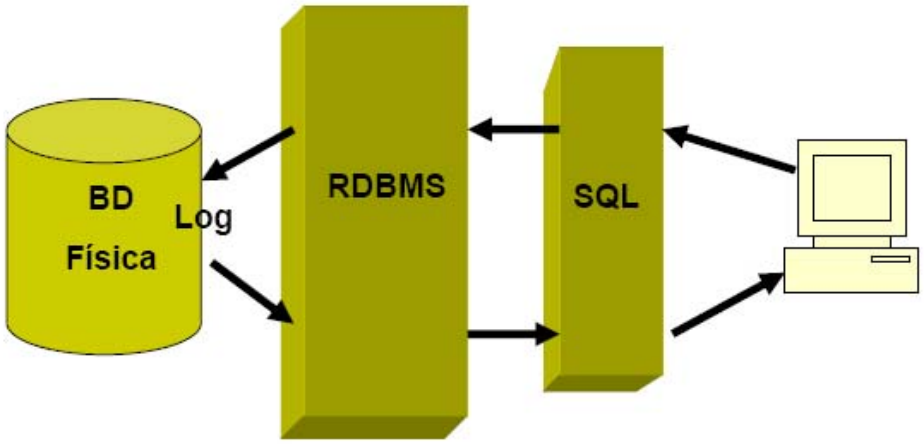


Ilustración 2-1. Operaciones de las RDBMS para dar una respuesta al usuario.

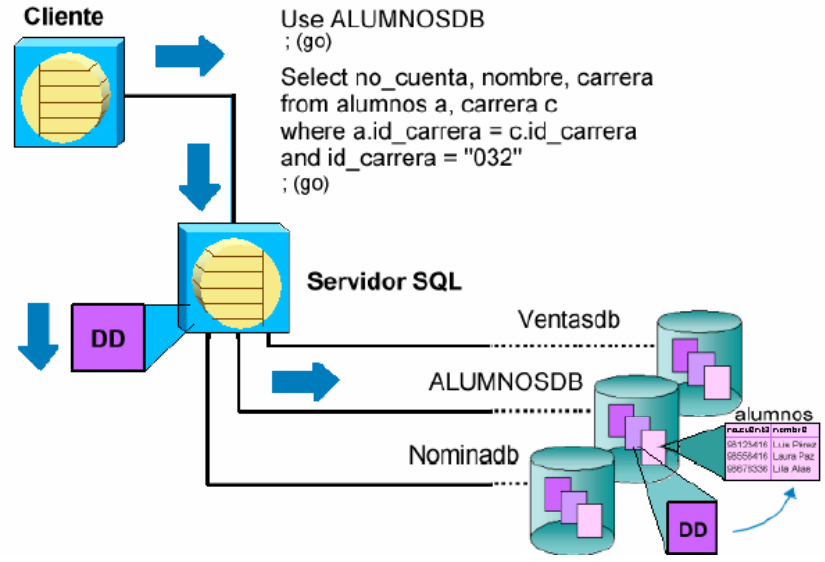


Ilustración 2-2. Seguimiento de una consulta SQL.

Esquemas de seguridad en el RDBMS

Dentro del Sistema Manejador de Base de Datos podemos encontrar un acceso multicapas, como el que se muestra a continuación:

- El usuario final debe tener una cuenta válida dentro de la capa del servidor (DBMS). *Seguridad a Nivel Servidor*
- El usuario final debe ser un usuario válido dentro de la capa de la base de datos. *Seguridad a Nivel de Base de Datos*
- El usuario final deberá tener permiso dentro de la capa de los datos. *Seguridad a Nivel de Permisos sobre Objetos y Comandos.*

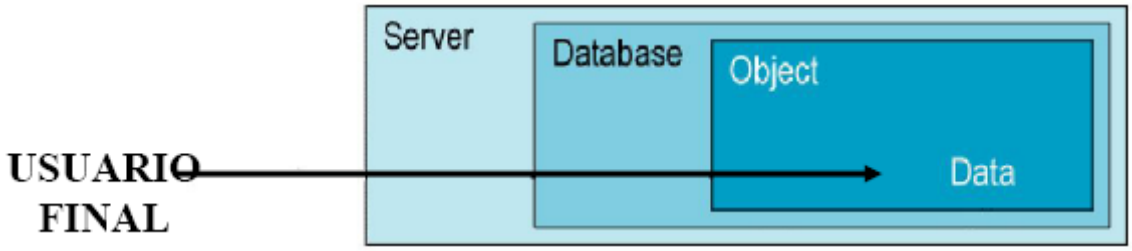


Ilustración 2-3. Niveles de seguridad en un DBMS

2.3 COMPONENTES DE UN RDBMS

DDL o Lenguaje de Definición de Datos: Se utiliza para crear, eliminar o modificar tablas, índices, vistas, triggers, procedimientos; es decir, nos permite definir la estructura de la base de datos mediante comandos como crear **(Create)**, eliminar **(Drop)**, o alterar **(Alter)**.

- **CREATE**. Utilizado para crear nuevas bases de datos, tablas, campos, índices, vistas, defaults, reglas, procedimientos, procedimientos, triggers.
- **ALTER**. Utilizado para modificar la estructura de una tabla para agregar campos o constraints.
- **DROP**. Utilizado para eliminar bases de datos, tablas, campos, índices, vistas, defaults, reglas, procedimientos, procedimientos, triggers.

El DML se utiliza para realizar la consulta y edición de la información contenida en la base de datos, esto implica: seleccionar, insertar, borrar, modificar.

Los DML se distinguen por sus sublenguajes de recuperación subyacentes; se pueden distinguir dos tipos de DML, el procedural y el no procedural. La principal diferencia entre ambos es que en los lenguajes procedurales se tratan los registros individualmente, mientras que en uno no procedural se opera sobre un conjunto de registros.

Las instrucciones relacionadas con este componente son:

- **SELECT**. Permite realizar consultas a la base de datos.
- **INSERT**. Empleado para agregar registros a una tabla.

- **UPDATE**. Utilizado para modificar los valores de los campos de una tabla.
- **DELETE**. Utilizado para modificar los valores de los campos de una tabla.

DCL o Lenguaje de Control de Datos: Se utiliza para la definición de los privilegios de control de acceso y edición a los elementos que componen la base de datos (seguridad), es decir, permitir o revocar el acceso.

Los permisos a nivel base de datos pueden otorgarse a usuarios para ejecutar ciertos comandos dentro de la base o para que puedan manipular objetos y los datos que puedan contener estos.

Las instrucciones relacionadas con este componente son:

- **GRANT**. Permite otorgar permisos a los usuarios sobre los objetos definidos en la base de datos, así como las operaciones a utilizar sobre ellos.
- **REVOKE**. Permite revocar permisos sobre los objetos definidos en la base de datos y las operaciones sobre los mismos.

DD o Diccionario de Datos: El contenido del diccionario puede considerarse como "datos acerca de los datos" (los cuales comúnmente reciben el nombre de metadatos), es decir, definiciones de otros objetos de la base de datos.

En particular, todos los diversos esquemas (externo, conceptual e interno), se almacenan físicamente en el diccionario, tanto en forma fuente como en forma objeto. Un diccionario amplio incluirá también las referencias cruzadas que indican, por ejemplo que partes de datos utiliza cada programa, que informes necesita cada departamento, etc. De hecho, el diccionario puede integrarse a la base de datos que describe, y por tanto, incluir su propia descripción.

Debe ser posible consultar el diccionario de la misma manera que cualquier otra base de datos, de modo que, por ejemplo, el DBA pueda describir con facilidad que programas tienen probabilidad de ser afectados por un cambio propuesto al sistema.

Sus principales funciones son las siguientes:

- Describe todos los elementos en el sistema.
- Los elementos se centran en los datos.
- Comunica los mismos significados para todos los elementos del sistema.
- Documenta las características del sistema.
- Facilita el análisis de los detalles para evaluar las características y determinar cómo deben realizarse los cambios.
- Localiza errores y omisiones del sistema.

La arquitectura de un RDBMS se muestra en la Ilustración 2-4.

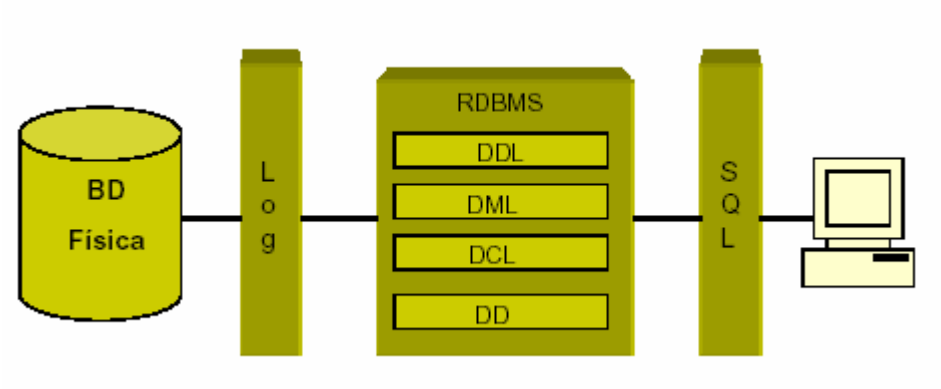


Ilustración 2-4. Arquitectura de un RDBMS

2.4 ANSI

ANSI SQL 89

En 1989, tanto ANSI como ISO, publicaron estándares que definían al modelo relacional en el manejo de Bases de Datos. Estos estándares son: ANSI X3.135-1989 e ISO/IEC9075:1989.

Características principales:

- Agregan la capacidad conocida como integridad referencial y la descripción de todo el modelo relacional.
- Se definió que el lenguaje SQL está compuesto por comando, cláusulas, operadores y funciones de agregado. Estos elementos se combinan para definir y manipular la base de datos.
- Se establecen los elementos de un DBMS (DDL, DML y DCL), así como las instrucciones y sintaxis relacionadas con cada uno de ellos.
- Establecimiento de las cláusulas del comando **`select`**, las cuales son: **`From`**, **`Where`**, **`Group By`**, **`Having`**, **`Order By`**.
- Definición de los operadores lógicos: **`AND`**, **`OR`** y **`NOT`**.
- Definición de los operadores de comparación.
- Se determinan las funciones de agregado, tales como: **`AVG`**, **`COUNT`**, **`SUM`**, **`MAX`**, **`MIN`**.

ANSI SQL 92

Características principales:

- Toma todas características definidas en el estándar ANSI SQL 89.
- Permite la definición de esquemas.

- Permite la definición de dominios por parte de los usuarios, es decir, tipos de datos definidos por el usuario.
- Menciona las consideraciones para realizar consultas sencillas, multi-tablas y subconsultas.
- Incluye los operadores **EXISTS** y **NOT EXISTS**.
- Contempla el uso de la palabra **DISTINCT** en una consulta.
- Menciona algunas consideraciones para el uso de las cláusulas **GROUP BY** y **HAVING**.
- Especifica la definición de vistas en una base de datos.

ANSI SQL 99

Características principales:

- Toma todas características definidas en los estándares ANSI SQL 89 y 92.
- Incluye nuevos tipos de datos escalares: **BOOLEAN**, **CLOB** (objeto de caracteres largo) y **BLOB** (objeto binario grande).
- Presenta dos nuevos operadores de totales: **EVER** y **ANY**.
- Incorpora generadores de tipo de dato: **REF**, **ARRAY** y **ROW**.
- Incluye los operadores **EXISTS** y **NOT EXISTS**.
- Contempla el uso de la palabra **DISTINCT** en una consulta.
- Soporta una opción **LIKE** en **CREATE TABLE**, lo cual permite que todas o algunas definiciones de columna de una nueva tabla sean copiadas a partir de otra ya existente.
- Incluye la cláusula **WITH** para introducir nombres abreviados para determinadas expresiones.
- Incorpora una nueva expresión de condición **IS DISTINCT** para la cláusula **FROM**.

2.5 PRINCIPALES RDBMS

Microsoft SQL Server

Características principales:

- Compatibilidad con estándares de W3C, incluyendo XML, Xpath, XSL, HTTP.
- Obtiene código XML de las consultas realizadas con SQL.
- Manipulación de documentos XML.
- Manejo de bases de datos distribuidas.
- Manejo de varias particiones físicas para almacenamientos de datos flexibles.
- Permite realizar algunas tareas de mantenimiento y administración de la base de datos sin tener que darla de baja.

- Permite realizar acciones OLAP (Online Analyzing Processing), herramienta que permiten analizar datos almacenados en una Base de Datos, por medio de cubos de información.
- Consulta y modificación de cubos virtuales de manera gráfica.
- Conectividad con clientes ODBC y JDBC.

Requerimientos

Sistema Operativo	Windows NT Server Windows 2000 Server Windows 2000 Advanced Server Windows 2000 Data Center Windows Server 2003
Hardware	PIII o superior, según los requerimientos del sistema operativo.
Memoria	Enterprise Edition: 64 MB como mínimo. Standard Edition: 32 MB como mínimo.
Disco Duro	SQL Server 2000, instalación completa 180 MB SQL Server 2000, instalación típica 170 MB SQL Server 2000, Instalación mínima 65 MB SQL Server 2000, solo herramientas cliente 90 MB
Observaciones	Es necesario disponer de Microsoft Windows 2000 Server para algunas características de SQL Server 2000. Nota: los requerimientos de hardware y software cambian dependiendo de la versión que se desee instalar.

Sybase

Características principales:

- Diseñado para soportar aplicaciones OLTP (On Line transaction Processor, ambiente diseñado para insertar, actualizar y borrar datos en una base de datos).
- Permite integrar aplicaciones basadas en XML con la base de datos y crear reglas de negocio que ejecuten Java Beans.
- Conectividad con clientes ODBC y JDBC.
- Soporte para Blob's (Large Objects).
- Permite realizar queries XQL, lo cual significa que utiliza un motor abierto para búsqueda dentro de contenidos XML almacenados en la base de datos, o en un URL
- Tamaño expandido de filas y datos, es decir soporta filas más grandes, columnas más grandes. Se soportan ahora tamaños de páginas de 2k, 4k, 8k o 16k (entre mas tamaño de página mayor rendimiento de operaciones SQL)
- Compresión de copias de respaldo.
- Bloqueo a: nivel de fila, nivel de pagina de datos, nivel de página (datos e índices), nivel de tabla.

Requerimientos

Plataformas Soportadas		Sistema Operativo	
	Empresa	32 Bits	64 Bits
	Sun Microsystems Hewlett Packard IBM	HP-UX AIX IRIX	Solaris HP-UX AIX

	SGI Compaq Linux Microsoft Windows	Red Hat NT,98 (sólo clientes)	TRU64
Memoria	Sistema Operativo	Memoria	
	AXP HP 32 HP 64 Linux NT Sun 32 Sun 64	94 MB 64 MB 90 MB 32 MB 46 MB 66MB 92 MB	
Disco Duro	240 MB		
Especificaciones del servidor	Bases de datos totales 32,767 • Tamaño de la base datos 4 TB • Bases de datos en update 16 • Tablas en una consulta 50 • Usuarios por base de datos 2,146,484,223 • Columnas por tabla 1024 • Tamaños de página 2k, 4k, 8k, 16k • Argumentos por procedimiento 1024		

Oracle

Características Principales

- Ofrece varias plataformas de desarrollo para Internet y aplicaciones tradicionales, tales como: XML, Enterprise Java Engine, SQL y PL/SQL, C, C++, entre otras.
- Soporte Unicode.
- Extiende las habilidades de una base de datos para Internet.
- Amplía distintos mecanismos para protección de datos.
- Soporta OLTP y OLAP.
- Contiene mecanismos de gran funcionalidad y flexibilidad para compartir la información almacenada en la base de datos con otras bases de datos o aplicaciones.
- Conectividad con clientes ODBC y JDBC.
- Soporte para Blob's.
- Ofrece escalabilidad y performance sin modificar las aplicaciones instaladas.
- Soporta columnas con cifrado de datos.
- Permite replicación de bases de datos (bases de datos distribuidas).
- Ofrece distintas herramientas para la administración de la base de datos.
- Redefinición de tablas en línea.
- Respaldo y recuperación en línea.

Plataformas:

Solaris

- HP-UX

- Compaq Tru64
- AIX
- HP Alpha
- Linux
- Windows NT/2000/XP Professional

Informix y DB2

Características principales:

- Soporta Bases de datos de más de 4 TB.
- Soporte para acceso a la base de datos vía Web.
- Provee acceso a cualquier tipo de cliente.
- Permite manejo de base de datos distribuidas.
- Capacidad de replicación de bases de datos.
- Permite realizar queries en paralelo.
- Contiene plataformas de desarrollo con SPL: (Informix Stored Procedure Language), C, Java, XML.
- Conectividad vía ODBC, JDBC, OLE/DB.
- Soporta aplicaciones para eCommerce, e inteligencia de negocios.
- Soporta OLAP y OLTP.

Plataformas que soporta:

- IBM AIX.
- SGI IRIX.
- Sun Solaris.
- HP-UX.
- Compaq Tru64.
- Linux.
- Windows NT/2000/XP/2003.

PostgreSQL

Características principales:

- Base de datos de distribución libre.
- Velocidad.
- Confiabilidad.
- Flexibilidad.

- Bajo costo de operación.
- Conformación a estándares ANSI.
- Estrategia de almacenamiento MVCC[1] para grandes volúmenes.
- Soporta replicación de bases de datos.
- Interfases nativas para ODBC, JDBC, C, C++, PHP, Perl, TCL, XML.
- Soporta SSL nativo.

Plataformas que soporta:

- Corre bajo cualquier plataforma UNIX
- PostgreSQL también corre nativamente sobre sistemas operativos basados en Microsoft Windows NT tales como Win2000, WinXP y Win2003. Se encuentra disponible un instalador preempacado http://pgfoundry.org/projects/pginstaller.
- Las versiones de Windows basadas en MSDOS (Win95, Win98, WinMe) pueden correr PostgreSQL usando Cygwin.
- Existe un port para Novell Netware 6 en http://forge.novell.com
- Hay una versión para OS/2 (eComStation) en http://hobbes.nmsu.edu/cgi-bin/h-search?sh=1&button=Search&key=postgreSQL&stype=all&sort=type&dir=%2F.

MySQL

Características principales:

- Soporta los estándares ANSI.
- Contiene esquemas de almacenamiento independiente que se pueden seleccionar de acuerdo a las necesidades.

InnoDB para transacciones y bloqueo de registros.

MyISAM sin transacciones

- Soporte para SSL.
- Query's con manejo de cache que puede incrementar el performance de la base de datos en un 200%.
- Permite manejo de replicación de bases de datos.
- Soporta indexado de texto.

---

[1] MVCC - Multi-Version Concurrency Control o Control de Concurrencia Multi-Versión, es una tecnología que usa PostgreSQL para el tratamiento de bloqueos. Existen RDBMS con capacidades SQL, como MySQL o Access, que en ocasiones la lectura de los datos tiene que esperar para acceder a la información. La espera está provocada por usuarios que están escribiendo en la base de datos. El lector está bloqueado por los escritores que están actualizando registros. Usando MVCC PostgreSQL posibilita que sea poco perceptible los bloqueos en sus bases de datos.

Plataformas que soporta:

- Linux
- Windows.
- FreeBSD
- Sun Solaris.
- IBM-AIX.
- MacOS X.
- HP-UX.

Clasificación por ámbito

- Comerciales (SQL Server, Sybase, Oracle, Informix, DB2)

Tipos de Licencia:

Por usuario conectado.

Por procesador.

- Software Libre (PostgreSQL, MySQL, Sybase (Linux)).

Clasificación por volumen de información

- Corporativo (Oracle, Informix, Sybase, DB2, PostgreSQL)
- Departamental (SQL Server, SQL Anywhere, MySQL).

2.6 CONSIDERACIONES DE HARDWARE

El éxito para las compañías reside en comprender los elementos de los negocios electrónicos y de los avances de la tecnología.

Sin embargo, un sitio de comercio electrónico no sólo depende de una página bonita, sino de los componentes que están atrás de esta, como son:

- el hardware empleado,
- los servicios,
- el software,
- el soporte, etc.

La elección del Hardware no es algo que debe considerarse a la ligera y no sólo involucra lo poderoso que pueda ser, ni que tanto deseamos abarcar en nuestro sitio, sino también lo afecta en el mayor de los casos nuestro presupuesto. Por eso hay veces que es necesario considerar:

- Construir nuestro propio sitio, escogiendo uno a uno los componentes, el equipo, los proveedores de servicios, el software, el soporte técnico, etc.
- Rentar el equipo y los servicios de forma total o de manera parcial por una cuota mensual

Procesador

La velocidad del procesador afecta de manera directa la rapidez de procesamiento de la computadora, pudiéndose tomar esta como referencia para evaluar su desempeño, por ejemplo, influye en la compilación de programas. De igual forma, utilizar el servidor de consola implica desviar memoria RAM de otros procesos.

Tarjeta madre

Otra parte esencial de una PC es la tarjeta madre, es la parte más complicada de actualizar por que se encuentra muy ligada al procesador. Hay que tener en cuenta que existen fabricantes que utilizan tecnología propietaria en sus equipos y actualizar el equipo significa cambiar: gabinete, tarjeta madre, procesador y memoria.

Tipos de canal (BUS)

PCI, Interconexión de Componentes Periféricos. Está trabaja a 64 bits, un ancho de banda de 133 MBps[2] y se usa en equipos 80486, Pentium, Pentium MMX, Pentium Pro hasta Pentium III.

AGP Puerto Acelerado de Gráficos (AGP, Accelerated Graphics Port). Una especificación de interfaz nueva desarrollada por Intel Corporación. AGP esta basado en PCI, pero es diseñado sobre todo para las exigencias de rendimiento de gráficos 3-D, ya que el controlador de gráficos puede tener acceso directamente a la memoria principal.

El canal AGP es de 32 bits de ancho y corre a 66 MHZ. Esto se traduce en un ancho de banda total de 266 MBps, a diferencia del ancho de banda PCI de 133 MBps. AGP también aporta dos modos opcionales más rápidos, con rendimientos de 533 MBps y 1.07 GBps[3]. Además, permite que las texturas 3-D sean almacenadas en la memoria principal en lugar de hacerlo en la memoria de vídeo.

Memoria

La memoria RAM se encuentra también ligada al diseño de la tarjeta Madre, es importante la velocidad de acceso en nanosegundos (ns), que es el tiempo que le lleva al CPU tener acceso a la RAM. Entre más nanosegundos tarde más lento es el acceso.

Canal (BUS)

De igual relevancia es el tamaño del canal (expresado en bits), ya que un dato se transmite en paralelo, n-bits a la vez. Entre más bits haya en el canal, más rápido pasarán los datos por la RAM. La cantidad de memoria es un factor significativo con respecto a cuántos procesos a la vez puede manejar el servidor y que tan rápido puede procesarlos.

---

[2] MBps - Unidad de transferencia de datos, Megabits por segundo, Mbit/s.

[3] GBps - Unidad de transferencia de datos, Gigabit por segundo, Gbit/s.

Disco duro

Para un sitio con uno o más usuarios, el disco duro no representa una consideración primaria en el momento del desempeño. Sin embargo, cuando se incrementa tanto el número de usuarios como la demanda de servicios, el tamaño y tipo del disco duro debe de ser tomado en cuenta.

El tipo de disco duro IDE (Integrated Drive Electronics) es la más usada, son buenas para manejar muchos usuarios pero sólo si acceden éstos un archivo a la vez. Por lo que si queremos tener un sitio grande seria recomendable utilizar un disco duro SCSI (Small Computer System Interface) con el que se pueden tener acceso a múltiples archivos a la vez, además se dirige más rápido a un espacio swap que una IDE.

El tipo de disco duro SCSI tiene una velocidad de transmisión de información más rápida (hasta 80 megabytes por segundo), permite el acceso a múltiples archivos a la vez, además se dirige más rápido a un espacio swap que un IDE.

Se puede encontrar los siguientes tipos de SCSI:

- SCSI-1: Usa un BUS de 8 bits, y soporta velocidades de transmisión de datos de 4 MBps.
- SCSI-2: Tiene las mismas características que el SCSI-1, pero usan un conector de 50 terminales en vez de un conector de 25 terminales, y soporta dispositivos múltiples.
- Wide SCSI: Usa un cable más amplio (cable de 168 líneas con 68 terminales) para soportar transferencias a 16 bits.
- Fast SCSI: Usa un canal de 8 bits, pero dobla la velocidad de reloj para soportar una velocidad de transferencia de datos a 10 MBps.
- Fast Wide SCSI: Usa un canal de 16 bits y soporta una velocidad de transferencia de datos de hasta 20 MBps.
- Ultra SCSI: Usa un canal de 8 bits y soporta velocidades de transferencia de datos de hasta 20 MBps.
- Wide Ultra SCSI: Usa un canal de 16 bits y soporta velocidades de transferencia de datos de hasta 80 MBps.
- SCSI-3: Usa un canal de 16 bits y soporta velocidades de transferencia de datos de hasta 40 MBps. También llamado Ultra Wide SCSI.
- Ultra2 SCSI: Usa un canal de 8 bits y soporta velocidades de transferencia de datos de hasta 40 MBps.

RAID

Un RAID contiene varias unidades de disco que funcionan en un arreglo en paralelo para la tolerancia a fallas y el mejoramiento del funcionamiento. Actúa como una enorme unidad de disco, pero almacena la información del sitio en varios discos en lugar de uno.

Existen diferentes niveles de RAID. Los más comunes, son: 0, 1, 3 y 5:

Nivel 0: Implementa un arreglo de discos fragmentados, los datos son fragmentados en bloques y cada bloque es escrito a una unidad de disco por separado. El funcionamiento de las Entradas/Salidas (I/O) es mejorado al extender la carga de I/O a través de varios canales y discos.

Ventajas: Es fácil de implementar ya que tiene un diseño sencillo. No gasta tiempo en operaciones sin paridad.

Desventajas: No es un verdadero RAID ya que no soporta tolerancia a fallas. La falla de un sólo disco puede resultar en la pérdida de todos los datos en el arreglo.

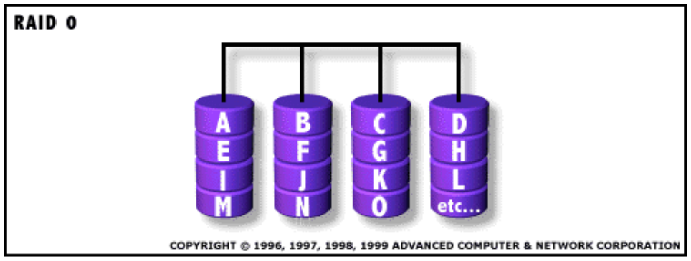


Ilustración 2-5. RAID 0

Nivel 1: Proporciona un espejo del disco.

Para un alto funcionamiento, el control debe ser capaz de llevar a cabo dos lecturas simultáneas o dos escrituras separadas por cada par espejeado.

El RAID 1 requiere que un mínimo de 2 discos para implementarlo.

Ventajas: La velocidad de la lectura doble es igual a la de escritura. Redundancia del 100 % en los datos, lo que significa que no será necesaria la reconstrucción de los datos en caso de falla, solamente se necesitará una copia del disco de reemplazo.

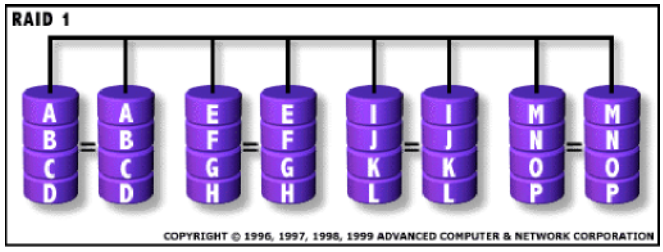


Ilustración 2-6. RAID 1

Ventajas: La velocidad de transferencia por bloques es igual que un disco sencillo. Bajo ciertas circunstancias, el RAID 1 puede soportar varias fallas simultáneas. Es de fácil diseño.

Desventajas: No soporta la remoción de unidades en línea cuando esta implementada por software.

Nivel 3: Igual que el Nivel 0, el bloque de datos es subdividido y escrito sobre los discos de datos. La paridad de partición es generada sobre la Escritura, por lo que se reserva un disco dedicado a la corrección de errores en los datos y revisada en la lectura. Esto proporciona un buen funcionamiento y cierto nivel de tolerancia a fallas.

Requiere de al menos tres unidades de discos para su implementación.

Ventajas: Tiene una alta tasa de transferencia de datos en la lectura y escritura. La falla de un disco tiene bajo impacto sobre el rendimiento.

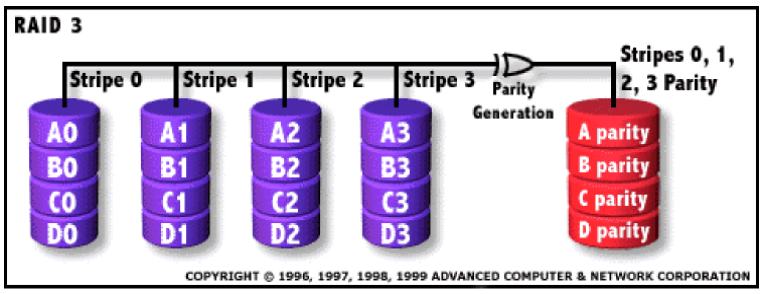


Ilustración 2-7. RAID 3

Desventajas: El diseño del controlador es muy complejo. Es difícil de implementar y se necesitaría muchos recursos. La tasa de transacción es igual al de una unidad de disco en el mejor de los casos.

`Nivel 5`: Proporciona datos fragmentados a nivel byte y también corrección errores en los datos por fragmentación. Este resulta excelente para un buen funcionamiento y la tolerancia de fallas.

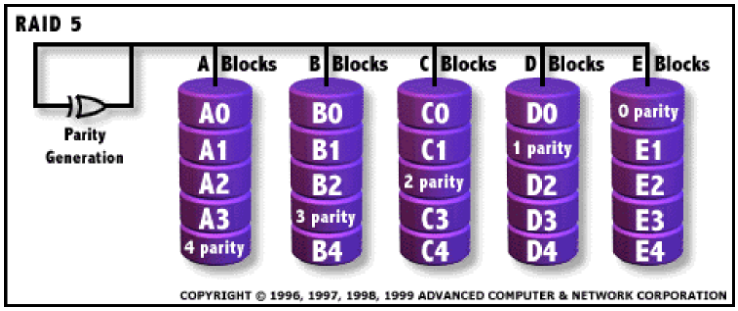


Ilustración 2-8. RAID 5

Requiere de al menos tres unidades de discos para su implementación.

Ventajas: Tiene una alta tasa de transferencia de datos en la lectura y mediana en la escritura.

Desventajas: En caso de falla tiene un mediano impacto en el rendimiento. Es muy complejo de implementar y difícil de reconstruir los datos en caso de falla comparado con el RAID 1. La tasa de transferencia de datos por bloque individual es la misma que la de un simple disco.

2.7 USUARIOS EN UNA BASE DE DATOS

El programador de aplicaciones, quien se encarga de escribir los programas de aplicación que utilizan la base de datos.

El usuario final, el cual tiene acceso a los datos de la base a través de alguna aplicación desarrollada o utilizando una interfaz incluida como parte integral de los programas del DBMS.

El DBA (database administrator, Administrador de la base de datos).

Roles o funciones

Los roles o perfiles sirven como medio para conceder privilegios sobre todo el sistema a un usuario que los requiera.

Estos permisos pueden verse reflejados sobre objetos o sobre el mismo sistema. En los RDBMS ya vienen predeterminados algunos, sin embargo no en todos se pueden crear nuevos roles.

- DBA (database administrator, administrador de la base de datos).
- DBO (database owner, Dueño de la base de datos).
- SSO (Security System Officer, Oficial del sistema de seguridad).
- Oper (Operador).

Lleva a cabo tareas administrativas inconexas a aplicaciones específicas. El administrador del sistema no es necesariamente único; el rol puede ser otorgado a cualquier número de cuentas individualmente y para ello las funciones del DBA deben estar centralizadas o muy bien coordinadas. Las tareas del DBA incluyen:

- Decidir el contenido de la base de datos.
- Crear la estructura de almacenamiento y los métodos de acceso.
- Administrar y controlar la seguridad física y lógica de la base de datos.
- Monitoreo del comportamiento y crecimiento de la base de datos.
- Procedimientos de respaldo y depuración de la base de datos.
- Salvaguardar la documentación, respaldos y diccionario de datos tanto de la base datos como de sus aplicaciones.
- Procedimientos de contingencia y recuperación de la base de datos.
- Modificar la base de datos o la descripción de la organización física.
- Otorgar permisos de acceso y prioridades a los diferentes usuarios.
- Especificar las limitaciones de integridad.
- Ser el enlace con el usuario.
- Instalación del servidor SQL.
- Diagnóstico de problemas del sistema, así como la corrección de los mismos.
- Otorgar y revocar roles en el servidor.
- Configuración del servidor SQL.
- Control de procesos en el servidor.

SSO (Oficial del sistema de seguridad)

El Oficial del Sistema de Seguridad es responsable de la seguridad del sistema y tiene las siguientes funciones.

- Crear cuentas de usuario.
- Bloquear cuentas.
- Otorgar y revocar los roles de SSO y Oper.
- Cambiar password a cualquier cuenta en el sistema.
- Poner intervalos de expiración a los passwords.
- Manejar el sistema de auditoria.

Oper (Operador)

El operador del sistema es responsable de respaldar y recuperar todas las Bases de Datos del Sistema. Depende del DBMS para la nomenclatura y funciones del operador.

En Oracle tenemos:

- OSOPER: Permite al usuario llevar a cabo el inicio y apagado del servidor, respaldo y mantenimiento de bases de datos.
- OSDBA: Administrador del sistema, tiene todos los privilegios del sistema, sólo este role permite crear bases de datos y recuperarlas.

En SQL Server tenemos:

- sysadmin: Administrador del sistema (sa o dba).
- serveradmin: Se usa para conceder a un usuario la autoridad para realizar cambios en la configuración del servidor.
- processadmin: Permite gestionar los procesos activos en SQL Server.
- dbcreator: Proporciona a un usuario la capacidad de crear y modificar bases de datos en el sistema.
- diskadmin: Este se proporcionar por lo regular con el anterior y sirve para manejar los dispositivos de almacenamiento (archivos).

2.8 LENGUAJES TRANSACCIONALES

Para comunicarse con el servidor SQL y manipular objetos almacenados en él, los programas cliente y los procedimientos almacenados usan una variedad de SQL. Contiene extensiones importantes que permiten flexibilidad y facilidad en el lenguaje (estructuras de control de flujo, variables locales, procedimientos almacenados y triggers).

Algunos lenguajes transaccionales:

- PostgreSQL
    - PL/pgSQL - SQL Procedural Language
    - PL/Tcl - Tcl Procedural Language
    - PL/Perl - Perl Procedural Language
    - PL/Python - Python Procedural Language
- Sybase

    - T-SQL
- Oracle
    - PL/SQL
- SQL Server
    - Transact-SQL

.NET Framework languages

Transact-SQL. Contiene estructuras de control de flujo de programas, declaración de variables, paso de parámetros, entre otras. Estas permiten crear, por ejemplo, procedimientos almacenados y triggers.

- Permite la definición de tipos de datos por parte del usuario.
- Amplía la función de la instrucción **SELECT**.
- Agrega una gran variedad de funciones para el manejo de datos tipo carácter, fecha y numéricos, a las contenidas en el estándar SQL.

2.9 CONECTIVIDAD

ODBC

Open Database Connectivity, Método estándar de acceso a bases de datos desarrollado por Microsoft. Su propósito es hacer posible el acceso de cualquier dato, gestionado por un RDBMS, desde cualquier aplicación.

JDBC

JDBC "Java Database Connectivity" (Conectividad de Base de Datos de Java), es un marco de programación para los desarrolladores Java que escriben programas que tienen acceso a la información guardada en bases de datos. JDBC se utiliza comúnmente para conectar un programa del usuario con una base de datos por "detrás de la escena", sin importar qué software de administración o manejo de base de datos se utilice para controlarlo. Esto se logra utilizando las clases JDBC API, que está disponible en el sitio de Sun.

```
Connection getConnection() {
  try {
    Class.forName("com.sybase.jdbc.SybDriver");
    con = DriverManager.getConnection("jdbc:sybase:Tds:teseo.dcaa.unam.mx:4100/Base",
"Usuario","password");
  } catch (SQLException sqle) {
    sqle.printStackTrace();
  } catch (ClassNotFoundException cnfe) {
    cnfe.printStackTrace();
  }
return con;
}
```

Código 2-1. Conexión a una Base de Datos en Sybase mediante JDBC

PHP

Hypertext Preprocessor. Lenguaje de programación de distribución libre que permite realizar scripts con código HTML "embebido" para crear páginas Web dinámicas. Ofrece acceso a RDBMS's por medio de la compilación de librerías específicas con APACHE.

```
<?
class conexionPg {
  var $conexion;
  function getConexion() {
    $this->conexion=@pg_connect("host=localhost port=5432 user=cPhp password=curs0dbname=
Agenda");
    if($this->conexion) {
      #echo "Se obtuvo la conexión";
      return $this->conexion;
    }else {
      echo "No Hay Conexión";
    }
  }
}
?>
```

Código 2-2. Conexión a una Base de Datos en PostgreSQL mediante PHP

Cliente

Es cualquier aplicación que correo se ejecuta en cualquier computadora personal, Workstation y se comunica con un equipo servidor para ejecutar alguna operación especifica. Por ejemplo: acceso a bases de datos, accesos a correo electrónico, acceso a FTP, etc.

Ejemplos:

- Sybase:
    - OpenClient: isql (UNIX) y wisql (Windows)
- Oracle:
    - Sqlplus
- Informix:
    - Informix Client
    - Dbaccess

2.10 ¿CUAL MANEJADOR DE BASES DE DATOS ELEGIR? CASO PRÁCTICO

¿Cuál Manejador de bases de datos elegir? La respuesta a esta interrogante no se puede dar sin un análisis de las necesidades a las que se necesite dar solución. Antes de aventurarse a dar alguna respuesta es necesario hacerse un par de preguntas: ¿Para que tipo de empresa? ¿Cual es la cantidad de información que se pretende manejar?

Entre las diferentes empresas y la cantidad de información que manejan se pueden englobar en los siguientes tipos:

- Transnacionales y Corporativos con gigabytes o hasta terabytes de información.
- Grandes empresas con gigabytes de información.
- Pequeñas y medianas empresas con megabytes o gigabytes de información.
- Microempresas con kilobytes o megabytes de información.
- Presencia en Internet con kilobytes o megabytes de información.

Transnacionales y corporativos con gigabytes o terabytes de información. Estas empresas son las menos versátiles en cambiar su esquema de hacer las cosas, ya que tienen soluciones probadas que les resuelve su necesidad de mantener la información, por lo que es difícil recomendar una solución distinta a la ya usada, pudiendo usar distintos manejadores de bases de datos para satisfacer sus necesidades de administrar la información.

Grandes empresas con gigabytes de información. Estas empresas se pueden dar el lujo de contratar una solución que además de solucionarles su necesidad de almacenar información en un manejador de bases de datos les permite conjuntarlo con una solución que les controle el acceso, control y manejo de la información administrando a su vez varias de las características de su negocio. Soluciones como el producto SAP[4] que usa la RDBMS de Oracle mantienen una conjunción de la lógica organizacional. Al contratar una solución adecuada con el ámbito del negocio les brinda la cobertura para controlar las transacciones de toda la empresa. Pero también las grandes empresas tienen opción de elegir un RDBMS como SQL Server, Informix, Sybase, Oracle y hasta PostgreSQL para realizar un proyecto de desarrollo para administrar tanto la información, como las actividades del negocio por lo que tienen las grandes empresas la facilidad de elegir cualquier base de datos de grandes prestaciones como lo son Sybase, Oracle o PostgreSQL.

Pequeñas y medianas empresas con megabytes o gigabytes de información. Estas empresas, dependiendo de su presupuesto pueden optar por elegir entre bases de datos por las que haya que pagar licencia de uso o las de software libre con poco o nulo costo para su utilización, siendo estas últimas las más recomendables. Entre las alternativas de software libre pueden estar PostgreSQL y MySQL, recomendando usar PostgreSQL.

---

[4] SAP (Systeme, Anwendungen und Produkte) (Sistemas, Aplicaciones y Productos), con sede en Walldorf (Alemania) comercializa Software empresarial, un conjunto de aplicaciones de software para soluciones integradas de negocios, entre ellas mySAP Business Suite, que provee soluciones escalables que permiten mejorar continuamente, con más de 1,000 procesos de negocio consideradas las mejores prácticas empresariales.

Microempresas con kilobytes o megabytes de información. Para las empresas establecidas que no pueden contratar programadores para mantener un sistema de información pueden optar por alguna solución como las de la Empresa Aspel[5] como sus productos estrella COI, NOI y SAE que explotan las bases de datos en Paradox.

Presencia en Internet con kilobytes o megabytes de información. Este tipo de solución aunque debiera integrarse con la información empresarial puede estar desvinculada de las grandes bases de datos, por lo que requiere una mínima porción de los requerimientos informáticos. Una base de datos como MySQL debiera ser suficiente para la mayoría de las necesidades, pero como la mayoría de bases de datos ya proporciona la conectividad para la publicación en Internet cualquier manejador de bases de datos cumple con los requisitos informáticos para ser elegida.

[5] Aspel de México es una empresa mexicana líder en el mercado de software administrativo que desarrolla y comercializa sistemas de cómputo y servicios relacionados que automatizan la administración de las micro, pequeñas y medianas empresas favoreciendo la correcta toma de decisiones, simplificando sus actividades administrativas e integrándolas fácilmente a la era digital.

Capítulo 3

SQL (STRUCTURED QUERY LANGUAGE)

Conocer y aplicar el lenguaje SQL para la definición, control y consulta de la información necesaria en el desarrollo de las aplicaciones en su entorno de trabajo. Utilizar el SQL para extraer la información necesaria en el desarrollo de las aplicaciones en su entorno de trabajo a través de la elaboración de las consultas que se puedan requerir y de la programación de los procedimientos a utilizar, así mismo, entenderá la ventaja de conocer las normas ANSI SQL vistas anteriormente y la sintaxis de las principales instrucciones del SQL dentro de la migración de consultas y procedimientos de un RDBMS a otro.

3.1 ALGEBRA RELACIONAL.

Conjunto de operaciones para manipular las tuplas de las relaciones o tablas. El resultado de cada operación es una nueva relación que podemos manipular posteriormente.

Operaciones

- Seleccionar (σ)
- Proyectar (π)
- Operaciones de Teoría de Conjuntos:
  - Unión (∪),
  - Intersección (∩),
  - Diferencia (–),
  - Producto Cartesiano (Ξ).
- Reunión (α)

Seleccionar (σ)

Por medio de esta operación se posibilita la selección de un subconjunto de tuplas de una relación que corresponden a una condición (columna OPERADOR valor) determinada. El grado (total de columnas de la Relación), se conserva.

Formato de Uso: σ **(condición) (RELACIÓN)**

Esta operación es conmutativa, es decir:

σ (condición1) (σ (condición2) (R)) = σ (condición2) (σ (condición1) (R))

Ejemplos:

PERSONA

Cedula	Nombre	Primer_Apellido	Segundo_Apellido	Sexo	Dirección	Teléfono	*Salario*
71134534	Juan	Mesa	Uribe	M	Cra 25 22-1	2567532	*1,600,000*
23423445	Ana María	Betancur	Bermudez	F	Cra 45 11-13	3433444	*1,300,000*
12453535	Gloria	Betancur	Garces	F	Tr. 12 43-5	2756533	*1,700,000*
75556743	Pedro	Ochoa	Pelaez	M	Cll.6ta 14-45	2686885	*1,200,000*
43533322	Patricia	Angel	Guzmán	F	Cll. 45 23-1	2674563	*1,350,000*
78900456	*Carlos*	*Betancur*	*Agudelo*	*M*	*Cir. 5 12-5*	*4445775*	*1,500,000*

La selección, permite extraer todas las filas (tuplas) que cumple una condición determinada. Esta condición permite la utilización de los operadores de comparación: **=**, **>**, **<**, **>=**, ≠, además de los conectores lógicos "**y**" – "**o**" :

a. **σ cedula = 71134534 (PERSONA)**

Resultado:

Cedula	Nombre	Primer_Apellido	Segundo_Apellido	Sexo	Dirección	Teléfono	*Salario*
71134534	*Juan*	*Mesa*	*Uribe*	*M*	*Cra 25 22-1*	*2567532*	*1,600,000*

b. **σ sexo ='F' (PERSONA)**

Resultado:

Cedula	Nombre	Primer_Apellido	Segundo_Apellido	Sexo	Dirección	Teléfono	*Salario*
23423445	Ana María	Betancur	Bermudez	F	Cra 45 11-13	3433444	*1,300,000*
12453535	*Gloria*	*Betancur*	*Garces*	*F*	*Tr. 12 43-5*	*2756533*	*1,700,000*

c. **σ (primer_apellido ='Betancur') y (sexo='F') (PERSONA)**

Resultado:

Cedula	Nombre	Primer_Apellido	Segundo_Apellido	Sexo	Dirección	Telefono	*Salario*
23423445	Ana María	Betancur	Bermudez	F	Cra 45 11-13	3433444	*1,300,000*
12453535	*Gloria*	*Betancur*	*Garces*	*F*	*Tr. 12 43-5*	*2756533*	*1,700,000*

d. σ **(sexo = 'M'') o (Salario >=1,350,000) (PERSONA)**

Resultado:

Cedula	Nombre	Primer_Apellido	Segundo_Apellido	Sexo	Dirección	Teléfono	*Salario*
71134534	Juan	Mesa	Uribe	M	Cra 25 22-1	2567532	*1,600,000*
12453535	Gloria	Betancur	Garces	F	Tr. 12 43-5	2756533	*1,700,000*
75556743	Pedro	Ochoa	Pelaez	M	Cll.6ta 14-45	2686885	*1,200,000*
43533322	Patricia	Angel	Guzmán	F	Cll. 45 23-1	2674563	*1,350,000*
78900456	*Carlos*	*Betancur*	*Agudelo*	*M*	*Cir. 5 12-5*	*4445775*	*1,500,000*

Proyectar (π)

Esta operación permite seleccionar algunas columnas de una relación.

Formato de Uso: π **<lista de Atributos> (RELACION)**

Ejemplos:

Se construyen con base en la Relación anterior: **PERSONA**.

a. π **cedula, nombre, primer_apellido, segundo_apellido (PERSONA)**

Resultado

Cedula	Nombre	Primer_Apellido	*Segundo_Apellido*
71134534	Juan	Mesa	*Uribe*
23423445	Ana María	Betancur	*Bermudez*
12453535	Gloria	Betancur	*Garces*
75556743	Pedro	Ochoa	*Pelaez*
43533322	Patricia	Angel	*Guzmán*
78900456	*Carlos*	*Betancur*	*Agudelo*

b. π **cedula, salario (PERSONA)**

Resultado:

Cedula	Salario
71134534	1,600,000
23423445	1,300,000
12453535	1,700,000
75556743	1,200,000
43533322	1,350,000
78900456	1,500,000

La operación **SELECCIÓN** combinada con la operación **PROYECCIÓN**, podemos tener:

c. π **cedula, nombre, salario (σ (sexo = 'M') o (Salario >=1,350,000) (PERSONA))**

Resultado:

Cedula	Nombre	*Salario*
71134534	Juan	*1,600,000*
12453535	Gloria	*1,700,000*
75556743	Pedro	*1,200,000*
43533322	Patricia	*1,350,000*
78900456	*Carlos*	*1,500,000*

El resultado de las operaciones pueden ser llevados a relaciones temporales de la siguiente forma:

REL_TEMP ← π cedula, nombre, salario (σ (sexo = 'M') o (Salario >=1,350,000) PERSONA))

Resultado:

REL_TEMP	Cedula	Nombre	*Salario*
	71134534	Juan	*1,600,000*
	12453535	Gloria	*1,700,000*
	75556743	Pedro	*1,200,000*
	43533322	Patricia	*1,350,000*
	78900456	*Carlos*	*1,500,000*

3.2 DEFINICIÓN DE DATOS.

SQL (Structured Query Language; Lenguaje Estructurado de Consulta) es un lenguaje de consulta para bases de datos, siendo adoptado como estándar de la industria en 1986. Desde entonces se han realizado revisiones al estándar para incorporar nueva funcionalidad conforme la industria de las bases de datos lo va requiriendo. Una de las revisiones más importantes fue la de 1992, conocida como ANSI SQL92.

La ventaja de la adopción del ANSI SQL, es que los diversos RDBMS (Relational DataBase Management System; Sistema Manejador de Bases de Datos Relacional) tienen que acoplarse al estándar, permitiendo así una mayor compatibilidad entre ellos. Esto implica que conociendo una variante del SQL, se tienen los conocimientos necesarios para poder utilizar otros RDBMS: Ms SQL Server, Oracle, Sybase, Interbase, MySQL, PostgreSQL, DB2, etc.

Aunque los distintos fabricantes tratan de acoplarse al estándar ANSI SQL, es cierto que cada uno implementa funcionalidades extra que le dan un valor agregado a su producto pero sacrificando un poco la compatibilidad, por lo cual se podrán notar ciertas diferencias entre distintos RDBMS.

SQL es un lenguaje fácil de entender ya que su estructura utiliza palabras en inglés, lo que lo hace fácil de aprender y utilizar y las instrucciones se enfocan a qué buscar, dejando al RDBMS la tarea de cómo recuperar la información.

Los componentes del RDBMS relacionados con el SQL son:

- DDL (Data Definition Language; Lenguaje de Definición de Datos). Permite crear, modificar y eliminar estructuras de datos como: tablas, bases de datos, índices, etc.
- DML (Data Manipulation Language; Lenguaje de Manipulación de Datos). Permite consultar, insertar, modificar y eliminar datos de las tablas.
- DCL (Data Control Language; Lenguaje de Control de Datos). Permite establecer los privilegios de acceso a los datos, en otras palabras, establece la seguridad de la base de datos.

Antes de comenzar a trabajar con SQL, es necesario conocer los elementos que intervienen en la definición de la información en una base de datos, para poder manipularla de manera adecuada. La tabla es el elemento fundamental de una base de datos relacional, la cual consiste de una serie de renglones (registros) que representan la información. Cada renglón está dividido en columnas (campos) los cuales deben de tener un tipo de dato establecido.

Tipos de datos del sistema

Cada columna dentro de una tabla debe tener asociado un tipo de dato, siendo la labor del diseñador de la base de datos, el de encontrar el mejor tipo de dato que satisfaga las necesidades de almacenamiento y recuperación de cierta información.

Los tipos de datos que se manejan en una base, pueden variar ligeramente entre diferentes RDBMS, sin embargo el estándar ANSI, asegura que cierto tipo de datos estará presente en cualquier RDBMS asegurando así la compatibilidad.

Algunos RDBMS implementan sinónimos para los tipos de datos, de manera que puedan cumplir con el ANSI SQL92, sin embargo internamente son convertidos a un tipo de dato que si esté soportado. Por ejemplo Ms SQL Server acepta el tipo de dato DOUBLE PRECISION pero lo convierte y maneja como un FLOAT.

Tipo en SQL92/SQL3	**MySQL**	**PostgreSQL**	**Oracle**	**Sybase**	**Ms SQL Server**	***Descripción***
Tinyint	tinyint			tinyint	tinyint	
int, integer	int, integer	int4	int (lo convierte a number)	int	int	*Entero con signo de 4 bytes*
Smallint	smallint	int2	Smallint (lo convierte a number)	smallint	smallint	*Entero con signo de 2 bytes*
numeric(p,d)	numeric(p,d)	numeric(p,d)	number(p,d)	numeric(p,s)	decimal(p,s)	*Numérico con precisión p y d decimales*
float()	float	float4, float8	float()	float	float	*Número de punto flotante*
Real			Real (lo convierte a float)	real	real	*Número Real*

Double			double precision (lo convierte a float)	double precision	double precision (se convierte en float)	*Número Doble*
character varying(n)	varchar(n)	varchar(n)	varchar2(n)	varchar(n)	varchar(n)	*Carácter de longitud variable*
Bolean	bit	bool		bit	bit	*Valor booleano*
Date	date	date				*Fecha sin hora del día*
Time	time	time				*Hora del día*
timestamp	timestamp	timestamp	date	datetime	datetime	*Fecha y hora del día*
char, character(n)	char(n)	char(n)	char(n)	char(n)	char(n)	*Cadena de caracteres de longitud fija*
interval		interval				*Tiempo de intervalo*
Blob	blob		blob	image	image	*Binary large object*
clob	*text*	*text*	*clob*	*text*	*text*	*Character large object*

Tabla 3-1. Comparativa de los tipos de datos entre diversos RDBMS y el estándar SQL 92.

El valor nulo

Es importante conocer el concepto de valor nulo, en el contexto de una base de datos, debido a que frecuentemente un valor nulo es confundido con un valor numérico de 0 o una cadena vacía. Un valor nulo se representa en SQL con la cláusula **`NULL`** y representa la ausencia de información.

Por ejemplo si en un listado de empleados algunos aparecen en el dato de comisión como **`NULL`**, esto indica que no se tiene dicha información aunque erróneamente se podría interpretar que estos empleados tienen comisión del 0%. Ahora, suponiendo que los datos que aparecen como **`NULL`** corresponden a la edad, indican que el dato no fue proporcionado y por lo tanto se carece de dicha información y claramente se observa que no es lo mismo que una edad de 0.

Esto es importante recordarlo cuando se deseen realizar ciertas operaciones, por ejemplo un promedio de edades, dado que la función para determinar el promedio no contemplará valores nulos.

```
Edad 1        25
Edad 2        NULL
Edad 3        30
Edad 4        NULL
Promedio:     27.5
```

En el ejemplo anterior se puede confirmar que los valores nulos no son considerados como cero.

Hasta este momento se ha hablado únicamente de tipos de datos numéricos, aunque los valores nulos también se utilizan en cualquier otro tipo de dato, por ejemplo en texto: No es lo mismo una cadena vacía que un valor nulo.

En Oracle cuando uno trata de insertar una cadena de texto vacía, automáticamente lo convierte a valor **NULL**. Esta característica particular, no es compatible con el estándar y Oracle podría en un futuro cambiar este comportamiento.

Tablas

Las tablas son el elemento fundamental que compone a una base de datos relacional porque todo gira en torno a ellas. Las tablas son estructuras de almacenamiento que albergan la información en forma de registros (renglones) que deben de ser identificados de manera única y esto se logra a través de una llave primaria.

La tabla es la representación física en la base de datos de una Entidad mientras que las relaciones son representadas mediante restricciones.

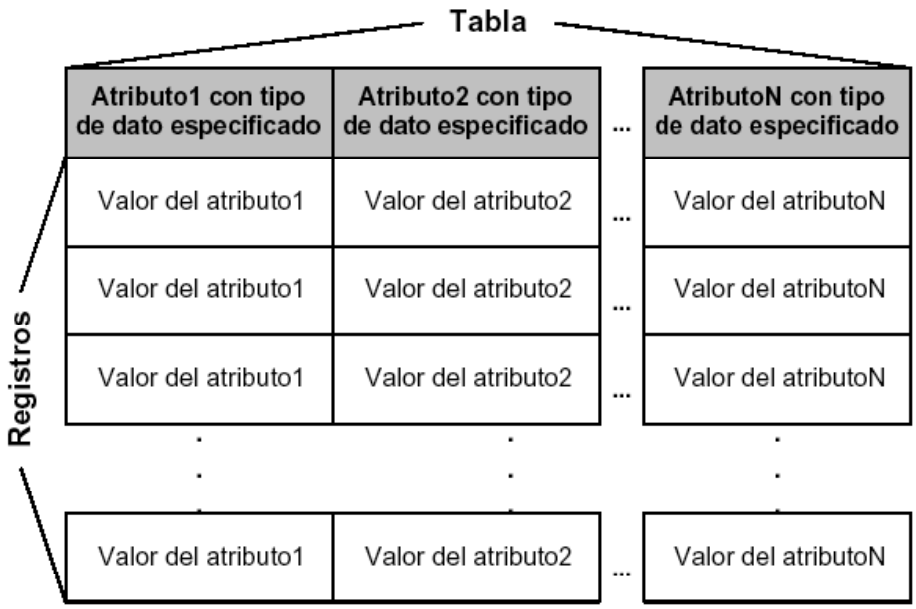


Ilustración 3-1. Representación de una Tabla en una RDBMS.

Convención de nombres

Los nombres que se le asignan a los objetos de una base de datos, están sujetos a ciertas reglas que varían entre RDBMS, incluso de una versión a otra del mismo. A continuación se listan algunas de las reglas que deben observarse:

- Deben empezar con una letra. (En Sybase y Oracle)
- La longitud del nombre de la tabla y columna depende del RDBMS.
- Pueden contener letras: **A**-**Z**, **a**-**z**, números: **0**-**9**, _, **#**, **$**.
- No pueden usarse palabras reservadas del servidor SQL.
- No pueden tener el mismo nombre que otro objeto que pertenezca al usuario que las creo.

- Es aconsejable usar nombres descriptivos.

Creación de tablas

La sintaxis básica para la creación de una tabla es la siguiente:

```
CREATE TABLE <nombre_tabla>
(
       <nombre_campo1> <tipo_de_dato> [NULL | NOT NULL]
       <nombre_campo2> <tipo_de_dato> [NULL | NOT NULL]
       <nombre_campo3> <tipo_de_dato> [NULL | NOT NULL]…
       <nombre_campoN> <tipo_de_dato> [NULL | NOT NULL]
)[DEFAULT <val_predeterminado>],
 [DEFAULT <val_predeterminado>],
 [DEFAULT <val_predeterminado>],
 [DEFAULT <val_predeterminado>]
```

Código 3-1. Sintaxis básica de creación de tablas.

Ejemplos:

```
CREATE TABLE pais
(
       id_pais numeric(3) NOT NULL,
       nombre varchar(100) NOT NULL
)
```

Código 3-2. Creación de una tabla con dos campos.

```
CREATE TABLE marcador
(
       id_pais1 numeric(3) NOT NULL,
       id_pais2 numeric(3) NOT NULL,
       marcador varchar(40) NOT NULL
)
```

Código 3-3. Creación de una tabla con tres campos.

Eliminación de tablas

Para eliminar una tabla y los datos que contiene, así como sus índices, triggers y permisos se utiliza la siguiente instrucción:

```
DROP TABLE <nombre_tabla>
```

Código 3-4. Sintaxis para la eliminación de tablas.

Ejemplo:

```
DROP TABLE empleados
```

Código 3-5. Eliminación de una tabla.

Modificación de tablas

Dependiendo del RDBMS esta modificación de estructura de la tabla ofrece mayor o menor flexibilidad. Por ejemplo, MySQL es de las más flexibles, porque permite cambiar incluso el tipo de dato de una columna mientras que en Sybase o Ms SQL Server es necesario eliminar y volver a crear la tabla con la estructura deseada.

Características soportadas:	MySQL	PostgreSQL	Oracle	Sybase	MS SQL Server
Agregar campo	Si	Si	Si	Si	Si
Eliminar campo	Si	No	Si	No	No
Renombrar Campo	Si	Si	No	No	No
Cambiar el tipo de dato del campo	Si	No	Si	No	No
Definir llave primaria	Si	Si	Si	Si	Si
Definir llave Foránea	Si	Si	Si	Si	Si
Renombrar tabla	Si	Si	Si	No (pero brinda un procedimiento almacenado)	No (pero brinda un procedimiento almacenado)

Tabla 3-2. Características soportadas por diversos RDBMS.

La sintaxis varía de un RDBMS, debido a las diferentes características soportadas, sin embargo todos ellos utilizan la cláusula **ALTER TABLE** para realizar las modificaciones posibles, a una tabla.

La sintaxis general para agregar un campo es la siguiente:

```
ALTER TABLE <nombre_tabla>
ADD <nombre_campo> <tipo_dato> [NOT NULL | NULL] [default <valor_predeterminado>]
```

Código 3-6. Sintaxis para modificar la estructura de una tabla.

Cabe señalar que siempre que se agregue un campo a una tabla, éste debe permitir valores NULOS.

La sintaxis para definir una llave primaria o foránea utilizando la instrucción **ALTER TABLE**, se presentan más adelante en este capítulo.

Reglas

Las reglas dentro de la base de datos, permiten definir condiciones que debe cumplir la información para que sea válida. Por ejemplo se puede definir una regla que especifique que la percepción de un empleado no sobrepase los $ 10,000 pesos.

En la práctica, muchas de estas reglas no se definen en la base de datos sino mediante la lógica de una aplicación desarrollada en algún lenguaje, que tiene acceso a la información, ya que el exceso de reglas puede disminuir el rendimiento del RDBMS en los procesos de inserción y modificación de información.

En Sybase y Ms SQL Server las reglas se pueden crear como objetos independientes que se vinculan a distintas tablas. En cambio en Oracle y PostgreSQL no son objetos independientes y sólo pueden ser definidas como restricciones que afecta a una sola tabla.

	MySQL	PostgreSQL	Oracle	Sybase	*MS SQL Server*
Soporta definición de reglas	*No*	*Si*	*Si*	*Si*	*Si*

Tabla 3-3. Soporte en la definición de reglas de distintos RDBMS.

Sintaxis para Sybase y Ms SQL Server:

```
CREATE RULE <nombre_regla>
AS <condicion>
```

Código 3-7. Sintaxis para crear reglas en Sybase y Ms SQL Server.

Ejemplo:

```
CREATE RULE regla_salario
AS @sueldo_empleado <= 10000
sp_bindrule regla_salario, 'empleado.sueldo_empleado'
```

Código 3-8. Ejemplo para crear reglas en una tabla.

Sintaxis para Oracle y PostgreSQL:

```
ALTER TABLE <nombre_tabla>
ADD CONSTRAINT <nombre_restriccion>CHECK <condicion>
```

Código 3-9. Ejemplo para crear reglas en una tabla usando Oracle y PostgreSQL.

Defaults

Los defaults establecen que valor será registrado de manera predeterminada para una columna, en caso de que no se especifique al momento de introducir los datos. En algunos RDBMS los defaults son objetos que se pueden emplear en diferentes tablas, mientras que en los demás, están ligados a la definición de la tabla. Al igual que con las reglas, la funcionalidad de los defaults pueden implementarse utilizando la lógica de la aplicación.

Nuevamente en Sybase y Ms SQL Server es posible definir un **DEFAULT** como un objeto independiente que se puede vincular a varios campos de una o más tablas, mientras que en Oracle, MySQL y PostgreSQL los defaults están ligados a un solo campo.

	MySQL	PostgreSQL	Oracle	Sybase	MS SQL Server
Soporta definición de defaults	Si	Si	Si	Si	Si

Tabla 3-4. Soporte de definición de defaults en diversas RDBMS.

Sintaxis en Sybase/Ms SQL Server para crear un **default** como objeto de la base de datos.

```
CREATE DEFAULT <nombre_default> AS <expresión_constante>
```

Código 3-10. Sintaxis para crear defauls usando Sybase / Ms SQL Server.

Ejemplo:

```
CREATE DEFAULT estado_predeterminado AS "Distrito Federal"
```

Código 3-11. Ejemplo para crear valores por defecto.

Una vez creado el **`default`**, con el procedimiento almacenado del sistema ligamos o unimos el **`default`** a una columna.

```
sp_binddefault <nombre_default> <Nombre_tabla.columna>
```

Código 3-12. Unión del procedimiento almacenado del default con el campo en la tabla.

Ejemplo:

```
sp_binddefault estado_predeterminado empleado.estado
```

Código. Ejemplo de la unión de un procedimiento almacenado del default con el campo de una tabla.

En todos los RDBMS, los valores predeterminados se pueden especificar al momento de generar una tabla, o en algunos casos es posible modificar la tabla para agregar estos valores predeterminados:

Ejemplo.

```
Create table cliente
(
        id_cliente numeric(10) not null primary key,
        fecha_afiliacion date default sysdate
)
```

Código 3-13. Creación de un valor por defecto al crear una tabla.

Nota: Este ejemplo es específico para Oracle. Para que funcione en PostgreSQL y MySQL es necesario cambiar **`sysdate`** (que representa la fecha y hora actual) por **`current_date`**.

Llaves e índices

Un índice es una estructura de almacenamiento físico que permiten recuperar datos de una manera muy eficiente.

En un esquema relacional, cada registro dentro de una tabla debe de ser identificado de manera única, y esto se logra a través de una llave primaria, a la cual se le genera de manera automática un índice, que ayuda a ser más eficiente el proceso de consulta de la información. También existen las llaves foráneas, que son las columnas que hacen referencia a la llave primaria de otra tabla. A través de ellas se establecen relaciones entre tablas.

Es un error frecuente confundir entre índices y llaves, quizá sea el hecho de que al crear una llave primaria se genera un índice, lo cual no sucede para una llave foránea. Hay que tener presente que los índices tienen como función acelerar el proceso de recuperación de la información.

Llaves primarias.

Una llave primaria permite identificar de manera única un registro dentro de una tabla. Al crear una llave de este tipo se genera automáticamente un índice de valores únicos, por lo que los valores de los campos que involucra la llave primaria no se pueden repetir ni ser nulos. Existen varias formas de declarar una llave primaria:

1. Al crear una tabla se puede especificar que campo o campos forman parte de la llave primaria.

   Ejemplo:

```
CREATE TABLE pais
(
        id_pais numeric(3) NOT NULL PRIMARY KEY,
        nombre varchar(100)
)
```

Código 3-14. Asignación de una llave primaria al crear una tabla.

   En el ejemplo anterior, la tabla solamente consta de un campo que forma la llave primaria "**id_pais**", por ello es posible utilizar la cláusula **PRIMARY KEY**. Pero cuando la llave primaria consta de dos o más campos la sintaxis anterior no es útil, por lo cual se emplea la siguiente sintaxis:

```
CREATE TABLE pedido
(
        id_cliente numeric(10) NOT NULL,
        id_producto numeric(10) NOT NULL,
        fecha date,
        PRIMARY KEY (id_cliente, id_producto)
)
```

Código 3-15. Asignación de una llave primaria compuesta de dos campos al crear una tabla.

   En el ejemplo anterior, se puede observar que se ocupa la cláusula **PRIMARY KEY**, como si fuera un atributo más de la tabla, especificando entre paréntesis todos los campos que formen parte de la llave primaria.

2. Si la tabla ya existe y se desea especificar los campos que forman parte de la llave primaria se emplea la cláusula **ALTER TABLE**.

   Ejemplo:

```
ALTER TABLE pedido
ADD PRIMARY KEY (id_cliente, id_producto)
```

Código 3-16. Modificar una tabla existente para asignarle una llave primaria compuesta de dos campos.

Llaves foráneas.

Las llaves foráneas son atributos de una tabla que hacen referencia a la llave primaria de otra tabla. Estas llaves foráneas permiten establecer relaciones entre las distintas tablas que existen dentro de la base de datos. La mayoría de los RDBMS implementan restricciones (constraints) cuando se genera una llave foránea, de este modo asegura la integridad de la información almacenada en la base de datos.

MySQL no implementa estas restricciones, por lo que se puede llegar a dar una falta de integridad en la información, sin embargo esto no es del todo malo porque las restricciones se pueden

implementar con lógica de aplicación en caso de que exista y porque la falta de implementación de estas restricciones, permite que MySQL trabaje más rápido que cualquier otro RDBMS. Dependiendo del tipo de información y de aplicación a desarrollar se puede sacrificar la característica a cambio de rapidez en respuesta.

Las llaves foráneas se pueden crear dentro de la definición de la tabla, o una vez que esta ya existe, se puede utilizar la cláusula **ALTER TABLE** para agregar esta restricción. A diferencia de las llaves primarias, las llaves foráneas no generan un índice, por lo que de ser necesario se deberá crear con la cláusula **CREATE INDEX**.

Ejemplo al momento de crear la tabla:

```
CREATE TABLE Marcador (
        id_pais1 NUMBER(3) NOT NULL,
        id_pais2 NUMBER(3) NOT NULL,
        resultado VARCHAR2(15) NOT NULL,
        FOREIGN KEY (id_pais2) REFERENCES Pais,
        FOREIGN KEY (id_pais1) REFERENCES Pais
);
```

Código 3-17. Crear dos llaves foráneas al crear una tabla.

A través de un **ALTER TABLE**:

```
ALTER TABLE Marcador
ADD ( FOREIGN KEY (id_pais1) REFERENCES Pais ) ;
```

Código 3-18. Crear una llave foránea en una tabla existente.

Índices.

Un índice es una estructura de almacenamiento físico que ocupa un espacio. Los índices ayudan al Servidor SQL a localizar datos y son transparentes para el usuario. El propósito principal de un índice es proporcionar un acceso más rápido a los datos, aunque en algunos casos su propósito es asegurar que el contenido de un campo sea único.

Creación de índices

Para crear índices se utiliza el siguiente comando:

```
CREATE [UN IQUE] INDEX <nombre_índice> ON <nombre_tabla>(<campo>,...)
```

Código 3-19. Sintaxis para la creación de un índice en una tabla.

Ejemplo.

```
CREATE INDEX idx_empleado ON empleado(fecha_contratacion);
```

Código 3-20. Creación de un índice en una tabla.

Abusar del empleo de índices puede llevar a que se degrade el tiempo de respuesta del servidor en lugar de mejorarlo, esto se debe a que en operaciones que involucran inserción, modificación o eliminación de datos, los índices deben de ser actualizados lo cual puede consumir un tiempo considerable. Por lo general se suele crear un índice únicamente cuando una columna es empleada frecuentemente en una cláusula **WHERE** y la tabla tiene un tamaño considerable.

Eliminación de índices

Para eliminar un índice se utiliza la siguiente instrucción:

```
DROP INDEX <nombre_índice>
```

Código 3-21. Sintaxis para eliminar un índice.

Ejemplo.

```
DROP INDEX idx_empleado
```

Código 3-22. Ejemplo para eliminar de un índice.

3.3. MANIPULACIÓN DE DATOS.

La mayor parte del trabajo con SQL girará entorno a cuatro comandos:

- **`Select`**. Permite seleccionar (recuperar) información de una tabla.
- **`Insert`**. Permite agregar información a una tabla.
- **`Delete`**. Permite eliminar información de una tabla.
- **`Update`**. Permite actualizar información que existe en una tabla.

Para el empleo de cualquiera de los comandos mencionados es indispensable tomar en cuenta dos puntos:

- Para expresar un valor de tipo alfanumérico o fecha, es requisito entrecomillarlo con comillas simples.
- Todo valor que no se especifique entre comillas simples, será interpretado como tipo de dato numérico.

Selección, proyección y join.

SQL se basa en el álgebra relacional, por ello es importante conocer las operaciones del álgebra relacional y su relación con SQL.

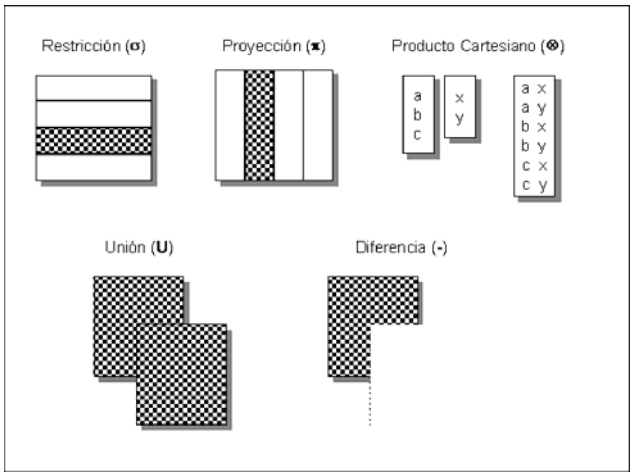


Ilustración 3-2. Representación gráfica de las operaciones del álgebra relacional: restricción, proyección, producto cartesiano, unión y diferencia.

Selección (Restricción).

La operación de selección genera un subconjunto de los renglones de una tabla, con base en un criterio (restricción) establecido.

Esta operación del álgebra relacional es realizada por la cláusula **WHERE** de SQL.

Id_ empleado	nombre_ empleado	edad_ empleado	sexo_ empleado	sueldo_ empleado	porcentaje_ comision	fecha_ contratacion	Id_ departamento	Id_ cargo
1	Gilberto Aparicio	25	H	7000	NULL	25/11/2000	4	1
2	Javier González	30	H	7500	NULL	25/11/2000	4	1
3	Alma Benítez	19	M	10000	NULL	01/01/2001	4	1
4	Carolina Fernández	32	M	15000	NULL	01/03/2001	4	2
5	Alberto Ibáñez	32	H	25000	NULL	04/05/2001	4	2
6	Carlos López	32	H	27000	NULL	04/05/2001	4	3
7	Arturo Rangel	45	H	18000	5	25/10/2001	1	4
8	Javier Bárcenas	29	H	18000	10	25/10/2001	1	4
9	Ernesto Almaraz	32	H	20000	NULL	07/01/2002	1	2
10	Susana García	31	M	18000	NULL	07/01/2002	3	1
11	Fernanda Ramírez	18	M	18000	NULL	15/05/2002	3	1
12	Roxana Guerrero	25	M	26500	NULL	15/07/2002	3	2
13	Manuel Alavez	35	H	12600	NULL	23/09/2002	2	2
14	Angeles Nolasco	29	M	6000	NULL	23/09/2002	5	1

15	Teresa Barba	37	M	12000	NULL	23/09/2002	5	2
16	Alberto González	56	H	35000	NULL	27/10/2002		5
17	Ricardo Contreras	58	H	50500	NULL	27/10/2002		6

Tabla 3-5. Información de una tabla con los datos de los empleados de una empresa.

Teniendo el conjunto de registros anteriores, un ejemplo de selección sería mostrar únicamente aquellos que sean del sexo femenino:

Id_ empleado	nombre_ empleado	edad_ empleado	sexo_ empleado	sueldo_ empleado	porcentaje_ comision	fecha_ contratacion	Id_ departamento	Id_ cargo
3	Alma Benítez	19	M	10000	NULL	01/01/2001	4	1
4	Carolina Fernández	32	M	15000	NULL	01/03/2001	4	2
10	Susana García	31	M	18000	NULL	07/01/2002	3	1
11	Fernanda Ramírez	18	M	18000	NULL	15/05/2002	3	1
12	Roxana Guerrero	25	M	26500	NULL	15/07/2002	3	2
14	Angeles Nolasco	29	M	6000	NULL	23/09/2002	5	1
15	Teresa Barba	37	M	12000	NULL	23/09/2002	5	2

Tabla 3-6. Información de los datos de los empleados de una empresa del sexo femenino.

Proyección.

La proyección selecciona y genera un subconjunto con los atributos (columnas) indicados de una tabla. También es conocida como operación vertical.

Esta operación es realizada por la cláusula **SELECT** del SQL.

Ejemplo:

Mostrar clave, nombre y sueldo del empleado

Id_empleado	nombre_ empleado	sueldo_empleado
1	Gilberto Aparicio	7000
2	Javier González	7500
3	Alma Benítez	10000
4	Carolina Fernández	15000
5	Alberto Ibáñez	25000
6	Carlos López	27000
7	Arturo Rangel	18000
8	Javier Bárcenas	18000
9	Ernesto Almaraz	20000
10	Susana García	18000
11	Fernanda Ramírez	18000
12	Roxana Guerrero	26500
13	Manuel Alavez	12600
14	Angeles Nolasco	6000
15	Teresa Barba	12000
16	Alberto González	35000
17	Ricardo Contreras	50500

Tabla 3-7. Proyección de la tabla empleados obteniendo sólo 3 campos de 9 que tiene la tabla.

Unión.

La operación unión realiza la misma acción que en el álgebra de conjuntos, es decir **{1,4,5,10} U {1,4,3,9} = {1,3,4,5,9,10}.**

Esta operación se realiza con la cláusula **UNION** del SQL.

Producto Cartesiano

El producto cartesiano es el producto cruz entre 2 tablas:

{a,b} X {1,2} = {a1, a2, b1, b2}

El resultado es la combinación de cada renglón de una tabla con cada renglón de la otra tabla.

En SQL esta operación se lleva a cabo cuando se ocupa la cláusula **FROM** especificando 2 o más tablas, y no se especifica una restricción que indique la relación entre las tablas.

Join

La operación Join es en esencia un producto cartesiano, donde se selecciona los renglones que satisfagan las condiciones indicadas que establecen la relación entre las tablas involucradas. Esta es una operación muy común en las bases de datos relacionales.

Selección de datos.

Las tablas dentro de una base de datos son las estructuras que tienen almacenada la información en forma de registros. Para poder recuperar esa información almacenada, se requiere del comando **SELECT** de SQL.

El comando **SELECT** es sumamente útil, ya que a través de él es posible realizar desde una consulta simple que sólo involucra una tabla, hasta una consulta compleja donde intervienen dos o más tablas, varias condiciones, agrupaciones de datos y ordenamientos.

```
SELECT {* | [DISTINCT] <campo>, <campo> ... }
FROM <nombre_tabla>, [ <nombre_tabla> ]
[WHERE <condición> ]
[GROUP BY <campo>, <campo>,... ]
[HAVING <condición> ]
[ORDER BY <campo>, <campo>,.... ]
```

Código 3-23. Sintaxis del comando SELECT.

Lo que se puede apreciar en la estructura de la instrucción **SELECT**, es que nunca debe faltar ni la palabra **SELECT**, ni **FROM**. Todos los demás elementos son opcionales.

Cláusula SELECT

Esta cláusula indica que la instrucción a ejecutar es una consulta a la base de datos. **SELECT** nos permite indicar el nombre de los campos que queremos mostrar en la consulta. En caso de querer mostrar todos los campos de una tabla se emplea el comodín asterisco: *.

Cuando se realiza una consulta que involucra dos o más tablas, al nombre de cada campo se le antepone el de la tabla a la que pertenece.

```
<nombre _tabla> .<nombre_campo>
```

Código 3-24. Invocación de un campo en una consulta que involucra dos o más tablas.

Es posible utilizar seudónimos (alias) para cambiar el nombre de las columnas mostradas en una consulta, esto puede servir para hacer más legible los resultados mostrados, o porque así lo requiere alguna aplicación. Para colocar los seudónimos es necesario especificarlo mediante la cláusula **AS**, de la siguiente forma: **<nombre_campo> AS <otro_nombre>**.

Ejemplos:

Nota: los siguientes ejemplos son sentencias completas de SQL, donde el texto resaltado corresponde a la ejemplificación del concepto en cuestión.

```
SELECT empleado.nombre_empleado, departamento.nombre_departamento
FROM empleado, departamento
WHERE departamento.id_departamento = empleado.id_departamento
```

Código 3-25. Seleccionar el nombre del empleado de la tabla empleado y el nombre de departamento de la tabla departamento.

```
SELECT * FROM empleado
```

Código 3-26. Seleccionar todos los campos de la tabla empleado.

```
SELECT empleado.*, departamento.nombre_departamento
FROM empleado, departamento
WHERE departamento.id_departamento = empleado.id_departamento
```

Código 3-27. Seleccionar todos los campos de la tabla empleado y el nombre del departamento de la tabla departamento.

Cláusula FROM.

La cláusula **FROM** sirve para indicar las tablas de las cuales se desea mostrar la información. Cuando una consulta involucra dos o más tablas, es indispensable establecer las relaciones que existen entre ellas (join) mediante una cláusula **WHERE**.

Ejemplos:

1.

```
SELECT empleado.nombre_empleado, departamento.nombre_departamento
FROM empleado, departamento
WHERE departamento.id_departamento = empleado.id_departamento
```

Código 3-28. Seleccionar el nombre del empleado de la tabla empleado y el nombre de departamento de la tabla departamento.

2.

```
SELECT *
FROM empleado
```

Código 3-29. Seleccionar todos los campos de la tabla empleado.

3.

```
SELECT empleado.*, departamento.nombre_departamento
FROM empleado, departamento
WHERE departamento.id_departamento = empleado.id_departamento
```

Código 3-30. Seleccionar todos los campos de la tabla empleado y el nombre del departamento de la tabla departamento.

Alias para tablas.

Cuando una consulta involucra más de una tabla, es necesario especificar de cual tabla es el campo que se desea mostrar. Esto lo podemos ver en el ejemplo 3, que involucra a las tablas empleado y departamento. Es frecuente el uso de seudónimos para las tablas, con la finalidad de simplificar la escritura de la consulta. El seudónimo se coloca inmediatamente después del nombre de la tabla.

```
<tabla> [seudónimo]
```

Código 3-31. Sintaxis para asignarle un seudónimo a una tabla.

El ejemplo 3 del Código 3-30 reescrito con seudónimos quedaría de la siguiente manera:

```
SELECT e.*, d.nombre_departamento
FROM empleado e, departamento d
WHERE d.id_departamento = e.id_departamento
```

Código 3-32. Uso de alias en tablas.

Cláusula WHERE.

La cláusula **Where** permite delimitar los registros que serán mostrados en la consulta, a través de criterios o condiciones. Es posible utilizar los operadores lógicos: **OR**, **AND** y **NOT** para combinar expresiones y refinar el criterio de consulta.

Para escribir condiciones de manera adecuada es muy importante recordar que:

- Para expresar un valor de tipo alfanumérico o fecha, es requisito encerarlo con comillas simples.

- Todo valor que no se especifique entre comillas simples, será interpretado como tipo de dato numérico.

El valor **NULL** es un valor especial por lo cual se debe tener sumo cuidado cuando se desee utilizar condiciones con **NULL**. La única forma de comparar contra un valor **NULL** es utilizar el operador **IS** o **IS NOT**.

```
SELECT * FROM empleado WHERE comision = NULL; --incorrecto
SELECT * FROM empleado WHERE comision IS NULL; --correcto
```

Código 3-33. Uso correcto e incorrecto del valor NULL en la cláusula WHERE.

Igualdad	=	empleado.id_departamento = departamento.id_departamento
Desigualdad	!=	nombre_cargo != 'Director'
Mayor que	>	sueldo > 15000
Menor que	<	Edad < 35
Mayor o igual que	>=	sueldo >=15000
Menor o igual que	<=	Edad <= 35
Similar a	LIKE	Nombre_empleado like 'Al%' (% es un comodín)
Es	IS	Edad IS NULL
No es	IS NOT	Edad IS NOT NULL

Tabla 3-8. Las expresiones más frecuentes son las que involucran una comparación entre dos elementos.

En SQL es posible abreviar la forma de escribir ciertas condiciones:

La expresión:	**Se simplifica usando:**	**Quedando de la siguiente manera:**
sueldo >= 10000 AND sueldo <=15000	BETWEEN	Sueldo BETWEEN 10000 and 15000
nombre_cargo = 'Gerente' OR nombre_cargo = 'Presidente' OR nombre_cargo = 'Vicepresidente' OR nombre_cargo = 'Director'	IN	nombre_cargo IN ('Gerente', 'Presidente', 'Vicepresidente', 'Director')
nombre_cargo != 'Jefe de departamento' AND nombre_cargo != 'Vendedor'	NOT IN	nombre_cargo NOT IN ('Jefe de departamento', 'Vendedor')

Tabla 3-9. Simplificación de expresiones SQL.

La comparación de similitud que se hace mediante el uso de la cláusula **LIKE**, requiere de incluir comodines, que sustituyan uno o varios caracteres.

- **%** representa 0 o más caracteres.
- _ representa 1 carácter

Ejemplos. Finalmente complementando los 3 ejemplos anteriores quedarían de la siguiente manera:

```
SELECT empleado.nombre_empleado, departamento.nombre_departamento
FROM empleado, departamento
WHERE departamento.id_departamento = empleado.id_departamento
```

Código 3-34. Seleccionar el nombre del empleado de la tabla empleado y el nombre de departamento de la tabla departamento.

```
SELECT * FROM empleado
```

Código 3-35. Seleccionar todos los campos de la tabla empleado. (En este caso no es necesario colocar ninguna condición adicional por lo que queda sin modificación)

```
SELECT empleado.*, departamento.nombre_departamento
FROM empleado, departamento
WHERE departamento.id_departamento = empleado.id_departamento
```

Código 3-36. Seleccionar todos los campos de la tabla empleado y el nombre del departamento de la tabla departamento.

Cláusula GROUP BY

En la cláusula **GROUP BY** se indica el o los campos por los cuales se desea agrupar un conjunto de registros. Comúnmente esta agrupación va acompañada con una serie de funciones que realizan ciertas operaciones sobre el valor de los campos indicados. Estas funciones son conocidas como funciones de agregación:

Función	Acción
Count(*)	Regresa el número de registros encontrados
Count(<*campo*>)	Regresa el número de registros cuyo valor del campo especificado no es nulo
Sum(<*campo*>)	Suma los valores de la columna especificada
Avg(<*campo*>)	Promedia los valores del campo especificado
Min(<*campo*>)	Regresa el valor mínimo del campo especificado
Max(<*campo*>)	Regresa el valor máximo del campo especificado

Tabla 3-10. Acciones de las funciones de agregación.

Ejemplos:

```
SELECT id_departamento, SUM(sueldo) FROM empleado
GROUP BY id_departamento
```

Código 3-37. Mostrar la clave de departamento y el dinero total empleado para pagar a los empleados de dicho departamento.

```
SELECT id_departamento, AVG(edad) FROM empleado
GROUP BY id_departamento
```

Código 3-38. Mostrar la edad promedio por cada clave de departamento.

```
SELECT d.nombre_departamento, SUM(e.sueldo_empleado)
FROM empleado e, departamento d
WHERE e.id_departamento = d.id_departamento
GROUP BY d.nombre_departamento
```

Código 3-39. Mostrar el nombre de departamento y el dinero total empleado para pagar a los empleados de dicho departamento.

Cláusula HAVING

Esta cláusula es el equivalente a la cláusula **WHERE**, es decir, especifica un criterio o condición, pero la diferencia radica en que se ocupa únicamente cuando se desea especificar una función de agregación en la condición.

Ejemplos:

Mostrar la clave de departamento y sueldo promedio de sus empleados, de aquellos departamentos cuyo salario promedio sea mayor que 12000

```
SELECT id_departamento, AVG(sueldo_empleado)
FROM empleado
WHERE AVG(sueldo_empleado)>12000 --Incorrecto
GROUP BY id_departamento
```

Código 3-40. Uso incorrecto de la cláusula WHERE.

Es incorrecto usar la cláusula **WHERE** junto con una función de agregación, la consulta anterior **es incorrecta si se escribe**. Debiendo de escribir como sigue:

```
SELECT id_departamento, AVG(sueldo_empleado)
FROM empleado
GROUP BY id_departamento
HAVING AVG(sueldo_empleado)>12000
```

Código 3-41. Uso de la cláusula HAVING.

Cláusula ORDER BY

Esta cláusula nos permite indicar los campos por los cuales se desea ordenar la información mostrada. Es posible indicar si el orden es descendente o ascendente, de manera predeterminada es ascendente. Algunos RDBMS permiten realizar este ordenamiento especificando, en lugar del nombre del campo, la posición de este.

Ejemplos:

```
SELECT *
FROM empleado
ORDER BY nombre_empleado DESC
```

Código 3-42. Mostrar todos los datos de los empleados ordenados por nombre de manera descendente.

```
SELECT id_empleado, nombre_empleado, sueldo_empleado
FROM empleado
ORDER BY 3
```

Código 3-43. Mostrar clave, nombre del empleado y salario ordenados por salario.

Inserción de datos.

A través de la instrucción **INSERT** de SQL, se introduce la información a una tabla. La estructura general de este comando es la siguiente:

```
INSERT INTO <tabla> [(<nombreCampo1>, <nombreCampo2>, <nombreCampo3> ...)]
{
       VALUES (<valorCampo1>, <valorCampo2>, <valorCampo3> ... ) | <Expresión select>
}
```

Código 3-44. Estructura del comando INSERT.

Cláusula INSERT.

Esta cláusula permite indicar que la operación a realizar es la inserción de un registro.

Cláusula INTO.

Esta cláusula permite indicar la tabla en donde se realizará dicha inserción. Únicamente se puede especificar una tabla a la vez. Después del nombre de la tabla puede o no ir el nombre de los campos donde se va insertar información, esto es opcional, pero es muy recomendable no omitirlos, por que le resta legibilidad a la instrucción.

Si no se especifica el nombre de los campos que se van a insertar, el RDBMS identifica que se desea insertar información en cada uno de los campos, en el orden definido por la estructura de la tabla.

Por ejemplo, tomando a la tabla empleado cuya estructura define 9 campos con el siguiente orden: id_empleado, nombre_empleado, edad_empleado, sexo_empleado, sueldo_empleado, porcentaje_comision, fecha_contratación, id_departamento, id_cargo.

```
INSERT INTO empleado      Es equivalente a escribir:       INSERT INTO empleado(
                                                               id_empleado,
                                                               nombre_empleado,
                                                               edad_empleado,
                                                               sexo_empleado,
                                                               sueldo_empleado,
                                                               porcentaje_comision,
                                                               fecha_contratación,
                                                               id_departamento,
                                                               id_cargo
                                                           )
```

Código 3-45. Uso de la cláusula INSERT INTO.

Cláusula VALUES.

Esta cláusula permite especificar los valores a insertar para cada uno de los campos involucrados en la sentencia.

Para especificar los valores a insertar de manera adecuada es muy importante recordar que:

- Para expresar un valor de tipo alfanumérico o fecha, es requisito encerrarlo con comillas simples.
- Todo valor que no se especifique entre comillas simples, será interpretado como tipo de dato numérico.

Ejemplo:

Insertar el registro de un nuevo empleado con clave 18, llamada Lorena Aguilar, de 19 años de edad, con un sueldo de 7,000 pesos, teniendo cargo de empleado, entrando el 26 de Abril del 2002 al departamento de recursos humanos.

```
INSERT INTO empleado (
       id_empleado,
       id_departamento,
       id_cargo,
```

```
        nombre_empleado,
        edad_empleado,
        sexo_empleado,
        sueldo_empleado,
        fecha_contratacion
) VALUES ( 18, 2, 1, 'Lorena Aguilar',19, 'M', 7000.00, '2002-04-26' )
```

Código 3-46. Ingreso de un registro a una tabla usando la cláusula VALUES.

En este ejemplo se puede ver que el orden de los campos especificados se encuentran en un orden diferente al de la estructura física de la tabla (cuyo orden es: **id_empleado, nombre_empleado, edad_empleado, sexo_empleado, sueldo_empleado, porcentaje_comision, fecha_contratacion, id_departamento, id_cargo**) y además se omite el campo de porcentaje comisión ya que este solo se emplea para los vendedores y admite valores nulos.

Nota: El formato para especificar la fecha puede variar dependiendo del RDBMS, es necesario leer la documentación para determinar cual es la manera adecuada para especificar una fecha.

Eliminación de registros.

La instrucción **DELETE** elimina registros de la tabla indicada con la posibilidad de indicar un criterio, en caso de omitirlo, se eliminan todos los registros de la tabla.

Cláusula DELETE.

La sintaxis es la siguiente:

```
DELETE FROM <nombre_tabla> [ WHERE <condición> ]
```

Código 3-47. Sintaxis de la instrucción DELETE.

La cláusula **DELETE** permite indicar que la operación a realizar es una eliminación de registros.

Cláusula FROM

La cláusula FROM permite especificar la tabla de donde se desea eliminar registros.

Nota: En algunos RDBMS, la cláusula **FROM** se puede emplear cuando es necesario especificar varias tablas que se encuentran involucradas en la condición de borrado.

Cláusula WHERE

La cláusula **WHERE** permite delimitar el conjunto de registros a eliminar, si no se especifica esta condición, se eliminaran todos los registros que contenga la tabla especificada.

Actualización de datos.

Para modificar o actualizar los valores de los registros de una tabla se utiliza el comando **UPDATE.** Si no se especifica una condición con la cláusula **WHERE**, todos los registros que existan en la tabla serán actualizados.

La sintaxis es la siguiente:

```
UPDATE <nombre_tabla>
SET <campo 1> = <valor1>, <campo2> = <valor2>, ....
[ FROM <nombre_tabla 1>, .... ]
[ WHERE <condición> ]
```

Código 3-48. Sintaxis de la instrucción UPDATE.

Cláusula UPDATE.

La cláusula **UPDATE** es la que indica que la operación a ejecutar es una actualización. Después de la cláusula se especifica el nombre de la tabla en donde se encuentra la información que deseamos modificar. Sólo se puede especificar una tabla a la vez.

Ejemplo:

Actualizar el salario a 27,000 pesos y edad a 27 años de la empleada con clave 12 "Roxana Guerrero"

```
UPDATE empleado
SET salario_empleado = 27000, edad = 27
WHERE id_empleado = 12
```

Código 3-49. Actualizar ciertos campos de un registro usando UPDATE.

Cláusula SET.

Esta cláusula permite especificar los campos que se desean modificar y su nuevo valor. La cláusula se coloca una sola vez aunque sean varios campos los que se deseen modificar.

Ejemplo:

Actualizar el salario a 27,000 pesos y edad a 27 años de la empleada con clave 12 "Roxana Guerrero"

```
UPDATE empleado
SET salario_empleado = 27000, edad = 27
WHERE id_empleado = 12
```

Código 3-50. Uso de la cláusula SET en la instrucción UPDATE.

Cláusula WHERE.

Esta cláusula permite especificar un criterio para delimitar el conjunto de registros a modificar.

Ejemplo:

Actualizar el salario a 27,000 pesos y edad a 27 años de la empleada con clave 12 "Roxana Guerrero"

```
UPDATE empleado
```

```
set salario_empleado = 27000, edad = 27
WHERE id_empleado = 12
```
Código 3-51. Uso de la cláusula WHERE en la instrucción UPDATE.

Vistas.

Una vista es una tabla qué no ocupa espacio de almacenamiento para la información que contiene, porque su estructura e información está definida a través de la ejecución de una instrucción **SELECT**. Las vistas tiene dos usos: el primero es para simplificar el acceso a datos que se ocupan frecuentemente y que requieren una sentencia de SQL muy compleja para dicho acceso; y el segundo es con fines de seguridad, que permitan mantener ocultas ciertas columnas.

Aunque las vistas forman parte del estándar, algunos RDBMS como MySQL no las implementan actualmente.

Las vistas pueden ser utilizadas como si fueran tablas, pero con ciertas restricciones.

Creación de vistas

```
CREATE VIEW <nombre_vista> AS <instrucción SELECT>
```
Código 3-52. Sintaxis para crear una vista.

Ejemplo:

```
CREATE VIEW empleados_h AS
       SELECT * FROM empleados
       WHERE tipo_emp = 'H'
```
Código 3-53. Ejemplo de creación de una vista usando la instrucción CREATE VIEW.

Eliminación de Vistas

Para eliminar una vista se utiliza la siguiente instrucción.

```
DROP VIEW <nombre_vista>
```
Código 3-54. Sintaxis para eliminar vistas.

Ejemplo:

```
DROP VIEW empleados_h
```
Código 3-55. Uso de la instrucción DROP VIEW.

Consideraciones

La instrucción **SELECT** que define una vista puede:

- Incluir criterios de selección (cláusula **WHERE**).

La instrucción **SELECT** que define una vista NO puede:

- Usar funciones de agrupamiento (**COUNT**, **AVG**, **SUM**,...).
- Usar columnas calculadas.
- Consultar más de una tabla.
- Consultar otra vista.
- Utilizar la cláusula **ORDER BY**. Sin embargo, la cláusula **ORDER BY** se pueden aplicar a la vista una vez creada, siendo consultada como cualquier tabla.

Definición de privilegios.

Unos de los componentes de un RDBMS es el DCL (Data Control Language) que permite controlar y establecer restricciones de acceso a la información contenida en la base de datos. Es tarea del administrador de la base de datos el encargado de asignar o revocar permisos y/o crear usuarios en la base de datos. El manejo de usuario varía considerablemente de un RDBMS a otro, siendo que en algunos es más completo, el esquema del manejo de usuario y permisos, que en otros.

En Sybase:

Antes de que un usuario pueda acceder a alguna base de datos, éste debe primero darse de alta como usuario en el servidor. Para esto se usa el siguiente procedimiento almacenado, teniendo activa la base de datos **MASTER**:

```
sp_addlogin <nombre_usuario> , [<contraseña>] , [<base_de_datos_default>]
```

Código 3-56. Sintaxis para la creación de usuarios en Sybase.

Ejemplo:

```
sp_addlogin capturista, 'captdat', pubs2
```

Código 3-57. Crear el usuario "capturista" con la contraseña "captdat" con base de datos por defecto "pubs2" en Sybase

Posteriormente debe darse de alta al usuario en la base de datos que se desea pueda acceder; para esto se activa la base de datos deseada y se utiliza el siguiente procedimiento almacenado para darlo de alta:

```
sp_adduser <nombre_usuario>
```

Código 3-58. Sintaxis para dar de alta a un usuario en la base de datos activa en Sybase.

Por ejemplo, estando activa la base de datos **pubs2**:

```
sp_adduser capturista
```

Código 3-59. Dar de alta al usuario "capturista" en la base de datos activa en Sybase.

Se pueden controlar los privilegios a los usuarios de creación de objetos en la base de datos, así como la manipulación de la información contenida en ella, mediante la siguiente sintaxis mostrada en el Código 3-60 y el Código 3-61.

```
GRANT <privilegio> TO <usuario>
```
Código 3-60. Sintaxis para la dar privilegios a un usuario.

```
REVOKE <privilegio> FROM <usuario>
```
Código 3-61. Sintaxis para la revocar privilegios a un usuario.

Donde **GRANT** se utiliza para otorgar privilegios y **REVOKE** para eliminarlos.

Ejemplo:

```
GRANT ALL TO admon
```
Código 3-62. Otorga todos los privilegios al usuario "admon".

```
revoke create table from capturista
```
Código 3-63. Quita el privilegio de crear tables al usuario "capturista".

En PostgreSQL:

El esquema del manejo de usuarios y permisos no es tan refinado como en otros RDBMS.

Los usuarios se crean con la siguiente sintaxis:

```
CREATE USER <nombre_usuario>
[ WITH {       PASSWORD <contraseña> |
               CREATEDB |
               NOCREATEDB |
               CREATEUSER |
               NOCREATEUSER
       }
]
```
Código 3-64. Sintaxis para la creación de usuarios en PostgreSQL.

En MySQL:

No existe una sintaxis propia para crear usuarios sin embargo en el momento en el que se otorgan permisos, si el usuario al que se refiere la instrucción no existe, entonces es creado.

Por ejemplo, la siguiente instrucción otorga todos los permisos al usuario "miNuevoUsuario" sobre la base de datos "miBaseDeDatos" estableciéndose como contraseña "miContraseña":

```
GRANT ALL ON miBaseDeDatos.* TO miNuevoUsuario IDENTIFIED BY 'miContraseña';
```
Código 3-65. Otorgamiento de permisos en MySQL.

En Oracle:

La sintaxis básica para crear un nuevo usuario es la siguiente:

```
CREATE USER <nombre_usuario> IDENTIFIED BY <contraseña>;
```
Código 3-66. Sintaxis para la creación de usuarios en Oracle.

Una vez creado el usuario es necesario establecer los permisos correspondientes con la cláusula **GRANT**, para que se pueda utilizar dicha cuenta.

3.4 FUNCIONES DE UTILIDAD.

Aunque el estándar SQL92 define las funciones con las que debe contar un RDBMS, desafortunadamente en la práctica, no todos los RDBMS cumplen con el estándar.

Funciones para datos de tipo carácter.

El SQL92 define funciones de texto con sintaxis específica. A continuación se muestran aquellas funciones de uso frecuente. Los tipos de carácter soportados para SQL92 son char, varchar y text.

Función	Descripción	Ejemplo
char_length(<*cadena*>) character_length(<*cadena*>)	Determina la longitud de la cadena especificada	char_length('jose')
lower(<*cadena*>)	convierte el texto a minúsculas	lower('TOM')
octet_length(<*texto*>)	almacena el tamaño del texto	octet_length('jose')
position(<*cadena1*> in <*cadena2*>)	posición de un subtexto especificado	position('o' in 'Tom')
substring(<*cadena*> [from <*entero*>] [for <*entero*>])	extrae un subtexto especificado	substring('Tom' from 2 for 2)
trim([leading|trailing|both] [<*cadena 1*>] from <cadena2>)	remueve caracteres de un texto	trim(both 'x' from 'xTomx')
upper(<*cadena*>)	convierte un texto a mayúsculas	Upper('tom')
<cadena 1> || <cadena2>	Concatenación de cadenas	'Hola' || 'Mundo'
Case WHEN <*expresion*> THEN <*valor1*> [ELSE <*valor2*>] END	Evalúa la expresión si es verdadera regresa valor1 de lo contrario regresa valor2	CASE WHEN id_departamento is null THEN 'No tiene asignado Departamento' END

Tabla 3-11. Funciones definidas en el SQL92 para datos de tipo carácter.

Función SQL92	MySQL	PostgreSQL	Oracle	Sybase	Ms SQL Server
char_length	Si	Si	No	No	No
character_length					
lower	Si	Si	Si	Si	Si
octet_length	Si	Si	No	No	No
position	Si	Si	No	No	No
substring	Si	Si	No estándar	No estándar	No estándar
trim	Si	Si	Si	No	No
upper	Si	Si	Si	Si	Si
||	No	Si	Si	No	No
case	Si	Si	Si	No	No

Tabla 3-12. Soporte de funciones en distintos RDBMS.

A continuación se muestran las funciones de Sybase/ Ms SQL Server que remplazan a las ausentes del estándar SQL92:

Funcion	Sintaxis	Descripción
Substring	substring (<*cadena*>,#empiezo, #longitud)	Extrae una cadena de la cadena a partir de #empiezo hacia la izq. de #longitud
Charindex	charindex (<*cadena 1*>,<*cadena2*>)	Regresa la posición donde se encuentra la cadena1 dentro de la cadena2
ltrim	ltrim (<*cadena*>)	Elimina los blancos a la izquierda de la cadena
rtrim	rtrim (<*cadena*>)	Elimina los blancos a la derecha de la cadena
datalength	datalength (<*cadena*>)	Regresa la longitud de la cadena

Tabla 3-13. Funciones de Sybase / Ms SQL Server que remplazan a las no contempladas del estándar SQL92.

A continuación se muestran las funciones de MySQL que remplazan a las ausentes del estándar SQL92:

Función	Sintaxis	Descripción
Concat	concat(<*cadena1*>, <*cadena2*>)	Realiza la concatenación de las cadenas de texto especificadas

Tabla 3-14. Funciones de MySQL que remplazan a las no contepladas del estándar SQL92.

A continuación se muestran las funciones de Oracle que remplazan a las ausentes del estándar SQL92:

Función	Sintaxis	Descripción
Length	Length(<*cadena*>)	Obtiene la longitud de la cadena de texto especificada.
Instr	Instr(<*cadena1*>,<*cadena2*>)	Busca la cadena2 dentro de la cadena1, regresando la posición donde se encontró.
Substr	Substr(<*cadena*>,#empiezo, #longitud)	Extrae una cadena de la cadena a partir de #empiezo hacia la izq. de #longitud (Si este último se omite, se regresa hasta el final de la cadena)

Tabla 3-15. Funciones de Oracle que remplazan a las no contepladas del estándar SQL92.

Funciones matemáticas.

A continuación se muestra un listado de las funciones que forman parte del estándar SQL92 y en su caso, la función equivalente.

SQL92	MySQL	PostgreSQL	Oracle	Sybase	Ms SQL Server
abs	Si	Si	Si	Si	Si
acos	Si	Si	Si	Si	Si
asin	Si	Si	Si	Si	Si
atan	Si	Si	Si	Si	Si
ceiling	Si	ceil	ceil	Si	Si
cos	Si	Si	Si	Si	Si

cot	Si	Si	No	Si	Si
degrees	Si	Si	No	Si	Si
floor	Si	Si	Si	Si	Si
log	Si	Ln	Ln	Si	Si
log10	Si	log	Log	Si	Si
pi	Si	Si	No	Si	Si
power	Si	pow	Si	Si	Si
radians	Si	Si	No	Si	Si
rand	Si	random	No	Si	Si
round	Si	Si	Si	Si	Si
sign	Si	Si	Si	Si	Si
sin	Si	Si	Si	Si	Si
sqrt	Si	Si	Si	Si	Si
tan	Si	Si	Si	Si	Si

Tabla 3-16. Soporte de funciones matemáticas del estándar SQL92 en diversos RDBMS.

Funciones para datos tipo fecha.

Las funciones estándar para manejo de fecha son las siguientes:

Función	Descripción
Current_date / Current_date()	Regresa la fecha actual en formato 'YYYY-MM-DD' o YYYYM M DD
Current_time / Current_time()	Regresa la hora actual en formato 'HH:MM:SS' o HHMMSS
Current_timestamp / Current_timestamp()	Regresa la fecha y hora actual con formato 'YYYY-MM-DD HH:MM:SS' o YYYYMMDDHHMMSS

Tabla 3-17. Funciones de tipo fecha definidas por el estándar SQL92.

Desafortunadamente ni Oracle, ni Sybase, ni Ms SQL Server implementan las funciones definidas por el estándar, manejando sus propias funciones, y los campos tipo fecha incluyen también la hora como si fueran un campo de tipo timestamp.

Función SQL92	MySQL	PostgreSQL	Oracle	Sybase	Ms SQL Server
Current_date / Current_date()	Si	Si	sysdate	getdate()	getdate()
Current_time / Current_time()	Si	Si	Sysdate	getdate()	getdate()
Current_timestamp / Current_timestamp()	Si	Si	Sysdate	getdate()	getdate()

Tabla 3-18. Soporte de funciones de tipo fecha en diversos RDBMS.

3.5 MANEJO DE TRANSACCIONES.

Los RDBMS deben de implementar un mecanismo a través del cual, los cambios a realizar en la información, a través de una instrucción **INSERT**, **DELETE** o **UPDATE**, no son efectuados hasta que el usuario lo indique explícitamente. Una transacción puede comprender una o más instrucciones SQL.

MySQL	PostgreSQL	Oracle	Sybase	Ms SQL Server
Recientemente implementado en versión 4	Si	Si	Si	Si

Tabla 3-19. Soporte para transacciones en diversos RDBMS.

Una transacción es atómica, porque todas las instrucciones de SQL deben completarse con éxito, o ninguna de ellas. Una vez que el RDBMS determina que la transacción fue exitosa es necesario que la información sea almacenada de manera permanente. Una base de datos transaccional garantiza

que todas las operaciones realizadas en una transacción sean guardadas en almacenamiento permanente antes de que ésta sea reportada como completada, previniendo así, perdida de información por fallas del equipo, por ejemplo en un corte del suministro de energía.

Cuando múltiples usuarios realizan transacciones de manera concurrente, cada uno de ellos no debe ver los cambios incompletos realizados por los demás. En el momento que transacción finaliza adecuadamente y es almacenada permanentemente, los cambios se vuelven visibles para todos los demás usuarios.

Commit y rollback.

El comando *commit* permite indicar que los cambios a realizar dentro de una transacción sean llevados a cabo de manera permanente, y el comando *rollback* permite deshacer los cambios. El RDBMS reserva un espacio de almacenamiento, donde registra todas las instrucciones de SQL que se deben de ejecutar. De esta manera es posible deshacer los cambios realizados por operaciones **UPDATE**, DELETE o **INSERT**.

Las transacciones se inician de distinta manera, aunque similar. Por ejemplo en PostgreSQL, una transacción es iniciada mediante la cláusula **BEGIN** y en Sybase se emplea **BEGIN TRANSACTION**, seguida de las instrucciones de SQL a realizar y finalmente se emplea la cláusula **COMMIT** para indicar que las modificaciones deben realizarse de manera permanente o en caso que se desee cancelar la operación, se ocupa la cláusula **ROLLBACK**.

Ejemplo:

```
BEGIN;
UPDATE cuenta SET saldo = saldo - 1000.00 WHERE cliente = 'Fernanda';
UPDATE cuenta SET saldo = saldo + 1000.00 WHERE cliente = 'Alfredo';
COMMIT;
```

Código 3-67. Ejemplo de una transacción usando la instrucción COMMIT.

En el ejemplo anterior, se muestra las operaciones necesarias para realizar una transacción monetaria de $1,000 pesos de la cuenta de Fernanda a la cuenta de Alfredo. Si por alguna razón alguna de las dos instrucciones fallara todo el proceso sería cancelado.

La cláusula **COMMIT**, realiza los cambios de manera permanente. Si se diera el caso que Fernanda deseara cancelar su operación, en lugar de emplear la cláusula **COMMIT** se debería emplear la cláusula **ROLLBACK**.

Ejemplo:

```
BEGIN;
UPDATE cuenta SET saldo = saldo - 1000.00 WHERE cliente = 'Fernanda';
UPDATE cuenta SET saldo = saldo + 1000.00 WHERE cliente = 'Alfredo';
ROLLBACK;
```

Código 3-68. Ejemplo de una transacción usando la instrucción ROLLBACK.

Commit y rollback por fases.

Es posible especificar un punto en la transacción al cual se puede posteriormente realizar un **rollback** sin que esto afecte a toda la operación sino únicamente hasta el punto de referencia indicado (**SAVEPOINT**).

Este tipo de transacciones con **`rollback`** por fases, no está disponible en todos los RDBMS.

En Oracle.

```
UPDATE empleado
SET sueldo_empleado = 8000
WHERE nombre_empleado = 'Javier González';
SAVEPOINT javier_salario;

UPDATE empleado
SET sueldo_empleado = 7500
WHERE nombre_empleado = 'Gilberto Aparicio';
SAVEPOINT gilberto_salario;

SELECT SUM(sueldo_empleado) FROM empleado;
```

Código 3-69. Ejemplo de SAVEPOINT en Oracle.

Suponiendo que al realizar la consulta anterior se percibiera que solamente se puede aumentar el salario a Javier y no a Gilberto, porque la suma total de sueldos, rebasa el monto establecido por políticas de la empresa; entonces se puede hacer un rollback no a toda la operación sino únicamente hasta el punto de referencia llamado **`javier_salario`**.

```
ROLLBACK TO SAVEPOINT javier_salario
UPDATE empleado
SET sueldo_empleado = 7000
WHERE nombre_empleado = 'Gilberto Aparicio';
COMMIT;
```

Código 3-70. Uso de ROLLBACK en Oracle.

En Sybase.

Varían las cláusulas para establecer el punto de referencia. En el siguiente ejemplo se resaltan las diferencias:

```
UPDATE empleado
SET sueldo_empleado = 8000
WHERE nombre_empleado = 'Javier González';
SAVE TRANSACTION javier_salario;

UPDATE empleado
SET sueldo_empleado = 7500
WHERE nombre_empleado = 'Gilberto Aparicio';
SAVE TRANSACTION gilberto_salario;

SELECT SUM(sueldo_empleado) from empleado;

ROLLBACK TRANSACTION javier_salario
UPDATE empleado
SET sueldo_empleado = 7000
WHERE nombre_empleado = 'Gilberto Aparicio';
COMMIT;
```

Código 3-71. Uso de SAVE TRANSACTION y ROLLBACK en Sybase.

Este tipo de estructuras se emplea en el desarrollo de programas, que en un RDBMS puede ser un procedimiento almacenado. Los RDBMS comerciales son los que se han desarrollado más en el aspecto de programación en la base de datos, sin embargo las instrucciones aunque similares, difieren entre uno y otro.

If-Then-End If

```
IF <expresión>
        <bloque A de instrucciones>
ELSE
        <bloque B de instrucciones>
END IF
```

Código 3-72. Sintaxis de IF-THEN-END-IF en Sybase, Ms SQL Server, Oracle.

Si la expresión evaluada es verdadera se ejecuta el bloque A de instrucciones y de lo contrario se ejecuta el bloque B.

If-Then-Else-End If

```
IF <expresión 1>
        <bloque A de instrucciones>
ELSE IF <expresión 2>
        <bloque B de instrucciones>
ELSE IF <expresión 3>
        <bloque C de instrucciones>
ELSE
        <bloque D de instrucciones>
END IF
```

Código 3-73. Sintaxis en Sybase y Ms SQL Server.

La sintaxis en Oracle únicamente difiere en la cláusula elseif, la cual se escribe como ELSIF.

Si la `expresión 1` evaluada es verdadera se ejecuta `el bloque A de instrucciones`, de lo contrario se evalúa la `expresión 2` y si es verdadera se ejecuta `el bloque B de instrucciones`, de lo contrario se evalúa la siguiente expresión y así sucesivamente.

For Loop

```
FOR <variable> IN <valor inicial>..<valor final>
LOOP <lista_de_instrucciones>
END LOOP;
```

Código 3-74. Sintaxis en Oracle del For Loop.

Este ciclo iterativo inicializa la variable especificada y ejecuta las instrucciones cíclicamente hasta que la variable alcanza el valor final.

While Loop

Esta estructura se utiliza cuando se desea realizar una o un grupo de instrucciones repetidamente. La sintaxis en Sybase y Ms SQL Server es:

```
WHILE <expresión>
<instrucciones a ejecutar mientras la expresión sea verdadera> [BREAK]
[CONTINUE]
```

Código 3-75. Sintaxis de la instrucción WHILE en Sybase y Ms SQL Server

donde **break** se utiliza para salir del ciclo en el punto donde esta palabra se encuentra y **continue** se utiliza para regresar directamente a evaluar nuevamente la expresión.

Ejemplo:

```
WHILE (SELECT AVG(precio) FROM productos) < 1000
BEGIN
        IF (SELECT MAX(precio) FROM productos) <= 1500
        BEGIN
                PRINT "Actualizando Precios"
                UPDATE productos SET precio * 1.02
                CONTINUE
        END
        ELSE
        BEGIN
                PRINT "Llego al límite establecido"
                BREAK
        END
END
```

Código 3-76. Uso del WHILE en Sybase y Ms SQL Server.

En Oracle la sintaxis es la siguiente:

```
WHILE <condicion> LOOP <lista_de_instrucciones>
END LOOP;
```

Código 3-77. Sintaxis de la instrucción WHILE en Oracle

Etiquetas

Aunque las etiquetas no se recomiendan emplearlas en un ambiente de programación estructurado, en ocasiones pueden ser muy útiles para resolver un problema de una manera muy sencilla.

Las etiquetas en Sybase se emplean junto con la instrucción **GOTO** para trasladar el flujo del programa a la línea donde se encuentra la etiqueta señalada por el **GOTO**.

```
GOTO <etiqueta>
```

Código 3-78. Sintaxis para trasladar el flujo del programa con GOTO.

Ejemplo:

```
DECLARE @contador int
SELECT @contador = 0
inicia:
        SELECT @contador = @contador + 1
        PRINT "Ciclo numero %1!" , @contador
        IF @contador <= 10
                GOTO inicia
```

Código 3-79. Uso de etiquetas en Sybase.

3.7 PROCEDIMIENTOS ALMACENADOS.

Un procedimiento almacenado es un conjunto de comandos de SQL que pueden ser compilados y almacenados en el servidor. Una vez realizado esto, los clientes no necesitan volver a teclear todas las instrucciones sino únicamente hacer referencia al procedimiento. Esto mejora el rendimiento del servidor, ya que la instrucción de SQL solamente es revisada una sola vez y menos información debe ser enviada entre el cliente y el servidor.

MySQL	PostgreSQL	Oracle	Sybase	Ms SQL Server
No	Si	Si	Si	Si

Tabla 3-20. Soporte de procedimientos almacenados en diversos RDBMS.

El lenguaje que se emplea para programar los procedimientos almacenados, varía de un RDBMS a otro, y existen algunos que permiten programar en más de un lenguaje.

Declaración de variables.

Las variables son elementos fundamentales en la programación de procedimientos. Las variables deben ser declaradas al inicio del programa, antes de utilizarlas, para ello existe la cláusula DECLARE en Sybase, Ms SQL Server y Oracle.

Las variables locales sólo existen durante la ejecución del procedimiento donde son declaradas; cuando éste termina, las variables locales son eliminadas.

En Sybase y Ms SQL Server se debe observar lo siguiente para el manejo de variables:

- Los nombres de las variables locales deben estar precedidos invariablemente por un carácter de **@.**
- Antes de utilizar una variable ésta debe ser declarada indicando el nombre y el tipo de dato de la misma.

Para declarar las variables se utiliza la cláusula **DECLARE**, de la siguiente manera:

```
DECLARE <@variable> <tipo de dato> <(tamaño)> [, ]
```

Código 3-80. Sintaxis para declarar variables en Sybase y Ms SQL Server.

Ejemplo:

```
DECLARE @nombre char(50), @edad smallint, @sueldo numeric(10,2)
```

Código 3-81. Declaración de variables en Sybase y Ms SQL Server.

Las variables deben ser declaradas al inicio del batch

Para asignar valores a una variable se utiliza la instrucción **SELECT**, de la siguiente forma:

```
SELECT <@variable> = <expresión> [,...]
[FROM ....] [WHERE...]
```

Código 3-82. Sintaxis para asignar valores a una variable.

En este caso el uso de las cláusulas **FROM** y **WHERE** son opcionales.

Ejemplo:

```
SELECT @descuento = 0.45, @fecha = getdate()
```

Código 3-83. Asignando constantes a las variables.

```
SELECT @salario = salario, @nombre = nombre_emp
FROM empleados
WHERE clave_emp = "AA01"
```

Código 3-84. Asignando valores de una tabla a las variables

Si una instrucción **SELECT** que recupera datos de una tabla no regresa registros, las variables ahí involucradas conservan su valor.

Sybase y Ms SQL Server utiliza una serie de variables para alojar información a distintos niveles, ya sea a nivel de servidor, a nivel de sesión o a nivel de proceso; a dichas variables se les llama variables globales y el usuario no puede declararlas o asignarles valores. De esto se encarga automáticamente el servidor. Las variables globales van precedidas por doble **@.**

Algunas de las variables globales más útiles son:

Variable	Contiene
@@error	número de error generado por el último comando
@@rowcount	número de registros afectados en el último comando
@@version	número de versión de SQL server
@@spid	número de proceso en el SQL server
@@servername	nombre del servidor
@@sqlstatus	estatus del último comando fetch en un cursor

Tabla 3-21. Variables globales ocupadas en Sybase y Ms SQL Server.

En el caso de Oracle el manejo de variables es similar, porque también se emplea la cláusula **DECLARE** para definir las variables.

Ejemplo:

```
DECLARE
producto VARCHAR(20); precio NUMBER(6,2);
```

Código 3-85. Declaración de variables en Oracle.

En algunos casos, es posible que se desee que el tipo de una variable coincida con el tipo usado para una columna de una tabla determinada, en esos casos se puede usar la construcción:

```
DECLARE
sueldo empleado.sueldo_empleado%TYPE;
```
Código 3-86. Declaración de variables para que coincida con el tipo de dato usado por un campo de una tabla determinada en Oracle.

Con lo cual se logra que la variable sueldo tenga el mismo tipo que la columna sueldo_empleado de la tabla empleado.

También es posible inicializar las variables, mediante el operador := . Además, mediante el uso del mismo operador es posible hacer asignaciones en el cuerpo del programa.

Ejemplo:

```
DECLARE
salario NUMBER := 15000;
BEGIN
salario := salario + 1500; END;
```
Código 3-87. Uso del operador := al definir variables en Oracle

Creación de procedimientos.

Para crear procedimientos almacenados se utiliza la siguiente instrucción en Sybase y Ms SQL Server:

```
CREATE PROCEDURE <nombre_procedimiento>
[(@<nombre_parámetro> <tipo_de_dato>)] AS
<instrucciones SQL>
RETURN
```
Código 3-88. Sintaxis para la creación de procedimientos almacenados en Sybase y Ms SQL Server.

Ejemplo:

```
CREATE PROCEDURE nombres_empleados AS
SELECT nombre_empleado
FROM empleados
ORDER BY nombre_empleado
RETURN
```
Código 3-89. Creación de un procedimiento almacenado en Sybase y Ms SQL Server.

En Oracle no difiere mucho la sintaxis:

```
CREATE PROCEDURE <nombre_procedimiento>
[(<nombre_parámetro> [IN | OUT] <tipo_de_dato>) ] AS
<instrucciones SQL>;
```
Código 3-90. Sintaxis para la creación de procedimientos almacenados en Oracle.

Ejemplo:

```
CREATE PROCEDURE nombre_empleados BEGIN
SELECT nombre_empleado
FROM empleados
ORDER BY nombre_empleado END;
```
Código 3-91. Creación de un procedimiento almacenado en Oracle.

Ejecución de procedimientos.

Para ejecutar un procedimiento almacenado en Sybase o Ms SQL Server, sólo se tiene que indicar su nombre, en caso de que sea la primera instrucción dentro de un programa. En caso contrario se antepone la palabra exec. Si el procedimiento almacenado recibiera parámetros, éstos deben de indicarse después del nombre.

```
[EXEC] <nombre_procedimiento> [<parámetro>, ...]
```

Código 3-92. Sintaxis para ejecutar procedimientos almacenados en Sybase o Ms SQL Server.

Ejemplo:

```
EXEC nombre_empleados
```

Código 3-93. Ejecutar procedimientos almacenados en Sybase o Ms SQL Server.

En el caso de Oracle se emplea la instrucción CALL para llamar al procedimiento almacenado:

```
CALL nombre_empleados
```

Código 3-94. Ejecutar procedimientos almacenados en Oracle.

Eliminación de procedimientos.

Para eliminar un procedimiento almacenado, se dispone de la instrucción **DROP PROCEDURE**. La sintaxis es la siguiente:

```
DROP PROCEDURE <nombre_procedimiento>
```

Código 3-95. Sintaxis para eliminar un procedimiento almacenado.

Paso de parámetros.

El uso de parámetros incrementa la flexibilidad de un procedimiento almacenado. Éstos se definen desde la creación del procedimiento y deben ser proporcionados por el usuario al momento de ejecutarlo.

Un ejemplo en Sybase de un procedimiento con parámetros es el siguiente:

```
CREATE PROCEDURE sueldo_empleado (@sueldo numeric(8,2))
AS
SELECT nombre_empleado, sueldo_empleado FROM empleado
WHERE sueldo_empleado <= @sueldo ORDER BY nombre_empleado
RETURN
```

Código 3-96. Creación de un procedimiento con parámetros en Sybase.

Para ejecutar el procedimiento:

```
exec sueldo_empleado 10000
```

Código 3-97. Ejecutar un procedimiento almacenado pasándole parámetros en Sybase

En Oracle el ejemplo anterior quedaría de la siguiente manera:

```
CREATE PROCEDURE sueldo_empleado (sueldo IN number(8,2))
AS
BEGIN
SELECT nombre_empleado, sueldo_empleado
FROM empleado
WHERE sueldo_empleado <= sueldo
ORDER BY nombre_empleado;
END;
```

Código 3-98. Creación de un procedimiento con parámetros en Oracle.

Para ejecutar el procedimiento:

```
CALL sueldo_empleado 10000
```

Código 3-99. Ejecutar un procedimiento almacenado pasándole parámetros en Oracle

3.8. TRIGGERS.

Características.

Un trigger es un procedimiento almacenado que es invocado cuando un evento en particular ocurre. Por ejemplo, se puede ejecutar (disparar) un procedimiento almacenado cada vez que se borre, actualice o inserte un registro. Los procedimientos almacenados que se invocan tiene la restricción de que no pueden manejar parámetros ni ser invocados directamente.

Los TRIGGERS no forman parte del estándar SQL92, pero fueron contemplados posteriormente en el SQL99.

MySQL	PostgreSQL	Oracle	Sybase	*Ms SQL Server*
No	*Si*	*Si*	*Si*	*Si*

Tabla 3-22. Soporte de Triggers en diversos RDBMS.

En Sybase y Ms SQL Server un trigger al dispararse crea dos tablas temporales las cuales sólo pueden ser accesadas dentro del Trigger, y poseen la misma estructura de la tabla a la que está ligada. Estas tablas son: **`Inserted`** y **`Deleted`**.

Tabla	Contenido	En Triggers
Inserted	Contiene los registros que se van a agregar a la tabla, como resultado de los comandos Insert y Update	Insert, Update
Deleted	Contiene los registros que se van a eliminar de la tabla, como resultado de los comandos Delete y Update	Delete, Update

Tabla 3-23. Descripción de las tablas temporales creados al dispararse los Triggers en Sybase y Ms SQL Server..

En Oracle se utiliza el alias **`OLD`** y **`NEW`** para hacer referencia a los valores antes de la actualización y el nuevo valor, respectivamente.

Creación.

Para crear un Trigger se utiliza la siguiente sintaxis:

Sybase/Ms SQL Server	Oracle
CREATE TRIGGER <*nombre _trigger*> ON <nombre_tabla> FOR [INSERT | UPDATE | DELETE] AS <instrucciones SQL> ... RETURN	CREATE [OR REPLACE] TRIGGER <*nombre _trigger*> {BEFORE|AFTER} {I NSERT|DELETE|UPDATE} [OF COLUMN <nombre_columna>] ON <nombre_tabla> [FOR EACH ROW] <código pl/sql>

Código 3-100. Sintaxis para crear Triggers usando Sybase, Ms SQL Server y Oracle

A continuación se analiza la sintaxis en Oracle:

- **CREATE.** Esta cláusula indica la creación de un nuevo objeto.
- **OR REPLACE.** En caso de que el trigger ya exista, lo sobrescribe con el contenido actual. Esta cláusula es opcional.
- **TRIGGER.** La cláusula **TRIGGER** indica que el objeto que se desea crear es un **TRIGGER**. Inmediatamente después de esta cláusula, se coloca el nombre del trigger.
- **BEFORE** y **AFTER**. El trigger es una acción que se ejecuta cuando se da un determinado evento, por ejemplo una inserción. Con las cláusulas **BEFORE** y **AFTER** se especifica en que momento debe de activarse la acción, si Antes de la realización del evento o después. A un lado de cualquiera de estas dos cláusulas, se especifica el evento que va disparar la acción, pudiendo ser **INSERT**, **DELETE** o **UPDATE** que corresponde a una inserción, eliminación ó actualización de datos, respectivamente.
- **OF COLUMN.** En el caso de una actualización es posible especificar que únicamente cuando la modificación afecte a cierta columna se dispare la acción. Esto se realiza empleando la cláusula **OF COLUMN** seguida del nombre de la columna.
- **ON.** Permite especificar en que tabla se va a asociar el disparador de acción. Después de la cláusula **ON** se coloca el nombre de la tabla.
- **FOR EACH_ROW.** Esta cláusula permite especificar que la acción se ejecutara a nivel renglón. Es decir por cada registro afectado por la inserción, modificación o eliminación se realizara la acción especificada en el trigger.

Ejemplo:

```
CREATE TRIGGER ins_nomina on nomina
FOR INSERT
AS
/* Si no afectó ningún registro, sale del trigger*/
IF @@rowcount = 0
RETURN

/* Error si No existe registro en la tabla de empleados con la misma clave*/
IF ( SELECT COUNT(*) FROM empleados, inserted
     WHERE empleados.clave_emp = inserted.clave_emp ) = 0
BEGIN

/* No existe el empleado por tanto se cancela la transacción */
ROLLBACK TRANSACTION
RETURN
END RETURN
```

Código 3-101. Creación de un Trigger en la tabla nomina que verifique que exista la clave del empleado en la tabla de empleados cada vez que se inserte un registro usando Sybase / Ms SQL Server:

Eliminación.

La sintaxis para eliminar un Trigger es muy simple, únicamente se utiliza la instrucción **DROP TRIGGER**.

```
DROP TRIGGER <nombre_trigger>
```

Código 3-102. Sintaxis para eliminar un Trigger.

3.9 CURSORES.

Los cursores son apuntadores que tienen acceso de manera individual a un registro dentro de un conjunto de resultados, provenientes de una consulta a la base de datos.

Los cursores se emplean dentro procedimientos almacenados cuando se necesita realizar operaciones complejas de selección o actualización de datos, o cuando se necesita realizar operaciones que afecte registro por registro.

Uso de Cursores

Al utilizar cursores se deben de seguir los siguientes pasos:

- Declaración del cursor
- Apertura del cursor
- Recorrer los registros del cursor (realizar procesos)
- Cerrar el cursor

Declaración del cursor.

En la declaración de un cursor se indica: el nombre del cursor y una instrucción **SELECT** la cual definirá los campos y registros que se consultarán a la base de datos. Para declarar un cursor se sigue la siguiente sintaxis:

Sybase/Ms SQL Server	**Oracle**
DECLARE <*nombre_cursor*> CURSOR FOR <instrucción Select> [FOR {READ ONLY | UPDATE [OF <*campos*>] *}]*	DECLARE CURSOR <nombre_cursor> IS <instrucción Select>;

Código 3-103. Sintaxis para declarar un Cursor usando Sybase, Ms SQL Server y Oracle.

Apertura del cursor

Un cursor internamente es una sentencia **SELECT** cuyo resultado se guarda en el servidor en tablas temporales y que se va retornando cada una de las filas según se va pidiendo desde el cliente.

El primer paso es ejecutar la sentencia **SELECT** y guardar su resultado dentro de las tablas temporales. Este paso se denomina "Abrir el cursor". La apertura del cursor debe realizarse sólo una vez.

La sintaxis de apertura de un cursor es:

```
OPEN <nombre_cursor>;
```

Código 3-104. Sintaxis para la apertura de un cursor en Sybase, Ms SQL Server y Oracle.

Una vez que el cursor está abierto, se podrá empezar a pedir los resultados al servidor.

Recorrer los registros.

Una vez que el cursor está abierto en el servidor se podrá hacer la petición de recuperación de fila. Este paso es equivalente a hacer una consulta **SELECT** de una sola fila, ya que estamos seguros de que no vamos a recuperar más de una fila.

La sintaxis de recuperación de fila de un cursor es:

```
FETCH <nombre_cursor> INTO <variables>;
```

Código 3-105. Sintaxis para la recuperación de un cursor en Sybase, Ms SQL Server y Oracle.

Podremos recuperar filas mientras la consulta SELECT tenga filas pendientes de recuperar. Para saber cuándo se debe detener el ciclo de instrucciones FETCH, se realiza lo siguiente:

Sybase/Ms SQL Server	**Oracle**
Se evalúa la variable @@sqlstatus que contiene los siguientes valores: 0 - Se logró el acceso al registro 1 - No se logró el acceso al registro 2 - No hay más registros	Se realiza la evaluación de alguno de los siguientes atributos: <*nombre_cursor*>%FOUND BOOLEAN Retorna si la última fila recuperada fue válida <*nombre_cursor*>%ISOPEN BOOLEAN Retorna si el cursor está abierto <*nombre_cursor*>%NOTFOUND BOOLEAN Retorna si la última fila fue inválida <*nombre_cursor*>%ROWCOUNT NUMBER Retorna el número de filas recuperadas

Tabla 3-24. Evaluación para encontrar si se debe detener el ciclo de instrucciones FETCH en Sybase, Ms SQL Server y Oracle.

Así la acción más típica es recuperar filas mientras queden alguna por recuperar en el servidor. Esto se logra través del siguiente bloque de código:

Sybase/Ms SQL Server	**Oracle**
FETCH <nombre_cursor> INTO <@variables> WHILE (@@sqlstatus = 0) BEGIN <instrucciones para procesar cada registro> /* Avanza al siguiente registro */. FETCH <nombre_cursor> INTO <@variables> END	LOOP FETCH <nombre_cursor> INTO <variables>; EXIT WHEN <*nombre_cursor*>%NOTFOUND; <instrucciones para procesar cada registro> END LOOP;

Código 3-106. Recuperar filas usando FETCH en Sybase, Ms SQL Server y Oracle.

Cierre del cursor.

Finalmente para liberar los recursos empleados por el cursor, este debe ser cerrado. La sintaxis para liberar el cursor es:

```
DEALLOCATE [CURSOR] <nombre_cursor>
```

Código 3-107. Sintaxis para cerrar un cursor usando Sybase y Ms SQL Server.

```
CLOSE <nombre_cursor>;
```

Código 3-108. Sintaxis para cerrar un cursor usando Oracle.

3.10 JOINS INTERNOS.

Un **JOIN** se emplea para definir la relación que existe entre dos tablas. El **INNER JOIN** es una cláusula que nos permite definir estas relaciones. Este tipo de join puede expresarse de otra manera utilizando restricciones en la sección del **WHERE** en una consulta.

Ejemplo:

Consulta empleando **INNER JOIN**:

```
SELECT e.nombre_empleado, d.nombre_departamento
FROM empleado e INNER JOIN departamento d
     ON e.id_departamento = d.id_departamento
```

Código 3-109. Consulta empleando INNER JOIN.

La consulta anterior puede escribirse sin utilizar el **INNER JOIN** de la siguiente manera:

```
SELECT e.nombre_empleado, d.nombre_departamento
FROM empleado e, departamento d
WHERE e.id_departamento = d.id_departamento
```

Código 3-110. Consulta sin emplear el INNER JOIN.

El resultado de las dos consultas mostradas es el mismo. Lo único que varía es la forma de especificar la relación entre las tablas.

3.11 JOINS EXTERNOS

Si se deseara mostrar un listado de todos empleados con su respectivo departamento se pensaría en utilizar la siguiente consulta:

```
SELECT e.nombre_empleado, d.nombre_departamento
FROM empleado e, departamento d
WHERE e.id_departamento = d.id_departamento
```

Código 3-111. Consulta empleando WHERE en vez de INNER JOIN.

Esto nos traería el siguiente resultado:

nombre_empleado	nombre_departamento
Gilberto Aparicio	Sistemas
Javier González	Sistemas
Alma Benítez	Sistemas
Carolina Fernández	Sistemas
Alberto Ibáñez	Sistemas
Carlos López	Sistemas
Arturo Rangel	Ventas
Javier Bárcenas	Ventas
Ernesto Almaraz	Ventas
Susana García	Diseño
Fernanda Ramírez	Diseño
Roxana Guerrero	Diseño
Manuel Alavez	Recursos Humanos
Angeles Nolasco	Administración
Teresa Barba	Administración

Tabla 3-25. Resultado de una consulta con INNER JOIN pero que no obtiene los resultados nulos.

Sin embargo faltan dos empleados por mostrar:

Alberto González y Ricardo Contreras no aparecen en el listado anterior porque no tienen asignado ningún departamento todavía. Al poner la condición **`e.id_departamento = d.id_departamento`**, ninguno de los dos registros la cumple porque no hay un departamento con valor de **`NULL`**.

Es en estos casos donde es de gran utilidad el uso de joins externos, que permiten ignorar esta falta de correspondencia. El join externo se maneja de manera distinta en cada RDBMS.

En Oracle se añade un signo de más entre paréntesis para denotar el join externo. **`(+)`**

En Sybase y Ms SQL Server se antepone un asterisco al signo. **`*=`**

En MySQL y PostgreSQL no existe la implementación de joins externos, pero se puede lograr el mismo resultado utilizando joins derechos o izquierdos (**`LEFT JOIN`** ó **`RIGHT JOIN`**).

Por lo tanto para mostrar el listado de manera adecuada, se emplearían las siguientes consultas:

En Sybase ó Ms SQL Server:

```
SELECT e.nombre_empleado, d.nombre_departamento
FROM empleado e, departamento d
WHERE e.id_departamento *= d.id_departamento
```

Código 3-112. Uso de Join externo en Sybase ó Ms SQL Server

En Oracle:

```
SELECT e.nombre_empleado, d.nombre_departamento
FROM empleado e, departamento d
WHERE e.id_departamento = d.id_departamento(+)
```

Código 3-113. Uso de Join externo en Oracle

En MySQL ó PostgreSQL:

```
SELECT e.nombre_empleado, d.nombre_departamento
FROM empleado e LEFT JOIN departamento d
     ON e.id_departamento = d.id_departamento
```

Código 3-114. Uso de Join externo en MySQL ó PostgreSQL

Los empleados que no tienen departamento tendrán un valor **NULL** en su columna de **nombre_departamento**.

nombre_ empleado	nombre_departamento
Gilberto Aparicio	Sistemas
Javier González	Sistemas
Alma Benítez	Sistemas
Carolina Fernández	Sistemas
Alberto Ibáñez	Sistemas
Carlos López	Sistemas
Arturo Rangel	Ventas
Javier Bárcenas	Ventas
Ernesto Almaraz	Ventas
Susana García	Diseño
Fernanda Ramírez	Diseño
Roxana Guerrero	Diseño
Manuel Alavez	Recursos Humanos
Angeles Nolasco	Administración
Teresa Barba	Administración
Alberto González	
Ricardo Contreras	

Tabla 3-26. Resultado de una consulta con INNER JOIN que obtiene los resultados nulos.

3.12 OPERACIONES DE CONJUNTOS

La operación más usual y disponible en la mayoría de los RDBMS es la unión del resultado de dos consultas. Antes de emplear alguna de las operaciones con conjuntos permitidas, es mejor analizar si no existe una mejor alternativa, por ejemplo el uso de joins en lugar de una operación de conjunto, ya que el proceso de obtención del resultado es más rápido porque puede tomar ventaja de los índices. Oracle adicionalmente brinda las operaciones de intersección (**INTERSECT**) y la de resta (**MINUS**).

Union

Permite realizar la unión del resultado de dos consultas. El proceso de **UNION** elimina los registros duplicados, aunque algunos RDBMS proporcionan una cláusula que permite mostrar todos los registros resultantes de la unión.

Ejemplo. Mostrar el sueldo máximo y el mínimo que perciben los empleados.

```
SELECT MAX(sueldo_empleado)
FROM empleado UNION
SELECT MIN(sueldo_empleado)
FROM empleado
```

Código 3-115. Muestra el sueldo máximo y el mínimo que perciben los empleados usando UNION.

Tablas de ejemplo de empresa.

Id_departamento	nombre_departamento
1	Ventas
2	Recursos Humanos
3	Diseño
4	Sistemas
5	Administración
6	Redes

Tabla 3-27. Tabla departamento.

Id_cargo	nombre_cargo
1	Empleado
2	Jefe de Departamento
3	Coordinador
4	Ejecutivo de cuenta
5	Subdirector
6	Director

Tabla 3-28. Tabla cargo.

Id_empleado	nombre_empleado	edad_empleado	sexo_empleado	sueldo_empleado	porcentaje_comision	fecha_contratacion	Id_departamento	Id_cargo
1	Gilberto Aparicio	25	H	7000	NULL	25/11/2000	4	1
2	Javier González	30	H	7500	NULL	25/11/2000	4	1
3	Alma Benítez	19	M	10000	NULL	01/01/2001	4	1
4	Carolina Fernández	32	M	15000	NULL	01/03/2001	4	2
5	Alberto Ibáñez	32	H	25000	NULL	04/05/2001	4	2
6	Carlos López	32	H	27000	NULL	04/05/2001	4	3
7	Arturo Rangel	45	H	18000	5	25/10/2001	1	4
8	Javier Bárcenas	29	H	18000	10	25/10/2001	1	4
9	Ernesto Almaraz	32	H	20000	NULL	07/01/2002	1	2
10	Susana García	31	M	18000	NULL	07/01/2002	3	1
11	Fernanda Ramírez	18	M	18000	NULL	15/05/2002	3	1
12	Roxana Guerrero	25	M	26500	NULL	15/07/2002	3	2
13	Manuel Alavez	35	H	12600	NULL	23/09/2002	2	2
14	Angeles Nolasco	29	M	6000	NULL	23/09/2002	5	1
15	Teresa Barba	37	M	12000	NULL	23/09/2002	5	2
16	Alberto González	56	H	35000	NULL	27/10/2002		5
17	Ricardo Contreras	58	H	50500	NULL	27/10/2002		6

Tabla 3-29. Tabla empleado.

3.13 MANEJO DEL SQL EN UN TORNEO INTERNACIONAL. CASO PRÁCTICO

Se desea llevar a cabo un torneo de clase internacional y registrar los resultados de los encuentros. Para ello se decidió crear las tablas **`pais`** y **`resultado`**.

Ilustración 3-3. Modelado de datos inicial para la tabla pais y resultado.

- Elegir los tipos de datos más convenientes, si el marcador debe almacenarse como una cadena. Por ejemplo "**`5-4`**".

```
CREATE TABLE pais (id_pais numeric(5,0), nombre varchar(45))
CREATE TABLE resultado (id_pais1 numeric(5,0), id_pais2 numeric(5,0), marcador varchar(10))
```

Código 3-116. Creación de las tablas pais y resultado.

- Crear las llaves primarias para las tablas **`pais`** y **`resultado`** con "**`alter table`**"

```
ALTER TABLE pais
ADD CONSTRAINT pais1_pk PRIMARY KEY (id_pais)

ALTER TABLE resultado
ADD CONSTRAINT resultado1_pk PRIMARY KEY (id_pais1, id_pais2)
```

Código 3-117. Agregar llave primaria a las tablas pais y resultado.

- Crear las llaves foráneas de las tablas **`pais`** y **`resultado`** utilizando "**`alter table`**"

```
ALTER TABLE resultado
ADD CONSTRAINT pais_resultado1_fk
FOREIGN KEY (id_pais1) REFERENCES pais (id_pais)

ALTER TABLE resultado
ADD CONSTRAINT pais_resultado2_fk
FOREIGN KEY (id_pais2) REFERENCES pais (id_pais)
```

Código 3-118. Creación de llaves foráneas en la tabla resultado haciendo que dos campos referencien a un mismo campo en la tabla pais.

- En un esquema relacional mostrar la relación que existe entre la tabla pais y resultado

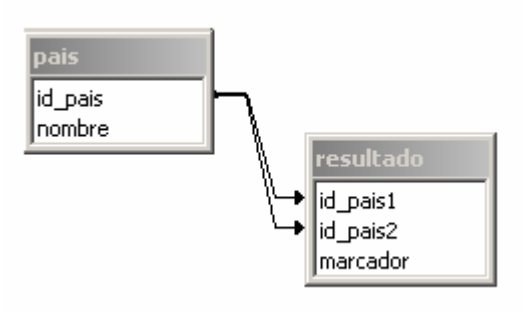


Ilustración 3-4. Modelado de datos para la tabla pais y resultado con llaves foráneas en la tabla resultado haciendo que dos campos referencien a un mismo campo en la tabla pais.

id_pais	Nombre
1	Alemania
2	Argentina
3	Bélgica
4	Brasil
5	Corea
6	Canadá
7	Dinamarca
8	España
9	Ecuador
10	Estados Unidos
11	Francia
12	Holanda
13	Inglaterra
14	México
15	Paraguay

Tabla 3-30. Tabla pais.

Id_pais1	Id_pais2	Marcador
1	2	3-4
2	4	1-1
3	10	0-0
4	8	0-0
5	14	1-1
14	1	3-1

Tabla 3-31. Tabla resultado.

En el torneo participan los países descritos en la tabla 3-30 y los resultados se van conformando como lo muestra la tabla 3-31. Usando SQL el código para insertar los países y resultados quedaría como lo muestra el código 3-119.

```
INSERT INTO pais (id_pais, nombre) VALUES (1,'Alemania');
INSERT INTO pais (id_pais, nombre) VALUES (2,'Argentina');
INSERT INTO pais (id_pais, nombre) VALUES (3,'Belgica');
INSERT INTO pais (id_pais, nombre) VALUES (4,'Brasil');
INSERT INTO pais (id_pais, nombre) VALUES (5,'Corea');
INSERT INTO pais (id_pais, nombre) VALUES (6,'Canada');
INSERT INTO pais (id_pais, nombre) VALUES (7,'Dinamarca');
INSERT INTO pais (id_pais, nombre) VALUES (8,'Espana');
INSERT INTO pais (id_pais, nombre) VALUES (9,'Ecuador');
INSERT INTO pais (id_pais, nombre) VALUES (10,'Estados Unidos');
INSERT INTO pais (id_pais, nombre) VALUES (11,'Francia');
INSERT INTO pais (id_pais, nombre) VALUES (12,'Holanda');
INSERT INTO pais (id_pais, nombre) VALUES (13,'Inglaterra');
INSERT INTO pais (id_pais, nombre) VALUES (14,'Mexico');
INSERT INTO pais (id_pais, nombre) VALUES (15,'Paraguay');

INSERT INTO resultado (id_pais1, id_pais2, marcador) VALUES (1,2,'3-4');
INSERT INTO resultado (id_pais1, id_pais2, marcador) VALUES (2,4,'1-1');
INSERT INTO resultado (id_pais1, id_pais2, marcador) VALUES (3,10,'0-0');
INSERT INTO resultado (id_pais1, id_pais2, marcador) VALUES (4,8,'0-0');
INSERT INTO resultado (id_pais1, id_pais2, marcador) VALUES (5,14,'1-1');
INSERT INTO resultado (id_pais1, id_pais2, marcador) VALUES (14,1,'3-1');
```

Código 3-119. SQL para insertar los paises y los resultados del torneo.

- Es necesario realizar una consulta que muestre el nombre de los países que jugaron y el marcador de dicho encuentro.

```
SELECT p1.nombre,
       p2.nombre,
       r.marcador
FROM resultado r,
     pais p1,
     pais p2
WHERE p1.id_pais = r.id_pais1 AND
      p2.id_pais = r.id_pais2
ORDER BY r.id_pais1;
```

Código 3-120. Consulta para identificar los resultados y los nombres de los países en una sola instrucción SQL.

- También se necesitará mostrar únicamente los encuentros, en los que se hayan empatado.

```
SELECT p1.nombre,
       p2.nombre,
       r.marcador
FROM resultado r,
     pais p1,
     pais p2
WHERE p1.id_pais = r.id_pais1 AND
      p2.id_pais = r.id_pais2 AND
      substring(marcador from 1 for 1)  = substring(marcador from 3 for 1)
ORDER BY r.id_pais1;
```

Código 3-121. Consulta para identificar los resultados de los encuentros que resultaron con empate.

- Mostrar los datos del partido en el cual haya habido el mayor número de goles.

```
SELECT pais1.nombre,
       pais2.nombre,
       resultadoygoles.marcador,
       resultadoygoles.goles
FROM ( /*Datos del partido con goles totales*/
       SELECT id_pais1,
              id_pais2,
              marcador,
              (
                CAST (SUBSTRING(marcador FROM 1 FOR 1) AS numeric(2,0) ) +
                CAST (SUBSTRING(marcador FROM 3 FOR 1) AS numeric(2,0) )
              ) AS goles
       FROM resultado
     ) AS resultadoygoles,
     pais AS pais1,
     pais AS pais2
WHERE goles = ( /*Máximo número de goles*/
                 SELECT MAX (
                        CAST (SUBSTRING(marcador FROM 1 FOR 1) AS numeric(2,0) ) +
                        CAST (SUBSTRING(marcador FROM 3 FOR 1) AS numeric(2,0) )
                          ) AS maxgoles
                FROM resultado
              ) AND
      pais1.id_pais = resultadoygoles.id_pais1 AND
      pais2.id_pais = resultadoygoles.id_pais2;
```

Código 3-122. Consulta en SQL de los datos del partido en el cual haya habido el mayor número de goles.

C a p í t u l o 4

ACCESO A DATOS A TRAVÉS DE LA PROGRAMACIÓN DE CLIENTES

Utilizar distintos elementos que brindan de los lenguajes de programación, para la consulta y manipulación de información en las bases de datos.

4.1 WORLD WIDE WEB (WWW)

En la Web es utilizada la arquitectura denominada Cliente / Servidor en la que en una máquina central (servidor) se encuentran varios servicios a los que una o varias computadoras (cliente) pueden acceder, dependiendo de los privilegios que el usuario tenga dados de alta el servidor podrá tener acceso a ciertos servicios.

La relación entre el navegador del Cliente y el Servidor al ingresar a una página de hipertexto estática se ilustra en la siguiente figura:

Ilustración 4-1. Relación entre el navegador del Cliente y el Servidor.

4.2 HTML

HyperText Markup Language o HTML es un sistema para estructurar documentos. Estos documentos pueden ser mostrados por los navegadores de páginas Web en Internet, como Netscape, Mozilla, Galeon, Microsoft Explorer, etcétera. Cualquier cosa que hagamos debe ir entre etiquetas.

```
<ETIQUETA parámetros> ... </ETIQUETA>
```

Código 4-1. Sintaxis básica de estructurar etiquetas en HTML.

Existen etiquetas que no necesitan cerrarse, así como etiquetas que no requieren de parámetros, más adelante se verán algunos ejemplos.

Un documento escrito en HTML contendría básicamente lo siguiente:

```
<HTML>          Indica el inicio del documento.
 <HEAD>         Indica el encabezado del documento.
  <TITLE>       Indica el título del documento.
  </TITLE>      Indica el fin de la etiqueta del título.
 </HEAD>        Indica el fin de la etiqueta de encabezado.
 <BODY>         Indica el cuerpo del documento.
 </BODY>        Indica el fin del cuerpo del documento.
</HTML>         Indica el fin del documento.
```

Código 4-2. Documento escrito en HTML con las etiquetas básicas para su formación.

Encabezado

La etiqueta **<HEAD> </HEAD>** delimita el encabezado, la etiqueta más importante dentro de ésta es **<TITLE>** en la cual se coloca el título de la página.

Ejemplo:

```
<HEAD>
 <TITLE>Mi primera p&aacute;gina en html</TITLE>
</HEAD>
```

Código 4-3. Uso de la etiqueta HEAD.

Cuerpo

La etiqueta **<BODY> </BODY>** indica el contenido de nuestra página. Entre esta etiqueta pondremos texto, gráficos, ligas, etcétera. Tiene una serie de parámetros opcionales que permiten definir la apariencia general del documento:

- **background= "nombre de imagen"**. Indica el nombre de una imagen que servirá como "fondo" de nuestra página. Si la imagen no rellena todo el fondo del documento, esta será reproducida tantas veces como sea necesario.
- **bgcolor = "código de color"**. Indica un color para el fondo de nuestro documento. Se ignora si se usó el parámetro background.
- **text = "código de color"**. Indica un color para el texto que incluyamos en nuestro documento. Por defecto es negro.
- **link = "código de color"**. Indica el color de los textos que dan acceso a una liga. Por defecto es azul.
- **vlink = "código de color"**. Indica el color de los textos que dan acceso a una liga que ya hemos visitado. Por defecto es morado.

Código de color

El código de color es un número compuesto por tres pares de cifras hexadecimales que indican la proporción de los colores rojo, verde y azul. El código de color se antecede del símbolo **#**.

Ejemplos:

- **#000000** Color Negro
- **#FFFFFF** Color Blanco

Etiquetas para dar formato

 	Sirve para indicar un salto de línea.
<P>	Indica un cambio de párrafo y deja una línea en blanco en medio.
<P> </P>	Cuando se utiliza así, define las características de un párrafo en específico. Se puede usar el parámetro align cuyos posibles valores son LEFT, RIGHT y CENTER.
<HR>	Inserta una línea horizontal en la página.
<FONT></FONT>	Permite variar el tamaño, el color y el tipo de letra de un texto determinado. Utiliza para ello los parámetros size, color y face. size = "valor" Da al texto un tamaño determinado. color = "código de color" Escribe el texto en el color cuyo código se especifica. face = "nombre de tipo de letra". Escribe el texto en el tipo de letra especificado. Si este tipo de letra no existe en la computadora que "lee" la página se usará el tipo de letra predeterminado del navegador.

*Tabla 4-1. Descripción de las etiquetas
, <P>, <HR> y <FONT>.*

Ejemplo:

```
<FONT size="2" color="# FF0000" face="Arial">Texto de prueba </FONT>
```

Código 4-4. Uso de la etiqueta <FONT>.

<!-- -->	Todo lo que este dentro de esta etiqueta es un comentario que no será mostrado en la página.
<A> </A>	Permite definir ligas, el texto o imagen que este dentro de esta etiqueta será sensible, cuando se presione el botón izquierdo del ratón sobre esta sección se ejecutará lo que esta definido en el parámetro href de la etiqueta.

Tabla 4-2. Descripción de las etiquetas <!-- --> y <A>.

Ejemplo:

```
<A href="www.unam.mx">UNAM</A>
```

Código 4-5. Uso de la etiqueta <A>.

Caracteres especiales

´	Se utiliza para acentuar una vocal.
ñ	Se utiliza para escribir le letra ñ.
¿	Se utiliza para escribir el signo de interrogación que inicia una pregunta.

Tabla 4-3. Caracteres de acento, ñ, ¿.

Ejemplo:

```
Cu&aacute;l
```

Código 4-6. Escribirá en el navegador "Cuál".

Tablas

Son elementos muy útiles para ordenar los contenidos de nuestras páginas. Se definen mediante la etiqueta **<TABLE>**. **</TABLE>**, los parámetros opcionales son:

- **border = "número"**. Indica el ancho del borde de la tabla.
- **cellspacing ="número"**. Indica el espacio en píxeles que separa las celdas que están dentro de la tabla.
- **cellpadding = "número"**. Indica el espacio en píxeles que separa el borde de cada celda y el contenido de esta.
- **width = "número o porcentaje"**. Indica el ancho de la tabla en píxeles o en porcentaje en función del ancho de la ventana del navegador. Si no se indica este parámetro, el ancho se adecuará al tamaño de los contenidos de las celdas.
- **height = "número o porcentaje"**. Indica la altura de la tabla en píxeles o en porcentaje en función del alto de la ventana del navegador.
- **bgcolor = "código de color"**. Especifica el color de fondo de toda la Tabla.

Para definir las celdas que componen la tabla se utilizan las directivas **<TD>** y **<TH>**. **<TD>** indica una celda normal, y **<TH>** indica una celda de "título" , es decir, el contenido será resaltado en negrita y en un tamaño ligeramente superior al normal.

Los parámetros opcionales de ambas directivas son:

- **align = "LEFT / CENTER / RIGHT / JUSTIFY"**. Indica como se debe alinear el contenido de la celda, a la izquierda, centrado, a la derecha, o justificado.
- **valign = "TOP / MIDDLE / BOTTOM"**. Indica la alineación vertical del contenido de la celda, en la parte superior, en el centro o en la inferior.
- **rowspan = "número"**. Indica el número de filas que ocupará la celda. Por defecto ocupa una sola fila.
- **colspan = "número"**. Indica el número de columnas que ocupará la celda. Por defecto ocupa una sola columna.
- **width = "número o porcentaje"**. Indica el ancho de la columna en píxeles o en porcentaje en función del ancho de la ventana del navegador. Si no se indica este parámetro, el ancho se adecuará al tamaño de los contenidos.
- **bgcolor = "código de color"**. Especifica el color de fondo del elemento de la Tabla.

Para indicar una fila de la tabla se utiliza la etiqueta **<TR></TR>**.

Ejemplo:

```
<TABLE border=4 cellspacing=4 cellpadding=4 width =80%>
 <TH align=center>Nombre </TH>
 <TH align=center colspan=2>Apellidos </TH>
 <TR>
  <TD align=LEFT>Ana </TD>
  <TD align = LEFT>García </TD>
  <TD align = LEFT>Hernández </TD>
 </TR>
 <TR>
  <TD align = LEFT>José </TD>
  <TD align = LEFT>Martínez </TD>
  <TD align = LEFT>Cruz </TD>
 </TR>
</TABLE>
```

Código 4-7. Ejemplo de uso de tablas.

Formularios

Permiten dentro de una página solicitar información al visitante y procesarla, se pueden solicitar diferentes datos cada uno de los cuales quedará asociado a una variable. Una vez se hayan introducido los valores en los campos, el contenido de estos será enviado a la dirección donde tengamos el programa que pueda procesar las variables.

Este programa deberá estar escrito en un lenguaje de programación como ASP, Perl, PHP, etcétera.

La etiqueta **<FORM> </FORM>** permite definir formularios, sus parámetros son:

- **name="nombre de la forma"**. Indica el nombre de la forma.
- **action="programa"**. Indica la dirección del programa que va a procesar los datos recibidos de la forma.
- **method="POST / GET"**. Indica el método de transmisión a usar, **POST** para que los datos viajen ocultos y **GET** que si permite que se vean los valores y nombres de las variables en la barra de direcciones del navegador, este último es el valor por defecto.

Elementos de una forma

Ya que todos los elementos de una forma serán recibidos por un programa que va a procesarlos deberán tener obligatoriamente el parámetro: name="nombre del elemento".

Cajas de texto

Para definirlas se utiliza la etiqueta **<INPUT>** con el parámetro **type="text"**, otros parámetros opcionales son:

- **maxlenght= "número"**. Número máximo de caracteres a introducir en el campo.
- **size = "número"**. Tamaño en caracteres que se mostrará en pantalla.
- **value = "texto"**. Valor inicial del campo. Normalmente será**" "**, o sea, vacío.

Cajas de texto para introducir contraseñas

Para definirlas se utiliza la etiqueta **<INPUT>** con el parámetro **type="password"** Mostrará asteriscos (*****) en lugar de las letras escritas. Sus parámetros opcionales son los mismos que para las cajas de texto normales.

Cuadros de selección múltiple

Se utiliza la etiqueta **<INPUT>** con el parámetro **type="checkbox"** para definirlos. Permite elegir varios cuadros o casillas. Los valores de las casillas serán indicados por: **value="valor" checked**. Parámetro opcional con el que la casilla aparecerá marcada por defecto.

Cuadros de selección sencilla

Se usa la etiqueta **<INPUT>** con el parámetro **type="radio"** para definirlos. Únicamente permite elegir un valor. Se utiliza **value="valor"** para definir los valores de las casillas.

Botón para envío de datos

El tipo deberá ser igual a submit. Al pulsar este botón la información de todos los campos se envía al programa indicado en el parámetro action de **<FORM>**. Tiene el parámetro **value = "texto"** que indica el texto que aparecerá en el botón.

Botón para borrar los datos

Se utiliza la etiqueta **<INPUT>** con el parámetro **type="reset"** Al presionar el botón se borra el contenido de todos los campos. El parámetro value="texto" indica la leyenda que aparecerá en el botón.

Valores ocultos

La etiqueta **<INPUT>** deberá tener el parámetro **type="hidden"**. Este campo no será visible para el usuario. El valor se indica con el parámetro **value="valor"**

Listas de selección

Despliegan una lista de opciones entre las que se pueden escoger una o varias. La etiqueta que se usa es **<SELECT></SELECT>** y sus parámetros son:

- **size="número"**. Número de opciones visibles. Si se indica 1 se presenta como un menú desplegable, si se indica más de uno se presenta como una lista con barra de desplazamiento.
- **multiple**. Permite seleccionar más de un valor para el campo.

Las diferentes opciones de la lista se indican con la etiqueta **<OPTION>**. Esta etiqueta puede incluir el parámetro selected para indicar cual es la opción por defecto. En caso de que no se especifique, se tomará por defecto la primera opción de la lista.

Áreas de texto

Es un campo de texto que permite incluir varias líneas, para definirlo se usa la etiqueta **<TEXTAREA> </TEXTAREA>**, sus parámetros son:

- **cols="número"**. Número de columnas de texto visibles.
- **rows="número"**. Número de filas de texto visibles.
- **wrap="VIRTUAL / PHYSICAL"**. Justifica el texto automáticamente en el interior de la caja. La opción **PHYSICAL** envía las líneas de texto separadas en líneas físicas. La opción **VIRTUAL** envía todo el texto seguido.

4.3 USO DE PHP ESTRUCTURADO

PHP (Hipertext Preprocesor) se combina con código HTML para generar páginas Web dinámicas. Es ejecutado en el servidor y el resultado se envía al navegador, como se muestra en la siguiente figura.

Ilustración 4-2. Dinámica Cliente / Servidor que usa PHP.

Para indicarle al navegador que vamos a usar PHP debemos usar las siguientes etiquetas **<?php** y **?>** o **<?** y **?>**, todo lo que este entre esas etiquetas será interpretado como código PHP.

Usando un editor de texto cualquiera podemos crear nuestros scripts de PHP, dependiendo de la configuración del servidor en donde se alojarán los programas la extensión de los archivos podrá ser **.php**, **.html** o **.phtml**.

Al final de cada instrucción deberá existir un punto y coma (**;**)

Para agregar comentarios en este lenguaje es igual que en C o C++ usando **/* */** , **//**. También se puede comentar renglón por renglón con el símbolo de número:

```
/*Todo un texto comentado entre estos caracteres*/
//Un renglón comentado
#Comentario por
#cada renglón
```

Código 4-8. Opciones para comentar código en PHP.

Una variable en este lenguaje define su tipo según su contenido, las variables son de tipo escalar y tomarán un tipo determinado dependiendo de la asignación que se haga, podrán ser booleanas, enteras, de punto flotante o cadenas.

En éste lenguaje también contamos con arreglos y objetos.

Es importante tomar en cuenta que PHP es sensible a mayúsculas y minúsculas, además de que no debe hacerse una declaración de variables previa ya que las variables se van generando conforme se van necesitando.

Estructuras de control en PHP

if

if. Permite la ejecución condicional de fragmentos de código.

if (expr). La sentencia expr se evalúa a su valor condicional. Si **expr** se evalúa como **VERDADERO**, se ejecutará la sentencia, y si se evalúa como **FALSO** se ignorará.

El siguiente ejemplo mostraría "a es mayor que b" si **$a** fuera mayor que **$b**:

```
if ($a > $b)
print "a es mayor que b";
```

Código 4-9. Uso de IF en PHP.

Si se requiere ejecutar más de una sentencia cuando se dé cierta condición se deberá poner entre llaves.

```
if ($a > $b){
        print "a es mayor que b";
        $b = $a;
}
```

Código 4-10. Uso de IF para ejecutar un conjunto de sentencias en PHP

else

Extiende una sentencia **if** para ejecutar una sentencia en caso de que la expresión en la sentencia **if** se evalúe como **FALSO**.

Ejemplo:

```
if ($a > $b){
        print "a es mayor que b";
} else {
        print "a NO es mayor que b";
}
```

Código 4-11. Uso de ELSE en PHP.

elseif

Extiende una sentencia **if** para ejecutar una sentencia diferente en caso de que la expresión **if** original se evalúe como **FALSO**. A diferencia de else, se ejecutará esa expresión alternativa solamente si la expresión condicional elseif se evalúa como **VERDADERO**.

La sentencia elseif se ejecuta sólo si la expresión **if** precedente y cualquier expresión **elseif** precedente se evalúan como **FALSO**, y la expresión elseif actual se evalúa como **VERDADERO**.

Ejemplo:

```
if ($a > $b) {
        print "a es mayor que b";
} elseif ($a == $b) {
        print "a es igual que b";
} else {
        print "a es mayor que b";
}
```

Código 4-12. Uso de ELSEIF en PHP.

Extiende una sentencia **if** para ejecutar una sentencia diferente en caso de que la expresión **if** original se evalúe como **FALSO**. A diferencia de **else**, se ejecutará esa expresión alternativa solamente si la expresión condicional elseif se evalúa como **VERDADERO**.

La sentencia **elseif** se ejecuta sólo si la expresión **if** precedente y cualquier expresión **elseif** precedente se evalúan como **FALSO**, y la expresión **elseif** actual se evalúa como **VERDADERO**.

while

Estos ciclos son los más simples en PHP. Su sintaxis es:

```
while (expr) sentencia
```

Código 4-13. Sintaxis del ciclo WHILE en PHP.

Se ejecuta(n) la(s) sentencia(s) anidada(s) repetidamente, mientras la expresión while se evalúe como VERDADERO. El valor de la expresión es comprobado cada vez al principio del ciclo, así que incluso si este valor cambia durante la ejecución de la(s) sentencia(s) anidada(s), la ejecución no parará hasta el fin de la iteración. Si la expresión **while** se evalúa como FALSO desde el principio de todo, la(s) sentencia(s) anidada(s) no se ejecutarán ni siquiera una vez.

Ejemplo:

```
$a=1;
$b=3;
while ($a < $b) {
       print "a es menor que b";
       $a++;
}
```

Código 4-14. Uso del ciclo WHILE en PHP.

do ... while

Son muy similares a los ciclos while a diferencia que primero se ejecutan las sentencias y al final se evalúa la condición. Hay una sola sintaxis para los bucles **do...while**:

```
do {
       sentencia
} while (expr)
```

Código 4-15. Sintaxis de DO … WHILE en PHP.

La sentencia siempre se ejecutará una sola vez.

Ejemplo:

```
$a=1;
$b=3;
do {
       print "a es mayor que b";
} while ($a > $b);
```

Código 4-16. Uso de DO … WHILE en PHP.

La sentencia siempre se ejecutará una sola vez aunque la condición no se cumpla.

for

Es el ciclo más complicado en PHP. Su sintaxis es:

```
for (expr1; expr2; expr3) sentencia
```

Código 4-17. Sintaxis de FOR en PHP.

`expr1` se evalúa (ejecuta) incondicionalmente una vez al principio del ciclo.

Al comienzo de cada iteración, se evalúa **`expr2`**. Si se evalúa como **`VERDADERO`**, el bucle continúa y las sentencias anidadas se ejecutan. Si se evalúa como **`FALSO`**, la ejecución del bucle finaliza. Al final de cada iteración, se evalúa (ejecuta) **`expr3`**.

Ejemplo:

```
for ($i = 1; $i <= 10; $i++) {
        print $i;
}
```

Código 4-18. Uso del ciclo FOR en PHP.

Ejemplo de un programa que imprime los números del uno al diez:

```
<html>
 <body bgcolor="#C8C8C8">
<?php
#Ciclo que hará la impresión de los números.
#Variable $i inicia en uno, el ciclo terminara cuando $i sea igual a diez
for ($i = 1 ; $i <= 10 ; $i++) {
        print "N&uacute;mero $i <br>"; #Realiza la impresión
}
print "Termin&oacute; la impresi&oacute;n de n&uacute;meros.";
?> </body>
</html>
```

Código 4-19. Ejemplo del ciclo FOR en PHP.

switch

Es similar a una serie de sentencias **`if`** en la misma expresión. En algunas ocasiones se requiere comparar una misma variable con valores diferentes y ejecutar diferentes sentencias dependiendo del valor de la variable.

```
switch (variable) {
        case valor1:
        sentencias;
        break;
        case valor2:
        sentencias;
        break;
}
```

Código 4-20. Sintaxis de SWITCH en PHP.

Ejemplo:

```
switch ($i) {
        case 0:
                print "i es $i";
        break;

        case 1:
                print "i es $i";
        break;

        case 2:
                print "i es $i";
        break;
}
```

Código 4-21. Uso de SWITCH en PHP.

break

break escapa de las estructuras de control **for**, **while**, o **switch**. Acepta un parámetro opcional, el cual determina de cuantas estructuras de control hay que escapar.

Acceso a bases de datos

Para utilizar PHP con Sybase es necesario tener acceso a una cuenta de usuario que tenga instalado PHP y una cuenta con acceso a Sybase, puede ser la misma pero se deberá tener acceso tanto al lenguaje de programación como al manejador de base de datos. PHP debe estar compilado para soportar conexiones a Sybase.

Variables de configuración.

- **sybct.allow_persistent booleano**. Permite conexiones persistentes con Sybase CT. El valor por default es **"on"**.
- **sybct.max_persistent entero**. Número máximo de conexiones persistentes por proceso, el valor por default es **-1**, esto significa que es ilimitado.
- **sybct.max_links entero**. Número máximo de conexiones por proceso, incluyendo conexiones persistentes. El valor por default es **-1** y significa que no hay un límite de conexiones.
- **sybct.hostname cadena**. El nombre del servidor al que se va a conectar, para ser mostrado por **sp_who**. El valor por omisión es vacío.

Funciones que facilitan la interacción con Sybase.

- **sybase_affected_row**s. Trae el número de columnas afectadas en la última consulta. Únicamente para **INSERT**, **DELETE**, y **UPDATE**.

```
int sybase_affected_rows ( [int conexión] )
```

Código 4-22. Sintaxis de la función sybase_affected_rows para usarla en PHP.

- **sybase_close**. Cierra una conexión con Sybase. Regresa verdadero en caso de que sea exitoso. No cierra conexiones persistentes.

```
bool sybase_close ( int conexión )
```

Código 4-23. Sintaxis de la función sybase_close para usarla en PHP.

- **sybase_connect**. Abre una conexión al servidor Sybase. Devuelve falso en error.

```
int sybase_connect ( nombre_servidor, nombre_usuario, password [,string charset] )
```

Código 4-24. Sintaxis de la función sybase_connect para usarla en PHP.

- **sybase_data_seek**. Coloca en una determinada posición el cursor de la consulta. Mueve el puntero interno asociado con el resultado de la consulta al número de registro especificado, la siguiente llamada a **sybase_fetch_row()** regresará ese registro. Regresa: **TRUE** en éxito, **FALSE** en falla.

```
bool sybase_data_seek ( int variable_resultado_consulta, int número_registro )
```

Código 4-25. Sintaxis de la función sybase_data_seek para usarla en PHP.

- **sybase_fetch_array**. Pone el resultado de una consulta en un arreglo. Devuelve un arreglo que corresponde al registro, o **FALSE** si no hay más registros. Es una extensión a la versión de **sybase_fetch_row()**.

```
array sybase_fetch_array ( int variable_consulta )
```

Código 4-26. Sintaxis de la función sybase_fetch_array para usarla en PHP.

- **sybase_fetch_field**. Trae la información del campo, regresa un objeto con dicha información. Puede ser usado para obtener información acerca de los campos en cierto resultado de una consulta.

```
object sybase_fetch_field ( int variable_resultado [, int campo_offset] )
```

Código 4-27. Sintaxis de la función sybase_fetch_field para usarla en PHP.

Las propiedades del objeto son:

name.- Nombre de la columna, si la columna es el resultado de una función, esta propiedad es puesta en **#N**, donde **#N** es un número **serial**.

column_source.- La tabla de la cual fue tomada la columna.

max_length.- Tamaño máximo de la columna.

numeric.- **1** si la columna es numérica.

- **sybase_fetch_object**. Trae un registro como objeto. Devuelve un objeto con propiedades que corresponden al registro, o **FALSE** si no hay más registros. Es similar a **sybase_fetch_array()**, con una diferencia se regresa un objeto en lugar de un arreglo. Únicamente se pueden acceder a los datos por medio de los nombres de los campos y no por sus **offsets**.

```
int sybase_fetch_object ( int variable_resultado )
```

Código 4-28. Sintaxis de la función sybase_fetch_object para usarla en PHP.

- **sybase_fetch_row**. Toma el registro como un arreglo enumerado. Regresa un arreglo que corresponde al registro traído, o **FALSE** si no hay más renglones. **sybase_fetch_row()** trae un renglón de datos del resultado asociado con el identificador de resultado. El registro es devuelto como un arreglo. Cada columna del resultado es almacenada es un arreglo, el arreglo comienza en **0**. La siguiente llamada a **sybase_fetch_row()** regresará el siguiente renglón en el resultado, o **FALSE** si no hay más columnas.

```
array sybase_fetch_row ( int variable_resultado )
```

Código 4-29. Sintaxis de la función sybase_fetch_row para usarla en PHP.

- **sybase_field_seek**. Se coloca en un campo específico, si la siguiente llamada a **sybase_fetch_field()** no incluye el campo de salida, el anterior será regresado.

```
int sybase_field_seek ( int variable_resultado, int campo_salida )
```

Código 4-30. Sintaxis de la función sybase_field_seek para usarla en PHP.

- **sybase_free_result**. Libera la memoria, únicamente se tiene que llamar si se esta preocupado de usar mucha memoria mientras el programa esta corriendo. La memoria será liberada cuando el programa termine.

```
bool sybase_free_result ( int variable_resultado )
```

Código 4-31. Sintaxis de la función sybase_free_result para usarla en PHP.

- **sybase_get_last_message**. Devuelve el último mensaje del servidor.

```
string sybase_get_last_message ( void )
```

Código 4-32. Sintaxis de la función sybase_get_last_message para usarla en PHP.

- **sybase_num_fields**. Da el número de campos en el resultado.

```
int sybase_num_fields ( int variable_resultado )
```

Código 4-33. Sintaxis de la función sybase_num_fields para usarla en PHP.

- **sybase_num_rows**. Da el número de renglones en un resultado.

```
int sybase_num_rows ( int variable_resultado )
```

Código 4-34. Sintaxis de la función sybase_num_rows para usarla en PHP.

- **sybase_pconnect**. Abre una conexión persistente. Devuelve un identificador de conexión o FALSE si hay error. Las diferencias entre **sybase_pconnect()** y **sybase_connect()** son que cuando se conecta la primera trata de encontrar una conexión persistente con el mismo servidor, usuario y password, si es encontrada va a regresar ese valor en lugar de abrir otra conexión. Otra diferencia es que la conexión al servidor SQL no se va a cerrar cuando el programa termine, la conexión queda abierta para un uso futuro.

```
int sybase_pconnect ( nombre_servidor, usuario, password [,string charset] )
```

Código 4-35. Sintaxis de la función sybase_pconnect para usarla en PHP.

- **sybase_query**. Manda una consulta al servidor que esta asociado con el identificador de conexión, si el identificador no ha sido especificado, el último identificador abierto es tomado. Si no existe una conexión abierta la función trata de establecer un identificador como si **sybase_connect()** hubiese sido llamado y lo usa. Devuelve el identificador si tuvo éxito, regresa **FALSE** si hay error.

```
int sybase_query ( string query, int identificador )
```

Código 4-36. Sintaxis de la función sybase_query para usarla en PHP.

- **sybase_result**. Da el resultado de una consulta. Devuelve el resultado de una celda en un registro. El argumento campo puede ser el nombre del campo, la tabla haciendo referencia al campo (tabla.campo). Si el nombre de la columna tiene un alias debe usarse el alias en lugar del nombre de la columna. Cuando se trabaja con largos resultados, se debe considerar usar una de las funciones que devuelven un solo registro. Estas funciones regresan el contenido de múltiples celdas en una llamada a la función, son más rápidas que **sybase_result().**

```
string sybase_result ( int variable_resultado, int renglón, mixed campos )
```

Código 4-37. Sintaxis de la función sybase_result para usarla en PHP.

- **sybase_select_db**. Selecciona la base de datos a usar. Regresa **TRUE** cuando hay éxito y **FALSE** cuando hay un error. Si no hay un identificador de conexión especificado, se toma el último por omisión. Si no hay una conexión abierta la función trata de establecer una como si **sybase_connect()** fuese llamada para usarse. Las siguientes llamadas a **sybase_query()** serán para la base de datos activa.

```
bool sybase_select_db ( string nombre_bd, int identificador_conexión )
```

Código 4-38. Sintaxis de la función sybase_select_db para usarla en PHP.

A continuación se encuentra un ejemplo que muestra como insertar en la base de datos de una agenda. Primeramente se necesita poner todos los datos para realizar la conexión a la base de datos este archivo tendrá el nombre de **conexión.php** y deberá estar incluido en todos los archivos que requieran conectarse a la base de datos. El archivo contendrá lo siguiente:

```
<?php
$conexion=sybase_connect ("servidor", "usuario", "password");
sybase_select_db("base_de_datos",$conexion);
?>
```

Código 4-39. Conexión a la base de datos en Sybase nombrando al archivo conexión.php.

Se deberá contar con una forma html en la que podamos introducir los datos de los nuevos miembros de la agenda.

```
<?php
include ("./conexion.php"); #Archivo que contiene los datos de conexión a la base de datos.

?>
<html>
 <head><title>Registro</title></head>
 <body bgcolor="#FFFFFF">
  <center>REGISTRO</center>
  <!--Aquí se escribe a que archivo se mandarán los datos de la forma. -->
  <form name="forma" method="post" action="registro2.php">
   <table width="100" border="1" cellpadding="0" cellspacing="0" align="center">
    <tr valign="middle" align="left">...
     <td align="left" valign="middle" width="76%" height="12">
      <font face="Verdana, Aria l, Helvetica, sans-serif" size="1">
      <select name="pais" size="1">
<?php
#Realiza una consulta a la base de datos a la tabla PAIS
$select = "SELECT * FROM PAIS ORDER BY NOMBRE";
$query = sybase_query($select, $conexion);
$num_ren = sybase_num_rows($query);

for ($i=0;$i<$num_ren;$i++){ #Toma los valores que fueron devueltos en la consulta.
 $id_p = sybase_result ($query, $i, "ID");
 $nombre_p = sybase_result ($query, $i, "NOMBRE");
 #Imprime los valores devueltos por la consulta en una caja de selección.
 if ($id_p == 1) { print "<option value=\"1 \" selected> $nombre_p </option>";  }
 else { print "<option value=\"$id_p\"> $nombre_p </option>"; }
}?>
...
...
      </select> </font>
     </td>
    </tr>
   </table>
  </form>
 </body>
</html>
<?php
#Cierra la conexión a la base de datos
sybase_close($conexion);
?>
```

Código 4-40. Uso de la funciones para interactuar con la base de datos de Sybase.

Datos del programa **registro2.php** el cual recibirá los datos de la forma:

```
<html>
 <head><title>Registro</title></head>
 <body bgcolor="#FFFFFF">
<?php
#Archivo que contiene los datos para realizar la conexión a la base de datos.
include ("./conexion.php");
#Archivo que contiene algunas funciones de validación da datos.
include ("./funciones.php");
#Validación del registro de clientes
if ( ValidaNombres( $nombre ) ) {
 print "<center> El <b>nombre</b> que ingres&oacute; no es v&aacute;lido, debe contener letras
&uacute;nicamente.<br>";
 $error = 1;
...
}
if ($error != 1) {
 #Si no hay errores en los datos ingresados se procede a hacer la inserción de datos.
 $fecha = $anio ."/". $mes ."/". $dia;
 #Primeramente se verifica que el usuario no este registrado en la tabla AGENDA.
 #Se asigna la consulta a una variable.
 $select = "SELECT * FROM AGENDA WHERE UPPER(NOMBRE) = UPPER('$nombre') AND UPPER(AP_PAT) =
UPPER('$ap_pat') AND FECHA_NAC = '$fecha'";
 #Se manda a ejecutar la consulta.
 $query = sybase_query($select, $conexion);
 #El número de columnas devueltas por la consulta se asignan a una variable.
 $existe = sybase_num_rows($query);
 #Si el valor de el número de columnas devuelta por la consulta es cero, el usuario no esta
registrado en la base de datos.
 if ($existe == 0){
  #Se asignan los datos de la inserción a una variable.
  $insert = "INSERT INTO AGENDA (NOMBRE, AP_PAT, AP_MAT, CORREO_ELECT, FECHA_NAC, PAIS) VALUES
('" . $nombre . "','". $ap_pat . "','".$ap_mat. "','" . $email. "','" . $fecha. "',CONVERT
(NUMERIC(2), '". $pais . "'))";
  #Se ejecuta la inserción de datos.
  $query_ins = sybase_query($insert, $conexion);
  #Si fue exitosa o no la inserción se imprime en pantalla.
  if ($query_ins){
   print "<center>El nuevo miembro qued&oacute; registrado.</center>";
  } else {
   print "<center>No se pudo registrar al nuevo miembro en la agenda.</center>";
  }
 } else {
  #Si el miembro ya estaba registrado en la base de datos le informa al usuario del sistema.
  print "<center>Esta persona ya esta registrada en la base de datos.<br></center>";
  $error = 1;
 }
} else {
 print "<center>Por favor presione el bot&oacute;n \"Regresar\" o \"Back\" de su navegador y
realice los cambios pertinentes.</center>";
}
#Cierra la conexión a la base de datos.
sybase_close($conexion);
?>
</body>
</html>
```

Código 4-41. Uso de la funciones para insertar datos en la base de datos de Sybase.

4.4 JAVA SERVER PAGES

Para usar aplicaciones con servlets y jsp se deberá tener en el servidor Java Servlet, Java Server Pages y un servidor Web con capacidad para Servlets como Apache Tomcat.

Java Server Pages (JSP) combina HTML con fragmentos de Java para producir páginas Web dinámicas. Una página JSP es un archivo de texto simple que consiste en contenido HTML o XML con elementos JSP.

Cuando un cliente pide una página JSP del sitio Web y no se ha ejecutado antes, la página es inicialmente pasada al motor de JSP, el cual compila la página convirtiéndola en Servlet, la ejecuta y devuelve el contenido de los resultados al cliente. Cuando ya existe el Servlet solo se manda a llamar, sin necesidad de volver a compilarse.

Ilustración 4-3. Dinámica Cliente / Servidor que usa JSP.

Instrucciones básicas del código JSP.

El código fuente de una página JSP incluye:

- Directivas
- Declaraciones JSP
- Scripts de JS
- Expresiones de JSP

Directivas

Una directiva de JSP proporciona la información del motor de JSP para la página que la pide. Su sintaxis general es **`<%@ directiva {atributo ="valor"} %>`** dónde la directiva debe tener un número de atributos.

Directivas en JSP son:

- **`Page`**: Información para la página. Posee varios atributos:
  - **`language="java"`**. Comunica al servidor el lenguaje que va a ser utilizado en el archivo. Java en el único posible es esta especificación.
  - **`extends="package.class"`**. La variable extends, define la clase padre del servlet generado. Normalmente no es necesario utilizar otras que no sean las clases base del proveedor.
  - **`import="package.*,package.class"`**. Sirve para especificar los paquetes y clases que se quieran utilizar.
  - **`session="true | false"`**. Por omisión session vale true, manteniendo los datos de la sesión para la página.

- **errorPage="pagina_error"**. Página que manejará las excepciones de errores.
- **isThreadSafe="true | false"**. Por omisión vale **true**, le hace señales al motor de JSP para que múltiples pedidos del cliente puedan ser tomadas como uno.
- **info="text".** Información en la página a la que puede accederse a través del método **Servlet.getServletInfo()**
- **isErrorPage="true | false"**. Marca a la página como la página que manejará los errores.

- **Include**: Incluye archivos completos palabra por palabra.
- **Taglib**: La dirección de la biblioteca de etiquetas (tags) que se usará en la página.

Declaraciones JSP

Una declaración de JSP, puede definirse como una declaración de variables y métodos a nivel de clase que son usadas en la página.

Un bloque de declaraciones sería **<%! declaración %>**. Un ejemplo de declaración de script sería el siguiente:

```
<HTML><HEAD><TITLE>Página JSP</TITLE></HEAD>
<BODY>
<%! String strCadena = "x";
int intContador = 0;
%>
</BODY></HTML>
```

Código 4-42. Declaraciones de variables en JSP.

Scripts de JSP

Los scripts son bloques de código Java residentes entre las etiquetas **<%** y **%>**.

Los bloques de código estarán dentro del Servlet generado incluidos en método **_jspService()**. Los scripts pueden acceder a cualquier variable o beans que haya sido declarado.

También hay algunos objetos implícitos disponibles para los scripts desde entorno del Servlet.

Objetos

- **request**. Es la petición del cliente. Es normalmente una subclase de la **case HttpServletRequest**.
- **pageContext**. Los atributos de la página y los objetos necesitan ser accesibles a través de API, para permitir al motor de JSP compilar la página. Pero cada servidor tiene implementaciones específicas de cada uno de esos atributos y objetos. Para solucionar este problema, el motor de JSP utilizar la clase **Factory** para devolver la implementación de clase **PageContext** del servidor. Esta clase PageContext es iniciada con los objetos response y request y algunos atributos de la directiva de la página (**errorpage**, **session**, **buffer** y **auto flush**) y facilita los otros objetos para la página de petición.
- **response**. Es la página JSP de respuesta y es una subclase de **HttpServletResponse**.

- **session**. El objeto de sesión HTTP asociado a la petición.
- **application**. Lo que devuelve el servlet cuando se llama a **getServletConfig().getContext()** **out**. El objeto que representa la salida de texto por pantalla.
- **config**. El objeto **ServletConfig** de la página.
- **page**. Es la forma que tiene la página para referirse a si misma. Se usa como alternativa al objeto this.
- **exception**. Es una subclase libre de **Throwable** que es pasada a la página que maneja los errores.

El siguiente fragmento de código muestra como obtener el valor de una parámetro mediante el objeto **request**, y como pasarlo a una cadena para mostrarlo en pantalla.

```
<%
String strNombre = request.getParameter("nombre");
out.println(strNombre);
%>
```

Código 4-43. Obtener el valor de un parámetro mediante el objeto request usando JSP.

Expresiones de JSP

Las expresiones son una herramienta para insertar código embebido dentro de la página HTML. Cualquier cosa que este entre los tags **<%=** y **%>** será evaluado, convertido a cadena y posteriormente mostrado en pantalla.

La conversión desde el tipo inicial a String es manejada automáticamente. Es importante remarcar que la expresión no termina en punto y coma (**;**). Esto es así porque motor de JSP, pondrá la expresión automáticamente entre **out.println()**.

Las expresiones JSP permiten parametrizar las páginas HTML (es parecido a cuando se parametriza una consulta SQL pero difieren la forma de los valores).

Una y otra vez, en el código de la página HTML, se verán ciclos o condiciones usando código Java, simplemente empezando y acabando las condiciones o ciclos entre los tags **<%** y **%>**.

Ejemplo:

```
<%
for (int i=1; i<10; i++) { %>
<BR>N&uacute;mero <%=i%>
<%
} %>
<BR>Termino la impresi&oacute;n de n&uacute;meros.
```

Código 4-44. Expresión de JSP para el ciclo FOR.

Estructuras de control en JSP

if

Usa la misma lógica que otros lenguajes.

Si a fuese mayor que **b** se mostraría "**a es mayor que b**" ejemplificado en el Código 4-45.

```
if(a > b){
 out.println(“a es mayor que b");
}
```

Código 4-45. Uso de IF en JSP.

else

`else` funciona de la misma manera que en PHP.

else if

Es una extensión del **`if`**, funciona como el **`elseif`** de PHP pero a diferencia de este ambas palabras van separadas.

Ejemplo:

```
if(a > b){
        printf (“a es mayor que b”);
} else if (a == b) {
        printf (“a es igual que b”);
}
```

Código 4-46. Uso de ELSE IF en JSP.

while

Funciona de la misma forma que en otros lenguajes.

Ejemplo:

```
a=1;
b=3;
while (a < b)
{
out.println(“a es menor que b");
a++;
}
```

Código 4-47. Uso de WHILE en JSP.

do ... while

Esta sentencia siempre se ejecutará una sola vez.

Ejemplo:

```
a=5;
b=3;
do {
        out.println (“a es mayor que b”);
        a--;
} while (a > b);
```

Código 4-48. Uso de DO … WHILE en JSP.

for

Su sintaxis es:

```
for (expr1; expr2; expr3)
sentencia
```

Código 4-49. Sintaxis de FOR en JSP.

Ejemplo:

```
for (int i = 1; i < 1 0; i++) {
        out.println (i +" ");
}
```

Código 4-50. Uso de FOR en JSP.

switch

Sintaxis:

```
switch (expr) {
        case valor1:
        sentencias;
        break;

        default:
        sentencias;
        break;
}
```

Código 4-51. Sintaxis de SWITCH en JSP.

Ejemplo:

```
switch (i) {
        case 0:
        out.println("i es igual a 0");
        break;

        case 1:
        out.println("i es igual a 1");
        break;

        case 2:
        out.println ("i es igual a 2");
        break;
}
```

Código 4-52. Uso de SWITCH en JSP

break

Escapa de las estructuras de control cíclicas o del switch.

Acceso a bases de datos

Para trabajar con la base de datos es necesario realizar estos siete pasos:

1. **Cargar el controlador JDBC.** El controlador es la pieza de software que sabe como hablar con la base de datos. Para cargar el controlador todo lo que se tiene que hacer es cargar la clase apropiada. Usamos el método **`Class.forName`** para pasarle la cadena que representa al nombre de la clase correcta para hacer la conexión, el método carga la clase correspondiente.

Esta línea cargaría el controlador para un manejador Sybase.

```
Class.forName("com.sybase.jdbc.SybDriver");
```

Código 4-53. Cargar el controlador para ocupar la base de datos de Sybase en JSP.

Se puede usar este método para cualquier clase en el **CLASSPATH**. La mayoría de los vendedores distribuye sus controladores dentro de archivos **JAR**, debemos estar seguros de que el archivo **JAR** este definido en el **CLASSPATH**.

2. **Definir la conexión URL**. Después de cargar el controlador JDBC es necesario especificar el lugar en donde se encuentra el servidor de la base de datos. Por URL nos referimos al protocolo jdbc: y al nombre del servidor, puerto y nombre de la base de datos. El formato exacto de un URL esta definido en la documentación de cada controlador, a continuación se encuentran dos ejemplos de diferentes controladores:

```
String nombreServ = "servidor.basededatos.com";
String nombreBd = "miBase";
int puerto = 1234;
String oracleURL = "jdbc:oracle:thin:@" + nombreServ + ":" + puerto + ":" + nombreBd;
String sybaseURL = "jdbc:sybase:Tds:" + nombreServ + ":" + puerto + "/" + nombreBd;
```

Código 4-54. Definir la conexión con la base de datos de Sybase usando JSP.

3. **Establecer la conexión**. Es necesario pasar al método **getConnection** de la clase **DriverManager** el URL, el usuario de la base de datos y la contraseña.

```
String usuario = "miUsusario";
String pwd = "miContraseña";
Connection conexion = DriverManager.getConnection(sybaseURL, usuario, pwd);
```

Código 4-55. Establece la conexión con la base de datos de Sybase usando JSP.

La clase Connection incluye los métodos:

- **prepareStatement** (crea una consulta predefinida),
- **rollback** (deshace las consultas de a partir del último **commit**),
- **commit** (termina las consultas desde el último **commit**),
- **isClosed** (revisa si la conexión fue cerrada).

4. **Crear el objeto statement**. El objeto Statement es usado para enviar las consultas y comandos a la base de datos y es creado a partir de **Connection**:

```
Statement statement = conexion.createStatement();
```

Código 4-56. Uso del objeto STATEMENT en JSP.

5. **Ejecutar la sentencia SQL**. Una vez creado el objeto **`Statement`**, se puede usar para enviar consultas SQL utilizando el método **`executeQuery`**, el cual regresa un objeto de tipo **`ResultSet`**. Para modificar la base de datos (**`UPDATE`**, **`INSERT`** o **`DELETE`**) se usa el método **`executeUpdate`**.

```
String consulta = "SELECT columna 1 FROM miTabla";
ResultSet rs = statement.executeQuery(consulta);
```

Código. Ejecutar sentencia SQL en una consulta de una tabla en Sybase usando JSP.

Además se pueden crear consultas parametrizadas, si es que vamos a ejecutar código SQL similar varias veces usando los statements precompilados (**`prepared statements`**).

La idea es crear un objeto parametrizado en una forma estándar que es enviado a la base de datos para hacer la compilación antes de ser usada.

Se utiliza un signo de interrogación para indicar los lugares en donde el valor va a ser sustituido en el statement.

Cada vez que se utilice el statement precompilado simplemente se remplazan los valores con una llamada a **`setXXXX`** que corresponda al tipo de parámetro que se desea usar (**`setInt`**, **`setString`**) y a la entrada del campo.

Ejemplo:

```
Connection conexion =
DriverManager.getConnection(sybaseURL,usuario, pwd);
String template = "UPDATE miTabla SET colFlotante = ?"+ " WHERE colEntero = ? ";
PreparedStatement actualiza = conexion.prepareStatement(template);
actualiza.setDouble(45.67);
actualiza.setInt(4);
actualiza.executeUpdate();
```

Código 4-57. Actualizar registros en Sybase usando JSP.

6. **Procesar los resultados**. La manera más simple de obtener los resultados es procesarlos uno a la vez usando el método **`next`** del **`ResultSet`**, que se mueve uno a uno por los renglones devueltos por la consulta. **`ResultSet`** proporciona varios métodos **`getXXXX`** que toman como argumento el índice o el nombre de una columna.

```
while(rs.next()){
        System.out.println(rs.getInt(1) + " " + rs.getString(2));
}
```

Código 4-58. Procesar el resultado de una consulta en Sybase usando JSP.

7. **Cerrar la conexión**. Para cerrar la conexión solo hay que poner la siguiente línea:

```
conexion.close();
```

Código 4-59. Cerrar la conexión a la base de datos en Sybase usando JSP.

Deberá estar hasta el final, después de haber realizado todas las consultas necesarias.

Ejemplo más completo de un programa:

```
<%@ page contentType="text/html;charset=windows-1 252 import ="java.sql %>
<%
try{
 Class.forName("com.sybase.jdbc2.jdbc.SybDriver");
} catch (ClassNotFoundException e) {
 out.println("<h1>No se encontr&oacute; el controlador:" + e + e.getMessage() + "</h1>" );
}

try{
 String strSQL;
 String host = “dirección IP";
 String port = “puerto de Sybase";
 String url = "jdbc:sybase:Tds:" + host + ":" + port;
 Connection conexion = DriverManager.getConnection (url, “usuario",“password");
 Statement Stm = conexion.createStatement();
 strSQL = "CREATE TABLE AGENDA (ID NUMERIC(3) IDENTITY PRIMARY KEY, NOMBRE VARCHAR(30) NOT
NULL, AP_PAT VARCHAR(25) NOT NULL, AP_MAT VARCHAR(25) NULL, CORREO_ELECT VARCHAR(90) NOT NULL,
FECHA_NAC DATETIME, PAIS NUMERIC(2) NOT NULL)";
 Stm.executeUpdate(strSQL);
 conexion.close();
} catch (Exception e) {
 out.println( "<h1>Excepci&oacute;n: +e+e.getMessage()+"</h1>" );
}
%>
```

Código 4-60. Crear una tabla en una base de datos en Sybase usando JSP.

Lo más común es mostrar los resultados en pantalla de una sentencia **SELECT**, lo podemos hacer de la siguiente forma:

```
<%@ page contentType="text/html;charset=windows-1 252" import ="java.sql.*" %>
<%
try{
 Class.forName("com.sybase.jdbc2.jdbc.SybDriver");
} catch (ClassNotFoundException e) {
 out.println("<h1>No se encontr&oacute; el controlador:" + e + e.getMessage() + "</h1>" );
}

try{
 String host = “dirección IP";
 String port = “puerto de Sybase";
 String url = "jdbc:sybase:Tds:" + host + ":" + port;
 Connection conexion = DriverManager.getConnection (url, “usuario", “password");
 Statement st = conexion.createStatement();
 String select = "SELECT ID, NOMBRE, AP_PAT, AP_MAT FROM AGENDA ORDER BY AP_PAT";
 ResultSet rs = st.executeQuery(select);
 out.println ("<TABLE>");
 out.println("<TR>");

 while (rs.next()) {
  out.println("<TR>");
  out.println("<TD>" + rs.getInt(1)+ "</TD>");
  out.println("<TD>" + rs.getString(2) + "</TD>");
  out.println("<TD>" + rs.getString(3) + "</TD>");
  out.println("<TD>" + rs.getString(4)+ "</TD>");
  out.println ("</TR>");
 }
 out.println ("</TAB LE>");
 conexion.close();
} catch (Exception e) {
 out.println( "<h1>Excepci&oacute;n: "+e+e.getMessage()+"</h1>" );
}
%>
```

Código 4-61. Mostrar el resultado de una sentencia Select usando JSP.

4.5 ACTIVE SERVER PAGES

Active Server Pages (ASP) es una tecnología creada por Microsoft con la que el usuario puede recibir páginas generadas dinámicamente en el servidor. Estas páginas contienen código HTML y fragmentos de código que son utilizados para llevar a cabo la interacción entre el servidor y el navegador.

Ilustración 4-4. Dinámica Cliente / Servidor que usa ASP.

ASP puede conectarse a cualquier motor que disponga de ODBC.

Para procesar una página ASP es necesario un servidor Web de Microsoft.

Se utiliza el archivo **`ASP.DLL`** para interpretar el código, siendo el servidor más extendido Internet Information Server (IIS).

Estos son los servidores de contenidos ASP posibles para plataformas Microsoft

- Internet Information Server 3.0 o superior
- Personal Web Server

Para plataformas Unix es necesario añadir un software que actúe de intérprete siendo algunos de los más conocidos:

- Chilisoft
- Instant ASP

El código ASP va entre las etiquetas **`<%`** y **`%>`**, esto le indicará al servidor cual es la parte que debe procesar.

Contenido de una página ASP

Se puede elegir de entre dos lenguajes con cual trabajar los programas ASP, estos lenguajes son VBScript y Jscript el más usado es el primero y tiene su origen en el lenguaje de programación Visual Basic, Jscript se parece más a JavaScript.

En la propiedad **`LANGUAGE`** deberá especificarse el tipo de lenguaje a utilizar.

```
<%@ LANGUAGE="VBSCRIPT" %> o
```

Código 4-62. Especificar el lenguaje VBSCRIPT a utilizar en ASP.

```
<%@ LANGUAGE="JSCRIPT" %>
```
Código 4-63. Especificar el lenguaje JSCRIPT a utilizar en ASP.

Para especificar un comentario en una página ASP se debe introducir una comilla simple o la palabra **REM**.

```
<%
REM Así se documentan las secciones de los programas
'Cualquier comentario que no, será interpretado por el servidor
%>
```
Código 4-64. Dos formas de poner comentarios en ASP.

Para ocupar ASP se utilizará el lenguaje VBSCRIPT en el transcurso del capitulo.

El tipo de dato que utiliza VBSCRIPT es **Variant** que es una clase especial de datos que puede contener diferentes tipos de información, se comporta como número cuando se utiliza en un contexto numérico, y como una cadena de caracteres cuando se usa en un contexto de cadena, no obstante podemos forzar a que los números se comporten como cadenas poniéndolos entre comillas (**" "**).

Variables

La declaración de variables es opcional, aunque su práctica es una buena costumbre ya que se evitan errores y se facilita la lectura del código, se puede forzar la declaración incluyendo la sentencia

```
<% Option Explicit %>
```
Código 4-65. Forzar la declaración de variables en ASP.

En la declaración se utiliza la palabra reservada **Dim** se pueden anidar varias declaraciones mediante el separador "**,**".

```
<% Dim Variable %>
```
Código 4-66. Sintaxis para la declaración de variables.

Es importante saber como hacer la asignación de variables:

- Las cadenas se asignan entre comillas: **Variable = "Hola"**
- Los números van sin comillas: **Variable = 22**
- Las fechas van entre el símbolo de número: **Variable = #12-1-2003#**

La declaración de arreglos es igual, la única diferencia es que éstos llevan paréntesis a continuación del nombre de la variable, dentro de los paréntesis deberá ir el número de elementos que habrá en el arreglo.

```
Dim Mi_Arreglo(5)
```
Código 4-67. Declaración de arreglos en ASP.

Los elementos del arreglo serán enumerados a partir de cero.

Así **Mi_Arreglo(5)** tendría seis elementos del cero al cinco. Los arreglos pueden cambiar de dimensión (arreglos dinámicos), el arreglo debe declararse sin número de dimensiones visto en el Código 4-68.

```
Dim ArregloDin()
```
Código 4-68. Arreglos dinámicos en ASP.

Con la sentencia **Redim** se determinan las dimensiones:

```
Redim ArregloDin(20)
```
Código 4-69. Determinar las dimensiones de un arreglo en ASP.

Si deseamos conservar los valores almacenados en el arreglo cuando realizamos el redimensionamiento, se deberá añadir la sentencia **Preserve**.

```
Redim ArregloDin(20)
......
Redim Preserve ArregloDin(26)
```
Código 4-70. Preservar los valores almacenados en un arreglo al redimensionar en ASP.

Las constantes se definen con la sentencia **CONST**

```
Const NombreConstante = Valor_que_no_cambia
```
Código 4-71. Sintaxis para la asignación de constantes en ASP.

Estructuras de control

If … Then … Else

Su sintaxis es:

```
If exprThen
        sentencias
Else
        sentencias
End If
```
Código 4-72. Sintaxis de IF … THEN … ELSE en ASP.

Puede o no ocuparse la estructura **Else**.

```
If a > b Then
        Response.Write("a es mayor que b")
End If
```
Código 4-73. Uso de IF … THEN en ASP.

```
If a > b Then
        Response.Write("a es mayor que b")
Else
        Response.Write("a no es mayor que b")
End If
```
Código 4-74. Uso de IF … THEN … ELSE en ASP.

Elseif

Para poner un **elseif** se puede utilizar la siguiente sintaxis:

```
If expr Then
        sentencias
Elseif expr Then
        sentencias
End If
```
Código 4-75. Sintaxis de ELSEIF en ASP.

También puede hacerse así:

```
If expr Then
        sentencias
Else If expr Then
        sentencias
End If
End If
```

Código 4-76. Sintaxis de ELSE IF en ASP.

While … Wend

Funciona como los otros lenguajes. Su sintaxis es:

```
While expr
sentencias
Wend
```

Código 4-77. Sintaxis de WHILE… WEND en ASP.

Ejemplo:

```
I = 0
While I <> 10
        Response.Write ("i = " & i)
        I = I + 1
Wend
```

*Código 4-78. Uso de **WHILE … WEND** en ASP.*

For

Realiza ciertas instrucciones una determinada cantidad de veces. Su sintaxis es:

```
For inicio to fin
sentencias
Next
```

Código 4-79. Sintaxis de FOR en ASP.

Ejemplo:

```
For I = 1 to5
Response.Write ("Esto se repite 5 veces")
Next
```

Código 4-80. Uso de FOR en ASP.

Objetos

Los objetos son programas compilados e instalados en el servidor y que han sido programados para realizar un conjunto de operaciones fácilmente accesibles por otros programas.

- Objeto **Application**: el objeto **Application** se utiliza para compartir información entre todos los usuarios de una aplicación. Tiene dos métodos
    - **Lock** que ayuda a asegurarnos que otros clientes no puedan modificar ninguna de las variables y propiedades de la aplicación hasta que no se llame el método.
    - **Unlock** con el que se anula el bloqueo del objeto **Application** para el resto de clientes.

- Objeto **`Request`**: el objeto **`Request`** se utiliza para tener acceso a la información que se pasa en las peticiones HTTP. Entre dicha información se incluyen los parámetros que se pasan desde los formularios HTML mediante el método **`POST`** o el método **`GET`**, cookies y certificados de cliente. Los métodos más usados son:
    - **`FORM`**. Al consultar formularios los pares nombre/valor se guardan aquí.
    - **`QUERYSTRING`**. En él, la información en forma de pares nombre/valor se agrega y transmite a través de la URL, justo después de la dirección de una página HTML, se guarda ahí y si es preciso se proporciona.
    - **`COOKIES`**. – Aquí se guarda información cookies, si el cliente remite cookies.
    - **`SERVERVARIABLES`**.- Conserva en listas variables del servidor HTTP.
    - **`CLIENTCERTIFICATE`**.- Lista de valores certificados y datos de seguridad.
- Objeto **Response**. El objeto Response se utiliza para controlar la información que se envía al usuario. Esto incluye el envío de información directamente al explorador, la redirección del explorador a otra dirección URL o el establecimiento de valores de las cookies. Algunos de sus métodos son:
    - **`WRITE`**.- Envía información al navegador.
    - **`REDIRECT`**.- Con él se pueden elegir otras páginas ASP pero también páginas HTML sencillas.
- Objeto **`Server`**. El objeto Server proporciona acceso a los métodos y las propiedades del servidor. El método utilizado más frecuentemente es el que crea una instancia de un componente ActiveX (**`Server.CreateObject`**). Con la propiedad Timeout establecemos cuanto tiempo puede tardarse en ejecutar una página.
- Objeto **`Session`**. El objeto **`Session`** permite almacenar la información necesaria para una determinada sesión de usuario. Las variables almacenadas en el objeto Session no se descartan cuando el usuario pasa de una página a otra dentro de la aplicación, si no que dichas variables persisten durante todo el tiempo que el usuario tiene acceso a las páginas de la aplicación. También puede utilizar los métodos de Session para terminar explícitamente una sesión y establecer el periodo de tiempo de espera de inactividad de las sesiones.

Conexión con DSN

La conexión con DSN es la más cómoda, pero sólo se puede utilizar si se tiene acceso al Panel de Control de la máquina servidor.

Pasos para configurar el DSN en el entorno Windows.

- Creamos nuestra base de Datos en Access y la guardamos.
- Luego vamos a Inicio > Configuración > Panel de Control y allí elegimos Fuentes de Datos ODBC.
- Al ingresar nos encontramos con una pantalla que es el administrador de orígenes de datos ODBC. En la solapa DSN de Usuario presionamos el botón Agregar.
- Luego seleccionamos Microsoft Access Driver (*.mdb) y presionamos Finalizar. Ahora se hará la conexión ODBC.
- Presionamos el botón Seleccionar y elegimos nuestra Base de Datos e ingresamos el nombre de la base en el primer campo.

- Por último el botón Aceptar.

```
<%
'Definimos la variable para la conexión.
Dim Conex
Set Conex = Server.CreateObject ("ADODB.Connection")

'Y ya estamos conectados a nuestra base de datos.
Conex.Open "nombre de la BD"
'Se continúa con el programa.
%>
```

Código 4-81. Conexión a una base de datos con DNS en ASP.

Conexión sin DSN

Este tipo de conexión es más "complicada", se hace la conexión a la base de datos mediante comandos.

```
<%
Dim Conex
'Creamos el objeto de conexión
Set Conex = Server.CreateObject ("ADODB.Connection")
Conex.Open "driver={SQL Server};
server=TU_SERVIDOR;
database=NOMBRE_BASE;uid=sa;pwd=xx"
%>
```

Código 4-82. Conexión a una base de datos mediante comandos en ASP.

Mostrar información de la base de datos

```
<%
Dim Conexion, RS

'Abrir la conexión a la base de datos.
Set Conexion = Server.CreateObject ("ADODB.Connection")
Conexion.Open ("DBQ=" & server.mappath("base_datos.mdb") & "; Driver={Microsoft Access Driver
(*.mdb)};")

'Ejecutar consultas en la BD.
Set RS = Server.CreateObject ("ADODB.RecordSet")

'Entre comillas va el nombre de la tabla, y luego de la coma se indica
'la BD que contiene esa tabla.
'El número que le sigue, el "1", es la forma de abrir la tabla 1 es solo lectura.
RS.Open "persona", Conexion, 1

'Mostrar los campos mientras que el RS no termine.
Do While Not RS.EOF
  Response.Write RS("id") &" "
  Response.Write RS("nombre") &" "
  Response.Write RS("ap_pat")&" "
  Response.Write RS("ap_mat") & "<BR>"
  'MoveNext nos moverá al siguiente registro de la tabla.
  RS.MoveNext
Loop

'primero cerramos los objetos y luego los limpiamos.
RS.Close
Conexion .Close
Set RS = nothing
Set Conexion = nothing
%>
```

Código 4-83. Obtener información de una base de datos usando ASP.

Tipos de cursores.

Para el objeto Recordset el cursor es el puntero que nos va a permitir desplazarnos por los registros del recordset. Dependiendo del tipo de cursor será la manera en que podremos movernos por los datos o hacer cambios a los mismos.

Valor	Características
0	Solo nos permite recorrer la tabla en forma secuencial, no podemos movernos hacia atrás y no permite modificaciones.
1	Permite modificaciones y permite movernos en ambos sentidos. Se pueden ver lo cambios realizados por otro recorset a excepción de las altas.
2	Igual al anterior solo que aquí sí vemos todos los cambios realizados por otro recordset.
3	Nos permite movernos en dos sentidos y no permite modificaciones. No se ven los cambios a la tabla originados por otro recordset.

Tabla 4-4. Características de los valores del RECORDSET.

Tipos de bloqueo para el objeto Recordset

Esto nos sirve para evitar que dos usuarios modifiquen el mismo registro a la vez.

Valor	Características
1	No permite modificar los datos de la tabla.
2	Cuando se abra la tabla, nadie más podrá hacerlo.
3	Cierra la tabla cuando se invoque al método Update del objeto Recorset

Tabla 4-5. Tipos de bloque para el RECORDSET.

```
‘Abre el recordset para que se puedan hacer modificaciones y
‘para que cuando se abra la tabla nadie más lo haga.
rs.Open sql,Conexion, 1, 2
‘Asigna el valor de las variables a los distintos campos de la tabla proveedores
rs.Fields("nombre").value = nombre
‘Actualiza el registro del recordset tras haberlo modificado
rs.update
```

Código 4-84. Control en el acceso y manipulación de las tablas usando el RECORDSET en ASP.

4.6 Acceso a los datos de una universidad usando PHP. Caso práctico.

Una universidad tiene un catálogo de carreras y un catalogo de facultades, es necesario crear funciones que ayuden a visualizar en un navegador de Internet la información de estos catálogos.

```
function fn_conectardb(){
  global $conn;
  $dbuser = "postgres";    $dbpass = "";              /* Parámetros de conexión */
  $dbhost = "localhost";   $dbname = "catalogos";     /* a la base de datos */
  $dsn = "pgsql://$dbuser:$dbpass@$dbhost/$dbname";   /* Data Source Name */
  $conn =  DB::connect($dsn);                         /* Conectar a la base de datos */
  if (DB::isError($conn)){ die ($conn -> getMessage()); }
}
```

Código 4-85. Función fn_conectardb escrita en PHP para conectar a una base de datos usando Pear[1].

[1] PEAR, o PHP Extensión and Application Repository. Consiste en una lista bastante grande de bibliotecas de código PHP disponible para reutilizar código escrito previamente por otras personas.

```
function fn_desconectardb(){
  global $conn;
  /* Desconectar de la base de datos */
  $conn->disconnect();
}
```

Código 4-86. Función fn_desconectardb escrita en PHP para desconectar de una base de datos usando Pear.

En el código 4-85 se muestra una forma de realizar una conexión a una base de datos en PostgreSQL, mientras que en el código 4-86 se presenta la forma de desconectarse de la base de datos.

```
function fn_sql_carreras(){
  global $conn;
  fn_conectardb();
  $qry = "SELECT cve_carrera, nombre, FROM cat_carreras";
  $res_carreras = $conn -> query($qry);
  while ($res_carreras -> fetchInto ( $row ) ){
    $arreglo[] = $row[0];
    $arreglo[] = $row[1];
  }
  fn_desconectardb();
  return $arreglo;
}
```

Código 4-87. Función fn_sql_carreras escrita en PHP usando Pear.

```
function fn_sql_facultades(){
  global $conn;
  fn_conectardb();
  $qry = "SELECT cve_facultad,nombre FROM cat_facultades";
  $res_facultades = $conn -> query($qry);
  while ($res_facultades -> fetchInto ( $row ) ){
    $arreglo[] = $row[0];
    $arreglo[] = $row[1];
  }
  fn_desconectardb();
  return $arreglo;
}
```

Código 4-88. Función fn_sql_facultades escrita en PHP usando Pear.

```
function fn_sql_nombrecarrera($cvecarrera){
  global $conn;
  fn_conectardb();
  $qry = "SELECT nombre FROM cat_carreras WHERE cve_carrera = " . $cvecarrera;
  $row = $conn->getRow($qry);
  fn_desconectardb();
  return $row[0];
}
```

Código 4-89. Función fn_sql_nombrecarrera escrita en PHP usando Pear.

```
function fn_sql_nombrefacultad($cvefac){
  global $conn;
  fn_conectardb();
  $qry = "SELECT nombre FROM cat_facultades WHERE cve_facultad = " . $cvefac;
  $row = $conn->getRow($qry);
  fn_desconectardb();
  return $row[0];
}
```

Código 4-90. Función fn_sql_nombrefacultad escrita en PHP usando Pear.

```
print “<select>\n”;
$a_carreras = fn_sql_carreras()
for($k=0;$k<count($a_carreras);$k++){
  print “<option value=\"".$a_carreras[$k][0]."\">".$a_carreras[$k][1]."</option>\n";
}
print “</select>\n”;
```

Código 4-91. Mostrar las carreras en un <select> de HTML.

```
print “<select>\n”;
$a_facultades = fn_sql_facultades()
for($k=0;$k<count($a_facultades);$k++){
  print “<option value=\"".$a_facultades[$k][0]."\">".$a_facultades[$k][1]."</option>\n";
}
print “</select>\n”;
```

Código 4-92. Mostrar las facultades en un <select> de HTML.

```
$int_cvecarrera = “712”;
$nombre_carrera = fn_sql_nombrecarrera($int_cvecarrera);
print $nombre_carrera;
```

Código 4-93. Mostrar el nombre de una carrera.

```
$int_cvefacultad = “24”;
$nombre_facultad = fn_sql_nombrefacultad($int_cvefacultad);
print $nombre_facultad;
```

Código 4-94. Mostrar el nombre de una facultad.

Capítulo 5

FUNDAMENTOS DE SISTEMAS OPERATIVOS

Emplear los comandos y herramientas básicas de administración que intervienen en la configuración de un servidor en diferentes plataformas en las cuales puede residir el RDBMS.

5.1 CONCEPTOS BÁSICOS

Componentes principales de una computadora:

- EL núcleo de todos los sistemas de computadoras es el hardware que trabaja con el software del sistema para desempeñar varias tareas.
- El hardware de la computadora esta formada de varios componentes diferentes, como el CPU, memoria y disco, cada una con un propósito especifico. Para que estos componentes trabajen juntos como un equipo, ellos requieren ser administrados por un sistema operativo.
- El sistema operativo es una colección de programas y archivos con la función primaria de dar instrucciones a la computadora que hacer con el hardware.

Hardware

Los principales cuatro componentes de hardware de una computadora es la memoria RAM, el CPU, los dispositivos de entrada y salida, y el disco duro u otros dispositivos de almacenamiento.

Memoria RAM

La memoria RAM es la memoria principal de la computadora, también conocida como memoria física. Los programas y datos deben ser cargados dentro de la memoria física para que el sistema las procese. El enunciado "El sistema tiene 64 MB de memoria" se refiere a la cantidad de memoria física o RAM que el sistema tiene instalada.

Un programa de software reside en el disco duro y cuando es activado, una imagen o copia del programa es cargado en la memoria RAM.

El programa se queda en la RAM el tiempo que sea necesario. Cuando el programa no es requerido por un tiempo, se pueden sobrescribir copias de otros programas. Si el sistema reinicia o experimenta una baja de energía, los datos de la memoria física son borrados.

CPU

La Unidad Central de procesamiento es el chip lógico de la computadora que ejecuta las instrucciones recibidas de la memoria física, Esas instrucciones son almacenadas en lenguaje binario.

Dispositivos de entrada y Salida

Los componentes de entrada y salida leen las entradas de un dispositivo como el teclado a la memoria, y escribe la salida de la memoria al dispositivo, como una ventana de Terminal.

Los dispositivos de entrada incluyen el teclado y el ratón, El monitor, la impresora y los dispositivos de cinta son los dispositivos de salida primarios.

Disco Duro

El disco duro es un dispositivo de almacenamiento magnético en el cual los archivos, directorios y las aplicaciones son almacenadas.

Sistema operativo

El sistema operativo interpreta las instrucciones del usuario o de las aplicaciones y dice a la computadora que hacer. Manipula las entradas y las salidas, guarda el seguimiento de los datos almacenados en el disco y se comunica con periféricos como el monitor, disco duro, impresoras, módems, etc.

Los tres principales componentes del sistema operativo Linux son:

- El kernel
- El Shell
- El árbol de directorios

El Kernel

El kernel es el núcleo del sistema operativo. Es el programa maestro que administra todos los recursos de la computadora, incluyendo:

- El sistema de archivos
- Administración de dispositivos
- Administración de procesos
- Administración de la memoria

El Shell

El Shell es una interfaz entre el usuario y el kernel. La función primaria del shell es ser un intérprete de instrucciones o comandos. Esto es, el Shell acepta las instrucciones que se ingresan, los interpreta y después las ejecuta. El shell por default en Linux es el BASH (Bourne Again Shell).

El Árbol de directorios

El árbol de directorios se refiere al conjunto de directorios que donde se almacena toda la información del sistema operativo.

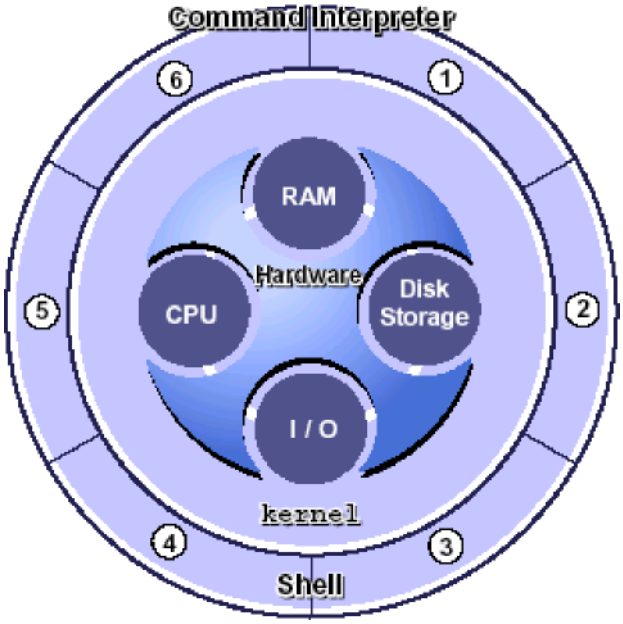


Ilustración 5-1. Componentes básicos de un sistema de cómputo.

5.2 RESPONSABILIDADES DEL ADMINISTRADOR

El administrador del sistema es el responsable de que las operaciones diarias del sistema se encuentren funcionando, por lo que es necesario que el administrador domine las tareas básicas de administración. Entre las tareas más comunes tenemos las siguientes:

- Administración de Cuentas de Usuario y Grupos
- Mantenimiento de la seguridad del sistema
- Configurar Dispositivos
- Instalar y particionar unidades de disco
- Manejar y Administrar Sistemas de archivos
- Calendarizar Tareas Relacionadas al sistema
- Configurar archivos de inicialización del sistema
- Instalación del Sistema Operativo
- Administrar paquetes de software y parches
- Realizar tareas de respaldo y recuperación
- Administración de la recuperación del sistema
- Etc.

Cuenta de usuario

Una importante tarea del administrador del sistema es configurar las cuentas de los usuarios para que estos tengan acceso al sistema. Cada cuenta de usuario se conforma de 5 componentes principales:

- Un nombre de Usuario: Un nombre único en el sistema que le permite acceder a los recursos, comúnmente llamado login.
- Contraseña: Una combinación de seis o más letras, números y/o caracteres especiales que un usuario debe proporcionar para tener acceso al sistema.
- Directorio Home de usuario: Es el directorio en el cual se ubica al usuario una vez que entra al sistema. Ahí puede crear y almacenar archivos.
- Shell: El interprete de comandos proporcionado al usuario.
- Archivos de inicialización del usuario: Scripts en shell que determinan como esta configurado el ambiente de trabajo para cuando el usuario tenga acceso al sistema.

Cada usuario debe tener una cuenta de usuario en el sistema para ingresar. Todas las cuentas de usuario son definidas en el archivo **`/etc/passwd`** y contienen los elementos que identifican cada usuario único en el sistema.

Los administradores del sistema son responsables de crear y administrar las cuentas de usuario.

La cuenta **`Root`**

La cuenta y contraseña del usuario root son configuradas durante el proceso de instalación. Esta cuenta de acceso es usada por los administradores del sistema para realizar tareas administrativas en el sistema.

Administrando cuentas.

Para agregar cuentas de usuario en el sistema es necesario determinar la siguiente información para cada usuario:

- Nombre del login: Cada nombre de usuario debe ser único y debe tener de dos a ocho letras y números. El primer carácter debe de ser una letra y al menos un carácter debe ser una letra minúscula. Los nombres no deben tener guiones bajos ni espacios en blanco.
- Número de identificación del usuario UID: un identificador numérico único de usuario para el sistema. Los UID para usuarios regulares están en el rango de 100 a 60000.
- Número de identificación de grupo GID: un identificador numérico único de grupo al cual pueden permanecer usuarios. Los UID para usuarios regulares están en el rango de 100 a 60000.
- Comentario. Tradicionalmente es el nombre completo del usuario
- Directorio Home. Especifica el directorio en el cual el usuario puede crear y almacenar sus archivos personales.
- Shell de entrada. Define el shell en el cual el usuario una vez que entre al sistema.

La información de las cuentas de los usuarios se almacena en los siguientes archivos:

- **/etc/passwd** **/etc/shadow** **/etc/group**

Usuarios del sistema autorizados tienen una entrada en el archivo:

- **/etc/passwd**

Todas las contraseñas son cifradas y almacenadas en un archivo separado llamado:

- **/etc/shadow.**

Para fomentar el control de las contraseñas de los usuarios manteniendo el archivo **/etc/shadow**.

El archivo **/etc/group** define las grupos que pueden existir en el sistema.

El archivo /etc/passwd

Cada cuenta de usuario tiene una línea en el archivo **/etc/passwd** que contiene siete campos separados por dos punto (**:**).

Usuario 1 :x: 102:1 0:Cuenta del usuario 1: /export/home/usuario1:/bin/ksh

Nombre de usuario. Especifica el nombre usado por el sistema para identificar al usuario. Depende del administrador del sistema, los nombres de usuarios son combinaciones de nombres y apellidos, por ejemplo, a un usuario llamado José Luís Cruz se le asigna el nombre **jlcruz**, **jluisc** o **jluis**. El nombre de usuario debe ser único.

Lugar fijo. Mantiene el campo para el password, el cual es guardado en el archivo **/etc/shadow**, el cual contiene las contraseñas cifradas e información sobre la contraseña (por ejemplo, cuando debe cambiarse la contraseña o la cuenta expire). Este archivo puede ser leído solamente por el administrador del sistema.

- **UID**. Identificador numérico único de usuario
- **GID**. Identificador numérico único de grupo
- **Comentario**. Tradicionalmente es el nombre completo del usuario
- **Directorio Home**. Especifica el directorio en el cual el usuario puede crear y almacenar sus archivos personales.
- **Shell de entrada**. Define el shell en el cual el usuario una vez que entre al sistema.

Cuentas por default

Nombre o login	ID de usuario	Descripción
Root	0	Cuenta de superusuario, no tiene restricciones y sobrescribe todos los logins, protecciones y permisos. Tiene acceso a todo el sistema
Bin	1	Cuenta administrativa que es dueña de la mayoría de los comandos
daemon	2	Cuenta de sistema que controla los procesos en segundo plano
Adm	3	Cuenta administrativa que es dueña de distintos archivos de sistema

Tabla 5-1. Cuentas que generalmente vienen por defecto al instalar un sistema operativo tipo Unix.

El archivo /etc/shadow

Debido a la naturaleza critica del archivo **/etc/shadow**, no es recomendable editarlo directamente. En lugar de eso es recomendable mantener los campos del archivo usando los comandos useradd o passwd. El archivo **/etc/shadow** puede ser leído únicamente por el usuario root.

Ejemplo de entradas del archivo **/etc/shadow**

```
root: LXeoktCoMtwZN :6445::::::
daemon: NP: 6445::::::
bin: NP: 6445::::::
```

Código 5-1. Entradas del archivo /etc/shadow

Cada línea contiene los siguientes nueve campos separados por "**:**"

```
loginID:password:lastchg:min:max:warn:inactive:expire:
```

Código 5-2. Descripción de las entradas en el archivo /etc/shadow.

- **LoginID**: contiene el login del usuario
- **Contraseña**: contiene un conjunto de caracteres que representa la contraseña cifrada.
- **Lastchg**: Indica el numero de días entre el primero de enero de 1970 y la fecha de última modificación de la contraseña
- **Min**: Contiene el mínimo número de días requeridos entre cambios de contraseñas
- **Max**: contiene el máximo número de días en que la contraseña es valida antes de que el usuario la tenga que cambiar
- **Warn**: Contiene el número de días en el que se le advertirá al usuario que su contraseña va a caducar.
- **Inactive**: contiene el número de días inactivos permitidos antes de que la cuenta sea bloqueada.
- **Expire**: Contiene la fecha en la que la cuenta expira, una vez que expire, el usuario no se puede volver a loguear.
- El noveno campo esta reservado para usos futuros.

El archivo /etc/group

Cada usuario debe pertenecer a un grupo, el cual es el grupo primario y que esta definida por el GID en el archivo **/etc/passwd**. Cada grupo puede pertenecer hasta a 15 grupos adicionales conocidos como grupos secundarios, los cuales son especificados en el archivo **/etc/group**.

Ejemplo de entradas del archivo **/etc/group**

```
root::0:root
other::1:
bin: :2:root,bin,daemon
noaccess: :60002:
nogroup: :65534:
```

Código 5-3. Entradas del archivo /etc/group.

Cada línea en el archivo **/etc/group** contiene los siguientes 4 campos separados por un ":"

```
groupname : group-password : GID: username-list
```

Código 5-4. Descripción de las entradas en el archivo /etc/group.

- **groupname**: contiene el nombre asignado al grupo, los nombres de grupo pueden tener un máximo de 8 caracteres*.
- **group-password**: contiene un asterisco o un campo vacío, que por lo regular tenía una contraseña de grupo. Actualmente no es aplicable y esta en desuso.
- **GID**: contiene el número identificador de grupo, deber ser único en el sistema.
- **username-list**: contiene una lista de usuarios separados por comas, que representan que esos usuarios forman parte de ese grupo como grupo secundario.

Creando cuentas de usuario

Puedes agregar cuentas de usuario utilizando el comando useradd. Este programa crea los registros correspondientes en los archivos **/etc/passwd** y **/etc/shadow**. El comando **useradd** copia automáticamente el contenido del directorio **/etc/skel** en el directorio home del usuario

Formato

```
useradd [ -u uid ][ -g gid ][ -G gid [,gid,.. ]][ -d dir ][ -m ]
      [ -s shell ][ -c comment ] loginname
```

Opciones

- **-u id** : Configura el id único para cada usuario
- **-g grupo** : Especifica un gid o nombre de grupo predefinido
- **-G grupo** : Define los grupos secundarios para el usuario
- **-d directorio** : Define la ruta hacia el directorio del usuario
- **-m** : Crea el directorio en el caso que no exista
- **-s shell** : Define la ruta al shell que el usuario tendrá predefinido
- **-c comentario** : Usado para poner el nombre del usuario

- **`-e expiración`** : Configura la fecha de expiración de la cuenta en el formato (mm/dd/yy)
- **`-f inactivo`** : Configura el número de días inactivos permitidos para una cuenta de usuario

Por convención el nombre del directorio home del usuario es el mismo que su login

Ejemplo:

```
useradd -u 100 -g other -d /export/home/newuser1 -m -s /bin/ksh –c "Regular User Account"
newuser1
```

Código 5-5. Agregar un usuario newuser1, con el grupo other, con el directorio /export/home/newuser1, creando el directorio si es que no existe, con el shell /bin/ksh, con el comentario "Regular User Account".

Actualizando las cuentas de los usuarios

Es posible utilizar el comando usermod para modificar los componentes de una cuenta de usuario.

Formato

```
usermod [ -u uid [ -o ] ] [ -g group ] [ -G group [ , group . . . ] ]
        [ -d dir ] [ -m ] ] [ -s shell ] [ -c comment ] [ -l newlogname ]
        [ -f inactive ] [ -e expire ] login
```

Opciones

En general, las opciones para el comando usermod funcionan como el comando useradd, con la excepción de las siguientes opciones:

- **`-l newlogname`** : Cambia el login para una cuenta existente
- **`-m`** mueve el directorio home del usuario al nuevo lugar especificada con la opción **`–d`**

Ejemplo:

```
usermod -d /export/home/guest1 -m -l guest1 newuser1
```

Código 5-6. Modifica el usuario newuser1 para que tenga el directorio /export/home/guest1 moviendo su directorio anterior a una nueva ubicación y cambiándole el login a guest1.

Borrando usuarios

Puedes usar el comando userdel para borrar las cuentas de los usuarios del sistema. Este comando también remueve los directorios home de los usuarios y todo su contenido si es que se requiere.

Formato

```
Userdel –r login
```

Opciones

Se puede usar las siguientes opciones con el comando **`userdel`**:

- **`-r`** borra el directorio home del usuario del sistema de archivos.

Ejemplo:

```
userdel -r guest1
```

Código 5-7. Borra el usuario guest1 y su directorio home

Agregando grupos

Como root, puedes crear nuevos grupos en el sistema local usando el comando groupadd. Este comando agrega entradas para nuevos grupos en el archivo **/etc/group**

Formato

groupadd [-g gid [-o]] groupname

Opciones

Puedes usar las siguientes opciones con el comando **groupadd**

- **-g gid** : Asigna el id de grupo para el nuevo grupo
- **-o** : Permite que el gid se duplique

Ejemplo:

```
groupadd -g 301 class1
```

Código 5-8. Agregar el grupo class1 con un identificador de grupo de 301.

Modificando cuentas de grupo

Puedes usar el comando groupmod para modificar las definiciones del grupo especificado para alterar el archivo **/etc/group**

Formato

groupmod [-g gid [-o]] [-n name] groupname

- **-g gid** : Especifica el nuevo id de grupo
- **-o** : permite que el gid sea duplicado
- **-n nombre** : especifica un nuevo nombre para el grupo

Ejemplo:

```
groupmod -g 400 class1
```

Código 5-9. Modificar el identificador de grupo a 400 del grupo class1.

Borrando grupos

Puedes usar el comando groupdel para borrar un grupo del sistema y lo borra su entrada del archivo **/etc/group**

Formato

groupdel groupname

Ejemplo:

```
groupdel class1
```

Código 5-10. Borrar grupo.

5.4 VARIABLES DE AMBIENTE

Una variable es el nombre que nos refiere a un área de almacenamiento temporal en la memoria. Las variables contienen información usada para personalizar el shell e información requerida para que otros procesos funcionen de manera adecuada. El shell permite que almacenemos valores en variables.

El shell permite definir variables que son exportadas a subprocesos y variables que no.

Acción	**Comando**
Para configurar una variable	VARIABLE = valor
Para eliminar una variable	Uset VARIABLE
Para desplegar todas las variables	Set, env o export
Para desplegar el valor almacenado en una variable	Echo $variable

Tabla 5-2. Acciones de manejo de variables en el Shell de Unix.

Si nombre de variable de shell es valida anteponiéndola al símbolo **$**, el shell interpreta que existe un valor almacenado en la variable y es sustituida en ese punto.

Variable	**Significado**
HOME	Configura el directorio al que nos manda el comando cd cuando lo ejecutamos sin parámetros
LOGNAME	Configura el nombre del usuario
PATH	Especifica la lista de directorios separados por dos puntos(:) en donde se buscaran los comandos que se desean ejecutar
PS1	Especifica el prompt primario del Shell
PS2	Especifica el prompt secundario del Shell
SHELL	Especifica el nombre del Shell

Tabla 5-3. Variables configuradas por el Shell al iniciar sesión.

Si se quiere que los cambios en las variables de personalización se hagan cada vez que inicies sesión debes agregarlas en los archivos de inicialización correspondiente.

Archivos de inicialización

Es posible configurar los archivos de inicialización y ubicarlos dentro del directorio **HOME** de cada usuario. El primer trabajo para modificar un archivo de inicialización es definir las características del ambiente de trabajo para el usuario, rutas en donde puede buscar los comandos, variables de ambiente, etc.

Solo el dueño de los archivos o el usuario root pueden modificar el contenido de los archivos.

Shell	**Archivo de inicialización de todo el sistema**	**Archivo de inicialización del usuario al iniciar sesión**	**Archivo de inicialización del usuario al iniciar un nuevo shell después de tener sesión**	**Ruta del shell**
Bourne	/etc/profile	$HOME/.profile		/bin/sh
BASH	/etc/profile	$HOME/.bash_profile $HOME/.bash_login $HOME/.profile	$HOME/.bashrc	/bin/bash

Tabla 5-4. Archivos de configuración de los shells Bourne y Bash.

Estos archivos permiten al usuario configurar el entorno de su cuenta automáticamente cuando entra en el sistema, cuando arranca un subshell o ejecutar comandos cuando sale del sistema.

Los nombres de estos archivos son **.bash_profile**, **.bashrc**. Si ninguno de estos archivos existe en el directorio del usuario, **/etc/profile** es utilizado por el sistema como archivo de configuración de bash.

bash_profile es el más importante. Es leído y los comandos incluidos en el, ejecutados, cada vez que el usuario entra en el sistema. Cualquier cambio hecho en este archivo no tendrá efecto hasta que salgamos y entremos en el sistema de nuevo. Una alternativa para no tener que salir del sistema es ejecutar el comando source .bash_source.

Bash permite dos sinónimos para este archivo, **.bash_login** (derivado del C shell) y **.profile** (derivado del Bourne y Korn shell). Si **.bash_profile** no existe, el sistema buscara primero **.bash_login** y luego **.profile**. Solamente uno de estos archivos es leído, en el caso que existan simultáneamente.

bashrc es leído cuando el usuario arranca un subshell, escribiendo por ejemplo bash en la línea de comandos. Esto nos permite ejecutar diferentes comandos para la entrada al sistema o para la ejecución de un subshell.

bash_logout es el archivo leído por Bash, cuando salimos del sistema. Podemos definir, por ejemplo que se borren los archivos temporales creados en nuestra ultima sesión o registrar el tiempo que hemos estado utilizando el sistema. Si **bash_logout** no existe, ningún comando será ejecutado a nuestra salida.

5.5 ARRANQUE

A continuación obtendrá las etapas básicas del proceso de arranque para un sistema x86:

- La BIOS del sistema comprueba y lanza la primera etapa del gestor de arranque del MBR del disco duro primario.
- La primera etapa del gestor de arranque se autocarga en memoria y lanza la segunda etapa del gestor de arranque desde la partición **/boot/**.
- La segunda etapa del gestor de arranque carga el kernel en memoria, lo cual en su momento carga los módulos necesarios y monta la partición root para sólo-lectura.
- El kernel transfiere el control del proceso de arranque al programa **/sbin/init**.
- El programa **/sbin/init** carga todos los servicios y herramientas de espacio del usuario y monta todas las particiones listadas en **/etc/fstab**.
- Se le presenta al usuario una pantalla de conexión para el sistema Linux apenas arrancado.

Niveles de ejecución

El sistema de niveles de ejecución **SysV init** provee de un proceso estándar para controlar cuáles programas init lanza o detiene cuando se inicializa un nivel de ejecución. **SysV init** fue escogido porque es más fácil de usar y más flexible que el proceso tradicional init estilo BSD.

Los ficheros de configuración para **SysV init** están en el directorio **/etc/rc.d/**. Dentro de este directorio, se encuentran los scripts **rc**, **rc.local**, **rc.sysinit**, y, opcionalmente, los scripts **rc.serial** así como los siguientes directorios:

```
init.d/
rc0.d/
rc1.d/
rc2.d/
rc3.d/
rc4.d/
rc5.d/
rc6.d/
```

Código 5-11. Procesos de iniciación del sistema en diversos niveles de ejecución.

El directorio **`init.d/`** contiene los scripts usados por el comando **`/sbin/init`** cuando se controlan los servicios. Cada uno de los directorios numerados representa los seis niveles de ejecución predeterminados configurados por defecto bajo Linux.

Los niveles de ejecución son un estado, o modo, definido por los servicios listados en el **`SysV`** directorio **`/etc/rc.d/rc<x>.d/`**, donde **`<x>`** es el número de nivel de ejecución.

La idea detrás de los niveles de ejecución de **`SysV init`** gira alrededor del hecho que sistemas diferentes se pueden usar de formas diferentes. Por ejemplo, un servidor corre de forma más eficiente sin el consumo de recursos del sistema excesivo creado por el sistema X. Otras veces, el administrador del sistema puede necesitar operar el sistema en un nivel más bajo de ejecución para realizar tareas de diagnóstico, como reparar corrupción del disco duro en el nivel de ejecución 1.

Las características de un nivel de ejecución dado determinan qué servicios son detenidos o iniciados por init. Por ejemplo, el nivel de ejecución 1 (modo único usuario) detiene cualquier servicio de red, mientras que el nivel 3 arranca estos servicios. Asignando servicios específicos a ser detenidos o arrancados en un nivel dado, init puede fácilmente cambiar el modo de la máquina sin que el usuario tenga que manualmente arrancar o detener servicios.

Los siguientes niveles de ejecución están definidos por defecto para Linux

- 0 — Parar
- 1 — Modo texto usuario único
- 2 — Sin usar (usuario-definible)
- 3 — Modo texto multiusuario completo
- 4 — Sin usar (usuario-definible)
- 5 — Modo gráfico multiusuario completo (con una pantalla de inicio de sesión basada en X)
- 6 — Rearrancar

Generalmente, los usuarios utilizan Linux al nivel de ejecución 3 o nivel de ejecución 5 — ambos modos multiusuario. Ya que los niveles de ejecución 2 y 4 no son usados, los usuarios a veces personalizan estos niveles para cubrir necesidades específicas.

El nivel de ejecución por defecto para el sistema está listado en **`/etc/inittab`**. Para saber el nivel de ejecución por defecto de un sistema, busque por la línea similar a la que se muestra abajo cerca de la parte superior de **`/etc/inittab`**:

```
id:5:initdefault:
```

Código 5-12. Configuración del nivel de ejecución configurado en el archivo /etc/inittab.

El nivel de ejecución predeterminado en el ejemplo de arriba es cinco, como indica el número después del punto y coma. Para cambiarlo, modifique **/etc/inittab** como usuario root.

Apagar

Para apagar Linux, el usuario root puede ejecutar el comando **/sbin/shutdown**. La página man para shutdown tiene una lista completa de opciones, pero las dos más usadas son:

```
/sbin/shutdown -h now
/sbin/shutdown -r now
```

Código 5-13. Diferentes opciones para apagar un equipo tipo Unix.

Después de apagar todo, la opción -h detendrá la máquina, y la opción -r la reiniciará.

Los usuarios no root pueden usar los comandos reboot y halt para apagar el equipo mientras se está en niveles de ejecución 1 hasta 5. Sin embargo, no todos los sistemas operativos Linux soportan esta característica.

Si la computadora no se apaga, tenga cuidado de no apagar la computadora hasta que aparezca un mensaje indicando que el sistema ha sido detenido.

Si no espera por este mensaje puede significar que no todas las particiones de discos duros han sido desmontadas, y puede llevar a un sistema de archivos corrupto.

Arrancar en modo mono usuario

Una de las ventajas del modo monousuario es que no necesita un disquete o CD-ROM de arranque; sin embargo, no le da la opción de montar sistemas de archivos como sólo lectura o de no montar ninguno.

En el modo monousuario, su computadora arranca en el nivel de ejecución 1. Se montan sus sistemas de archivos locales, pero no se activa la red. Tiene una shell utilizable para hacer el mantenimiento del sistema. A diferencia del modo de rescate, el modo monousuario automáticamente intenta montar su sistema de archivos; no utilice el modo monousuario si su sistema de archivos no se pueda montar exitosamente. No puede usar el nivel de ejecución 1 si la configuración de su sistema está corrupta.

Si su sistema arranca, pero no le permite conectarse cuando ha terminado de arrancar, inténtelo con el modo monousuario. Si está utilizando GRUB, siga los siguientes pasos para arrancar en modo monousuario:

- *Si ha configurado una contraseña GRUB, teclee p e introduzca la contraseña.*
- *Seleccione Linux con la versión del kernel que desee arrancar y escriba e para editar. Se le presentará una lista de ítems en el archivo de configuración para el título que haya seleccionado.*
- *Seleccione la línea que inicia con kernel y teclee e para modificar la línea. Vaya al final de la línea y teclee single como una palabra por separado (pulse [Barra espaciadora] y teclee single). Pulse [Intro] para salir del modo modificar.*
- *De vuelta en la pantalla de GRUB, escriba b para arrancar en el modo monousuario.*
- *Si está usando LILO, en la línea de comandos de LILO (si está usando el LILO gráfico, debe presionar [Ctrl]-[x] para salir de la pantalla gráfica y vaya a la línea de comandos boot: prompt) escriba: linux single*

Usando Linux, todas las comunicaciones de red acontecen entre interfaces, que son dispositivos de red conectados al sistema, configurados de un modo determinado y usando un protocolo, al menos, para intercambiar datos con otros sistemas. Los diferentes tipos de interfaz que existen son tan variados como los dispositivos que los soportan.

Los archivos de configuración para las diferentes interfaces de red y scripts para activarlos o desactivarlos están ubicados en el directorio **`/etc/sysconfig/network-scripts`**. Mientras que la existencia de archivos de interfaces particulares puede diferir de sistema a sistema dependiendo del uso, los tres tipos de archivos diferentes que existen en este directorio, archivos de configuración de interfaz, scripts de control de interfaz y archivos de función de red, funcionan conjuntamente para habilitar Linux para el uso de diversos dispositivos de red disponibles.

Antes de revisar los archivos de configuración de interfaz estudiemos los archivos de configuración principales que usa Linux para configurar la red. La comprensión del papel que desempeñan en la configuración de la red es fundamental para configurar el sistema.

Los principales archivos de configuración de la red son los siguientes:

`/etc/hosts` — El principal propósito de este archivo es resolver los nombres de hosts que no se pueden resolver en otra manera. Se puede usar solamente para resolver nombres de hosts en pequeñas redes sin servidor DNS. Sin tener en cuenta el tipo de red que la computadora use, este archivo contiene un línea que especifica la direcciól̀n IP del dispositivo loopback (127.0.0.1) como por ejemplo localhost.localdomain. Para mayor información consulte la página man del host.

`/etc/resolv.conf` — Este archivo especifica las direcciones IP de los servidores DNS y el dominio de búsqueda. A menos que se haya configurado para algo diferente, los scripts de inicialización de la red llenan este archivo.

`/etc/sysconfig/network` — Especifica la información del routing y del host para todas las interfaces de red.

`/etc/sysconfig/network-scripts/ifcfg-<interface-name>` — Para cada interfaz de red del sistema Linux existe un script de configuración de interfaz para una interfaz de red determinada.

Servicio xinetd

El demonio[1] **`xinetd`** es un super servicio TCP que controla el acceso a un subconjunto de servicios de red populares incluyendo FTP, IMAP y Telnet. También proporciona opciones de configuración específicas al servicio para el control de acceso, registro mejorado, redireccionamiento y control de utilización de recursos.

[1] Demonio, daemon o dæmon (de sus siglas en inglés Disk And Execution Monitor), es un tipo especial de proceso informático que se ejecuta en segundo plano en vez de ser controlado directamente por el usuario (es un proceso no interactivo). Este tipo de programas se ejecutan de forma continua (infinita), vale decir, que aunque se intente cerrar o matar el proceso, este continuará en ejecución o se reiniciará automáticamente. Todo esto sin intervención de terceros y sin dependencia de consola alguna.

Cuando un host cliente intenta conectarse a un servicio de red controlado por xinetd, el super servicio recibe la petición y verifica por cualquier regla de control de acceso wrappers TCP. Si se permite el acceso, xinetd verifica que la conexión sea permitida bajo sus propias reglas para ese servicio y que el servicio no esté consumiendo más de la cantidad de recursos o si está rompiendo alguna regla. Luego comienza una instancia del servicio solicitado y pasa el control de la conexión al mismo. Una vez establecida la conexión, xinetd no interfiere más con la comunicación entre el host cliente y el servidor.

5.7 ADMINISTRACIÓN DE SOFTWARE

Todo el software en un sistema Linux está dividido en paquetes RPM los cuales pueden ser instalados, actualizados o eliminados. Esta parte describe como manejar los paquetes RPM en un sistema Linux usando herramientas gráficas y de línea de comandos

El Administrador de paquetes (RPM) es un sistema de empaquetado abierto que trabaja en Linux además de otros sistemas Linux y UNIX y que está a la disposición de cualquiera.

RPM's

RPM facilita las actualizaciones de sistema para el usuario final. Es posible instalar, desinstalar y actualizar paquetes RPM por medio de comandos breves. RPM mantiene una base de datos de los paquetes instalados y de sus archivos, y usted puede hacer consultas y verificaciones poderosas en su sistema. Si prefiere una interfaz gráfica, puede utilizar Herramienta de administración de paquetes para ejecutar muchos comandos RPM.

RPM tiene cinco modos de operación básicos (sin contar la construcción de paquetes): instalación, desinstalación, actualización, consulta y verificación.

Instalación de RPM

Los paquetes RPM normalmente tienen nombres de archivo como **`foo-1 .0-1.i386.rpm`**. El nombre de archivo incluye el nombre de paquete (foo), versión (1.0), lanzamiento (1) y arquitectura (i386). La instalación de un paquete es tan simple como teclear el siguiente comando en el intérprete de comandos de shell:

```
rpm -ivh foo-1 .0-1 .i386.rpm
```

Código 5-14. Instalación típica de un programa empaqutado en un rpm.

Desinstalación

Desinstalar un paquete es tan simple como instalarlo. Teclee el siguiente comando en el intérprete de comandos de la shell:

```
rpm -e foo
```

Código 5-15. Desinstalar un programa instalado mediante rpm.

Actualización

Actualizar un paquete es parecido a instalarlo. Teclee el siguiente comando en un intérprete de comandos de la shell:

```
rpm -Uvh foo-2.0-1 .i386.rpm
```

Código 5-16. Actualizar un programa instalado mediante rpm.

Consultas

Use el comando rpm -q para hacer consultas a la base de datos de los paquetes instalados. El comando rpm -q foo imprimirá el nombre de paquete, versión y número de lanzamiento del paquete foo instalado: En vez de especificar el nombre del paquete, se pueden usar las siguientes opciones con -q para especificar lo(s) paquete(s) que desea consultar. Se llaman Opciones de especificación de paquetes.

- **`-a`** consulta todos los paquetes actualmente instalados.
- **`-f <file>`** consultará el paquete que posea **`<file>`**. Cuando especifique un archivo, deberá especificar la ruta completa del archivo (**`/usr/bin/ls`**, por ejemplo).
- **`-p <package file>`** consulta el paquete **`<package file>`**.

Verificación

La verificación de un paquete tiene que ver con comparar la información sobre archivos instalados de un paquete con la misma información del paquete original. Entre otras cosas, la verificación compara el tamaño, la suma MD5, los permisos, el tipo, el dueño y el grupo de cada archivo.

```
rpm -Vf /bin/vi
```

Código 5-17. Verifica un paquete que contiene un determinado archivo.

```
rpm –Va
```

Código 5-18. Verifica todos los paquetes instalados.

```
rpm -Vp foo-1 .0-1 .i386.rpm
```

Código 5-19. Verifica un paquete instalado contra un archivo de paquete RPM.

Este último comando puede ser útil si sospecha que sus bases de datos de RPM están dañadas.

Si todo fue verificado correctamente, no habrá salida. Si se encuentran discrepancias, serán mostradas. El formato de la salida es una cadena de ocho caracteres (una c identifica un archivo de configuración) seguido por el nombre del archivo. Cada uno de los ocho caracteres señala el resultado de una comparación entre un atributo del archivo al valor de ese atributo escrito en la base de datos de RPM. Un sólo . (punto) significa que ha pasado la prueba. Los siguientes caracteres señalan que ciertas pruebas no han sido pasadas:

- 5 — MD5 suma de verificación
- S — tamaño de archivo
- L — enlace simbólico
- T — hora de modificación de archivo
- D — dispositivo
- U — usuario
- G — grupo
- M — modo (incluye permisos y tipos de archivos)
- ? — archivo que no se puede leer

Si se ve alguna salida, se debe usa el buen juicio para determinar si se debería quitar o reinstalar el paquete o resolver el problema de otra manera.

Instalación de software utilizando código fuente

Este el el método universal para todas las distribuciones de Linux® ya que funciona tanto en distribuciones basadas sobre RedHat, como Debian, Stampede o Slackware.

Es necesario que lea la documentación que acompaña a dicho paquete y seguir las instrucciones proporcionadas por el autor. Por lo general son necesarios al menos tres pasos:

```
./configure --prefix=/usr
make
make install
make clean
```

Código 5-20. Pasos de instalación comunmente usadas al instalar por código fuente.

./configure

Este prepara el **`Make file`** y configura las opciones de compilación, mismas que en algunos casos pueden resultar demasiado complejas para un usuario novato. Además de verificar si el sistema posee las bibliotecas de desarrollo necesarias para la compilación.

make

Este es el que realiza la compilación del código fuente. El procesos puede durar varios minutos.

make install

Este se encarga de realizar la instalación de los binarios y módulos compilados en los lugares correctos.

make clean

Opcionalmente podemos utilizar este comando para limpiar los remanentes que se originaron por la compilación a fin de recuperar espacio en el disco duro.

make uninstall

Si por alguna razón necesita desinstalar el programa resultante, puede utilizar make uninstall.

5.8 RESPALDOS

Creación y recuperación de respaldos

Podemos crear y recuperar respaldos utilizando las herramientas gzip, zip, gunzip y tar.

Comprimir con Gzip y Zip

Los archivos comprimidos utilizan menos espacio en el disco y se descargan más rápido que los archivos no comprimidos. Puede comprimir archivos Linux con la herramienta de compresión open-source Gzip o con Zip, reconocida por la mayoría de sistemas operativos.

Por convención, a los archivos comprimidos se les da la extensión .gz. El comando **Gzip** crea un archivo comprimido que finaliza con **.gz**; **Gunzip** extrae los archivos comprimidos y suprime el archivo **.gz**.

Para comprimir un archivo, teclee el siguiente comando en el indicador de comandos de la shell:

```
gzip filename.ext
```

Código 5-21. Sitaxis general para comprimir un archivo.

El archivo será comprimido y guardado como **filename.ext.gz**. Para expandir un archivo comprimido, hay que ejecutar la instrucción del Código 5-22:

```
gunzip filename.ext.gz
```

Código 5-22. Sintaxis general para descomprimir un archivo.

El **filename.ext.gz** se borra y se reemplaza con **filename.ext**.

Si intercambia archivos con usuarios no-Linux, le conviene zip para evitar problemas de compatibilidad Linux puede abrir archivos zip o gzip fácilmente, pero los sistemas operativos que no sean Linux puede que tengan problemas.

Para comprimir un archivo con zip, escriba lo siguiente:

```
zip -r filename.zip files
```

Código 5-23. Comprimir un archivo con zip.

En este ejemplo, filename representa el archivo que ha creado y los archivos representan aquéllos que quiere introducir en un nuevo archivo:

Para extraer el contenido de una línea zip, teclee:

```
unzip filename.zip
```

Código 5-24. Extraer el contenido de un archivo .zip.

Puede hacer **zip** o **gzip** múltiples archivos al mismo tiempo. Haga un listado de los archivos con un espacio entre cada uno.

```
gzip filename.gz file1 file2 file3 /user/work/school
```

Código 5-25. Uso de gzip en multiples archivos.

El comando anterior comprimirá **file1**, **file2**, **file3**, y el contenido del directorio **/user/work/school** y los guardará en **filename.gz**.

Archivar con Tar

Los archivos tar ubican diferentes archivos o los contenidos de un directorio o directorios en un archivo. Éste es un buen modo de crear copias de seguridad y archivos. Habitualmente, los archivos tar acaban con la extensión **.tar**.

```
tar -cvf filename.tar ruta/directorio
```

Código 5-26. Crea un archivo tar del contenido de un directorio.

En este ejemplo, filename.tar representa el archivo que está creando y **ruta/directorio** representa los archivos o directorios que quiere introducir en el nuevo archivo.

```
tar -cvf foo.tar /home/mine/work /home/mine/school
```

Código 5-27. Crea un archivo tar usando un nombre de ruta absoluto.

El comando anterior situará todos los archivos en el subdirectorio **/work** y el subdirectorio **/school** en un nuevo archivo llamado **foo.tar** en el directorio donde está trabajando.

```
tar -cvf foo.tar file1.txt file2.txt file3.txt
```

Código 5-28. Sitúa file1.txt, file2.txt y file3.txt en un nuevo archivo llamado foo.tar.

```
tar -tvf foo.tar
```

Código 5-29. Lista los contenidos de un archivo tar.

```
tar -xvf foo.tar
```

Código 5-30. Extrae los contenidos de un archivo tar.

Este comando no suprime el archivo **.tar file**, pero ubica copias del contenido de **.tar** en el directorio en el que está trabajando actualmente.

El comando **tar** no comprime archivos automáticamente. Se puede comprimir archivos **tar** con:

```
tar -czvf foo.tar
```

Código 5-31. Comprimir archivos mediante el comando tar.

A los archivos tar se les da generalmente la extensión **.tgz** y son comprimidos con **gzip**.

```
tar -xzvf foo.tgz
```

Código 5-32. Expande un archivo tar comprimido.

5.9 DIAGNOSTICO DEL SISTEMA.

Se puede hacer uso de una gran gama de herramientas que vienen incluidas en la distribución de Linux para poder diagnosticar cuales son los problema que se presenta en el servidor. A continuación describiremos algunas de ellas y su forma de uso mas frecuente.

Monitoreo de procesos

Se puede hacer uso de los siguientes comandos para ver las tareas que se están ejecutando en nuestro sistema y la cantidad de recursos que ocupa cada uno de ellos.

```
ps
ps –fea
ps –ue usuario
ps
```
Código 5-33. Comandos para listar los procesos que están corriendo en el servidor.

```
top
```
Código 5-34. Comando que muestra la información de los procesos de más demanda.

Con el comando **top** se muestra la información de los procesos dentro del sistema que mas demanda de recursos tienen. Así como porcentajes de memoria y procesador están siendo utilizados en tiempo real.

Espacio en disco

```
df
df –h
df –k
```
Código 5-35. El comando df sirve para determinar cuando espacio se esta utilizando en las particiones.

```
du
du –sh archivo
du –sk archivo
```
Código 5-36. El comando du sirve para determinar cuando espacio tiene un archivo o directorio.

Monitoreo de la red

- **ifconfig** permite configurar las interfaces de red y mostrar información sobre ellas.
- **nmap** realiza un mapeo de puertos a cualquier servidor, para determinar que sistema operativo esta utilizando, o que puertos tiene abiertos.

```
nmap servidor
nmap –p 80-1 024 servidor
```
Código 5-37. Uso de nmap para monitorear puertos.

- **ping** permite enviar paquetes de información a través de la red para determinar si se tiene acceso a una máquina remota.
- **traceroute** permite trazar una ruta de hosts por donde pasa un paquete para llegar al host destino seleccionado.

```
traceroute host
```
Código 5-38. Uso de traceroute.

Como administrador de una máquina Linux, revisa los archivos de registro. Sirven para detectar posibles fallos de seguridad en nuestra máquina Linux. Los archivos de registro se encuentran en:

`/var/log`

- **`messages`**: mensajes de seguridad y autenticación.
- **`cron`**: mensajes generados por el demonio cron.
- **`secure`**: mensajes de seguridad de acceso (conexiones, wrappers...). access.log: mensajes de acceso al servidor WWW, si lo hubiese.
- **`Error_log`**: mensajes de error durante el acceso al servidor WWW, si lo hubiese.
- **`utmp`**: contiene información sobre los usuarios que se encuentran actualmente en el sistema. La información la consultamos con comandos como who, w, users...
- **`wtmp`**: contiene información sobre las entradas y salidas de los usuarios al sistema. La información la consultamos con el comando last.

Si se ve que en estos archivos se repiten registros continuos desde máquinas desconocidas, probablemente la máquina esté siendo objeto de una posible incidencia de seguridad.

El archivo de configuración de los registros es el **`/etc/syslog.conf`** y el demonio encargado de registrar los eventos es **`syslogd`**.

Cron

Cron es un demonio que ejecuta procesos o scripts a intervalos regulares (por ejemplo, cada minuto, día, semana o mes). Los procesos que deben ejecutarse y la hora en la que deben hacerlo se especifican en el archivo **`crontab`**.

En Windows se usa **`Tareas Programadas`** mientras que en Linux/Unix se usa **`cron`**. Los usuarios habilitados para crear su archivo **`crontab`** se especifican en el archivo **`cron.allow`** y los que no lo tienen permitido figuran en **`cron.deny`**.

```
SHELL=/bin/bash
PATH=/sbin:/bin:/usr/sbin:/usr/bin
MAILTO=root
HOME=/

# run-parts
01  * * * * root nice -n 19 run-parts /etc/cron.hourly
50  0 * * * root nice -n 19 run-parts /etc/cron.daily
22  4 * * 0 root nice -n 19 run-parts /etc/cron.weekly
42  4 1 * * root nice -n 19 run-parts /etc/cron.monthly
```

Código 5-39. Ejemplo de un archivo crontab.

Para agregar, quitar o modificar tareas, hay que editar crontab. Esto se hace con **`crontab -e`** que abre un editor y carga el archivo **`crontab`** correspondiente al usuario que este logueado.

Cuando se ejecuta **`crontab`**, se envía un mensaje al usuario que aparece en la variable de entorno **`MAILTO`**, indicándole la tarea realizada.

El símbolo de gato **#** es un comentario, todo lo que se encuentre después de ese carácter no será ejecutado por **cron**.

El momento de ejecución es en la 1ª posición va los minutos (0-59), en la 2ª la hora (0-23), en la 3ª. El día (1-31), en la 4ª el mes (1-12), en la 5ª el día de la semana (0-6), siendo 1=Lunes, 2=Martes,... 6=sábado y 0=Domingo. Para especificar todos los valores posibles de una variable se utiliza un asterisco (*). La última columna corresponde a la ruta absoluto del binario o script que se quiere ejecutar. Con el formato similar a [Minuto] [Hora] [Día] [Mes] [dia semana] [ruta].

5.11 RESPALDO DE BASES DE DATOS CON CRONTAB. CASO PRÁCTICO

Una de las operaciones más comunes en el manejo de bases de datos es realizar respaldos. Una buena práctica es realizarlos periódicamente y en un horario donde haya mínima actividad en las bases de datos que se desee respaldar. El respaldar es un proceso repetitivo y hasta fastidioso si el horario en que se requiere hacer el respaldo es en un día no laboral a altas horas de la noche. Es por ello que se le encarga al administrador del servidor Unix el programar el respaldo de las bases de datos de PostgreSQL y de MySQL que se tienen en la empresa y a continuación en el código 5-40 y 5-41 presenta la solución.

```
#!/bin/bash
MAILTO=root

#Directorio de trabajo
DIRECTORIO="/respaldos/bases_hades"

#Fecha de respaldo
DATE=`date +%d%m%Y`

#Seleccionar las bases a respaldar (todas menos template0 y template1
LIST=$(psql -lt -U postgres | awk '{print $1}' | grep -vE '^-|^List|^Name|template[0|1]')

#Respaldar las bases de PostgreSQL y ubicar el respaldo en el directorio DIRECTORIO
for d in $LIST
do
pg_dump $d -D -U postgres > $DIRECTORIO/$d-$DATE.psql
gzip $DIRECTORIO/$d-$DATE.psql
done


#Respaldar las bases de MySQL
mysqldump becas -u root --password=123 > $DIRECTORIO/becas -$DATE.mysql
gzip $DIRECTORIO/becas-$DATE.mysql

mysqldump files -u root --password=123 > $DIRECTORIO/files-$DATE.mysql
gzip $DIRECTORIO/files-$DATE.mysql

mysqldump planes -u root --password=123 > $DIRECTORIO/planes-$DATE.mysql
gzip $DIRECTORIO/planes-$DATE.mysql

mysqldump desarrollo -u root --password=123 > $DIRECTORIO/desarrollo-$DATE.mysql
gzip $DIRECTORIO/desarrollo-$DATE.mysql

mysqldump mysql -u root --password=123 > $DIRECTORIO/mysql-$DATE.mysql
gzip $DIRECTORIO/mysql-$DATE.mysql
```

Código 5-40. Script backup_bases.sh que respalda bases tanto de PosgreSQL como de MySQL y comprime los respaldos.

```
0 2 * * 0 root /respaldos/cron/backup_bases.sh
```

Código 5-41. Instrucciones del cron para ejecutar backup_bases todos los domingos a las 2 de la mañana los domingos.

Capítulo 6

HABILIDADES DIRECTIVAS PARA ADMINISTRADORES

Comprender la función del liderazgo y las formas para evitar la resistencia al cambio organizacional. Mejorar la capacidad para realizar presentaciones efectivas. Lograr una mejor administración del tiempo en relación con las juntas de trabajo. Son habilidades directivas necesarias en todo administrador de bases de datos.

6.1 ¿PARA QUÉ LAS HABILIDADES DIRECTIVAS?

- Las personas dedicadas a diversos aspectos relacionados con la computación deben tener claro que su trabajo forma parte de un sistema que integra recursos técnicos, humanos y materiales destinados a lograr objetivos comunes.
- Administrar es alcanzar un objetivo mediante el esfuerzo humano coordinado, de ahí la necesidad de desarrollar la habilidad para comunicarse con las personas con las que colaboramos, comprender las necesidades del grupo en lo individual y lo general, descubrir las posibilidades técnicas y humanas para obtener lo mejor de cada persona y el mejor funcionamiento grupal.
-

6.2 DIFERENCIAS ENTRE JEFE Y LÍDER

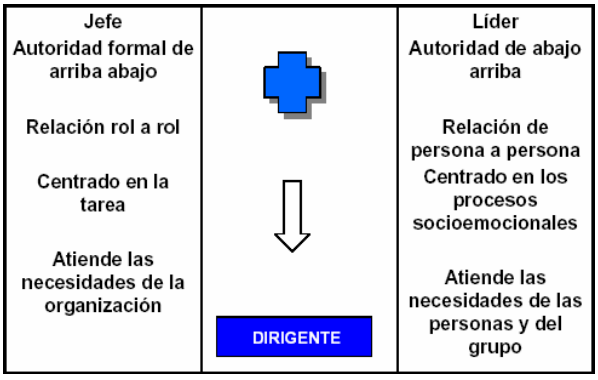


Ilustración 6-1. Diferencias entre jefe y líder.

Jefe		Líder
Formal Conferida por el sistema	Autoridad	Informal Otorgada por el equipo
Por premio o castigo	Resultados	Por la confianza y convencimiento del grupo
Según la estructura	Duración	Creado y mantenido por el grupo y sus objetivos
De la autoridad formal	Poder	De influir, reconocido y aceptado por el grupo
Son externos al grupo el cual acepta los objetivos a cambio de lo que se le paga	Objetivos	Son de todo el grupo y se reconoce que el líder es capaz de guiarlos hacia el cumplimiento

Tabla 6-1. Roles del jefe vs. el líder

6.3 DIFERENCIAS ENTRE ADMINISTRADOR Y LÍDER

Administrador	Líder
administra	innova
copia	crea
mantiene	desarrolla
manda	convence
acepta la realidad	investiga la realidad
enfoque a sistemas y estructuras	enfoque en la gente
control	confianza
cómo y cuando	qué y por qué
hace las cosas bien	hace las cosas correctas
da la hora	construye relojes

Tabla 6-2. Diferencias entre administrador y líder.

Pasar de jefe a líder

Pasar de jefe a líder significa reconocer tres áreas de resultados igualmente importantes:

- **Tareas**: Donde se trata de lograr resultados específicos, en la cantidad, con calidad y la oportunidad requeridas.
    - El reto actual: Asegurar una mayor productividad y calidad.
- **Grupos**: Donde se requiere ocuparse de los individuos a su cargo para asegurar su capacitación, motivación y progreso.
    - **El reto actual**: Promover una mayor preparación, involucramiento y responsabilidad.
- **Equipos**: Donde debe asegurarse su integración y desarrollo, así como un clima de comunicación y colaboración.
    - **El reto actual**: Es conseguir una mayor participación y trabajo en equipo.

6.4 LIDERAZGO

El líder debe trabajar con un enfoque de desarrollo:

- **Inspirar confianza en su gente**: Cambiar la actitud tradicional de intervenir para dar instrucciones y corregir, a otra capaz de inspirar una visión común, modelar con el ejemplo,

reconocer y recompensar todos los esfuerzos, son las únicas vías para ganar la confianza de su gente y por tanto, reforzar consistentemente el cambio.

- **Centrarse en las personas**: Olvidar la tendencia de considerar a las personas como cosas o simplemente como recursos humanos y responder a las necesidades que cada individuo tiene como ser humano.
- **Desarrollar un equipo**: Derribar las barreras entre funciones y personas, trabajar de manera interdisciplinaria, aumentar la participación y ser capaz de utilizar y fomentar los mejores recursos del grupo, son habilidades indispensables de un líder empeñado en mejorar la calidad y productividad.
- **Generar un clima motivador**: Los procesos de calidad y el buen servicio se basan en la participación de la gente y estimular su iniciativa para que actúen con responsabilidad sin necesidad de vigilancia.

Entonces, se puede señalar que Liderazgo es el proceso de influir, guiar o dirigir a los miembros del grupo hacia el éxito en la consecución de metas y objetivos organizacionales.

Tipos de liderazgo.

Existe diferentes tipos de liderazgo entre los que destacan

Administración de Club Campestre

El interés por las necesidades de los empleados para lograr relaciones satisfactorias, conduce a un ambiente y ritmo de comodidad y sociabilidad muy agradable en el trabajo.

Estilo de mando caracterizado por trabajo en equipo

La realización del trabajo se debe a la dedicación plena de los empleados. La interdependencia basada en un interés común de la empresa, conduce a relaciones de respeto y confianza.

Estilo de mando basado en el hombre-organización

Es posible obtener un nivel adecuado de producción estableciendo un equilibrio entre la necesidad de obtener unidades y la de mantener la moral del personal en un nivel satisfactorio.

Estilo de mando empobrecido

Su preocupación por las necesidades de la organización o las personas que colaboran en ellas es mínima. No se involucra. Cumple con los requisitos de su trabajo y se involucra lo menos posible con la gente. Hace el mínimo necesario para acumular antigüedad sin ninguna consideración real por hacer una contribución. La necesidad de conservar el trabajo personal es lo que conduce a la motivación negativa.

Estilo de mando autoridad – obediencia

Condiciones de trabajo organizadas de forma que la producción se consigue a través de una mínima interferencia del elemento humano.

6.5 EL CAMBIO

El cambio es una modificación o movimiento de un plano o estado a otro que implica dejar a un lado determinadas estructuras, procedimientos, comportamientos, etc., para adquirir otras que permitan la adaptación al contexto en el cual se encuentra el sistema, y así lograr una estabilidad que facilite la eficacia y efectividad en la ejecución de acciones.

Etapas que facilitan el proceso de cambio de sistemas

Existen tres etapas esenciales y secuenciales que facilitan el proceso de cambio de sistemas:

Descongelamiento

Implica desequilibrio, insatisfacción, toma de conciencia de la situación. Se perciben procedimientos, hábitos, costumbres, actitudes que obstaculizan la adaptación. Genera ansiedad y dudas del propio modo de conducirse.

Se tiene la necesidad de identificar las estructuras sujetas al cambio, satisfacer nuevas necesidades, equilibrio y logro de una situación deseada.

Movimiento

Genera desequilibrio, inestructura, inestabilidad, inseguridad, incertidumbre. Se tiene la necesidad de voltear la mirada al entorno, generar información, buscar alternativas, seleccionar alternativas, abandonar viejas estructuras o esquemas. Se requiere mayor adaptación y adoptar nuevos esquemas y estructuras.

Recongelamiento

Se establece claridad en la situación, se llega a un equilibrio y se tiene mayor adaptabilidad. Esto implica integrar nuevos esquemas, establecer contacto genuino con la opción elegida, considerar el efecto del cambio en el resto de los subsistemas y una cierta permanencia en esta nueva situación.

Resistencia al cambio

El cambio puede generar resistencia. Esta resistencia al cambio es una reacción esperada por parte del sistema, el cual estando en un periodo de equilibrio, percibe la amenaza de la inestabilidad e incertidumbre que acarrean consigo las modificaciones. Por tanto, se puede definir como las fuerzas restrictivas que obstaculizan un cambio.

Los hábitos

Son un obstáculo por el grado de arraigo que los caracteriza y porque resultan una medida de economía, ya que al aplicarlos nos evitamos reflexionas en cada situación, de tal forma que un cambio de hábito implica mayor inversión de energía, o sea, llevar a cabo un esfuerzo adicional.

Miedo a lo desconocido

El mañana no está aquí, por lo tanto resulta un misterio, una fantasía; de tal modo que muchas personas prefieren no enfrentar el riesgo de encontrar sorpresas, sean éstas buenas o mala, por lo que se inclinan a permanecer en el lugar en donde están hoy.

Apego a lo conocido

"Más vale malo por conocido que bueno por conocer". Una vez vivenciado el éxito que se obtiene con determinada acción, se convierte en hábito y se instala dentro de los modelos típicos de comportamiento.

Tendencia a conservar la estabilidad

Existe una gran tendencia a mantener un ambiente predecible, estructurado y seguro, aunque no podemos negar la necesidad de explorar y arriesgar; sin embargo, se puede afirmar que entre más se aferre el individuo a sus modelos de comportamiento se resistirá al cambio.

Apego a lo elaborado por el individuo mismo

Cuando un sujeto es el autor de determinada situación, el cambio se convierte en un desprestigio y poca valoración de su esfuerzo.

Disminuir la resistencia al cambio

Tras ver que existe la resistencia al cambio, la pregunta obligada es ¿Cómo disminuir la resistencia al cambio?

- Escuchar las expresiones de resistencia y manifestar empatía.
- Generar información de hechos, necesidades, objetivos y efectos del cambio.
- Ajustar el modo de implantación del cambio a las características de la organización.
- Reducir incertidumbre e inseguridad.
- Buscar apoyos que fomenten la credibilidad.
- No combatir la resistencia, es sólo un síntoma… hay que buscar la raíz.
- ¿Cómo disminuir la resistencia al cambio?
- No imponer el cambio.
- Hacer un cambio participativo.
- Establecer el diálogo e intercambiar y confrontar percepciones y opiniones.
- Plantear problemas, no soluciones unilaterales.
- Realizar cambios continuamente, aun cuando sean pequeños.
- Crear un compromiso común.
- Plantear el costo – beneficio del cambio.

6.6 COMUNICACIÓN

Se ha señalado que la opinión que tenemos de los demás suele basarse en tres características principales:

- Contenido verbal = 7 %
- Interés del discurso oral = 38 %
- Lenguaje corporal = 55 %

Antes de empezar

- Apariencia
- Voz
- Volumen
- Tono
- Entonación
- Pronunciación
- Calentamiento
- Conocer a la audiencia

Lenguaje corporal

- Quieto en un punto determinado, inclinado hacia un lado.
    - Mensaje oculto: «Estoy aburrido y preferiría estar en otra parte».
    - Solución: Cuando esté quieto, equilibre su peso y nivele las caderas.
- Inclinado encima del atril.
    - Mensaje oculto: Estoy demasiado cansado para mantenerme de pie, me da pereza hacer esto.
    - Solución: Cuando utilice un atril, mejor colóquese a un lado de éste.
- Sentado encima de la mesa con las notas, el retroproyector, etc.
    - Mensaje oculto: «Aquí no tengo que esforzarme, porque soy más importante que los demás».
    - Solución: No importa lo relajado que se sienta, ¡póngase de pie!
- En definitiva, no existe ninguna postura ideal. Por tanto, haga lo que le parezca bien, dentro de lo razonable.

El movimiento

- No cerrar las manos delante de usted
- No colocar las manos tras la espalda
- No cerrar los puños a menos que quiera generar un efecto
- No cruzar los brazos
- No jugar con objetos

Trabajar con diapositivas

Cuando muestre una diapositiva, alterne su atención entre la diapositiva y el contacto visual con la audiencia. Hable con sus participantes, no con la pantalla.

Preguntas

Los secretos básicos para abordar preguntas

- La reacción natural es ir preparando la respuesta a una pregunta mientras la persona que pregunta aún está hablando. No obstante, resulta imposible prestar atención simultáneamente a dos líneas de pensamiento diferentes.
- Reformule la pregunta y luego repítala íntegramente a la audiencia antes de responderla. Dirija la respuesta a la audiencia en general.
- Evite caer en la trampa del monopolio de un grupo de personas interesadas en formular preguntas.
- Una vez haya aceptado y repetido la pregunta, y mire a la persona interesada (en un 20 o 25 %), el resto (entre un 75 y un 80 %) debe dirigirse a la audiencia en general.

¿Qué hacer cuando no se sabe la respuesta?

- Sea sincero. Decir «no lo sé» o incluso mejor, «No lo sé pero intentaré averiguarlo».
- «Ahora mismo no lo sé, pero intentaré averiguarlo al terminar el descanso de la comida/mañana por la mañana, etc.»

6.7 REUNIONES EFECTIVAS

Administrar el tiempo

- Asumir el control de los requerimientos que te hacen en tu tiempo disponible.
- Asegurar que el uso que haces de tu tiempo –una distribución limitada – se ajuste de la mejor manera a tus metas y necesidades personales.
- Establecer tus metas personales. Si no están claras, no tienes un marco de referencia para poder distribuir el tiempo.

Ejercicios prácticos

- Creador de agendas
- Basura adentro, basura afuera
- Descargando ideas
- Control de multitudes
- No cumplimos
- Nota bene
- Aplicando control natal
- Los filtros de tu mundo
- Diferentes filtros
- Flexibilidad
- Crear un clima de confianza – Rapport

6.8 DIAGNÓSTICO: ¿COMO NOS ENFRENTAMOS AL CONFLICTO Y EL CAMBIO?

Tus deseos difieren de los de otras personas ¿Cómo responderías? (Escoger solo un inciso por apartado):

1. A. En ocasiones dejo a otros la responsabilidad de resolver el problema

   B. En lugar de negociar sobre los aspectos que no estamos de acuerdo, yo trato de enfatizar los puntos en los que sí estamos de acuerdo

2. A. Trato de encontrar una solución en que ambos cedamos

   B. Intento manejar todos mis intereses conforme a los de la otra persona

3. A. Habitualmente intento alcanzar mis metas con firmeza

   B. Intento apaciguar los sentimientos de la otra persona y conservar nuestra relación

4. A. Trato de encontrar una solución en que ambos cedamos

   B. Algunas veces sacrifico mis propios deseos por los deseos de la otra persona

5. A. Conscientemente busco la ayuda de la otra persona para encontrar una solución

   B. Trato de hacer lo que sea necesario para evitar tensiones inútiles

6. A. Trato de evitar crearme una situación desagradable

   B. Trato de triunfar en mi postura

7. A. Trato de posponer el asunto hasta que tenga tiempo para pensarlo

   B. Renuncio en ciertos puntos para ganar en otros

8. A. Generalmente soy firme en la persecución de mis metas

B. Intento expresar abiertamente todas las preocupaciones y problemas de inmediato

9. A. Siento que siempre vale la pena preocuparas por las diferencias

B. Me esfuerzo por ganar el argumento

10. A. Soy firme para lograr mis metas

B. Intento encontrar una resolución en que ambos cedamos

11. A. De inmediato intento sacar a luz todos los intereses y problemas

B. Intento apaciguar los sentimientos de la otra persona y conservar nuestra relación

12. A. En ocasiones evito expresar opiniones que puedan crear controversia

B. Intento por encontrar una solución en que ambos cedamos

13. A. Propongo transigir

B. Presiono para dejar bien clara mi opinión

14. A. Le explico mis ideas a la otra persona y le pido que explique las suyas

B. Intento demostrar a la otra persona la lógica y beneficios de mi postura

15. A. Intento apaciguar los sentimientos de la otra persona y conservar nuestra relación

B. Intento hacer lo que sea necesario para evitar tensiones

16. A. Trato de no lastimar los sentimientos de la otra persona

B. Intento convencer a la otra persona de los méritos de mi postura

17. A. Usualmente persigo mis metas con firmeza

B. Intento hacer lo que sea necesario para evitar tensiones inútiles

18. A. Dejo que la otra persona sostenga su punto de vista, si esto la hace feliz

B. Dejo que la otra persona gane algunos argumentos si me permite ganar a mí en los míos

19. A. De inmediato intento sacar a la luz todos los intereses y problemas

B. Intento posponer los problemas hasta que he tenido tiempo de pensar

20. A. De inmediato intento tratar nuestras diferencias

B. Intento encontrar una justa combinación de puntos ganados y perdidos para ambos

21. A. Al abordar las negociaciones intento ser considerado hacia los deseos de la otra persona

B. Siempre me inclino a tener una abierta discusión del problema

22. A. Intento encontrar una postura intermedia entre su opinión y la mía

B. Afirmo mis deseos

23. A. Con frecuencia me interesa satisfacer todos nuestros deseos

B. En ocasiones dejo que otros asuman la responsabilidad de resolver el problema

24. A. Si la opinión de la otra persona parece ser muy importante para él, intentaría cumplir con sus deseos

B. Intento hacerlo transigir

25. A. Intento mostrarle la lógica y beneficios de mi persona

B. Al abordar las negociaciones, intento ser considerado hacia los deseos de la otra persona

26. A. Propongo que ambos transijamos

B. Casi siempre me interesa satisfacer todos nuestros deseos

27. A. En ocasiones evito asumir posturas que puedan crear controversia

B. Dejo que la otra persona sostenga sus puntos de vista, si esto la hace feliz

28. A. Usualmente persigo mis metas con firmeza

B. Usualmente busco la ayuda de la otra persona para encontrar una solución

29. A. Propongo que ambos transijamos.

B. Siento que no siempre vale la pena preocuparse por las diferencias

30. A. Intento no lastimar los sentimientos de la otra persona

B. Siempre comparto el problema con la otra persona con el fin de llegar a una solución

En situaciones de conflicto podemos describir el comportamiento de una persona con base en dos dimensiones: A) afirmación: grado hasta el cual una persona intenta satisfacer sus propios intereses, B) cooperación: el grado hasta el cual la persona intenta satisfacer los intereses de la otra persona.

Cuantificar en la siguiente tabla la cantidad de respuestas por columna.

	Competir	Integrar	Transigir	Evadir	Complacer
1				A	B
2		B	A		
3	A				B
4			A		B
5		A		B	
6	B			A	
7			B	A	
8	A	B			
9	B			A	
10	A		B		
11		A			B
12			B	A	
13	B		A		
14	B	A			
15				B	A
16	B				A
17	A			B	
18			B		A
19		A		B	
20		A	B		
21		B			A
22	B		A		
23		A		B	
24			B		A
25	A				B
26		B	A		
27				A	B
28	A	B			
29			A	B	
30		B			A
Total					

Tabla 6-3. Resultado del autodiagnóstico ¿Como nos enfrentamos al conflicto y el cambio?

6.9 DIAGNÓSTICO DE ESTILOS DE LIDERAZGO

¿Qué haría este líder en las siguientes situaciones?

1. Situación: Sus subordinados no están respondiendo últimamente a su conversación amistosa y obvia preocupación por su bienestar. El rendimiento de sus subordinados desciende rápidamente.

- A. Insista en el uso de procedimientos uniformes y en la necesidad del cumplimiento de las tareas
- B. Esté disponible para tratar los asuntos pero no presiones para participar en la discusión
- C. Hable con los subordinados y luego establezca los objetivos
- D. No intervenga intencionalmente

2. El rendimiento observable de su grupo está aumentando. Usted ha estado haciendo lo posible por asegurarse de que todos los miembros conozcan sus responsabilidades y sus niveles de rendimiento que de ellos se esperan.

- A. Inicie una interacción amistosa, pero continúe asegurándose que todos los miembros estén al tanto de sus responsabilidades y de los niveles de rendimientos que ellos se esperan
- B. No realice ninguna acción determinada

- C. Haga lo que pueda para que el grupo se sienta importante e involucrado en los asuntos de la empresa
- D. Dé importancia a las tareas y fechas límite

3. Los miembros de su grupo no pueden solucionar un problema por si solos. Normalmente usted los ha dejado solos. El rendimiento y las relaciones interpersonales han sido buenos.

- A. Trabaje con el grupo y trate de solucionar los problemas
- B. Deje que el grupo resuelva solo
- C. Actúe rápida y firmemente para corregir la situación y dirigir al grupo
- D. Anime al grupo para que trabaje en el problema y esté a su disposición para cualquier discusión

4. Usted está considerando un cambio, sus subordinados tienen excelentes antecedentes de logros. Ellos respetan la necesidad de cambio.

- A. Permita que el grupo se involucre en el desarrollo del cambio, pero no sea demasiado autoritario
- B. Anuncie los cambios y luego haga que se cumplan bajo una estrecha supervisión
- C. Permita al grupo que formule su propia dirección
- D. Incorpore las recomendaciones del grupo, pero dirigiendo usted mismo el cambio

5. El rendimiento de su grupo se ha deteriorado en los últimos meses. Los miembros no se preocupan por lograr los objetivos. La redefinición de los roles y responsabilidades ha ayudado en el pasado. Han necesitado que se les recuerde continuamente que tienen que cumplir con sus tareas a tiempo

- A. Permita que el grupo formule su propia dirección
- B. Incorpore la solución a las recomendaciones del grupo, pero vigilando que se alcancen los objetivos
- C. Redefina los roles y responsabilidades y supervise estrictamente
- D. Permita que el grupo se involucre en la determinación de los roles y responsabilidades pero no sea demasiado autoritario

6. Usted entra a ocupar una posición en la organización donde las operaciones son eficientes. El administrador anterior controla estrictamente la situación. Usted quiere mantener una situación productiva, pero desearía comenzar a humanizar el ambiente.

- A. Haga lo que pueda para que el grupo se sienta importante e involucrado en los asuntos
- B. Participar con el grupo en el desarrollo del cambio pero dejar que los miembros organicen la realización
- C. Mostrarse dispuesto a hacer los cambios recomendados, pero manteniendo el control de la realización de los mismos
- D. Evitar la confrontación, dejando las cosas como están

7. Este líder está considerando algunos cambios importantes en su estructura organizativa. Miembros del grupo han hecho sugerencias sobre la necesidad de cambio. El grupo ha sido productivo y ha demostrado flexibilidad en sus operaciones.

- A. Definir el cambio y supervisarlo estrictamente
- B. Participar con el grupo en el desarrollo del cambio pero dejar que los miembros organicen la realización
- C. Mostrarse dispuesto a hacer los cambios recomendados, pero manteniendo el control de la realización de los mismos
- D. Evitar la confrontación, dejando las cosas como están

8. El rendimiento del grupo y sus relaciones interpersonales son buenos. Este líder se siente algo inseguro por su falta de dirección del grupo.

- A. Dejar al grupo solo
- B. Discutir la situación con el grupo y luego iniciar con el los cambios necesarios
- C. Tomar medidas para dirigir a los subordinados para que trabajen de una manera determinada
- D. Mostrar que respalda al grupo en la discusión de la situación, pero sin ser demasiado autoritario

9. Este líder ha sido nombrado por su superior jefe de un grupo que ha tardado bastante en presentar sus recomendaciones respecto de la ejecución de ciertos cambios. El grupo además no sabe con claridad cuáles son sus objetivos. La asistencia a las sesiones ha sido escasa. Sus reuniones se han convertido casi en fiestas sociales. Sin embargo, potencialmente tienen el talento necesario para ayudar.

- A. Dejar que el grupo busque solo las soluciones a sus problemas
- B. Incorporar a la solución las recomendaciones del grupo, pero vigilar que se alcancen los objetivos.
- C. Redefinir los niveles de calidad y supervisar cuidadosamente
- D. Permitir que el grupo intervenga en la determinación de los objetivos, pero sin ser demasiado autoritario

10. Los subordinados – normalmente capaces de responsabilizarse – no están respondiendo a la reciente redefinición de niveles de calidad del líder

- A. Redefinir los niveles de calidad y supervisar cuidadosamente
- B. Evitar la confrontación a través de no aplicar presión; dejar la situación sin intervenir
- C. Incorporar a la solución las recomendaciones del grupo, pero vigilar que se alcancen los niveles de calidad

11. Este líder ha sido ascendido a una nueva posición. El jefe anterior no se involucraba en los asuntos del grupo. El grupo ha manejado bien sus tareas y la dirección. Las interrelaciones del grupo son buenas.

- A. Hacer que los subordinados se vean involucrados en la toma de decisiones y reforzar las buenas contribuciones

- B. Discutir el rendimiento previo con el grupo y luego examinar la necesidad de prácticas nuevas
- C. Continuar dejando solo al grupo

12. Información reciente indica que existen algunas dificultades internas entre los subordinados. El grupo tiene antecedentes notables por sus logros. Los miembros han logrado efectivamente objetivos de largo alcance. Han trabajado en armonía durante el año anterior. Todos están bien capacitados para la tarea.

- A. Intentar con los subordinados la solución propuesta por el líder y examinar la necesidad de nuevas prácticas
- B. Permitir que los miembros del grupo encuentren solos las soluciones
- C. Actuar rápida y firmemente para corregir y dirigir
- D. Participar en la discusión del problema proporcionando apoyo a los subordinados

MODELO TRIDIMENDIONAL DE EFICACIA EN EL DIRIGENTE

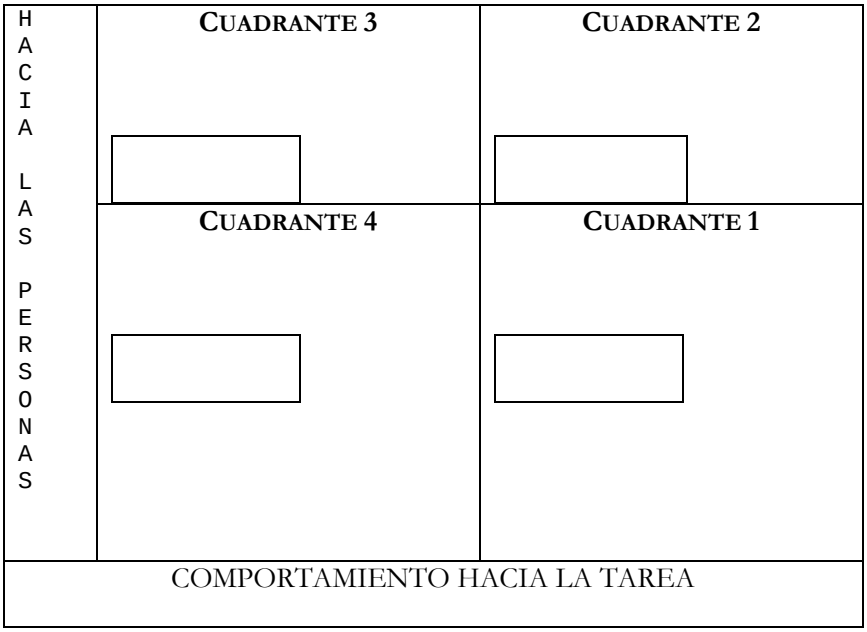


Ilustración 6-2. Modelo tridimensional de eficacia en el dirigente.

Tabla I: Determinación de estilo de dirección

Soluciones	Alternativas de acción			
	1	2	3	4
1	A	C	B	D
2	D	A	C	B
3	C	A	D	B
4	B	D	A	C
5	C	B	D	A
6	B	D	A	C
7	A	C	B	D
8	C	B	D	A
9	C	B	D	A
10	B	D	A	C
11	A	C	B	D
12	C	A	D	B

Tabla II: Determinación de las posibilidades de adaptación en cuanto al estilo de dirección

Soluciones	Alternativas de acción			
	A	B	C	D
1	+2	-1	+1	-2
2	+2	-2	+1	-1
3	+1	-1	-2	+2
4	+1	-2	+2	-1
5	-2	+1	+2	-1
6	-1	+1	-2	+2
7	-2	+2	+1	-1
8	+2	-1	-2	+1
9	-2	+1	+2	-1
10	+1	-2	-1	+2
11	-2	+2	-1	+1
12	-1	+2	-2	+1

Ilustración 6-3. Tablas del modelo tridimensional de eficacia en el dirigente.

6.10 PROCESO DE CAMBIO DE RDBMS EN UNA EMPRESA. CASO PRÁCTICO.

Una empresa ha tenido en diferentes áreas RDBMS distintos, pero se ha decidido por los directores el uso institucional del RDBMS PostgreSQL para ahorrar en el pago de licencias.

¿Qué medidas debe de adoptar el jefe del proyecto a cargo para que el personal de sistemas adopte el nuevo RDBMS?

El jefe del proyecto realizó lo siguiente:

- Escuchó las expresiones de resistencia y manifestó empatía.
    - La resistencia de los administradores de las RDBMS era que en PostgreSQL la administración se hace por medio de comandos en una Terminal en vez de forma gráfica como lo venían acostumbrando.
    - Solución realizada por el jefe del proyecto. Se cotizó varios administradores gráficos del RDBMS PostgreSQL y con el fin de reducir los costos se escogió el administrador gráfico de phpPgAdmin ajustando el cambio a las características de la organización.
- Dio a conocer al personal los hechos, necesidades, objetivos y efectos del cambio, planteando también el costo/beneficio del cambio.
    - El jefe del proyecto manifestó que la decisión por cambiar de RDBMS es para reducir los costos, disminuir el pago de consultorías a los proveedores de licencias de las RDBMS y que esto ayudaría a profesionalizar a los administradores de bases de datos otorgándoles cursos e incentivos por su adaptación.

- El jefe de proyecto buscó apoyos que fomenten la credibilidad del nuevo RDBMS.
    - Una de las cosas que hizo el jefe de proyecto fue el fomentar la entrevista con un experto en PostgreSQL dando una plática al grupo de administradores de bases de datos para que les expresara los casos de éxito que tiene este RDBMS.

- Para no imponer el cambio y hacer el cambio participativo dio oportunidad de que cotizaran y manifestaran opciones de RDBMS que se pudieran manifestar a la directiva dando opciones que no generaran costos, pero al no haber otra opción que sostuviera la competencia en Software Libre del RDBMS PostgreSQL terminaron los administradores de bases de datos en aceptarlo.

Capítulo 7

ADMINISTRACIÓN DE BASES DE DATOS

La administración de bases de datos es la responsabilidad de procurar controlar los aspectos ambientales que tienen las bases de datos. En general estos incluyen los siguientes:

- Recuperación de las bases ante una falla o perdida - Crear y probar Respaldos.
- Integridad - Verificar o ayudar a la verificación en la integridad de datos.
- Seguridad - Definir y/o implementar controles de acceso a los datos.
- Disponibilidad - Asegurarse del mayor tiempo de encendido.
- Desempeño - Asegurarse del máximo desempeño incluso con las limitaciones.
- Desarrollo, soporte y pruebas - Ayudar, controlar y orientar al programador a utilizar eficientemente la base de datos.

7.1 ADMINISTRADOR DE BASE DE DATOS EN SYBASE / SQL SERVER

Analizar las principales tareas que debe cubrir el perfil de un administrador de bases de datos, así como la manera de llevarlas a cabo dentro del RDBMS.

- Describir el ambiente operacional del SQL Server
- Describir lo que pasa cuando se instala un SQL Server
- Identificar los parámetros necesarios para la instalación
- Describir lo que pasa cuando se accede a un SQL Server
- Usar y configurar las variables de ambiente del Sistema Operativo
- Arranque y desconexión de un SQL Server

Estructura del software de Sybase.

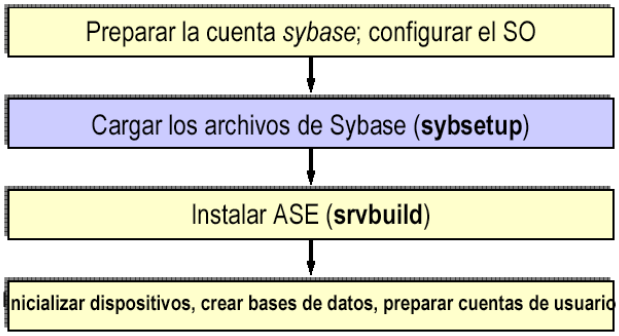


Ilustración 7-1. Secuencia de Instalación SQL Server.

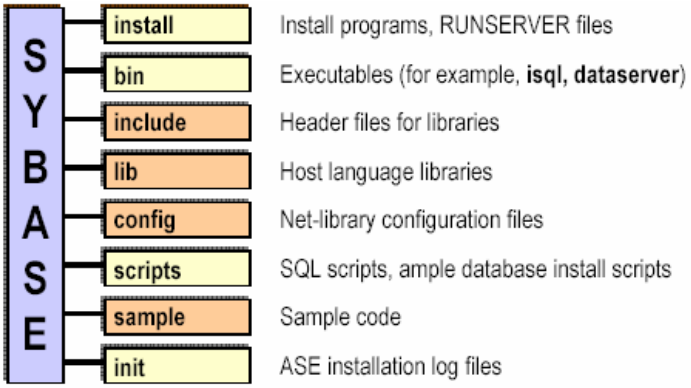


Ilustración 7-2. Estructura del Software de Sybase Parte 1.

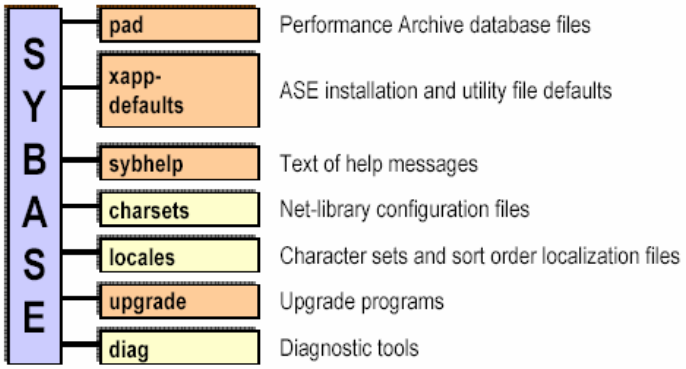


Ilustración 7-3. Estructura del Software de Sybase Parte 2.

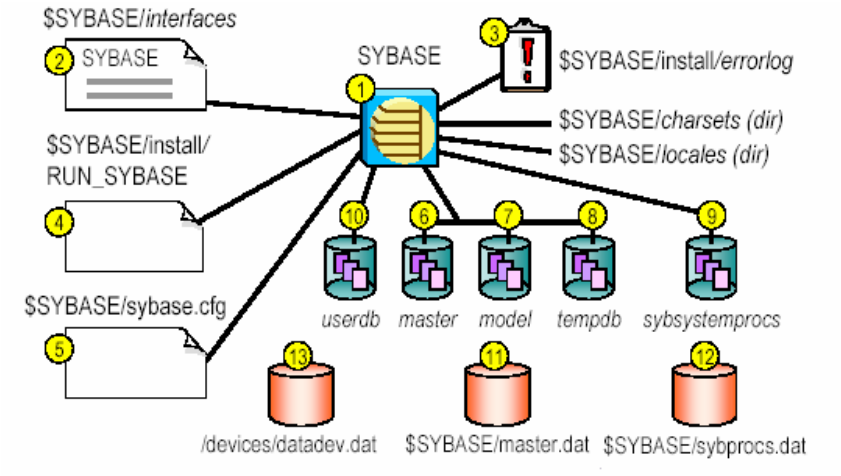


Ilustración 7-4. Previo: Entorno de Operación de SQL Server.

Archivos de SQL Server interfaces y RUNSERVER

Archivo de Interfaces

- Determinar por cual puerto de red correrá el servidor SYBASE.
- Esta localizado en el directorio **`$SYBASE/interfaces`**

```
##SYBASE on surya
##      Services
##              query   tcp     (2550)
##              master  tcp     (2550)
SYBASE
        query   tcp     ether   surya   2550
        master  tcp     ether   surya   2550
```

Código 7-1. Fragmento de código que se encuentra en el la ruta $SYBASE/interfaces.

Formato del Archivo Interfaces

- El archivo interfaces debe estar formateado correctamente para evitar errores de conectividad.
- Usando **`dsedit`** se puede darle formato al archivo interfaces.
- Un usuario experto edita el archivo interfaces manualmente utilizando un editor de texto como **`vi`** o **`emacs`**

```
# comments
<newline>
SERVERNAME<tab>retry_attempts<tab>delay_interval
<newline>
<tab>service_type protocol network machine port
<newline>
<blank lines or comments between entries for different SQL Servers>
```

Código 7-2. Formato del archive interfaces.

¿Para que es Usado el Archivo Interfaces?

Proporciona detalles de conectividad para todos los servidores.

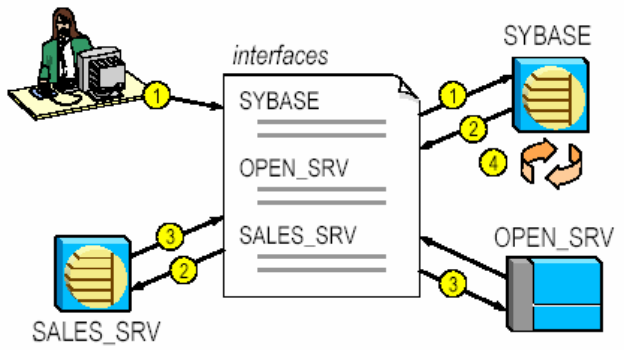


Ilustración 7-5. Secuencia de conexiones con servidores en Sybase.

Mantenimiento del Archivo Interfaces.

Una copia del Archivo interfaces debe existir en cada nodo

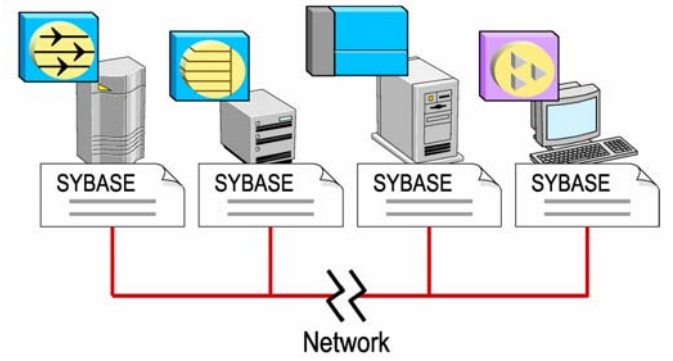


Ilustración 7-6. Disposición del archivo interfaces.

Variables de ambiente

Las variables de ambiente son requeridas para (1) las conexiones cliente y (2) para arrancar el servidor.

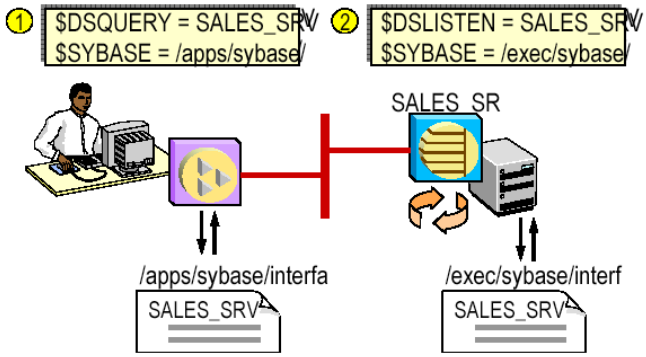


Ilustración 7-7. Esquema del uso de las variables de ambiente.

Error Log de SQL Server

- Todos los errores son registrados en el Archivo **errorlog**.
- La localización y el nombre del archivo **errorlog** son especificados durante la instalación.

```
00:00000:00001:1998/04/07 14:44:30.90 server on top of default carácter set:
00:00000:00001:1998/04/07 14:44:30.90 server 'ISO_1' (ID = 1).
00:00000:00001:1998/04/07 14:49:09.75 server DBCC TRACEON 8399, SPID 1
```

Código 7-3. Ejemplo de la bitácora de errores (error log).

El archivo RUNSERVER

- Es usado para arrancar un SQL Server
- El Formato es **RUN_SERVERNAME**

```
#!/bin/sh
#
# SQL Server Information:
# name:         SYBASE
# master device:
/work/sybase/ASE/devices/master.dat
# master device size:  10752
# errorlog:     /work/sybase/ASE/install/errorlog
# interfaces:   /work/ASE
#
/work/sybase/ASE/bin/dataserver
-c/work/sybase/ASE/servername.cfg
-d/work/sybase/ASE/devices/master.dat \
-sSYBASE
-e/work/sybase/ASE/install/errorlog \
-i/work/sybase/ASE
```

Código 7-4. Fragmento del archivo RUN_SERVERNAME.

Archivo de configuración del SQL Server

Se usa el comando **sp_configure** para configurar manualmente un SQL Server o editando el archivo de configuración.

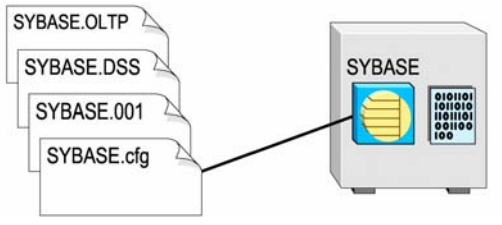


Ilustración 7-8. Estructura de los archivos de configuración de Sybase.

```
############################################################
# Configuration File for the Sybase SQL Server
# Please read the System Administration Guide (SAG)
# before changing any of the values in this file.
############################################################

[Configuration Options]

[General Information]

[Backup/Recovery]
recovery interval in minutes = DEFAULT
print recovery information = DEFAULT
tape retention in days = DEFAULT

[Cache Manager]
number of oam trips = DEFAULT
number of index trips = DEFAULT
procedure cache percent = DEFAULT
memory alignment boundary = DEFAULT

[Named Cache:default data cache]
cache size = DEFAULT
cache status = default data cache

[Disk I/O]
allow sql server async i/o = DEFAULT
disk i/o structures = DEFAULT
page utilization percent = DEFAULT
number of devices = DEFAULT

[Network Communication]
default network packet size = DEFAULT
max network packet size = DEFAULT
remote server pre-read packets = DEFAULT
number of remote connections = DEFAULT
allow remote access = DEFAULT
number of remote logins = DEFAULT
number of remote sites = DEFAULT
max number network listeners = DEFAULT
tcp no delay = DEFAULT

[O/S Resources]
max async i/os per engine = DEFAULT
max async i/os per server = DEFAULT
```

Código 7-5. Archivo de Configuración de Sybase SQL Server

Instalación de un servidor de bases de datos de SQL Server

Parámetros a configurar.

Record Server Information for Interfaces File	Server name (in 7-bit ascii characters):_____ Host Name:_____ Query port number:_____ Alias:_____
Choose Master Device Size	Size for master device in megabytes (>=30Mb)_____
Choose Master Device Location	***Raw Disk Partition Installation*** Size of the raw partition in megabytes or pages: _____ Name of partition:_____ ☐ Partition deleted from */etc./tstab*. ☐ Owner and permissions changed.

Ilustración 7-9. Opciones de instalación se SQL Server.

Parámetro	Usado en un ejemplo de srvbuild
Server name	SYBASE
Master device location	/work/ASE/devices/master.dat
Master device size	60 MB (and the master db is on this device)
Device name for sybsystemprocs database	/work/ASE/devices/sybprocs.dat
Error log location	/work/ASE/install/errorlog
Backup Server name	SYB_BACKUP
Language	us_english
Character set	ISO 8859-1
Sort order	Binary
Activate auditing (y/n)	No

Tabla 7-10. Parámetros críticos para la instalación del SQL Server.

Requisitos en la Preinstalación

Los dispositivos del SQL Server son porciones de un disco o archivos que son utilizados para almacenar bases de datos o porciones de bases de datos. Se puede inicializar dispositivos de base de datos utilizando archivos regulares del sistema operativo, o Raw Devices.

- ¿Qué tipo de dispositivo es mejor?
    - Sistema Operativo: UNIX
- ¿Qué tipos de dispositivos usar?
    - Particiones crudas preferentemente, los archivos de disco son aceptables para el dispositivo de master (si son espejeados).
    - Los archivos de UNIX no son recomendados para bases de datos de producción porque se escriben los buffers I/O de UNIX.
    - Durante el desarrollo, es preferible escoger un dispositivo que sea menos seguro pero más fácil de administrar, archivos de disco en UNIX.

Instalación de SQL Server usando srvbuild.

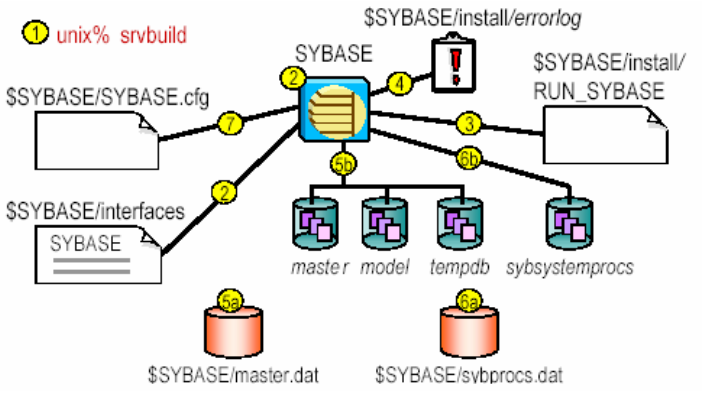


Ilustración 7-11. Instalación de un servidor en SQL Server usando srvbuild.

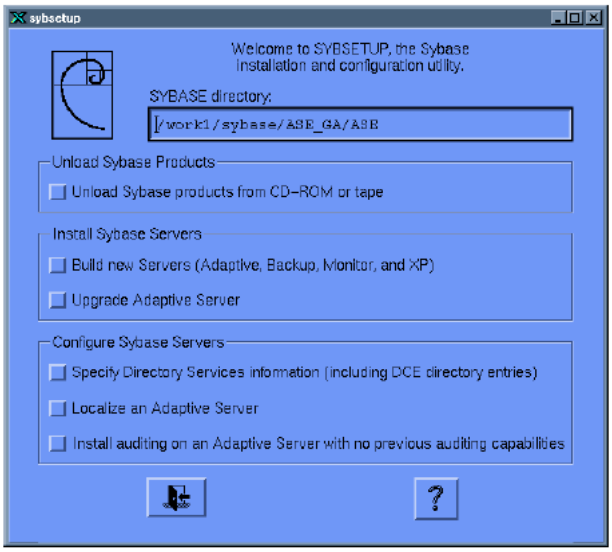


Ilustración 7-12. Instalar un servidor en SQL Server usando srvbuild, pantalla inicial.

Ilustración 7-13. Instalar un servidor en SQL Server usando srvbuild, pantalla Select Servers to Build.

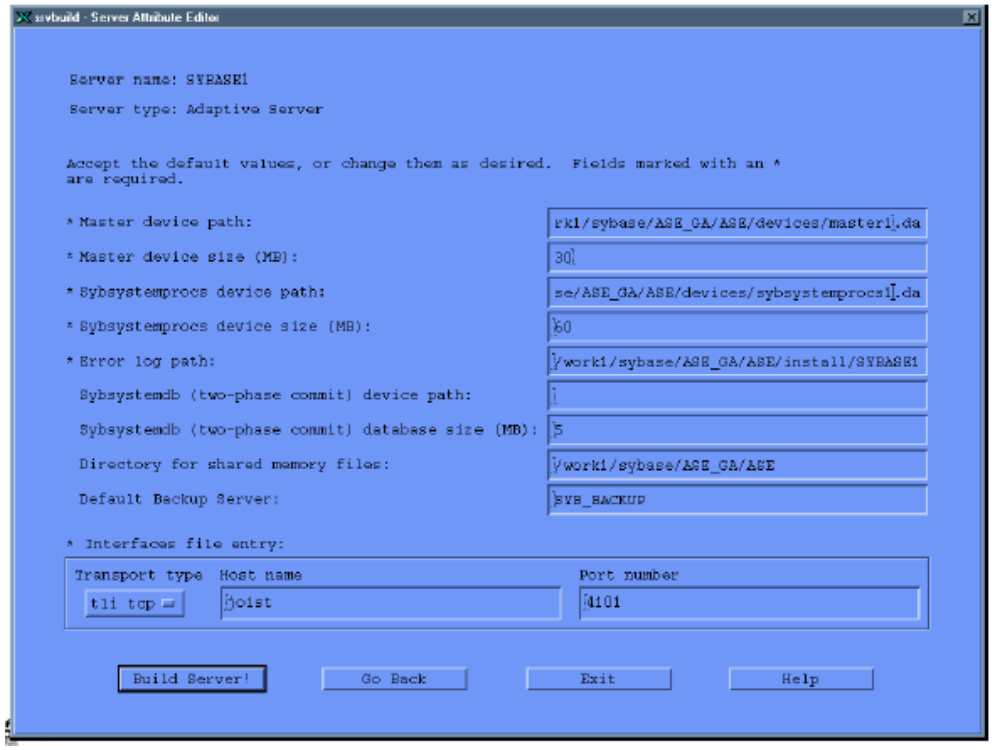


Ilustración 7-14. Instalar un servidor en SQL Server usando srvbuild, pantalla Server Attribute Editor.

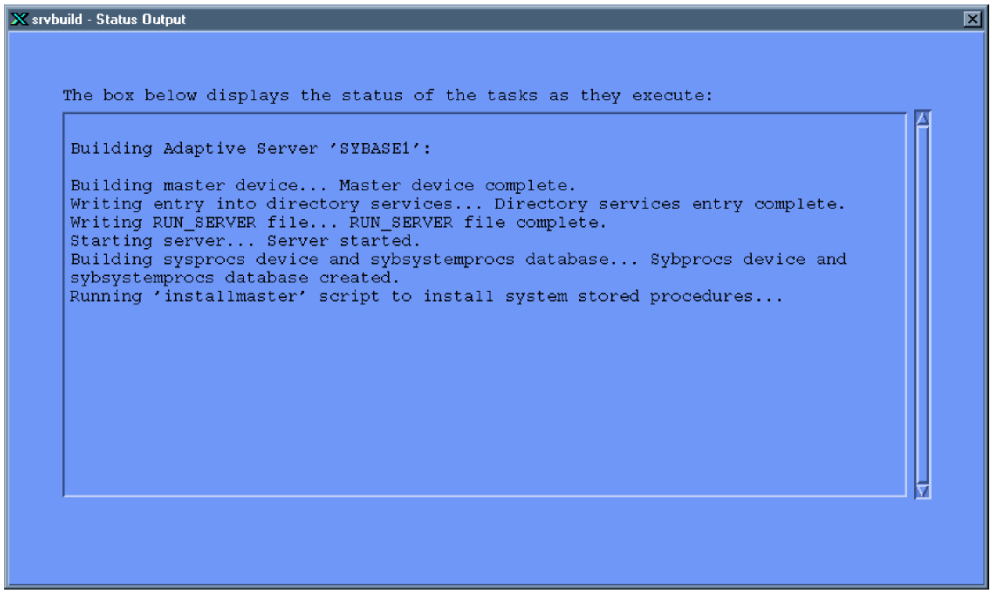


Ilustración 7-15. Instalar un servidor en SQL Server usando srvbuild, pantalla Status Output.

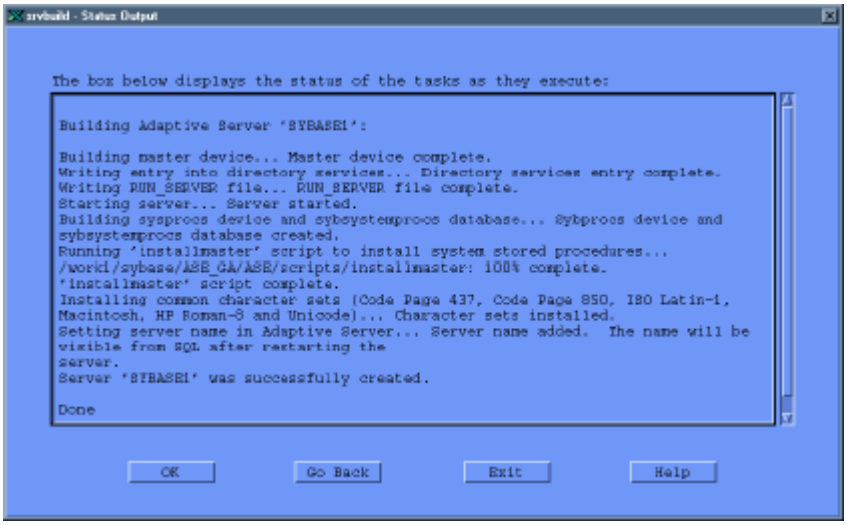


Ilustración 7-16. Instalar un servidor en SQL Server usando srvbuild, pantalla Status Output (continuación).

Crear servidores de SQL Server desde archivos fuente.

- Se puede crear un SQL Server o un **`Backup Server`** con valores especificados en un archivo fuente que define los atributos de el servidor
- Para crear un servidor desde un archivo fuente, se usa el comando **`srvbuildres`**

```
surya% srvbuildres –r [archivo_fuente]
```

Código 7-6. Crear un servidor de SQL Server desde archivos fuente.

```
srvbuild11 12.001-SYBASE.rs

srvbuild. release_directory: /work1/sybase/ASE_GA/ASE
srvbuild.product: sqlsrv
srvbuild.server_name: SYBASE
srvbuild.new_config: yes
srvbuild.do_add_server: yes
srvbuild.do_upgrade: no
srvbuild.network_protocol_list: tcp
srvbuild.network_hostname_list: joist
srvbuild.network_port_list: 4100
srvbu ild. master_device_physical_name: /work1/sybase/ASE_GA/ASE/
devices/master.dat
srvbuild.master_device_size: 30
srvbuild .errorlog: /work1/sybase/ASE_GA/ASE/install/SYBASE.log
srvbuild .sybsystemprocs_device_physical_name: /work1/sybase/
ASE_GA/ASE/devices/sybsystemp
rocs.dat
srvbuild .sybsystemprocs_device_size: 60
srvbuild.default_backup_server: SYB_BACKUP
.....................................

srvbuild11 12.002-SYBASE_back.rs

srvbuild. release_directory: /work1/sybase/ASE_GA/ASE
srvbuild.product: bsrv
srvbuild.server_name: SYBASE_back
srvbuild.do_add_backup_server: yes
srvbuild.network_protocol_list: tcp
srvbuild.network_hostname_list: joist
srvbuild.network_port_list: 4202
srvbuild.language: USE_DEFAULT
```

```
srvbuild.character_set: USE_DEFAULT
srvbuild.tape_config_file: USE_DEFAULT
srvbuild.errorlog: /work1/sybase/ASE_GA/ASE/install/ SYBASE_back.log
```

Código 7-7. Abstracto Archivo fuente para instalar un servidor de SQL Server.

Instalando un Backup Server

`srvbuild` también puede ser usado para instalar un **`Backup Server`**. Un **`Backup Server`** realiza todos los backups de un SQL Server

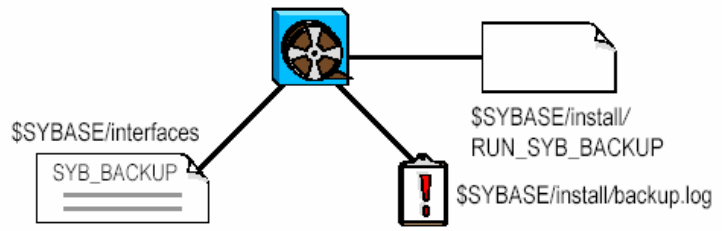


Ilustración 7-17. Instalación de un Backup Server en SQL Server usando srvbuild.

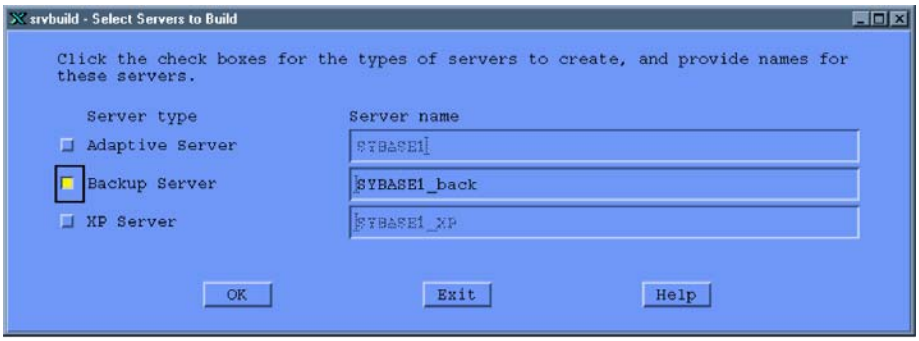


Ilustración 7-18. Instalación de un Backup Server en SQL Server usando srvbuild, pantalla Select Servers to Build.

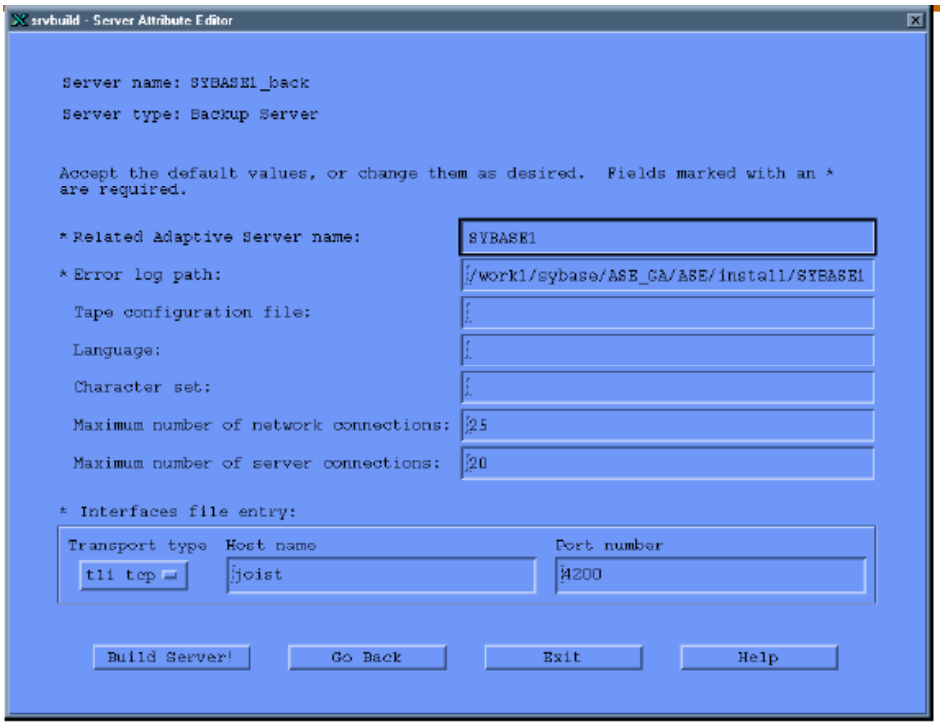


Ilustración 7-19. Instalación de un Backup Server en SQL Server usando srvbuild, pantalla Server Attribute Editor.

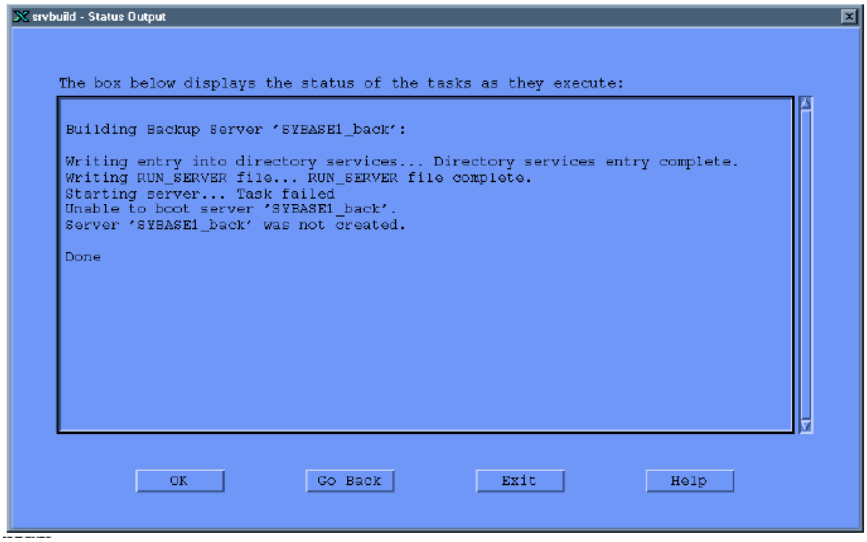


Ilustración 7-20. Instalación de un Backup Server en SQL Server usando srvbuild, pantalla Status Output.

Desconexión y arranque de un SQL Server

Usar el comando **`shutdown`** para desconectar un servidor.

```
1> shutdown
2> go
Server SHUTDOWN by request.
The SQL Server is terminating this process.
DB-LIBRARY error:
Unexpected EOF from SQL Server.
%
```

Código 7-8. Desconexión de un servidor.

Usar el comando **`startserver`** para arrancar un servidor

```
surya% cd $SYBASE/install
surya% startserver -f RUN_SYBASE
[startup messages deleted]
```

Código 7-9. Arrancar un servidor de SQL Server.

Verificar que el servidor esté arriba y ejecutándose

```
surya% showserver
USER PID %CPU %MEM SZ RSS TT STAT START
TIME COMMAND
sybase 12155 0.0 6.1 900 1672 pbS 13:52 2:48
/work/sybase/ASE/bin/dataserver -d/work/
sybase/ASE/devices/master.dat -sSYBASE -e/work/
sybase/ASE/install/errorlog -i/work/sybase/ASE
surya%
```

Código 7-10. Verificar si el servidor se está ejecutándose.

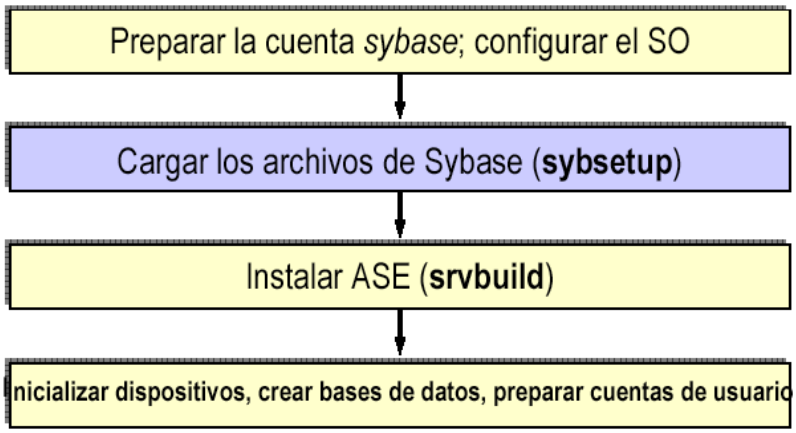


Ilustración 7-21. Secuencia de instalación de SQL Server.

Notas importantes:

- Escoger cuidadosamente una ubicación para el dispositivo **`master`**
- Estar seguro que la cuenta sybase sea dueño del dispositivo **`master`**
- Escoger un número de puerto que no este en uso
- Arrancar un SQL Server como sybase, y no como un súper usuario**`/root`**
- En el archivo interfaces, las líneas **`query`** y **`master`** deben comenzar con un tabulador

Recursos de asignación y dispositivos.

Entre las tareas de un administrador de Sybase están las siguientes:

- Describir como son usados los recursos de disco y listarlas consideraciones para el almacenamiento
- Definir dispositivos de base de datos
- Explicar como inicializar y eliminar dispositivos de base de datos
- Describir dispositivos predeterminados y como son usados
- Desplegar información acerca de los dispositivos inicializados
- Preparar duplicado de disco para dispositivos de base de datos
- Describir información contenidas en tablas de base de datos o del sistema con respecto a los dispositivos, especialmente información en sysdevices

Recursos de disco

- El administrador del sistema tiene control sobre:
    - La asignación de recursos de disco para el SQL Server
    - El lugar físico de las bases de datos, tablas e índices sobre esos recursos
- Cuando se configura un nuevo sistema, el SA (System Administrator) debe:
    - Inicializar y asignar un conjunto predeterminado de dispositivos de base de datos
    - Determinar y duplicar dispositivos de base de datos para su recuperación
- Después de que los recursos de disco han sido asignados al SQL Server, el SA, dbo, y dueños de objetos de base de datos, deben:

  Considerar como colocar base de datos y objetos de base de datos sobre dispositivos de base de datos

Dispositivos de base de datos

- Un dispositivo de base de datos almacena objetos de base de datos que se crean en esa base de datos
- El término dispositivo no se refiere necesariamente a un dispositivo físico diferente
    - El dispositivo puede ser alguna porción de un disco, como una partición de disco, o un archivo en el sistema de archivos
- Cada dispositivo de base de datos o archivo debe ser preparado y hacérselo saber al SQL Server antes que pueda ser usado para el almacenamiento de base de datos
    - Este proceso es conocido como inicialización
- Después de que el dispositivo de base de datos ha sido inicializado, este puede ser:
    - Asignado al banco predeterminado de dispositivos para los comandos **`create`** y **`alter database`**
    - Asignado al banco de espacio disponible para un usuario de base de datos

- Asignado a un usuario de base de datos y usado para almacenar uno o más objetos de base de datos
- Asignado para almacenar logs de transacciones de la base de datos

Consideraciones para el manejo de almacenamiento

- La recuperación es clave para motivarse a usar varios dispositivos de disco
- La recuperación ininterrumpida puede ser llevada a cabo por dispositivos de duplicado
- La recuperación total puede asegurarse guardando el log de las bases de datos en un dispositivo físico diferente
- El rendimiento del Sistema puede ser mejorado colocando los objetos de base de datos y logs sobre dispositivos separados:

  Colocando una tabla en un disco duro y los índices nonclustered sobre otro se asegura que las lecturas y escrituras físicas sean más rápidas

  Dividiendo las tablas grandes a través de dos discos puede mejorar el rendimiento, especialmente en las aplicaciones multiusuario

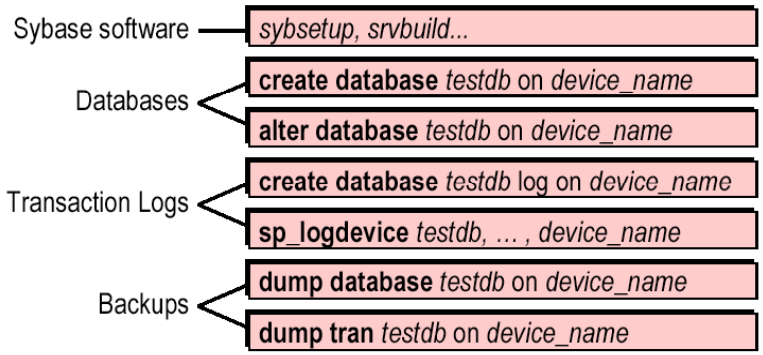


Ilustración 7-22. Dispositivos y aplicaciones que requieren recursos de espacio de almacenamiento en Sybase.

- Los administradores del sistema deciden donde serán almacenados los recursos y cuanto espacio será asignado a ellos.

Colocando el Software de Sybase en discos

- Descarga del Software Sybase

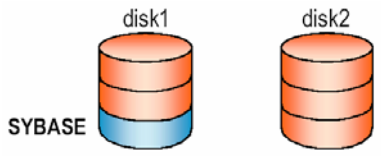


Ilustración 7-23. Ubicación del software de Sybase en una descarga usual.

- La descarga del software Sybase copia ejecutables y otros archivos importantes en el directorio **SYBASE** en cualquier parte de un disco físico

- En este caso, el software Sybase esta en una partición del disk1
- Se instala SQL Server

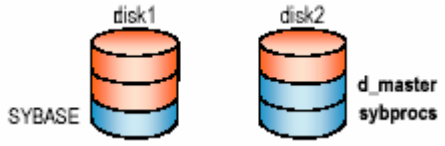


Ilustración 7-24. Disposición de dispositivos en discos al instalar Sybase.

- Cuando instala SQL Server, se afectan ciertos archivos del árbol de directorio SYBASE
- En adición, se crea un dispositivo **master** que contenga tablas de **master**, **sybsystemprocs**, **model**, **sybsecurity**[1] y **tempdb**
- En este caso, el dispositivo **master** esta en una partición del disco 2

Tablas del Sistema que manejan el almacenamiento

- Dos tablas en la base de datos **master** :
  - **sysusages**
  - **sysdevices**
- Dos tablas en la base de datos de usuarios:
  - **syssegments**
  - **sysindexes**
- Las tablas saben el lugar de las bases de datos, tablas e índices (incluyendo **syslogs**)
- La relación entre las tablas se da entre segmap en **sysusuages** y **segment** en **syssegments**

Inicializando dispositivos

Ilustración 7-25. Disposición en discos al inicializar dispositivos.

- Se debe inicializar los dispositivos y después crear bases de datos y segmentos sobre los dispositivos.
- El SA inicializa un nuevo dispositivo de base de datos con el comando **disk init**

[1] El dispositivo de sybsecurity almacena la base de datos de sybsecurity. La base de datos de sybsecurity es creada como parte del proceso de configuración para auditoria. Contiene todas las tablas de sistema de la base de datos modelo así como la tabla de sistema para seguir el rastro del rango de opciones del servidor a auditar y el sistema de tablas para el rastro de la auditoria.

- **`disk init`** mapea un disco físico (o un archivo del sistema operativo) a un nombre de dispositivo de base de datos

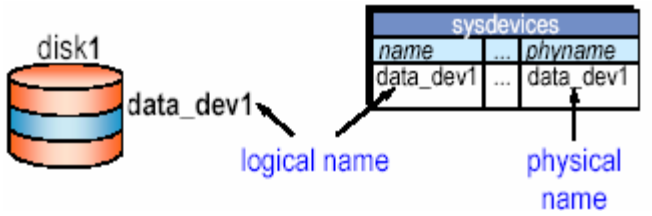


Ilustración 7-26. Manejo de sysdevices para mapear el nombre lógico del dispositivo y el nombre físico.

  - El nuevo dispositivo es listado en **`master..sysdevices`**
- Después de inicializado el dispositivo, se pueden crear base de datos en él

Usando Disk Init

```
disk init
name = "device_name",
physname = "physical_name",
vdevno = virtual_device_number,
size = number_of_pages
```

Código. Sintaxis para disk init.

```
disk init
name = "data_dev1",
physname = "/dev/c0t1d2s3",
vdevno = 2,
size = 92160
```

Código 7-11. Ejemplo de uso del comando disk INIT en UNIX en Sybase.

Consideraciones

- Solo los administradores del sistema pueden usar **`disk init`**
- Antes de ejecutar **`disk init`**:
  - Respalde la base de datos **`master`**
  - Asegúrese de que hay suficiente espacio en disco
  - Asegúrese de que el archivo o dispositivo no haya sido ya inicializado
  - Asegúrese que la cuenta sybase tenga permisos de escritura sobre ese dispositivo

  Asegúrese de que el SQL Server sea configurado apropiadamente considerando dispositivos, memoria y conexiones
- Máximo de dispositivos de base de datos configurables: 255
- Después de ejecutar **`disk init`**, respalde la base de datos

Dispositivos Predeterminados

- Con **`create database`**, se especifica el dispositivo sobre el cuál la base de datos deberá ser creada

- Si no hay un dispositivo especificado, la base de datos se crea sobre el dispositivo predeterminado
- Después de que un dispositivo es inicializado, el SA puede preparar el dispositivo como un dispositivo predeterminado usando **sp_diskdefault**

```
sp_diskdefault logical_name,
       {defaulton | defaultoff}
```

Código 7-12. Sintaxis del comando sp_diskdefault en Sybase.

- Ejemplo: Inicialice **data_dev1** y entonces se agrega como un dispositivo predeterminado:

```
disk init name = "data_dev1",
physname = "/dev/c0t1d2s3", etc.
exec sp_diskdefault data_dev1, defaulton
```

Código 7-13. Uso del comando disk init para agregar un dispositivo predeterminado en Sybase

- **sp_diskdefault** marca esos dispositivos como dispositivos predeterminados en **sysdevices**
- Cuando un SQL Server es instalado, el dispositivo **master** es un dispositivo predeterminado
    - Recomendado: cambiar esto para **master** y así evitar desorden en la base

```
exec sp_diskdefault master, defaultoff
exec sp_diskdefault data_dev1, defaulton
```

Código 7-14. Desactiva el dispositivo master como el dispositivo por defecto y activa al dispositivo data_dev1 como el dispositivo predeterminado.

- Con comandos **sp_diskdefault** por separado, se pueden especificar más de un dispositivo predeterminado
- Varios dispositivos predeterminados se van usando conforme un orden alfabético como se vayan creando

Eliminando Dispositivos

- ¿Cuando es necesario eliminar un dispositivo?
    - Al cambiar, reparar, o agregar hardware
    - Al cambiar el tamaño del dispositivo, se debe quitar y luego volver a inicializar
- Para quitar un dispositivo:

```
sp_dropdevice device_name
```

Código 7-15. Quitar un dispositivo en Sybase.

  Entonces resetear el SQL Server para que pueda rehusarse el número de dispositivo **vdevno** (Virtual Device Number)

- Para archivos de disco, borrar el archivo después de eliminar el dispositivo para liberar el espacio en disco
- No se puede borrar un dispositivo si aún contiene base de datos

Obteniendo información del dispositivo

- Cuando se ejecuta **disk init**, un nuevo renglón es agregado a la tabla **sysdevices** en **master**
- **sysdevices** lista base de datos y dispositivos de respaldo

low	high	status	cntrltype	name	physname	mirrorname
16777216	16782335	2	0	data_dev1	/dev/c0t1d2s3	NULL
0	10239	3	0	master	d_master	NULL
0	20000	16	3	tapedump1	/dev/rmt4	NULL
0	20000	16	2	tapedump2	/dev/rst0	NULL

*Ilustración 7-27. Salida de select * from sysdevices en Sybase.*

- El procedimiento **`sp_helpdevice`** consulta **`sysdevices`**, evalúa ciertos valores, e imprime información útil acerca de los dispositivos, como se muestra a continuación:

```
device_name          physical_name
     description
     status cntrltype  device_number          low    high
-------------------------------------------------------------------------------
data_dev1            /dev/rsd2d
     special, physical disk, 10.00 MB
          2       0       1    16777216  16782335
master               d_master
     special, default disk, physical disk, 20 MB
          3       0       0           0      10239
tapedump1                 /dev/rmt4
     tape, 625 MB, dump device
     16         3              0       0      20000
etc.
```

Ilustración 7-28. Información obtenida con sp_helpdevice en Sybase.

Tips para la planeación de dispositivos

- Use nombres significativos.
- Asegúrese de que las particiones especificadas realmente estén disponibles; siga las restricciones de la plataforma especifica sobre que partes del disco pueden ser usadas.
- Cree y guarde scripts de comandos de asignación de los dispositivos.

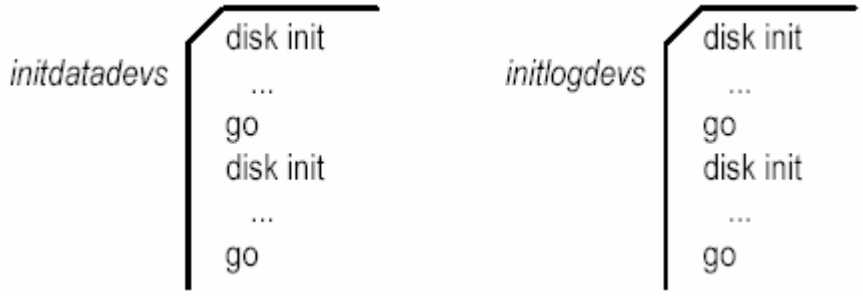


Ilustración 7-29. Recomendación para el manejo de dispositivos, crear scripts para dispositivos de datos y de logs, entre otros.

- Mantenga una copia de la tabla **`sysdevices`**

- Mantener una copia del trazado físico y lógico

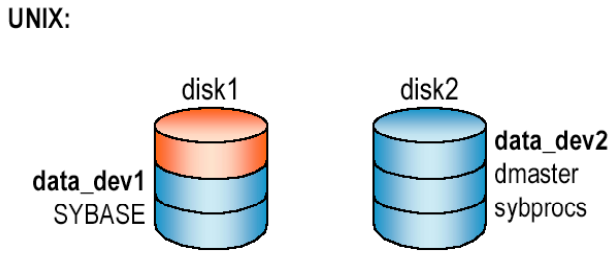


Ilustración 7-30. Estructura física y lógica de dispositivos en Sybase.

Duplicado

- El espejo de un SQL Server duplica todo el contenido de un dispositivo
    - Escribe en ambos dispositivos
    - Las lecturas vienen desde el lado primario

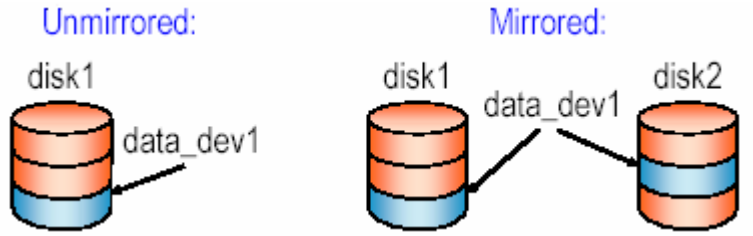


Ilustración 7-31. Representación del espejeo de dispositivos.

- Si un disco de falla, SQL Server lo nota al tratar de leer o escribir en ese disco y por lo tanto, continua con el otro

¿Porqué duplicar?

- Beneficios:
    - Previene la interrupción de tareas y/o procesos debido alguna falla en el disco
    - Asegura la recuperación total e ininterrumpida
- Costos:
    - Usa recursos adicionales (almacenamiento en disco)
    - Las escrituras son más lentas porque se están duplicando
- No obstante los costos, la duplicación de discos es recomendada ampliamente

¿Qué se debe duplicar?

- Se debe duplicar todos los dispositivos de base de datos predeterminados así estará protegido de comandos create o alter database afecta un dispositivo de base de datos en la lista predeterminada

- Duplicar los dispositivos más valiosos y vulnerables
    - **`Master device`**
    - **`Log devices`**
    - **`Active devices`**
- Es conveniente poner el duplicado del log de transacciones sobre un dispositivo separado

¿Qué sucede cuando se duplica un dispositivo?

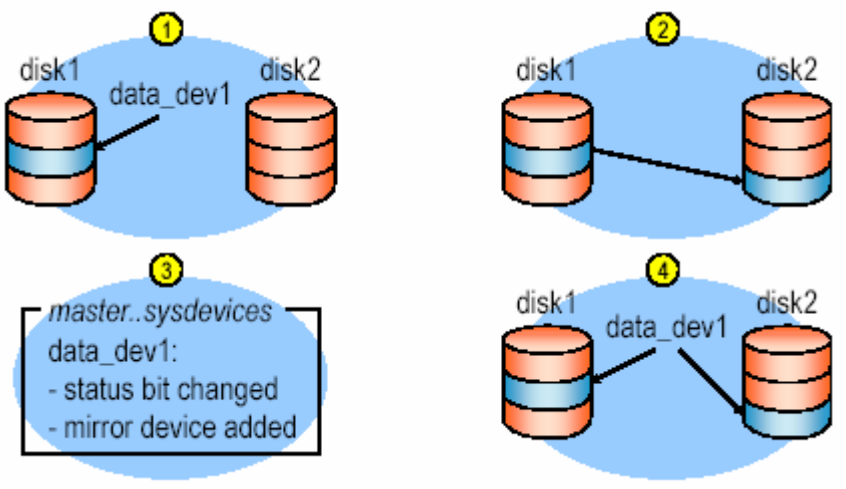


Ilustración 7-32. Secuencia de acciones al duplicar dispositivos.

Comandos de duplicación de discos:

- Tres comandos controlan el duplicado de discos.
- **`disk mirror`** comienza el duplicado de discos.
    - No inicialice el dispositivo duplicado con **`disk init`**; un dispositivo de base de datos y su duplicado constituyen un solo dispositivo lógico.
    - El comando disk mirror agrega el nombre del duplicado en la columna mirror name de la tabla sysdevices.
- **`disk unmirror`** detiene el proceso de duplicado cuando es necesario el mantenimiento del hardware o el hardware del dispositivo necesita ser cambiado.
- **`disk remirror`** restaura el proceso de duplicado que había sido suspendido debido a una falla del dispositivo o por el uso de disk unmirror.

Usando el comando **`disk mirror`**

```
disk mirror
name = "device_name",
mirror = "physical_name"
[ ,writes = {serial | noserial} ]
```

Código 7-16. Sintaxis de disk mirror

```
disk mirror
name = "data_dev1",
mirror = "/dev/c0t1d2s4"
```
Código 7-17. Uso del comando usando UNIX de disk mirror en Sybase.

Desactivando el duplicado

- Intencional o Automático
    - Intencional: Ejecutando **`disk unmirror`**
    - Automático: Si un error I/O es detectado sobre algún disco
- ¿Qué sucede si hay un error de I/O?
    - Los bits de estados están en sysdevices y ahí se indica la desactivación y cuál disco sigue operando
    - La I/O es deshabilitada en el dispositivo defectuoso
    - Se realiza un **`checkpoint`** sobre la base de datos **`master`**
    - Un mensaje de error es enviado al **`error log`**
    - Cualquier proceso **`waitfor mirrorexit`** es habilitado

Resumen de los comandos de duplicación

- Duplicar un dispositivo:

```
disk mirror name = "device_name", mirror = "physical_name"
```
Código 7-18. Uso del comando disk mirror para duplicar un dispositivo en Sybase.

- Suspender el duplicado de un dispositivo:

```
disk unmirror name ="device_name"
```
Código 7-19. Uso del comando disk unmirror para suspender el duplicado un dispositivo en Sybase.

- Continuar el duplicado de un dispositivo:

```
disk remirror name ="device_name"
```
Código 7-20. Uso del comando disk remirror para continuar con el duplicado de un dispositivo en Sybase.

- Deshabilitar el duplicado de un dispositivo permanentemente:

```
disk unmirror name ="device_name", mode = remove
```
Código 7-21. Uso del comando disk unmirror para deshabilitar el duplicado de un dispositivo permanentemente en Sybase.

- Deshabilitar el dispositivo primario temporalmente:

```
disk unmirror name ="device_name", side = "primary", mode = retain
```
Código 7-22. Uso del comando disk unmirror para deshabilitar el dispositivo primario temporalmente en Sybase.

- Para reinstalar el dispositivo primario, use **`disk remirror`**

Desplegado de dispositivos duplicados

El procedimiento **`sp_helpdevice`** despliega información acerca de las duplicaciones

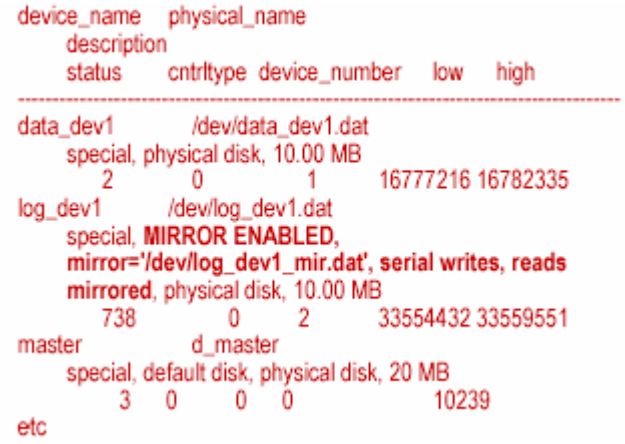


Ilustración 7-33. Ejemplo de salida de sp_helpdevice.

Puntos importantes acerca del duplicado

- Cuando se duplica un dispositivo, solo hay un solo dispositivo lógico.
- El dispositivo físico secundario debe ser al menos tan grande como el dispositivo primario, y deben estar en discos físicos separados.
- Asegurarse de seguir las restricciones de su plataforma para el duplicado de dispositivos diferentes.
- Para duplicar una base de datos completamente, debe encontrarse en ambos dispositivos físicos.
- Duplicar discos cuando la actividad es baja, y respaldar **`master`** cuando haya terminado.
- Use **`sp_helpdevice`** para desplegar información acerca de los duplicados.

Proporcionando el nombre del duplicado del dispositivo **`master`** en el arranque

- Si el dispositivo **`master`** es duplicado, edite el archivo **`RUNSERVER`** para agregar el nombre físico del duplicado.
    - Si hay un problema con el dispositivo primario **`master`**, SQL puede acceder con el duplicado.

```
dataserver -ddevicename ...
[-rmastermirror_devicename]...
```

Código 7-23. Sintaxis para configurar el acceso al dispositivo master duplicado en Sybase.

Resumiendo sección de dispositivos:

- Examine las necesidades de recursos de discos y haga un plan de dispositivos.
- Preguntarse: ¿Qué recursos de dispositivos son requeridos?
    - Software Sybase.
    - Databases (**`data`** and **`log`**), **`backups`**.

- Para agregar dispositivos de base de datos, ejecutar **`disk init`**.
- Para borrar dispositivos de base de datos, ejecute **`sp_dropdevice`**.
- Para duplicar dispositivos de base de datos, ejecutar **`disk mirror`**.
- Para desplegar información dispositivos de base de datos, ejecutar **`sp_helpdevice`**, el cual lee de **`sysdevices`**.

Creación de bases de datos.

Crear y modificar base de datos, entender las opciones de base de datos, monitorear el espacio de las bases de datos, manejar la asignación de espacio del log de transacciones, entender el mapeo de bases de datos a tablas del sistema **`sysdatabases`** y **`sysdevices`**.

Creando y Dándoles Tamaño a las Bases de Datos

¿Qué sucede cuando se crea una base de datos?

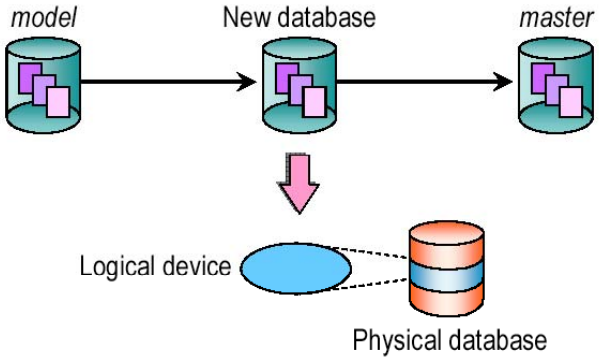


Ilustración 7-34. Operaciones que se realizan al crear una base de datos en Sybase.

create database

```
create database database_name
[on database_device [= size]
       [,database_device = size]]...]
[log on database_device = [size]]...]
```

Código 7-24. Sintaxis para crear bases de datos en Sybase.

`size` es especificado en megabytes

Ejemplos:

```
create database pubs2
```

Código 7-25. Creación de una base de datos con nombre pubs2.

```
create database salesdb on data_dev1=5
```

Código 7-26. Creación de una base de datos con nombre salesdb en el dispositivo data_dev1 de tamaño de 5 Mb.

```
create database salesdb on data_dev1=5 log on log_dev1 = 2
```

Código 7-27. Creación de una base de datos con nombre salesdb en le dispositivo data_dev1 de 5 Mb y el log ubicarlo en log_dev1 de 2 Mb.

Prerrequisitos al crear base de datos:

Antes de crear base de datos, hay que decidir:

- El tamaño de la base de datos.
- La ubicación y el espacio necesario.
- Si un dispositivo de log es necesario y, si lo es, de que tamaño.

Se debe estar en la base de datos **`master`** para ejecutar el comando create database

Database size	5 MB
Database name	pubs2
Database location	data_dev
Location of log	log_dev

Tabla 7-1. Opciones a configurar al crear una nueva base de datos.

Tamaño de una base de datos:

- El tamaño es en megabytes, mínimo 2MB
- Fácil de extender, no debe de estar con poco espacio
    - Para hacer más chica la base de datos, se debe volverla a crear y copiar datos de nuevo con la utilidad de copiado **`bcp`**
- Al estimar el tamaño de una base de datos, considerar principalmente:
    - Tablas
    - Índices
    - Log de transacciones
- Dejar algún espacio libre, dependiendo de la actividad anticipada
- Usar **`sp_estspace`** para estimar el tamaño de las tablas y sus índices

name	type	idx_level	Pages	Kbytes
titles	data	0	983	1966
titleidind	clustered	0	7	14
titleidind	clustered	1	1	2
titleind	nonclustered	0	279	558
titleind	nonclustered	1	9	18
titleind	nonclustered	2	1	2

Total_Mbytes: 2.50

(followed by summary data for titleidind, titleind)

Ilustración 7-35. Uso de sp_estpace indicando sp_estspace, 1000 en Sybase

Tamaño del log:

- El tamaño del Log depende de la actividad (tipo y cantidad de transacciones) y la frecuencia de los respaldos
    - Un buen punto de partida: 10-25% del tamaño global de la base de datos
    - Todos los inserts, deletes, y updates son registrados
    - Para **`select into`** y **`fast bulk copy`**: solo la asignación y desasignación de espacio son registrados
- Simule correr aplicaciones con el mismo número de usuarios; entonces mida el uso del log usando **`dbcc checktable(syslogs)`**
- El log es fácil de extender, imposible contraer

Dejando los respaldos del log de transacciones en un dispositivo separado:

- Ayuda en el monitoreo del uso de espacio y manejo del espacio.
- Permite el duplicado de disco del log para una recuperación a tiempo.

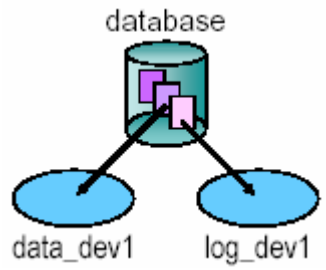


Ilustración 7-36. Esquema de una base de datos donde los datos y el log se encuentran en dispositivos separados.

```
create database database_name
[on{default | database_device}[ = size]
       [,database_device [= size]] …]
[log on database_device [=size,…]
       [,database_device [= size]]…]
[with override]
[for load]
```

Código. Sintaxis para crear bases de datos en Sybase.

```
create database pubs2 on default =4
```

Código 7-28. Crear una base de datos con nombre pubs2 en el dispositivo predeterminado de tamaño de 4 Mb.

```
create database mydb on default = 3, data_dev1 = 2
```

Código 7-29. Crear una base de datos con nombre mydb almacenando en el dispositivo predeterminado de tamaño 3 Mb y en el dispositivo data_dev1 2 Mb.

```
create database salesdb on sales_dev1 = 2
       log on sales_dev1 =2 with override
```

Código 7-30. Crear una base de datos con nombre salesdb almacenando los datos en el dispositivo sales_dev1 de tamaño de 2 Mb y el log de transacciones en el dispositivo sales_dev1 con la opción de override que permite mezclar datos y log en el mismo dispositivo.

Es recomendado poner el log en un dispositivo diferente al de datos.

Propiedad de la base de datos

¿Quién crea las bases de datos?

- Los administradores del sistema (logins con el rol SA) pueden crear bases de datos.
- El login que crea una base de datos es propietario de ella inicialmente.

Transfiriendo la propiedad de la base de datos

- Para transferir la propiedad de la base de datos, hay que acceder a esa base da datos usando **`use dbname`** seguido de **`sp_changedbowner`** en otro lote

```
sp_changedbowner login_name
```

Código 7-31. Sintaxis de sp_changedbowner en Sybase.

- El login debe ser válido en el servidor y no puede estar como usuario en la base de datos a transferir la propiedad

```
use productsdb
go
sp_changedbowner fred
go
```

Código 7-32. Uso de sp_changedbowner en Sybase.

- La propiedad de la base de datos **`master`** no puede transferirse

Otorgando permisos para **`create database`**

- Los administradores del sistema pueden otorgar permisos de **`create database`** a usuarios específicos:

```
grant create database to mary
```

Código 7-33. Uso de grant en Sybase.

- El control de uso de disco puede estar restringido en la autorización en la creación de base de datos

Efectos sobre las tablas del sistema.

- Cuando se crea una base de datos, las siguientes tablas del sistema en **`master`** son afectadas:
  - **`sysdatabases.`** Contiene un registro por cada base de datos en el SQL Server, incluyendo el nombre de la base de datos y su propietario
  - **`sysusages`**. Contiene un registro para cada fragmento del dispositivo para cada base de datos, indicando el tamaño y la dirección del disco del comienzo lógico para ese fragmento
- Para desplegar esta información, se puede consultar estas tablas o ejecutando el comando **`sp_helpdb`**.

Desplegando información de base de datos

```
name           db_size        owner   dbid    ...status
------------------------------------------------------------------------------
master         8.0 MB         sa      1       ... no options set
model          2.0 MB         sa      3       ... no options set
pubs2          2.0 MB         sa      4       ... no options set
salesdb        6.0 MB         sa      5       ... select into/bulkcopy
tempdb         2.5 MB         sa      2       ... select into/bulkcopy
```

Código 7-34. Salida de sp_helpdb en Sybase.

Cuando es seguido por un nombre de base de datos, **`sp_helpdb`** reporta información sobre esa base de datos.

Desplegando ubicación de la base de datos

¿Cómo sabe SQL Server en que dispositivo esta una base de datos?

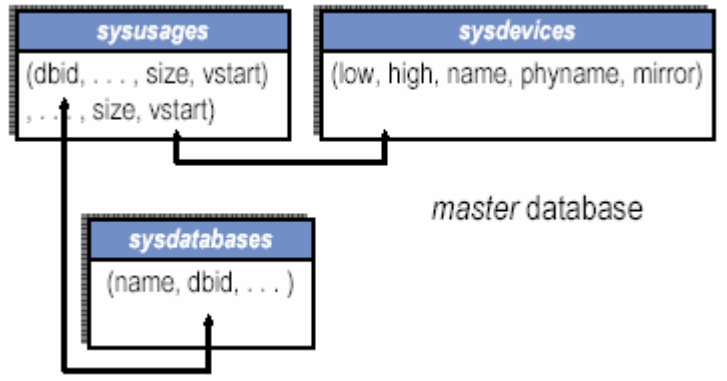


Ilustración 7-37. Esquema de sysusages, sysdevices, sysdatabases para ubicar dispositivos que ocupa una base de datos.

```
select * from sysusages where dbid = db_id("smalldb")
```

Código 7-35. Mostrar el contenido de la tabla sysusages donde la base se llama smalldb en Sybase.

dbid	segmap	lstart	size	vstart	pad	unreservedpgs
5	3	0	1024	16777216	NULL	680
5	4	1024	512	33554432	NULL	496

Ilustración 7-38. Salida del contenido de la tabla sysusages donde la base se llama smalldb en Sybase.

Para desplegar el mapeo de uno de estos fragmentos de dispositivos:

```
select low, high, name, phyname, mirrorname from sysdevices where 16777216
between low and high
```

Código 7-36. Mostrar campos de sysdevices para ubicar el nombre del dispositivo, su nombre físico de la dirección de la base de datos relacionada al número obtenido de vstart de sysusages.

low	high	name	physname	mirrorname
16777216	16782335	data_dev1	/syb/data_dev1.dat	NULL

Ilustración 7-39. Salida de los campos de sysdevices ubicando la posición inicial de una base de datos.

Desplegando ubicación de la base de datos y uso

`sp_helpdb db_name` despliega ubicación y uso del disco para esa base de datos:

```
sp_helpdb smalldb
```

Código 7-37. Uso del comando sp_helpdb en Sybase.

name	db size	owner	dbid	created	status
smalldb	3.0 MB	sa	5	May 5, 1993	no options set

device fragments	size	usage	free kbytes
data_dev1	2.0 MB	data only	1376
log_dev1	1.0 MB	log only	1008

Ilustración 7-40. Salida de sp_helpdb en Sybase.

Personalizando una base de datos

Cuando se crea una base de datos, el contenido de **`model`** es copiado a la nueva base de datos

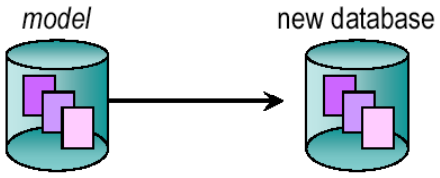


Ilustración 7-41. Esquema del uso de model en la creación de una base de datos.

Se puede personalizar **`model`** para que contenga procedimientos almacenados, tablas, reglas, tipos de datos de usuario, privilegios, y opciones para todas las futuras bases de datos. Con la restricción que sólo el administrador del sistema puede actualizar **`model`**.

Fijando las opciones de base de datos

Usando **`sp_dboption`**, las siguientes opciones de base de datos pueden ser fijadas:

- **`abort tran on log full`**
- **`allow nulls by default`**
- **`dbo use only`**
- **`ddl in tran`**

- **identity in nonunique index**
- **no chkpt on recovery**
- **no free space acctg**
- **read only**
- **select into/bulkcopy/pllsort**
- **single user**
- **trunc log on chkpt**

El usuario tipo SSO pueden habilitar o deshabilitar **free-space accounting**; los demás serán fijados por el dueño de la base de datos (o un SA actuando como un dbo). Las opciones no podrán ser cambiadas en la base de datos **master**

```
sp_dboption [dbname, option_name, {true |false}]
```

Código 7-38. Sintaxis de sp_dboption en Sybase.

```
use master /* debe estar en master */
go
sp_dboption "smalldb", "read only", true
go
use smalldb /* debe usar la base de datos y ... */
go
checkpoint /* checkpoint la base de datos */
go
```

Código 7-39. Uso de sp_dboption

Cualquier usuario puede usar **sp_helpdb** para desplegar las opciones actuales de la base de datos.

Uso del espacio

Monitoreando el uso del espacio

La base de datos, tanto el segmento de log como el segmento de datos, pueden llenarse. Si esto sucede, todas las modificaciones de datos son suspendidas. Se pueden usar las siguientes herramientas para monitorear el uso del espacio en la base de datos:

- **sp_helpdb**
- **sp_helpsegment**
- **sp_spaceused**
- Threshold Manager

¿Qué hacer cuando no se tiene espacio?

Si no se tiene espacio en el log, se tiene que truncarlo. Pudiendo ser posibles soluciones: el uso de **dump**/**truncate** frecuentemente.

Si no se tiene espacio en los segmentos de datos, hay que intentar de liberar espacio eliminando objetos no usados.

Otra alternativa: Extender la base de datos (datos y/o log).

Ilustración 7-42. Ejemplificación de la expansión de una base de datos en Sybase.

Expansión de la base de datos

Usando **`alter database`**, los propietarios de base de datos y administradores del sistema asignan espacio adicional a una base de datos sobre los mismos o diferentes dispositivos

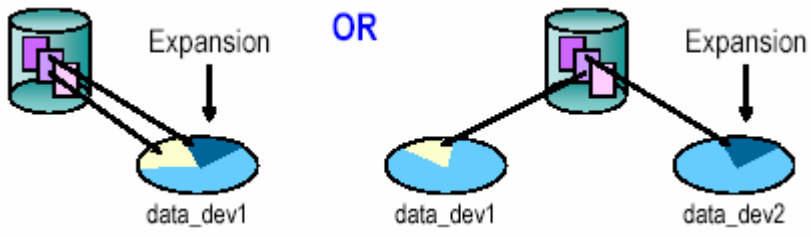


Ilustración 7-43. Comparación de expandir el espacio en un dispositivo y en diferentes dispositivos.

El tamaño de la base de datos no puede ser disminuido.

```
alter database database_name
       [on {default | database_device} [= size]]
       [log on database_device [= size]]
```

Código 7-40. Sintaxis de alter database en Sybase.

El tamaño es especificado como incrementos de espacio adicional; predeterminado, 1 MB

Ejemplos:

```
alter database pubs2
```

Código 7-41. Incrementa en 1 Mb el tamaño de la base de datos pubs2 en Sybase.

```
alter database pubs2 on data_dev1 = 3
```

Código 7-42. Incrementa en 3 Mb el tamaño de la base de datos pubs2 en el dispositivo data_dev1 en Sybase.

```
alter database pubs2 on default = 2
```

Código 7-43. Incrementa en 3 Mb el tamaño de la base de datos pubs2 en el dispositivo predeterminado en Sybase.

```
create database salesdb on data_dev1 = 5
alter database salesdb on data_dev2 = 2
alter database salesdb on data_dev3 = 1
```

Código 7-44. Crea una base de datos con nombre salesdb en el dispositivo data_dev1 con tamaño de 5 Mb, incrementa el tamaño de la base de datos salesdb en 2 Mb en el dispositivo data_dev2, incrementa el tamaño de la base datos salesdb en 1 MB en el dispositivo data_dev3.

Tips para la expansión de una base de datos.

- Puede expandir una base de datos mientras este en uso.
- Solo se puede expandir la base de datos **`master`** en el dispositivo **`master`**.
- Si **`tempdb`** o **`model`** es expandido y el dispositivo **`master`** es reconstruido entonces se vuelve a expandir.
- No hay que hacer a **`model`** más grande que **`tempdb`**.
- Si **`model`** es más grande que **`tempdb`**, SQL Server no reiniciará.

Separando el log y moverlo a un dispositivo diferente.

Para una base de datos creada sin la opción **`log on`** (esto es, sin un log separado), hay que hacer lo siguiente:

- Respaldar el log al truncarlo.
- Modificar la base de datos hacia un nuevo dispositivo.
- Ejecutar **`sp_logdevice`** para hacer ese dispositivo en dispositivo de log.

Realizar estos pasos con un mínimo de tiempo entre cada uno para evitar que los registros sean escritos al dispositivo original.

Efecto:

- Las páginas antiguas serán asignadas al dispositivo anterior.
- Los registros existentes serán desasignados tanto el **`log`** sea truncado.
- Resultado: El **`log`** crece en el nuevo dispositivo.

Expandiendo el log

En el mismo dispositivo:

```
alter database pubs2 log on log_dev1 = 1
```

Código 7-45. Expande el log de la base de datos pubs2 en el dispositivo log_dev1 en 1 Mb.

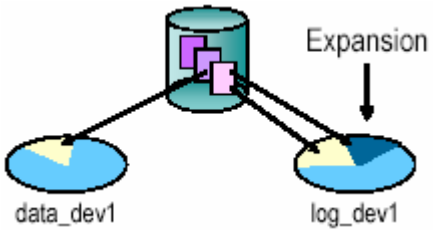


Ilustración 7-44. Esquema de expandir el log en el mismo dispositivo en que esta guardado el log.

En un nuevo dispositivo:

```
alter database pubs2 log on log_dev2 = 4
```

Código 7-46. Expande el log de la base de datos pubs2 en el dispositivo log_dev2 en 4 Mb.

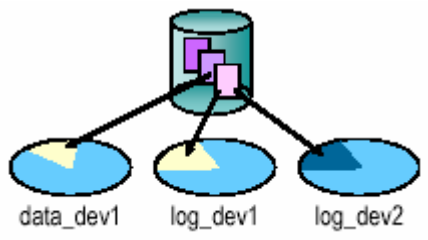


Ilustración 7-45. Esquema de expandir el log en otro dispositivo del que esta guardado el log.

Eliminando bases de datos

Los DBO y SA pueden borrar una base de datos ejecutando **`drop database`**. La base de datos no debe estar en uso.

Cuando se borra una base de datos el sistema realiza:

- En bases de datos experimentales o antiguas, se libera el espacio.
- Antes se trata de recuperar las base de datos dañadas.

Control de acceso

Entre las tareas del control de acceso al RDBMS de Sybase se encuentran:

- Describir el sistema de roles del SQL Server y autorizaciones asociadas.
- Permisos y manejo de tablas del sistema del SQL Server referidos a roles.
- Describir logins del SQL Server, usuarios de base de datos, y grupos.
- Adicionar y manejar logins, usuarios de base de datos y grupos.
- Otorgar y revocar privilegios.
- Describir roles del usuario.
- Otorgar y manejar roles de usuario.
- Describir autorizaciones de proxy y como son usados.

Roles de Usuario y de Sistema

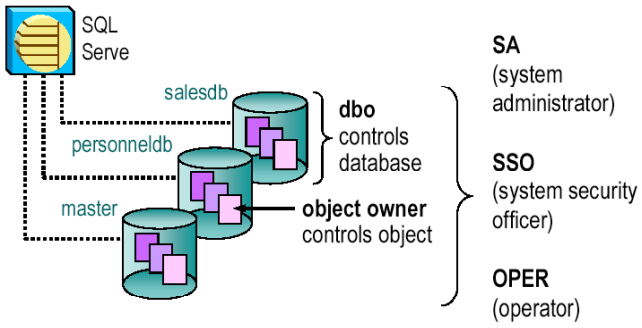


Ilustración 7-46. Roles de SQL Server y Usuarios especiales.

El login SA

Cuando SQL Server en instalado por primera vez, el login SA:

- Ha sido asignado con los tres roles especiales (**`system administrator`**, **`system security officer`**, y **`system operator`**).
- Ejecuta todos los comandos SQL.
- Es propietario de la base de datos **`master`**.
- Es tratado como dueño de la base de datos (dbo) en todas las bases de datos.
- Tiene acceso a todas las bases de datos y objetos de base de datos.

El password del login sa es inicialmente nulo, pero debe ser cambiado. Una vez cambiado, no puede ser nulo otra vez.

Ilustración 7-47. Roles especiales del login sa.

En la instalación, el login sa puede hacer todo, puede:

- Mantener los tres roles administrativos
- Otorgarlos a uno o más logins y luego bloquear el login sa.
- Revocar los roles del login sa.

Administradores del Sistema (Rol SA)

- Manejo del almacenamiento del disco.
- Borrar, modificar, bloquear, y desbloquear logins.
- Otorgar/revocar roles SA.
- Crear bases de datos de usuario; otorgar propiedad sobre ellas.
- Otorgar ciertos permisos a los usuarios del SQL Server.
- Afinar SQL Server cambiando los parámetros de configuración.
- Cerrar SQL Server y sus procesos.
- Monitorear la recuperación de base de datos en el arranque del SQL Server y utilizar ciertas herramientas para el diagnóstico de problemas en el sistema.

Oficiales de Seguridad del Sistema (Rol SSO)

- Los Oficiales de Seguridad del Sistema:
- Crean logins en el SQL Server (asignando passwords iniciales)
- Cambian passwords
- Fijan un intervalo de expiración del password
- Crean, otorgan, y revocan roles de usuario
- Otorgan autorización para uso del proxy
- Otorgan y revocan roles SSO y Oper
- Manejan el sistema de auditoria
- Bloquean y desbloquean logins
- SSO no puede modificar o borrar logins; esto sólo puede hacerlo un login con rol SA

Operadores SQL Server (Rol Oper)

- Los Operadores pueden respaldar y cargar todas las bases de datos y logs de transacciones en el servidor.
    - Realizan **`dump db`**, **`dump transaction`**, **`load db`**, y **`load transaction`**.
    - No tienen que ser propietarios de una base de datos para realizar tareas de mantenimiento.
- Los propietarios de base de datos realizan tareas de mantenimiento de sus propias bases de datos en lugar de, o en adición al operador.

Propietarios de Bases de Datos (dbo)

- Un propietario de una base de datos es:
    - El creador de una base de datos.
    - Alguien a quién se le ha transferido la propiedad de una base de datos.
    - Se le ha otorgado la autorización de crear una base de datos.

- Adiciona y retira usuarios de la base de datos con **sp_adduser**.
- Otorga y revoca permisos a usuarios para crear objetos en la base de datos y ejecutar comandos con **grant**.
- Ejecuta algunas tareas de operador del sistema en su propia Base de Datos.
- Ejecuta **checkpoint** y **dbcc** en las Bases de Datos.
- Tiene todos los privilegios sobre todos los objetos en las Bases de Datos usando **setuser**.

Propietarios de Objetos de una Base de Datos

- Un usuario que crea objetos en una Base de Datos es dueño de esos objetos.
  - Generalmente es el dueño de la Base de Datos, pero el dbo puede otorgar permisos de crear objetos a otro usuario.
- Propietarios de objetos de Bases de Datos:
  - Se les otorga todos los permisos de acceso para sus objetos.
  - Pueden otorgar permisos en sus objetos a otros usuarios, incluyendo al dbo.
  - No pueden otorgar permisos que modifiquen el esquema de un objeto, como **create index**, **alter table**, **drop object**.
  - No pueden transferir la propiedad de un objeto.

Introducción a la Seguridad del SQL Server

SQL Server tiene un sistema de seguridad de capas:

Acceso a:		Controlado por:
server, database	→	adding/dropping logins, users
objects, commands	→	granting/revoking permissions

Para controlar el acceso:

- Agregar y/o borrar logins y usuarios de bases de datos.
- Agregar y/o borrar grupos y roles de usuarios.
- Otorgar y revocar permisos en objetos o comandos a usuarios, grupos, o roles.

Logins del sistema

- Para conectarse a un SQL Server, es necesario un login.
- Los detalles de un login son guardados en la tabla **syslogins** en la base de datos **master**.

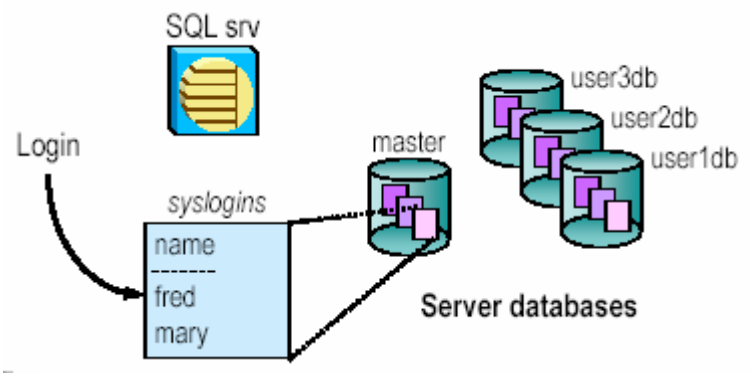


Ilustración. Manejo de los detalles de login en la base de datos master en Sybase.

Agregando Logins

Un rol SSO puede agregar logins usando **`sp_addlogin`**.

```
sp_addlogin login_name, passwd [, defaultdb
        [, deflanguage [, fullname]]]
```

Código 7-47. Sintaxis de sp_addlogin.

Ejemplos:

```
sp_addlogin claire, bleurouge, public_db, french, "Claire Barr"
```

Código 7-48. Uso de sp_addlogin para agregar un login con nombre Claire, con el password bleurouge, con base de datos por defecto public_db, con lenguaje por defecto french, y de nombre completo Claire Barr en Sybase.

```
sp_addlogin robby, playball, education
```

Código 7-49. Uso de sp_addlogin para agregar un login con nombre robby, con el password playball, con base de datos por defecto education en Sybase.

Los Passwords deben ser por lo menos de 6 bytes; el nulo no esta permitido.

Especificar una base de datos predeterminada (si no se especifica, **`master`** es la predeterminada, por lo que no es buena idea).

La tabla syslogins

Cuando se agrega un login, un registro es agregado a **`syslogins`** en **`master`**.

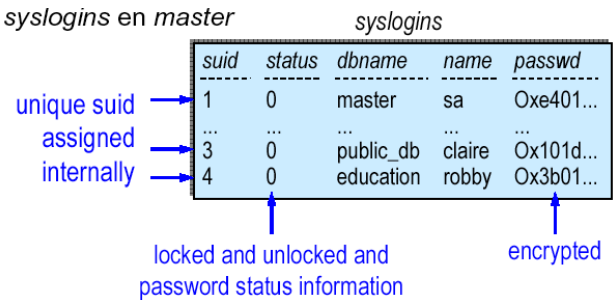


Ilustración 7-48. Descripción de syslogins en Sybase.

Para ver syslogins, hay que usar **`select`**.

Funciones de Logins

SA:

```
sp_modifylogin login_name, option, value
sp_droplogin login_name
```

Código 7-50. Sintaxis de uso de sp_modifylogin y sp_droplogin para el sa en Sybase

SSO:

```
sp_addlogin login_name, password,...
sp_password caller_passwd, new_passwd [,login_name]
sp_configure password, n
```

Código 7-51. Sintaxis de uso de sp_addlogin, sp_password y sp_configure para el sso en Sybase

SA o SSO:

```
sp_locklogin [login_name, "{lock | unlock"}]
```

Código 7-52. Sintaxis de uso de sp_locklogin para sa y sso en Sybase.

Usuarios de bases de datos

Para acceder a una base de datos, el usuario debe estar dado de alta en esa base de datos

Los usuarios son listados en la tabla **`sysusers`** en cada base de datos

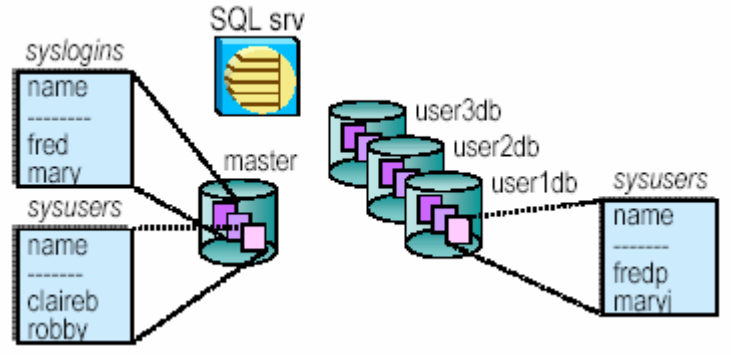


Ilustración 7-49. Distribución de sysusers en las bases de datos en Sybase.

Agregando Usuarios a la Base de Datos

```
sp_adduser login_name [, name_in_db
[, grpname]]
```

Código 7-53. Sintaxis de sp_adduser para agregar usuarios a una base de datos en Sybase.

```
sp_adduser claire, claireb
sp_adduser robby
sp_adduser guest
```

Código 7-54. Uso de sp_adduser en Sybase.

Solo el propietario de la base de datos puede agregar usuarios

Los administradores del sistema automáticamente son dbo en cada base de datos que accedan y por lo tanto pueden agregar usuarios también.

La tabla sysusers

Al agregar un usuario se inserta un renglón en **`sysusers`**

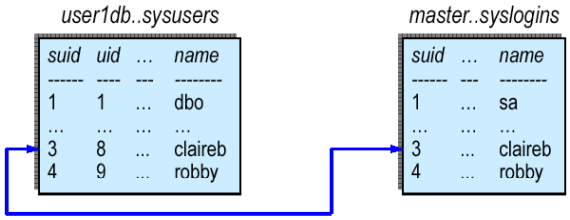


Ilustración 7-50. Manejo de sysusers y syslogins en Sybase.

Para ver **`sysusers`**, hay que usar **`select * from sysusers`**. Para desplegar una lista de los usuarios de una base de datos con su nombre de login, se puede usar **`sp_helpuser`**.

Acceso a la Base de Datos

Cuando intenta acceder a una base de datos, SQL Server:

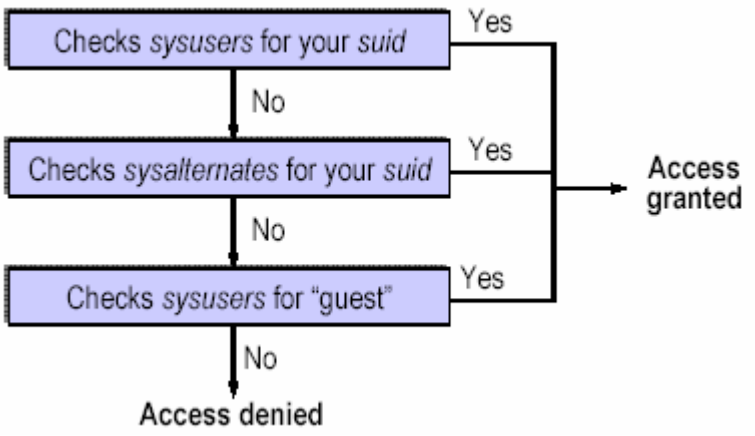


Ilustración 7-51. Pasos que se sigue al intentar acceder a una base de datos.

Alias

Se puede tratar a más de un login como un mismo usuario de una base de datos, tomando todos sus privilegios.

Hacer que logins sean dbo u otro nombre de usuario.

Si las actividades del alias son auditadas, la identidad real lo será también.

Para crear un alias, se usa **`sp_addalias`**.

```
sp_addalias loginame, name_in_db
```
Código 7-55. Sintaxis de sp_addalias para crear un alias.

Usando **sp_addalias** agrega un registro en **sysalternates** en la base de datos.

Por ejemplo, Mary es dueña de una base de datos; ella quiere que Jane use la base de datos como dueña; Jane tiene un login pero no es usuario de la base de datos de Mary; Mary ejecuta el comando: **sp_addalias jane, dbo**.

Revocando Acceso al SQL Server o Base de Datos

Para borrar cuentas de login, se usa:

```
sp_droplogin login_name
```
Código 7-56. Sintaxis de sp_droplogin en Sybase.

Para bloquear o desbloquear cuentas o desplegar una lista de las cuentas bloqueadas, se usa:

```
sp_locklogin [login_name, "{lock | unlock}"]
```
Código 7-57. Sintaxis de sp_locklogin en Sybase.

Para borrar un usuario, se usa:

```
sp_dropuser name_in_db
```
Código 7-58. Sintaxis de sp_dropusr en Sybase.

Para borrar el alias de un usuario, use:

```
sp_dropalias login_name
```
Código 7-59. Sintaxis de sp_dropalias en Sybase.

Para borrar un usuario o alias de una base de datos, debe de estar en esa misma base de datos

Grupos de usuarios

Agregando y Asignando Grupos

Proporcionar una forma conveniente para otorgar y revocar permisos a más de un usuario y nombre colectivo de muchos usuarios

```
sp_addgroup grpname
```
Código 7-60. Sintaxis de sp_addgroup para agregar un grupo de usuarios en Sybase.

```
sp_addgroup senioreng
```
Código 7-61. Uso de sp_addgroup en Sybase.

Se crea grupos antes de agregar usuarios porque el comando **sp_adduser** deja que se le asigne un grupo al usuario, por ejemplo:

```
sp_adduser robert, bob, senioreng
```
Código 7-62. Uso de sp_adduser para agregar un usuarios a un grupo.

Para reasignar o asignar un usuario a un grupo, se usa **sp_changegroup**:

```
sp_changegroup senioreng, steve
```

Código 7-63. Uso de sp_changegroup para reasignar un grupo a un usuario en Sybase.

El grupo **public** esta siempre presente, y todos están en él; el usuario sólo puede estar asignado a un grupo.

Desplegando Información de Grupos

Se usa **sp_helpgroup** para desplegar los grupos en una base de datos use:

```
sp_helpgroup [grpname]
```

Código 7-64. Sintaxis de sp_helpgroup en Sybase.

sp_helpgroup sin un parámetro, regresa todos los grupos en la base de datos (incluyendo roles de usuario y de sistema).Con un parámetro, regresa usuarios de ese grupo.

La tabla **sysusers** tiene un registro para cada grupo. Los roles de usuario y de sistema aparecen como grupos en la tabla **sysusers** de todas las bases de datos aunque trabaja de manera diferente.

Funciones y Procedimientos Almacenados Útiles

Funciones del Sistema:

- **suser_id( )**
- **db_id( )**
- **user_id( )**
- **suser_name( )**
- **db_name( )**
- **user_name( )**
- **proc_role( )**

Ejemplos:

```
select suser_id( )
select suser_id ("fred")
```

Código 7-65. Uso de la función suser_id en Sybase.

Procedimientos almacenados:

- **sp_helpuser**
- **sp_helpgroup**

Logins del sistema

Roles de Sistema del SQL Server

Cuando SQL Server es instalado por primera vez, el login sa tiene los roles SA, SSO, y Oper. Para otorgar esos roles a otros logins, se usa **sp_role**. La sintaxis para **sp_role** es:

```
sp_role "{grant | revoke}",
"{sa_role | sso_role | oper_role}", login_name
```

Código 7-66. Sintaxis de sp_role en Sybase.

```
sp_role "grant", "sa_role", robby
```
Código 7-67. Uso de sp_role en Sybase.

Usuarios con el rol de SSO otorgan y revocan los roles SSO y Oper; los usuarios con el rol SA otorgan y revocan el rol SA.

Al ejecutar **sp_role** agrega o borra un registro en **master..sysloginroles**.

```
sp_role "grant", "sa_role", robby
```
Código 7-68. Uso de sp_role en Sybase.

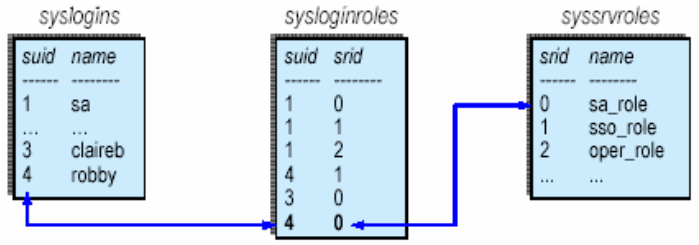


Ilustración 7-52. Tratamiento de sysloginroles al otorgar un rol a un usuario.

sysloginroles contiene asignaciones de roles configurados. Estos toman efecto en el siguiente ingreso al servidor.

proc_role()

La función **proc_role** checa los roles cuando un procedimiento es ejecutado. Si regresa **1** el usuario posee el rol específico. Un ejemplo usando **proc_role** en el procedimiento almacenado **test_proc** en el que se requiere que lo ejecute un administrador del sistema:

```
create proc test_proc
as
if (proc_role("sa_role") = 0)
begin
        print "You do not have the right role."
        return -1
end
else
        print "You have SA role."
return 0
```
Código 7-69. Uso de la función proa_role en un procedimiento almacenado en Sybase.

Activando y Desactivando Roles del Sistema

Al entrar, los roles que han sido asignados a su login son activados automáticamente (si hubiera alguno). Para desactivar un rol, se usa **set role**

```
set role "{sa_role | sso_role | oper_role}" off
```
Código 7-70. Sintaxis de set role para (des)activar un rol en Sybase.

```
set role "sa_role" off
```
Código 7-71. Uso de set role en Sybase.

Toma efecto inmediatamente y dura para toda la sesión, a menos que se reactive. Para reactivar un rol, se usa **set role ... on**.

Desplegando Información acerca de Logins

sp_displaylogin despliega información acerca de un login, incluyendo que roles le han sido otorgados.

```
1> sp_displaylogin sa
2> go
Suid: 1
Loginame: sa
Fullname:
Configured Authorization: sa_role sso_role oper_role replication_role
Locked: NO
Date of Last Password Change: Nov 17 1994 12:02PM
(return status = 0)\
```

Código 7-72. Salida de sp_displaylogin en Sybase.

Los roles listados están actualmente configurados.

Manejo de Roles del Sistema

Se recomienda otorgar roles SA, SSO, y Oper a logins específicos, y luego bloquear el login **sa**, con el fin de incrementar la responsabilidad de un usuario ya que se auditan sus actividades.

Para bloquear un login, se usa **sp_locklogin**.

```
sp_locklogin [login_name, "{lock | unlock}"]
```

Código 7-73. Sintaxis de sp_locklogin en Sybase.

Usando **sp_locklogin** sin parámetros regresa una lista de logins bloqueados.

SQL Server no permite bloquear el último login desbloqueado con el rol SA, ni el último login desbloqueado con el rol SSO. **sp_droplogin**, **sp_locklogin**, y **sp_role** tienen que verificar que se cumpla esta regla

Permisos en los comandos

Otorgando y revocando privilegios a usuarios

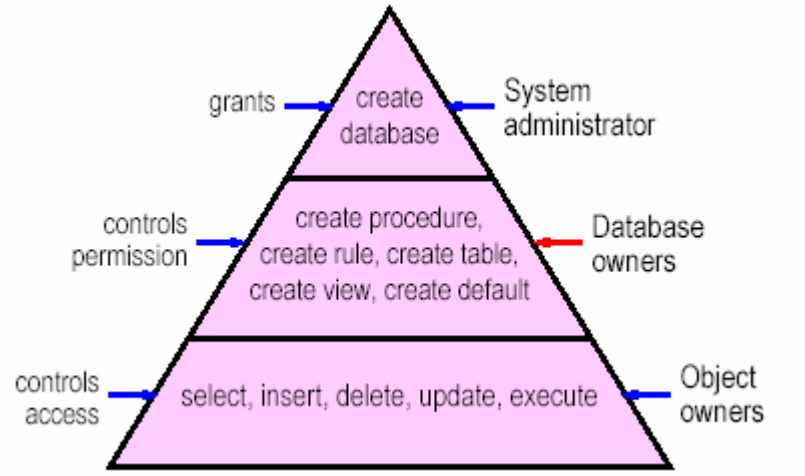


Ilustración 7-53. Niveles de permiso para el dueño de objetos, de bases de datos y del sa.

Otorgando privilegios en comandos

Se usa **grant** para dar permisos en comandos:

```
grant {all [privileges] | command_list} to {public | name_list | role_name}
```

Código. Sintaxis de grant para otorgar privilegios en comandos.

```
grant create rule to fred
grant create rule, create table to engineering
grant create database to mary, john  #(SA only)
```

Código 7-74. Uso de grant para otorgar privilegios en comandos en Sybase.

Los permisos de comandos sobre bases de datos que no pueden ser dados: **dbcc**, **dump database**, **dump tran**, **load database**, **load transaction**, **setuser**.

Revocando privilegios en comandos

Se usa **revoke** para quitar permisos de comandos:

```
revoke {all [privileges] | command_list}
       from {public | name_list | role_name}
```

Código 7-75. Sintaxis de revoke en Sybase.

```
revoke create table from public
revoke create rule from fred
revoke create database from mary, john
```

Código 7-76. Uso de revoke para quitar privilegios en comandos a usuarios en Sybase.

Sólo un rol SA puede establecer **revoke create database**.

Permisos en los objetos

Se usa **grant** para dar permisos en objetos específicos

```
grant {all [privileges] | permission_list}
on {table_name [(column_list)]
       | view_name [(column_list)]
       | stored_procedure_name}
to {public | name_list | role_name}
[with grant option]
```

Código 7-77. Sintaxis de grant para otorgar privilegios en objetos en Sybase.

```
grant insert, delete on titles to mary, sales
grant update on titles (price, advance) to public
grant execute on new_proc to sa_role
```

Código 7-78. Uso de grant para otorgar privilegios en objetos en Sybase.

Permisos que no pueden ser otorgados: **create index**, **create trigger**, **alter table**, **drop table**, **truncate table**, **update statistics**.

Revocando permisos sobre los objetos.

Se usa **revoke** para quitar permisos sobre objetos específicos

```
revoke [grant option for]
{all [privileges] | permission_list}
on {table_name [(column_list)]
       | view_name [(column_list)]
       | stored_procedure_name}
```

```
from {public | name_list | role_name}
[cascade]
```
Código 7-79. Sintaxis de revoke para quitar permisos sobre objetos en Sybase.

```
revoke insert, delete on titles from mary, sales
revoke update on titles (price, advance) from public
revoke execute on new_proc from oper_role
```
Código 7-80. Uso de revoke para quitar permisos sobre objetos en Sybase.

La opción de **`grant...with grant`** permite algún(os) usuarios otorgar ciertos privilegios a otros usuarios

Ejemplo:

```
grant select on mytable to philip with grant option
# Como philip:
grant select on mytable to mary
```
Código 7-81. Uso de grant … with grant para otorgar privilegios a otro usuario en Sybase.

Se puede dar la opción **`grant...with grant`** solo a usuarios individuales, no a public, grupos, o roles. Es necesario manejar con cuidado esta opción, o se puede perder el control sobre permisos de los objetos.

Uso de **`revoke grant...cascade`**:

```
revoke select on mytable from philip
```
Código 7-82. grant y select han sido revocados a philip, y select (y grant, han sido revocados) a mary en Sybase.

```
revoke select on mytable from philip, cascade  Lo mismo de arriba
```
Código 7-83. grant y select han sido revocados a philip, y select (y grant, han sido revocados) a mary en Sybase.

Autorización **`create schema`**

El comando **`create schema`** crea una colección de objetos que son propiedad de un usuario. Se puede incluir permisos otorgados en esos objetos.

```
create schema authorization anna
create table tableA (col1 int, col2 int)
create view v1 as select col1 from tableA
grant select on v1 to public
```
Código 7-84. Uso de create schema en Sybase.

Con **`create schema`** permite crear y dar permisos de tablas relacionadas en una transacción. Si cualquier sentencia dentro del esquema falla, todo el comando es cancelado (**`roll back`**)

Propiedad de Objetos

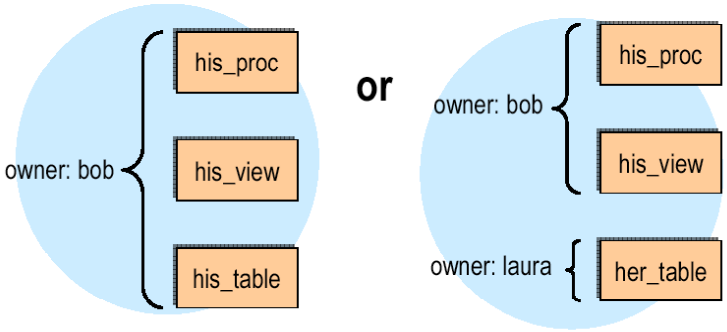


Ilustración 7-54. Esquema de dos casos de propiedad objetos.

Desplegando Permisos

Los comandos que otorgan y agregan privilegios agregan un registro en **sysprotects**. Para desplegar permisos en un objeto dado se usa **sp_helprotect**.

```
sp_helprotect object_name
```

Código 7-85. Sintaxis de sp_helprotect para desplegar permisos en un objeto en Sybase.

```
sp_helprotect titles
```

Código 7-86. Uso de sp_helprotect para desplegar permisos en un objeto en Sybase.

Para desplegar permisos en un usuario dado, también se usa **sp_helprotect**.

```
sp_helprotect user_name
```

Código 7-87. Sintaxis de sp_helprotect para desplegar permisos de un usuario en Sybase.

```
sp_helprotect laura
```

Código 7-88. Uso de sp_helprotect para desplegar permisos de un usuario en Sybase.

Permisos sobre Procedimientos del Sistema

Los administradores del sistema controlan el acceso a los procedimientos y tablas del sistema en **master**. Las características de los procedimientos del sistema residen en **sybsystemprocs** pero pueden ejecutarse en cualquier base de datos mientras que los permisos se fijan en **sybsystemprocs..sysprotects** por los administradores del sistema.

El script **installmaster** otorga permisos a public para algunos procedimientos del sistema, los administradores del sistema pueden revocar esos permisos.

Para crear nuevos procedimientos de sistema que todos puedan usar se tienen que crear en **sybsystemprocs**; y que comiencen con **sp_** y hay que otorgar los permisos apropiados y respaldar la base de datos.

Roles definidos por usuarios

Roles definidos a usuarios:

- Las asignaciones de roles en todo el servidor pueden incluir muchos logins, por ejemplo, un rol de contador, maestro.
- Son diferentes de los grupos ya que un grupo es específico de una base de datos y se le asocian ids de usuario.
- Son otorgados a un login específico o a otro rol.
- Son administrados dentro de jerarquías de roles para un eficiente manejo de roles.

Cualquier login puede tener muchos roles. Una vez definido, los roles pueden ser apagados y prendidos dinámicamente por el usuario, teniendo así mayor flexibilidad con respecto a los grupos.

Grupos versus roles de usuarios

Las personas pueden ser parte de un grupo, como de ingeniería, porque trabajan en el departamento de ingeniería. Un departamento puede contener muchos roles. Por lo tanto, las personas pueden tener roles de usuarios asociados con su login, como: **`senioreng_role`**, **`manager_role`** y **`teacher_role`**.

El control de acceso puede ser establecido basado en: el grupo al cuál pertenece una persona o el rol que la persona juega en la organización.

Limitaciones de los grupos

- Los grupos son internos en una base de datos y no a un servidor, en contraste con los roles de usuario.
- Un grupo no puede tener usuarios de muchas bases de datos.
- Los permisos pueden asignarse a objetos solo en la base de datos en la cuál el grupo fue creado.
- Todos los usuarios son miembros del grupo public; para excluirlos, necesita ejecutar sentencias de **`revoke`** explicitas.
- Los usuarios pueden ser miembros de solo dos grupos en la misma base de datos, public y otro grupo.
- Cuando quita un usuario de un grupo, los privilegios de acceso asociados con sus logins siguen teniendo efecto.

¿Por qué Implementar Roles de Usuario?

Los roles de usuario cumplen con el estándar ANSI SQL3 para el control de acceso basado en roles y proporcionan extensiones para mejorar su uso como:

- La jerarquía de roles pueden ser formados para manejar y controlar el acceso.
- Los roles pueden ser definidos para una aplicación específica.
- Los roles pueden ser definidos mutuamente excluyentes, mejorando la seguridad.

Planeando roles de usuario

Antes de crear roles de usuario, hay que desarrollar una estrategia, preguntándose lo siguiente:

- ¿Qué rol de usuario quiero crear?
- ¿Cuales son las responsabilidades para cada rol de usuario?
- ¿Qué posición jugará en la jerarquía de roles cada rol de usuario (por ejemplo, controlador, contador)?
- ¿Cuales son los roles en la jerarquía que deben ser mutuamente excluyentes?

Implementando roles de usuario

Cuatro pasos son requeridos para implementar los roles de usuario:

- Un login con privilegios de SSO debe crear el rol de usuario.
- Después de que el rol es creado, cualquier login con el permiso **`with grant`** puede otorgar privilegios de acceso a un rol de usuario.
- El login con privilegios SSO puede otorgar a otros logins la membresía en un rol de usuario.
- A los usuarios que les fueron otorgados la membresía a un rol deben activar explícitamente sus roles para obtener los privilegios asociados.

```
#Un login con privilegios SSO crea el rol
create role manager_role

#Un usuario con el permiso with grant otorga privilegios de acceso al rol
grant update on employee_table to manager_role
grant select on employee_list to manager_role

#El login con privilegios SSO otorga a otros logins la membresía del rol
grant role manager_role to mary

#Un usuario activa el rol
set role manager_role on
```

Código 7-89. Implementación de roles en Sybase.

Creando roles de usuario

Para crear un rol de usuario se usa **`create role`**:

```
create role role_name [with passwd password]
```

Código 7-90. Sintaxis de create role para crear roles de usuario en Sybase.

```
create role ta_role
create role teacher_role
create role professor_role with passwd publishme
```

Código 7-91. Uso de create role en Sybase.

Este comando, **`create role`**, requiere privilegios SSO. Usualmente se usan nombres convencionales para los roles de usuario. Por ejemplo, se puede usar "**`_role`**" como un sufijo.

Creando passwords de roles de usuario

Es recomendable crear un password para un rol de usuario cuando el rol es creado inicialmente

```
create role professor_role with passwd publishme
```

Código 7-92. Uso de create role asignado al role un password en Sybase.

Para cambiar un password o un conjunto de passwords después de que el rol de usuario es creado, se usa **`alter role`**.

```
alter role role_name add passwd password
```
Código 7-93. Sintaxis de alter role para cambiar el password a un role en Sybase.

```
alter role teacher_role add passwd paymemore
```
Código 7-94. Uso de alter role para cambiar el password a un role en Sybase.

Se usa este comando para ambos roles, de sistema y de usr. El comando falla si un password esta ya asignado.

```
alter role role_name drop passwd
```
Código 7-95. Sintaxis de alter role para borrar un password en Sybase.

Otorgando privilegios de acceso a roles de usuario

Un SSO define los roles de usuario por motivos de seguridad. Después de crear un rol, el SSO otorga privilegios de acceso, por ejemplo:

```
grant update on student_info to teacher_role
```
Código 7-96. Otorgamiento de privilegios de actualización a usuarios.

Los roles de usuarios son a nivel servidor, y se almacenan en la base de datos **`master`**, y están asignados a los logins. Se puede otorgar o revocar permisos a los roles de usuarios solo desde la base de datos donde residen actualmente los objetos. Los usuarios con el privilegio **`with grant`** también pueden otorgar privilegios de acceso a los roles de usuario. El SSO puede asignar o revocar roles de usuarios a cualquier login.

Jerarquía de roles de usuario

Una jerarquía de roles es creada cuando los roles "contienen" otros roles

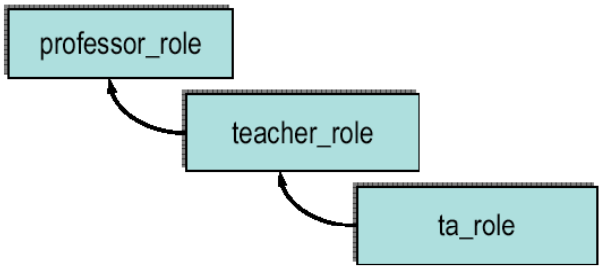


Ilustración 7-55. Esquema de jerarquía de roles.

Precedencia de acceso: grupos versus roles de usuario

Cuando los privilegios de acceso son determinados, los permisos son verificados en ese orden. Como se vio en el caso anterior, la implicación de que ambos roles **`ta_role`** y **`teacher_role`** son roles definidos por el usuario como parte del **`professor_role`**, entonces **`ta_role`** es implícitamente parte de **`professor_role`**.

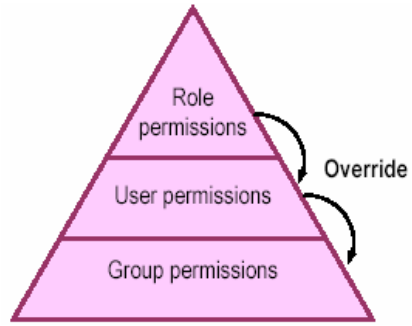


Ilustración 7-56. Esquema de precedencia de acceso sobre permisos y roles.

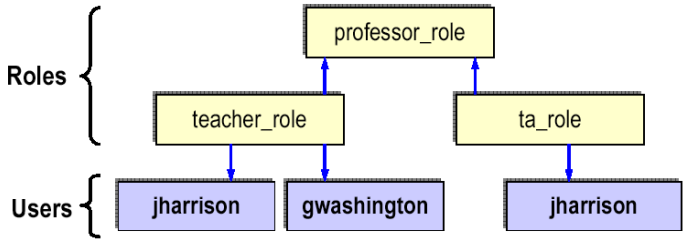


Ilustración 7-57. Otorgando roles de usuario a logins (u otros roles de usuario).

Activando el rol de usuario

Asignando un rol de usuario a un login le da la oportunidad al usuario de activar el rol (cuando entra al servidor). El login debe activar el rol de usuario para ganar privilegios asociados con el rol. Para activar un rol de usuario, se usa **`set role`**.

```
set role role_name [with passwd password] on
```

Código 7-97. Sintaxis de set role para activar un rol de usuario en Sybase.

```
set role ta_role on
set role professor_role with password publishme on
```

Código 7-98. Uso de set role para activar roles de usuario en Sybase.

Revocando roles de usuario de los logins (u otros roles de usuario)

Para revocar un rol de usuario de un login u otro rol de usuario, se usa **`revoke role`**

```
revoke role role_revoked [, role_revoked, ...] from grantee [, grantee, ...]
```

Código 7-99. Sintaxis de revoke role para revocar un rol de usuario en Sybase.

```
revoke role ta_role from jharrison
```

Código 7-100. Uso de revoke role para revocar un rol de usuario en Sybase.

El que otorga puede ser un login u otro rol de usuario, debe tener privilegios de SSO para ejecutar esta tarea. No puede revocar un rol de usuario de un login si a este nunca le fue otorgado el rol.

Borrando un Rol de Usuario

Un login con privilegios SSO borra el rol

```
drop role role_name [with override]
```

Código 7-101. Sintaxis de drop role para borrar un rol en Sybase.

```
drop role teacher_role
```

Código 7-102. Uso de drop role para borrar un rol en Sybase.

Si la opción **`override`** es usada, la sentencia **`drop`** quita los permisos otorgados asociados al rol en cada base de datos, o todo el servidor. Si la opción **`override`** no es usada, cualquier permiso otorgado al rol deberá ser revocada directamente antes de ejecutar este comando, de otra manera, se marcará un error.

Cuando se completa con éxito, **`drop`** borra todos los miembros logins del rol y borra el rol de cualquier jerarquía de roles en las cuales estaba asociado

Desplegando la información de un rol de usuario

Un procedimiento del sistema llamado **`sp_displayroles`** es usado para desplegar:

- Roles otorgados a un login
- Roles contenidos por otro rol
- Roles descendentes para un rol particular

```
sp_displayroles [login_name | role_name] [,expand_up | expand_down]
```

Código 7-103. Sintaxis de sp_displayroles en Sybase.

Procedimientos almacenados para roles de usuario

Para ver que roles están activos actualmente, se puede usar **`sp_activeroles`**:

```
sp_activeroles [expand_down]
```

Código 7-104. Sintaxis del procedimiento almacenado sp_activeroles en Sybase.

Para desplegar información de permisos otorgados a un usuario, el cuál incluye permisos para un grupo o miembro de un rol, se usa **`sp_helprotect`**:

```
sp_helprotect name, username, 'grant' [, 'none' | 'granted' | 'enabled' ] [,
rolename]
```

Código 7-105. Sintaxis de sp_helprotect para desplegar permisos de usuario incluyendo de grupo o rol en Sybase.

```
sp_helprotect student_info, gwashington, NULL, 'teacher_role'
```

Código 7-106. Uso de sp_helprotect para desplegar permisos de usuario incluyendo el rol en Sybase.

Esta sentencia regresa una lista de permisos que han sido otorgados a **`gwashington`** en **`student_info`** con el rol **`teacher_role`**

Funciones acerca de roles de usuario

Hay varias funciones que obtienen información de los roles:

`role_contain( )` checa si un rol esta contenido en otro rol; regresa un estado de 1 si el rol es contenido, 0 si no.

```
role_contain(role1, role2)
```
Código 7-107. Sintaxis de la función role_containt en Sybase.

```
select role_contain ('pay_clerk, 'receptionist')
```
Código 7-108. Uso de la función role_containt en Sybase.

role_name( ) regresa el nombre del rol para un srid[2] dado.

```
role_name(srid)
```
Código 7-109. Sintaxis de la función role_name en Sybase.

role_id( ) regresa un srid para un nombre de rol dado.

```
role_id (role_name)
```
Código 7-110. Sintaxis de la función role_id en Sybase.

proc_role( ) ha sido actualizado para incluir un chequeo en los roles que un login tiene activo.

show_role( ) despliega cuales roles están activados actualmente para su login.

Conducta predeterminada de los roles de usuario

La conducta predeterminada para todos los roles de usuario es de OFF en el login. Los roles del sistema están:

- ON en un login si no tiene un password asociado al rol del sistema
- OFF en el login por default si tiene un password asociado al rol del sistema

Autorización de Proxy

¿Qué es la autorización proxy?

En versiones anteriores, el comando **setuser** permitía a un usuario tomar la identidad de otro usuario. Por lo tanto, estaba limitado a una base de datos y solo podía ser usado por SA o dbo.

Muchos clientes tienen entornos en los cuáles muchos usuarios necesitan compartir una cuenta (identidad) a través de muchas bases de datos, por esta razón, la autorización proxy deja que los usuarios asuman la identidad de otros usuarios en todo el servidor. En este caso, los usuarios con autorización proxy pueden ser conocidos como otro usuario en el servidor y los usuarios deben estar en la base de datos para los cuáles están establecidos actualmente, y deben tomar la identidad de un usuario que también debe existir en la base de datos

Implementando autorización proxy

El SSO debe otorgar permisos a un usuario para usar la sesión de autorización como sigue:

```
grant set session authorization to <login>
grant set proxy to <login>
```
Código 7-111. Sintaxis de gran set para otorgar permisos a un usuario para usar la sesión de autorización en Sybase.

[2] Srid. System Role ID. Identificador del role de sistema.

En este punto, el usuario puede cambiar la sesión de autorización:

```
set session authorization <login_name>
```

Código 7-112. Sintaxis de set session para cambiar la sesión de autorización en Sybase.

Cuando los usuarios quieran regresar a sus logins originales, ellos usan el mismo comando como sigue:

```
set session authorization <original_login>
```

Código 7-113. Sintaxis de set session para regresar a su login original de autorización en Sybase.

Limitaciones de la autorización proxy

El comando **`set session authorization`**/**`set proxy`** no puede ser usado en una sesión activa. No son permitidos múltiples cambios de niveles de identidad. Un usuario no puede ejecutar **`set session authorization`** en un usuario, y desde este punto ejecutar otra autorización proxy en otro usuario. Para cambiar un nuevo login, el usuario debe retornar primero el login original.

El comando **`set session authorization`**/**`set proxy`** no puede ser usado en un login que esté bloqueado. Cuando un procedimiento almacenado es usado, **`set session authorization`**/ **`set proxy`** debe ser ejecutado antes de salirse.

Resumiendo, al desarrollar un lugar con políticas de seguridad, considere lo siguiente:

- Si tiene un "todopoderoso" sa, o usa roles.
- ¿Cuáles usuarios (si hay) tienen privilegios especiales?
- ¿Cuáles procedimientos almacenados y objetos de base de datos necesitan protección?
- Al crear un plan en una jerarquía de roles; recuerde que el acceso puede ser otorgado basado en:
    - Deberes relacionados con el servidor.
    - Creación de bases de datos, y tareas de mantenimiento.
    - Objetos de base de datos y necesidades de los usuarios.
    - Roles que las personas juegan en la organización.
    - Grupos (como departamentos y organizaciones) de las cuáles las personas son parte.

Monitoreando y arreglando problemas

Aprender las bases de monitoreo de log de SQL Server, examinar como se monitorea el uso del espacio, validar/mantener la integridad de la base de datos, herramientas que pueden ser usadas para monitorear la actividad del SQL Server, fijar los límites de los recursos son las tareas que hay que prever en la resolución de problemas en la base de datos.

Monitoreando los Error Logs y Logs del Backup Server del SQL Server

El error log del SQL Server

Usualmente localizado en el directorio **`install`** en el directorio **`sybase`**, la información es agregada cada vez que se inicia SQL Server y también cuando hay un error fatal o error del kernel.

El formato del error log es:

```
date          time          sender message

00:00000:00001:1998/04/07 14:44:30.90 server on top of default character set:
00:00000:00001:1998/04/07 14:44:30.90 server 'ISO_1' (ID = 1).
00:00000:00001 :1998/04/07 14:49:09.75 server DBCC TRACEON 8399, SPID 1
```

Código 7-114. Fragmento del error log en Sybase.

Cuando ocurre un error, Sybase genera un mensaje de error

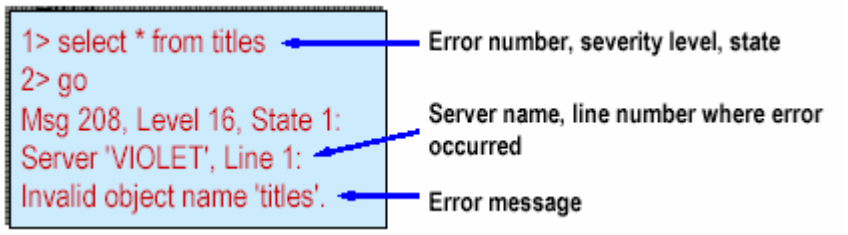


Ilustración 7-58. Descripción de lo que sucede cuando se genera un error en Sybase.

Los mensajes del sistema proporcionados se almacenan en **`master..sysmessages`**, pero cuidado, no se tiene que alterar esta tabla.

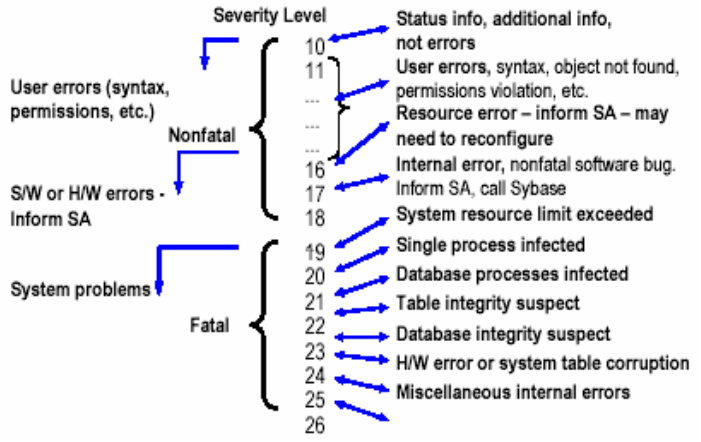


Ilustración 7-59. Severidad del nivel de error en Sybase.

Es sugerible monitorear el error log para encontrar errores de software y hardware de nivel de severidad 17 y superiores. Los usuarios no pueden reportar errores de severidad 17 y 18 si su trabajo no es interrumpido; necesita monitorear el error log para esos errores. Prepare una rutina que navegue por el error log buscando errores números de error específicos o niveles.

Los niveles 10-16 aparecen solo en la pantalla del usuario.

Cuando los mensajes no indican qué hacer acerca del problema, hay que ver el Troubleshooting Guide o llamar al soporte técnico de Sybase

Tips al monitorear el error log

- El error log crece constantemente y necesitar ser limpiado regularmente.
    - Asegurarse de cerrar el servidor primero.
    - Hacer una copia del error log antes de borrar.
- Los usuarios pueden especificar ciertos mensajes que son escritos al error log del SQL Server.
- Los procedimientos **`sp_addmessage`** o **`sp_altermessage`** pueden usarse para especificar algún mensaje definido por el usuario que debe ser escrito al error log.
- Los usuarios pueden especificar que los siguientes eventos sean escritos al error log:
    - Exitosas conexiones al SQL Server
    - Fallidas conexiones al SQL Server

Monitoreando el log del backup server

- Es creado cuando es instalado el **`Backup Server`**
- Por default, en **`$SYBASE/install/backup.log`**
- Contiene información del arranque, errores
- Monitorea frecuentemente, después de que cada base de datos haya sido respaldada

Monitoreando el uso del espacio usando procedimientos almacenados, dbcc, y el manejador threshold

Técnicas para el monitoreo del uso del espacio

Asegure un buen funcionamiento en el entorno de la base de datos, verificando que haya espacio suficiente para el log, objetos de base de datos, database objects, y todo lo crítico. Si no hay espacio en el log, las consultas trabajarán, pero las modificaciones no. Si el espacio en las tablas u otros objetos se termina, entonces si se quiere insertar en estas tablas o crear nuevos objetos no se podrá hacer.

Se puede usar las siguientes herramientas y funciones para monitorear el uso del espacio:

- **`sp_helpdb`**
- **`sp_helpsegment`**
- **`sp_spaceused`**
- **`dbcc checktable`**
- Threshold Manager

sp_helpdb

```
1> sp_helpdb sales
2> go

name      db_size     owner      dbid      created          status
--------- ----------- ---------- --------- ---------------- ----------------
sales     4 MB        sa              5    Oct 16 1992      no options
                                                            set

device_fragments        size        usage           free kbytes
----------------------- ----------- --------------- ----------
                dev1     2 MB  data only       1376
                dev2     1 MB  log only        1008
                dev3     1 MB  data only       1008
```

Ilustración 7-60. Salida de sp_helpdb en Sybase.

sp_helpsegment

```
1> sp_helpsegment seg1
2> go
segment name   status
-------------- ------------
seg1                      0

device              size        free-pages
------------------- --------------------------- -------------------
data_dev3       1.0MB                                       430

table_name      index name                  indid
--------------- --------------------------- -------------------
tableA          x_tableA                        1
```

Ilustración 7-61. Salida de sp_helpsegment en Sybase.

sp_spaceused

sp_spaceused despliega el espacio libre dentro de las tablas o en la base de datos

```
1> sp_spaceused titles
2> go
name   rows  reserved  data        index_size      unused
------ ------ ---------- -------- --------------- ----------
titles     18       48 KB      6 KB       4 KB            36 KB


1> sp_spaceused
2> go
db_name   db_size    reserved    data    index_size     unused
--------- ---------- ----------- ------- -------------- ----------
pubs2     2.0 MB  1386 KB   452     94 KB              840 KB
```

Ilustración 7-62. Salida de sp_spaceused en Sybase.

sp_spaceused usa dos funciones que pueden ejecutarse independientemente:

- **data_pgs(object_id, {doampg | ioampg})** estima páginas usadas por tabla
- **rowcnt(doampg)** estima registros en una tabla

```
select data_pgs(8, doampg) from sysindexes where id = 8
```

Código 7-115. Uso de data_pgs( ) para syslogs en Sybase.

sp_spaceused y esas dos funciones son más exactas si se ejecutan después de **dbcc checktable()**.

dbcc checktable

dbcc checktable (tablename) reporta con más exactitud cuántas páginas de datos están en la tabla especificada. Si el segmento del log está en su propio dispositivo, **dbcc checktable (syslogs)** reporta la cantidad de espacio de log usado y qué porcentaje de log está libre.

```
Checking syslogs
The total number of data pages in this table is 7.
*** NOTICE:  Space used on the log segment is 0.01
Mbytes, 1.37%.
*** NOTICE:  Space free on the log segment is 0.99
Mbytes, 98.63%.
Table has 174 data rows.
```

Ilustración 7-63. Salida de dbcc checktable en Sybase.

dbcc checkcatalog

dbcc checkcatalog checa la consistencia dentro y entre las tablas del sistema encontradas en una base de datos. Verifica que:

- Cada tipo en **syscolumns** tenga un registro coincidente en **systypes**
- Cada tabla y vista en **sysobjects** tenga por lo menos una columna en **syscolumns**
- El último **checkpoint** en **syslogs** sea válido

El manejador threshold

El manejador Threshold notifica cuando los segmentos de log o datos se han llenado, y respalda o borra el log de transacciones. Hay que usar el threshold como última oportunidad (solo segmentos de log). Preparar uno o más thresholds que liberan espacio sobre segmentos de datos o log

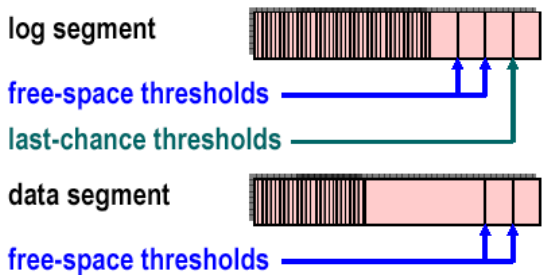


Ilustración 7-64. Esquema del threshold en el segmento de log y de datos en Sybase.

Desplegando información acerca de los thresholds

Se usa **sp_helpthreshold** para ver información acerca de los thresholds.

```
sp_helpthreshold [segment_name]
```

Código 7-116. Sintaxis de sp_helpthreshold en Sybase.

Sin un nombre de segmento, despliega información acerca de todos los thresholds en la base de datos.

```
1> sp_helpthreshold data_seg
2> go
segment name free_pages last chance? threshold procedure
------------ ---------- ------------ ----------------------
data_seg       200            0       data_proc
```

Código 7-117. Salida de sp_helpthreshold.

Abortando o suspendiendo transacciones de usuario

Cuando un threshold de última oportunidad en el log es cruzado, los procesos del usuario que intenten escribir en el log serán suspendidos o abortados. Por defecto son suspendidos.

Hay que usar **sp_dboption** (en **master**) para cambiar:

```
sp_dboption salesdb, "abort tran on log full", true
```

Código 7-118. Uso de sp_dboption para habilitar la escritura en el log de transacciones en Sybase.

Una vez habilitado, entonces hay que ejecutar **checkpoint** en la base de datos especificada. Si el procedimiento threshold vuelca el log de transacciones, hay que preguntarse que es mejor si los procesos del usuario son suspendidos o abortados.

Thresholds y el Log

Los thresholds de última oportunidad previenen desastrosas situaciones al quedarse sin espacio, pero se pueden suspender o abortar los usuarios del proceso. Preparando un procedimiento threshold espacio-libre que respalde el log minimiza la posibilidad de bloquear a los usuarios y respaldar el log de transacciones cuando sea necesario. Si el procedimiento escribe al error log usando sentencias print, entonces también puede ser monitoreado.

Procedimientos de Sistema para el Manejo de Thresholds

En resumen, use los siguientes procedimientos del sistema para el manejo de thresholds:

- **sp_addthreshold** agrega un **threshold**
- **sp_dropthreshold** borra un **threshold**
- **sp_helpthreshold** despliega información acerca de los thresholds
- **sp_dboption** fija la conducta cuando el log se llena
- **sp_helpsegment** despliega información acerca del segmento(s)

Validando /manteniendo la integridad de la base de datos usando dbcc, y dbcc Paralelos

Técnicas de validación de la integridad de la base de datos

SQL Server asegura que las estructuras internas de cada base de datos sean consistentes, pero pueden llegar a ser incosistentes. Se usa **dbcc**[3] como tarea regular para monitorear la integridad de las bases de datos y objetos. Muchos sitios VLDB[4] no están habilitados para realizar chequeos de consistencia, es mejor usar **dbcc** paralelos. Los **dbcc** paralelos han sido introducidos con ASE 11.5 y trabajan usando el comando **dbcc checkstorage**. Toma ventaja de los **prefetch** asíncronos.

Introducción al dbcc paralelo

dbcc checkstorage difiere de otros **dbcc's** porque:

- Ejecuta en paralelo usando múltiples procesos de trabajo.
- El bloqueo de tablas es mínimo durante su proceso.
- Los fallos en la integridad de los Logs en una base de datos son enviados a la salida estándar.
- Las escalas son casi lineales tantos recursos sean agregados.
- Requiere de un manejo dedicado de la base de datos a ejecutar.
- Proporciona procedimientos **dbcc** para la administración y análisis.
- No reporta errores espurios.
- Realiza algunos chequeos que no hacen **checkalloc** o **checkdb.**
- No verifica la consistencia de los índices.

Instalar dbcc checkstorage

Antes de que **dbcc checkstorage** pueda ser usado, SQL Server debe ser preparado para esto. Al instalar **dbcc checkstorage** puede ser dividido en siete pasos:

- Planear recursos con **sp_dbcc_plandb**.
- Instalar la base de datos **dbccdb**.
- Configurar caches para **dbccdb**.
- Agregar segmentos de trabajo.
- Crear espacio de trabajo para **dbccdb**.
- Configurar las bases de datos destino con **sp_dbcc_updateconfig**.
- Evaluar la instalación con **sp_dbcc_evaluatedb**.

---

[3] dbcc. Database Consistency Checker. Verificar la consistencia de la base de datos.

[4] VLDB. Very Large DataBases. Bases de datos muy grandes.

Paso #1 - Planear recursos.

- Los recursos requeridos para **dbcc checkstorage** dependen de:
    - Tamaño de la base(s) de datos que ser(án) verificada(s).
    - Número de bases de datos checadas concurrentemente.
    - Tiempo en el cuál el chequeo debe completarse.
- Entre los recursos que necesitan configurarse están:
    - Procesos Worker.
    - Caches.
    - Dispositivos lógicos.
    - Espacio de trabajo.
- El procedimiento de sistema **sp_plan_dbccdb** proporciona la ayuda para determinar que tan grande debe ser cada recurso.
- El procedimiento del sistema **sp_dbcc_plandb** da información necesaria para preparar la base de datos **dbccdb**.

```
1> sp_plan_dbccdb pubs2
2> go
Recommended size for dbccdb database is 32MB (data = 30MB, log = 2MB).
dbccdb database already exists with size 12MB.
Recommended values for workspace size, cache size and process count are:
dbname                    scan ws    text ws    cache    process count
 pubs2                         64K             48K    640K       1
(return status = 0)
```

Ilustración 7-65. Salida de sp_plan_dbccdb para una base de datos en Sybase.

- **sp_dbcc_plandb** toma el nombre de la base de datos destino como un argumento y regresa los valores recomendados para: espacio de trabajo, número de procesos **worker** y tamaño del cache.

Paso #2 - Instalar la base de datos de manejo de **dbcc**.

- El comando **dbcc checkstorage** requiere de una base de datos llamada **dbccdb**.
- Para un mejor rendimiento, esta base de datos debe estar sobre uno o más dispositivos.
- El tamaño de la base de datos **dbccdb** puede ser determinado por el procedimiento **sp_plan_dbccdb** y el comando **create database** es usado para crearla.
- Una vez creada, el script **installdbccdb** debe ejecutarse para:
    - Crear e inicializar las tablas del sistema **dbcc**.
    - Crear los procedimientos almacenados **dbcc**.
- Finalmente, por lo menos dos espacios de trabajo deben ser creados.
-

Paso #3 - Configurar caches para **dbccdb**.

- Cuando configure el tamaño del cache para **dbccdb**, use el valor recomendado de **sp_plan_dbccdb**.
- Hay que:
    - Configurar el cache con **sp_cacheconfig**.
    - Configurar bancos de buffer para este cache usando **sp_poolconfig**.
    - Vincular el cache a la base de datos **dbccdb** usando **sp_bindcache**.

Paso #4 - Agregar segmentos de espacio de trabajo.

Crear segmentos para buscar espacio de trabajo.

Usar **sp_addsegment** para agregar segmentos a dbccdb.

Paso #5 - Crear espacios de trabajo para **dbccdb**.

Cuando es creada **dbccdb**, dos espacios de trabajo deben ser creados:

- Un **Scan workspace** contiene un registro por cada página de la base de datos destino.
- El texto **workspace** contiene un registro por cada tabla en la base de datos destino que contenga columnas de texto o de imagen.

Cada **workspace** debe tener un nombre único.

```
sp_dbcc_createws dbname, segname, [ wsname], wstype, "wssize[K|M]"
```

Código 7-119. Sintaxis de sp_dbcc para crear un workspace en Sybase.

Paso #6 - Configure las bases de datos destino.

Después de que la base de datos del manejo **dbcc** haya sido completamente instalada e inicializada, debe configurar varios atributos con respecto a cada base de datos destino. Más específicamente, para cada base de datos a ser verificada, el usuario debe indicar:

- Nombre y tamaño del cache a usar.
- Número máximo de procesos **worker** a usar.
- Nombres y tamaños de los **workspaces** a usar.

dbcc checkstorage no checa cualquier base de datos que no tenga esos valores en sus atributos.

Hay que configurar esos atributos con el procedimiento **sp_dbcc_updateconfig**.

Usar **sp_dbcc_updateconfig** para actualizar la tabla **dbcc_config** de la base de datos destino.

Cada parámetro **dbcc** debe ser actualizado separadamente para cada base de datos destino

```
sp_dbcc_updateconfig pubs2, "max worker processes", "2"
sp_dbcc_updateconfig pubs2, "dbcc named cache", pubs2_cache, "10K"
sp_dbcc_updateconfig pubs2, "text workspace", text_pubs2
```

Código 7-120. Uso de sp_dbcc_updateconfig para actualizar la tabla dbcc_config en Sybase.

Paso #7 – Ejecutar **sp_dbcc_evaluatedb**

Si las características de una base de datos de usuario cambian, usar el procedimiento almacenado **sp_dbcc_evaluatedb** para reevaluar la configuración actual de **dbccdb**.

Los cambios posteriores a una base de datos de usuario debe afectar la configuración de **dbccdb**.

Cuando una base de datos de usuario es creada, borrada o alterada, el tamaño de los **workspaces** y del cache deben ser afectados.

Los cambios en el tamaño del cache o los procesos **worker** cuentan para que **dbcc_checkstorage** deba requerir una reconfiguración en el cache del buffer del SQL Server y los procesos **worker**.

Resumiendo los pasos para instalar **dbccdb**. Para crear la base de datos **dbccdb**, hay que completar los siguientes pasos generales:

- Ejecutar **sp_plan_dbccdb** para obtener los valores recomendados para preparar la base de datos **dbccdb**.
- Usar **disk init** para inicializar un dispositivo de disco para **dbccdb**.
- Usar **create database** para crear **dbccdb** en el dispositivo inicializado en el paso 2.
- Correr el script **installdbccdb** encontrado en el directorio **$SYBASE/scripts**.
- Configurar un cache que sea usado para **dbccdb** con **sp_cacheconfig**.
    - Recordar crear un banco del buffer para este cache y vincular el cache con la base de datos **dbccdb**.
- Usar **sp_addsegment** para agregar los segmentos de búsqueda y de texto.
- Usar **sp_dbcc_createws** para crear los **workspaces** de búsqueda y de texto.
- Usar **sp_dbcc_updateconfig** para:
    - Configurar el número de procesos **worker** a ser usados.
    - Inicializar la tabla dbcc_config para la base de datos destino.
- Ejecutar **sp_dbcc_evaluatedb** para verificar la configuración para la base de datos destino.

Flujo de ejecución del **dbcc** paralelo.

Explicación como **dbcc checkstorage** se ejecuta:

1. Inicialización – Verifica la configuración de la base de datos destino y la disponibilidad de los recursos requeridos.

2. Búsqueda en la base de datos – Lee toda la base de datos tan rápido como sea posible, realiza algunos chequeos, y escribe un breve resumen de cada página del **workspace** en **dbccdb**.

3. Los chequeos que quedan son realizados en **dbccdb**.

4. Cuando una inconsistencia es encontrada no puede ser atribuida a la actividad de actualización concurrente, esto es registrado en **dbccdb**.

A continuación se muestra el flujo de ejecución del **dbcc paralelo** presentando un ejemplo usando dos procesos **worker**, donde los **workspaces** residen en la base de datos **dbccdb**.

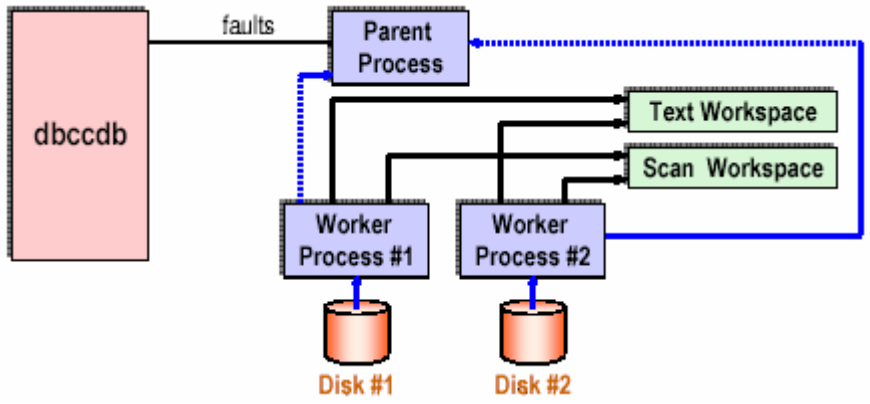


Ilustración 7-66. Flujo de ejecución del dbcc checkstorage en Sybase.

Se examina los resultados del **checkstorage** en una de estas formas:

- Use los procedimientos almacenados **dbcc**.
- Consultando el manejador **dbcc** de la base de datos usando **triggers**.

Procedimientos proporcionados por Sybase:

- **sp_dbcc_summaryreport**: Despliega un resumen de datos como el número de fallas reportadas.
- **sp_dbcc_faultreport**: Despliega información detallada de cada falla reportada.
- **sp_dbcc_fullreport**: Despliega la configuración, estadísticas, o datos del fallo para una base de datos u objeto.

```
1> sp_dbcc_faultreport short jakedb
2> go
```

Database Name: jakedb

Table Name	Index	Type Code	Description	Page Number
mytab	0	100002	page free offset err	313
mytab	0	100022	chain start error	313
sales	2	100017	OAM ring error	376
sales	2	100031	page not allocated	376
sales	2	100022	chain start error	377
sales	2	100031	page not allocated	377

Ilustración 7-67. Ejemplo de salida de sp_dbcc_faultreport.

Con la información del **sp_dbcc_faultreport** puede ser usada para determinar:

- Objetos específicos **indid's** con fallas.
- Número de páginas involucradas.
- Breve descripción de las fallas detectadas

Monitoreando la actividad del SQL Server usando Sybase Central.

Para que **Sybase Central** monitoree su entorno, deberá realizar los siguientes pasos:

- Usar **dsedit** para configurar el archivo de interfaces para incluir los nombres de los servidores a ser monitoreados.
- Agregar el plug-in usando la opción **Sybase Central Plug-ins**.
- Conectarse a SQL Server a través de **Sybase Central**.
- Crear un perfil de conexión

Usando **Sybase Central**, se pueden realizar las siguientes tareas administrativas:

- Conectarse y desconectarse del SQL Server.
- Desplegar información acerca del SQL Server y los objetos que controla.
- Fijar los parámetros de configuración del SQL Server.
- Desplegar procesos de usuario y eliminar procesos.
- Manejar servidores remotos.

Manejando Procesos del Sybase SQL Server. Se puede usar **Sybase Central** para:

- Ver el estado de los procesos del usuario en el SQL Server.
    - Para desplegar procesos de usuario en el SQL Server, seleccione el fólder de procesos (cada proceso tiene un icono).
    - La etiqueta parámetros de la hoja de procesos lista información adicional acerca de los procesos.
- Fijar atributos de ejecución de los procesos.
    - Cambiar la prioridad de un proceso asignándole a un motor de grupo.
    - La etiqueta de atributos de ejecución de la hoja de procesos deja cambiar los atributos a su valor predeterminado
- Eliminar procesos

Manejando Servidores Remotos

Para tener acceso a un servidor remoto, un SSO puede crear un servidor remoto y entonces definirlo en el SQL Server local. La instalación del SQL Server puede ser configurada para que un usuario conectado al SQL Server pueda requerir la ejecución de un procedimiento almacenado en otro SQL Server. El resultado de esta llamada de un procedimiento remoto (Remote Procedure Call – RPC) es regresada al proceso que llamó ejecutándose en el SQL Server en el cuál el usuario está conectado. Si **Component Integration Services (CIS)** es habilitado para el SQL Server, también se pueden acceder a datos de un servidor remoto como si fuese un servidor local.

Un manejador de sitio, el cuál es un método predeterminado para manejar la interacción entre servidores locales y remotos, crea una conexión física entre el servidor local y el servidor remoto. Entonces se crea una conexión lógica por cada RPC al servidor remoto. El manejador CIS RPC siempre es usado para conexiones involucrando tablas Proxy. Se crean conexiones usando funciones Client-Library. Su pueden habilitar para usarse con todos los RPC's.

Monitores del SQL Server.

El Monitor del SQL Server consiste en cuatro componentes que generan o despliegan datos de rendimiento:

- **Servidor Monitor**: Colecciona datos de rendimiento del SQL Server en tiempo real y lo hace disponible a otros componentes monitores del SQL Server
- **Servidor Histórico**: Obtiene datos de rendimiento del SQL Server del servidor monitor y lo guarda en archivos para su análisis posterior.
- **Monitores en el Plug-in SQL Server para Sybase Central (Visor de monitor)**: Obtiene datos de rendimiento del SQL Server del servidor monitor y despliega los datos en tiempo real en tablas y en gráficos.
- **Monitor Client Library**: Da apoyo a los usuarios quienes están en desarrollo de aplicaciones de monitoreo application programming interface (API).

Usando Monitores SQL Server con **`Sybase Central`**

Un servidor monitor debe estar configurado en la misma máquina como el SQL Server se quiera monitorear. El servidor monitor debe ejecutarse. Un registro del servidor monitor configurado debe existir en el sql.ini o archivo de servicios de directorio en la máquina donde Sybase Central está corriendo. La versión correcta de una máquina virtual Java debe estar instalada en su máquina. En el árbol de **`Sybase Central`**, un fólder de monitores debe aparecer bajo el icono ASE. El login del SQL Server debe tener privilegios apropiados.

Resultados de los Datos de el Monitoreo

Para coleccionar el tipo de datos que necesita, se puede fijar lo siguiente:

- El tipo estadístico: muestra versus datos acumulativos.
- El intervalo de muestra en horas, minutos, y segundos.
- Pausando y refrescando datos en la ventana para examinar los resultados.
- Fijar filtros pata limitar los datos a monitorear.

Tipo y nivel de detalle	Nombre del monitor	Descripción
Server-wide performance overviews	Performance Summary Monitor	Despliega la tasa de impacto, de los dispositivos de E/S, de las páginas de E/S y un bloque de estadísticas.
	Performance Trends Monitor	Parcela de valores hasta de intervalos de 60 para la estadística seleccionada.
Server-wide activity summaries	Application Activity Monitor	Despliega en resumen el uso del CPU, de las páginas de E/S, y peticiones de bloqueo de aplicaciones.
	Engine Activity Monitor	Despliega el uso del CPU y del sistema operativo y la tasa de rendimiento por tarea.
	Network Activity Monitor	Despliega estadísticas acerca de los paquetes y bytes enviados y recibidos por el ASE.
	Process Activity Monitor	Despliega en forma resumida el uso del CPU, de las páginas de E/S y las peticiones de bloqueo de por los procesos.
	Stored Procedure Activity Monitor	Despliega el resumen de las páginas de E/S, las peticiones de bloque para los procedimientos almacenados y triggers.
	Transaction Activity Monitor	Despliega métrica de updates, deletes, inserts y selects. También muestra métrica segmentados.
Server-wide detail: cache management	Cache Monitor	Despliega páginas E/S y eficiencia para el cache de datos y estadísticas de ejecución para el cache de procedimientos. Muestra el tamaño del cache de datos y del cache de procedimientos.
	Data Cache Monitor	Despliega la actividad y eficiencia de los 10 caches de datos más activos.
	Memory Utilization Monitor	Despliega las particiones de memoria asignadas al ASE en formato de gráfica de pay.
Server-wide detail: I/O management	Device I/O Monitor	Despliega la actividad física de E/S y la tasa de éxito por dispositivo.
	Object Page I/O Monitor	Despliega estadísticas de las páginas físicas y lógicas de E/S de todas las tablas incluyendo las de sistema y temporales.
Server-wide detail: lock management	Object Lock Status Monitor	Despliega detalles de tablas detenidas o bloqueadas.
Process detail	Process current SQL Statement Monitor	Despliega el estado del SQL actualmente en ejecución y el plan de la consulta para el proceso seleccionado.

Tabla 7-2. Elementos del Sybase Central para monitoreo.

Desarrollando una Estrategia de Monitoreo

La integridad de datos es más importante:

- Al adoptar buenos hábitos de respaldo.
- Ejecutar **dbcc** y **dbcc paralelos** frecuentemente.

Horario de monitoreo dependiendo de situaciones y prioridades:

- Tamaño de la base de datos.
- Tipos de consulta.

- Actividad.
- Importancia de tiempo continúo.
- Consideraciones de recuperación.
- Número de usuarios.

Automatizar usando archivos del sistema para monitorear y verificar la salida diariamente:

- Los problemas iniciales son detectados, haciendo más fácil su reparación.

Reglas de monitoreo:

- Monitorear el error log varias veces al día.
- Monitorear la integridad de la base de datos antes de los respaldos, y basadas en la actividad.
- Con mucha actividad, monitorear frecuentemente.
- Identificar periodos de baja actividad para ejecutar **dbcc**.
- Verificar el respaldo del log después de los respaldos.
- Monitorear el uso del espacio basado en la actividad.
- Monitorear toda la actividad del servidor como sea requerido.

Resumiendo **Sybase Central**:

- Nombrar un producto que pueda ser monitoreado con **Sybase Central** usando un módulo plug-in.
- Listar los cuatro pasos básicos requeridos para usar **Sybase Central** para monitorear el entorno SQL Server.
- ¿Cuáles son las opciones de configuración del SQL Server que un SSO puede predeterminar?
- ¿Qué tareas pueden realizarse contra un servidor remoto a través de **Sybase Central**?
- ¿Cuáles son los cuatro componentes de monitoreo del SQL Server que generan o despliegan datos de rendimiento?
- Verificar el nombre del monitor y la descripción en la siguiente tabla:

Monitor Name	Description
Performance Summary Monitor	Despliega la tasa de impacto, de los dispositivos de E/S, de las páginas de E/S y un bloque de estadísticas.
Application Activity Monitor	Despliega en resumen el uso del CPU, de las páginas de E/S, y peticiones de bloqueo de aplicaciones.
Network Activity Monitor	Despliega estadísticas acerca de los paquetes y bytes enviados y recibidos por el ASE.
Transaction Activity Monitor	Despliega métrica de updates, deletes, inserts y selects. También muestra métrica segmentados.
Cache Monitor	Despliega páginas E/S y eficiencia para el cache de datos y estadísticas de ejecución para el cache de procedimientos. Muestra el tamaño del cache de datos y del cache de procedimientos.
Object Lock Status Monitor	Despliega detalles de tablas detenidas o bloqueadas.
Process current SQL Statement Monitor	Despliega el estado del SQL actualmente en ejecución y el plan de la consulta para el proceso seleccionado.

Tabla 7-3. Descripción de los monitores del Sybase Central.

System error log	dbcc checkdb	OS monitoring programs	dbcc checktable
ASE error log	dbcc checkcatalog	Backup Server error log	dbcc checkalloc
Threshold Manager	dbcc tablealloc	sp_helpdb	dbcc indexalloc
sp_helpsegment	dbcc memusage	sp_spaceused	Select data_pgs()
select rowcnt()	sp_monitor	sp_sysmon	Sybase Central

Tabla 7-4. Herramientas de monitoreo de Sybase.

Fijando los Límites de Recursos con el Administrador de Recursos

¿Porqué fijar límites en los recursos?

El cliente **Runaway** requiere tanto consultas aleatorias ad-hoc o consultas de aplicaciones erróneas que pueden tener un efecto adverso en el funcionamiento de todo el servidor:

- Puede ser verdadero siempre que la mayoría de accesos de datos sean controlados por una aplicación.
- Las demandas pueden requerir un alto y anormal número de recursos que afectan negativamente el rendimiento del servidor.

Descripción del administrador de recursos:

El administrador de recursos deja al administrador del sistema:

- Fijar límites en varios recursos de SQL Server.
- Definir que acción deberá ser tomada cuando un límite es violado.
- Definir cuáles veces del día o semana los límites de recursos estarán en efecto.

Los límites de recursos pueden estar basados en logins individuales o aplicaciones. El uso de esos límites permite al administrador del sistema prevenir consultas con fallas para reducir el funcionamiento del SQL Server.

Límites de Recursos y Acciones:

Tipos de límites. Se pueden definir límites para los siguientes recursos:

- Número de I/O (estimado y actual).
- Tiempo de ejecución colapsado (de un lote o transacción).
- Número de registros.

Tipos de acción. Posibles acciones cuando un límite es violado:

- Desplegar un mensaje de advertencia y continuar el proceso.
- Abortar la consulta por lote.
- Abortar la transacción.
- Desconectar la sesión del usuario.

Creando Límites de Recursos.

Es recomendable usar estos pasos para crear y usar los límites de recursos:

- Activar los límites de recursos. Con **sp_configure** y el parámetro **allow resource limits**.
- Crear un rango de tiempo. Con el nuevo procedimiento de sistema **sp_add_time_range**.
- Crear un límite de recurso. Con el nuevo procedimiento de sistema **sp_add_resource_limit**.

Cuando se crean límites de recursos contra aplicaciones de los usuarios, use las llamadas **Open Client** apropiadas para nombrar la aplicación.

sp_add_resource_limit

```
sp_add_resource_limit name, appname, rangename, limittype, limit_value,
enforced, action, scope
```

Código 7-121. Sintaxis de sp_add_resource_limit en Sybase.

```
#Acceder con el login UserN (su nombre de usuario)
#Agregar un límite de recurso en TomN (donde TomN es un usuario creado
anteriormente)
1> sp_add_resource_limit Tom42, NULL, "at all times",
2> row_count, 100, 2, 2, 1
3> go
#Salga de la sesión y vuelva a conectarse como TomN
#Ejecutar:
select * from pub2..intro1
#Examine el error log con “more <logname> | tail -20”
```

Código 7-122. Uso de sp_add_resource_limit en Sybase.

Respaldando bases de datos y logs de transacciones

El respaldo de una base de datos y del log de transacciones, el uso del manejador **threshold** para preparar una copia de seguridad automática del log de transacciones, el restaurar una base de datos a partir de información previamente guardada, desarrollo de una estrategia para guardar regularmente los datos en un lugar seguro, son las actividades sobre resguardo de la información que tiene que seguirse en Sybase.

Respaldo de bases de datos

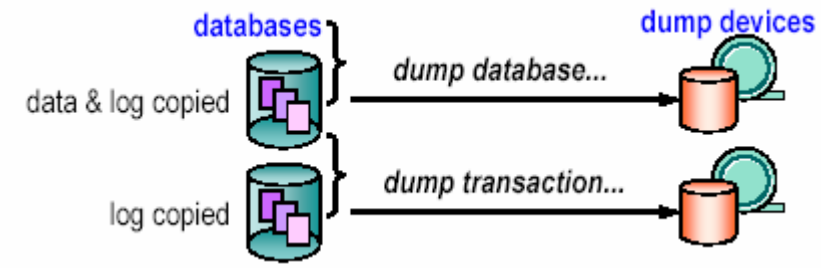


Ilustración 7-68. Esquema de respaldo en Sybase.

- **dump database** hace respaldos físicos de toda una base de datos, tanto datos, como el log de transacciones.
- **dump transaction** hace respaldos físicos del log de transacciones solamente.

No se usa comandos de copiado del sistema operativo.

Si una falla ocurre, se vuelven a crear las bases de datos cargando los respaldos en el orden en que fueron creados

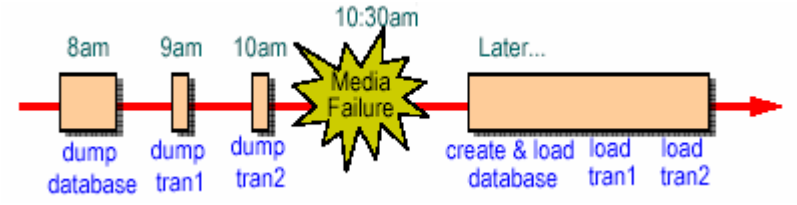


Ilustración 7-69. Proceso de respaldo y recuperación de una base de datos en Sybase.

Si el log está en un dispositivo diferente, la base de datos puede ser recuperada hasta el tiempo en fue la falla (10:30). Por otra parte, la base de datos puede ser recuperada hasta el último respaldo del log de transacciones (10 a.m.).

Todos los respaldos del SQL Server son realizados por el servidor de respaldos. Un programa servidor ejecutándose en la misma máquina como un SQL Server.

Características:

- Dinámico — Pueden realizarse mientras los usuarios están activos.
- Puede respaldar dispositivos locales o dispositivos de otras máquinas.
- Puede realizar respaldos multiarchivos (muchas bases de datos en un solo dispositivo).
- Puede realizar respaldos multivolumen (grandes bases de datos en varios dispositivos).
- Puede preparar respaldos automáticos.

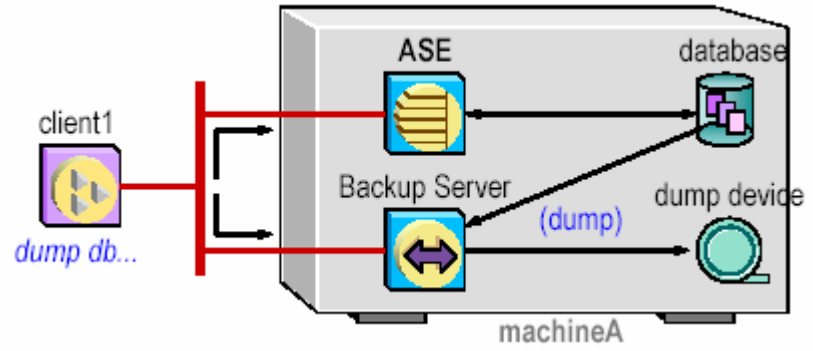


Ilustración 7-70. Esquema de uso del Backup Server en respaldo de bases de datos en Sybase.

Todos los respaldos del SQL Server son realizados por el **`Backup Server`**. La arquitectura de respaldo usa el paradigma cliente/servidor, con el SQL Server como cliente de un **`Backup Server`**.

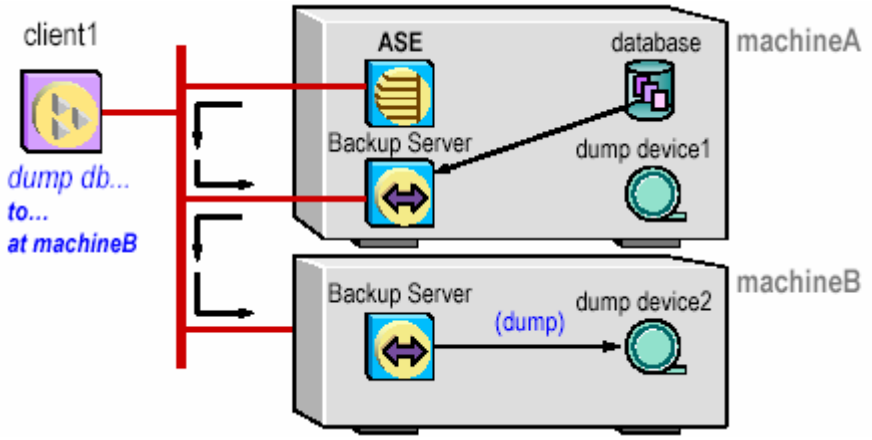


Ilustración 7-71. Esquema de uso del Backup Server en respaldo de bases de datos en un servidor remoto en Sybase.

Los respaldos remotos son posibles. El **Backup Server** local se comunica con un **Backup Server** de una máquina remota.

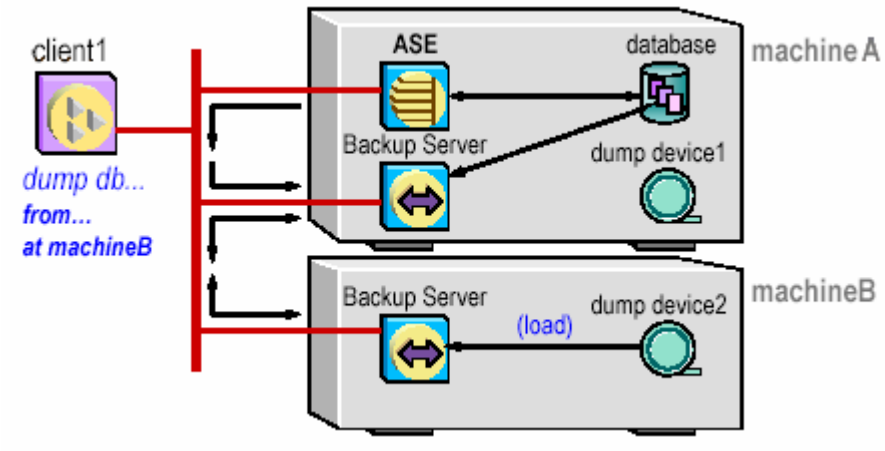


Ilustración 7-72. Esquema de uso del Backup Server en cargas remotas en un servidor remoto en Sybase.

Las cargas remotas son posibles. El **Backup Server** local se comunica con un **Backup Server** de una máquina remota.

Servidores de Respaldo ¿Locales o Remotos?

- Un **Backup Server** local siempre es necesario. El **Backup Server** local puede manejar el dispositivo de respaldo si este dispositivo es conocido por la máquina local.
- Si muchos SQL Servers están en muchas máquinas, se tiene que usar un **Backup Server** remoto sobre una red. ¿Por qué? La máquina puede tener operadores de tiempo completo o muchos dispositivos de respaldo.
- Cada SQL Server tiene que conectarse a un **Backup Server** local para poder usar un **Backup Server** remoto.

- Si las plataformas locales y remotas son homogéneas, los volúmenes escritos local y remotamente son intercambiables.

Identificando un **Backup Server** Local

SQL Server usa el nombre **SYB_BACKUP** para referirse al **Backup Server** local y, por defecto, se ve este nombre en el archivo de interfaces siempre que se hagan respaldos o cargas. Si se quiere que SQL Server use otro nombre en el **Backup Server**, se especifica el nombre durante la instalación. O, agregándolo en el archivo de interfaces y se usa **sp_addserver** para hacérselo conocer al SQL Server.

```
sp_addserver SYB_BACKUP, null, B_VIOLET
#donde B_VIOLET es el nombre en el archivo interfaces
```

Código 7-123. Uso de sp_addserver para que se use otro nombre del Backup Server en Sybase.

Hay que asegurarse de configurar SQL Server para acceso remoto.

Identificando un **Backup Server** Remoto

Para identificar un **Backup Server** remoto, se agrega en el archivo de interfaces. Cuando el nombre de un **Backup Server** remoto es referido en los comandos **dump**/**load**, el **Backup Server** local lo busca en el archivo de interfaces. Hay que recordar: El registro en el interfaces debe ser exacto y formateado correctamente.

Verificando el **Setup**

- Arrancar el **Backup Server** usando desde el comando: **startserver -f RUN_B_VIOLET.**
- Configure SQL Server para acceso remoto (si es necesario, reiniciar SQL Server).
- Verificar que:
  - El registro del **Backup Server** local en **sysservers** tenga un nombre de red exacto usando la instrucción **sp_helpserver**.
  - El usuario que inició el proceso del **Backup Server** (usualmente sybase) tenga permisos de escritura para los dispositivos de respaldo.

Verificando el **Setup: interfaces**

- Tanto el **Backup Servers** local y remoto debe ser listado correctamente en el archivo interfaces.

```
#
B_VIOLET
query tcp sun-ether violet 2001
master tcp sun-ether violet 2001
#
B_SHANTI
query tcp sun-ether shanti 9996
master tcp sun-ether shanti 9996
```

Código 7-124. Archivo de interfaces mostrando tanto el Backup Server local y remoto en Sybase.

- El registro **Backup Server** local es creado cuando SQL Server es instalado.
- El **Backup Server** remoto se agrega manualmente (si hay).

- Si los registros son inexactos, SQL Server puede contactar un **Backup Server** erróneo y escribir o leer de o en dispositivos erróneos.

Seleccionando un Dispositivo de Respaldo

Se prefieren archivos del disco porque:

- Son más rápidos para el respaldo como para la recuperación que las cintas.
- Llegan a ser más baratos para grandes cantidades de datos.
- Desventaja, los respaldos se pierden si el disco falla.

Las cintas pueden ser usadas:

- Ventaja, Son más seguras y más portátiles que los discos.
- Permiten que una biblioteca de base de datos y respaldos del log de transacciones estar fuera de línea.

Agregando un dispositivo de respaldo (local solamente)

En muchas plataformas, SQL Server instala automáticamente en **sysdevices** uno o dos alías para los dispositivos de cinta como **tapedump1/dev/nrmt4** ó **tapedump2/dev/n rst0**.

Para agregar más, se usa **sp_addumpdevice**

```
sp_addumpdevice "tape | disk", "logicalname", "physicalname"
```

Código 7-125. Sintaxis de sp_addumpdevice para agregar un dispositivo en Sybase.

Agrega un registro en **sysdevices**. Puede ser usado en subsecuentes comandos **dump** o **load**.

```
sp_addumpdevice "tape", "tape3", "/dev/nrmt4" ,300
sp_addumpdevice "disk", "disk1", "/usr/backups/disk.dump
```

Código 7-126. Uso de sp_addumpdevice para agregar dispositivos de cinta y disco en Sybase.

dump database. Hace una copia de respaldo de la base de datos y del log de transacciones. Es dinámico, puede ser hecho mientras los usuarios están activos.

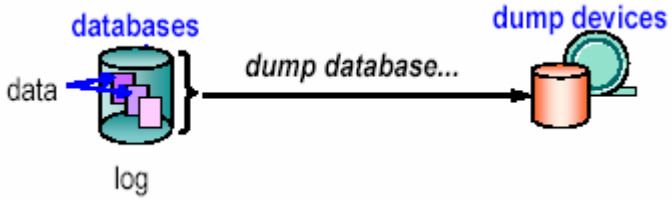


Ilustración 7-73. Esquema de dump database para hacer respaldo de los datos de una base de datos en Sybase.

```
dump database salesdb to data_dev1
```

Código 7-127. Uso de dump database para respaldar en un dispositivo una base de datos en Sybase.

¿Qué hace un dump database?:

- Realiza un **`checkpoint`**, esto es, los datos y el log son copiados del cache al disco (sólo páginas sucias).
- Copia páginas asignadas (log y datos) para respaldar al dispositivo.
- Captura el estado cercano al final del respaldo.

```
dump database database_name
to stripe_device [at b_server_name]
[stripe on stripe_device [at b_server_name] ...
[with {blocksize = number_bytes,
       capacity = number_kilobytes,
       dumpvolume = volume_name,
       file = file_name,
       [nounload | unload],
       retaindays = number_days,
       [noinit | init],
       notify = client | operator_console}
}]]
```

Código 7-128. Sintaxis de dump database en Sybase.

Respaldo multiarchivo (sólo en cinta)

Usando las opciones **`noinit | init y nounload | unload`** para respaldar varias bases de datos en una cinta.

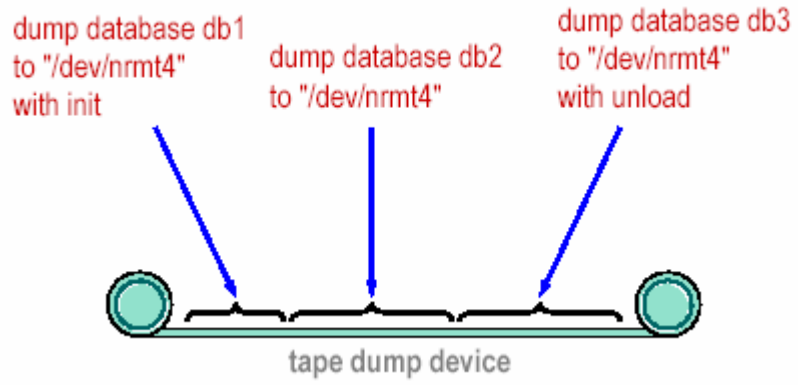


Ilustración 7-74. Esquema respaldo de varias bases de datos en una cinta.

Respaldo a múltiples dispositivos

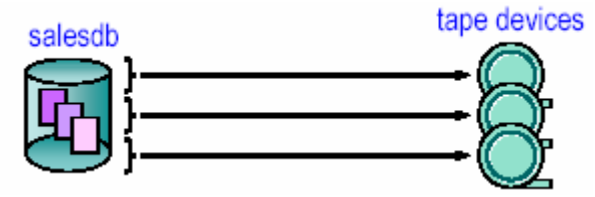


Ilustración 7-75. Esquema de respaldo de una base de datos en múltiples dispositivos.

Se puede escribir una base de datos a través de múltiples dispositivos concurrentemente usando la opción **`stripe on`**.

```
dump database db1 to "/dev/nrmt4"
stripe on "/dev/nrmt5"
stripe on "/dev/nrmt0"
```

Ilustración 7-76. Uso de stripe on al respaldar una base de datos en múltiples dispositivos.

Aproximadamente una tercera parte de la base de datos se va a cada dispositivo.

Respaldos a multivolumen o multidispositivo.

Respaldo a múltiples volúmenes (cinta solamente).

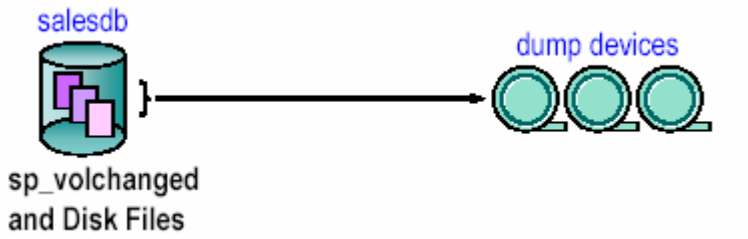


Ilustración 7-77. Uso de sp_volchanged para el respaldo de una base de datos a múltiples volúmenes en Sybase.

En el caso de archivos de disco, los volúmenes no pueden cambiarse; sin embargo, **`sp_volchanged`** puede ejecutarse. Los Operadores usan **`sp_volchanged`** para notificar al **`Backup Server`** que volumen ha sido montado.

`Backup Server` Local y Remoto

Todas las demandas del **`Backup Server`** van al **`Backup Server`** local. Para pasar la demanda del respaldo a un **`Backup Server`** remoto, hay que especificar el nombre en el comando **`dump database`**.

```
dump database salesdb to "/dev/rmt4" at B_SHANTI
stripe on "/dev/rmt5" at B_SHANTI
```

Código 7-129. Uso de dump database para hacer el respaldo en un Backup Server remoto en Sybase.

En muchas plataformas, se usa el nombre del camino físico para un **`Backup Server`** remoto; los nombres del dispositivo lógico de SYBASE no trabajarán. Hay que asegurarse que los registros del archivo de interfaces sean exactos para ambos **`Backup Servers`**.

Respaldos de Base de Datos Manuales y Automáticos

Hay que ejecutar el comando **`dump database`** manualmente regularmente o cuando sea necesario. Puede hacerse que los respaldos se realicen automáticamente usando utilidades del Sistema Operativo que pueden ejecutar los scripts de respaldo o el Manejador **`Threshold`**.

Ventajas:

- Útil en pequeños lugares, o cuando es importante saber quién realiza los respaldos.
- Fácil manejo de cintas.
- Útil cuando se necesita saber el nombre del archivo al cargar un volumen multiarchivo.

Desventajas:

- El personal debe estar disponible cuando se realicen los respaldos.
- El uso de espacio debe estarse monitoreando muy de cerca, o el espacio en la base de datos o en el log puede ejecutarse fuera.

Respaldos de Base de Datos Automáticos

Las utilidades del sistema operativo pueden ejecutar scripts **`dump database`** ya sea en cierto tiempo o en intervalos. Los sistemas operativos UNIX y NT tienen utilidades de horario cron que preparan respaldos regulares en cierto horario.

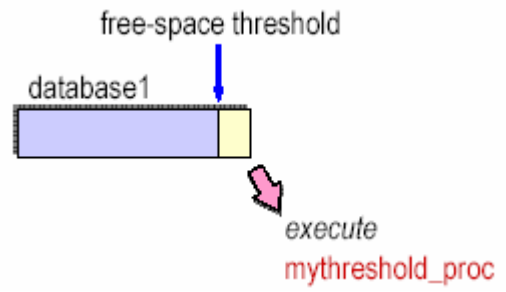


Ilustración 7-78. Esquema de uso de mythreshold_proc para respaldo de bases de datos.

```
dump database pubs2 to "/dev/nrmt0"
dump database pubs2 to "/dev/nrmt0" with init
dump database pubs2 to "MTA1:", stripe on "MTA2:"
dump database pubs2 to "MTA0:" at B_SHANTI
dump database pubs2 to dump_dev
```

Código 7-130. Ejemplos del uso de dump database.

Asegurando la consistencia de la Base de Datos

Aunque no es estrictamente necesario, aquí a continuación se presenta una breve guía para asegurar la consistencia de la base de datos antes de respaldar la base de datos:

```
dbcc checkstorage, dbcc checkcatalog
dbcc checkdb, dbcc checkalloc, dbcc checkcatalog
```

Código 7-131. Uso de dbcc cheksotorage y checkdbe para asegurar la consistencia de una base de datos en Sybase.

Ejecute en el modo **`single-user`**; por otra parte, la salida puede indicar las inconsistencias donde no haya ninguna. No es posible ejecutar estos comandos en modo activo, no se puede ejecutar **`dbcc checktable tablename`** y **`dbcc tablealloc`** para tablas activas. **`dbcc checkstorage`** deberá específicamente prepararse en el SQL Server antes de que pueda ser usada.

```
load database salesdb from data_dev1
```

Código 7-132. Uso de load database para recuperar la información desde un dispositivo en Sybase.

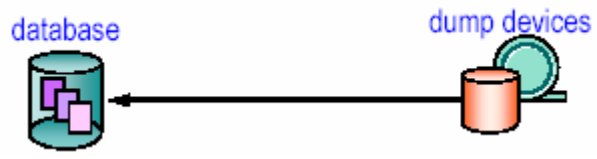


Ilustración 7-79. Esquema de uso de load database para recuperar una base de datos en Sybase.

Inicializa páginas no usadas en la base de datos y reemplaza el contenido existente con los datos de la imagen respaldada. El dueño de la base de datos cargada es el dueño de la base de datos al tiempo de respaldo. Ejecuta recuperaciones sobre alguna transacción sin **commit** en el tiempo en que el respaldo se canceló.

```
load database database_name
from stripe_device [ at b_server_name]
[, stripe on stripe_device
                [at b_server_name]
        [with {[dismount | nodismount],
                [nounload | unload] }]
```

Código 7-133. Sintaxis de load database en Sybase.

```
load database pubs2
from "data_dev1"
load database pubs2
from "MTA0:"
load database pubs2
from "/dev/rmt4" at B_SHANTI,
stripe on "/dev/rmt5" at B_SHANTI
```

Código 7-134. Uso de load database en Sybase.

Cuándo es usada la compatibilidad de Respaldo

Cuando la instalación del SQL Server es actualizada a una nueva versión, todas las bases de datos asociadas con ese servidor son actualizadas automáticamente. Como resultado, los respaldos de una base de datos y del log de transacciones creados con algún SQL Server anterior se deberán actualizarse antes de que puedan usarse con Adaptive Server release 11.5.x. Adaptive Server release 11.5.x proporciona un mecanismo automático de actualización. Se actualizan automáticamente versiones del SQL Server 10 y recientemente la 11.0.

Cuándo es usada la compatibilidad de Respaldo

Para actualizar sólo una base de datos de un servidor completo: Construir un nuevo servidor 11 .x. Respaldar las bases de datos 10.x que se quieran actualizar. Cargar el respaldo en el nuevo servidor 11.0. No se pueden cargar bases de datos 11.0 en un sistema 10.x.

Estado fuera de línea de la base de datos.

Al usar el comando **load database** automáticamente la base de datos está fuera de línea. Elimina la necesidad del administrador de marcar las bases de datos dbo se usa solamente cuando se carga una base de datos y el log de transacciones.

Nueva secuencia de carga:

```
load database database_name from
<pathname_where_dump_is_stored>
load transaction database_name from
<pathname_where_dump_is_stored>
online database database_name
```

Código 7-135. Sintaxis para la secuencia de carga de una base de datos en Sybase.

No hay comando que ponga fuera de línea una base de datos manualmente.

Comando **Online Database**

Ejecutando **online database** hace una base de datos disponible para otros usuarios.

```
online database database_name
```

Código 7-136. Sintaxis de online database en Sybase.

sp_helpdb database_name despliega si la base de datos tiene un estado **online** u **offline**.

online database debe ejecutarse desde la base de datos **master**.

Cuando una base de datos es marcada como **online** u **offline**, el servidor emite un mensaje indicando el estado de la base de datos al **error log**. Los dueños de base de datos (dbo's) o usuarios con los roles **oper_role** o **sa_role** puede ejecutar este comando.

Notas sobre la carga de una base de datos

La base de datos no debe estar actualmente en uso. Se pueden cargar sólo respaldos hechos en el mismo tipo de máquina. La base de datos que se va a cargar debe existir, y debe ser tan grande como la base de datos respaldada. Se usa **sp_helpdb** para ver la cantidad de espacio asignado a una base de datos. Al cargar datos a una base de datos existente se sobrescriben esos datos; las cargas parciales no son posibles; para crear una nueva base de datos para carga, se usa la opción **for load** en **create database**. Si la base de datos es "sospechosa" (estado 256 en **sysdatabases**), hay que borrarla, y volver a crearla cargando los datos del respaldo.

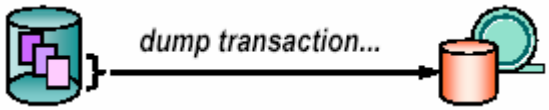


Ilustración 7-80. Esquema de uso de dump transaction en Sybase.

Hace una copia respaldo del log de transacciones de una base de datos

```
dump transaction salesdb to saleslog_dev
```

Código 7-137. Uso de dump transaction en Sybase.

Respaldos incrementales, usados después de **dump database**, también el log. **dump database** respalda pero no trunca el log de transacciones.

```
dump transaction database_name
to stripe_device [ at b_server_name]
...
[, stripe on stripe_device [at b_server_name]
...
[with { dumpvolume = volume_name,
       [dismount | nodismount],
       [nounload | unload],
       ...
       [truncate_only | no_log |
       no_truncate] } ]
```

Código 7-138. Sintaxis de dump transaction en Sybase.

Las mismas opciones de **dump database**, más opciones de truncado.

Respalda todo el log y borra porciones inactivas del log (transacciones con commit).

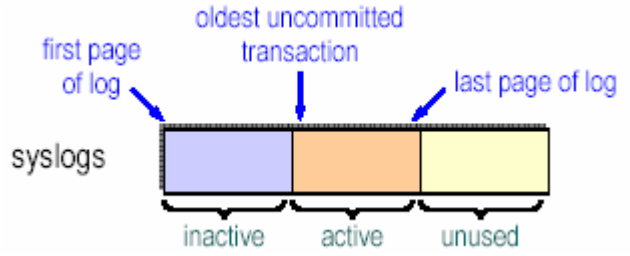


Ilustración 7-81. Esquema del log de transacciones en Sybase.

Opciones de truncado:

- **`with truncate_only`** borra el log sin respaldar.
- **`with no_log`** borra el log sin respaldar, y no registra la transacción.

¿Qué sucede cuando el log se llena? El log de transacciones es añadido cada vez que se modifica la base de datos; y puede llegar a llenarse rápidamente. Si el log se llena, o el threshold (o umbral) del log es cruzado y no trunca el log, todos los trabajos de modificación se detienen. El log necesita ser truncado para que esto no suceda.

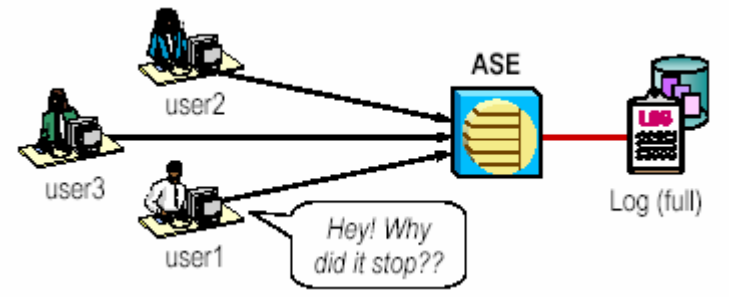


Ilustración 7-82. ¿Qué sucede cuando el log se llena en Sybase?

Existen varias formas de truncar el log:

- **`dump transaction database_name to stripe_device...`**
- **`dump transaction .. with no_log`**
- **`dump transaction .. with truncate_only`**

Si los respaldos del log de transacción no son guardados, haga que log sea truncado cada vez que ocurre el proceso **`checkpoint-checking`** (aproximadamente una vez por minuto).

```
sp_dboption dbname,"trunc. log on chkpt.",true
```

Código 7-139. Uso de sp_dboption para truncar el log cada que ocurra el proceso checkpoint-checking en Sybase.

Si se pone esta opción, hay que respaldar frecuentemente la base de datos, porque el log no esta siendo guardado.

Transacciones que llenan el Log. Las siguientes transacciones llenan el log:

- Actualizando cada registro en una tabla grande.
- Borrando una tabla grande.
- Inserciones basadas en una subconsulta.
- Realizar **`Bulk copy`** en tablas grandes que tienen índices.

Monitoreando el log de transacciones. Se monitoree el log usando las siguientes herramientas:

```
sp_spaceused syslogs
select data_pgs (8, doampg) from sysindexes where id = 8
dbcc checktable (syslogs)
```

Código 7-140. Monitoreo del log de transacciones en Sybase.

Se usa el manejador threshold manager para: notificar cuando se llena el log y también hace respaldos automáticos después de que cierto threshold ha sido cruzado. Hay que verificar transacciones que mantienen el log truncándolo cuando es necesario.

Thresholds de última oportunidad. Si un log esta sobre su propio segmento, SQL Server automáticamente crea un threshold de última oportunidad que monitorea el uso del espacio en el segmento. SQL Server lo pone, y deja que se registre el respaldo del log.

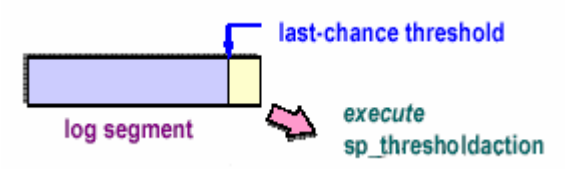


Ilustración 7-83. Esquema de uso del threshold de última oportunidad.

Cuando el **`log segment`** de SQL Server es cruzado, SQL Server aborta o suspende las transacciones de usuario que intentan escribir en el log. Envía un mensaje de error al log por cada transacción suspendida. Ejecuta **`sp_thresholdaction`**. SQL Server no suministra **`sp_thresholdaction`**; lo tiene que escribir.

```
create procedure sp_thresholdaction
        @dbname varchar(30),
        @segmentname varchar(30),
        @space_left int,
        @status int
as
dump transaction @dbname to "/dev/rmt4"
        print "LOG DUMP: '%1!' for '%2!' dumped",
        @segmentname @dbname
```

Código 7-141. Un procedimiento threshold que respalda el log de transacciones e imprime un mensaje en el error log en Sybase.

No hay que olvidar otorgar permisos:

```
grant exec on sp_thresholdaction to dbo
```

Código 7-142. Uso de grant para otorgar permisos sobre sp_thresholdaction en Sybase.

El procedimiento threshold de última oportunidad: realiza respaldos del log de transacciones, escribe en el error log (usando sentencias **`print`**), ejecuta llamadas a procedimientos remotos, extiende el log.

Cualquier usuario con el permiso **`create procedure`** puede crear un procedimiento threshold. Típicamente, el dueño de la base de datos lo crea. Si el threshold de última oportunidad es cruzado y SQL Server no puede encontrar **`sp_thresholdaction`**, un mensaje de error se va al error log. Todo el trabajo de esa base de datos se cuelga hasta que el log es respaldado.

Espacio libre de Thresholds.

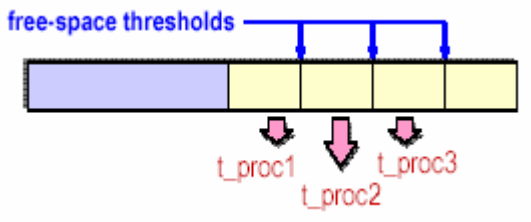


Ilustración 7-84. Esquema de los espacios libres de thresholds en Sybase.

Crear adicionales thresholds, llamados **`free-space thresholds`**, en cualquier segmento de la base de datos (datos o log). Varios **`free-space thresholds`** puede estar en un segmento dado. Crear un procedimiento almacenado asociado con cada **`free-space threshold`**. Cuando ese threshold es cruzado, SQL Server ejecuta el procedimiento almacenado.

`load transaction` lee la anterior copia del respaldo del log de transacciones en el actual log de transacciones para la recuperación.

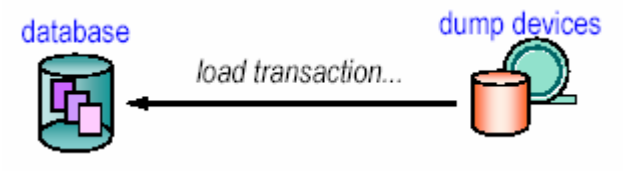


Ilustración 7-85. Esquema de uso de load transaction en Sybase.

```
load transaction salesdb from logdump_dev
```

Código 7-143. Uso de load transaction en Sybase.

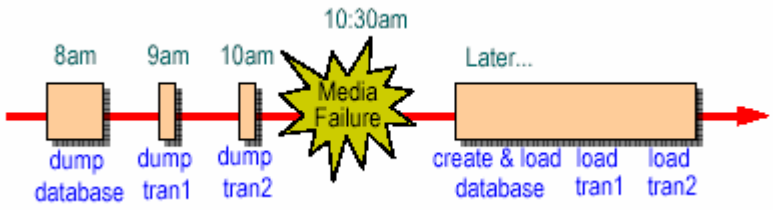


Ilustración 7-86. Proceso de respaldo y recuperación de una base de datos en Sybase.

Las transacciones deberán ser cargadas en el orden correcto, y tiene que estar todas. **`dump transaction`** sin un previo **`dump database`** no pasa nada. **`load transaction`** sin un previo **`load database`** fallará. Cargados parciales no son posibles.

Recuperación Up-to-the-Minute

En el caso de una falla en el medio, ¿puede recuperar cambios que ocurren desde el último respaldo de transacciones? Si el log y la base de datos están en el mismo disco físico, y ese disco falla, la recuperación de esos cambios no es posible. Si están en discos separados, tres posibles escenarios de falla existen.

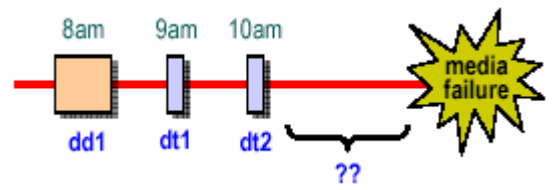


Ilustración 7-87. Esquema de falla en el medio en Sybase.

Tres Escenarios

Si el disco de la base de datos esta bien, pero el disco del log falla... ¿Qué pasa si el dispositivo del log es duplicado? ¿Qué pasa si el dispositivo del log no es duplicado?

Si el disco de la base de datos falla, pero el disco del log esta bien... ¿Qué sucede?

Si fallan el disco de la base de datos y del log... ¿Qué pasa si el dispositivo del log es duplicado? ¿Qué pasa si el dispositivo del log no es duplicado?

Recuperando el log

Usando **`dump transaction with no_truncate`** se obtiene una recuperación up-to-the-minute conocida también como "recovering the log".

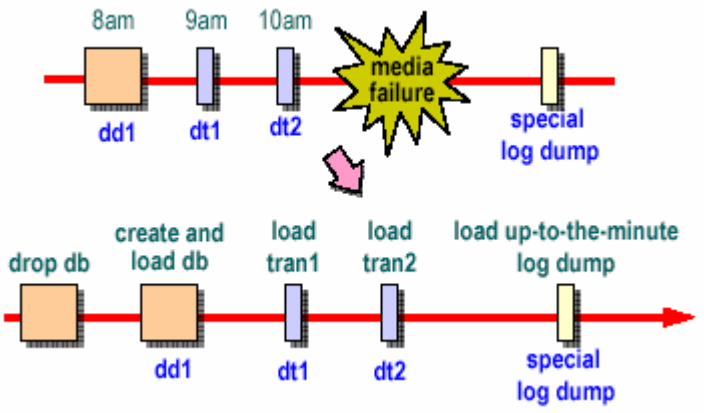


Ilustración 7-88. Esquema de uso de recuperación recovering the log en Sybase.

Requerimientos. El Log y la base de datos deben estar en discos separados. La base de datos **`master`** debe estar disponible. Antes de borrar la base de datos defectuosa, es recomendado ejecutar **`dump transaction with no_truncate`**.

Recuperación Point-in-Time

La recuperación Point-in-time recupera la base de datos en un tiempo o punto particular en el log de transacciones usando una nueva opción de usuario **`load transaction`**. Puede ser usado por cualquier base de datos en la cuál el log de transacciones puede respaldarse o cargarse. No se aplica en bases de datos en donde el log y los datos están en el mismo dispositivo. No puede ser usado en cualquier base de datos que haya tenido un log truncado desde el último **`dump database`**.

Beneficios. Recuperación en una sentencia. Si una consulta errónea destruye datos, la base de datos puede ser llevada al punto antes que los datos fueran destruidos. Puede ser usado para obtener una aproximación cercana de la consistencia del cruce de base de datos. Puede ser usada para servidores con trabajo semipesado.

Recuperación Point-in-Time: Interfase de Usuario

```
load transaction dbname from devname with until_time = "date-time"
```

Código 7-144. Sintaxis de load transaction con recuperación Point-in-Time en Sybase.

```
1> load transaction employees_db
2> from "/dev/nrmt5"
3> with until_time = "Jan 16, 1997 12:45:59:650PM"
4> go
```

Código 7-145. Uso de load transaction con recuperación Point-in-Time en Sybase.

Formato Fecha-Hora: Mes, día, año hh:mm:ss:ms [am|pm]. La resolución es más/menos 1/300th de un segundo (siempre que sea especificado en milisegundos).

Un usuario accidentalmente borró una tabla importante. (Idealmente, tiene el tiempo exacto en milisegundos antes de que ocurriera el borrado).

1. Usar **`sp_who`** para listar los usuarios que han estado utilizando la base de datos, entonces tiene los usuarios que se han desconectado de la base (Si es necesario, use el comando **`kill`** para eliminar procesos del usuario).

2. Cambiar la base de datos al modo **`single-user`**

```
1>use master
2>go

1>sp_dboption employees_db "single user", true
2>go

1>use employees_db
2>go

1>checkpoint
2>go
```

Código 7-146. Cambiar una base de datos a modo single-user en Sybase.

3. Respaldar el log de transacciones de la base de datos que contiene el error.

```
1>dump transaction employees_db to "/dev/nrmt5"
2>go
```

Código 7-147. Respaldo de log de transacciones de una base de datos en Sybase.

4. Cargar el respaldo más reciente que fue hecho para la base de datos

```
1>use master
2>go

1>load database employees_db from "/dev/nrmt4"
2>go
```

Código 7-148. Cargar respaldo de la base de datos en Sybase.

5. Cargar el log de transacciones que fue creado después que la base de datos fuera respaldada por última vez. Entonces cargar el log de transacciones que registra el error usando la opción **until_time** a un punto antes del error.

```
1>load transaction employees_db from "/dev/ nrmt5"
2>with until_time = "Jan 16 1997 12:45:59:650PM"
3>go
```

Código 7-149. Cargar el log de transacciones hasta un punto anterior al error en Sybase.

6. Permita a los usuarios entrar a la base de datos apagando el modo **single user**.

```
1>online database
2>go

1>sp_dboption employees_db "single user", false
2>go

1>use employees_db
2>go

1>checkpoint
2>go
```

Código 7-150. Proceso para desactivar el modo single user en Sybase.

Estrategia de Respaldo. La base de datos **master** contiene información crítica. Mantener respaldos recientes de **master**, especialmente después de crear, alterar, o borrar bases de datos, agregar o eliminar dispositivos, agregar cuentas y usuarios, mantener scripts para realizar esas operaciones.

Si **master** está dañada o ya no es recuperable, hay que reconstruirla, el disco que mantiene el dispositivo fallido de **master**, SQL Server no puede arrancar, **dbcc** reporta daños, también debe reconstruir **master** para duplicarla en otra máquina.

Reconstruyendo **master** con un respaldo Up-to-Date.

Si se tiene respaldos up-to-date. Usar **buildmaster –m** para construir una nueva base de datos **master**. Reiniciar el servidor para un solo usuario. Agregar un dispositivo de respaldo, si es necesario. Cargar **master** desde un respaldo. Reiniciar SQL Server con **startserver**. Verificar la consistencia: ejecutando **dbcc checkalloc** en cada base de datos y examine las tablas importantes. Si todo está bien, reinicie SQL Server para uso multiusuario. Nunca ejecute **buildmaster** mientras SQL Server esté funcionando.

Si no tiene respaldos up-to-date, entonces siga las siguientes instrucciones:

- Reagregar dispositivos usando **disk reinit**.
- Reducir base de datos creadas usando **disk refit**.
- Agregar todos los logins y cuentas de usuarios usando scripts **isql**.
- Reproducir algunos cambios en **model** y (tamaño de) **tempdb**.

- Si es más fácil mantener respaldos up-to-date.

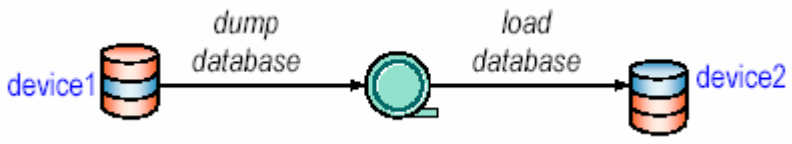


Ilustración 7-89. Esquema de mover una base de datos a otro dispositivo.

Usar respaldo y cargar para mover una base de datos a un diferente dispositivo.

```
disk init name = "device2", ...
dump database salesdb to tape_dev
drop database salesdb
create database salesdb on device2
load database salesdb from tape_dev
```

Código 7-151. Proceso de mover una base de datos a otro dispositivo en Sybase.

Respaldar y cargar a través de plataformas usualmente no es posible. Volver a crear y cargar tablas manualmente.

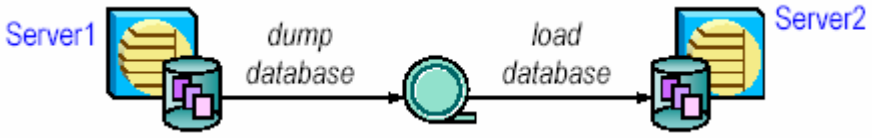


Ilustración 7-90. Esquema de movimiento de una base de datos de un servidor a otro.

Posibles conflictos: suid, uid, nombre login, nombre de usuario. Asegurarse que la base de datos que se crea en Server2 coincida con la original. Asegurarse que el SQL Server destino se use el mismo set de ordenamiento y set de caracteres como el original.

Buenas prácticas de respaldo. Desarrollar, documentar, y probar un conjunto de procedimientos de respaldo y recuperación: hacer respaldos frecuentes a **`master`**, respaldar el log de **`master`** frecuentemente, mantener un respaldo reciente de **`model`**, hacer respaldos frecuentes de base de datos y log de transacciones para todas las bases de datos, mantenga estadísticas acerca de cuánto tiempo toma un respaldo o una carga, y cuanto espacio es requerido, usar **`dbcc`** para verificar la integridad de sus bases de datos antes de realizar respaldos.

Respaldar la base de datos después de: **`bcp`** a alta velocidad. **`create index`**, **`select into`**, **`dump transaction with no_log`**, **`dump transaction with truncate_only`**.

Automatizar donde sea apropiado usando procedimientos threshold y scripts que ejecuten respaldos. Hay que asegurarse que el archivo de interfaces file esté correcto. Catalogue y etiquete sus medios de respaldos cuidadosamente.

Desarrollando una buena estrategia de respaldo. Decisiones necesarias con respecto a los respaldos: que dispositivos de respaldos usar, y cómo manejarlos, si se usa un **`Backup Server`** remoto, si atender y/o desatender los respaldos. Cada caso necesita involucrarse con los siguientes pasos: comenzar con respaldos manuales en un sólo dispositivo de respaldo, usando solo un **`Backup Server`** local, en un

horario fijo, monitoreando el uso del espacio manualmente, comenzar respaldos en múltiples dispositivos, automatizar usando utilerías del sistema operativo para ejecutar scripts de respaldos, o con el manejador threshold, agregar una red de respaldo e instalar un **Backup Server** en esa máquina.

Otras formas de protegerse contra fallos de medios son usar dispositivos duplicados, usar la replicación del servidor.

Resumiendo respaldos

- Respaldos de base de datos: **dump**/**load database**.
  - Realizados por el **Backup Server**.
  - Multiarchivo, multivolumen, local, remoto, manual, automático.
  - Usos: fallo del medio, mover bases de datos.
- Respaldos incrementales: **dump**/**load transaction**.
  - Usado sólo en conjunción con **dump**/**load database**
  - Existen varias opciones de truncado.
- La recuperación Up-to-the-minute es posible.
- Reconstruir la base de datos **master** usando **buildmaster**.
- Desarrollar, documentar, probar, procedimientos de respaldo y recuperación y manejar las cintas cuidadosamente.

7.2 ADMINISTRACIÓN DE BASES DE DATOS EN POSTGRESQL

Instalación y configuración de PostgreSQL

Instalación y configuración de PostgreSQL

Obtener PostgreSQL

PostgreSQL se incluye en su sitio Web con muchos sitios FTP que disponen de los archivos de PostgreSQL para su descarga; Se puede encontrar una lista completa de los mismos en **http://www.postgresql.org**.

Una vez haya conectado a un mirror FTP de PostgreSQL, se verá las versiones estables localizadas dentro de un directorio que comienza con **v** seguido por una versión (tal como **v7.1.3/0**). Debería haber también un enlace simbólico al directorio de la versión estable más reciente, llamada **latest/**.

Dentro de este subdirectorio se encontrará una larga lista de archivos de paquetes. El paquete completo de instalación de PostgreSQL se denomina **postgresql-[versión].tar.gz** y debería ser el archivo de mayor tamaño de la lista. Los siguientes subpaquetes también están disponibles para su descarga, y pueden ser instalados en cualquier combinación (aunque sólo el paquete base es requerido):

postgresql-base-[version].tar.gz . El paquete base. Contiene la cantidad mínima de código para construir y ejecutar PostgreSQL.

postgresql-docs-[version].tar.gz . El paquete de documentos contiene la documentación de PostgreSQL en formato HTML. Advierta que las páginas **man** de PostgreSQL son automáticamente instaladas con el paquete base.

postgresql-opt-[version].tar.gz . El paquete **opt** contiene varias extensiones opcionales para PostgreSQL, tales como las interfaces para C++ (libpq++), JDBC, ODBC, Perl, Python, y Tcl. También contienen el código fuente requerido para el soporte multibyte.

postgresql-test-[version].tar.gz . El paquete **test** contiene la utilidad de testeo de regresión. Este paquete es requerido para ejecutar los tests de regresión una vez compilado PostgreSQL.

Crear el usuario postgres

Crear una cuenta de usuario UNIX para poseer y gestionar los archivos de bases de datos de PostgreSQL. Normalmente, este usuario es denominado **postgres**, pero se puede usar el nombre que desee. El usuario **postgres** será considerado el usuario **root** o súper usuario de PostgreSQL.

Necesitará tener privilegios de **root** para crear al súper usuario de PostgreSQL.

```
$ su -c "useradd postgres"
```

Código 7-152. Creando el usuario postgres en Linux.

Aviso: No intentar usar al usuario **root** como súper usuario de PostgreSQL. Hacer esto representa un enorme riesgo de seguridad.

Instalar el paquete fuente de PostgreSQL

Una vez haya descargado el fuente de PostgreSQL, se debería copiar el paquete con las fuentes en un directorio temporal de compilación. Este directorio será la ruta donde instalará y configurará PostgreSQL. Dentro de esta ruta, extraerá el contenido del archivo **tar.gz** y procederá con la instalación.

Recordar que esta no será la localización definitiva de los archivos de la base de datos. Es una localización temporal para configurar y compilar el paquete con las fuentes. Si se ha bajado el paquete PostgreSQL desde Internet, es muy probable que no se haya almacenado en su directorio temporal de compilación (a menos que lo haya especificado explícitamente antes de la descarga). Una convención común para construir código fuente en máquinas UNIX y Linux es construir dentro de la ruta **/usr/local/src**. Seguramente necesitará permisos de **root** para acceder a esta ruta. Así, los restantes ejemplos en este capítulo implicarán al usuario **root** hasta que no especifiquemos lo contrario.

Nota: Si es usuario de una distribución comercial de Linux, verificar si ya tiene o no instalado PostgreSQL. En sistemas basados en RPM, tales como SuSe, Mandrake, o RedHat, esto puede ser hecho usando el siguiente comando: **rpm -qa | grep -i postgres**. Si se comprueba que tiene PostgreSQL instalado, es muy probable que no esté actualizado. Una instalación RPM de PostgreSQL normalmente instala scripts y programas tales como **postmaster** y **psql** en directorio de acceso global. Esto puede causar conflictos con las versiones de código fuente, así que antes de instalar una nueva versión, asegúrese de eliminar el RPM usando el comando **rpm -e <nombre paquete>**.

Para desempaquetar el código fuente PostgreSQL sobre un sistema Linux, primero mueva (o copie desde el CD) el archivo fuente comprimido en **/usr/local/src** (la mayoría de la gente mueve sus archivos fuente aquí para mantenerlos separados de sus directorios personales y otras localizaciones). Tras moverlo a la ubicación del sistema de archivos donde va a realizar el desempaquetado, use **tar** para desempaquetar los archivos fuente.

```
[root@host root]# mv postgresql-7.1.3.tar.gz /usr/local/src
[root@host root]# cd /usr/local/src
[root@host src]# tar -xzvf postgresql-7.1.3.tar.gz
postgresql-7.1.3/
postgresql-7.1.3/ChangeLogs/
postgresql-7.1.3/ChangeLogs/ChangeLog-7.1-7.1.1
postgresql-7.1.3/ChangeLogs/ChangeLog-7.1RC1-to-7.1RC2
postgresql-7.1.3/ChangeLogs/ChangeLog-7.1RC2-to-7.1RC3
postgresql-7.1.3/ChangeLogs/ChangeLog-7.1RC3-to-7.1rc4
postgresql-7.1.3/ChangeLogs/ChangeLog-7.1beta1-to-7.1beta3
postgresql-7.1.3/ChangeLogs/ChangeLog-7.1beta3-to-7.1beta4
postgresql-7.1.3/ChangeLogs/ChangeLog-7.1beta4-to-7.1beta5
postgresql-7.1.3/ChangeLogs/ChangeLog-7.1beta5-to-7.1beta6
postgresql-7.1.3/ChangeLogs/ChangeLog-7.1beta6-7.1RC1
postgresql-7.1.3/ChangeLogs/ChangeLog-7.1rc4-7.1
postgresql-7.1.3/ChangeLogs/ChangeLog-7.1.1-7.1.2
postgresql-7.1.3/ChangeLogs/ChangeLog-7.1.2-7.1.3
postgresql-7.1.3/COPYRIGHT
[...]
[root@host root]# chown -R postgres.postgres postgresql-7.1.3
```

Código 7-153. Desempaquetando el paquete fuente de PostgreSQL.

El comando **`chown -R postgres.postgres postgresql-7.1.3`**. Este comando garantiza que el propietario del directorio raíz de PostgreSQL es el usuario **`postgres`**, lo cual le permitirá compilar PostgreSQL como el usuario **`postgres`**. Una vez la extracción y el cambio de propietario se hayan producido, podrá cambiar al usuario **`postgres`** para compilar PostgreSQL, resultando que todos los archivos compilados automáticamente pertenecerán a este usuario.

Para propósitos de referencia, la siguiente lista es una descripción de las opciones de **`tar`** usadas para extraer la distribución fuente de PostgreSQL:

`x` (extraer). **`tar`** extraerá desde el nombre de archivo especificado (en oposición a la creación de un nuevo archivo).

`v` (verboso). **`tar`** imprimirá como salida los archivos que son extraídos. Se puede omitir esta opción si no desea ver la lista de archivos desempaquetados.

`z` (comprimido). **`tar`** usará **`gunzip`** para descomprimir el fuente. Esta opción asume que usted está usando las herramientas GNU; otras versiones de **`tar`** pueden no soportar la opción **`z`**. En el caso de que no esté usando herramientas GNU, necesitará descomprimir manualmente el archivo antes de que pueda desempaquetarlo con **`tar`**.

`f` (archivo). **`tar`** usará el archivo que sigue a continuación del parámetro **`f`** para determinar qué archivo desempaquetar. En nuestros ejemplos, este archivo es **`postgresql-7.1.3.tar.gz`**.

Una vez haya completado la extracción de los archivos, conviértase en el usuario **`postgres`** y cambie al directorio recién creado (por ejemplo **`/usr/local/src/postgres-7.1.3`**). El resto de pasos de nuestra instalación tendrán lugar en ese directorio.

Configurar el árbol de fuentes

Antes de la compilación, se debe configurar el archivo fuente, y especificar las opciones de instalación específicas para sus necesidades. Esto se realiza con el script **`configure`**.

El script **`configure`** se usa también para chequear dependencias del software requerido para compilar PostgreSQL. Además de comprobar las dependencias, él creará los archivos necesarios para usarlos con el comando **`gmake`**.

Para usar el script de instalación por defecto, utilizar el siguiente comando: **./configure**. Para especificar las opciones que activarán ciertas características que no están disponibles por defecto, añadir la opción al comando **./configure**. Para una lista de todas las opciones de configuración disponibles, usar **./configure -help**.

Es muy posible que la configuración que por defecto usa **configure** no sea la configuración que se necesita. Para una correcta instalación de PostgreSQL, se recomienda que se use por lo menos las siguientes opciones:

> **-with-CXX**. Le permite construir programas C++ para usarlos con PostgreSQL, construyendo la librería libpq++.
>
> **-enable-odbc**. Le permite conectar PostgreSQL con programas que tienen un driver ODBC compatible (tal como Microsoft Access).
>
> **-enable-multibyte**. Le permite usar caracteres multibyte, tales como los caracteres de lenguajes no anglosajones (por ejemplo Kanji).
>
> **-with-maxbackends=NÚMERO**. Establece **NÚMERO** como el máximo número de conexiones admitidas (32, por defecto).

Compilando el fuente

Tras usar el comando configure, se puede comenzar a compilar el fuente de PostgreSQL introduciendo el comando **gmake**.

Nota: En máquinas Linux, se debería usar **make** en vez de **gmake**. Los usuarios de BSD deberían usar **gnumake**.

```
[postgres@host postgresql-7.1.3]# gmake
gmake -C doc all
gmake[1]: Entering directory /usr/local/src/postgresql-7.1.3/doc'
gmake[1]: Nothing to be done for all'.
gmake[1]: Leaving directory /usr/local/src/postgresql-7.1.3/doc'
gmake -C src all
gmake[1]: Entering directory /usr/local/src/postgresql-7.1.3/src'
gmake -C backend all
gmake[2]: Entering directory /usr/local/src/postgresql-7.1.3/src/backend'
gmake -C utils fmgroids.h
gmake[3]: Entering directory
/usr/local/src/postgresql-7.1.3/src/backend/utils'
[...]
```

Código 7-154. Compilando el Fuente con GNU make en UNIX.

La compilación de PostgreSQL puede durar unos 10 minutos, una hora, o incluso más dependiendo la computadora en donde se instale. Una vez la compilación haya finalizado, debería aparecerle el siguiente mensaje: **"All of PostgreSQL is successfully made. Ready to install."**

Instalando los programas y librerías compilados

Una vez haya configurado y compilado el código fuente de PostgreSQL, es la hora de instalar las librerías, binarios y archivos de datos compilados en una ubicación más adecuada del sistema. Si se está actualizando desde una versión anterior de PostgreSQL, asegurarse de salvar su base de datos antes de iniciar este paso.

El comando **su** temporalmente accede como el usuario **root** para que pueda ejecutar los comandos requeridos.

Nota: Si se especificó un directorio de instalación distinto al directorio de instalación por defecto, se utiliza el directorio que se especificó, en lugar del directorio **/usr/local/pgsql**.

```
$ su -c "gmake install"
Password:
gmake -C doc install
gmake[1]: Entering directory /usr/local/src/postgresql-7.1.3/doc'
mkdir /usr/local/pgsql
mkdir /usr/local/pgsql/man
mkdir /usr/local/pgsql/doc
mkdir /usr/local/pgsql/doc/html
[...]
$ su -c "chown -R postgres.postgres /usr/local/pgsql"
Password:
```

Código 7-155. El comando de instalación gmake.

El comando **su -c "gmake install"** instalará el recientemente compilado código fuente en la estructura de directorio que se escogió con la opción de configuración **-prefix**, o, si no fue especificado, en el directorio por defecto **/usr/local/pgsql**. El uso del comando **su -c "chown -R postgres.postgres /usr/local/pgsql"** garantizará que el usuario **postgres** es el propietario de los directorios de instalación de PostgreSQL. El uso del comando **su -c** permite saltarse un paso, accediendo como el usuario **root** sólo durante la ejecución del comando.

Estableciendo las Variables de Entorno

El uso de las variables de entorno de PostgreSQL no es requerido. Sin embargo, son útiles cuando se realizan tareas dentro de PostgreSQL, incluyendo el arranque y parada de los procesos del **postmaster**. Las variables de entorno que deberían ser configuradas lo son para las páginas **man** y para el directorio **bin**. Se puede hacer esto añadiendo los siguientes elementos en el archivo **/etc/profile**. Esto debería funcionar para cualquier shell[5] basada en sh, incluyendo las shells bash y ksh.

```
PATH=$PATH:/usr/local/pgsql/bin
MANPATH=$MANPATH:/usr/local/pgsql/man
export PATH MANPATH
```

Código 7-156. Configuración de las variables de entorno en el archivo /etc/profile en Unix.

Accediendo de nuevo en el sistema una vez haya actualizado el archivo **/etc/profile** para que la shell las utilice.

Dependiendo de cómo maneja el sistema las librerías compartidas, puede que necesite informar al sistema operativo sobre dónde se encuentran las librerías compartidas de su instalación de PostgreSQL. Sistemas como Linux, FreeBSD, NetBSD, OpenBSD, Irix, HP/UX, y Solaris no lo necesitarán.

En una instalación por defecto, las librerías compartidas se ubican en **/usr/local/pgsql/lib** (esto puede ser diferente, dependiendo de que estableciese un cambio con la opción de configuración -**prefix**). Una de las formas más comunes de hacer esto es configurar la variable de entorno **LD_LIBRARY_PATH** a **/usr/local/pgsql/lib**.

```
$ LD_LIBRARY_PATH=/usr/local/pgsql/lib
$ export LD_LIBRARY_PATH
```

Código 7-157. Configurando LD_LIBRARY_PATH en una shell bash en Unix.

[5] Shell. Ver el Capítulo 5 Fundamentos de Sistemas Operativos.

```
$ setenv LD_LIBRARY_PATH /usr/local/pgsql/lib
```
Código 7-158. Configurando LD_LIBRARY_PATH en csh y tcsh en Unix.

Inicializando y Arrancando PostgreSQL

Si se accedió como el usuario **root**, en lugar de usar el comando su -c de los pasos anteriores, se necesitará ingresar como el usuario **postgres** que fue añadido. Una vez que se haya ingresado como el usuario **postgres** ejecutar:

```
$ /usr/local/pgsql/bin/initdb -D /usr/local/pgsql/data
```
Código 7-159. Inicializando la base de datos en Unix.

La opción **-D** en el ejemplo anterior es la localización donde los datos serán almacenados. Esta ubicación puede también ser configurada con la variable de entorno **PGDATA**. Si se ha configurado la variable de entorno **PGDATA**, la opción **-D** no es necesaria. Si se desea utilizar un directorio diferente para almacenar los archivos de datos, hay que asegurarse de que el usuario **postgres** puede escribir en ese directorio. Cuando se ejecute **initdb** se verá algo similar a lo que se muestra a continuación:

```
$ /usr/local/pgsql/bin/initdb -D /usr/local/pgsql/data
This database system will be initialized with username "postgres." This user
will own all the data files and must also own the server process.

Creating directory /usr/local/pgsql/data
Creating directory /usr/local/pgsql/data/base
Creating directory /usr/local/pgsql/data/global
Creating directory /usr/local/pgsql/data/pg_xlog
Creating template1 database in /usr/local/pgsql/data/base/1
DEBUG: database system was shut down at 2001-08-24 16:36:35 PDT
DEBUG: CheckPoint record at (0, 8)
DEBUG: Redo record at (0, 8); Undo record at (0, 8); Shutdown TRUE
DEBUG: NextTransactionId: 514; NextOid: 16384
DEBUG: database system is in production state
Creating global relations in /usr/local/pgsql/data/global
DEBUG: database system was shut down at 2001-08-24 16:36:38 PDT
DEBUG: CheckPoint record at (0, 108)
DEBUG: Redo record at (0, 108); Undo record at (0, 0); Shutdown TRUE
DEBUG: NextTransactionId: 514; NextOid: 17199
DEBUG: database system is in production state
Initializing pg_shadow.
Enabling unlimited row width for system tables.
Creating system views.
Loading pg_description.
Setting lastsysoid.
Vacuuming database.
Copying template1 to template0.
```
Código 7-160. Salida de initdb.

Ahora se puede iniciar el servidor de bases de datos usando:

```
/usr/local/pgsql/bin/postmaster -D /usr/local/pgsql/data

o

/usr/local/pgsql/bin/pg_ctl -D /usr/local/pgsql/data -l logfile start
```
Código 7-161. Iniciar el servidor de PostgreSQL.

Se puede indicar que PostgreSQL usará un directorio distinto especificando la localización del directorio con la opción **-D**. Esta ruta debe ser inicializada a través de **initdb**.

Autenticación de Cliente

La autenticación de cliente es una característica central para PostgreSQL. Sin ella, podría sacrificar la conectividad remota, o por el contrario podría permitir que cualquiera pudiera conectar a su base de datos y recibir y incluso modificar sus datos. PostgreSQL tiene varios tipos diferentes de autenticación de cliente a su disposición. Como administrador del sitio, usted necesita decidir cuál de ellos es mejor para su sistema.

A partir de PostgreSQL 7.1.x, los accesos de clientes basados en máquina (host) se encuentran especificados en el archivo **pg_hba.conf**. Los permisos y restricciones descritos en este archivo no deberían ser confundidos con los permisos de un usuario de PostgreSQL a los objetos dentro de la base de datos. El archivo **pg_hba.conf** le permite establecer el tipo de autenticación basada en la máquina a ser usada. Esta autenticación es realizada antes de que PostgreSQL establezca una conexión a la base de datos en cuestión, donde los permisos de usuarios serían relevantes.

El archivo **pg_hba.conf** está localizado en el directorio de datos de PostgreSQL (por ejemplo **/usr/local/pgsql/data/**), y es instalado automáticamente con la ejecución del comando **initdb** cuando PostgreSQL es instalado.

La autenticación de PostgreSQL basada en máquina es flexible, proporcionando una amplia variedad de opciones de configuración. Puede restringir los accesos a bases de datos a máquinas específicas, así como permitir acceso a un rango de direcciones IP usando máscaras de red. Cada máquina configurada tiene su propio registro de máquina, el cual es una línea simple en el archivo **pg_hba.conf**.

Con estos registros de máquinas, se puede especificar el acceso tanto a una base de datos en particular como a todas las bases de datos. Además, puede requerir a un usuario desde determinada máquina que se autentifique a través de la tabla de usuarios de PostgreSQL para calificar una conexión.

Dicho simplemente, el archivo **pg_hba.conf** le permite determinar quién está autorizado para conectarse a qué bases de datos desde qué máquinas, y para graduar cómo deben probar su autenticidad para obtener el acceso.

En el caso de la autenticación remota basada en contraseña, las contraseñas pueden ser transmitidas en texto plano dependiendo de si está o no usando sesiones encriptadas. Hay que asegurarse de comprender cómo está la aplicación comunicándose con PostgreSQL antes de permitir a los usuarios conectar remotamente a una base de datos PostgreSQL.

Autenticación de contraseña

Las contraseñas permiten a los usuarios PostgreSQL una forma de identificarse a sí mismos y de prevenir a individuos no autorizados la conexión. A partir de PostgreSQL 7.1.x, las contraseñas de usuario son almacenadas en un texto plano en la tabla de sistema **pg_shadow**. La estructura de esta tabla se ilustra en la siguiente tabla. Aunque que las contraseñas son almacenadas como texto plano, sólo los superusuarios de PostgreSQL tienen permiso para ver la tabla **pg_shadow**.

Columna	Tipo
usename	name
usesysid	integer
usecreatedb	boolean
usetrace	boolean
usesuper	boolean
usecatupd	boolean
passwd	text
valuntil	abstime

Tabla 7-5. Estructura de pg_shadow para almacenar contraseñas en PostgreSQL.

La tabla **pg_shadow** es una tabla del sistema, y por tanto es accesible desde cualquier base de datos. Esto significa que los usuarios no están adscritos a una base de datos específica. Si un usuario existe en la tabla **pg_shadow**, ese usuario estará habilitado para conectar a cualquier base de datos en el servidor, aunque no necesariamente desde cualquier máquina remota.

Los usuarios normalmente establecen contraseñas en PostgreSQL cuando son creados (con el comando **CREATE USER**) o una vez que el usuario ha sido creado (usando el comando **ALTER USER**). Alternativamente, se puede manualmente modificar una contraseña de usuario usando un **UPDATE**.

Si una contraseña no es definida, el valor por defecto para el campo contraseña (passwd) del usuario en la tabla es **NULL**. Si la autenticación basada en contraseña es activada en el archivo **pg_hba.conf**, los intentos de conexión siempre fallarán para ese usuario **user**. A la inversa, si la máquina que establece la conexión es una máquina autorizada (tal como **localhost**, por ejemplo), cualquiera desde la máquina autorizada se puede conectar como usuario con una clave **NULL**. De hecho, las contraseñas son ignoradas completamente para las máquinas autorizadas.

El comando **GRANT** le permite restringir o permitir una gran variedad de tipos de acceso a tablas dentro de una base de datos.

A menos que las necesidades de seguridad sean muy pequeñas, no se querrá usar sólo autenticación basada en contraseña en el servidor PostgreSQL. Usar éste método para autenticar usuarios permitirá a cualquier usuario válido acceder a cualquier base de datos del sistema, y la autenticación con una clave de texto plano puede resultar en que usuarios no autorizados adquieran contraseñas de usuarios. Si se desea conectar a una base de datos desde Internet de alguna forma, se sugiere el uso del archivo **pg_hba.conf** y la encriptación de sesión.

El Archivo pg_hba.conf

El archivo **pg_hba.conf** permite la autenticación de cliente entre el servidor PostgreSQL y la aplicación cliente. Este archivo consiste en una serie de entradas, las cuales definen a una máquina y a sus permisos asociados (por ejemplo, la base de datos a la que le está permitido conectar, el método de autenticación a usar, etc.).

Cuando una aplicación solicita una conexión, la petición especificará un nombre de usuario PostgreSQL y de base de datos a la cual intenta conectar. Opcionalmente, puede ser proporcionada una contraseña, dependiendo de la configuración esperada para la máquina que se conecta.

PostgreSQL tiene sus propias tablas de usuario y contraseñas, las cuales están separadas de las cuentas del sistema operativo. No se requiere que el usuario PostgreSQL coincida con los usuarios definidos a nivel de sistema operativo.

Cuando PostgreSQL recibe una petición de conexión revisará el archivo **pg_hba.conf** para verificar que la máquina desde la cual la aplicación está solicitando la conexión tiene permiso para conectar a la base de datos especificada. Si la máquina que solicita el acceso tiene permisos de conexión, PostgreSQL comprobará las condiciones que la aplicación debe cumplir en orden a una correcta autenticación. Esto afecta a las conexiones que son iniciadas localmente así como a las remotas.

PostgreSQL comprobará el método de autenticación a través del archivo **pg_hba.conf** para cada petición de conexión. Esta comprobación es realizada cada vez que una nueva conexión es solicitada desde el servidor PostgreSQL, así que no se necesita reiniciar PostgreSQL tras añadir, modificar o eliminar una entrada en el archivo **pg_hba.conf**.

```
# PostgreSQL HOST ACCESS CONTROL FILE
#
local all                                   trust
host all       127.0.0.1 255.255.255.255    trust
host booktown  192.168.1.3 255.255.255.255 ident sales
host all       192.168.1.4 255.255.255.255 ident audit
```

Código 7-162. Fragmento de un archivo pg_hba.conf en PostgreSQL.

Cuando una conexión es inicializada, PostgreSQL lee cada entrada del archivo **pg_hba.conf**. Tan pronto como un registro coincidente sea encontrado, PostgreSQL abandonará la búsqueda y permitirá o rechazará la conexión, en base a la entrada encontrada. Si PostgreSQL no encuentra un registro coincidente, la conexión se aborta igualmente.

Los permisos a nivel de tabla todavía se aplican a la base de datos, en el caso de que un usuario tenga permisos para conectar a la base de datos. Si se puede conectar, pero no puede seleccionar datos de una tabla, puede que se tenga que verificar que el usuario conectado tenga permisos para usar **SELECT** sobre esa tabla. Usando la aplicación de línea de comando **psql**, se puede comprobar los permisos de las tablas de una base de datos usando en comando **\z**. Para cualquier otra interfaz para PostgreSQL, se usa la consulta a la tabla **pg_class**.

```
testdb=# SELECT relname as "Relation", relacl as "Access permissions"
testdb-# FROM pg_class
testdb-# WHERE relkind IN ('r', 'v', 'S')
testdb-# AND relname !~ '^pg_'
testdb-# ORDER BY relname;

Relation  | Access permissions
----------+---------------------------------
foo       | {"=arwR","jdrake=arwR"}
my_list   | {"=","jdrake=arwR","jworsley=r"}
(2 rows)
```

Código 7-163. Comprobando Permisos de Usuario en PostgreSQL.

Estructura del archivo pg_hba.conf

El archivo **pg_hba.conf** contiene entradas secuenciales que definen las características que PostgreSQL debería usar durante el proceso de autenticación del cliente para una máquina específica. Este archivo está diseñado para ser fácilmente actualizable.

Dentro de este archivo, se puede asociar una dirección de máquina TCP/IP (o un rango de direcciones) con una base de datos en particular (o todas las bases de datos), y uno o varios métodos de autenticación. También puede especificar el acceso para conexiones locales usando el término **localhost**, o **127.0.0.1**, en vez de usar la dirección IP externa del sistema.

Varias reglas sintácticas se aplican al archivo **pg_hba.conf**. Primero, sólo se puede ubicar un registro de máquina por línea en el archivo. Consecuentemente, los registros de máquinas no pueden extenderse a varias líneas. Segundo, cada registro de máquina debe contener múltiples campos, los cuales deben estar separados bien por tabulaciones o bien por espacios. El número de campos en un registro de máquina está directamente relacionado con el tipo de entrada que está siendo definida. El ejemplo siguiente muestra dos registros de máquina, el primero con los campos separados por espacios, y el segundo separado mediante tabulaciones.

```
host all 127.0.0.1 255.255.255.255 trust
host     all      127.0.0.1 255.255.255.255 trust
```

Código 7-164. Una entrada válida en pg_hba.conf con espacios y tabulaciones en PostgreSQL.

Se permite el uso de comentarios dentro del archivo **pg_hba.conf** usando un signo de gato (**#**) al principio de cada línea de comentario.

```
# Book Town host entries
#
#
host all 127.0.0.1 255.255.255.255 trust
```
Código 7-165. Comentarios válidos en pg_hba.conf en PostgreSQL.

A parte del actual formato para cada registro de máquina, hay tres tipos generales disponibles en el archivo **pg_hba.conf** (el tipo **keyword** es siempre la primera palabra en el registro de máquina):

- **host**. Una entrada **host** es usada para especificar máquinas remotas que están autorizadas para conectar al servidor PostgreSQL. El **postmaster** de PostgreSQL debe ser ejecutado con la opción **-i** (TCP/IP) para que una entrada de máquina funcione correctamente.
- **local**. Una entrada local es semánticamente lo mismo que una entrada **host**. Sin embargo, no necesita especificar una máquina a la que le esté permitido conectar. La entrada local es usada para conexiones de clientes que son iniciadas desde la misma máquina en la que está operando el servidor PostgreSQL.
- **hostssl**. Una entrada **hostssl** es usada para especificar máquinas (remotas o locales) que están autorizadas para conectar al servidor PostgreSQL usando SSL. El uso de SSL le garantiza que todas las comunicaciones entre el cliente y el servidor están encriptadas. Para que esto funcione, tanto el cliente como el servidor deben soportar SSL. El proceso **postmaster** debe ser ejecutado con las opciones **-l** (SSL) y **-i** (TCP/IP).

El ejemplo siguiente ilustra la sintaxis general de cada tipo de registro de máquina disponible dentro del archivo **pg_hba.conf**. El formato es esencialmente idéntico para cada registro, con la excepción de que un registro local no requiere una dirección IP o máscara de red, ya que se asume que la conexión se realiza desde la misma máquina donde se encuentra en servidor PostgreSQL.

```
# Un registro "local".
local database auth_method [ auth_option ]

# un registro "host".
host database ip_addr netmask auth_method [ auth_option ]

# Un registro "hostssl".
hostssl database ip_addr netmask auth_method [ auth_option ]
```
Código 7-166. Sintaxis de entradas en el archivo pg_hba.conf en PostgreSQL.

Cada entrada en el archivo **pg_hba.conf** debe ser una línea simple. No se puede usar varias líneas o líneas divididas.

La siguiente lista es una descripción de las palabras clave para las entradas en el archivo **pg_hba.conf**:

- **database**. Este es el nombre de la base de datos a la que la máquina especificada está autorizada a conectar. La palabra clave **database** tiene tres posibles valores:
  - **all**. La palabra clave **all** especifica que el cliente puede conectar a cualquier base de datos alojada en el servidor PostgreSQL.
  - **sameuser**. Especifica que el cliente sólo puede conectar a una base de datos en la que coinciden los nombre de usuarios autenticados de los clientes.
  - **name**. Un nombre determinado puede ser especificado, de forma que el cliente sólo puede conectar a la base de datos especificada por ese nombre.
- **ip_addr**, **netmask**. Los campos **ip_addr** y **netmask** especifican una dirección IP, o rango de direcciones, que están autorizadas a conectar al servidor PostgreSQL. Dicho rango puede ser

especificado describiendo una red IP con su máscara de red asociada. De lo contrario, para una única dirección IP, el campo **netmask** debería ser establecido a **255.255.255.255**.

- **auth_method**. El método de autenticación especifica el tipo de autenticación que el servidor debería usar para un usuario que intenta conectar a PostgreSQL. Lo siguiente es una lista de opciones disponible para el método **auth_method**:
    - **trust**. El método **trust** permite a cualquier usuario del **host** definido conectar a una base de datos PostgreSQL sin el uso de una contraseña, como cualquier usuario PostgreSQL. Se está autorizando la autenticación basada en máquina con el uso de éste método, y para cualquier usuario de la máquina especificada. Esta es una condición insegura si la máquina especificada no es una máquina segura, o proporciona acceso a usuarios desconocidos.
    - **reject**. El método **reject** automáticamente deniega el acceso a PostgreSQL para esa máquina o usuario. Esta puede ser una configuración prudente para sitios en los que nadie está autorizado a conectar al servidor de bases de datos.
    - **password**. El método **password** especifica que debe existir una contraseña para una conexión de usuario. El uso de éste método requerirá que el usuario que conecta proporcione una contraseña que coincida con la contraseña encontrada en la tabla global de sistema **pg_shadow** para ese nombre de usuario. Si se usa éste método, la contraseña será enviada en texto plano.
    - **crypt**. El método **crypt** es similar al método **password**. Cuando se usa **crypt**, la contraseña no es enviada en texto plano, sino a través de un formato simple de encriptación. El uso de éste método no es muy seguro, pero es mejor que usar el método de contraseñas planas.
    - **krb4**, **krb5**. Los métodos krb4 y krb5 son usados para especificar la versión 4 ó 5 del sistema de autenticación Kerberos[6].
    - **ident**. El método **ident** especifica que un mapa de identidad debería ser usado cuando una máquina está solicitando conexiones desde una IP válida listada en el archivo **pg_hba.conf**. Éste método requiere una opción. La opción requerida puede ser tanto el término especial **sameuser**, o un mapa nombrado que es definido dentro del archivo **pg_ident.conf**.
    - **auth_option**. El campo **auth_option** puede o no ser requerido, en función del tipo de método de autenticación que sea usado; a partir de PostgreSQL 7.1.x, sólo el método **ident** requiere una opción.

No se aconseja el uso ni de **password** ni de **crypt** sin el uso de un mecanismo externo de encriptación por sesiones encriptadas.

[6] Kerberos es un protocolo de autenticación de redes de ordenador que permite a dos computadores en una red insegura demostrar su identidad mutuamente de manera segura.

El archivo pg_ident.conf

Cuando se especifica el término **ident** como método de autenticación de registros de máquina, PostgreSQL usa el archivo **pg_ident.conf** para identificar el nombre de identificación de usuario a un nombre de usuario de PostgreSQL. El nombre de identificación de usuario es el nombre proporcionado por el servicio de conexión de cliente **identd** (RFC 1413), el cual es requerido para identificar el nombre de la cuenta de sistema que está iniciando la conexión. Este método es similar al método **trust**, pero restringe el acceso en base al nombre de identificación del usuario.

Tal como se indica en la especificación del protocolo **ident**, "El Protocolo de Identificación no está indicado como protocolo de autorización o de control de acceso". Se trata sólo de un método útil de identificación para mayor seguridad, mecanismos de control, y no está pensado como control seguro para un amplio arreglo de máquinas externas. Esto se debe a que un demonio **identd** simplemente retorna un nombre de usuario arbitrario describiendo al actual usuario del sistema. Por ejemplo, permitir al usuario **jworsley** desde una subred completa o una dirección IP podría crear un serio riesgo de seguridad, porque cualquiera con una máquina en dicha subred podría crear un usuario llamado **jworsley** y convertirse como resultado en usuario "autorizado".

El archivo **pg_ident.conf** debería ubicarse en la misma ruta que el archivo **pg_hba.conf**. Este debería ser la ruta definida por la variable de entorno PGDATA (por ejemplo **/usr/local/pgsql/data**). Al igual que el archivo **pg_hba.conf**, los cambios en el archivo **pg_ident.conf** no requieren reiniciar PostgreSQL.

El contenido de **pg_ident.conf** asocia nombres de usuarios identificados con nombres de usuarios PostgreSQL a través de definiciones denominadas mapas de identificación. Esto es útil para aquellos usuarios cuyos nombres de usuario de sistema no coinciden con sus nombres de usuario de PostgreSQL. Se debe recordar algunas reglas cuando se defina y se use un mapa de identificación, como son las siguientes:

- Cada miembro del mapa de identificación es definido en una única línea, la cual asocia un nombre mapeado con un nombre de identificación de usuario y su traslado a un nombre de usuario PostgreSQL.
- El archivo **pg_ident.conf** puede contener múltiples nombres de mapas. Cada grupo de líneas simples con el mismo nombre de mapa asociativo es considerado como un único mapa.
- El archivo **pg_hba.conf** determina los tipos de conexiones que relacionan a los usuarios en este archivo.

Una única línea para definir un mapa de identificación consiste en tres elementos: el nombre del mapa, el nombre de usuario que se identifica, y su traslación a nombre de usuario en PostgreSQL. Esta sintaxis es introducida como sigue, donde cada elemento está separado por espacios o tabulaciones:

```
nombre_mapa nombre_identidad nombre_postgresql
```

Código 7-167. Sintaxis de configuración de una línea en el archivo pg_ident.conf en PostgreSQL.

nombre_mapa. El nombre de mapa usado en el archivo **pg_hba.conf** para referirse al mapa de identificación.

nombre_identidad. El nombre identificador de usuario, el cual es generalmente el nombre de usuario del sistema que intenta establecer una conexión a la base de datos. Este es el nombre proporcionado por el demonio **identd**, el cual debe ser ejecutado en el sistema al cual se intenta conectar.

nombre_postgresql. El nombre de usuario de la base de datos que está autorizado para nombre identificador del usuario. Se puede especificar varias líneas con el mismo nombre de identidad, pero con diferentes nombres postgresql, en orden a permitir a un mismo usuario del sistema acceder a varias cuentas, lo cual no necesariamente ha de estar referido a la misma base de datos.

Como ejemplo, suponiendo que el servidor de la Ciudad del Libro (Book Town) tiene una serie de cuentas de sistema denominadas **jdrake**, **jworsley** y **auditor**, usadas por dos vendedores y por un auditor interno, respectivamente.

Puede ser que se desee crear un par de mapas de identidad para estos dos grupos de usuarios. Suponiendo que la estación de trabajo del departamento de ventas tiene la dirección IP **192.168.1.3**, y que sólo necesita acceso a la base de datos **booktown**, mientras que la estación de trabajo del departamento de auditoria tiene la dirección IP **192.168.1.4**, y requiere acceso a todas las bases de datos. Este escenario podría resultar en un archivo **pga_hba.conf**, tal como el mostrado a continuación:

```
host booktown 192.168.1.3 255.255.255.255 ident sales
host all      192.168.1.4 255.255.255.255 ident audit
```

Código 7-168. Una configuración de identidad en un archivo pg_hba.conf en PostgreSQL.

Esta configuración de acceso de máquinas establece que la máquina de ventas puede conectar a la base de datos **booktown** usando un mapa de identificación llamado **sales**, y que la estación de trabajo de auditoria puede conectar a cualquier base de datos usando un mapa de identificación llamado **audit**. Cada uno de estos mapas debe ser configurado dentro del archivo **pg_ident.conf**. A continuación se muestra dicha configuración:

```
# MAP IDENT POSTGRESQL_USERNAME
sales jdrake   sales
sales jworsley sales
audit auditor  sales
audit auditor  postgres
```

Código 7-169. Una configuración de pg_ident.conf en PostgreSQL.

El archivo mostrado en el de **pg_ident.conf** permite a los usuarios de sistema **jdrake** o **jworsley** conectar como el usuario **sales**, y permite al usuario de sistema llamado **auditor** conectar a PostgreSQL tanto como el usuario **sales** o el usuario **postgres**.

Es posible que un nombre de usuario identificador sea mapeado a múltiples usuarios PostgreSQL.

Si sólo usará la identidad para identificar a sus usuarios remotos, no necesita usar el archivo **pg_ident.conf**. Puede usar en su lugar el término especial **sameuser** en el archivo **pg_hba.conf**, en lugar de usar un mapeado de nombres.

De nuevo, esto es similar al método real, sin embargo la identidad **sameuser** restringe las conexiones basadas en el nombre de usuario proporcionado por **identd**. Proporcionar un nombre de usuario PostgreSQL para conectar (por ejemplo, con la opción **-U** en **psql**) que es diferente al nombre enviado por **identd** resultará en un fallo de conexión. El uso del mapeado **sameuser** se demuestra a continuación:

```
host booktown 192.168.1.0 255.255.255.0 ident sameuser
```

Código 7-170. Una configuración sameuser en pg_hba.conf PostgreSQL.

El registro de máquina entonces permite a cualquier máquina en el bloque de red 192.168.1 conectar a la base de datos **booktown**, usando el nombre de usuario en PostgreSQL que coincide con el nombre

de usuario proporcionado por **identd**. El término **sameuser** causa que PostgreSQL implícitamente compare el solicitado nombre de usuario PostgreSQL contra el nombre proporcionado por **identd**.

Fallo en la autentificación

Cuando se produce un fallo en la autentificación, PostgreSQL usualmente lanza un útil mensaje de error, en vez de simple fallo. Los siguientes son mensajes de error típicos con los que se puede encontrar, con sus respectivas explicaciones:

```
FATAL 1: user "testuser" does not exist
```

Código 7-171. Error el nombre de usuario "testuser" no existe.

El nombre especificado de usuario no fue encontrado en la tabla de sistema **pg_shadow**, significando que el usuario no existe.

```
FATAL 1: Database "testdb" does not exist in the system catalog
```

Código 7-172. Error en que la base de datos "testdb'" no existe en el catálogo de sistema.

Esta base de datos no pudo ser encontrada porque no existe. Advierta que si no especifica un nombre de base de datos a la conexión PostgreSQL, éste intentará conectar al nombre de usuario proporcionado.

```
No pg_hba.conf entry for host 123.123.123.1, user testuser, database testdb
```

Código 7-173. Error donde no hay entrada en pg_hba.conf para la máquina 123.123.123.1, usuario testuser, base de datos testdb.

Ha tenido éxito en cuanto a conectar con el servidor, pero el servidor no aceptó la conexión. El servidor rechazó la conexión porque no pudo encontrar una entrada para **testuser** usando **testdb** en la dirección IP (**123.123.123.1**) en el archivo **pg_hba.conf**.

```
Password authentication failed for user 'testuser'
```

Código 7-174. Falló la autentificación de contraseña para el usuario "testuser"

Ha tenido éxito en cuanto a contactar con el servidor, pero la conexión falló debido a un error en la autentificación. Comprobar la contraseña que se ha proporcionado al servidor, y asegurarse de que es correcta. Además, puede verificar los programas Kerberos o Ident si los está usando para autenticación de contraseñas. También se puede comprobar si el usuario dispone de contraseña. Para todos aquellos usuarios sin una contraseña definida, se asigna una contraseña **NULL** a cada usuario. Cuando el usuario intenta ingresar y no especifica una contraseña, se comparará la contraseña **NULL** con la entrada **NULL**, y retornará **false**. Por otro lado, si el usuario intenta suministrar una contraseña (aunque sea una en blanco), se comparará la entrada con la contraseña **NULL** y retornará **false**. Si está usando autentificación con contraseña, debe asignar una contraseña a todos los usuarios. Si no se asigna una contraseña al usuario en este escenario, la autentificación de contraseña siempre fallará, y el usuario nunca podrá acceder.

Encriptando Sesiones

En la era digital, la privacidad e integridad de los datos se han convertido en dos de las áreas más activas de la computación. Parece que cada día alguien en algún lugar ha sido crackeado, o que un nuevo agujero de sistema ha sido encontrado en la aplicación en la que se confiaba.

Al mismo tiempo, la encriptación de las sesiones de datos se ha vuelto algo común en la mayoría de computadoras. Todo sitio respetable de comercio electrónico usa SSL (Secure Sockets Layer) para

proteger los datos de los usuarios mientras se transmite información personal tal como tarjetas de crédito y otros datos personales a través de Internet.

El tipo más común de crack ejecutado sobre una máquina no es realmente un "crack". Es usualmente un usuario no sospechoso usando un protocolo tal como POP o FTP para transferir información sobre Internet. Usando estos protocolos, el usuario puede inconscientemente transmitir su login y password en texto plano (en un formato no encriptado) a través de Internet.

La transmisión de datos tales como el login y las passwords en texto plano sobre Internet significa que cualquiera usando un programa sniffer[7] podría potencialmente obtener acceso a su información más personal. En el mundo de las bases de datos, el escenario no es diferente.

Si se conecta remotamente a PostgreSQL sin el uso de una tecnología de encriptación, hay un hueco potencial para los crackers en Internet. Si un cracker usa un sniffer sobre la red, o sobre una red entre su cliente y el servidor de bases de datos al que se está conectando, este puede obtener acceso total a la información que está almacenada dentro de PostgreSQL.

Encriptación de datos entre PostgreSQL y las conexiones de los clientes:

- **`Built-in SSL`** El soporte **`built-in SSL`** en PostgreSQL, activándose con la opción **`enabled with the -with-ssl`** en tiempo de compilación, permite a **`psql`** (o a cualquier cliente específicamente escrito para conectar a PostgreSQL a través de SSL) conectar de modo seguro a PostgreSQL.
- **`SSH/OpenSSH`** Una sesión **`SSH`** (Secure SHell) puede ser usada para crear un túnel hacia un servidor remoto, suponiendo que un demonio **`SSH`** (por ejemplo, **`sshd`**) está instalado y es accesible por el usuario que conecta. Esto requiere acceso desde shell al sistema que corre PostgreSQL para cada usuario que desee conectar.
- **`Stunnel`**. **`Stunnel`** es una aplicación que crea un túnel encriptado entre un cliente y el servidor PostgreSQL. El método **`Stunnel`** requiere acceso shell para configurar, pero puede ser configurado para correr en un sistema cliente para un usuario que no tiene acceso directo desde shell al servidor remoto.

Built-in SSL

PostgreSQL proporciona la opción de compilarlo con soporte para SSL con el parámetro de configuración **`-with-ssl`**. Esta opción es una buena elección si se va a realizar la mayoría del trabajo con PostgreSQL a través de **`psql`**, ya que nativamente éste soporta dicho método de conexión. La mayoría de la gente escoge usar PostgreSQL como motor para una variedad de aplicaciones cliente. Si este es el caso, necesitará desarrollar su propio cliente para comprender conexiones SSL hacia PostgreSQL, o bien seleccionar un método externo de encriptación de sesiones entre su cliente o aplicación y el servidor PostgreSQL (tal como SSH, o Stunnel).

[7] Sniffer. Un paquete sniffer es un programa de captura de las tramas de red. Generalmente se usa para gestionar la red con una finalidad docente, aunque también puede ser utilizado con fines maliciosos.Captura automática de contraseñas enviadas sin encriptar y nombres de usuario de la red. Esta capacidad es utilizada en muchas ocasiones por crackers para atacar sistemas.

SSH/OpenSSH

OpenSSH proporciona un excelente método para usar encriptación externa entre cliente y servidor. OpenSSH es un estándar comúnmente implementado entre los profesionales de la seguridad y los administradores de sistemas. Es más comúnmente usado para aplicaciones terminales o de transferencia de archivos. El protocolo SSH es un método general de encriptación, y puede ser aplicado de forma general para cualquier aplicación.

Suponiendo que se tiene acceso a una cuenta de sistema en el servidor remoto, se puede autentificarse ante el sistema y abrir un túnel entre la máquina remota y la local con el la opción -L. Dicho túnel escuchará por un puerto específico en la máquina local, encriptará los paquetes de datos entrantes, y los enviará al servidor remoto en un formato encriptado. Los datos luego serán desencriptados y reenviados a otro puerto en el servido remoto.

De esta forma, se puede fácilmente crear un túnel encriptado de datos entre el cliente y el servidor. Además, todo el proceso es invisible para PostgreSQL, el cual creerá que está aceptando paquetes entrantes desde la misma máquina en la que está corriendo, desde el usuario que se autentificó con la creación del túnel. Teniendo cuidado que el archivo **pg_hba.conf** deberá apuntar a la máquina apropiada.

El ejecutable SSH es denominado normalmente **ssh**, y puede ser usado para crear un túnel con la siguiente sintaxis:

```
ssh -L puerto_local:maquina_remota:puerto_remoto usuario@maquina_remota
```

Código 7-175. Sintaxis de ssh para especificar el puerto y máquina tanto como local y remota a conectarse en Unix.

El puerto local es cualquier puerto por el que se quiera escuchar localmente. Este puerto debe estar por debajo del **1024**, a menos que se haya ingresado como el usuario **root**, lo cual no es aconsejable. Este número será el puerto local por el cual el cliente creará que está conectado al servidor PostgreSQL. Realmente, los datos recibidos sobre este puerto serán reenviados a la máquina remota a su puerto de escucha SSH (normalmente el **22**), serán desencriptados, y luego reenviados por el servido remoto a sí mismo, al número de puerto remoto especificado.

La frase **usuario@maquina_remota** debe ser proporcionada en orden a autentificar a un usuario de sistema válido. Sin una cuenta de sistema válida, un túnel SSH no puede ser creado. Todo este proceso es demostrado a continuación, en el cual los puntos suspensivos separan un par de sesiones de terminal. La primera conexión de terminal crea el túnel SSH, y debe permanecer activa en orden a que el túnel exista. La segunda sesión de terminal toma ventaja del túnel para hacer una conexión al puerto local del túnel, el cual es entonces reenviado a la máquina remota, desencriptado, y pasado al servidor PostgreSQL.

```
[user@local ~]$ ssh -L 4001:remotehost:5432 user@remotehost
user@remotehost's password:
[user@remote ~]$
...
 [user@local ~]$ psql -h localhost -p 4001 template1
Welcome to psql, the PostgreSQL interactive terminal.

Type: \copyright for distribution terms
      \h for help with SQL commands
      \? for help on internal slash commands
      \g or terminate with semicolon to execute query
      \q to quit

template1=#
```

Código 7-176. Creando un túnel SSH para PostgreSQL.

Cuando se utilice el comando **ssh**, se puede especificar la opción **-T** si no necesita que se le proporcione una línea de comandos tras crear el túnel SSH, que es la característica por defecto. Esto provocará que el terminal parezca colgarse tras la autenticación. La sesión puede ser terminada con **CTRL-C** cuando finalice.

El único problema a la hora de usar un túnel SSH es que éste requiere una cuenta de sistema para el usuario que está conectando al servidor PostgreSQL. SSH no proporciona acceso completamente transparente a cadenas de datos encriptados hasta que no inicie una conexión y se autentifique contra el demonio del servicio **ssh**, el cual es normalmente denominado servicio **sshd**. Dependiendo de las necesidades esta podría ser una restricción positiva o negativa.

Gestión de usuarios y grupos

Al igual que en la mayoría de los sistemas de bases de datos, la gestión de los usuarios y grupos juega un importante papel dentro de PostgreSQL. Utilizado correctamente, los usuarios y grupos tendrán un sencillo acceso a los objetos de las bases de datos.

PostgreSQL almacena tanto los datos de usuarios como los de grupos dentro de sus propios catálogos de sistema. Estos son distintos a los definidos dentro del sistema operativo sobre el cual está instalado el software. Cualquier conexión a PostgreSQL debe ser realizada con un usuario específico, y cualquier usuario puede pertenecer a uno o más grupos definidos.

La tabla de usuarios controla los permisos de acceso y quién está autorizado a realizar acciones en el sistema (y qué acciones puede realizar). Los grupos existen como un mecanismo para simplificar la ubicación de estos permisos. Tanto las tablas de usuarios como de grupos existen como objetos globales de base de datos, lo que significa que no están adscritas a ninguna base de datos en particular.

Gestionando Usuarios

En orden a establecer una conexión a PostgreSQL, se debe proporcionar una forma básica de identificación. Esta es denominada un nombre de usuario, que identifica al usuario al cual el sistema reconocerá como conectado a una base de datos. Los usuarios dentro de PostgreSQL no están necesariamente relacionados con los usuarios del sistema operativo (los cuales son también denominados cuentas del sistema), aunque se puede escoger nominar a sus usuarios de PostgreSQL como a las cuentas de sistema a través de las que ellos accederán.

Cada usuario tiene un **ID** de sistema interno en PostgreSQL (llamado **sysid**), así como una contraseña, aunque la contraseña no es necesariamente imprescindible para conectar (dependerá de la configuración del archivo **pg_hba.conf**. El **ID** de sistema del usuario es utilizado para asociar objetos en una base de datos con su propietario (el usuario que está autorizado para dar/quitar privilegios sobre un objeto).

Así como son usados para asociar objetos de base de datos con su propietario, los usuarios también pueden tener permisos globales asignados a ellos cuando estos son creados. Estos permisos o privilegios determinan si un usuario podrá o no crear y eliminar bases de datos, o si el usuario es o no un superusuario (un usuario que tiene todos los permisos, en todas las bases de datos, incluyendo la capacidad de crear a otros usuarios). La asignación de estos permisos puede ser modificada en cualquier momento por cualquier superusuario.

PostgreSQL crea por defecto a un súper usuario llamado **postgres**. Todos los demás súper usuarios pueden ser creados por éste, o por cualquier otro súper usuario creado posteriormente.

Distinguiendo a los usuarios

Toda la información relativa los usuarios es almacenada en una tabla de sistema de PostgreSQL llamada **pg_shadow**. Esta tabla es sólo seleccionable por los súper usuarios, aunque una vista limitada de esta tabla, llamada **pg_user**, es accesible por los usuarios normales.

Columna	Tipo
usename	name
usesysid	integer
usecreatedb	boolean
usetrace	boolean
usesuper	boolean
usecatupd	boolean
passwd	text
valuntil	abstime

Tabla 7-6. La tabla pg_shadow en PostgreSQL.

La principal diferencia entre los datos seleccionables en **pg_user** y **pg_shadow** es que el actual valor de la columna **passwd** no es mostrado (éste es reemplazado por asteriscos). Esta es una característica de seguridad para garantizar que los usuarios normales no están capacitados para determinar las contraseñas de los demás.

La columna **usename** almacena el nombre del usuario del sistema, el cual es una cadena de caracteres única (no pueden haber dos usuarios con mismo nombre, ya que los usuarios son objetos globales de base de datos). Similarmente, la columna **usesysid** almacena un valor entero único asociado con el usuario. Las columnas **usecreatedb** y **usesuper** respectivamente corresponden al par de privilegios que pueden ser establecidos tras la creación del usuario.

Creando usuarios

PostgreSQL proporciona dos métodos para la creación de usuarios de bases de datos. Cada uno de ellos requiere autenticación como súper usuario, ya que sólo los súper usuarios pueden crear nuevos usuarios.

El primer método es a través del uso del comando **SQL CREATE USER**, el cual puede ser ejecutado a través de cualquier cliente PostgreSQL, por ejemplo **psql**. El segundo es un programa de línea de comandos llamado **createuser**, el cual puede ser más conveniente para un administrador de sistemas, ya que puede ser ejecutado como un simple comando sin necesidad de interactuar a través de un cliente PostgreSQL.

Creando un usuario con el comando SQL CREATE USER

El comando **CREATE USER** requiere sólo un parámetro: el nombre del nuevo usuario. También hay una variedad de opciones que pueden ser establecidas, incluyendo una contraseña, un ID de sistema explícito, grupo, y un juego de permisos que pueden ser específicamente definidos.

```
CREATE USER nombre_usuario
[ WITH [ SYSID uid ]
       [ PASSWORD 'password' ] ]
[ CREATEDB | NOCREATEDB ]
[ CREATEUSER | NOCREATEUSER ]
[ IN GROUP groupname [, ...] ]
[ VALID UNTIL 'abstime' ]
```

Código 7-177. Sintaxis completa para CREATE USER.

En ésta sintaxis, **nombre_usuario** es el nombre del nuevo usuario que va a ser creado. No se puede tener dos usuarios con el mismo nombre. Mediante el uso de la palabra clave **WITH**, pueden aplicarse las palabras clave **SYSID** y **PASSWORD**.

Cada una de las otras palabras clave opcionales pueden seguir en el orden mostrado (no se requiere el uso de **WITH**). Lo siguiente es una explicación detallada de cada palabra clave opcional y su significado:

- **SYSID uid**. Especifica que el **ID** de sistema que va a definirse debe establecerse al valor de **uid**. Si se omite, un razonable y único valor numérico por defecto es escogido.
- **PASSWORD 'password'**. Establece la nueva contraseña del usuario a **password**. Si no se especifica, la contraseña por defecto es **NULL**.
- **CREATEDB | NOCREATEDB**. Usando la palabra clave **CREATEDB** se le garantiza al nuevo usuario el privilegio de crear nuevas bases de datos, así como el de destruir las de su propiedad. Usando **NOCREATEDB** se deniega este permiso (que es lo que ocurre por defecto).
- **CREATEUSER | NOCREATEUSER**. Garantiza el privilegio de crear nuevos usuarios, lo cual implícitamente crea a un súper usuario. Un usuario con los privilegios de crear a otros usuarios tendrá todos los privilegios, en todas las bases de datos (incluyendo los permisos para crear una base de datos, aunque se haya especificado **NOCREATEDB**). **NOCREATEUSER** explícitamente fuerza a la situación por defecto, que deniega el privilegio.
- **IN GROUP nombre_grupo [, ...]**.Añade al nuevo usuario al grupo llamado **nombre_grupo**. Pueden ser especificados múltiples nombres de grupo, separándolos mediante comas. El/los grupos deben existir para que funcione el estamento.
- **VALID UNTIL 'abstime'**. Establece que la contraseña del usuario expirará el **abstime**, el cual debe ser un formato reconocible de fecha/hora (timestamp). Tras esa fecha, la contraseña se resetea, y la expiración se hace efectiva.
- **VALID UNTIL 'infinity'**. Establece validez permanente para la contraseña del usuario.

Si no se especifica **CREATEDB** o **CREATEUSER**, los usuarios son implícitamente "normales", sin privilegios especiales. No pueden crear bases de datos u otros usuarios, ni pueden eliminar bases de datos o usuarios. Estos usuarios pueden conectar a bases de datos en PostgreSQL, pero sólo pueden ejecutar condiciones para los que han sido autorizados.

```
template1=# CREATE USER salesuser
template1-# WITH PASSWORD 'N0rm4!';
CREATE USER
```

Código 7-178. Creando un usuario normal en PostgreSQL.

El ejemplo anterior crea a un usuario normal llamado **salesuser**. También establece una contraseña de **N0rm4!** mediante el uso de la cláusula **WITH PASSWORD**. Omitiendo la cláusula **VALID UNTIL**, esta contraseña nunca expirará. El mensaje del servidor **CREATE USER** retornado indica que el usuario fue creado correctamente. Otros mensajes que se puede recibir son los siguientes:

- **ERROR: CREATE USER: permission denied**. Este mensaje es retornado si el usuario que utilizó el comando **CREATE USER** no es un súper usuario. Sólo los súper usuarios pueden crear nuevos usuarios.
- **ERROR: CREATE USER: user name "salesuser" already exists**. Este mensaje indica que un usuario con el nombre **salesuser** ya existe.

Si se desea crear un usuario con la capacidad de crear bases de datos en PostgreSQL pero que no pueda crear o eliminar a usuarios PostgreSQL, puede especificar la palabra clave **CREATEDB** en vez de **CREATEUSER**. Esto permite que el usuario pueda crear bases de datos arbitrariamente, así como eliminar cualquier base de datos de la que él sea propietario.

```
template1=# CREATE USER dbuser CREATEDB
template1-# WITH PASSWORD 'DbuS3r'
template1-# VALID UNTIL '2008-11-11';
CREATE USER
```
Código 7-179. Creando un usuario con permisos CREATEDB en PostgreSQL.

El ejemplo anterior ilustra la creación de un usuario llamado **dbuser** quien tiene permisos para crear nuevas bases de datos. Esto se hace mediante la especificación de la palabra clave **CREATEDB** tras el nombre de usuario. Advierta también el uso de las palabras clave **WITH PASSWORD** y **VALID UNTIL**. Estas establecen la contraseña para **dbuser** a **DbuS3r**, la cual será válida hasta el 11 de Noviembre de 2008.

El hecho de resetear una contraseña de usuario expirada no modifica el valor de **VALID UNTIL**. En orden a reactivar el acceso de un usuario cuya contraseña ha expirado, tanto la palabra clave **WITH PASSWORD** y **VALID UNTIL** deben ser proporcionadas al comando **ALTER USER**. Los valores en **VALID UNTIL** sólo son relevantes en sistemas que no son de confianza; los sitios de confianza no requieren contraseñas.

Se puede desear crear un súper usuario alternativo al usuario **postgres**, aunque debería tener cuidado a la hora de crear súper usuarios. Estos usuarios tienen garantizados todos los privilegios dentro de PostgreSQL, incluyendo la creación de usuarios, eliminación de usuarios, y eliminación de bases de datos.

```
template1=# CREATE USER manager CREATEUSER;
CREATE USER
```
Código 7-180. Creación de un súper usuario en PostgreSQL.

El ejemplo anterior demuestra la creación de un súper usuario PostgreSQL llamado manager desde el prompt de **psql**.

Creando un usuario con el script createuser

El script **createuser** es ejecutado directamente desde la línea de comandos, y puede operar de dos formas. Si se utiliza sin argumentos, él interactivamente pedirá el nombre de usuario y cada uno de los privilegios que se le van a asignar, e intentará realizar una conexión local a PostgreSQL. Alternativamente, se puede optar por especificar las opciones y el nombre del usuario a ser creado en la misma línea de comandos.

Al igual que con otras aplicaciones de línea de comandos para PostgreSQL, los argumentos pueden ser suministrados en formato corto (con un único guión, y un carácter), o en su formato largo (con dos guiones, y el nombre completo del argumento).

```
createuser [ opciones ] [ nombre_usuario ]
```
Código 7-181. Sintaxis de createuser en desde el shell de Unix.

El **nombre_usuario** en la sintaxis representa el nombre del usuario que se va a crear. A continuación se describen las opciones de **createuser**:

- **-d, -createdb**. Equivalente a la palabra clave **CREATEDB** del comando **SQL CREATE USER**. Permite al nuevo usuario crear bases de datos.

- **-D, -no-createdb**. Equivalente a la palabra clave **NOCREATEDB** del comando **SQL CREATE USER**. Explícitamente indica que el nuevo usuario no puede crear bases de datos. Esta es la situación por defecto.
- **-a, -adduser**. Equivalente a la palabra clave **CREATEUSER** del comando **SQL CREATE USER**. Permite al nuevo usuario la creación de otros usuarios, y asigna el status de súper usuario al usuario (activando todos los privilegios dentro de PostgreSQL).
- **-A, -no-adduser**. Equivalente a la palabra clave **NOCREATEUSER** del comando **SQL CREATE USER**. Explícitamente indica que el nuevo usuario no es súper usuario. Esta es la situación por defecto.
- **-i SYSID, -sysid=SYSID**. Establece el nuevo **ID** de sistema del usuario a **SYSID**.
- **-P, -pwprompt**. Resulta en una petición de introducción de contraseña, permitiéndole establecer la contraseña del nuevo usuario.
- **-h NOMBRE_MAQUINA, -host=NOMBRE_MAQUINA**. Especifica desde qué **NOMBRE_MAQUINA** se conectará, además de la local (**localhost**), o la máquina definida por la variable de entorno **PGHOST**.
- **-p PUERTO, -port=PUERTO**. Especifica que la conexión de base de datos se realizará por el puerto **PUERTO**, en vez de por el puerto por defecto (usualmente el 5432).
- **-U NOMBRE_USUARIO, -username=NOMBRE_USUARIO**. Especifica que **NOMBRE_USUARIO** será el usuario que conecte a PostgreSQL (por defecto se conecta usando el nombre de usuario del sistema).
- **-W, -password**. Resulta en una petición de contraseña para el usuario que conecta, lo cual ocurre automáticamente si el archivo **pg_hba.conf** está configurado para no confiar en la máquina solicitante.
- **-e, -echo**. Causa que el comando **CREATE USER** envíe a PostgreSQL para ser desplegado todo lo que se ejecute con **createuser**.
- **-q, -quiet**. Previene que la salida sea enviada a **stdout** (aunque los errores serán enviados a **stderr**).

Si alguno de los argumentos **-d**, **-D**, **-a**, **-A**, o **nombre_usuario** son omitidos, **createuser** pedirá la introducción de cada uno de ellos. Esto se debe a que PostgreSQL no realizará ninguna asunción sobre los privilegios que se deben asignar al nuevo usuario, ni sobre el nombre del mismo.

```
[jworsley@booktown ~]$ createuser -U manager -D -A newuser
CREATE USER
```

Código 7-182. Creando a un usuario con createuser en PostgreSQL.

El ejemplo anterior crea a un usuario llamado **newuser**, el cual no tiene permisos ni para crear bases de datos ni para crear nuevos usuarios. La opción "**-U manager**" pasado al script **createuser**. Este indica que el usuario que el nombre de usuario con el que se realizará la conexión a PostgreSQL es manager, y no **jworsley** como el script podría haber asumido, basándose en el nombre de la cuenta de sistema que ha invocado al script.

Si se prefiere que de forma interactiva se le vaya preguntando por cada parámetro de configuración (en vez de tener que recordar el significado de cada opción), puede simplemente omitirlos. El script **createuser** le preguntará por las opciones básicas de **createuser**. Estas opciones incluyen el nombre de usuario PostgreSQL, si el usuario podrá crear bases de datos, y si el usuario podrá o no crear nuevos usuarios para PostgreSQL.

```
[jworsley@booktown ~]$ createuser
Enter name of user to add: newuser
Shall the new user be allowed to create databases? (y/n) n
Shall the new user be allowed to create more new users? (y/n) n
CREATE USER
```

Código 7-183. Creación interactiva con createuser.

El ejemplo anterior demuestra el uso del script **createuser** en modo interactivo.

Eliminando Usuarios

Los usuarios de PostgreSQL pueden ser eliminados en cualquier momento del sistema por los superusuarios. La única restricción es que un usuario no puede ser eliminado si existen bases de datos de su propiedad. Si un usuario es propietario de una base de datos, esa base de datos debe ser eliminada antes de poder eliminar al usuario.

Al igual que con la creación de usuarios en PostgreSQL, hay dos métodos mediante los cuales los usuarios pueden ser eliminados. Estos son el comando **SQL DROP USER**, y el ejecutable en línea de comandos **dropuser**.

Manejando Grupos

Los grupos sirven para simplificar la asignación de privilegios. Los privilegios ordinarios deben ser asignados a un único usuario, uno cada vez. Esto puede resultar tedioso si los usuarios a los que hay que asignar los mismos accesos a una gran variedad de objetos de base de datos.

Los grupos son creados para evitar este problema. Un grupo requiere un nombre, y puede ser creado vacío (sin usuarios). Una vez creado, los usuarios que se pretende compartan permisos comunes son añadidos todos al grupo, y quedan asociados al grupo por su número de miembro. Los permisos en los objetos de base de datos son entonces asignados al grupo, en vez de a cada uno de los miembros del grupo. Para un sistema con muchos usuarios y bases de datos, los grupos hacen que la asignación de permisos sea más cómoda para el administrador. Los usuarios pueden pertenecer a cualquier número de grupos, o a ninguno.

Creando un Grupo

Cualquier súper usuario puede crear un nuevo grupo en PostgreSQL con el comando **CREATE GROUP**.

```
CREATE GROUP nombre_grupo
            [ WITH [ SYSID groupid ]
            [ USER username [, ...] ] ]
```

Código 7-184. Sintaxis de CREATE GROUP en PostgreSQL.

En ésta sintaxis, **nombre_grupo** es el nombre del grupo a ser creado. Un nombre de grupo debe comenzar por un carácter alfabético, y no puede superar los 31 caracteres de longitud. El proporcionar la palabra clave **WITH** palabra clave le permite especificar cualquiera de los atributos opcionales. Si se desea especificar el **ID** de sistema a usar con el nuevo grupo, se utiliza la palabra clave **SYSID** para especificar el valor de **groupid**. Se usa la palabra clave **USER** para incluir a uno o más usuarios al grupo en tiempo de creación. Es necesario separar los distintos usuarios mediante comas.

Adicionalmente, el usuario PostgreSQL y las tablas de grupos operan separadamente las unas de las otras. Esta separación que los ID de usuarios y grupos puedan ser idénticos dentro del sistema PostgreSQL.

```
booktown=# CREATE GROUP sales
booktown-# WITH USER allen, vincent;
CREATE GROUP
```

Código 7-185. Creando un grupo en PostgreSQL.

En este código se crea el grupo sales, y añade dos usuarios al grupo en tiempo de creación. Estos usuarios son **allen** y **vincent**. El mensaje del servidor **CREATE GROUP** indica que el grupo fue creado con éxito. Se debe verificar la creación del grupo, así como ver todos los grupos existentes, esto se hace con una consulta a la tabla de sistema **pg_group**.

```
booktown=# SELECT * FROM pg_group;
    groname | grosysid | grolist
------------+----------+------------
      sales | 1        | {7017,7016}
 accounting | 2        |
  marketing | 3        |
(3 rows)
```

Código 7-186. Verificando la existencia un grupo en PostgreSQL.

La columna **grolist** es un arreglo, que contiene el **ID** de usuario de PostgreSQL de cada usuario en el grupo. Estos son los mismos IDs de usuario que se vieron en la vista de **pg_user**.

Eliminando un grupo

Cualquier superusuario puede también eliminar un grupo con el comando SQL **DROP GROUP**. Debería ejecutar con cuidado éste comando, ya que es irreversible, y no será interrogado para que verifique la orden de eliminación del grupo (aunque todavía haya usuarios en el grupo). A diferencia de **DROP DATABASE**, **DROP GROUP** puede ser realizado dentro de un bloque de transacciones:

```
DROP GROUP nombre_grupo
```

Código 7-187. Sintaxis de DROP GROUP en PostgreSQL.

El **nombre_grupo** es el nombre del grupo que va a ser eliminado permanentemente.

```
booktown=# DROP GROUP marketing;
DROP GROUP
```

Código 7-188. Eliminando un grupo en PostgreSQL.

El ejemplo anterior elimina el grupo marketing de la base de datos **booktown**. El mensaje del servidor **DROP GROUP** retornado indica que el grupo fue eliminado con éxito. La eliminación de un grupo no elimina los permisos asignados al mismo, sino que le libera de los mismos. Cualquier permiso asignado a un objeto de base de datos que tiene los permisos asignados a un grupo eliminado aparecerán asignados a un ID de grupo, en vez de a un grupo. Los grupos eliminados por error pueden ser restaurados creando un nuevo grupo con el mismo ID de sistema que el eliminado. Esto implica a la palabra clave SYSID, si se asigna permisos de grupo a una tabla y luego elimina el grupo, los permisos del grupo en la tabla serán mantenidos. Sin embargo, necesitará añadir a los adecuados usuarios al recientemente creado grupo para que los permisos de tabla sean efectivos para los miembros de ese grupo.

Asociando usuarios con grupos

Los usuarios pueden ser añadidos y eliminados de los grupos en PostgreSQL a través del comando SQL **ALTER GROUP**.

```
ALTER GROUP nombre_grupo { ADD | DROP } USER username [, ... ]
```
Código 7-189. Sintaxis del comando ALTER GROUP en PostgreSQL.

El **nombre_grupo** es el nombre del grupo a ser modificado, mientras que **nombre_usuario** es el nombre del usuario a ser añadido o eliminado, dependiendo del uso de las palabras clave **ADD** o **DROP**.

Otorgando Privilegios

PostgreSQL mantiene controlado altamente un juego de listas de control de acceso (ACL[8]). Esta información describe qué usuarios están autorizados para realizar consultas, actualizaciones, etc. o modificar objetos dentro de una base de datos. Existe un juego de privilegios de acceso y de restricciones aplicable para cada objeto de base de datos en PostgreSQL (por ejemplo, tablas, vistas y secuencias). Los súper usuarios y los propietarios de los objetos de bases de datos mantienen estos ACLs a través de un par de comandos SQL: **GRANT** y **REVOKE**.

Cuando un usuario crea por primera vez una base de datos, se vuelve implícitamente el propietario de esa base de datos. Similarmente, cada vez que alguien crea un objeto de base de datos, este es poseído por el individuo que ejecutó el correspondiente comando SQL **CREATE**.

Además de los súper usuarios de PostgreSQL (que pueden manipular cualquier objeto de base de datos de cualquier forma), sólo los propietarios de los objetos de base de datos pueden dar y/o revocar privilegios sobre los objetos de su propiedad. Aunque cualquier usuario pueda conectar a una base de datos, si desea acceso a objetos dentro de la base de datos debe tener los correspondientes privilegios explícitamente otorgados.

Comprendiendo el Control de Acceso

Como ya mencionamos antes en esta sección, las listas de control de acceso se aplican a tres tipos de objetos de base de datos: tablas, listas y secuencias. Para estos objetos, hay cuatro privilegios generales que pueden ser otorgados, o revocados, a un usuario o grupo. Los usuarios y/o grupos no poseen privilegios por defecto.

Desde el cliente **psql**, se puede ver el sumario de permisos de las ACL usando el comando rápido **\z**. Este comando despliega todos los permisos de acceso sobre la base de datos actualmente accedida. Para ver los permisos de un objeto específico, se indica el nombre del objeto como parámetro del comando **\z**. Se puede usar una expresión regular en lugar de un nombre para ver los privilegios de un grupo de objetos.

La tabla siguiente lista cada uno de los privilegios de control de acceso disponibles en PostgreSQL. Cada privilegio también tiene un símbolo asociado, el cual aparece como un carácter alfabético único. Estos símbolos son la abreviatura del privilegio descrito, y son usados por el comando **\z** de **psql** cuando se despliegan sumarios o permisos de acceso.

[8] ACL. Access Control List. Listas de control de acceso.

P.Clave	Símbolo	Descripción
SELECT	r	Permite a un usuario obtener datos de una tabla, vista o secuencia (aunque la función netxval() no puede ser llamada sólo con privilegios SELECT). También conocido como permisos de ``lectura``.
INSERT	a	Permite a un usuario insertar nuevas filas en una tabla. También conocido como permisos de ``adición``.
UPDATE, DELETE	w	Permite a un usuario modificar o eliminar filas de datos de una tabla. Si sólo se asigna uno de los privilegios, el otro es implícitamente otorgado. También conocidos como permisos de "escritura".
RULE	R	Permite a un usuario crear una regla de reescritura sobre una tabla o vista.
ALL	arwR	Representa una forma corta de garantizar o revocar todos los permisos de una sola vez. ALL no es un privilegio en sí mismo. Asignar ALL implica garantizar los permisos SELECT, INSERT, UPDATE, DELETE, y RULE.

Tabla 7-7. Privilegios ACL en PostgreSQL.

Garantizando Privilegios con GRANT

Para asignar un privilegio a un usuario o a un grupo, use el comando de SQL **GRANT**.

```
GRANT privilegio [, ...] ON objeto [, ...]
      TO { PUBLIC | nombre_usuario | GROUP nombre_grupo }
```

Código 7-190. Sintaxis de GRANT en PostgreSQL.

En esta sintaxis, privilegio es cualquiera de los privilegios listados en la tabla anterior, objeto es el nombre del objeto de base de datos (tabla, vista o secuencia) para el que es asignado el privilegio, y el elemento que sigue a la palabra clave **TO** describe a quién es garantizado el privilegio. Se pueden indicar múltiples privilegios y objetos, separados los unos de los otros mediante comas. Sólo uno de los términos a continuación de **TO** puede ser usados en una única instrucción **GRANT**. El otorgamiento de privilegios con la palabra clave **PUBLIC** indiscriminadamente garantiza el privilegio al objetivo especial **"public".** Los privilegios **PUBLIC** son compartidos por todos los usuarios. Especificar un nombre de usuario garantiza el privilegio al usuario especificado. Por el contrario, especificar un nombre de grupo garantiza el privilegio al grupo especificado.´

Supongamos, por ejemplo, que el usuario **manager** necesita todos los permisos para las tablas **customers**, **books**, **editions** y **publishers**. El ejemplo siguiente da al usuario manager dichos privilegios, con una única instrucción **GRANT**.

```
booktown=# GRANT ALL ON customers, books, editions, publishers
booktown-# TO manager;
CHANGE
```

Código 7-191. Otorgando privilegios de usuario.

El uso de la palabra clave **ALL** en el ejemplo anterior otorga todos los posibles permisos ACL (**SELECT**, **UPDATE**, etc.) para los objetos especificados al usuario **manager**. El mensaje **CHANGE** del servidor indica que los privilegios fueron modificados con éxito. Recordando que se puede usar el comando rápido **\z** en **psql** para verificar los permisos establecidos sobre un objeto de base de datos.

```
booktown=# \z publishers
Access permissions for database "booktown"
Relation    | Access permissions
------------+----------------------
publishers  | {"=","manager=arwR"}
(1 row)
```

Código 7-192. Uso de la instrucción \z en la Terminal psql en PostgreSQL.

Como otro ejemplo, el uso de la palabra clave **GROUP** para otorgar privilegios a los miembros de un grupo. Por ejemplo, todo el departamento de ventas (**sales**) **booktown** debería tener permisos para ver la tabla **customers**, pero no para modificarla. El ejemplo siguiente otorga privilegios **SELECT** sobre la tabla **customers** a cualquier miembro del grupo **sales**.

```
booktown=# GRANT SELECT ON customers TO GROUP sales;
CHANGE

booktown=# \z customers
Access permissions for database "booktown"
Relation   | Access permissions
-----------+--------------------------------
customers  | {"=","manager=arwR","group sales=r"}
(1 row)
```

Código 7-193. Otorgando privilegios de grupo en PostgreSQL.

Restringiendo Permisos con REVOKE

Por defecto, un usuario normal no tiene ningún privilegio sobre ningún objeto de la base de datos de la cual no es propietario. Para revocar explícitamente un privilegio que actualmente está otorgado, el propietario del objeto (o un súper usuario) puede usar el comando **REVOKE**. Este comando es muy similar en formato al comando **GRANT**.

```
REVOKE privilege [, ...] ON object [, ...]
       FROM { PUBLIC | username | GROUP groupname }
```

Código 7-194. Sintaxis de REVOQUE en PostgreSQL.

La sintaxis de la estructura del comando **REVOKE** es idéntica a la del comando **GRANT**, con la excepción de que el comando SQL es **REVOKE** en lugar de **GRANT**, y la palabra clave **FROM** es usada, en vez de la palabra clave **TO**.

La revocación de privilegios a **PUBLIC** sólo afecta al grupo especial **"public"**, el cual incluye a todos los usuarios. La revocación de privilegios para **PUBLIC** no afectará a ningún usuario a los que explícitamente se les hayan asignado dichos privilegios.

Supongamos que los privilegios **UPDATE** sobre la tabla **books** han sido otorgados al usuario **david**. Cuando David es transferido a otro departamento, y no necesita más la capacidad de modificar información en la tabla **book**, se debería revocar el privilegio **UPDATE** de **david** sobre la tabla de **libros**.

El ejemplo siguiente usa el comando rápido **\z** de **psql** para comprobar los permisos sobre la tabla de libros, revelando que **david** tiene privilegios de escritura a dicha tabla. Una instrucción **REVOKE** explícitamente revoca entonces los privilegios **UPDATE** y **DELETE** sobre la tabla de libros para el usuario **david**. Finalmente, otra ejecución de **\z** es ejecutada para verificar la revocación del privilegio.

```
booktown=# \z books Access permissions for database "booktown"
Relation  | Access permissions
----------+-------------------------------
books     | {"=","manager=arwR","david=w"}
(1 row)

booktown=# REVOKE UPDATE, DELETE ON books
booktown-# FROM david;
CHANGE

booktown=# \z books
Access permissions for database "booktown"
Relation  | Access permissions
----------+----------------------
books     | {"=","manager=arwR"}
(1 row)
```

Código 7-195. Revocando privilegios en PostgreSQL.

Usando vistas para el control de acceso

Aunque no se puede controlar el acceso de lectura a específicas columnas o filas de una tabla, puede hacerse indirectamente a través del uso adecuado de vistas. Mediante la creación de una vista sobre una tabla, y forzando a los usuarios a acceder a la tabla a través de dicha vista, se puede permitir el acceso sólo a las deseadas filas o columnas seleccionadas.

Limitando las columnas especificando una lista de columnas en la instrucción **`SELECT`** de la vista cuando se crea. La vista sólo retornará las columnas que se especifique. Limitando las filas escribiendo una cláusula **`WHERE`** en la instrucción **`SELECT`** de la vista. La vista entonces retornará sólo aquéllas filas que coincidan con la cláusula **`WHERE`**.

Como los privilegios ACL pueden ser aplicados tanto a las vistas como a las tablas, se puede entonces garantizar privilegios **`SELECT`** para la vista limitada, pero no para la tabla misma. Los usuarios podrán seleccionar desde la vista aunque no tendrán acceso a la tabla.

Por ejemplo, La base **`booktown`** tiene una tabla de **`stocks`** correlacionando un número ISBN de libro con su precio de costo, precio de venta y stock actual. La estructura de la tabla sugerida se muestra en la Tabla 7-8.

Columna	Tipo	Modificador
isbn	text	NOT NULL
cost	numeric(5,2)	
retail	numeric(5,2)	
stock	integer	

Tabla 7-8. Estructura de una tabla de nombre stocks.

Suponiendo que el gerente de Book Town no desea que el personal del ventas tenga acceso al costo de cada libro. Esta información puede ser restringida generando una vista que retorna sólo los datos relativos al isbn, precio de venta y stock existente. El ejemplo siguiente crea dicha vista, otorga permisos al grupo de ventas (**`sales`**), y verifica los privilegios con el comando rápido **`\z`** de **`psql`**.

```
booktown=# CREATE VIEW stock_view
booktown-# AS SELECT isbn, retail, stock
booktown-# FROM stock;
CREATE

booktown=# GRANT SELECT ON stock_view TO GROUP sales;
CHANGE

booktown=# \z stock
Access permissions for database "booktown"
Relation      | Access permissions
--------------+-------------------------------------
stock         |
stock_backup  |
stock_view    | {"=","manager=arwR","group sales=r"}
(3 rows)
```

Código 7-196. Controlando privilegios SELECT con una vista.

El ejemplo siguiente demuestra la adición de un nuevo usuario, **barbara**. Esta garantiza privilegios **SELECT** sobre la vista **stock_view**. Como la usuaria **barbara** no tiene ningún privilegio implícito sobre la tabla de **stocks**, ésta es inaccesible; este es el caso, incluso aunque la vista sobre la tabla es accesible como resultado de la instrucción **GRANT**.

```
booktown=# CREATE USER barbara;
CREATE USER

booktown=# GRANT USER barbara SELECT ON stock_view;
booktown=# \c - barbara
You are now connected as new user barbara.

booktown=> SELECT * FROM stock;
ERROR: stock: Permission denied.

booktown=> SELECT * FROM stock_view;
isbn        | retail | stock
------------+--------+-------
0385121679  | 36.95  | 65
039480001X  | 32.95  | 31
0394900014  | 23.95  | 0
044100590X  | 45.95  | 89
0441172717  | 21.95  | 77
0451160916  | 28.95  | 22
0451198492  | 46.95  | 0
0451457994  | 22.95  | 0
0590445065  | 23.95  | 10
0679803335  | 24.95  | 18
0694003611  | 28.95  | 50
0760720002  | 23.95  | 28
0823015505  | 28.95  | 16
0929605942  | 21.95  | 25
1885418035  | 24.95  | 77
0394800753  | 16.95  | 4
(16 rows)
```

Código 7-197. Controlando SELECT.

Cuando se conecta como el usuario **barbara**, la instrucción **SELECT** para la vista **stock_view** tiene éxito, mientras que la tabla **stock** presenta un error de Permiso Denegado.

Conjunto de caracteres

```
CREATE DATABASE name
    [ [ WITH ] [ OWNER [=] dbowner ]
           [ TEMPLATE [=] template ]
           [ ENCODING [=] encoding ]
           [ TABLESPACE [=] tablespace ]
           [ CONNECTION LIMIT [=] connlimit ] ]
```

Código 7-198. Sintaxis de create database en PostgreSQL.

Usando la instrucción **CREATE DATABASE** de una base de datos nueva en PostgreSQL.

Por defecto, la nueva base de datos será creada clonando la base de datos **template1**. Si se usa **psql –l** en la línea de comandos de Unix se verá que codificación tiene **template 1**. Esto se define al iniciar el cluster de PostgreSQL con **initdb**. **initdb** define el juego de caracteres por defecto por ejemplo:

```
initdb -E EUC_JP
```

Código 7-199. Uso de initdb para implementar un juego de caracteres en PostgreSQL.

En el ejemplo anterior **initdb** especifica el juego de caracteres (encoding) en **EUC_JP** (Extended Unix Code for Japanese). El juego de caracteres soportados en PostgreSQL 8.2 se muestra en la Tabla 7-9.

Si la codificación es diferente de la que se desea, por ejemplo si esta con **SQL_ASCII** y se desea **LATIN1**, se debe crear la base de datos con la opción **ENCODING**.

Como ejemplo, para crear una base de datos de nombre **music** con la opción que soporte el juego de caracteres de **ISO-8859-1** se tendría que declarar como sigue:

```
CREATE DATABASE music ENCODING 'LATIN1';
```

Código 7-200. Uso de encoding en la instrucción create database en PostgreSQL.

Tareas de mantenimiento de la base de datos.

VACUUM. Recolecta basura y opcionalmente analiza una base de datos.

```
VACUUM [ FULL ] [ FREEZE ] [ VERBOSE ] [ table ]
VACUUM [ FULL ] [ FREEZE ] [ VERBOSE ] ANALYZE [ table [ (column [, ...] ) ] ]
```

Código 7-201. Sintaxis de vacuum en PostgreSQL.

Descripción:

VACUUM recupera almacenamiento ocupado por registros borrados. En la operación normal de PostgreSQL, registros que se borran o son obsoletos al realizar una actualización no son quitados físicamente de la tabla; ellos se quedan presentes hasta que un **VACUUM** se haga cargo. Por lo tanto, es necesario para hacer el **VACUUM** periódicamente, especialmente en tablas que son con frecuencia actualizadas.

Ejecutando **VACUUM** sin ningún parámetro, procesa cada tabla en la base de datos actual. Con un parámetro, **VACUUM** procesa sólo esa tabla.

El **VACUUM ANALYZE** realiza el **VACUUM** y entonces un **ANALYZE** analizando cada tabla escogida. Esta es una forma para realizar rutinarias en scripts para el mantenimiento.

Nombre	Descripción	Lenguaje	Server ?	Bytes/ Char	Aliases
BIG5	Big Five	Traditional Chinese	No	1-2	WIN950, Windows950
EUC_CN	Extended UNIX Code-CN	Simplified Chinese	Yes	1-3	
EUC_JP	Extended UNIX Code-JP	Japanese	Yes	1-3	
EUC_KR	Extended UNIX Code-KR	Korean	Yes	1-3	
EUC_TW	Extended UNIX Code-TW	Traditional Chinese, Taiwanese	Yes	1-3	
GB18030	National Standard	Chinese	No	1-2	
GBK	Extended National Standard	Simplified Chinese	No	1-2	WIN936, Windows936
ISO_8859_5	ISO 8859-5, ECMA 113	Latin/Cyrillic	Yes	1	
ISO_8859_6	ISO 8859-6, ECMA 114	Latin/Arabic	Yes	1	
ISO_8859_7	ISO 8859-7, ECMA 118	Latin/Greek	Yes	1	
ISO_8859_8	ISO 8859-8, ECMA 121	Latin/Hebrew	Yes	1	
JOHAB	JOHAB	Korean (Hangul)	Yes	1-3	
KOI8	KOI8-R(U)	Cyrillic	Yes	1	KOI8R
LATIN1	ISO 8859-1, ECMA 94	Western European	Yes	1	ISO88591
LATIN2	ISO 8859-2, ECMA 94	Central European	Yes	1	ISO88592
LATIN3	ISO 8859-3, ECMA 94	South European	Yes	1	ISO88593
LATIN4	ISO 8859-4, ECMA 94	North European	Yes	1	ISO88594
LATIN5	ISO 8859-9, ECMA 128	Turkish	Yes	1	ISO88599
LATIN6	ISO 8859-10, ECMA 144	Nordic	Yes	1	ISO885910
LATIN7	ISO 8859-13	Baltic	Yes	1	ISO885913
LATIN8	ISO 8859-14	Celtic	Yes	1	ISO885914
LATIN9	ISO 8859-15	LATIN1 with Euro and accents	Yes	1	ISO885915
LATIN10	ISO 8859-16, ASRO SR 14111	Romanian	Yes	1	ISO885916
MULE_INTERNAL	Mule internal code	Multilingual Emacs	Yes	1-4	
SJIS	Shift JIS	Japanese	No	1-2	Mskanji, ShiftJIS, WIN932, Windows932
SQL_ASCII	unspecified	any	Yes	1	
UHC	Unified Hangul Code	Korean	No	1-2	WIN949, Windows949
UTF8	Unicode, 8-bit	all	Yes	1-4	Unicode
WIN866	Windows CP866	Cyrillic	Yes	1	ALT
WIN874	Windows CP874	Thai	Yes	1	
WIN1250	Windows CP1250	Central European	Yes	1	
WIN1251	Windows CP1251	Cyrillic	Yes	1	WIN
WIN1252	Windows CP1252	Western European	Yes	1	
WIN1253	Windows CP1253	Greek	Yes	1	
WIN1254	Windows CP1254	Turkish	Yes	1	
WIN1255	Windows CP1255	Hebrew	Yes	1	
WIN1256	Windows CP1256	Arabic	Yes	1	
WIN1257	Windows CP1257	Baltic	Yes	1	
WIN1258	Windows CP1258	Vietnamese	Yes	1	ABC, TCVN, TCVN5712, VSCII

Tabla 7-9. Juego de caractereres soportados en PostgreSQL 8.2.

En **VACUUM** simple (sin **FULL**) recupera simplemente el espacio y lo hace disponible para volverlo a emplear. Esta forma de la instrucción puede operar paralelamente con la lectura y la escritura normal de una tabla, no obteniendo un bloqueo exclusivo. **VACUUM FULL** hace un procesamiento más extenso, incluyendo el de mover registros a través de bloques para tratar de comprimir la tabla a un número mínimo de bloques en el disco. Esta forma es mucho más lenta y requiere un bloqueo exclusivo en cada tabla mientras se procesa.

Parámetros:

- **FULL**. Seleccionando "**full**" **vacuum**, puede recuperar más espacio, pero toma más tiempo y bloquea exclusivamente la tabla.
- **VERBOSE**. Imprime un informe detallado de la actividad del **vacuum** para cada tabla.
- **ANALYZE**. Actualiza la estadística utilizada por el planificador para determinar la manera más eficiente de ejecutar una consulta.
- **table**. El nombre (o esquema-calificado) de una tabla especificada para ser usada por el **vacuum**. Por defecto se realiza a todas las tablas en la base de datos actual.
- **column**. El nombre de una columna específica a analizar. Por defecto son todas columnas.

Salida:

- Cuándo **VERBOSE** es especificado, el **VACUUM** emite mensajes sobre el progreso indicando cuál tabla se esta procesando actualmente. Varias estadísticas acerca de las tablas se despliegan también.

Notas:

- **VACUUM** no puede ser ejecutado dentro de un bloque de las transacciones.
- Se recomienda que en las bases de datos activas y en producción sea llevado a cabo el **VACUUM** con frecuencia (por lo menos cada noche), para quitar los registros caducados. Después de agregar o borrar un gran número de registros, informando con la instrucción **VACUUM ANALYZE** para la tabla afectada. Esto actualizará los catálogos de sistema con los resultados de todos los cambios recientes, y permitirá al planificador de consultas de PostgreSQL hacer mejores elecciones en la planificación de consultas.
- La opción **FULL** no se recomienda para un uso rutinario, pero puede ser útil en casos especiales. Un ejemplo es cuando se ha borrado la mayor parte de las filas en una tabla y se querría que la tabla encogiese físicamente para ocupar menos espacio de disco. **VACUUM FULL** encogerá generalmente la tabla más de lo que un **VACUUM** simple hace. La opción **FULL** no encoge los índices; un **REINDEX** periódico se recomienda. De hecho, es a menudo más rápido eliminar todos los índices, hacer el **VACUUM FULL**, y recrear los índices.
- El **VACUUM** causa un aumento substancial en el tráfico de E/S, que puede causar un desempeño pobre para otras sesiones activas. Por lo tanto, es a veces conveniente utilizar la característica de **cost-based vacuum delay**.
- PostgreSQL incluye una característica de "**autovacuum**" que puede automatizar la rutinaria de mantenimiento del **vacuum**.

En el Código 7-202 se muestra un ejemplo de **VACCUM** en una tabla en la base de datos de nombre **regression**.

```
regression=# VACUUM VERBOSE ANALYZE onek;
INFO:  vacuuming "public.onek"
INFO:  index "onek_unique1" now contains 1000 tuples in 14 pages
DETAIL:  3000 index tuples were removed.
0 index pages have been deleted, 0 are currently reusable.
CPU 0.01s/0.08u sec elapsed 0.18 sec.
INFO:  index "onek_unique2" now contains 1000 tuples in 16 pages
DETAIL:  3000 index tuples were removed.
0 index pages have been deleted, 0 are currently reusable.
CPU 0.00s/0.07u sec elapsed 0.23 sec.
INFO:  index "onek_hundred" now contains 1000 tuples in 13 pages
DETAIL:  3000 index tuples were removed.
0 index pages have been deleted, 0 are currently reusable.
CPU 0.01s/0.08u sec elapsed 0.17 sec.
INFO:  index "onek_stringu1" now contains 1000 tuples in 48 pages
DETAIL:  3000 index tuples were removed.
0 index pages have been deleted, 0 are currently reusable.
CPU 0.01s/0.09u sec elapsed 0.59 sec.
INFO:  "onek": removed 3000 tuples in 108 pages
DETAIL:  CPU 0.01s/0.06u sec elapsed 0.07 sec.
INFO:  "onek": found 3000 removable, 1000 nonremovable tuples in 143 pages
DETAIL:  0 dead tuples cannot be removed yet.
There were 0 unused item pointers.
0 pages are entirely empty.
CPU 0.07s/0.39u sec elapsed 1.56 sec.
INFO:  analyzing "public.onek"
INFO:  "onek": 36 pages, 1000 rows sampled, 1000 estimated total rows
VACUUM
```

Código 7-202. Uso de vacuum en PostgreSQL.

Compatibilidad:

- No existe una declaración de VACUUM en el estándar SQL.

Respaldos y restauración de información

Respaldos

La idea atrás del método **dump** es el generar un archivo de texto con comandos de SQL que, cuando se regresen al servidor, se recreará la base de datos en el mismo estado que tenía al realizar el **dump**. PostgreSQL provee el programa **pg_dump** donde la forma básica de uso es:

```
pg_dump dbname > outfile
```

Código 7-203. Sintaxis de pg_dump en PostgreSQL.

pg_dump escribe sus resultados en la salida estándar lo que puede ser útil.

pg_dump es una aplicación del cliente regular de PostgreSQL. Esto quiere decir que se puede realizar el proceso de respaldo desde cualquier host remoto que tenga acceso a la base de datos. Pero **pg_dump** no opera con permisos especiales. Particularmente, se debe tener acceso de lectura a todas las tablas que se deseen respaldar. En la práctica casi siempre se correrá como el súper usuario de la base de datos.

Para especificar a que servidor de base de datos **pg_dump** debe contactar, se usa el comando con la opción **-h** para el host y **-p** para el puerto. El host por defecto es **localhost** o cualquiera que este declarado en la variable de ambiente **PGHOST**. Similarmente, el puerto por defecto esta indicado por la variable de ambiente **PGPORT**.

Como cualquier otra aplicación cliente de PostgreSQL, **pg_dump** será por defecto conectado a la base de datos que tenga asignado el nombre de usuario y coincida con el nombre de usuario del sistema.

Para hacer caso omiso de esto la opción **-U** especifica la variable de ambiente de **PGUSER**. Con los mismos mecanismos de autentificación.

```
pg_dump databasename –D –U databaseowner > outfile-ddmmyyyy.psql
```

Código 7-204. Sintaxis general de uso de pg_dump en PostgreSQL.

Restaurando el respaldo

El archivo de texto creado por **pg_dump** esta destinado para ser leído por el programa **psql**. La instrucción general para restaurar un respaldo es:

```
psql dbname < infile
```

Código 7-205. Restaurando un respaldo con psql en PostgreSQL.

Donde **infile** es el archivo que se creó con la instrucción **pg_dump**. Pero la base de datos **dbname** no será creada con esta instrucción, así que debe ser creada antes de ejecutar **psql** (por ejemplo **createdb -T template0 dbname**). **psql** soporta opciones similares a **pg_dump**.

Antes de restaurar un respaldo SQL, todos los usuarios que tenían objetos o se les había dado permisos a objetos en el respaldo en la base de datos respaldada deben de existir. Si no están, entonces la restauración fallará en recrear los objetos con el dueño original y/o permisos.

Por defecto, el script **psql** continuará su ejecución hasta que un error en SQL sea encontrado. Puede desearse que se tenga otro comportamiento, usando la siguiente instrucción para alterar el comportamiento de salida del **psql** a un status 3 en la salida si un error en SQL ocurre.

```
\set ON_ERROR_STOP
```

Código 7-206. Modificación del comportamiento de salida de psql al ocurrir un error en PostgreSQL.

De cualquier forma, se tendrá solamente una restauración parcial. Alternativamente, se puede especificar que todo el proceso de restauración se tratado como una única transacción, así será realizada completamente o completamente **rolled back**. Este modo puede ser especificado pasándole la opción **-1** o la opción **–single-transaction** en la línea de comandos al ejecutar **psql**. Cuando se usa este modo, hasta los más pequeños errores puede realizar el **rollback** en una restauración que tenga haciéndose por varias horas. Sin embargo, esto sigue siendo preferible a que manualmente se limitada una compleja base de datos después de haber sido parcialmente restaurada.

La habilidad de **pg_dump** y de **psql** de escribir o leer desde técnicas de entubamiento del shell hace posible que el respaldo a una base de datos sea realizado desde una base de datos y directamente sea restaurado en otro servidor, por ejemplo:

```
pg_dump -h host1 dbname | psql -h host2 dbname
```

Código 7-207. Respaldando y restaurando una base de datos usando técnicas de entubamiento del shell de Unix.

Los respaldos producidos por **pg_dump** son relativos a la base **template0**. Esto significa que cualquier lenguaje, procedimiento, etc. adicionado a **template1** será también respaldado por **pg_dump**. Como un resultado, cuando se restaura, si se usa una base **template1** arreglada, se debe crear una base de datos vacía desde **template0** esto para el ejemplo anterior.

Después de restaurar un respaldo, será sabio en correr un **ANALYZE** en cada base de datos para que el optimizador de consultas tenga estadísticas útiles. Una forma fácil de hacer esto es correr **vaccumdb –a –z**, esto es equivalente de correr **VACUUM ANALYZE** en cada base de datos manualmente.

pg_dumpall

pg_dump descarga solamente una base de datos, y esto no obtiene información a cerca de roles o tablespaces (porque esos son cluster lejos de ser bases de datos). Para soportar una conveniente obtención de todos los contenidos del cluster de base de datos, el programa **pg_dumpall** es provisto. **pg_dumpall** respalda cada base de datos en un cluster dado, y también preserva los datos del cluster así como definiciones de tablespace. El uso básico de este comando es:

```
pg_dumpall > outfile
```

Código 7-208. Sintaxis de pg_dumpall en PostgreSQL.

El archivo resultante puede ser restaurado con **psql**:

```
psql -f infile postgres
```

Código 7-209. Restauración de un pg_dumpall en PostgreSQL.

7.3 SCRIPTS DE INSTALACIÓN DE SYBASE. CASO PRÁCTICO

Al administrador de bases de datos de una compañía que tiene Sybase se le encarga tener los scripts que definen la instalación de Sybase, de la creación del servidor de bases de datos y de la creación de las 3 bases de datos usadas en la empresa.

```
#rpm –ihv sybase-common-11.9.2-3.i386.rpm
#rpm –ihv sybase-ase-11.9.2-3.i386.rpm
```

Código 7-210.Instalar Sybase.

```
#passwd sybase
#su – sybase
sybase#cd /opt/sybase-11.9.2
sybase#mkdir dev
sybase#mkdir logs
sybase#cd /opt/sybase-11.9.2/bin
sybase#export SYBASE=/opt/sybase-11.9.2
sybase#./srvbuild                              (configurar en modo gráfico)
sybase#./srvbuildtext                          (configurar en modo texto)
sybase#./srvbuildres –r [archivo-fuente]       (configurar desde un archivo fuente)
```

Código 7-211. Crear un servidor de bases de datos de Sybase.

```
sybinit.release_directory: USE_DEFAULT
sybinit.product: sqlsrv
sqlsrv.server_name: SQLSRV
sqlsrv.new_config: yes
sqlsrv.do_add_server: yes
sqlsrv.network_protocol_list: tcp
sqlsrv.network_hostname_list: localhost
sqlsrv.network_port_list: 8100
sqlsrv.master_device_physical_name: /opt/sybase-11.9.2/dev/SQL_master.dat
sqlsrv.master_device_size: 4000
sqlsrv.master_database_size: 1000
sqlsrv.errorlog: /opt/sybase-11.9.2/logs/SQLSRV.log
sqlsrv.do_upgrade: no
sqlsrv.sybsystemprocs_device_physical_name: /opt/sybase-11.9.2/dev/SQL_sysproc.dat
sqlsrv.sybsystemprocs_device_size: 60
sqlsrv.sybsystemprocs_database_size: 60
sqlsrv.sybsystemdb_device_size: USE_DEFAULT
sqlsrv.sybsystemdb_database_size: USE_DEFAULT
sqlsrv.default_backup_server: SQLSRVBACKUP
```

Código 7-212. Archivo fuente para configurar un servidor de bases de datos de Sybase usando ./srvbuildres –r.

```
sybase#cd install/
sybase#./startserver -f ./RUN_SQLSRV
```

Código 7-212. Arrancar servidor de Sybase.

```
sybase#isql -Usa -S SQLSRV -w 200
Password:              (La primera vez no pide password teclear ENTER)
1> quit                (Salir del cliente isql)
```

Código 7-213.Ejecutar cliente isql.

```
sybase#su -
#gunzip sqsh-2.1-linux-11.1.1.tar.gz
#tar –xvf sqsh-2.1-linux-11.1.1.tar
#SYBASE=/opt/sybase-11.9.2/
#export SYBASE
#./configure
#make
#make install
#cd sqsh-2.1
#cp sqsh /usr/local/bin
#cd /opt/sybase-11.9.2
#chown –R sybase:sybase *
```

Código 7-214.Instalar sqsh[9]

```
sybase#sqsh -Usa -S SQLSRV -w 200
Password:              (La primera vez no pide password teclear ENTER)
1> quit                (Salir del cliente isql)
```

Código 7-215.Ejecutar cliente isql.

```
sybase#isql -Usa -S SQLSRV -w 200
1> sp_help             (Muestra elementos de la base de datos en que estemos ubicados)
2> go
1> sp_helpdb           (Muestra las bases de datos)
2> go
1> sp_helpdb [base]    (Muestra información más detallada de [base])
2> go
```

Código 7-216.Comando sp_help y sp_helpdb en cliente isql.

```
1> disk init
2> name="dev_dat1"                                    (Nombre para el dispositivo)
3> physname= "/opt/sybase-11.9.2/dev/dev_dat1",
4> vdevno=2,                                          (Número de dispositivo)
5> size=512000                                  (1000MB=1024000 KB=1024000/2=512000 páginas)
6> go
1> disk init
2> name="dev_log1",
3> physname="/opt/sybase-11.9.2/dev/dev_log1",
4> vdevno=3,
5> size=512000
6> go
```

Código 7-217. Creación de dos dispositivos de 100MB.

[9] Sqsh es una utileria de código abierto que es usado por los administradores de Sybase como sustituto al isql proporcionado por Sybase ASE. Es un cliente interactivo de consultas SQL que proporciona mayores características que el isql.

```
1> create database db0
2> go
CREATE DATABASE: allocating 1024 pages on disk 'master'
1> create database db1 on dev_dat1=200
2> go
CREATE DATABASE: allocating 102400 pages on disk 'dev_dat1'
1> create database db2 on dev_dat1=200 log on dev_log1=100
2> go
CREATE DATABASE: allocating 102400 pages on disk 'dev_dat1'
CREATE DATABASE: allocating 51200 pages on disk 'dev_log1'
```

Código 7-218. Creación de tres bases de datos.

Capítulo 8

BUENAS PRÁCTICAS EN LA FUNCIÓN DE LA ADMINISTRACIÓN

Fundamentos de la auditoria informática y controles, en el ámbito de la administración de bases de datos, así como las mejores prácticas a seguir en la administración de bases de datos son buenas prácticas en la función de la administración de bases de datos.

8.1. CONCEPTO DE AUDITORIA INFORMÁTICA

La informática hoy, está inmersa en la gestión integral de la organización.

A finales del siglo XX, los Sistemas de TIC han constituido las herramientas más poderosas para cualquier organización, puesto que apoyan la toma de decisiones, generando un alto grado de dependencia, así como una elevada inversión en TIC.

Debido a la importancia que tienen los Sistemas de TIC en el funcionamiento de una organización, existe la auditoria informática.

Los auditores han sido catalogados a través del tiempo como personajes siniestros que se dedican a identificar todo lo que esté mal, para denunciarlo y alertar a quien deba ser alertado. En general, el rol es percibido como una especie de representante de la inquisición dentro de la organización.

Conforme han avanzando las teorías de administración de empresas, el papel y la percepción de la auditoria en las organizaciones fue cambiando, aunque lamentablemente, no con la velocidad necesaria.

Hoy día, debemos pensar en el auditor como un elemento imprescindible para una sana operación de las instituciones. Su papel ha pasado de ser un detector de problemas, a un agente de cambio, identificador de oportunidades y emisor de propuestas de valor, su compromiso profesional va más allá de fungir como un mecanismo detectivo.

Actualmente es un asesor de negocios que brinda soluciones adecuadas al entorno y la situación interna de la organización con el fin de que ésta logre sus objetivos estratégicos.

Al igual que las demás áreas de la organización, las bases de datos deben estar sometidos a controles, por las siguientes razones: las computadoras y los centros de procesamiento de datos son blancos apetecibles para el espionaje, la delincuencia y el terrorismo. Al perder de vista la naturaleza y calidad de los datos de entrada a los Sistemas de TIC se genera información errónea, con la posibilidad de que se provoque un efecto cascada y afecte a otras aplicaciones. Un Sistema de TIC mal diseñado puede convertirse en una herramienta muy peligrosa para la gestión y la organización de la organización.

La auditoria informática puede ser definida como:

"Un proceso evolutivo que mediante técnicas y procedimientos aplicados en una organización por personal independiente a la operación de la misma, evalúa la función de tecnología de información y su aportación al cumplimiento de los objetivos institucionales; emite una opinión al respecto y efectúa recomendaciones para mejorar el nivel de apoyo al cumplimiento de dichos objetivos".

Beneficios de la auditoria informática

Mejora la imagen pública.

Es una manifestación ante los involucrados de la organización (propietarios, clientes, proveedores, empleados, dependencias normativas) de que hay una ocupación constante en mejorar la gestión de los recursos de la organización.

Generación de confianza en los usuarios sobre la seguridad y control de los servicios de TIC.

Uno de los atributos de la información que dependen de la percepción del usuario es su confiabilidad; el usuario tendrá confianza en la información en la medida en que confíe en la fuente de información y la infraestructura relacionada.

Optimiza las relaciones internas y del clima de trabajo.

Promueve que sean claras y compatibles las actividades de los roles involucrados en la generación, transmisión, resguardo/destrucción de la información permitiendo que cada uno de los participantes sepa qué hacer y con quien debe interactuar, evitando conflictos en los equipos de trabajo.

Disminución de costos de la mala calidad (reprocesos, rechazos, reclamos, entre otros).

Al observar a la información y la tecnología relacionada como activos que deben ser administrados en una serie de procesos, las relaciones internas deben definirse y las características que deben guardar dichos procesos deben establecerse, por tanto, existe menor probabilidad de que hayan rechazos, reprocesos o reclamos.

Balance de los riesgos en TIC.

La información y la tecnología relacionada con ella están expuestas a una serie de riesgos que deben ser mitigados, transferidos, aceptados o rechazados con conocimiento de causa. El balance de los riesgos estará en función del apetito de riesgo que tenga el tomador de decisiones.

Control de la inversión en un entorno de TIC a menudo impredecible.

La organización debe administrar la información que necesita para el logro de sus objetivos y para adaptarse al entorno cambiante; por tanto, las inversiones en la tecnología deben adecuarse a esta situación evitando utilizar recursos en tecnología que no aportan de manera sustancial a la organización o descuidar inversiones por una austeridad extrema no justificada. En resumen, las inversiones deben responder a un análisis de costo beneficio para la organización.

Proceso de auditoria

La metodología que se emplea en la Auditoria Informática es similar a las fases que componen una auditoria tradicional:

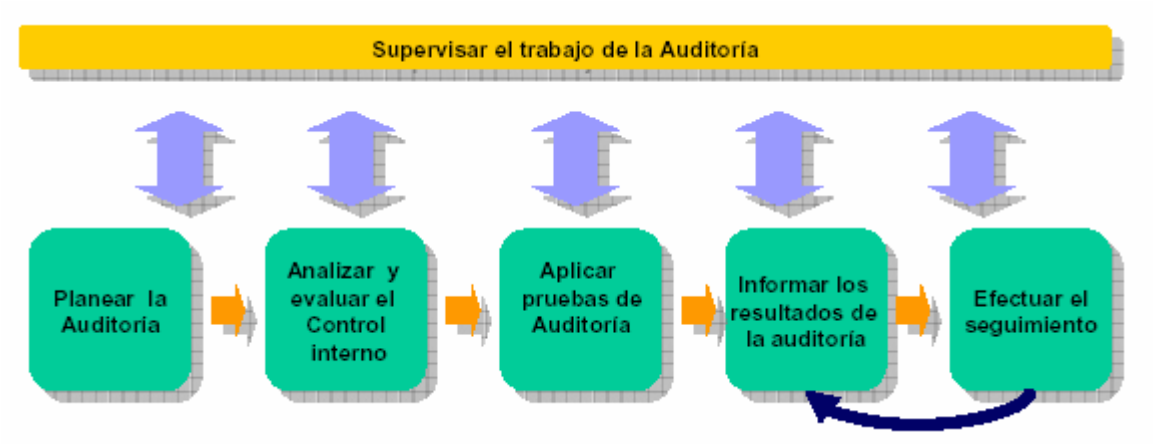


Ilustración 8-1. Metodología empleada en la Auditoria.

8.2. ROLES Y RESPONSABILIDADES

Las estructuras organizacionales (organigramas) en las áreas de TIC difieren de organización a organización; sin embargo, en cualquiera de ellas son un elemento importante para que todos los empleados tengan un conocimiento claro de la responsabilidad y autoridad de cada uno de sus integrantes.

Adicionalmente las descripciones de puestos proporcionan a los integrantes del departamento de Sistemas de Información una clara definición de sus roles y responsabilidades.

En los siguientes apartados se describe de forma puntual aquellas funciones generales asignadas a los administradores de bases de datos.

Funciones del administrador de bases de datos

De acuerdo al manual de revisión de los Certified Information Systems Auditor (CISA) las responsabilidades del administrador de bases de datos son:

1. Custodia información de la organización.
2. Define y mantiene la estructura de los datos en el sistema corporativo de BD.
3. Debe comprender a la empresa, a los datos de usuario y las relaciones de éstos.
4. Responsable de la seguridad y clasificación de la información de los datos compartidos, almacenados en los sistemas de BD.
5. Responsable del diseño real, definición y mantenimiento de las BD corporativas.
6. Especificar la definición física de los datos y cambiarla para su mejor desempeño.
7. Seleccionar e implementar herramientas de optimización de la BD.
8. Probar y evaluar las herramientas de programadores.

9. Dar soporte técnico a programadores sobre estructura de la BD.

10. Implementar controles de definición, acceso, actualización y concurrencia.

11. Monitorear el uso, recopilar estadísticas de desempeño y ajustar la BD.

12. Definir e iniciar los procedimientos de respaldo y recuperación.

Segregación de funciones

A continuación se presenta un ejemplo de la organización de un departamento de sistemas de información (Puestos y organigrama pueden variar entre empresas).

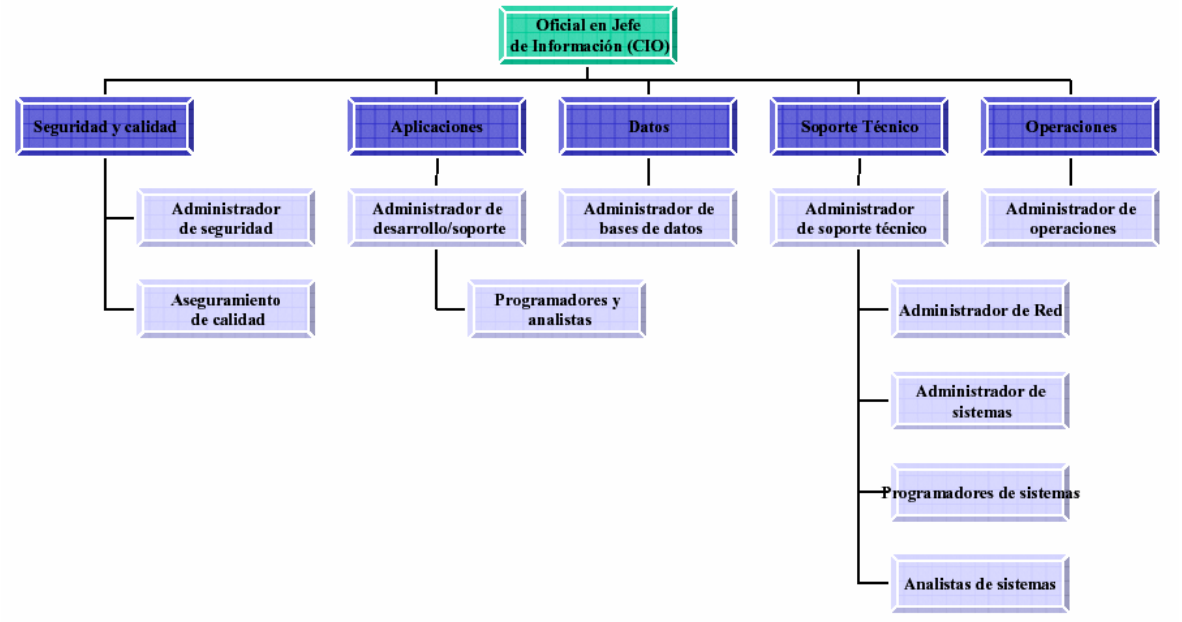


Ilustración 8-2. Organización de un departamento de sistemas de información.

En la segregación de funciones el auditor deberá determinar la relación entre las funciones, responsabilidad y autoridad.

Una adecuada segregación de funciones:

1. Evita que una sola persona pueda ser responsable de funciones diversas y críticas.

2. Evita que se cometan errores o apropiaciones indebidas difíciles de detectar.

3. Previene y disuade actos fraudulentos o maliciosos.

4. Puede restringir acceso a: la computadora, Biblioteca de datos de producción, Programas de producción, Documentación de programas, Sistema Operativo y utilerías.

5. Reduce el daño potencial por acciones de personas.

En empresas pequeñas, debe haber controles compensatorios, para mitigar el riesgo de no tener esta segregación. Control Compensatorio en un control interno que reduce el riesgo de una debilidad de control.

	Gpo Cnt	Ana Sis	Prog Apl	Help Desk	Usua Fin	Ing Dat	Oper Com	Adm BD	Adm Red	Adm Sis	Adm Seg	Cint	Prog Sis	Cnt Cal
Gpo Cnt		X	X	X		X	X	X	X	X			X	
Ana Sis	X			X	X		X				X	X		
Prog Apl	X			X	X	X	X	X	X	X	X	X	X	
Help Desk	X	X	X		X	X		X	X	X		X	X	
Usua Fin		X	X	X			X	X	X			X	X	X
Ing Dat	X		X	X			X	X	X	X	X		X	
Oper Com	X	X	X		X	X		X	X	X	X		X	
Adm BD	X		X	X	X	X	X		X	X			X	
Adm Red	X		X	X	X	X	X	X				X		
Adm Sis	X		X	X		X	X	X				X		
Adm Seg		X	X			X	X					X	X	
Cint		X	X	X	X				X	X	X		X	
Prog Sis	X		X	X	X	X	X	X			X	X		X
Cnt Cal					X								X	

X = Combinación de estas funciones puede crear una debilidad potencial de control

Esta matriz de control es sólo una guía

Ilustración 8-3. Diagrama de control en la segregación de funciones.

Controles para el reforzamiento de la funciones

Pueden usarse diversos mecanismos para el reforzamiento funciones:

- Acceso a datos.
- Autorización de transacciones.
- Custodia de activos.
- Formas de autorización.
- Tablas o registros de autorización de usuarios.

Autorización de transacciones

La autorización de transacciones es responsabilidad de los usuarios. La autorización es delegada al grado que sea relativo al nivel particular de responsabilidad del individuo autorizado en el departamento de usuario. Verificaciones periódicas deben ser desempeñadas por la administración y por auditoria para detectar la entrada autorizada de transacciones.

Custodia de activos

La custodia de activos corporativos, éstos deben ser determinados y asignados apropiadamente. El dueño de los datos es asignado a un usuario de departamento en particular, y sus obligaciones deberían ser especificadas por escrito. El propietario de los datos tiene la responsabilidad para determinar los niveles de autorización requeridos para proveer seguridad adecuada, mientras que el grupo de

administración es frecuentemente responsable de la implementación y reforzamiento del sistema de seguridad.

Acceso a datos

Controles sobre acceso a datos son provistos por una combinación de seguridad física, de sistema y de aplicación en el área del usuario y en el centro de procesamiento de datos.

El ambiente físico debe ser asegurado para prevenir el acceso no autorizado de personal a los diversos dispositivos tangibles conectados a la unidad central de procesamiento, y por lo tanto, prevenir el acceso a los datos.

La seguridad en los sistemas y aplicaciones son capas adicionales de seguridad que pueden prever que individuos no autorizados obtengan acceso a datos de la organización.

Acceso a datos desde conexiones externas es una preocupación creciente desde el advenimiento de la Internet.

Las decisiones de control de acceso están basadas en políticas organizacionales y en dos estándares aceptados en la práctica: la separación de funciones y el menor privilegio.

Los controles para que se usen efectivamente no deben interrumpir más de lo necesario el flujo usual de trabajo o establecer demasiada carga a los administradores, auditores y usuarios autorizados. Aún más, el acceso no debe ser incondicional y los controles de acceso deben proteger adecuadamente todos los recursos organizacionales; para asegurar esto es necesario primero categorizar los recursos.

Las políticas establecen niveles de sensibilidad para los datos y para los recursos tales como: ultra secreto, secreto, confidencial y no clasificado. Estos niveles deberían ser usados como guía en los procedimientos adecuados para el manejo de los activos de información. Esto también puede ser tomado en cuenta como base para el control de acceso. A los individuos sólo se les garantiza el acceso a recursos específicos o al nivel más bajo de sensibilidad.

Las etiquetas se usan para indicar el nivel de sensibilidad de documentos almacenados electrónicamente. Los controles basados en políticas pueden ser mandatarios o discrecionales.

Formas de autorización

Los administradores de los departamentos usuarios deben proveer al Sistema de Información de formas de autorización formal que definan los derechos de acceso a cada uno de sus colaboradores; en otras palabras, los responsables de los usuarios deben definir quién tendría acceso a qué.

Las formas de autorización deben ser evidenciadas adecuadamente con el nivel de aprobación de la administración. De manera general, todos los usuarios deberían estar autorizados con acceso específico a los sistemas vía solicitud formal de la administración.

En compañías muy grandes o en aquellas con sitios remotos, deberían mantenerse catálogos de firmas de autorización y las solicitudes formales deberían ser comparadas con dicho catálogo.

Los derechos de acceso (privilegios) deben ser revisados periódicamente para asegurar que están actualizados y son adecuados a las funciones del trabajo del empleado.

Tablas o registros de autorización de usuarios

El departamento de Sistemas de Información debería usar los datos de las formas de autorización para construir y mantener tablas de autorización de usuarios. Estas tablas indicarán quiénes están autorizados para actualizar, modificar, borrar y/o ver datos. Estos privilegios se proveen a nivel sistema, transacción o campo; en efecto, estas son las listas de control de acceso.

Las tablas de autorización deben ser aseguradas contra acceso no autorizado mediante protección adicional con contraseñas o encriptación de datos.

La bitácora de control debe registrar toda la actividad del usuario y el nivel apropiado de la administración debería revisarla periódicamente; todas las excepciones deben ser investigadas.

Controles compensatorios por falta de segregación de funciones

En organizaciones muy pequeñas en donde el departamento de Sistemas de Información consiste en cuatro o cinco personas deben existir medidas de control compensatorio para mitigar el riesgo resultante de la falta de la segregación de funciones. Algunos de los controles compensatorios son:

- Pistas de auditoria.
- Conciliación.
- Reportes de excepción.
- Registros de transacciones.
- Revisiones de supervisión.
- Revisiones independientes

Registros de auditoria.

Son un componente esencial en todos los sistemas bien diseñados. Éstas ayudan al departamento de Sistemas de Información y al de usuario a proveer un mapa para rastrear el flujo de una transacción. Esto permite al usuario y al auditor a recrear el flujo de transacción actual desde su punto de origen hasta su existencia o la actualización de un archivo.

En la ausencia de una adecuada segregación de funciones, los registros de auditoria pueden ser aceptados como un control compensatorio aceptable. El revisor debería ser capaz de determinar quien inicio la transacción, la hora del día y la fecha de ingreso, el tipo de ingreso, qué campos de información contenía y qué archivos actualizaba.

Conciliación.

Es responsabilidad última del usuario. En algunas organizaciones, la conciliación limitada de aplicaciones puede ser ejecutada por el grupo de control de datos con el uso de totales de control y hojas de balance. Este tipo de verificación independiente incrementa el nivel de confianza que la aplicación corrió adecuadamente y que los datos fueron adecuadamente balanceados.

Reportes de excepción

Deben ser manejados a nivel supervisión y deberían requerir evidencias, tales como iniciales en un reporte subrayando que la excepción fue tratada adecuadamente. La administración debería asegurarse que las excepciones son resueltas de forma oportuna.

Bitácoras de transacción

Pueden ser manuales o automáticas. Un ejemplo de bitácora manual es el registro de transacciones en batch antes de que sean enviadas a procesamiento. Una bitácora de transacción automática provee un registro de todas las transacciones procesadas y es mantenida por el sistema operativo.

Revisiones de supervisión

Pueden ser ejecutadas a través de la observación y preguntas o remotamente.

Revisiones independientes

Se llevan a cabo para compensar los errores o fallas intencionales al seguir los procedimientos prescritos. Estas son particularmente importantes cuando las obligaciones en organizaciones muy pequeñas no pueden ser apropiadamente segregadas. Dichas revisiones ayudarán a detectar errores o irregularidades.

8.3. ISACA, OBJETIVOS DE CONTROL, COBIT

ISACA

¿Cómo podrían asegurar las organizaciones, que construyen proyectos de tecnología de información, que están cumpliendo adecuadamente con las necesidades del cliente, en forma eficiente y oportuna y dentro del presupuesto contemplado?

Existe una asociación internacional denominada Information Systems Audit and Control Association (ISACA); comenzó en 1967 con un grupo pequeño de profesionales con actividades similares –verificar controles en los sistemas de cómputo que se estaban convirtiendo en parte crítica en las operaciones de sus organizaciones. En 1969 se conformaron como la asociación de auditores EDP con el fin de cubrir las necesidades únicas, diversas y de alta tecnología en el naciente campo de la TIC.

La misión de ISACA consiste en mejorar el reconocimiento de la profesión de auditoria y control de las TIC a través de la elaboración de materiales y marcos de trabajo, así como capacitación y certificación de sus miembros a través de la fundación (Information Systems Audit and Control Foundation).

En la actualidad cuenta con más de 50,000 miembros en más de 140 países; cuenta con 170 capítulos establecidos en más de 60 países.

Ilustración 8-4. ISACA en Internet http://www.isaca.org.

Objetivos de control

Los activos de información (información, aplicaciones, infraestructura y gente) están sometidos a una serie de amenazas del entorno. Dichas amenazas son valoradas por la administración de la organización para establecer salvaguardas o controles.

Las debilidades en los controles son conocidas como vulnerabilidades. De estos conceptos se desprenden el análisis y la administración de riesgos. De acuerdo con las guías para la administración de la seguridad de TIC un riesgo es "El potencial de que una amenaza dada explote vulnerabilidades de un activo o grupos de activos para causar pérdidas o daños a tales activos. El impacto o la severidad relativa del riesgo es proporcional al valor del negocio de la pérdida o daño y a la frecuencia estimada de la amenaza".

Ilustración 8-5. Decisiones a afrontar frente al riesgo en TIC.

El control es una medida para reducir los riesgos; se materializan en las políticas, procedimientos, prácticas y estructuras organizacionales establecidas por la organización para lograr sus objetivos y para que los eventos no deseados serán prevenidos o detectados y corregidos.

Ilustración 8-6. Diagrama de amenazas, tipos de riesgo e impacto en la TIC.

La clasificación de los controles según su naturaleza es:

Clase	Función	Ejemplos
Preventivo	Detectar problemas antes de que surjan Monitorear las operaciones y las entradas Intentar predecir problemas potenciales antes de que ocurran y se hagan ajustes Prevenir que ocurran errores, omisiones o actos maliciosos	Emplear sólo personal calificado Segregar funciones Controlar el acceso a instalaciones físicas Usar documentos bien diseñados Establecer procedimientos para la autorización de transacciones
Detectivo	Usar controles que detecten y reporten la ocurrencia de un error, omisión o acto malicioso	Totales hash Puntos de verificación en los Jobs de producción Verificación duplicada de cálculos Funciones de auditoria interna
Correctivo	Minimizar el impacto de una amenaza Remediar los problemas descubiertos por controles detectivos Identificar las causas de los problemas Corregir errores que surjan de los problemas Modificar los sistemas de procesamiento para minimizar las ocurrencias del problema en el futuro	Planes de contingencia Procedimientos de respaldo Procedimientos de recorrida

Tabla 8-1. Clasificación de los controles según su naturaleza.

Los objetivos de control (procesos de control) son materializados por los controles específicos. En otras palabras un objetivo de control es una declaración de un propósito o resultado deseable a ser alcanzado mediante la implementación de prácticas de control en una actividad de TIC en particular.

Control Objectives for Information and Related Technology (COBIT)

¿Cuáles son marcos de trabajo para verificar que los sistemas alcanzan la efectividad y eficiencia esperadas?

Un marco de trabajo conocido como Objetivos de Control para la Información y la Tecnología Relacionada (Control Objectives for Information and related Technology-COBIT®) sirve como guía para la buena práctica de la auditoria de las TIC, emitido por el IT Governance Institute. COBIT es un marco de control o sistema de control interno que tiene el objetivo de que las TIC 's sean exitosas en satisfacer los requerimientos de la organización.

COBIT es un marco de control generalmente aplicable y aceptado internacionalmente como buena práctica para controles de TIC.

Provee una serie de buenas prácticas a través de un marco de dominio y procesos y presenta actividades en una estructura lógica y manejable.

Las buenas prácticas de COBIT:

- Representan el consenso de expertos fuertemente enfocados en el control y menos en la ejecución.
- Ayudan a optimizar las inversiones en TIC , aseguran la entrega de servicios y proveen una medida en contra de la cual juzgar cuando las cosas van mal.

Divide la TIC en 34 procesos que se integran en 4 dominios y proporciona un objetivo de control de alto nivel para cada uno. Se soporta por una serie de 318 objetivos de control detallados.

- Planeación y Organización.
- Adquisición e Implementación.
- Entrega y Soporte.
- Monitoreo y Evaluación

Considera las necesidades fiduciarias, de calidad y de seguridad de una organización proveyendo 7 criterios de información que pueden ser usados para genéricamente definir qué requiere de TIC la organización.

- Eficacia.
- Eficiencia.
- Disponibilidad.
- Integridad.
- Confidencialidad.
- Confiabilidad.
- Cumplimiento

Características de COBIT

COBIT cubre cuatro características básicas:

Ilustración 8-7. Características básicas de COBIT.

Para proveer la información que la organización necesita para lograr sus objetivos, los recursos de TIC necesitan ser administrados por una serie de procesos naturalmente agrupados.

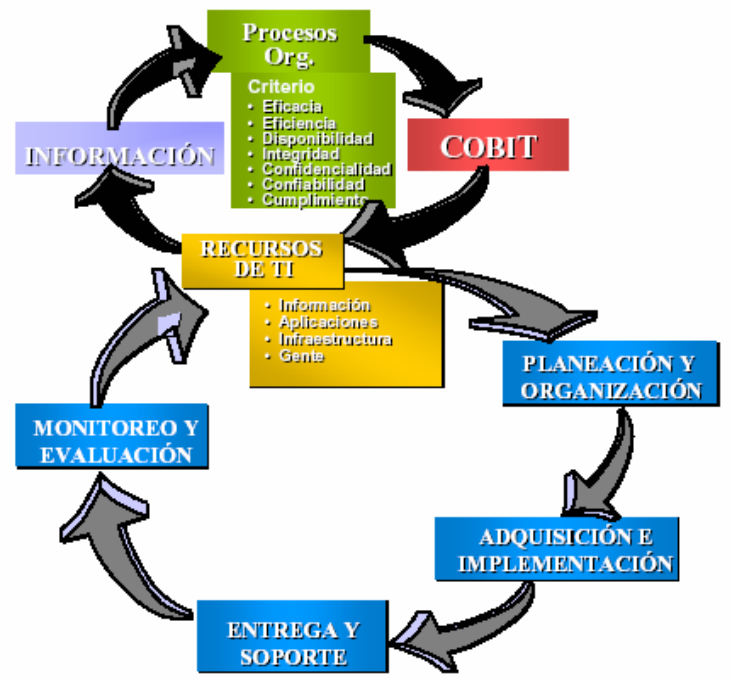


Ilustración 8-8. Proceso de organización ideal en la TIC.

La estructura documental de COBIT respecto de las áreas y niveles de organización puede verse representada en la siguiente imagen:

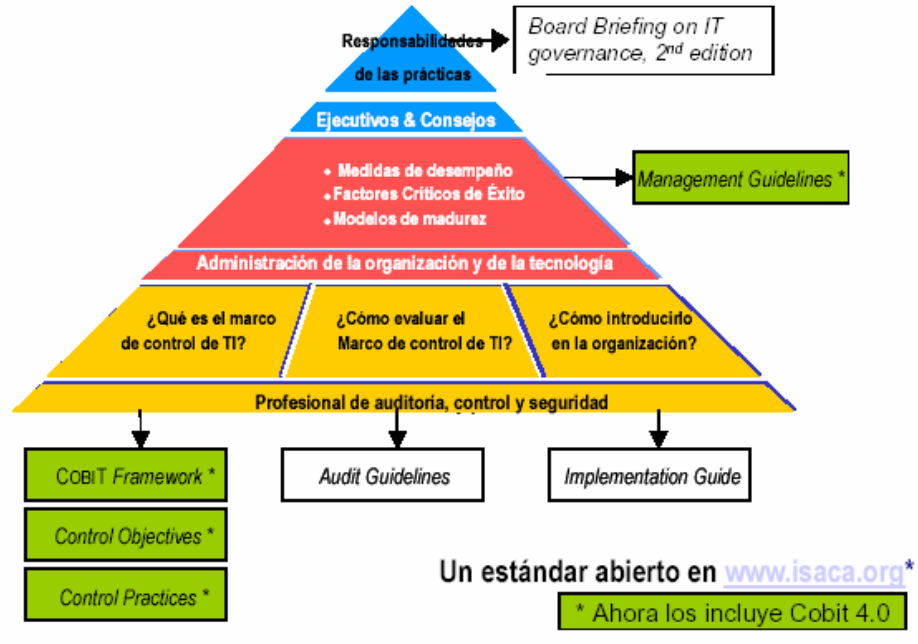


Ilustración 8-9. Estructura documental de COBIT respecto de las áreas y niveles de organización.

La forma de operar de COBIT puede verse representada en la siguiente imagen:

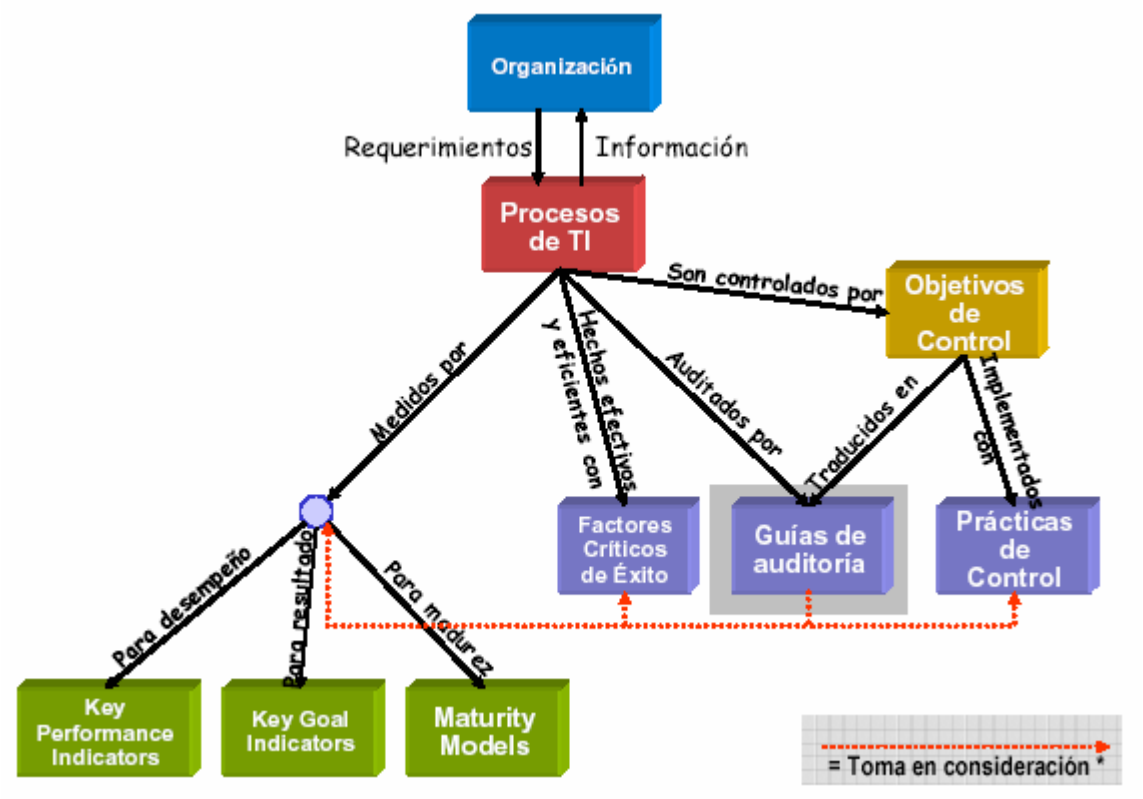


Ilustración 8-10. Forma de operar de COBIT.

Para gobernar efectivamente TIC, es importante determinar las actividades y los riesgos que requieren ser administrados. Éstos se pueden resumir en los siguientes objetivos de control:

PLANEAR Y ORGANIZAR

PO1 Planeación estratégica de TIC

PO2 Definir la arquitectura de la información

PO3 Determinar la dirección tecnológica

PO4 Definir los procesos, organización y relaciones de TIC

PO5 Administrar la inversión en TIC

PO6 Comunicar las aspiraciones y la dirección de la gerencia

PO7 Administrar recursos humanos de TIC

PO8 Administrar la calidad PO9 Evaluar y administrar los riesgos de TIC

PO10 Administrar proyectos

ADQUIRIR E IMPLANTAR

AI1 Identificar soluciones automatizadas

AI2 Adquirir y mantener software aplicativo

AI3 Adquirir y mantener infraestructura tecnológica

AI4 Facilitar la operación y el uso

AI5 Adquirir recursos de TIC

AI6 Administrar cambios

AI7 Instalar y acreditar soluciones y cambios

ENTREGAR Y DAR SOPORTE

DS1 Definir y administrar los niveles de servicio

DS2 Administrar los servicios de terceros

DS3 Administrar el desempeño y la capacidad

DS4 Garantizar la continuidad del servicio

DS5 Garantizar la seguridad de los sistemas

DS6 Identificar y asignar costos

DS7 Educar y entrenar a los usuarios

DS8 Administrar la mesa de servicio y los incidentes

DS9 Administrar la configuración

DS10 Administrar los problemas

DS11 Administrar los datos

DS12 Administrar el ambiente físico

DS13 Administrar las operaciones

MONITOREAR Y EVALUAR

ME1 Monitorear y evaluar el desempeño de TIC

ME2 Monitorear y evaluar el control interno

ME3 Garantizar el cumplimiento regulatorio

ME4 Proporcionar gobierno de TIC

8.4 OBJETIVOS DE CONTROL RELACIONADOS CON BASES DE DATOS

Como se ha señalado COBIT tiene 34 objetivos de control agrupados en cuatro dominios y aunque están estrechamente relacionados, la intención es identificar aquéllos que tienen mayor relación con los Administradores de Bases de Datos y las funciones que éstos realizan.

Definir la arquitectura de la información

PO2 Definir la arquitectura de la información. La función de sistemas de información debería crear y regularmente actualizar un modelo de información de la organización y definir los sistemas apropiados para optimizar el uso de información.

Esto incluye el desarrollo de un diccionario de datos corporativo con las reglas de sintaxis de datos, esquema de clasificación de datos y niveles de seguridad de la organización. Este proceso incrementa la calidad de la toma de decisiones de la administración teniendo la confianza que la información provista es confiable y segura, así mismo, esto permite racionalizar los recursos de los sistemas de información para alcanzar apropiadamente las estrategias organizacionales.

Este proceso de TIC es importante y necesario para incrementar la responsabilidad de la integridad y seguridad de los datos e incrementar la efectividad y control de compartir información a través de aplicaciones y entidades.

Los atributos de información que trata este proceso son: eficiencia, integridad, confidencialidad y efectividad.

Este objetivo satisface el requerimiento de la organización hacia las TIC para que sean ágiles en responder a los requerimientos, para proveer información confiable y consistente, así como, para hacer consistentes las aplicaciones integradas dentro de los procesos de la organización.

La definición de la arquitectura de la información se enfoca en el establecimiento de un modelo de datos organizacional que incorpora un esquema de clasificación de datos para asegurar la integridad y consistencia de todos los datos.

La definición de la arquitectura de información se logra:

1. Asegurando la precisión de la arquitectura de información y modelo de datos.

2. Asignando la propiedad de los datos.

3. Clasificando información usando un esquema de clasificación acordado.

La definición de la arquitectura de información se mide:

1. Porcentaje de elementos de datos duplicados/redundantes.

2. Porcentaje de aplicaciones que no cumplen con la arquitectura de información.

3. Frecuencia de las actividades de validación

Los recursos de información relacionados son la información y las aplicaciones.

Objetivos de control detallados

Modelo de arquitectura de información de la organización

Establecer y mantener un modelo de información de la organización para habilitar el desarrollo de aplicaciones y actividades de soporte de decisiones, consistentes con los planes de TIC. El modelo facilita la creación óptima, uso y compartimiento de información por la organización y un modo de mantener integridad y es flexible, funcional, efectivo en costo, en tiempo seguro y resistente a fallas.

Diccionario de datos de la organización y reglas de sintaxis de datos

Mantener un diccionario de datos organizacionales que incorpore las reglas de sintaxis de datos de la organización. El diccionario habilita el compartimiento de elementos de datos entre las aplicaciones y los sistemas, promueve un entendimiento común de datos entre TIC y los usuarios de negocio y previene que sean creados elementos de datos incompatibles.

Esquema de clasificación de datos

Establecer el esquema de clasificación que aplica a través de la organización, basadas en la criticidad y sensibilidad (por ejemplo, público, confidencial y ultra secreto) de los datos de organización. Este esquema incluye detalles acerca de la propiedad de datos, definición de niveles de seguridad apropiados y controles de aplicación y una breve descripción de los requerimientos de retención y destrucción de datos, criticidad y sensibilidad. Es usado como bases para aplicar controles tales como controles de acceso, archivo o encriptación.

Administración de integridad

Definir e implementar procedimientos para asegurar integridad y consistencia de todos los datos almacenados en forma electrónica tales como bases de datos, almacenes de datos y archivos de datos.

Entradas, salidas y actividades

Entradas		Salidas	
PO1	Planes estratégicos y tácticos	Esquema de clasificación de datos	AI1
AI1	Requerimientos del negocio Estudios de factibilidad	Plan de sistemas de negocio optimizado	PO3, AI2
AI7	Revisión de post implementación	Diccionario de datos	AI2, DS1 1
DS3	Información de capacidad y desempeño	Arquitectura de la información	PO3, DS5
ME1	Entradas de desempeño para la planeación de TIC	Clasificaciones de datos asignadas	DS1, DS4, DS5, DS11, DS12
		Clasificación de procedimientos y herramientas	*

Tabla 8-2. Entradas, salidas y actividades.

Actividades	CEO	CFO	Ejecutivo de negocio	CIO	Dueño del proceso de negocio	Jefe de operaciones	Arquitecto en jefe	Jefe de desarrollo	Jefe de administración de TIC	PMO	Cumplimiento, auditoria, riesgos y seguridad
Crear y mantener el modelo de información		C	I	A	C		R	C	C		C
Crear y mantener el o los diccionarios de datos				I	C		A/R	R			C
Establecer y mantener un esquema de clasificación de datos	I	C	A	C	C	I	C	C			R
Proveer a los dueños de los datos de procedimientos y herramientas para clasificar los sistemas de información	I	C	A	C	C	I	C	C			R
Utilizar el modelo de información, diccionario de datos y el esquema de clasificación para planear sistemas de negocio optimizados	C	C	I	A	C		R	C			I

A = Autoridad, R= Responsabilidad, C = Consultado, I = Informado

Tabla 8-3. Actividades y asignación de funciones del personal en la Arquitectura de Información.

Modelo de madurez

Nivel de Madurez	Arquitectura de información
0 Inexistente	* No hay conciencia de la importancia de la arquitectura de la información para la organización. * El conocimiento, la experiencia y las responsabilidades necesarios para desarrollar esta arquitectura no existen en la organización.
1 Inicial /Ad hoc	*La administración reconoce la necesidad de una arquitectura de la información, pero no ha formalizado ni un proceso ni un plan para desarrollar una. *Se lleva a cabo un desarrollo aislado y reactivo de componentes de una arquitectura de organización. *Hay implementaciones aisladas y parciales de diagramas de datos, documentación, y reglas de sintaxis de datos. Las definiciones se refieren a datos en lugar de información, y están impulsadas por ofertas del vendedor de software de aplicación. *Hay comunicación inconsistente y esporádica de la necesidad de una arquitectura de la información.
2 Repetible pero Intuitiva	*Hay conciencia de la importancia de una arquitectura de la información para la organización y se reconoce. * Surge un proceso y diferentes personas dentro de la organización; siguen procedimientos similares, aunque informales e intuitivos. * No hay entrenamiento formal y la gente obtiene sus habilidades en la práctica y a través de la aplicación repetida de técnicas. * Los requerimientos tácticos de las personas impulsan el desarrollo de los componentes de la arquitectura de la información.
3 Proceso definido	*La importancia de la arquitectura de la información es entendida, aceptada y la responsabilidad de su entrega es asignada y comunicada con claridad. *Los procedimientos relacionados, herramientas y técnicas, aunque no son sofisticadas, han sido estandarizadas, documentadas y forman parte de actividades informales de entrenamiento. *Se han desarrollado políticas básicas de arquitectura de la información, incluyendo algunos requisitos estratégicos, pero no se hacen cumplir en forma consistente las políticas, estándares y herramientas. *Está establecida una función de administración de datos definida formalmente, que fija estándares a nivel de toda la organización y que está comenzando a reportar sobre la entrega y el uso de la arquitectura de la información. *Están surgiendo herramientas automatizadas de administración de datos a nivel de toda la organización, pero los procesos y las reglas usadas están definidas por las ofertas del vendedor de software de base de datos.

4 Administrado y Medible	*El desarrollo y la puesta en vigor de la arquitectura de la información está totalmente soportado por medio de métodos y técnicas formales. *El proceso responde a cambios y a las necesidades de la organización. *Se está midiendo la responsabilidad del desempeño del proceso de desarrollo de la arquitectura de información. *Las actividades formales de capacitación están definidas, documentadas y aplicadas de manera consistente. *Se han propagado las herramientas automatizadas de apoyo, pero aún no están integradas. Las mejores prácticas internas son compartidas y están introducidas en el proceso. *La métrica básica ha sido identificada y está establecido un sistema de medición. *El proceso de definición de la arquitectura de información es proactiva y está enfocada a resolver las necesidades futuras de la organización. *La organización de la administración de datos está involucrada activamente en todos los esfuerzos de desarrollo de aplicaciones para asegurar la consistencia. *Un repositorio automatizado está totalmente implementado y los modelos más complejos de datos están siendo implementados para respaldar el contenido de información de las bases de datos. *Los sistemas ejecutivos de información y los sistemas de soporte de decisiones están respaldando la información disponible.
5 Optimizado	*La arquitectura de información es ejecutada de manera consistente en todos los niveles y constantemente se hace énfasis en su valor para la organización. *El personal de TIC tiene la experiencia y las habilidades necesarias para desarrollar y mantener una arquitectura de información robusta y responsable que refleja todos los requerimientos de la organización. *La información suministrada por la arquitectura de información es aplicada de manera consistente y extensiva. *Se hace uso extensivo de las mejores prácticas de la industria en el desarrollo y mantenimiento de la arquitectura de la información incluyendo un proceso constante de mejoramiento. *Está definida la estrategia para respaldar información a través almacenamiento de datos y tecnologías de minería de datos. *La arquitectura de la información está mejorando constantemente y toma en consideración la información no tradicional sobre los procesos, las organizaciones y los sistemas.

Tabla 8-4. Niveles de madurez de la Arquitectura de la Información en una organización

Planeación estratégica de TIC

PO1 Definir un plan estratégico de TIC.

Satisface los requerimientos del negocio de:

- Lograr un balance óptimo entre las oportunidades de tecnología de información y los requerimientos del negocio para TIC, así como para asegurar sus logros futuros.

Se hace posible a través de:

- Un proceso de planeación estratégica emprendido en intervalos regulares dando lugar a planes a largo plazo. Los planes a largo plazo deberán ser traducidos periódicamente en planes operacionales estableciendo metas claras y concretas a corto plazo:

Toma en consideración:

- Estrategia del negocio de la empresa.
- Definición de cómo TIC soporta los objetivos de negocio.
- Inventario de soluciones tecnológicas e infraestructura actual.

- Monitoreo del mercado de tecnología.
- Estudios de factibilidad oportunos y chequeos con la realidad.
- Análisis de los sistemas existentes.
- Posición de la empresa sobre riesgos, en el proceso de compra (time-on-market), calidad.
- Necesidades de la Administración senior en el proceso de compra, soportado en revisión crítica.

Objetivos de control detallados

Tecnología de Información como parte del Plan de la Organización a corto y largo plazo.

La alta gerencia será la responsable de desarrollar e implementar planes a largo y corto plazo que satisfagan la misión y las metas de la organización. A este respecto, la alta gerencia deberá asegurar que los problemas de TIC, así como las oportunidades, sean evaluados adecuadamente y reflejados en los planes a largo y corto plazo de la organización. Se deben desarrollar planes de TIC a largo y corto plazo para ayudar a asegurar que el uso de la TIC esté acorde con la misión y las estrategias de negocio de la organización.

Plan a largo plazo de TIC

La Gerencia de TIC y los dueños del proceso de negocio serán responsables de desarrollar regularmente planes de TIC a largo plazo, que apoyen el logro de la misión y las metas generales de la organización. El método de planeación deberá incluir mecanismos para solicitar información a los interesados internos y externos más importantes, que se ven afectados por los planes estratégicos de TIC. De la misma manera, la Gerencia deberá implementar un proceso de planeación a largo plazo, adoptar un enfoque estructurado y determinar la estructura para el plan.

Plan a largo plazo de TIC - Enfoque y Estructura

La Gerencia de TIC y los dueños del proceso de negocio deberán establecer y aplicar un enfoque estructurado al proceso de planeación a largo plazo. Esto deberá traer como resultado un plan de alta calidad que cubra las preguntas básicas de qué, quién, cómo, cuándo y por qué el proceso de planeación de TIC debe tomar en cuenta los resultados del análisis del riesgo, incluyendo los riesgos del negocio, entorno, tecnología y recursos humanos. Los aspectos que necesitan ser tomados en cuenta y ser cubiertos adecuadamente durante el proceso de planeación incluyen el modelo de organización y sus cambios, la distribución geográfica, la evolución tecnológica, los costos, los requerimientos legales y regulatorios, requerimientos de terceras partes o del mercado, el horizonte de planeación, reingeniería de procesos del negocio, la asignación de personal, proceso interno o por outsourcing, datos, sistemas de aplicación y arquitecturas de la tecnología. Los planes de TIC , de largo y corto alcance deberán incorporar indicadores y objetivos de desempeño. El plan mismo deberá hacer referencia a otros planes tales como el plan de calidad de la organización y el plan de manejo de riesgos de información.

Cambios al Plan a largo plazo de TIC

La Gerencia de TIC y los dueños del proceso del negocio deberán asegurar que se establezca un proceso con el fin de adaptar los cambios al plan a largo plazo de la organización y los cambios en las condiciones de la TIC.

La gerencia Senior deberá establecer una política que requiera que se desarrollen y se mantengan planes de largo y corto plazo de TIC.

Planeación a corto plazo para la Función de TIC

La Gerencia de TIC y los dueños del proceso del negocio deberán asegurar que el plan a largo plazo de TIC se traduzca regularmente en planes a corto plazo de TIC.

Estos planes a corto plazo deberán asegurar que se asignen los recursos apropiados de la función de servicios de TIC con una base consistente con el plan a largo plazo de TIC. Los planes a corto plazo deberán ser reevaluados y modificados periódicamente según se considere necesario respondiendo a las condiciones de cambios en el negocio y en TIC. La realización oportuna de estudios de factibilidad deberá asegurar que la ejecución de los planes a corto plazo sea iniciada adecuadamente.

Comunicación de los Planes de TIC

La gerencia debe asegurar que los planes de largo y de corto plazo de tecnología de la información sean comunicados a los dueños del proceso del negocio y a otras partes relevantes en toda la organización.

Monitoreo y Evaluación de los Planes de Tecnología de la Información

La gerencia debe establecer procesos para captar y reportar la retroalimentación de los dueños y usuarios del proceso del negocio respecto a la calidad y utilidad de los planes de largo y de corto plazo. La retroalimentación obtenida debe ser evaluada y considerada en la planeación futura de la tecnología de la información.

Evaluación de los Sistemas Existentes

Previo al desarrollo o modificación del Plan Estratégico o plan a largo plazo de TIC , la Gerencia de TIC debe evaluar los sistemas existentes en términos de: nivel de automatización de negocio, funcionalidad, estabilidad, complejidad, costo y fortalezas y debilidades, con el propósito de determinar el nivel de soporte que ofrecen los sistemas existentes a los requerimientos del negocio.

Modelo de madurez

Nivel de Madurez	**Planeación estratégica de TIC**
0 Inexistente	* No se realiza la planeación estratégica de TIC. * La administración no está conciente de que la planeación estratégica de TIC es necesaria para apoyar los objetivos del la organización.
1 Inicial /Ad hoc	* La administración de TIC está conciente de la necesidad de planeación estratégica de TIC , pero no se ha instalado un proceso estructurado de decisión. * La planeación estratégica de TIC se realiza con base a la necesidad en respuesta a un requerimiento específico de la organización y los resultados son por lo tanto esporádicos e inconsistentes. * La planeación estratégica de TIC es discutida ocasionalmente en las reuniones de administración de TIC, pero no en las reuniones de administración de la organización. * La correspondencia entre los requerimientos de la organización, las aplicaciones y la tecnología ocurren de manera reactiva, impulsadas por ofertas de los vendedores, en lugar de ser por una estrategia en toda la organización. * La posición estratégica de riesgo es identificada informalmente en cada proyecto.
2 Repetible pero Intuitiva	* La planeación estratégica de TIC es entendida por la administración de TIC, pero no está documentada. * La planeación estratégica de TIC es realizada por la administración de TIC, pero sólo es compartida con la administración del la organización en la medida en que se necesita. * La actualización del plan estratégico de TIC se lleva a cabo sólo en respuesta a solicitudes de la administración y no hay un proceso proactivo para identificar los desarrollos de TIC y del la organización que requieren actualizaciones del plan. * Las decisiones estratégicas son impulsadas por proyecto, sin consistencia con una estrategia general de la organización. * Los riesgos y los beneficios de usuario de las principales decisiones estratégicas son reconocidos de forma intuitiva.

	* Una política define cuándo y cómo realizar la planeación estratégica de TIC.
3 Proceso d e finid o	* La planeación estratégica de TIC sigue un método estructurado que está documentado y que es conocido por todos los miembros del personal. * El proceso de planeación de TIC es razonablemente adecuado y asegura que la planeación apropiada probablemente se realice. Sin embargo, se da discreción a los administradores individuales respecto a la implementación del proceso y no hay procedimientos para examinar el proceso regularmente. * La estrategia general de TIC incluye una definición consistente de riesgos que la organización está dispuesta a correr como un innovador o seguidor. * Las estrategias financiera, técnica y de recursos humanos de TIC impulsan cada vez más la adquisición de nuevos productos y tecnologías. * La planeación estratégica de TIC es discutida en reuniones de la administración de la organización.
4 Administrado y Medible	* La planeación estratégica de TIC es una práctica estándar y la administración señalaría las excepciones. * La planeación estratégica de TIC es una función definida de la administración con responsabilidades a nivel de alta gerencia. * Con respecto al proceso de planeación estratégica de TIC, la administración puede monitorearla, tomar decisiones informadas con base en ésta y medir su efectividad. * Tanto la planeación a corto como a largo plazo se realiza y cae en "cascada" a la organización haciéndose actualizaciones a medida que se necesitan. * La estrategia de TIC y la estrategia de toda la organización se están volviendo cada vez más coordinadas resolviendo procesos de la organización y capacidades de valor agregado y apalancando el uso de aplicaciones y de tecnologías a través de reingeniería del proceso de trabajo. * Hay un proceso bien definido para balancear los recursos internos y externos requeridos en el desarrollo y en las operaciones del sistema. * Benchmarking frente a normas y competidores de la industria se está volviendo cada vez más formalizada.
5 Optimizado	* La planeación estratégica de TIC, es un proceso documentado, viviente, es constantemente considerado en la determinación del objetivo de la organización y tiene como resultado un valor discernible para la organización a través de inversiones en TIC. * Constantemente se están actualizando las consideraciones de riesgo y de valor agregado en el proceso de planeación estratégica de TIC. * Hay una función de planeación estratégica de TIC que es integral con la función de planeación de la organización. * Se desarrollan planes realistas de TIC a largo plazo y los mismos están siendo constantemente actualizados para que reflejen la tecnología cambiante y los desarrollos relacionados con la organización. * Los planes de corto plazo de TIC contienen hitos de tareas de proyectos y productos que son constantemente monitoreados y actualizados, a medida que ocurren cambios. * El establecimiento de Benchmarking frente a normas de la industria bien entendidas y confiables es un proceso bien definido y está integrado con el proceso de formulación de estrategias. * La organización de TIC identifica y respalda nuevos desarrollos de la tecnología.

Tabla 8-5. Niveles de madurez en la planeación estratégica de la Tecnología de Información.

Dirección tecnológica

PO3 Determinar la dirección tecnológica.

Satisface los requerimientos de negocio de:

- Aprovechar la tecnología disponible y las que van apareciendo en el mercado para impulsar y posibilitar la estrategia del negocio.

Se hace posible a través de:

- La creación y mantenimiento de un plan de infraestructura tecnológica que establece y administra expectativas claras y realistas de lo que puede brindar la tecnología en términos de productos, servicios y mecanismos de entrega

Toma en consideración:

- Capacidad de la infraestructura actual.
- Monitoreo de desarrollos tecnológicos por la vía de fuentes confiables.
- Realización de prueba de conceptos.
- Riesgos, restricciones y oportunidades.
- Planes de adquisición.
- Estrategia de migración y mapas alternativos (roadmaps).
- Relaciones con los vendedores.
- Reevaluación independiente de la tecnología.
- Cambios de precio /desempeño de hardware y de software

Objetivos de control detallados

Planeación de la Infraestructura Tecnológica

La función de servicios de información deberá crear y actualizar regularmente un plan de infraestructura tecnológica que concuerde con los planes a largo y corto plazo de tecnología de información. Dicho plan deberá abarcar aspectos tales como arquitectura de sistemas, dirección tecnológica y estrategias de migración.

Monitoreo de Tendencias y Regulaciones Futuras

La función de servicios de información deberá asegurar el monitoreo continuo de tendencias futuras y condiciones regulatorias, de tal manera que estos factores puedan ser tomados en consideración durante el desarrollo y mantenimiento del plan de infraestructura tecnológica.

Contingencias en la Infraestructura Tecnológica

El plan de infraestructura tecnológica deberá ser evaluado sistemáticamente en cuanto a aspectos de contingencia (por ejemplo, redundancia, resistencia, capacidad de adecuación y evolución de la infraestructura).

Planes de Adquisición de Hardware y Software

La Gerencia de la función de servicios de información deberá asegurar que los planes de adquisición de hardware y software sean establecidos y que reflejen las necesidades identificadas en el plan de infraestructura tecnológica.

Estándares de Tecnología

Tomando como base el plan de infraestructura tecnológica, la Gerencia deberá definir normas de tecnología con la finalidad de fomentar la estandarización.

Modelo de madurez

Nivel de Madurez	**Dirección tecnológica**
0 Inexistente	*No hay conciencia de la importancia para la entidad de planear la infraestructura de la tecnología. *No existen los conocimientos y la experiencia necesarios para desarrollar un plan de infraestructura de tecnología. *Hay una falta de entendimiento de que la planeación para el cambio tecnológico es crítica para asignar recursos con efectividad.
1 Inicial /Ad hoc	*La administración reconoce la necesidad de planear la infraestructura de la tecnología, pero no ha formalizado ni un proceso ni un plan. *Los desarrollos del componente de tecnología y las implementaciones de la tecnología emergente son ad-hoc y aisladas. *Hay un enfoque reactivo y operativo de la planeación. *Las orientaciones de la tecnología están impulsadas por los planes de evolución de producto de hardware, software de sistemas y de los vendedores de software de aplicación a menudo contradictorios. *La comunicación del impacto potencial de cambios en la tecnología es inconsistente.
2 Repetible pero Intuitiva	*Hay un entendimiento implícito de la necesidad e importancia de planificar la tecnología. *Esta necesidad e importancia es comunicada. * La planeación es táctica y enfocada en generar soluciones técnicas a los problemas técnicos, en vez de estar enfocada hacia el uso de tecnología para satisfacer necesidades de la organización. *La evaluación de los cambios tecnológicos es dejada a diferentes personas que siguen procesos intuitivos pero similares. *No hay una capacitación formal y comunicación formal de roles y responsabilidades. *Las técnicas y estándares comunes están emergiendo para el desarrollo de los componentes de la infraestructura.
3 Proceso definido	*La administración está conciente de la importancia del plan de infraestructura de la tecnología. *El proceso de desarrollo del plan de infraestructura de tecnología es razonablemente correcto y está en concordancia con el plan estratégico de TIC. *Hay un plan de infraestructura de la tecnología que es definido, documentado y bien comunicado, pero se aplica de manera inconsistente. *La orientación de la infraestructura de la tecnología incluye una comprensión de dónde quiere estar la organización: a la vanguardia o quedar retrasada en el uso de tecnología, basada en los riesgos y en la concordancia con la estrategia de la organización. *Los vendedores claves son seleccionados con base en la comprensión de sus planes de largo plazo de desarrollo de la tecnología y de producto, consistente con la orientación de la organización.
4 Administrado y Medible	*El personal de TIC tiene la experiencia y las habilidades necesarias para desarrollar un plan de infraestructura de tecnología. *Hay un entrenamiento formal y especializado para la investigación de la tecnología. *El impacto potencial de cambiar y de las tecnologías emergentes es tomado en cuenta y validado. *La administración puede identificar desviaciones del plan y anticipar problemas. Se ha asignado responsabilidad por el desarrollo y el mantenimiento de un plan de infraestructura de tecnología. *El proceso es sofisticado y responde al cambio. *Se han introducido al proceso las mejores prácticas internas. *La estrategia de recursos humanos está en concordancia con la orientación de la tecnología, para asegurar que los miembros del personal de TIC puedan manejar los cambios de tecnología. Los planes de migración para introducir nuevas tecnologías están definidos. *Se está respaldando el outsourcing y su participación en la organización (participación como socios) para tener acceso a la experiencia y las habilidades necesarias.
5 Optimizado	*Existe una función de investigación para revisar las tecnologías emergentes que evolucionan y llevando acabo Benchmarks de la organización en contraposición con las normas de la industria. *La organización se guía por las normas y desarrollos internacionales de la industria, en lugar de estar impulsada por los vendedores de tecnología. *El impacto potencial del cambio tecnológico sobre la organización es

	revisado a nivel de la gerencia general y las decisiones de actuar reflejan la contribución de las influencias humanas y tecnológicas sobre las soluciones de información. *Hay aprobación ejecutiva formal de las orientaciones tecnológicas nuevas y de las cambiadas. *La participación en los organismos que establecen normas de la industria y grupos de usuarios de vendedores está formalizada. *La entidad tiene un plan sólido de infraestructura de la tecnología que refleja los requerimientos de la organización, responsable y puede ser modificada para reflejar cambios en el entorno de la organización. * Está establecido un proceso constante y ejecutado de mejoras al plan de Infraestructura de TIC. * Se usan extensamente las mejores prácticas de la industria para determinar la orientación.

Tabla 8-6. Niveles de madurez en la Dirección Tecnológica.

Identificar soluciones automatizadas

AI1 Identificar soluciones automatizadas.

Satisface los requerimientos de negocio de:

- Asegurar un efectivo y eficiente enfoque para satisfacer los requerimientos del usuario.

Se hace posible a través de:

- Una objetiva y clara identificación y análisis de oportunidades alternativas comparadas contra los requerimientos de los usuarios.

Tomar en consideración:

- Conocimientos de soluciones disponibles en el mercado.
- Metodologías de Adquisición e implementación.
- Involucramiento del usuario en el proceso de compra.
- Alineamiento con las estrategias de la empresa y de TIC.
- Definición de requerimientos de información.
- Estudios de factibilidad (de costo-beneficio, alternativas, etc.).
- Requerimientos de funcionalidad, operatividad, aceptación y sostenimiento.
- Cumplimiento con la arquitectura de información.
- Costo - efectividad de la seguridad y los controles.
- Responsabilidades de los proveedores

Objetivos de control detallados

Definición de Requerimientos de Información

La metodología del ciclo de vida de desarrollo de sistemas de la organización debe asegurar que los requerimientos del negocio ya satisfechos por el sistema actual y a ser satisfechos por el sistema nuevo propuesto o modificado (software, datos e infraestructura), estén claramente definidos antes de aprobar cualquier proyecto de desarrollo, implementación o modificación. La metodología del ciclo de vida de desarrollo de sistemas deberá exigir que los requerimientos de las soluciones funcionales y

operacionales sean especificados, incluyendo desempeño, protección, confiabilidad, compatibilidad, seguridad y legislación.

Formulación de Acciones Alternativas

La metodología del ciclo de vida de desarrollo de sistemas de la organización debe proveer el análisis de las acciones alternativas que deberán satisfacer los requerimientos del negocio, establecidos para un sistema nuevo o modificado.

Formulación de Estrategias de Adquisición

La adquisición, desarrollo y mantenimiento de sistemas de información debe ser considerada en el contexto de la tecnología de información de la organización, en sus planes a corto y largo plazo. La metodología del ciclo de vida de desarrollo de sistemas de la organización debe estipular un plan de estrategia de adquisición del software, definiendo si el software será "adquirido del anaquel", desarrollados internamente, a través de contratación, por mejoramiento del software existente o mediante una combinación de estos.

Requerimientos de Servicios de Terceros

La metodología del ciclo de vida de desarrollo de sistemas de la organización debe estipular la evaluación de requerimientos y las especificaciones para una RFP (Solicitud de Propuesta) cuando se negocie con un proveedor de servicios externo.

Estudio de Factibilidad Tecnológica

La metodología del ciclo de vida de desarrollo de sistemas de la organización debe estipular un examen de factibilidad tecnológica de cada alternativa con la finalidad de satisfacer los requerimientos de negocio establecidos para el desarrollo de un proyecto propuesto de cualquier sistema nuevo o modificado.

Estudio de Factibilidad Económica

La metodología del ciclo de vida de desarrollo de sistemas de la organización debe generar, en cada proyecto de desarrollo, implementación y modificación de sistemas de información propuesta, el análisis de los costos y beneficios asociados con cada alternativa considerada para satisfacer los requerimientos del negocio establecidos.

Arquitectura de Información

La Gerencia deberá asegurar que se tome en consideración el modelo de datos de la empresa al definir las soluciones y analizar la factibilidad de las mismas.

Reporte de Análisis de Riesgos

La metodología del ciclo de vida de desarrollo de sistemas de la organización debe asegurar, en cada proyecto de desarrollo, implementación y modificación de sistemas de información propuesto, el análisis y la documentación de las amenazas a la seguridad, puntos de impacto y debilidad y protecciones factibles de seguridad y control interno, con la finalidad de reducir o eliminar el riesgo identificado. Esto deberá llevarse a cabo en línea con el marco de referencia general de evaluación de riesgos.

Costo - efectivo de los controles de Seguridad

La Gerencia deberá asegurar que los costos y beneficios de seguridad sean examinados cuidadosamente en términos monetarios y no monetarios, para garantizar que los costos de los controles no excedan a los beneficios. La decisión requerirá la firma de aprobación formal de la Gerencia. Todos los requerimientos de seguridad deberán ser identificados en la fase de requerimientos de un proyecto y justificados, acordados y documentados como parte del caso total del negocio para un sistema de

información. Los requerimientos de seguridad para la administración de la continuidad del negocio deben ser definidos para asegurar que la activación planeada, la recuperación y la reactivación de procesos son soportadas por la solución propuesta.

Diseño de Pistas de Auditoria

La metodología del ciclo de vida de desarrollo de sistemas de la organización debe asegurar que existan mecanismos adecuados para que las pistas de auditoria estén disponibles o que dichos mecanismos puedan ser desarrollados para la solución identificada y seleccionada. Los mecanismos deberán proporcionar la capacidad de proteger datos sensitivos (ej. identificación de usuarios -user ID's- contra divulgación o mal uso).

Ergonomía

La Gerencia deberá asegurar que los proyectos de desarrollo, implementación y cambios emprendidos por la función de TIC, tomen en consideración los aspectos ergonómicos asociados con la introducción de soluciones automatizadas.

Selección del Software del Sistema

La Gerencia deberá asegurar que la función de servicios de información cumpla con un procedimiento estándar para identificar todos los programas de software potenciales que deberán satisfacer sus requerimientos operacionales.

Control de Abastecimiento

La Gerencia deberá desarrollar e implementar un enfoque central de abastecimientos que describa un conjunto común de procedimientos y estándares a ser seguidos en la adquisición de hardware, software y servicios relacionados con la tecnología de información. Los productos deberán ser revisados y probados antes de su utilización y pago.

Adquisición de Productos de Software

La adquisición de productos de software deberá seguir las políticas de adquisición de la organización.

Mantenimiento de Software de Terceras Partes

La Gerencia deberá asegurar que, para el software con licencia adquirido a terceras partes, los proveedores cuenten con los procedimientos apropiados para validar, proteger y mantener los derechos de integridad de los productos de software. Deberá tomarse en consideración el soporte del producto en cualquier acuerdo de mantenimiento relacionado con el producto entregado.

Contratos de Programación de Aplicaciones

La metodología del ciclo de vida de desarrollo de sistemas de la organización debe asegurar que los servicios de programación contratados estén justificados con una solicitud de servicios por escrito elaborada por un miembro designado de la función de servicios de información. El contrato deberá estipular que el software, la documentación y otros elementos entregables estén sujetos a pruebas y revisiones antes de ser aceptados. Además, deberá asegurar que los productos finales incluidos en el contrato de servicios de programación sean revisados y probados de acuerdo con los estándares definidos por el grupo de aseguramiento de calidad de la función de servicios de información y otras partes interesadas (como usuarios, administradores de proyecto, etc.) antes de pagar por el trabajo y aprobar el producto final. Las pruebas que deberán ser incluidas en las especificaciones del contrato deberán consistir en pruebas del sistema, pruebas de integración, pruebas de hardware y componentes, pruebas de procedimientos, pruebas de carga y estrés, pruebas de afinación y desempeño, pruebas de regresión, pruebas de aceptación del usuario y, finalmente, pruebas piloto del sistema total, con la finalidad de evitar fallas no esperadas del mismo.

Aceptación de Instalaciones

La Gerencia deberá asegurar que, dentro del contrato con el proveedor, se acuerde un plan de aceptación para las instalaciones que se proporcionarán, el cual defina los procedimientos y criterios de aceptación. Además, deberán llevarse a cabo pruebas de aceptación para garantizar que la instalación y el medio ambiente cumplan con los requerimientos especificados en el contrato.

Aceptación de Tecnología

La Gerencia deberá asegurar que, dentro del contrato con el proveedor, se acuerde un plan de aceptación para la tecnología específica a ser proporcionada, el cual defina los procedimientos y criterios de aceptación. Además, las pruebas de aceptación establecidas en el plan, deberán incluir inspección, pruebas de funcionalidad y seguimiento de cargas de trabajo.

Modelo de madurez

Nivel de Madurez	**Identificar soluciones automatizadas**
0 Inexistente	* La organización no requiere la identificación de requerimientos funcionales y operativos para el desarrollo, implementación o modificación de soluciones, como por ejemplo soluciones de sistema, de servicio, de infraestructura, de software y de datos. * La organización no mantiene una conciencia sobre las soluciones tecnológicas disponibles que son potencialmente relevantes para su organización.
1 Inicial /Ad hoc	* Existe una conciencia en la necesidad de definir requerimientos y de identificar soluciones tecnológicas. * Sin embargo, los métodos son inconsistentes y no están basados en una metodología específica de adquisición e implementación. * Los grupos de trabajo tienden a reunirse para discutir necesidades de manera informal y los requerimientos por lo general no están documentados. * Las soluciones que se identifican se basan en un conocimiento limitado del mercado, o en respuesta a ofertas de vendedores. * Existe un poco o ningún análisis estructurado o investigación de la tecnología disponible.
2 Repetible pero Intuitiva	* No hay una metodología de adquisición e implementación definida formalmente, pero los requerimientos tienden a ser definidos en una forma similar en toda la organización debido a prácticas comunes dentro de TIC. * Las soluciones son identificadas de manera informal con base en la experiencia interna y del conocimiento de la función de TIC. * El éxito de cada proyecto depende de la experiencia de unas pocas personas claves de TIC. *La calidad de la documentación y la toma de decisiones varía considerablemente. *Un enfoque no estructurado se utiliza para definir los requerimientos e identificar las soluciones de tecnología.
3 Proceso definido	* La organización ha establecido una metodología de adquisición e implementación, que requiere un método claro y estructurado para determinar soluciones de TIC para satisfacer requerimientos de la organización. * El método requiere la consideración de alternativas evaluadas a través de los requerimientos del negocio o del usuario, las oportunidades tecnológicas, la factibilidad económica, los análisis de riesgos y otros factores. * El proceso para la determinación de la solución de TIC es aplicado sólo para algunos proyectos con base en las decisiones hechas por el grupo de trabajo involucrado, la cantidad de tiempo de administración dedicado, el tamaño y la prioridad del requerimiento original del la organización. * Un enfoque estructurado es utilizado para definir los requerimientos e identificar las soluciones de TIC.
4 Administrado y Medible	* La organización ha establecido una metodología de adquisición e implementación, que ha evolucionado hasta el punto en que es inusual que no sea aplicada. * La documentación es de buena calidad y cada etapa es aprobada debidamente. * Los requerimientos son bien definidos y en conformidad con las estructuras predeterminadas. * Se consideran soluciones alternativas, incluyendo el análisis de costos y beneficios que permite que se hagan elecciones informadas.

	* La metodología es clara, definida, generalmente comprendida y medible. * Por lo tanto, las excepciones pueden ser determinadas y corregidas fácilmente por la administración. * Las soluciones responden eficientemente a los requerimientos de los usuarios y hay conciencia de que las soluciones consideradas con perspectiva al futuro pueden mejorar los procesos del la organización y la solución competitiva.
5 Optimizado	* La metodología de adquisición e implementación de la organización ha estado sujeta a constante mejoramiento y se ha mantenido a la par de los cambios en la tecnología. * Tiene flexibilidad, lo que le permite manejar la gama de proyectos desde aplicaciones de gran escala en toda la organización hasta proyectos específicos tácticos. * La metodología es soportada por bases de datos de conocimientos internos y externos que contienen materiales de referencia sobre soluciones de tecnología. * La metodología misma genera la documentación en una estructura predefinida que hace más eficiente la producción y el mantenimiento. * La organización a menudo identifica nuevas oportunidades para utilizar la tecnología y así obtener ventaja competitiva, influir en el proceso de reingeniería de la organización y mejorar la eficiencia general.

Tabla 8-7. Niveles de madurez en la identificación de soluciones automatizadas.

Adquirir y mantener software de aplicación

AI2 Adquirir y mantener software aplicativo.

Satisface los requerimientos de negocio de:

- Proporcionar funciones automatizadas que soporten efectivamente los procesos del negocio.

Se hace posible a través de:

- La definición de declaraciones específicas sobre requerimientos funcionales y operacionales y una implementación estructurada con requerimientos claros.

Toma en consideración:

- Pruebas funcionales y de aceptación.
- Controles de aplicación y requerimientos de seguridad.
- Requerimientos de documentación.
- Ciclo de vida del software de aplicación.
- Arquitectura en la información empresarial.
- Metodología para el ciclo de vida de desarrollo del sistema.
- Interfase usuario-máquina.
- Personalización de paquetes.

Objetivos de control detallados

Métodos de Diseño

La metodología del ciclo de vida de desarrollo de sistemas de la organización debe estipular que se apliquen técnicas y procedimientos apropiados, incluyendo una estrecha relación con los usuarios del sistema, en la creación de las especificaciones de diseño para cada nuevo proyecto de desarrollo de

sistemas de información, verificando las especificaciones del diseño contra los requerimientos del usuario.

Cambios Significativos a Sistemas Actuales

La Gerencia deberá asegurar que, en caso de presentarse la necesidad de realizar modificaciones significativas a los sistemas actuales, se siga un proceso de desarrollo similar al utilizado en el desarrollo de sistemas nuevos.

Aprobación del Diseño

La metodología del ciclo de vida de desarrollo de sistemas de la organización requerirá que las especificaciones de diseño para todos los proyectos de desarrollo y modificación de sistemas de información, sean revisadas y aprobadas por la Gerencia, por los departamentos usuarios afectados y por la alta gerencia de la organización, cuando esto sea pertinente.

Definición y Documentación de Requerimientos de Archivos

La metodología del ciclo de vida de desarrollo de sistemas de la organización debe asegurar la aplicación de un procedimiento apropiado para la definición y documentación del formato de los archivos para cada proyecto de desarrollo y modificación de sistemas de información. Este procedimiento deberá garantizar el respeto a las reglas de diccionario de datos.

Especificaciones de Programas

La metodología del ciclo de vida de desarrollo de sistemas de la organización debe requerir la preparación de especificaciones detalladas por escrito, de los programas para cada proyecto de desarrollo o modificación de sistemas de información. Además, la metodología deberá garantizar que las especificaciones de los programas correspondan a las especificaciones del diseño del sistema.

Diseño para la Recopilación de Datos Fuente

La metodología del ciclo de vida de desarrollo de sistemas de la organización debe requerir la especificación de mecanismos adecuados, para la recopilación y entrada de datos para cada proyecto de desarrollo o modificación de sistemas de información.

Definición y Documentación de Requerimientos de Entrada de Datos

La metodología del ciclo de vida de desarrollo de sistemas de la organización debe requerir que existan mecanismos adecuados para definir y documentar los requerimientos de entrada de datos para cada proyecto de desarrollo o modificación de sistemas de información.

Definición de Interfases

La metodología del ciclo de vida de desarrollo de sistemas de la organización debe estipular que se especifiquen, diseñen y documenten apropiadamente todas las interfases internas y externas.

Interfase Usuario-Máquina

La metodología del ciclo de vida de desarrollo de sistemas de la organización debe asegurar el desarrollo de una interfase entre el usuario y la máquina fácil de utilizar y que sea capaz de auto documentarse (por medio de funciones de ayuda en línea).

Definición y Documentación de Requerimientos de Procesamiento

La metodología del ciclo de vida de desarrollo de sistemas de la organización debe requerir que existan mecanismos adecuados para definir y documentar los requerimientos de procesamiento para cada proyecto de desarrollo o modificación de sistemas de información.

Definición y Documentación de Requerimientos de Salida de Datos

La metodología del ciclo de vida de desarrollo de sistemas de la organización debe requerir que existan mecanismos adecuados para definir y documentar los requerimientos de salida de datos para cada proyecto de desarrollo o modificación de sistemas de información

Controlabilidad

La metodología del ciclo de vida de desarrollo de sistemas de la organización debe requerir que se especifiquen mecanismos adecuados, para garantizar que se identifiquen los requerimientos de seguridad y control internos para cada proyecto de desarrollo o modificación de sistemas de información. La metodología deberá asegurar además que los sistemas de información estén diseñados para incluir controles de aplicación que garanticen que los datos de entrada y salida estén completos, sean precisos, oportunos y autorizados. Deberá llevarse a cabo una evaluación de sensibilidad durante el inicio del desarrollo o modificación del sistema. Los aspectos básicos de seguridad y control interno de un sistema a ser desarrollado o modificado deberán ser evaluados junto con el diseño conceptual del mismo, con el fin de integrar los conceptos de seguridad en el diseño, tan pronto como sea posible.

Disponibilidad como Factor Clave de Diseño

La metodología del ciclo de vida de desarrollo de sistemas de la organización debe asegurar que la disponibilidad sea considerada en el proceso de diseño de nuevos o modificados sistemas de información en la fase más temprana posible. La disponibilidad debe ser analizada y, en caso necesario, incrementada a través de mejoras de mantenimiento y confiabilidad.

Consideraciones de Integridad de TIC para el Software de Programas de Aplicación

La organización deberá establecer procedimientos para asegurar, cuando esto aplique, que los programas de aplicación contengan instrucciones que verifiquen rutinariamente las tareas realizadas por el software, para apoyar el aseguramiento de la integridad de los datos y el cual haga posible la restauración de la integridad a través de procedimientos de recuperación en reversa u otros medios.

Pruebas de Software de Aplicación

Deberán aplicarse pruebas unitarias, pruebas de aplicación, pruebas de integración y pruebas de carga y estrés, de acuerdo con el plan de prueba del proyecto y con los estándares de pruebas establecidos antes de ser aprobado por el usuario. Se deberán aplicar adecuadas medidas de seguridad para prevenir divulgación de información sensitiva durante las pruebas.

Materiales de Consulta y Soporte para Usuarios

La metodología del ciclo de vida de desarrollo de sistemas de la organización debe asegurar que se preparen manuales de referencia y soporte adecuados para los usuarios (preferiblemente en formato electrónico) como parte de cada proyecto de desarrollo o modificación de sistemas de información

Reevaluación del Diseño del Sistema

La metodología del ciclo de vida de desarrollo de sistemas de la organización debe asegurar que el diseño del sistema sea reevaluado siempre que ocurran discrepancias técnicas y/o lógicas durante el desarrollo o mantenimiento del sistema.

Modelo de madurez

Nivel de Madurez	**Adquirir y mantener software de aplicación**
0 Inexistente	* No hay un proceso para diseñar y especificar aplicaciones. * Típicamente, las aplicaciones se obtienen sobre la base de ofertas impulsadas por vendedores, reconocimiento de marcas o familiaridad del personal de TIC con productos específicos, con poca o ninguna consideración de los requerimientos reales.
1 Inicial /Ad hoc	* Hay conocimiento de que se requiere un proceso para adquirir y mantener aplicaciones. * Los métodos, sin embargo, varían de un proyecto a otro sin consistencia alguna y típicamente se encuentran aislados de los demás proyectos. * Es probable que la organización haya adquirido una variedad de soluciones individuales para requerimientos específicos y que ahora esté padeciendo de problemas e ineficiencias de productos de software existentes en la organización, incluyendo el mantenimiento y el soporte. * Los usuarios de la organización no pueden sacar mucho provecho de las inversiones de TIC.
2 Repetible pero Intuitiva	* Hay procesos similares para adquirir y mantener aplicaciones, pero éstos están basados en la experiencia dentro de la función de TIC, no en un proceso documentado. * La tasa de éxito con las aplicaciones depende en gran medida de las habilidades internas y de los niveles de experiencia dentro de TIC. * El mantenimiento es usualmente problemático y se ve afectado cuando se han perdido conocimientos internos de la organización.
3 Proceso definido	* Hay procesos documentados de adquisición e implementación. * Se hace un intento de aplicar consistentemente procesos documentados en todas las diferentes aplicaciones y proyectos, pero no siempre se les encuentra prácticos de implementar o que reflejen las soluciones tecnológicas actuales. * Los métodos, son generalmente inflexibles y difíciles de aplicar en todos los casos, de modo que las medidas son frecuentemente desviadas. * En consecuencia, las aplicaciones se adquieren a menudo en forma fragmentada. * El mantenimiento sigue un método definido, pero frecuentemente absorbe demasiado tiempo y es ineficiente.
4 Administrado y Medible	* Hay una metodología formal, clara y bien entendida de adquisición e implementación de sistemas, que incluye un diseño formal y procesos para especificaciones, criterios de adquisición de software de aplicación, un proceso para pruebas y la documentación de requerimientos, asegurando que todas las aplicaciones se adquieran y se mantengan en forma consistente. * Existen mecanismos formales de aprobación para asegurar que todos los pasos se siguen y que son autorizadas las excepciones. * Los procedimientos y prácticas han evolucionado y están adecuados a la organización, son utilizados por todo el personal y aplicables a la mayoría de los requerimientos de aplicación.
5 Optimizado	* Las prácticas de adquisición y mantenimiento de software de aplicación son acordes con los procesos definidos. * El método está basado en componentes, con aplicaciones predefinidas, estandarizadas y adaptadas a las necesidades de la organización. * Es usual que se tomen métodos a nivel de toda la organización. * El proceso de adquisición y mantenimiento se encuentra en un nivel avanzado que posibilita un desarrollo rápido y permite una alta capacidad de respuesta, así como también flexibilidad para responder a los requerimientos variables de la organización. * El proceso de adquisición e implementación de software de aplicación está sujeto a una mejora continua y respaldado por una base de conocimiento interno y externo con materiales de referencia y mejores prácticas. * La metodología genera la documentación en una estructura predefinida que hace más eficiente la producción y el mantenimiento.

Tabla 8-8. Nivel de madurez en adquirir y mantener software de aplicación en una organización.

Acreditar e instalar soluciones y cambios

AI7 Instalar y acreditar soluciones y cambios.

Satisface los requerimientos de negocio de:

- Verificar y confirmar que la solución sea adecuada para el propósito deseado.

Se hace posible a través de:

- La realización de una migración de instalación, conversión y plan de aceptación adecuadamente formalizado.

Toma en consideración:

- Entrenamiento del usuario y personal de operaciones de TIC.
- Conversión de datos.
- Una prueba ambiental reflejando al ambiente real.
- Acreditación.
- Revisiones post implementación y retroalimentación.
- Participación del usuario final en las pruebas.
- Planes continuos de mejoramiento de calidad.
- Requerimientos de continuidad del negocio.
- Medición de capacidad y desempeño a través del sistema.
- Acuerdos y criterios de aceptación.

Objetivos de control detallados

Entrenamiento

El personal de los departamentos de usuarios afectados y el grupo de operaciones de la función de servicios de información deberán estar entrenados de acuerdo al plan de entrenamiento definido y los materiales relacionados, como parte de cualquier proyecto de desarrollo, implementación o modificación de sistemas de información.

Dimensionamiento del Desempeño del Software de Aplicación

El dimensionamiento (optimización) del desempeño del software de aplicación deberá establecerse como una parte integral de la metodología del ciclo de vida de desarrollo de sistemas de la organización para predecir los recursos requeridos para operar software nuevo o significativamente modificado.

Plan de implementación

Un plan de implementación debe ser preparado, revisado y aprobado por partes relevantes y debe ser usado para medir el progreso. El plan de implementación debe estar direccionado hacia la preparación del sitio, adquisición e instalación de equipos, entrenamiento del usuario, instalación de cambios al software operativo, implementación de procedimientos operativos y conversión.

Conversión del Sistema

La metodología del ciclo de vida de desarrollo de sistemas de la organización debe asegurar, como parte de cada proyecto de desarrollo, implementación o modificación de sistemas de información, que los elementos necesarios del sistema anterior sean convertidos al sistema nuevo de acuerdo con el plan preestablecido.

Conversión de Datos

La Gerencia debe requerir que el plan de conversión de datos este preparado, definiendo los métodos de recolección y verificación de los datos que serán convertidos e identificando y resolviendo cualquier error encontrado durante la conversión. Las pruebas a ser desarrolladas incluyen la comparación, del archivo original, y el convertido, revisión de la compatibilidad de los datos transformados con el nuevo sistema, revisión de los archivos maestros después de la conversión para asegurar la precisión de los datos de los archivo maestros y así asegurar que las transacciones realizadas actualicen tanto a los archivos maestros antiguos como los nuevos durante el periodo entre la conversión inicial y la implementación final. Una verificación detallada de los procesos iniciales del nuevo sistema deben ser desarrollados para confirmar una implementación exitosa. La gerencia debe asegurar que la responsabilidad de la transformación exitosa de datos recaiga sobre los propietarios del sistema.

Planes y estrategias de prueba

Los planes y las estrategias de prueba deben estar preparados y autorizados por el propietario del sistema y por la gerencia de TIC.

Pruebas a Cambios

La Gerencia deberá asegurar que los cambios sean probados por un grupo de prueba independiente (distinto al de los desarrolladores) de acuerdo con la evaluación de impacto y recursos en un ambiente de prueba separado antes de comenzar su uso en el ambiente de operación regular. También deberán desarrollarse planes de respaldo externo. Las pruebas de aceptación deberán llevarse a cabo en un ambiente representativo del ambiente operacional futuro (por ejemplo, condiciones similares de seguridad, controles internos, cargas de trabajo, etc.)

Criterios y Desempeño de Pruebas en Paralelo/Piloto

Deben establecerse procedimientos para asegurar que las pruebas piloto o en paralelo sean llevadas a cabo de acuerdo con un plan preestablecido y que los criterios para la terminación del proceso de pruebas sean especificados con anterioridad.

Prueba de Aceptación Final

Los procedimientos deberán asegurar, como partes de las pruebas de aceptación final o de aseguramiento de calidad de sistemas de información nuevos o modificados, una evaluación y aprobación formal de los resultados de las pruebas por parte de la Gerencia de los departamentos usuarios afectados y de TIC. Las pruebas deben cubrir todos los componentes del sistema de información (software de aplicación, instalaciones, tecnología, procedimientos de usuarios).

Pruebas y Acreditación de Seguridad

La Gerencia deberá definir e implementar procedimientos para asegurar que la Gerencia de operaciones y la Gerencia usuaria acepten formalmente los resultados de las pruebas y el nivel de seguridad para los sistemas, junto con el riesgo residual existente. Esos procedimientos deben reflejar los roles y responsabilidades acordados entre el usuario final, desarrollo de sistemas, administración de red y del personal de operaciones del sistema, considerando los aspectos de segregación de cuentas, supervisión y control.

Prueba Operacional

La Gerencia deberá asegurar que, antes de poner el sistema en operación, el usuario o custodio designado (la parte designada para correr el sistema en nombre del usuario), valide su operación como un producto completo, bajo condiciones similares a las del ambiente de aplicación y de la misma manera en que el sistema será operado en un ambiente de producción.

Paso o promoción a Producción

La Gerencia deberá definir e implementar procedimientos formales para controlar la entrega del sistema de desarrollo a pruebas y a operación. La Gerencia debe solicitar que se obtenga la autorización del propietario del sistema antes que el nuevo sistema sea trasladado a producción y antes que el sistema viejo sea descontinuado. El nuevo sistema será sucesivamente operado durante los ciclos de producción diaria, mensual y trimestral. Los ambientes respectivos deberán separarse y protegerse apropiadamente.

Evaluación del Cumplimiento de los Requerimientos del Usuario

La metodología del ciclo de vida de desarrollo de sistemas de la organización debe requerir que se realice una revisión post – implementación de los requerimientos operacionales del sistema de información (por ejemplo, capacidad, desempeño de procesamiento a través del sistema etc.) con el fin de evaluar si las necesidades del usuario están siendo satisfechas por el sistema.

Revisión Gerencial Post - Implementación

La metodología del ciclo de vida de desarrollo de sistemas de la organización debe requerir que una revisión post - implementación del sistema de información operacional donde se evalúe y reporte si el sistema proporcionó los beneficios esperados de la manera más económica.

Modelo de madurez

Nivel de Madurez	**Acreditar e instalar soluciones y cambios**
0 Inexistente	* Hay una carencia completa de la instalación formal o los procesos de acreditación y ni la alta administración, ni el personal de TIC reconocen la necesidad de verificar que las soluciones son adecuadas para el propósito previsto.
1 Inicial /Ad hoc	* Hay un conocimiento de la necesidad de verificar y de confirmar que las soluciones puestas en ejecución respondan al propósito previsto. * Las pruebas se realizan para algunos proyectos, pero la iniciativa de prueba se deja a los equipos de trabajo por cada proyecto y el enfoque tomado varía. * La acreditación y finalización formal es rara o inexistente.
2 Repetible pero Intuitiva	* Hay una cierta consistencia y similitud entre el enfoque para pruebas y acreditación, pero no se basan típicamente en ninguna metodología. * Los equipos de trabajo de desarrollo deciden normalmente el enfoque para pruebas y hay generalmente ausencia de pruebas de integración. * Hay un proceso de aprobación informal.
3 Proceso definido	* Se establece en la organización una metodología formal referente a la instalación, migración, conversión y aceptación. * Los procesos de instalación y de acreditación se integran en el ciclo de vida del sistema y se automatiza a un cierto grado. * La capacitación, la prueba y la transición al estado de producción y acreditación, frecuentemente varía del proceso definido con base en tomas de decisiones personales. * La calidad de la producción de los sistemas es inconsistente con los nuevos sistemas generando a menudo un nivel significativo de problemas posteriores a la implementación.
4 Administrado y Medible	* Los procedimientos se formalizan, se desarrollan, son prácticos y se encuentran bien organizados con los ambientes y procedimientos de pruebas y acreditación. * La evaluación y conocimiento de los requerimientos del usuario es estandardizada y medible, produciendo métricas que permiten una evaluación y análisis efectivo para la administración. * La calidad de la producción de los sistemas es satisfactoria para la

	administración, con un nivel razonable de problemas después de la implementación. * La automatización de los procesos es a la medida de la organización y depende de cada proyecto. * La administración suele estar satisfecha con el nivel de la eficiencia actual a pesar de la falta de una evaluación después de la etapa de implementación. * El sistema de pruebas refleja adecuadamente el ambiente de desarrollo. * Se realizan pruebas de estrés para nuevos sistemas y los existentes en proyectos importantes.
5 Optimizado	* Los procesos de instalación y acreditación se han refinado a un nivel de buena práctica, con base en los resultados de una mejora continua. * Estos procesos se integran completamente en el ciclo de vida del sistema y se automatizan cuando es conveniente, permitiendo que la capacitación y las pruebas sean más eficientes, así como la transición al estado de producción de nuevos sistemas. * Los ambientes de pruebas están bien desarrollados, el registro de problemas y los procesos de la resolución aseguran una transición eficiente y eficaz al ambiente de producción. * La acreditación ocurre generalmente sin un retrabajo, y los problemas después de la implementación se limitan normalmente a correcciones de menor importancia. * Se estandariza el seguimiento que se le da después de la implementación, a través de lecciones aprendidas incorporadas dentro del proceso de mejora continua. * Las pruebas de estrés para nuevos sistemas y para los sistemas con alguna modificación se aplican constantemente.

Tabla 8-9. Nivel de madurez en acreditar e instalar soluciones y cambios.

Definir y administrar niveles de servicio

DS1 Definir y administrar los niveles de servicio.

Satisface los requerimientos de negocio de:

- Establecer un entendimiento común del nivel de servicio requerido.

Se hace posible a través de:

- El establecimiento de acuerdos de niveles de servicio que formalicen los criterios de desempeño contra los cuales se medirá la cantidad y la calidad del servicio.

Toma en consideración:

- Acuerdos o convenios formales.
- Definición de responsabilidades.
- Tiempos y volúmenes de respuesta.
- Cargos.
- Garantías de integridad.
- Acuerdos de confidencialidad.
- Criterio de satisfacción del cliente.
- Análisis costo-beneficio de los niveles de servicio requerido.

- Monitoreo y reporte.

Objetivos de control detallados

Marco de Referencia para el Acuerdo de Niveles de Servicio

La alta gerencia deberá establecer un marco de referencia en donde presente la definición de acuerdos de niveles de servicio formales y determine el contenido mínimo: disponibilidad, confiabilidad, desempeño, capacidad de crecimiento, niveles de soporte proporcionados al usuario, plan de contingencia/recuperación, nivel mínimo aceptable de funcionalidad del sistema satisfactoriamente liberado, restricciones (límites en la cantidad de trabajo), cargos por servicio, instalaciones de impresión central (disponibilidad), distribución de impresión central y procedimientos de cambio. Los usuarios y la función de servicios de información deberán contar con un convenio escrito que describa el nivel de servicio en términos cualitativos y cuantitativos. El convenio definirá las responsabilidades de ambas partes. La función de servicios de información deberá prestar la calidad y la cantidad de servicios ofrecida y los usuarios deberán ajustar los servicios solicitados a los límites acordados.

Aspectos sobre los Acuerdos de Nivel de Servicio

Deberá lograrse un acuerdo explícito sobre los aspectos que el convenio de nivel de servicios deberá tener. El convenio de nivel de servicio deberá cubrir por lo menos los siguientes aspectos: disponibilidad, confiabilidad, desempeño, capacidad de crecimiento, niveles de soporte proporcionados a los usuarios, plan de contingencia/recuperación, nivel mínimo aceptable de funcionalidad del sistema satisfactoriamente liberado, restricciones (límites en la cantidad de trabajo), cargos por servicio, instalaciones de impresión central (disponibilidad), distribución de impresión central y procedimientos de cambios.

Procedimientos de Desempeño

Deberán definirse procedimientos que aseguren que la forma y las responsabilidades sobre las relaciones que rigen el desempeño (por ejemplo, acuerdos de confidencialidad) entre todas las partes involucradas sean establecidas, coordinadas, mantenidas y comunicadas a todos los departamentos afectados.

Monitoreo y Reporte

La Gerencia de la función de servicios de información deberá designar a un Gerente de nivel de servicio que sea responsable de monitorear y reportar los alcances de los criterios de desempeño del servicio especificado y todos los problemas encontrados durante el procesamiento. Las estadísticas de monitoreo deberán ser analizadas oportunamente. Deberán tomarse acciones correctivas apropiadas e investigarse las fallas.

Revisión de Acuerdos y Contratos de Nivel de Servicio

La Gerencia deberá implementar un proceso de revisión regular de los convenios de nivel de servicio y de los contratos de proveedores de servicios como terceras partes.

Elementos sujetos a Cargo

Deberán incluirse provisiones para elementos sujetos a cargo en los acuerdos de niveles de servicio para hacer posible las comparaciones y decisiones de niveles de servicio contra su costo.

Programa de Mejoramiento del Servicio

La Gerencia deberá implementar un proceso para asegurar que los usuarios y los Gerentes de nivel de servicio concuerden regularmente en un programa de mejoramiento del servicio con el fin de dar seguimiento a mejoras al nivel de servicio cuyo costo esté justificado.

Modelo de madurez

Nivel de Madurez	**Definir y Administrar Niveles de Servicio**
0 Inexistente	* La administración no ha reconocido la necesidad de un proceso para definir niveles de servicio. * Las obligaciones de rendir cuentas y las responsabilidades del monitoreo de niveles de servicios no están asignadas.
1 Inicial /Ad hoc	* Existe consciencia sobre la necesidad de administrar niveles de servicio, pero el proceso es informal y reactivo. * La responsabilidad y obligación de rendir cuentas para definir y administrar servicios no está definida. * Las mediciones del desempeño, si existen, son cualitativas, pero tienen metas definidas sin precisión. * El reporte del desempeño es informal, poco frecuente e inconsistente.
2 Repetible pero Intuitiva	* Existen acuerdos celebrados sobre el nivel de servicio, pero son informales y no son revisados. * El reporte de nivel de servicio es incompleto, irrelevante o engañoso, dependiendo de las habilidades y de la iniciativa de los administradores individuales. * Se nombra un coordinador de nivel de servicio con responsabilidades definidas, pero con autoridad insuficiente. * El proceso de cumplimiento del acuerdo al nivel de servicio es voluntario y su cumplimiento no se exige.
3 Proceso definido	* Las responsabilidades están bien definidas, pero con autoridad discrecional. * El proceso de desarrollo de los acuerdos de nivel de servicio está establecido con puntos de verificación para revaluar los niveles de servicio y la satisfacción del cliente. * Los criterios de niveles de servicios están definidos y acordados con los usuarios, con un mayor nivel de estandarización. * Las carencias de nivel de servicio están identificadas, pero la planeación de la resolución es aún informal. * La relación entre la financiación brindada y los niveles de servicio esperados se está formalizando cada vez más. * El nivel de servicio se basa cada vez más en el Benchmarks de la industria y no puede resolver las necesidades específicas de la organización.
4 Administrado y Medible	* Los niveles de servicios son cada vez más definidos en la fase de definición de los requerimientos de sistema e incorporados en el diseño de los ambientes de aplicación y operación. * La satisfacción del cliente se mide y se determina de manera rutinaria. * Las medidas de desempeño están reflejando cada vez más las necesidades del usuario final, en lugar de reflejar únicamente las metas de TIC. * Los criterios de medición de los niveles de servicio de usuario se están volviendo estandarizados y reflejo de la norma de la industria. * El análisis de las causas originales se realiza cuando no se cumple con los niveles de servicio. * El sistema de reporte para monitorear los niveles de servicio se está volviendo cada vez más automatizado. * Los riesgos operativos y financieros asociados con el incumplimiento de los acuerdos de niveles de servicio están definidos y son entendidos con claridad.
5 Optimizado	* Los niveles de servicio son reevaluados constantemente para asegurar que estén en línea con los objetivos de TIC y de la organización, mientras que se saca provecho de los adelantos tecnológicos y de las mejoras en la proporción precio/desempeño de producto. * Todos los procesos de nivel de servicio están sujetos a procesos constantes de mejoramiento. * Los criterios para definir niveles de servicio están definidos en cuestión de los procesos críticos de la organización e incluyen la disponibilidad, confiabilidad, desempeño, capacidad de crecimiento, soporte de usuario, planeación de la continuidad y consideraciones de seguridad. * Los niveles de satisfacción del cliente son monitoreados y exigidos. * Los niveles de servicio esperados son evaluados contra las normas de la industria, pero también reflejan las metas específicas estratégicas de las unidades de la organización. * La administración de TIC tiene los recursos y la responsabilidad que se necesitan para lograr los objetivos de desempeño de nivel de servicio y la compensación ejecutiva está estructurada para proveer incentivos que satisfagan tales objetivos.* La alta administración monitorea los indicadores de desempeño y los indicadores de objetivo como parte de un proceso de mejora continua.

Tabla 8-10. Nivel de madurez para definir y administrar niveles de servicio.

Administrar el desempeño y la capacidad

DS3 Administrar el desempeño y la capacidad.

Satisface los requerimientos de negocio de:

- Asegurar que la capacidad adecuada está disponible y que se esté haciendo el mejor uso de ella para alcanzar el desempeño deseado.

Se hace posible a través de:

- Recolección de datos, análisis y reporte del rendimiento de los recursos, aplicación de mediciones y demanda de cargas de trabajo.

Toma en consideración:

- Requerimientos de disponibilidad y desempeño.
- Monitoreo y reporte automatizado.
- Herramientas de modelado.
- Administración de capacidad.
- Disponibilidad de recursos.
- Cambios en precio-rendimiento del hardware y software.

Objetivos de control detallados

Requerimientos de Disponibilidad y Desempeño

El proceso de administración deberá asegurar que las necesidades del negocio con respecto a disponibilidad y desempeño de los servicios de información sean identificados y convertidos en requerimientos y términos de disponibilidad.

Plan de Disponibilidad

La Gerencia deberá asegurar el establecimiento de un plan de disponibilidad para alcanzar, monitorear y controlar la disponibilidad de los servicios de información.

Monitoreo y Reporte

La Gerencia deberá implementar un proceso que asegure que el desempeño de los recursos de tecnología de información sea continuamente monitoreado y que las excepciones sean reportadas de manera oportuna y completa.

Herramientas de Modelado

La Gerencia deberá asegurar que se utilicen las herramientas de modelado apropiadas para producir un modelo del sistema actual, calibrado y ajustado según la carga de trabajo real y que sea preciso dentro de los niveles de carga recomendados. Las herramientas de modelado deberán utilizarse para apoyar el pronóstico de los requerimientos de capacidad, confiabilidad de la configuración, desempeño y disponibilidad. Deberán llevarse a cabo investigaciones técnicas profundas sobre el hardware de los sistemas y deberán incluirse pronósticos acerca de futuras tecnologías.

Manejo Proactivo del Desempeño

El proceso de administración del desempeño deberá incluir la capacidad de pronóstico para permitir que los problemas sean solucionados antes de que éstos afecten el desempeño del sistema. Deberán llevarse a cabo análisis de las fallas e irregularidades del sistema en cuanto a frecuencia, grado del impacto y magnitud del daño.

Pronóstico de Carga de Trabajo

Deberán establecerse controles para asegurar que se preparen pronósticos de carga de trabajo con el fin de identificar tendencias y proporcionar la información necesaria para el plan de capacidad.

Administración de Capacidad de Recursos

La Gerencia de la función de servicios de información deberá establecer un proceso de planeación para la revisión del desempeño y capacidad del hardware con el fin de asegurar que siempre exista una capacidad justificable económicamente para procesar las cargas de trabajo acordadas y para proporcionar la cantidad y calidad de desempeño requeridas, prescritas en los acuerdos de nivel de servicio. El plan de capacidad deberá cubrir escenarios múltiples.

Disponibilidad de Recursos

Cuando se identifiquen como recursos de alta disponibilidad, la gerencia deberá prevenir que estos recursos no estén disponibles, mediante la implementación de mecanismos de tolerancia de fallas, mecanismos de asignación equitativa de recursos y la definición de prioridades de tareas.

Programación de Recursos

La Gerencia deberá asegurar la adquisición oportuna de la capacidad requerida, tomando en cuenta aspectos como resistencia, contingencia, cargas de trabajo y planes de almacenamiento.

Modelo de madurez

Nivel de Madurez	**Administrar el desempeño y la capacidad**
0 Inexistente	* La administración no ha reconocido que los procesos clave de la organización pueden requerir altos niveles de desempeño de TIC o que la necesidad general de la organización para los servicios de TIC puede exceder la capacidad. * No está establecido un proceso de planeación de capacidad.
1 Inicial /Ad hoc	* El desempeño y la administración de la capacidad son reactivos y esporádicos. * Los usuarios tienen a menudo que inventar formas para superar las limitaciones de desempeño y de capacidad. * Existe muy poca apreciación de las necesidades de servicio de TIC por parte de los propietarios de los procesos de organización. * La administración de TIC está conciente de la necesidad de desempeño y capacidad de administración, pero la acción emprendida es por lo general reactiva o está incompleta. * El proceso de planeación es informal.
2 Repetible pero Intuitiva	* La administración de la organización está conciente del impacto de no administrar el desempeño y la capacidad. * Para las áreas críticas, las necesidades de desempeño son generalmente cubiertas, con base en el estudio de los sistemas individuales así como el conocimiento de los equipos de apoyo y de proyecto. * Se pueden usar algunas herramientas individuales para diagnosticar problemas de desempeño y de capacidad, pero la consistencia de los resultados depende de la experiencia de las personas clave. * No existe una apreciación general de la capacidad de desempeño de la infraestructura de TIC ni consideración de situaciones pico y de peor caso de carga. * Es probable que ocurran problemas de disponibilidad en una forma inesperada, al azar y que tome tiempo considerable diagnosticar e implementar correcciones.

3 Proceso definido	* Los requerimientos de desempeño y de capacidad están definidos como pasos que deben ser resueltos en todas las etapas de la metodología de adquisición e implementación de sistemas. * Existen requerimientos y métricas definidas a nivel de servicio que pueden usarse para medir el desempeño operativo. * Es posible diseñar y pronosticar los requerimientos futuros de desempeño. * Los reportes pueden ser producidos dando estadísticas de desempeño. * Aún es probable que ocurran problemas y que lleve tiempo corregirlos. * A pesar que los niveles de servicio están publicados el usuario final se sentirá ocasionalmente dudoso sobre la capacidad del servicio
4 Administrado y Medible	* Se cuenta con procesos y herramientas para medir el uso de sistemas y para compararlo con los niveles definidos de servicio. * Se cuenta con información actualizada, dando estadísticas estandarizadas de desempeño y alertando los incidentes tales como la capacidad o rendimiento insuficiente. * Los incidentes causados por fallas de capacidad y desempeño se manejan en conformidad con los procedimientos definidos y estandarizados. * Se usan herramientas automatizadas para monitorear recursos específicos tales como almacenamiento de disco, servidores de red y portales de red y gateways. * Existe un intento de reportar estadísticas de desempeño en términos de servicio de la organización, de modo que los usuarios finales puedan entender los niveles de servicio de TIC. * Los usuarios se sienten en general satisfechos con la actual capacidad de servicio y están exigiendo niveles nuevos y mejorados de disponibilidad.
5 Optimizado	* Los planes de desempeño como la capacitación están totalmente sincronizados con los pronósticos de la organización y con los planes operativos y objetivos. * La infraestructura de TIC está sujeta a revisiones regulares para asegurar que se logre la capacidad óptima al menor costo posible. * Los adelantos en la tecnología están monitoreados de cerca para aprovechar el desempeño mejorado de los productos. * El sistema para medir el desempeño de TIC ha sido afinado para concentrarse en las áreas clave y están traducidas en KGIs (indicadores clave de objetivo), KPIs (indicadores clave de desempeño) y CFSs (factores críticos de éxito) para todos los procesos críticos de la organización. * Las herramientas para monitorear los recursos críticos de TIC han sido estandarizados, donde es posible, en todas las plataformas y vinculados a un solo sistema de administración de incidentes en toda la organización. * Las herramientas de monitoreo pueden cada vez más detectar y corregir automáticamente los problemas de desempeño, por ejemplo, asignar más espacio de almacenamiento o redirigir el tráfico de red. * Se detectan las tendencias que muestran problemas inminentes de desempeño causados por mayores volúmenes de negocio, posibilitando la planificación y la prevención de incidentes inesperados. * Los usuarios esperan una disponibilidad de 24 horas al día, 7 días a la semana, 365 días al año.

Tabla 8-11. Nivel de madurez para administrar el desempeño y capacitación.

Asegurar la continuidad del servicio

DS4 Garantizar la continuidad del servicio.

Satisface los requerimientos de negocio de:

- Asegurar que los servicios de TIC estén disponibles cuando se requieran y asegurar el impacto mínimo en el negocio en el evento que se presente una interrupción mayor.

Se hace posible a través de:

- Tener un plan de continuidad de TIC probado y funcional, que esté alineado con el plan de continuidad del negocio y relacionado con los requerimientos de negocio

Tomar en consideración:

- Clasificación de criticidad (severidad).
- Procedimientos alternativos.
- Respaldo y recuperación.
- Pruebas y entrenamiento sistemáticos y regulares.
- Monitoreo y procesos de escalamiento.
- Responsabilidades organizacionales internas y externas.
- Activación de la continuidad del negocio, vuelta atrás (fallback) y plan de reactivación.
- Actividades de administración de riesgos.
- Análisis de puntos únicos de falla.
- Administración de problemas

Objetivos de control detallados

Marco de Referencia de Continuidad de Tecnología de Información

La Gerencia de TIC, en cooperación con los propietarios de los procesos del negocio, deberá crear un marco de referencia de continuidad que defina los roles, responsabilidades, el enfoque/metodología basada en riesgo a seguir y las reglas y la estructura para documentar el plan de continuidad, así como los procedimientos de aprobación.

Estrategia y Filosofía del Plan de Continuidad de TIC

La Gerencia deberá garantizar que el Plan de continuidad de tecnología de información se encuentra en línea con el plan general de continuidad de la empresa para asegurar consistencia. Aún más, el plan de continuidad de TIC debe tomar en consideración el plan a mediano y largo plazo de tecnología de información, con el fin de asegurar consistencia.

Contenido del Plan de Continuidad de TIC

La Gerencia de TIC deberá asegurar que se desarrolle un plan escrito conteniendo lo siguiente:

- Guías sobre la utilización del Plan de Continuidad;
- Procedimientos de emergencia para asegurar la integridad de todo el personal afectado;
- Procedimientos de respuesta definidos para regresar al negocio al estado en que se encontraba antes del incidente o desastre;
- Procedimientos para salvaguardar y reconstruir las instalaciones;
- Procedimientos de coordinación con las autoridades públicas;
- Procedimientos de comunicación con los socios y demás interesados: empleados, clientes clave, proveedores críticos, accionistas y gerencia; e
- Información crítica sobre grupos de continuidad, personal afectado, clientes, proveedores, autoridades públicas y medios de comunicación.

Reducción de requerimientos de Continuidad de Tecnología de Información.

La Gerencia de servicios de información deberá establecer procedimientos y guías para minimizar los requerimientos de continuidad con respecto a personal, instalaciones, hardware, software, equipo, formatos, consumibles y mobiliario.

Mantenimiento del Plan de Continuidad de Tecnología de Información

La Gerencia de TIC deberá proveer procedimientos de control de cambios para asegurar que el plan de continuidad se mantiene actualizado y refleja requerimientos de negocio actuales. Esto requiere de procedimientos de mantenimiento del plan de continuidad alineados con el cambio, la administración y los procedimientos de recursos humanos.

Pruebas del Plan de Continuidad de TIC

Para contar con un Plan efectivo de Continuidad, la gerencia necesita evaluar su adecuación de manera regular o cuando se presenten cambios mayores en el negocio o en la infraestructura de TIC ; esto requiere una preparación cuidadosa, documentación, reporte de los resultados de las pruebas e implementar un plan de acción de acuerdo con los resultados.

Entrenamiento sobre el Plan de Continuidad de Tecnología de Información

La metodología de Continuidad ante desastres deberá asegurar que todas las partes interesadas reciban sesiones de entrenamiento regulares con respecto a los procedimientos a ser seguidos en caso de un incidente o un desastre.

Distribución del Plan de Continuidad de TIC

Debido a la naturaleza sensitiva de la información del plan de continuidad, dicha información deberá ser distribuida solo a personal autorizado y mantenerse bajo adecuadas medidas de seguridad para evitar su divulgación.

Consecuentemente, algunas secciones del plan deberán ser distribuidas solo a las personas cuyas actividades hagan necesario conocer dicha información.

Procedimientos de respaldo de procesamiento alternativo para Departamentos usuarios

La metodología de continuidad deberá asegurar que los departamentos usuarios establezcan procedimientos alternativos de procesamiento, que puedan ser utilizados hasta que la función de servicios de información sea capaz de restaurar completamente sus servicios después de un evento o un desastre.

Recursos Críticos de Tecnología de Información

El plan de continuidad deberá identificar los programas de aplicación, servicios de terceros, sistemas operativos, personal, insumos, archivos de datos que resultan críticos así como los tiempos necesarios para la recuperación después de que se presenta un desastre. Los datos y las operaciones críticas deben ser identificadas, documentadas, priorizadas y aprobadas por los dueños de los procesos del negocio en cooperación con la Gerencia de TIC.

Sitio y Hardware de Respaldo

La Gerencia deberá asegurar que la metodología de continuidad incorpora la identificación de alternativas relativas al sitio y al hardware de respaldo, así como una selección alternativa final. En caso de aplicar, deberá establecerse un contrato formal para este tipo de servicios.

Almacenamiento de respaldo en el sitio alterno (Off-site)

El almacenamiento externo de copias de respaldo, documentación y otros recursos tecnológicos de información, catalogados como críticos, debe ser establecido para soportar el plan de recuperación y continuidad del negocio. Los propietarios de los procesos del negocio y el personal de la función de TIC deben involucrarse en determinar que recursos de respaldo deben ser almacenados en el sitio alterno. La instalación de almacenamiento externo debe contar con medidas ambientales para los medios y otros recursos almacenados; y debe tener un nivel de seguridad suficiente, que permita proteger los recursos de respaldo contra accesos no autorizados, robo o daño. La Gerencia de TIC debe asegurar que los acuerdos/contratos del sitio alterno son periódicamente analizados, al menos una vez al año, para garantizar que ofrezca seguridad y protección ambiental.

Procedimiento de afinamiento del Plan de Continuidad

Dada una exitosa reanudación de la función de TIC después de un desastre, la gerencia de servicios de información deberá establecer procedimientos para evaluar lo adecuado del plan y actualizarlo de acuerdo con los resultados de dicha evaluación.

Modelo de madurez

Nivel de Madurez	**Asegurar la continuidad del Servicio**
0 Inexistente	* No hay entendimiento de los riesgos, vulnerabilidades y amenazas de las operaciones de TIC o del impacto de la pérdida de los servicios de TIC para la organización. * La continuidad del servicio no es considerada como que necesita atención de la administración.
1 Inicial /Ad hoc	* Las responsabilidades de la continuidad del servicio son informales, con autoridad limitada. * La administración se está volviendo conciente de los riesgos relacionados con la continuidad del servicio y de la necesidad de éste. * El enfoque es sobre la función de TIC, en vez de ser sobre la función de organización. * Los usuarios están implementando formas de evadirlo. * La respuesta a las interrupciones mayores es reactiva e improvisada. * Los cortes planeados están programados para que satisfagan las necesidades de TIC, en vez de adaptarse a los requerimientos de la organización.
2 Repetible pero Intuitiva	* La responsabilidad de la continuidad del servicio está asignada. * Los enfoques del servicio continuo son fragmentados. * El reporte sobre la disponibilidad del sistema es incompleto y no toma en cuenta el impacto sobre la organización. * No existen planes documentados de usuario o de continuidad, a pesar de que hay dedicación a la disponibilidad de servicio continuo y que se conocen sus principios rectores. * Existe un inventario razonablemente confiable de sistemas críticos y componentes. * Está surgiendo la estandarización de prácticas de continuidad del servicio y el monitoreo del proceso, pero el éxito se basa en las personas.
3 Proceso definido	* La obligación de reportar no es ambigua, las responsabilidades de planificar y probar el servicio continuo están claramente definidas y asignadas. * Los planes están documentados basándose en la importancia del sistema y en el impacto sobre la organización. * Existe un reporte periódico de prueba en el servicio continuo. * Las personas toman la iniciativa para seguir las normas y recibir entrenamiento. * La administración comunica consistentemente la necesidad de la continuidad del servicio. * Los componentes de alta disponibilidad y la redundancia de sistema se están aplicando de manera fragmentada. * Se mantiene rigurosamente un inventario de sistemas críticos y componentes.
4 Administrado y Medible	* Se hacen cumplir las responsabilidades y las normas para la continuidad del servicio. * La responsabilidad de mantener el plan de continuidad del servicio está asignada. * Las actividades de mantenimiento toman en cuenta el entorno cambiante de la organización, los resultados de pruebas de continuidad del servicio y las mejores

	prácticas internas. * Se están recopilando, analizando, reportando y ejecutando datos estructurados sobre la continuidad del servicio. * Se provee entrenamiento para los procesos de continuidad del servicio. * Las prácticas de redundancia de sistema, que incluyen el uso de componentes de alta disponibilidad, están siendo implementadas de manera consistente. * Las prácticas de redundancia y la planeación de continuidad del servicio se influyen mutuamente. * Los incidentes de falta de continuidad son clasificados y el paso cada vez mayor de escala para cada uno es bien conocido para todos los que están involucrados
5 Optimizado	* Los procesos integrados de continuidad del servicio son proactivos, se ajustan solos, son automatizados, auto analíticos, tomando en cuenta los puntos de referencia y las mejores prácticas externas. * Los planes de continuidad del servicio así como los planes de continuidad del negocio están integrados, alineados y son mantenidos de manera rutinaria. * Los requerimientos de continuidad del servicio se asegura por los vendedores y los principales proveedores. * Se lleva a cabo la comprobación global y los resultados de las pruebas son utilizados como parte del proceso de mantenimiento. * La efectividad del costo de continuidad del servicio está optimizada a través de la innovación y de la integración. * La recopilación y el análisis de datos se usa para identificar oportunidades de mejoramiento. * Las prácticas de redundancia y la planeación de continuidad del servicio están totalmente alineadas. * La administración no permite puntos únicos de falla y provee soporte para su solución. * Las prácticas de escalamiento son entendidas y cumplidas plenamente.

Tabla 8-12. Nivel de madurez para asegurar la continuidad del servicio.

Garantizar la seguridad de los sistemas

DS5 Garantizar la seguridad de los sistemas.

Satisface los requerimientos de negocio de:

- Salvaguardar la información contra uso no autorizado, divulgación o revelación, modificación, daño o pérdida.

Se hace posible a través de:

- Controles de acceso lógico que aseguren que el acceso a sistemas, datos y programas estando restringido a usuarios autorizados.

Toma en consideración:

- Requerimientos de privacidad y confidencialidad.
- Autorización, autenticación y control de acceso.
- Identificación de usuarios y perfiles de autorización.
- Necesidad de saber y necesidad de tener (need-to-know and need-to-have).
- Administración de llaves criptográficas.
- Manejo, reporte y seguimiento de incidentes.

- Prevención y detección de virus.
- Firewalls[1].
- Administración centralizada de seguridad.
- Entrenamiento a los usuarios.
- Herramientas para monitoreo del cumplimiento, pruebas de intrusión y reportes.

Objetivos de control detallados

Administrar Medidas de Seguridad

La seguridad en TIC deberá ser administrada de tal forma que las medidas de seguridad se encuentren en línea con los requerimientos de negocio. Esto incluye:

- Trasladar información sobre evaluación de riesgos a los planes de seguridad de TIC ;
- Implementar el plan de seguridad de TIC ;
- Actualizar el plan de seguridad de TIC para reflejar cambios en la configuración de TIC ;
- Evaluar el impacto de las solicitudes de cambio en la seguridad de TIC ;
- Monitorear la implementación del plan de seguridad de TIC ; y
- Alinear los procedimientos de seguridad de TIC a otras políticas y procedimientos.

Identificación, Autenticación y Acceso

El acceso lógico y el uso de los recursos de TIC deberán restringirse a través de la implementación de mecanismos adecuados de identificación, autentificación y autorización relacionando los usuarios y los recursos con las reglas de acceso. Dicho mecanismo deberá evitar que personal no autorizado, conexiones telefónicas por marcado y otros puertos de entrada al sistema (redes) tengan acceso a los recursos de cómputo, de igual forma deberá minimizar la necesidad de autorizar usuarios para usar múltiples sign-on.

Asimismo deberán establecerse procedimientos para conservar la efectividad de los mecanismos de autenticación y acceso (por ejemplo, cambios periódicos de contraseñas).

Seguridad de Acceso a Datos en Línea

En un ambiente de tecnología de información en línea, la Gerencia de TIC deberá implementar procedimientos acordes con la política de seguridad que garantiza el control de la seguridad de acceso, tomando como base las necesidades individuales demostradas de visualizar, agregar, modificar o eliminar datos.

[1] Un cortafuegos (o firewall en inglés), es un elemento de hardware o software utilizado en una red de computadoras para controlar las comunicaciones, permitiéndolas o prohibiéndolas según las políticas de red que haya definido la organización responsable de la red. Su modo de funcionar es indicado por la recomendación RFC 2979, que define las características de comportamiento y requerimientos de interoperabilidad.

Administración de Cuentas de Usuario

La Gerencia deberá establecer procedimientos para asegurar acciones oportunas relacionadas con la solicitud, establecimiento, emisión, suspensión y cierre de cuentas de usuario. Deberá incluirse un procedimiento de aprobación formal que indique el propietario de los datos o del sistema que otorga los privilegios de acceso. La seguridad de acceso a terceros debe definirse contractualmente teniendo en cuenta requerimientos de administración y no revelación. Los acuerdos de outsourcing deben considerar los riesgos, los controles sobre seguridad y los procedimientos para los sistemas de información y las redes en el contrato que se establece entre las partes.

Revisión Gerencial de Cuentas de Usuario

La Gerencia deberá contar con un proceso de control establecido para revisar y confirmar periódicamente los derechos de acceso. Se debe llevar a cabo la comparación periódica entre los recursos y los registros de las cuentas para reducir el riesgo de errores, fraudes, alteración no autorizada o accidental.

Control de Usuarios sobre Cuentas de Usuario

Los usuarios deberán controlar en forma sistemática la actividad de su(s) propia(s) cuenta (s). También se deberán establecer mecanismos de información para permitirles supervisar la actividad normal, así como alertarlos oportunamente sobre actividades inusuales.

Vigilancia de Seguridad

La administración de seguridad de TIC debe asegurar que la actividad de seguridad sea registrada y que cualquier indicación sobre una inminente violación de seguridad sea notificada inmediatamente a todos aquellos que puedan verse afectados, tanto interna como externamente y se debe actuar de una manera oportuna.

Clasificación de Datos

La Gerencia deberá implementar procedimientos para asegurar que todos los datos son clasificados en términos de sensitividad, mediante una decisión explícita y formal del dueño de los datos de acuerdo con el esquema de clasificación de datos. Aún los datos que "no requieran protección" deberán contar con una decisión formal que les asigne dicha clasificación. Los dueños deben determinar la ubicación o disposición de sus datos y determinar quienes pueden compartir los datos aun si y cuando los programas y archivos sean mantenidos, archivados o borrados. Debe quedar evidencia de la aprobación del dueño y de la disposición del dato. Se deben definir políticas para soportar la reclasificación de la información, basados sobre cambios en la sensitividad. El esquema de clasificación debe incluir criterios para administrar el intercambio de información entre organizaciones, teniendo en cuenta tanto la seguridad y el cumplimiento como la legislación relevante.

Administración de Derechos de Acceso e Identificación Centralizada

Deben existir controles para asegurar que la identificación y los derechos de acceso de los usuarios, así como la identidad del sistema y la propiedad de los datos, son establecidos y administrados de forma única y centralizada, para obtener consistencia y eficiencia de un control de acceso global.

Reportes de Violación y de Actividades de Seguridad

La administración de la función de servicios de información deberá asegurar que las violaciones y la actividad de seguridad sean registradas, reportadas, revisadas y escaladas apropiadamente en forma regular para identificar y resolver incidentes que involucren actividades no autorizadas. El acceso lógico a la información sobre el registro de recursos de cómputo (seguridad y otros logs) deberá otorgarse tomando como base el principio de menor privilegio o necesidad de saber.

Manejo de Incidentes

La Gerencia deberá implementar la capacidad de manejar incidentes de seguridad computacional, dar atención a dichos incidentes mediante el establecimiento de una plataforma centralizada con suficiente experiencia y equipada con instalaciones de comunicación rápidas y seguras. Deberán establecerse las responsabilidades y los procedimientos de manejo de incidentes para asegurar una respuesta apropiada, efectiva y oportuna a los incidentes de seguridad.

Reacreditación

La Gerencia deberá asegurar que se lleve a cabo periódicamente una reacreditación de seguridad (por ejemplo, a través de equipos de personal técnico *"tigre")* con el fin de mantener actualizado el nivel de seguridad aprobado formalmente y la aceptando el riesgo residual.

Confianza en Contrapartes

Las políticas organizacionales deberán asegurar que se implementen prácticas de control para verificar la autenticidad de las contrapartes que proporcionan instrucciones o transacciones electrónicas. Esto puede lograrse mediante el intercambio confiable de passwords, tokens o llaves criptográficas.

Autorización de transacciones

Las políticas organizacionales deberán asegurar que, en donde sea apropiado, se implementen controles para proporcionar autenticidad a las transacciones y establecer la validez de la identificación solicitada por el usuario ante el sistema. Esto requiere el empleo de técnicas criptográficas para "firmar" y verificar transacciones.

No negación o no rechazo

Las políticas organizacionales deberán asegurar que, en donde sea apropiado, las transacciones no puedan ser negadas por ninguna de las partes participantes en la operación y que se implementen controles para que no se pueda negar el origen o destino de la transacción y que se pueda probar que se envió y recibió la transacción. Esto puede lograrse a través de firmas digitales, registro de tiempos y controles de terceros confiables, y adicionalmente con políticas apropiadas que tengan en cuenta los requerimientos regulatorios relevantes.

Sendero Seguro

Las políticas organizacionales deberán asegurar que la información de transacciones sensitivas es enviada y recibida exclusivamente a través de canales o senderos seguros *(trusted paths)*. La información sensitiva incluye: información sobre administración de seguridad, datos de transacciones sensitivas, passwords y llaves criptográficas. Para lograr esto, se pueden establecer canales confiables utilizando encriptación entre usuarios, entre usuarios y sistemas y entre sistemas.

Protección de las funciones de seguridad

Todo el hardware y software relacionado con seguridad debe encontrarse permanentemente protegido contra intromisiones para proteger su integridad y contra divulgación de sus claves secretas. Adicionalmente, la organización deberá mantener discreción sobre el diseño de su seguridad, pero no basar la seguridad en mantener el diseño como secreto.

Administración de Llaves Criptográficas

La Gerencia deberá definir e implementar procedimientos y protocolos a ser utilizados en la generación, cambio, revocación, destrucción, distribución, certificación, almacenamiento, entrada, utilización y archivo de llaves criptográficas con el fin de asegurar la protección de las mismas contra modificaciones y divulgación no autorizada. Si una llave se encuentra comprometida (en riesgo), la

gerencia deberá asegurarse de que esta información se hace llegar a todas las partes interesadas a través de una lista de revocación de certificados o mecanismos similares.

Prevención, Detección y Corrección de Software "Malicioso"

Con respecto al software malicioso, tal como los virus computacionales o *Caballos de Troya,* la Gerencia deberá establecer un marco de referencia de adecuadas medidas de control preventivas, detectivas y correctivas y responder y reportar su presencia. Las Gerencias de TIC y de negocios deben asegurar que se establezcan procedimientos a través de toda la organización para proteger los sistemas de información contra virus computacionales. Los procedimientos deben incorporar protección contra virus, detección, respuesta ante su presencia y reporte.

Arquitectura de Firewalls y conexión a redes públicas

La organización deberá contar con *Firewall* adecuados para proteger contra negación de servicios y cualquier acceso no autorizado a los recursos internos si existe conexión con Internet u otras redes públicas; se deberá controlar en ambos sentidos cualquier aplicación y el flujo de administración de infraestructura y se deberá proteger contra ataques de negación del servicio.

Protección de Valores Electrónicos

La Gerencia debe proteger la integridad continuada de todas las tarjetas o mecanismos de seguridad física similares, utilizadas para autenticación o almacenamiento de información financiera o sensitiva tomando en consideración las instalaciones o equipos relacionados, los dispositivos, los empleados y los métodos de validación utilizados.

Modelo de madurez

Nivel de Madurez	**Garantizar la seguridad de los sistemas**
0 Inexistente	* La organización no reconoce la necesidad de la seguridad de TIC. * Las responsabilidades y las obligaciones de reportar no están asignadas para asegurar la seguridad. * No están implementadas medidas que soporten la administración de la seguridad de TIC. * No hay ningún reporte de seguridad de TIC y ningún proceso de respuesta de las violaciones de seguridad de TIC. * Existe una carencia total de un proceso reconocible en la administración de seguridad de sistemas.
1 Inicial /Ad hoc	* La organización reconoce la necesidad de la seguridad de TIC, pero la consciencia de la seguridad depende de las personas. * La seguridad de TIC está resuelta de manera reactiva y no se mide. * Las violaciones de seguridad de TIC invocan respuestas de "señalamiento" si se detectan, porque las responsabilidades no están claras. * Las respuestas a las violaciones de seguridad de TIC son impredecibles.
2 Repetible pero Intuitiva	* Las responsabilidades y obligaciones de la seguridad de TIC están asignadas a un coordinador de seguridad de TIC que no tiene autoridad de administración. * La conciencia de seguridad es fragmentada y limitada. * La información de seguridad de TIC es generada, pero no es analizada. * Las soluciones de seguridad tienden a responder de manera reactiva a los incidentes de seguridad de TIC y adoptando propuestas de terceros, sin resolver las necesidades específicas de la organización. * Se están desarrollando políticas de seguridad, pero aún se siguen usando habilidades y herramientas inadecuadas. * El reporte de seguridad de TIC es incompleto, engañoso y no es pertinente.
3 Proceso definido	* Existe consciencia de la seguridad y la misma es promovida por la administración. * Se han estandarizado y formalizados reportes de conocimientos de la seguridad. * Los procedimientos de seguridad de TIC están definidos y encajan en una estructura para políticas y procedimientos de seguridad. * Las responsabilidades de seguridad de TIC están asignadas, pero no se hacen cumplir de manera consistente.

	* Existe un plan de seguridad de TIC, que impulsa el análisis del riesgo y soluciones de seguridad. * El reporte de seguridad de TIC está concentrado en TIC, en lugar de concentrarse en la organización. * Se realizan pruebas de intrusión Ad hoc.
4 Administrado y Medible	* Las responsabilidades de la seguridad de TIC están claramente asignadas, administradas y se hacen cumplir. * El análisis de riesgo e impacto de seguridad se lleva a cabo de manera consistente. * Las políticas y prácticas de seguridad son completadas con bases específicas de seguridad. * Los reportes de conocimiento de seguridad se han vuelto obligatorios. * La identificación, autenticación y autorización de usuario se está estandarizando. * Se está estableciendo la certificación de seguridad del personal. * La prueba de intrusión es un proceso estándar y formalizado que conduce a mejoras. * El análisis costo/beneficio, que soporta la implementación de medidas de seguridad, es cada vez más utilizado. * Los procesos de seguridad de TIC son coordinados con la función general de seguridad de la organización. * El reporte de seguridad de TIC está vinculado con los objetivos de la organización.
5 Optimizado	* La seguridad de TIC es una responsabilidad conjunta de la organización, como de la administración de TIC y está integrada con objetivos de seguridad corporativa del la organización. * Los requisitos de seguridad de TIC están claramente definidos, optimizados e incluidos en un plan verificado de seguridad. * Las funciones de seguridad están integradas con aplicaciones en la etapa de diseño y se les puede pedir a los usuarios finales que rindan cuenta de la seguridad a la administración. * El reporte de seguridad de TIC provee un aviso anticipado del riesgo cambiante y emergente, usando métodos activos automatizados de monitoreo para los sistemas críticos. * Los incidentes son prontamente resueltos con procedimientos formalizados de respuesta a incidentes soportados por herramientas automatizadas. * Las evaluaciones periódicas de seguridad evalúan la efectividad de la implementación del plan de seguridad. * Se recoge y analiza sistemáticamente la información sobre nuevas amenazas y vulnerabilidades, y se comunican e implementan en forma expedita los controles adecuados de mitigación. * Las pruebas de intrusión, el análisis de las causas raíz de los incidentes de seguridad y la identificación preactiva del riesgo es la base para el mejoramiento continuo. * Los procesos y las tecnologías de seguridad están integrados en toda la organización.

Tabla 8-13. Nivel de madurez para garantizar la seguridad de los sistemas.

Administrar datos

DS11 Administrar los datos.

Satisface los requerimientos de negocio de:

- Asegurar que los datos permanezcan completos, precisos y válidos durante su entrada, actualización y almacenamiento.

Se hace posible a través de:

- Una combinación efectiva de controles generales y de aplicación sobre las operaciones de TIC.

Toma en consideración:

- Diseño de formatos.

- Controles sobre documentos fuente.
- Controles de entrada, procesamiento y salida.
- Identificación, movimiento y administración de la librería de medios.
- Recuperación y almacenamiento de datos.
- Autenticación e integridad.
- Propiedad de datos.
- Políticas de administración de datos.
- Modelos de datos y estándares de representación de datos.
- Integración y consistencia en todas las plataformas.
- Requisitos legales y regulatorios.

Objetivos de control detallados

Procedimientos de Preparación de Datos

La Gerencia deberá establecer procedimientos de preparación de datos que deben ser seguidos por los departamentos usuarios. En este contexto, el diseño de formas de entrada de datos deberá ayudar a minimizar los errores y las omisiones. Durante la creación de los datos, los procedimientos de manejo de errores deberán asegurar razonablemente que los errores y las irregularidades sean detectados, reportados y corregidos.

Procedimientos de Autorización de Documentos Fuente

La Gerencia deberá asegurar que los documentos fuente sean preparados apropiadamente por personal autorizado que actúa dentro de su autoridad, y que se establezca una separación de funciones adecuada con respecto al origen y aprobación de documentos fuente.

Recopilación de Datos de Documentos Fuente

Los procedimientos de la organización deberán asegurar que todos los documentos fuente autorizados estén completos, sean precisos, registrados apropiadamente y transmitidos oportunamente para su ingreso a proceso.

Manejo de errores de documentos fuente

Los procedimientos de manejo de errores durante la creación de datos deberán asegurar razonablemente que los errores y las irregularidades sean detectados, reportados y corregidos.

Retención de Documentos Fuente

Deberán establecerse procedimientos para asegurar que la organización pueda retener o reproducir los documentos fuentes originales durante un período de tiempo razonable para facilitar la recuperación o reconstrucción de datos, así como para satisfacer requerimientos legales.

Procedimientos de Autorización de Entrada de Datos

La organización deberá establecer procedimientos apropiados para asegurar que la entrada de datos sea llevada a cabo únicamente por personal autorizado.

Chequeos de Exactitud, Suficiencia y Autorización

Los datos de transacciones, ingresados para su procesamiento (generados por personas, por sistemas o entradas de interfase) deberán estar sujetos a una variedad de controles para verificar su exactitud, suficiencia y validez. Asimismo, deberán establecerse procedimientos para asegurar que los datos de entrada sean validados y editados tan cerca del punto de origen como sea posible.

Manejo de Errores en la Entrada de Datos

La organización deberá establecer procedimientos para la corrección y reenvío de datos que hayan sido capturados erróneamente.

Integridad de Procesamiento de Datos

La organización deberá establecer procedimientos para el procesamiento de datos que aseguren que la segregación de funciones sea mantenida y que el trabajo realizado sea verificado rutinariamente. Los procedimientos deberán asegurar que se establezcan controles de actualización adecuados como totales de control "corrida a corrida –**`run to run`**-" y controles de actualización de archivos maestros.

Validación y Edición de Procesamiento de Datos

La organización deberá establecer procedimientos para asegurar que la validación, autenticación y edición del procesamiento sean llevadas a cabo tan cerca del punto de origen como sea posible. Cuando se utilicen sistemas de Inteligencia Artificial, dichos sistemas serán ubicados en una infraestructura de control interactiva con operadores humanos para asegurar que las decisiones vitales son aprobadas.

Manejo de Errores en el Procesamiento de Datos

La organización deberá establecer procedimientos para el manejo de errores en el procesamiento de datos que permitan la identificación de transacciones erróneas sin que éstas sean procesadas y sin interrumpir el procesamiento de otras transacciones válidas.

Manejo y Retención de Datos de Salida

La organización deberá establecer procedimientos para el manejo y la retención de datos producidos por sus programas de aplicación de TIC. En caso de que instrumentos negociables (por ejemplo Títulos valores) sean los receptores de la salida, se deberá prestar especial cuidado en prevenir usos inadecuados.

Distribución de Datos Salidos de los Procesos

La organización deberá establecer y comunicar procedimientos escritos para la distribución de datos de salida de tecnología de información.

Balanceo y Conciliación de Datos de Salida

La organización deberá establecer procedimientos para asegurar que los datos de salida sean balanceados rutinariamente con los totales de control relevantes. Deberán existir pistas de auditoria para facilitar el seguimiento del procesamiento de transacciones y la conciliación de datos con problema.

Revisión de Datos de Salida y Manejo de Errores

La Gerencia de la organización deberá establecer procedimientos para asegurar que la precisión de los reportes de los datos de salida sea revisada por el proveedor y por los usuarios responsables. Asimismo, deberán establecerse procedimientos para controlar los errores contenidos en los datos de salida.

Provisiones de Seguridad para Reportes de Salida

La organización deberá establecer procedimientos para garantizar que la seguridad de los reportes generados por los procesos sea mantenida para todos aquellos reportes que estén por distribuirse, así como para todos aquéllos que ya hayan sido distribuidos a los usuarios.

Protección de Información Sensible durante transmisión y transporte

La Gerencia deberá asegurar que durante la transmisión y transporte de información sensible, se proporcione una adecuada protección contra acceso o modificación no autorizada, así como contra envíos a direcciones erróneas.

Protección de Información Sensitiva Desechada

La Gerencia deberá definir e implementar procedimientos para impedir el acceso a la información sensitiva, al software de las computadoras, a los discos y otros equipos o medios cuando los mismos son desechados o transferidos a otro uso. Tales procedimientos deberán garantizar que ninguna información marcada como "borrada" o "desechada", pueda ser accedida por personas internas o externas a la organización.

Administración de Almacenamiento

Deberán desarrollarse procedimientos para el almacenamiento de datos que consideren requerimientos de recuperación, de economía y así mismo tengan en cuenta las políticas de seguridad de la organización.

Períodos de Retención y Términos de Almacenamiento

Deberán definirse los períodos de retención y los términos de almacenamiento para documentos, datos, programas, reportes y mensajes (de entrada y de salida), así como los datos (claves, certificados) utilizados para su encriptación y autenticación.

Sistema de Administración de la Librería de Medios

La función de servicios de información deberá establecer procedimientos para asegurar que el contenido de su librería de medios sea inventariado sistemáticamente, que cualquier discrepancia revelada por un inventario físico sea solucionada oportunamente y que se consideren las medidas necesarias para mantener la integridad de los medios magnéticos almacenados en la librería.

Responsabilidades de la Administración de la Librería de Medios

La Gerencia de la función de servicios de información deberá establecer procedimientos de administración para proteger el contenido de la librería de medios. Deberán definirse estándares para la identificación externa de medios magnéticos y el control de su movimiento y almacenamiento físico para soporte y registro. Las responsabilidades sobre el manejo de la librerías de medios (cintas magnéticas, cartuchos, discos y disquetes) deberán ser asignadas a miembros específicos del personal de servicios de información.

Respaldo (Back-up) y Restauración

La Gerencia deberá implementar una estrategia apropiada de respaldo y recuperación para asegurar que ésta incluya una revisión de los requerimientos del negocio, así como el desarrollo, implementación, prueba y documentación del plan de recuperación. Se deberán establecer procedimientos para asegurar que los respaldos satisfagan los requerimientos mencionados anteriormente.

Funciones de Respaldo

Deberán establecerse procedimientos para asegurar que los respaldos sean realizados de acuerdo con la estrategia de respaldo definida, y que las copias de respaldo sean verificadas regularmente.

Almacenamiento de Respaldos

Los procedimientos de respaldo para los medios relacionados con tecnología de información deberán incluir el almacenamiento apropiado de los archivos de datos, del software y de la documentación relacionada, tanto dentro como fuera de las instalaciones. Los respaldos deberán ser almacenados con seguridad y las instalaciones de almacenamiento deberán ser revisadas periódicamente con respecto a la seguridad de acceso físico y la seguridad de los archivos de datos y otros elementos.

Archivo

La Gerencia deberá implementar una política y procedimientos para asegurar que el archivo cumple con requerimientos legales y de negocio y que se encuentra debidamente protegido y su información adecuadamente registrada.

Protección de Mensajes Sensitivos

Con respecto a la transmisión de datos a través de Internet u otra red pública, la Gerencia deberá definir e implementar procedimientos y protocolos que deben ser utilizados para el aseguramiento de la integridad, confidencialidad y "no negación/rechazo" de mensajes sensitivos.

Autenticación e Integridad

Antes que alguna acción crítica sea tomada sobre información originada fuera de la Organización, que se reciba vía teléfono, correo de voz, documentos (en papel), fax o correo electrónico, se deberá verificar adecuadamente la autenticidad e integridad de dicha información.

Integridad de Transacciones Electrónicas

Tomando en consideración que las fronteras tradicionales de tiempo y de geografía son menos precisas y confiables, la Gerencia deberá definir e implementar apropiados procedimientos y prácticas para transacciones electrónicas que sean sensitivas y críticas para la Organización, que permitan asegurar su integridad y autenticidad de:

- *Atomicidad* (unidad de trabajo indivisible, todas sus acciones tienen éxito o todas ellas fallan);
- *Consistencia* (si la transacción no logra alcanzar un estado final estable, deberá regresar al sistema a su estado inicial);
- *Aislamiento* (el comportamiento de una transacción no es afectado por otras transacciones que se ejecutan concurrentemente); y
- *Durabilidad* (los efectos de una transacción son permanentes después que concluye su proceso los cambios que origina deben sobrevivir a fallas de sistema).

Integridad Continua de Datos Almacenados

La Gerencia deberá asegurar que la integridad y lo adecuado de los datos mantenidos en archivos y otros medios (por ejemplo tarjetas electrónicas) se verifique periódicamente. Atención específica deberá darse a dispositivos de tokens, archivos de referencia y archivos que contengan información privada.

Modelo de madurez

Nivel de Madurez	**Administrar Datos**
0 Inexistente	* No se reconocen los datos como un recurso y un activo corporativo. * No hay propiedad asignada de datos ni responsabilidad individual de la integridad y confiabilidad de los datos. * La calidad y seguridad de los datos son deficientes o inexistentes.
1 Inicial /Ad hoc	* La organización reconoce una necesidad de datos exactos. * Algunos métodos son desarrollados a nivel individual para prevenir y

	detectar el ingreso, procesamiento y errores en la salida de los datos. * El proceso de identificación y corrección de errores depende de las actividades manuales de la persona, y las reglas y requerimientos no son transmitidos a medida que se llevan a cabo movimientos y cambios de personal. * La administración asume que los datos son exactos porque una computadora está involucrada en el proceso. * La integridad y seguridad de los datos no son requerimientos de administración y, si existe la seguridad, ésta está administrada por la función de servicios de información.
2 Repetible pero Intuitiva	* La conciencia de la necesidad de la exactitud de los datos y de mantener la integridad prevalece en toda la organización. * La propiedad de los datos comienza a tener lugar, pero a nivel de un departamento o grupo. * Las reglas y requerimientos son documentados por personas clave y no son consistentes en toda la organización y plataformas. * Los datos están en custodia de la función de los servicios de información y las reglas y definiciones están impulsadas por los requerimientos de TIC. * La seguridad e integridad de los datos son primariamente responsabilidades de la función de los servicios de información con una participación departamental menor.
3 Proceso definido	* La necesidad de integridad de los datos dentro y en toda la organización es entendida y aceptada. * Las normas de ingreso, procesamiento y salida de datos han sido formalizadas y se hacen cumplir. * El proceso de identificación y corrección de errores es automatizado. * La propiedad de los datos es asignada, y la integridad y seguridad son controladas por el responsable. * Se utilizan técnicas automatizadas para prevenir y detectar errores e inconsistencias. * Las definiciones, reglas y requerimientos de datos están claramente documentados y son mantenidos por una función de administración de base de datos. * Los datos se vuelven consistentes en todas las plataformas y a través de toda la organización. * La función de los servicios de información tiene un rol de custodio, mientras que el control de integridad de datos pasa al propietario de los datos. * La administración se basa en los reportes y análisis para las decisiones y la planeación futuras.
4 Administrado y Medible	* Los datos son definidos como un recurso y un activo corporativo, a medida que la administración exige más soporte de decisiones y más reporte de rentabilidad. * La responsabilidad por la calidad de datos está claramente definida, asignada y comunicada dentro de la organización. * Los métodos estandarizados están documentados, mantenidos, y usados para controlar la calidad de los datos, se hacen cumplir las reglas y los datos son consistentes en todas las plataformas y unidades de la organización. * La calidad de los datos es medida y la satisfacción del cliente respecto a la información es monitoreada. * El reporte de administración asume un valor estratégico para asesorar clientes, tendencias y evaluaciones de productos. * La integridad de los datos se vuelve un factor significativo, con la seguridad de datos reconocida como un requerimiento de control. * Se ha establecido una función formal de administración de datos a nivel de toda la organización, con los recursos y la autoridad para hacer cumplir la estandarización de datos.
5 Optimizado	* La administración de datos es un proceso maduro, integrado y de funcionamiento cruzado que tiene una meta claramente definida y bien entendida de entregar información de calidad al usuario, con criterios claramente definidos de integridad, disponibilidad y confiabilidad. * La organización maneja activamente datos, información y conocimientos como los recursos y los activos corporativos, con el objetivo de maximizar el valor de la organización. * La cultura corporativa hace énfasis en la importancia de datos de alta calidad que necesitan ser protegidos y tratados como un componente clave de capital intelectual. * La propiedad de datos es una responsabilidad estratégica con todos los requerimientos, reglas, reglamentaciones y consideraciones claramente documentados, mantenidos y comunicados.

Tabla 8-14. Nivel de madurez en la administración de los datos.

Administración de instalaciones

DS12 Administrar el ambiente físico.

Objetivos de control detallados

Satisface los requerimientos de negocio de:

- Proporcionar un ambiente físico conveniente que proteja los equipos y al personal de TIC contra peligros naturales o fallas humanas.

Se hace posible a través de:

- La instalación de controles físicos y ambientales adecuados que sean revisados regularmente para garantizar su adecuado funcionamiento.

Toma en consideración:

- Acceso a instalaciones.
- Identificación del sitio (instalación).
- Seguridad física.
- Políticas de inspección y escalamiento.
- Plan de continuidad de negocios y administración de crisis.
- Salud y seguridad del personal.
- Políticas de mantenimiento preventivo.
- Protección contra amenazas ambientales.
- Monitoreo automatizado

Modelo de madurez

Nivel de Madurez	**Administración de instalaciones**
0 Inexistente	* No hay conciencia de la necesidad de proteger las Instalaciones o la inversión en los recursos de computación. * Los factores ambientales, incluyendo la protección contra incendios, el polvo, la energía y el calor y la humedad excesivos, no son monitoreados ni controlados.
1 Inicial /Ad hoc	* La organización ha reconocido un requerimiento del mercado de proveer un entorno físico adecuado que proteja los recursos y el personal de los riesgos naturales y provocados por el hombre. * No existen procedimientos estándar y la administración de Instalaciones y equipo depende de las habilidades y capacidades de las personas clave. * No se revisa el mantenimiento y la gente se mueve dentro de las Instalaciones sin restricción. * La administración no monitorea los controles ambientales de la instalación ni el movimiento de personal
2 Repetible pero Intuitiva	* La conciencia de la necesidad de proteger y de controlar el entorno físico de computación es reconocida y evidente en la asignación de los presupuestos y de otros recursos. * Los controles ambientales son implementados y monitoreados por el personal de operaciones. * La seguridad física es un proceso informal, impulsado por un pequeño grupo de empleados que tienen un alto nivel de preocupación sobre la seguridad de las Instalaciones físicas. * Los procedimientos de mantenimiento de las Instalaciones no están bien

	documentados y se basan en las mejores prácticas de unas pocas personas. * Las metas de la seguridad física no se basan en ningún estándar formal y la administración no asegura que se logren los objetivos de seguridad.
3 Proceso definido	* La necesidad de mantener un entorno controlado de computación es entendida y aceptada dentro de la organización. Los controles ambientales, de mantenimiento preventivo y de seguridad física son rubros del presupuesto aprobados y la administración les hace seguimiento. * Se aplican restricciones de acceso, permitiéndose el acceso a las Instalaciones de computación sólo al personal aprobado. * Los visitantes son registrados y a veces escoltados, dependiendo del personal responsable. * Las Instalaciones físicas tienen perfil bajo y no se pueden identificar fácilmente. * Las autoridades civiles monitorean el cumplimiento de las reglamentaciones sanitarias y de seguridad. * Los riesgos están asegurados, pero no se hace esfuerzo alguno para optimizar los costos de seguros.
4 Administrado y Medible	* La necesidad de mantener un entorno controlado de computación es entendida totalmente, como es evidente en la estructura organizacional y en la asignación de presupuesto. * Los requerimientos ambientales y de seguridad física están documentados y el acceso está estrictamente controlado y monitoreado. * La responsabilidad y la propiedad han sido establecidas y comunicadas. * El personal de las Instalaciones ha sido totalmente entrenado en situaciones de emergencia, así como también en prácticas sanitarias y de seguridad. * Están estandarizados los mecanismos de control para restringir el acceso a las Instalaciones y para resolver factores ambientales y de seguridad. * La administración monitorea la efectividad de los controles y el cumplimiento de las normas establecidas. * La recuperación de los recursos de computación está incorporada al proceso corporativo de administración de riesgos. * Están desarrollados planes para toda la organización, se llevan a cabo pruebas regulares e integradas y las lecciones aprendidas son incorporadas en las revisiones de los planes. * La información integrada se usa para optimizar la cobertura de seguros y los costos relacionados.
5 Optimizado	* Hay un plan a largo plazo para las Instalaciones que se requiere que soporten el entorno de computación de la organización. * Las normas están definidas para todas las Instalaciones, abarcando la selección del sitio, la construcción, custodia, seguridad de personal, sistemas mecánico y eléctrico, protección contra incendio, rayo e inundación. * Todas las Instalaciones son inventariadas y clasificadas en conformidad con el proceso de administración de riesgos de la organización en progreso. * El acceso está estrictamente controlado y se basa en la necesidad para el trabajo, es monitoreado constantemente y los visitantes son escoltados en todo momento. * El entorno es monitoreado y controlado por medio de equipo especializado y las salas de equipos se vuelven automatizadas. * Los programas de mantenimiento preventivo hacen cumplir estrictamente los programas y se aplican pruebas regulares a los equipos sensitivos. * La estrategia y normas de las Instalaciones están en correspondencia con los objetivos de disponibilidad de servicios de TIC y están integradas con la planeación de la continuidad del negocio y con la administración de crisis. * La administración revisa y optimiza las Instalaciones constantemente, capitalizando sobre las oportunidades de mejorar la contribución del negocio.

Tabla 8-15. Nivel de madurez en la administración de instalaciones.

Monitoreo y evaluación del funcionamiento de las TIC

ME1 Monitorear y evaluar el desempeño de TIC.

Modelo de madurez

Nivel de Madurez	**Monitoreo y Evaluación del Funcionamiento de las TIC**
0 Inexistente	* La organización no tiene ningún proceso de supervisión puesto en ejecución. * No realiza de forma independiente la supervisión de proyectos o de procesos. * Los informes útiles, oportunos y exactos no están disponibles. * No se reconoce la necesidad de objetivos de proceso claramente entendidos.
1 Inicial /Ad hoc	* La administración reconoce una necesidad de obtener y evaluar la información sobre monitoreo de procesos. * Los procesos estándares de obtención y verificación no se han identificado. * El monitoreo está implementado y las métricas se eligen sobre una base de caso-por-caso, según las necesidades específicas de los proyectos y procesos de TIC. * El monitoreo es implementado generalmente de forma reactiva a un incidente que ha causado alguna pérdida o comprometido a la organización. * La función de la contabilidad supervisa las razones financieras básicas para TIC.
2 Repetible pero Intuitiva	* Las medidas básicas que se supervisarán han sido identificadas. * Existen métodos y técnicas para recolección y evaluación, pero los procesos no han sido adoptados a través de la organización entera. * La interpretación del monitoreo de resultados se basa en la experiencia de los individuos clave. * Las herramientas limitadas se eligen y se ponen en ejecución para recopilar la información, pero la compilación no se basa en con un enfoque planeado.
3 Proceso de finido	* La gerencia ha comunicado e institucionalizado el proceso de monitoreo. * Los programas educativos y de entrenamiento para el monitoreo se han puesto en ejecución. * Una base de conocimiento formalizada de la información histórica de desempeño se ha desarrollado. * La evaluación todavía se realiza a nivel de proceso individual de TIC y del proyecto y no es integrado con todos los procesos. * Las herramientas para monitorear los procesos y los niveles de servicio han sido definidos. * Las medidas de la contribución de los servicios de información al desempeño de la organización han sido definidos, usando criterios financieros y operacionales tradicionales. * Se definen las medidas de funcionamiento específicas de TIC, las medidas no financieras, las medidas estratégicas, las medidas de la satisfacción de cliente y los porcentajes de disponibilidad. * Un marco se ha definido para el funcionamiento de monitoreo.
4 Administrado y Medible	* La administración ha definido las tolerancias bajo las cuales los procesos deben funcionar. * El reporte del monitoreo de resultados se está estandardizando y se está normalizando. Hay integración de métricas a través de todos los proyectos y los procesos. * Se formalizan los sistemas de reporte de la administración de la organización de TIC. * Las herramientas automatizadas están integradas y contribuyen a lo largo de la organización a obtener y supervisar la información operacional sobre aplicaciones, sistemas y procesos. * La administración puede evaluar el funcionamiento basado en los criterios acordados y aprobados por los involucrados. * Las medidas de la función de TIC están alineadas con los objetivos de la organización.
5 Optimizado	* Un proceso de mejora continua de calidad es desarrollado para actualizar los estándares y políticas de monitoreo e incorporar la mejores prácticas de la industria.* Todos los procesos de monitoreo son optimizados y apoyan a objetivos de la organización.* Las métricas de la organización se utilizan rutinariamente para medir el funcionamiento y son integradas dentro de marcos estratégicos de evaluación tales como el balanced scorecard (Tablero de Comando) de TIC.* El proceso de monitoreo y el rediseño constante son consistentes con los planes de mejora continua de procesos de la organización.* Benchmarking contra la industria y los competidores clave se han formalizado, con criterios bien entendidos de comparación.

Tabla 8-16. Nivel de madurez en el monitoreo y evaluación del funcionamiento de TIC.

8.5. MEJORES PRÁCTICAS EN LA ADMINISTRACIÓN DE BASES DE DATOS. CASO PRÁCTICO

A una empresa de auditoria informática se le encargó la revisión de riesgos informáticos de la compañía. A continuación se presenta las sugerencias que la empresa auditora dio referente a la gestión de bases de datos.

Controles de bases de datos

Es crítico que la integridad y disponibilidad de la base de datos se mantenga; esto de asegura a través de los siguientes controles:

- Establecer y obligar la definición de estándares, políticas y procedimientos.
- Establecer e implementar el respaldo de datos y los procedimientos de recuperación para asegurar la disponibilidad de la base de datos.
- Establecer los niveles necesarios de controles de acceso para los elementos de datos, tablas y archivos para prevenir acceso inadvertido o no autorizado.
- Establecer controles para asegurar que sólo el personal autorizado pueda actualizar la base de datos.
- Establecer controles para manejar los problemas de concurrencias, tales como múltiples usuarios deseando actualizar los mismos elementos de datos al mismo tiempo (por ejemplo, el uso de **`commit`**, el bloqueo de registros o archivos).
- Establecer controles para asegurar la precisión, totalidad y consistencia de los elementos de datos y las relaciones en las bases de datos. Es importante que estos controles estén contenidos en las definiciones de tablas o columnas.
- Usar puntos de verificación de las bases de datos para minimizar la pérdida de datos y los esfuerzos de recuperación para reiniciar el proceso después de una falla del sistema.
- Ejecutar la optimización de las bases de datos para reducir espacio de disco no utilizado y verificar las relaciones de datos definidas.
- Seguir los procedimientos de reestructuración cuando se hagan cambios lógicos, físicos y de procedimientos.
- Usar herramientas de reporte de desempeño de las bases de datos para monitorear y mantener la eficiencia de la base de datos.
- Minimizar la habilidad para usar medios que no sean sistemas de aplicación o fuera de los procedimientos de seguridad para acceder a los datos de la base.

Revisiones de bases de datos

Para la revisión de bases de datos deben considerarse los siguientes elementos:

Ilustración 8-11. Elementos a considerar en la revisión a bases de datos.

Diseño.

Debe verificarse:

- El **`modelo de base de datos`** existente y todas las entidades deben de tener un nombre significativo e identificarse por medio de una llave primaria o foránea.
- Las **`relaciones`** deben de tener una cardinalidad explícita y coherente.
- El **`modelo entidad-relación`** debe estar sincronizado con el esquema físico de la base de datos.
- El **`esquema lógico`**. Asegurarse de que todas las entidades existan en el diagrama entidad-relación, así como en las tablas o vistas. Todas las relaciones deben estar representadas a través de una llave primaria y una foránea, así mismo los atributos debe tener un nombre lógico y su relación especificada, no debe de aceptar valores nulos en las llaves primarias.
- El **`esquema físico`** debe revisarse, para reservar el espacio necesario de las tablas, logs y áreas temporales.

Acceso

El **acceso principal** a la base de datos como los procedimientos de almacenamiento y los triggers deben ser analizados. El uso de índices para minimizar el tiempo de acceso debe verificarse y realizar búsquedas, sino está basado en índices, debe existir una justificación. Si el DBMS permite la selección de los métodos o tipos de índices, se debe verificar que su uso es el correcto.

Administración

Los **niveles de seguridad** para todos los usuarios y sus roles deben de identificarse dentro de la base de datos, y los **permisos de acceso** para todos los usuarios y/o grupos de usuarios deben ser justificados. Verificar que existan **procedimientos de recuperación de desastres** así como de **respaldos**, que aseguren la confiabilidad y disponibilidad de la base de datos, así mismo, mecanismos y procedimientos que aseguren el adecuado manejo, consistencia e integridad durante los accesos concurrentes.

Interfaces

Para asegurar la integridad y confidencialidad de los datos, los **procedimientos de importación y exportación de la información** deben ser verificados con otros sistemas.

Portabilidad

Siempre que sea posible, **debe utilizarse** un Lenguaje de Consulta Estructurado SQL.

Análisis del impacto del negocio

Un análisis de impacto en el negocio es el primer paso para desarrollar planes de continuidad del negocio y en especial de los planes de recuperación de desastres.

Dentro de las diversas preguntas que deben formularse durante el análisis del impacto del negocio se encuentran:

- ¿Cuáles son los procesos de negocio?, esta pregunta es fundamental para determinar su criticidad para la organización.
- ¿Cuáles son los recursos de información críticos que están relacionados con los procesos de negocio críticos?, esta es la primera consideración debido a que una interrupción en un recurso de información no es un desastre en sí misma a menos que se relacione con procesos críticos de la organización.
- ¿Cuál es el periodo de tiempo crítico de recuperación para los recursos de información en los cuales los procesos de negocio deben ser resueltos antes de sufrir pérdidas significativas o inaceptables?, en gran parte la longitud del tiempo de recuperación depende de la naturaleza del negocio o del servicio interrumpido.
- ¿Cuál es la clasificación de riesgos de los sistemas?, esto involucra una determinación de riesgos basados en el impacto derivado del periodo de tiempo crítico de recuperación como de la probabilidad de que ocurra una interrupción adversa.

Clasificación de las operaciones y análisis de criticidad de los sistemas

Clasificar los riesgos de los sistemas, puede contener la siguiente clasificación:

Clasificación	**Descripción**
Crítico	Las funciones pueden realizarse sólo si las capacidades se reemplazan por otras idénticas – No pueden reemplazarse por métodos manuales. – Muy baja tolerancia a interrupciones. – Muy ALTO COSTO de interrupción.
Vitales	Pueden realizarse manualmente por periodo breve – Un poco más de tolerancia a interrupciones vs. Críticos – Costo de interrupción un poco más bajos, sólo si son restaurados dentro de un tiempo determinado (5 ó menos días).
Sensitivos	– Las funciones pueden realizarse manualmente por un periodo prolongado, a un costo tolerable. - Proceso manual difícil, requiriendo personal adicional.
No Críticos	- Las funciones pueden interrumpirse por tiempos prolongados a un costo pequeño o nulo. – No requiere de esfuerzos para actualizarse.

Tabla 8-17. Clasificación de las operaciones y análisis de criticidad de los sistemas.

Capítulo 9

SEGURIDAD EN BASE DE DATOS

Distinguir las principales vulnerabilidades en seguridad en el entorno de trabajo reforzará, tomando las medidas necesarias para su corrección, el total aprovechamiento de los elementos y bondades ofrecidos por las RDBMS.

9.1 SEGURIDAD DE LA INFORMACIÓN.

¿Por qué es importante?

Una base de datos con un bajo nivel de seguridad compromete no solamente a la base de datos misma, sino también al sistema operativo y a otros sistemas relacionados. Para proteger el acceso a la información sensible y a los activos digitales de una organización. Los sistemas de bases de datos son sistemas extremadamente complejos y con gran dificultad para *configurar* y *asegurar*.

La protección del sistema operativo y de los servicios de red en un servidor de bases de datos tiene una importancia crítica. Las bases de datos son la *base* de los nuevos negocios electrónicos, de los sistemas de planeación empresarial de recursos (ERP) y de otros sistemas críticos de negocios.

9.2. LA SEGURIDAD EN UNA BASE DE DATOS.

Aspectos de seguridad

- Secrecía y confidencialidad:
    - Los datos deben ser protegidos de *liberación* no autorizada:
        - Mediante recuperación directa o inferencia lógica.
        - Mediante lectura de usuarios no autorizados.
- Precisión, integridad y autenticidad:
    - Los datos deben ser protegidos de modificación accidental o maliciosa (considerando incluso la inserción de datos falsos o la destrucción de los mismos).
    - El origen de los datos debe ser verificable
- Disponibilidad y recuperabilidad:
    - Los sistemas de bases de datos deben mantenerse operando y recuperarse en caso de pérdida de datos.

Problemas de seguridad

La seguridad en bases de datos comprende medidas para evitar:

- Liberación inapropiada de información (pérdida de secrecía o confidencialidad).
- Modificación inapropiada de datos (pérdida de integridad).
- Negación de servicio (pérdida de disponibilidad).
- Robo o fraude.

Alcance de un dba

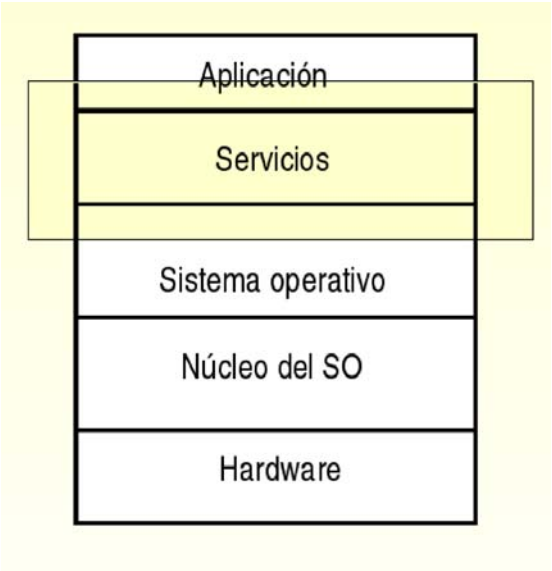


Ilustración 9-1. Esquema de la relación de una DBMS con los elementos de cómputo.

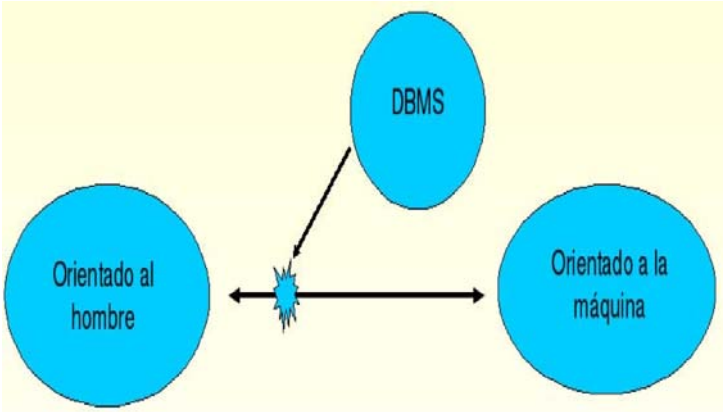


Ilustración 9-2. Esquema de la relación de una DBMS conforme su orientación al hombre o máquina.

Problemas de seguridad

Un manejador de base de datos debe proteger sus datos de:

- Accidentes, por ejemplo errores de captura o de programación.
- Uso malicioso de la base de datos.

- Fallas de hardware o software que corrompen datos.

Tipos de amenazas

- **Accidentales** o no fraudulentas (sin voluntad de causar daño):
    - Accidentes o desastres naturales.
    - Errores o bugs en hardware o software.
    - Errores humanos.
- **Intencionales** o fraudulentas (con voluntad explicita de causar daño)
    - Involucran a dos tipos de usuarios:
        - Usuarios autorizados que abusan de sus privilegios y autoridad.
        - Agentes hostiles.

Consecuencias

- Liberación inapropiada de información.
- Modificación inapropiada de datos.
- Negación de servicio.

Requerimientos

Los requerimientos mínimos necesarios que debe cumplir cualquier programa que se diga RDBM son:

- Autenticación de usuarios.
- Protección de acceso impropio.
- Integridad de la base de datos.
- Administración y protección de datos sensibles.
- Registro de eventos y auditoria.

1. Autenticación de Usuarios

Necesidad de identificar de forma única a los usuarios de la base de datos.

Puede ser realizada por:

- Sistema operativo.
- DBMS.

2. Protección de acceso impropio

Permitir acceso a los objetos de la BD solamente a usuarios autorizados.

Problemática: *Granularidad* más fina que en el caso de los sistemas operativos.

3. Integridad física

- Sistema de respaldo y recuperación.
- Uso de un log o *journal*.
- Instrucciones especiales para aplicar operaciones o cancelarlas.
    - Con puntos de inicio y de control (**`commit`**)

4. Administración y protección de datos sensibles

Datos sensibles:

- Datos que no deben ser hechos públicos.

Factores que pueden hacer sensibles a los datos:

- Inherentemente sensible.
- El valor mismo debe protegerse (sea declarado sensible o no).
- Proviene de una fuente sensible.
- Debe protegerse la fuente de la que procede.
- Se ha declarado sensible.
- Se derivan de un registro o un atributo sensible.

Valor de la información

Puede haber bases de datos:

- Solamente con datos sensibles
- Solamente con datos públicos
- Con ambos tipos de datos

Integridad de datos

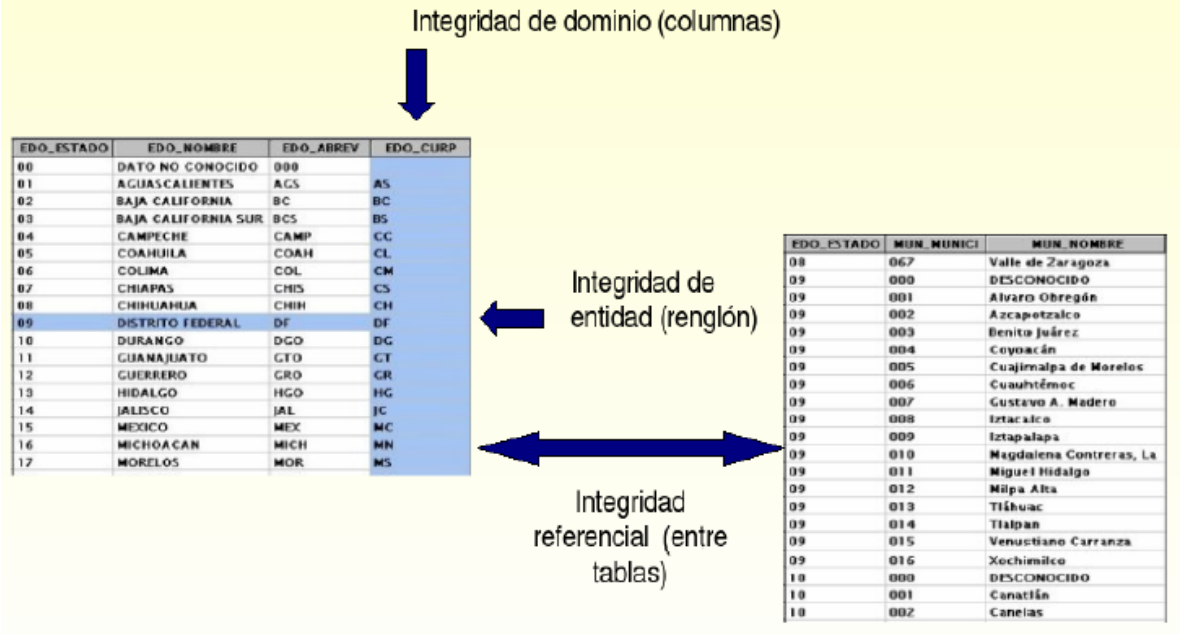
Expectativa de calidad de datos:

- Los datos tienen integridad si su calidad alcanza o supera los requerimientos de calidad que los usuarios *esperan* de ellos.
- Protección contra la modificación ***impropia*** de datos.
- Protección contra la modificación ***no autorizada*** de los datos.

Tipos de integridad

- Integridad de dominio.
    - Requiere que un conjunto de valores sean válidos para una columna y especifica si se permiten valores nulos. Se implementa mediante la verificación de validez y se puede forzar restringiendo el tipo de datos, formato o rango de valores permitidos en una columna.

- Integridad de entidad.
    - Requiere que todos los renglones de una tabla tengan un identificador único, conocido como llave primaria. El valor de una llave primaria puede cambiarse dependiendo del nivel de integridad requerido entre la llave primaria y otras tablas.
- Integridad referencial.
    - Asegura que las relaciones entre la llave primaria (en una tabla referenciada) y las llaves foráneas (en una tabla referenciante) siempre se mantenga.
    - Un renglón en una tabla referenciada no puede ser borrado y si un una llave foránea hace referencia a ese renglón, tampoco la llave primaria puede modificarse.



EDO_ESTADO	EDO_NOMBRE	EDO_ABREV	EDO_CURP
00	DATO NO CONOCIDO	000	
01	AGUASCALIENTES	AGS	AS
02	BAJA CALIFORNIA	BC	BC
03	BAJA CALIFORNIA SUR	BCS	BS
04	CAMPECHE	CAMP	CC
05	COAHUILA	COAH	CL
06	COLIMA	COL	CM
07	CHIAPAS	CHIS	CS
08	CHIHUAHUA	CHIH	CH
09	DISTRITO FEDERAL	DF	DF
10	DURANGO	DGO	DG
11	GUANAJUATO	GTO	GT
12	GUERRERO	GRO	GR
13	HIDALGO	HGO	HG
14	JALISCO	JAL	JC
15	MEXICO	MEX	MC
16	MICHOACAN	MICH	MN
17	MORELOS	MOR	MS

EDO_ESTADO	MUN_MUNICI	MUN_NOMBRE
08	067	Valle de Zaragoza
09	000	DESCONOCIDO
09	001	Alvaro Obregón
09	002	Azcapotzalco
09	003	Benito Juárez
09	004	Coyoacán
09	005	Cuajimalpa de Morelos
09	006	Cuauhtémoc
09	007	Gustavo A. Madero
09	008	Iztacalco
09	009	Iztapalapa
09	010	Magdalena Contreras, La
09	011	Miguel Hidalgo
09	012	Milpa Alta
09	013	Tláhuac
09	014	Tlalpan
09	015	Venustiano Carranza
09	016	Xochimilco
10	000	DESCONOCIDO
10	001	Canatlán
10	002	Canelas

Ilustración 9-3. Esquema de los tipos de integridad en DBMS.

Tipo de integridad	Tipo de restricción
Dominio	• Verificación de valor nulo • Valor por omisión (default) • Verificación de valor (check)
Entidad	• Llave primaria (primary key) • Valor único (unique)
Referencial	• Llave foránea (foreign key)

Tabla 9-1. Tipo de integridad y restricciones

Forzando la integridad

- Integridad declarativa.
    - Los criterios de integridad se declaran en las definiciones de los objetos.
    - Se implementa mediante el uso de restricciones declarativas que se definen directamente en tablas y columnas:
        - Restricciones de llave.

        - Valores por omisión.
        - Valores de verificación.
- Integridad procedimental
    - Criterios definidos en *scripts*.
    - Se implementa mediante el uso de procedimientos almacenados y triggers.
    - Se puede implementar en el cliente o en el servidor.

Control de acceso

Control de acceso discrecional (DAC)

Medio por el que se restringe el acceso a los objetos a usuarios específicos o grupos de usuarios. El propietario de un objeto puede otorgar privilegios de acceso a otros usuarios sobre el mismo objeto, directamente o indirectamente.

Privilegios

- Medio por el que SQL implementa el DAC.
- Se conceden privilegios por medio de una instrucción **`GRANT`** y se usan para especificar una acción permitida sobre un objeto específico.

Vistas

- Los usuarios tienen acceso a la vista, pero no a la tabla base.
- Las vistas son apropiadas para restricciones de columna.
- Soportan herramientas de acceso y consultas ad-hoc.
- Están especificadas en SQL – Son independientes del DBMS.
- Se soporta la actualización.

9.3. HERRAMIENTAS DE APOYO A LA SEGURIDAD

Mejores prácticas

- Usar un sistema de detección de intrusos, especialmente en servidores de bases de datos en línea y de alto riesgo.
- Cambiar las contraseñas de las cuentas creadas por omisión durante la instalación. Asignar contraseñas fuertes a las mismas.
- Deshabilitar las cuentas de invitado y las cuentas de demostración o ejemplo definidas durante la instalación. Eliminar estas cuentas en las bases de datos de producción.
- Mantener actualizado el DBMS con las versiones más recientes del software y de los parches de seguridad liberados por el fabricante del software.
- No permitir que las aplicaciones consulten o manipulen directamente la base de datos mediante instrucciones **`SELECT`**, **`INSERT`**, **`UPDATE`** o **`DELETE`**. Usar procedimientos almacenados en su lugar.
- En las aplicaciones, restringir la ejecución de instrucciones de SQL dinámico.

- Impedir que las aplicaciones acepten instrucciones de SQL de los usuarios y las ejecuten sobre la base de datos.
- Hacer que los usuarios consulten los datos mediante vistas en lugar de otorgarles acceso a las tablas base.
- Habilitar la auditoria de acceso al sistema operativo y al servidor de base de datos. Revisar el registro de auditoria buscando los eventos fallidos de acceso y buscar tendencias con la finalidad de detectar posibles intrusos.
- Monitorear cuidadosamente los registros (logs) de error y de eventos y disparar automáticamente alertas relacionadas con la seguridad y con errores. Proteger los archivos de registro (log) mediante permisos apropiados de sistema operativo.
- En ambiente de bases de datos distribuidas, eliminar el acceso a los servidores que no se utilicen. Utilizar cuentas de acceso con mínimos privilegios para los servidores relacionados.
- Almacenar los archivos utilizados para carga masiva de datos o por lotes (batch) en un directorio con los permisos apropiados. Eliminar los archivos una vez que hayan sido utilizados.
- Para asegurar la replicación de datos sobre Internet o sobre una red de área amplia (WAN), implementar una red privada virtual (VPN).
- Definir y aplicar una política de respaldo periódico. Almacenar los medios de respaldo en un lugar seguro. Realizar regularmente restauraciones de la base de datos a partir de los respaldos.

9.4 FORTALEZA DE LAS CONTRASEÑAS. CASO PRÁCTICO

Usualmente la intrusión a los sistemas por individuos externos es ocasionada por contraseñas débiles.

Una aplicación para averiguar si los passwords de nuestros usuarios son débiles, y usualmente usada por los intrusos es John The Ripper[1]. A continuación se describe la forma de uso en un sistema operativo tipo Linux.

Obtener John The Ripper

```
$ wget http://www.openwall.com/john/f/john-1.7.0.2.tar.gz
$ wget http://www.openwall.com/signatures/openwall-signatures.asc
$ wget http://www.openwall.com/john/f/john-1.7.0.2.tar.gz.sign
```

Código 9-1. Uso del comando wget para descargar la aplicación de John The Ripper.

[1] John The Ripper es un programa de criptografía que aplica fuerza bruta para descifrar contraseñas. Es capaz de romper varios algoritmos de cifrado o hash, como DES, SHA-1 y otros. Es una herramienta de seguridad muy popular, ya que permite a los administradores de sistemas comprobar que las contraseñas de los usuarios son suficientemente buenas. John the Ripper es capaz de autodetectar el tipo de cifrado de entre muchos disponibles, y se puede personalizar su algoritmo de prueba de contraseñas. Eso ha hecho que sea uno de los más usados en este campo.

Importar signaturas

```
$ gpg --import openwall-signatures.asc
gpg: /home/usuario/.gnupg/trustdb.gpg: se ha creado base de datos de confianza
gpg: clave 295029F1: clave pública "Openwall Project <signatures@openwall.com>" importada
gpg: Cantidad total procesada: 1
gpg:               importadas: 1  (RSA: 1)
```

Código 9-2. Uso del comando gpg para obtener la signatura en la biblioteca del home del usuario.

Verificar signaturas

```
$ gpg --verify john-1.7.0.2.tar.gz.sign john-1.7.0.2.tar.gz
gpg: Firmado el jue 23 mar 2006 08:23:17 CST usando clave RSA ID 295029F1
gpg: Firma correcta de "Openwall Project <signatures@openwall.com>"
```

Código 9-3. Verificar que el archivo descargado este su signatura en la biblioteca de confianza.

Extraer e instalar.

```
$ tar zxvf john-1.7.0.2.tar.gz
Instalar
$ cd john-1.7.0.2/src
$ make
$ make clean generis
$ cd ../run
$ ./john –test
```

Código 9-4. Instalación de John The Ripper en el Sistema Operativo.

Comprobar fortaleza de las contraseñas.

```
$ ./john /etc/shadow
```

Código 9-5. Ejecución de John The Ripper para comprobar la fortaleza de las contraseñas.

Capítulo 10

PERFORMANCE AND TUNNING

Entender la razón de reconfigurar una RDMS, usar herramientas usadas para reconfigurar, optimizar la asignación de memoria, ver como los parámetros relacionados con recursos afectan la asignación de memoria, entender las principales opciones de configuración. Estos conceptos a pesar de mostrarse principalmente para Sybase / SQL Server reflejan las características a optimizar en las diferentes RDBMS.

10.1 PERFORMANCE DEL SERVIDOR

Modificando la configuración predeterminada del SQL Server

Bases de la configuración del SQL Server

¿Por qué importa la configuración?

Los valores de los parámetros de configuración determinan el uso de los recursos del sistema, la configuración del SQL Server afecta directamente su rendimiento, los valores de configuración predeterminados, son mínimos y sólo permiten arrancar el SQL Server, los administradores del sistema pueden reconfigurar muchos valores, los oficiales de seguridad del sistema controlan: Password expiration, Updates of system tables, Audit queue size.

¿Qué puede ser configurado en el SQL Server?

Backup/recovery; Cache manager, Disk I/O, Network, Operating system resources, Memory, Processors, User environment, Lock manager, Parallel query.

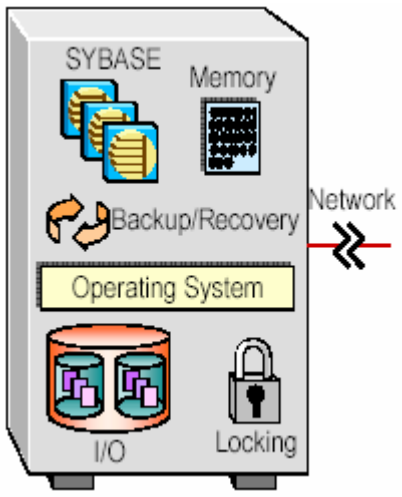


Ilustración 10-1. Elementos configurables del SQL Server.

Opciones de configuración estáticas y dinámicas

- Los parámetros de configuración son estáticos o dinámicos.
- Los parámetros dinámicos afectan inmediatamente después de ejecutarse **`sp_configure`**.
- Los parámetros estáticos requieren que SQL Server reasigne memoria.
- Al cambiar los parámetros estáticos se requiere reinicializar SQL Server.

Archivo de Configuración

SQL usa un archivo de configuración que asigna los recursos del servidor. En tiempo real, SQL Server utiliza **`$SYBASE/<SERVERNAME>.cfg`** a menos que se especifique otro archivo. SQL Server genera un nuevo archivo de configuración cada vez que un valor es cambiado usando **`sp_configure`**.

Ilustración 10-2. Generación de archivos de configuración en SQL Server.

Este es un ejemplo de un archivo de configuración creado en la instalación; el valor que más se da es **`DEFAULT`**. Los parámetros resaltados son los integrados en la versión 11.5

```
################################################################
#
#       Configuration File for the Sybase SQL Server
#       Please read the System Administration Guide (SAG)
#       before changing any of the values in this file.
#
################################################################
[Configuration Options]
[General Information]
[Backup/Recovery]
        recovery interval in minutes = DEFAULT
        print recovery information = DEFAULT
        tape retention in days = DEFAULT
[Cache Manager]
        number of oam trips = DEFAULT
        number of index trips = DEFAULT
        procedure cache percent = DEFAULT
        memory alignment boundary = DEFAULT
        global async prefetch limit = DEFAULT
[Named Cache:default data cache]
        cache size = DEFAULT
        cache status = default data cache
        cache replacement policy = DEFAULT
[Meta-Data Caches]
        number of open databases = DEFAULT
        number of open objects = DEFAULT
        open object spinlock ratio = DEFAULT
        number of open indexes = DEFAULT
        open index hash spinlock ratio = DEFAULT
        open index spinlock ratio = DEFAULT
[Disk I/O]
        allow sql server async i/o = DEFAULT
        disk i/o structures = DEFAULT
        page utilization percent = DEFAULT
        number of devices = DEFAULT
        disable character set conversions = DEFAULT
[Network Communication]
```

```
        default network packet size = DEFAULT
        max network packet size = DEFAULT
        remote server pre-read packets = DEFAULT
        number of remote connections = DEFAULT
        allow remote access = DEFAULT
        number of remote logins = DEFAULT
        number of remote sites = DEFAULT
        max number network listeners = DEFAULT
        tcp no delay = DEFAULT
        allow sendmsg = DEFAULT
        syb_sendmsg port number = DEFAULT
[O/S Resources]
        max async i/os per engine = DEFAULT
        max async i/os per server = DEFAULT
[Parallel Query]
        number of worker processes = DEFAULT
        memory per worker process = DEFAULT
        max parallel degree = DEFAULT
        max scan parallel degree = DEFAULT
[Physical Resources]
[Physical Memory]
        total memory = DEFAULT
        additional network memory = DEFAULT
        lock shared memory = DEFAULT
        shared memory starting address = DEFAULT
        max SQL text monitored = DEFAULT
[Processors]
        max online engines = DEFAULT
        min online engines = DEFAULT
[SQL Server Administration]
        default database size = DEFAULT
        identity burning set factor = DEFAULT
        allow nested triggers = DEFAULT
        allow updates to system tables = 1
        print deadlock information = DEFAULT
        default fill factor percent = DEFAULT
        number of mailboxes = DEFAULT
        number of messages = DEFAULT
        number of alarms = DEFAULT
        number of pre-allocated extents = DEFAULT
        event buffers per engine = DEFAULT
        cpu accounting flush interval = DEFAULT
        i/o accounting flush interval = DEFAULT
        sql server clock tick length = DEFAULT
        runnable process search count = DEFAULT
        i/o polling process count = DEFAULT
        time slice = DEFAULT
        deadlock retries = DEFAULT
        cpu grace time = DEFAULT
        number of sort buffers = DEFAULT
        number of large i/o buffers = DEFAULT
        size of auto identity column = DEFAULT
        identity grab size = DEFAULT
        lock promotion HWM = DEFAULT
        lock promotion LWM = DEFAULT
        lock promotion PCT = DEFAULT
        housekeeper free write percent = DEFAULT
        partition groups = DEFAULT
        partition spinlock ratio = DEFAULT
        allow resource limits = DEFAULT
        number of aux scan descriptors = DEFAULT
        SQL Perfmon Integration = DEFAULT
        allow backward scans = DEFAULT
[User Environment]
        number of user connections = DEFAULT
        stack size = DEFAULT
        stack guard size = DEFAULT
        permission cache entries = DEFAULT
        user log cache size = DEFAULT
        user log cache spinlock ratio = DEFAULT
[Lock Manager]
```

```
        number of locks = DEFAULT
        deadlock checking period = DEFAULT
        freelock transfer block size = DEFAULT
        max engine freelocks = DEFAULT
        address lock spinlock ratio = DEFAULT
        page lock spinlock ratio = DEFAULT
        table lock spinlock ratio = DEFAULT
[Security Related]
        systemwide password expiration = DEFAULT
        audit queue size = DEFAULT
        curread change w/ open cursors = DEFAULT
        allow procedure grouping = DEFAULT
        select on syscomments.text = DEFAULT
        auditing = DEFAULT
        current audit table = DEFAULT
        suspend audit when device full = DEFAULT
        max roles enabled per user = DEFAULT
        unified login required = DEFAULT
        use security services = DEFAULT
        msg confidentiality reqd = DEFAULT
        msg integrity reqd = DEFAULT
        msg replay detection reqd = DEFAULT
        msg origin checks reqd = DEFAULT
        msg out-of-seq checks reqd = DEFAULT
        secure default login = DEFAULT
        dump on conditions = DEFAULT
[Extended Stored Procedure]
        esp unload dll = DEFAULT
        esp execution priority = DEFAULT
        esp execution stacksize = DEFAULT
        xp_cmdshell context = DEFAULT
        start mail session = DEFAULT
[Error Log]
        event logging = DEFAULT
        log audit logon success = DEFAULT
        log audit logon failure = DEFAULT
        event log computer name = DEFAULT
[Rep Agent Thread Administration]
        enable rep agent threads = DEFAULT
        maximum dump conditions = DEFAULT
[Component Integration Services]
        enable cis = DEFAULT
        cis connect timeout = DEFAULT
        cis bulk insert batch size = DEFAULT
        max cis remote connections = DEFAULT
        max cis remote servers = DEFAULT
        cis packet size = DEFAULT
        cis cursor rows = DEFAULT
        cis rpc handling = DEFAULT
```

Código 10-1. Archivo de Configuración de SQL Server.

Se usa **`sp_configure`** para ver los valores predeterminados de la plataforma.

```
1> sp_configure 'Physical Memory'
2> go
Group: Physical Memory

Parameter Name                  Default     Memory Used Config Value Run Value
------------------------------- ----------- ----------- ------------ ----------
additional network memory             0           0            0          0
lock shared memory                    0           0            0          0
shared memory starting address        0           0            0          0
total memory                       7500           0         7500       7500

(return status = 0)
```

Ilustración 10-3. Uso de sp_configure para obtener información sobre la memoria de la plataforma instada.

- Verificar permisos y protección del archivo de configuración.
- Borrar archivos inutilizables, para evitar que el sistema de archivos se llene.
- Especificar una bandera en la línea de comandos al **`dataserver`** (en el archivo **`RUNSERVER`**) para usar un archivo de configuración específico:

```
/work/sybase/ASE/bin/dataserver -d/devices/master.dat \
-sSYBASE -e/work/sybase/ASE/install/errorlog \
-c/work/sybase/ASE/SYBASE.010 &
```

Código 10-2. Uso de dataserver del archive RUNSERVER para usar un archive de configuración específico.

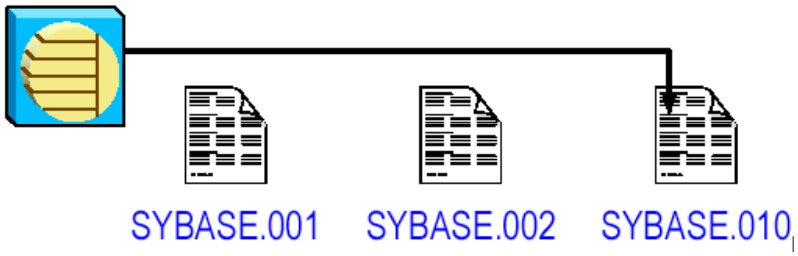


Ilustración 10-4. Indicar al ASE que use un archivo de configuración específico.

Respaldo de un archivo de configuración

Al reinicializar, el servidor crea el archivo **`$SYBASE/<SERVERNAME>.bak`**. Este archivo contiene los valores configurados hasta ese momento. El archivo no es renombrado como archivo de configuración; es sobrescrito en cada arranque.

Ilustración 10-5. Trato del archivo de configuración al arranque en SQL Server.

Este archivo proporciona una copia de respaldo del archivo de configuración en el evento en el que accidentalmente se sobrescribió.

Cuando se usan los archivos de configuración.

Los archivos de configuración con diferentes configuraciones pueden ser utilizados para muchas situaciones:

- Requerimientos específicos de procesamiento.
- Estandarizar una configuración en la empresa.
- Recuperación de una Base de Datos.

Configurando SQL Server usando sp_configure

Se usa el comando **sp_configure** para reconfigurar SQL Server. Se puede reconfigurar SQL Server (1) interactivamente usando **sp_configure** o (2) editando directamente el archivo de configuración. **sp_configure** (3) también para leer o escribir el archivo de configuración.

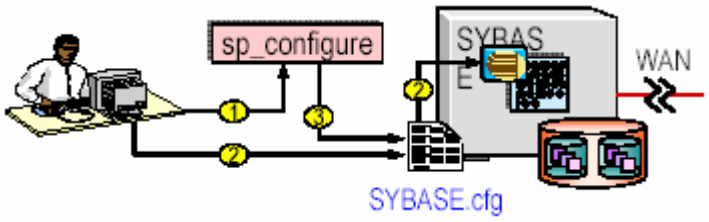


Ilustración 10-6. Esquema de uso del sp_configure para configurar la configuración de SQL Server.

sp_configure despliega parámetros de configuración en grupos jerárquicos, muestra valores de configuración y de ejecución.

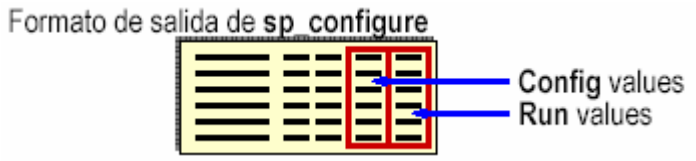


Ilustración 10-7. Formato de salida de sp_configure en SQL Server.

Realiza validación de opciones, cambia los valores de configuración:

```
sp_configure "number of locks", 1000
sp_configure "recovery interval", 3
```

Código 10-3. Uso de sp_configure en la configuración de number of locks y recovery intervals en SQL Server.

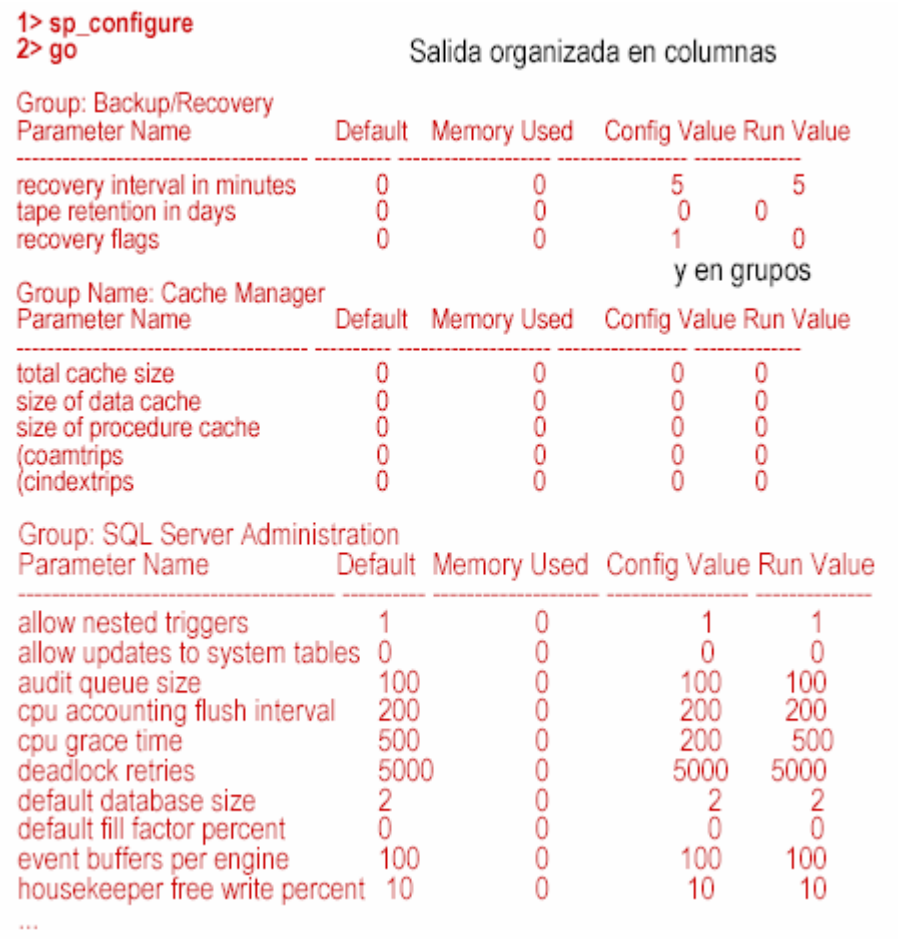


Ilustración 10-8. Salida de sp_configure organizado en columnas y en grupos en SQL Server.

Usando sp_configure y el archivo de configuración

`sp_configure "configuration file"` despliega el nombre del archivo de configuración actualmente cargado.

```
sp_configure "configuration_ file" [,0, "<sub-command>","<fileName>" ]
```

Código 10-3. Sintaxis de sp_configure usando el archivo de configuración en SQL Server.

Subcomandos válidos: **`read`**, **`verify`**, **`write`** y **`restore`**.

- **`sp_configure`**. Despliega los parámetros (dependiendo del nivel de desplegado del usuario).
- **`sp_configure [group name]`**. Despliega parámetros del grupo especificado.
- **`sp_configure [optname [, optvalue]]`**. Cambia un parámetro en el archivo de configuración y en **`sysconfigures`** (si es dinámico).
- **`sp_configure "configuration file", [0,"<sub-cmd>", "<filename>"]`**. Permite leer, escribir, o verificar el archivo de configuración.

Los parámetros del archivo de configuración son agrupados por función, haciendo más fácil su interpretación. Se pueden preparar diferentes archivos de configuración para diferentes tipos de

procesamiento. La nueva opción del **`sp_configure, configuration file`**, permite leer y escribir el archivo de configuración.

Configurando la memoria del SQL Server

Asignación de memoria del SQL Server

El total de memoria física incluye:

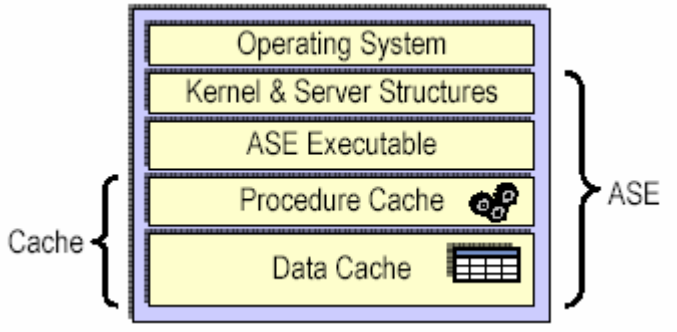


Ilustración 10-9. Distribución de la memoria en el sistema de SQL Server.

Asignación de Memoria del SQL Server al Arrancar

SQL Server preasigna memoria en esta secuencia:

1. Ejecutables del SQL Server.
2. Memoria Estática.
3. Memoria para parámetros de usuario.
4. Estructuras de datos sin cache.

La memoria que queda se divide entre:

5. Cache de datos.
6. Cache de procedimientos (basado en el valor del parámetro de configuración **`procedure cache size configuration`**).

Maximizando la memoria del SQL Server

La memoria que está disponible, todos los recursos que el SQL Server son para buffers internos y caches. Teniendo una adecuada memoria disponible para caches reduce el número de veces que el SQL Server tiene que leer datos del disco por información estática o planes de procedimientos compilados. No hay fallas en el rendimiento por configurar el SQL Server para que utilice la memoria máxima disponible en la computadora. Sin embargo, hay que asegurarse de evaluar otras necesidades de memoria en el sistema, y estar seguro que SQL Server usa sólo la memoria disponible que queda. SQL Server no reinicia si no puede tomar la memoria en la cuál esta configurado.

Pasos para maximizar la memoria en el SQL Server

1. Determine la cantidad total de memoria física en su computadora.

2. Restar la memoria requerida por el sistema operativo de la memoria física total.

3. Si la máquina no es sólo para SQL Server, restar la memoria que es utilizada para otras aplicaciones.

    - Por ejemplo, restar la memoria que utilizan las aplicaciones cliente ejecutándose en una máquina SQL Server.
    - Los sistemas de ventanas, como X Windows, requieren otro tanto de memoria y que puede interferir con el rendimiento del SQL Server.

4. Restar la memoria que se quiera asignar al parámetro de configuración **`additional network memory`**.

Verificando la configuración predeterminada de memoria para el SQL Server.

Monitoreando la memoria usando **`dbcc memusage`**. La salida de **`dbcc`** es de una instalación normal en la cuál **`total memory`** es configurada a 7500 páginas.

```
1> dbcc traceon (3604)
2> go
1> dbcc memusage
2> go
Memory Usage:
```

	Meg.	2K Blks	Bytes
Configured Memory:	14.6484	7500	15360000
Code size:	3.3339	1707	3495856
Kernel Structures:	2.1106	1081	2213098
Server Structures:	3.7646	1928	3947480
Cache Memory:	4.2422	2172	4448256
Proc Buffers:	0.0394	21	41344
Proc Headers:	1.1543	591	1210368

Ilustración 10-10. Uso de dbcc memusage para obtener la memoria total configurada en SQL Server.

Monitoreando la memoria usando el Error Log. Examinando los mensajes de arranque en el error log se puede encontrar salidas como la siguiente:

```
00:95/11/10 08:47:20.94
server Number of proc buffers allocated: 543.
00:95/11/10 08:47:21.12
server Number of blocks left for proc headers: 591.
00:95/11/10 08:47:21.12
server Memory allocated for the default data cache cache: 4078 Kb
```

Código 10-4. Salida del error log en lo que se refiere a la memoria en SQL Server.

Identificando las primeras configuraciones para ajustar los defaults.

Primeros ajustes a la memoria del SQL Server.

- Configurar las conexiones de usuario.
- Calcular el tamaño del cache de procedimientos.
- Determinar el tamaño del cache de datos.

¿Cuántas conexiones se configuran?

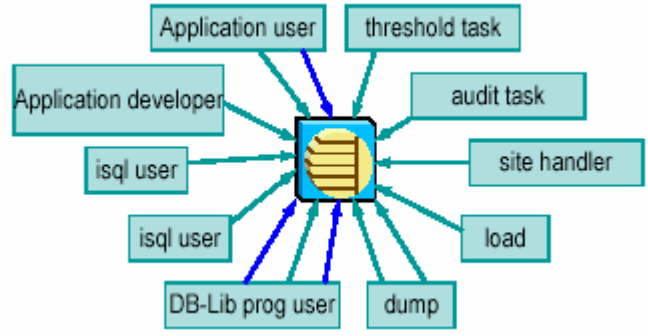


Ilustración 10-11. Conexiones a SQL Server.

La opción **`user connections`** se fija el número máximo de conexiones de usuario que pueden conectarse al SQL Server. El sistema operativo puede limitar el número total de conexiones.

Recuperando el cache de datos y de procedimientos perdido

1. Para recuperar el cache de datos y procedimientos, hay que configurar **`total memory`** agregando el **`número calculado de páginas perdidas del cache + 1.5%`** (incrementando el porcentaje si usuarios de la aplicación son agregados, acomodar la demanda extra en memoria)

```
(Memoria total actual + número de páginas para conexiones + sobrecarga)

12000 + 702 + (702 * 0.015) = 12712
```

```
1> sp_configure "total memory", 12712
2> go
00:00000:00007:1998/01/20 06:55:21.83 server  Configuration file '/
work1/sybase/ASE_GA/ASE/SYBASE.cfg' has been written and the
previous version has been renamed to
'/work1/sybase/ASE_GA/ASE/SYBASE.069'.
00:00000:00007:1998/01/20 06:55:21.93 server  The configuration option
'total memory' has been changed by 'sa' from '12000' to '12712'.
Parameter Name              Default     Memory Used Config Value
     Run Value
--------------------------------- ------------ ------------ ----------
    ----------
total memory                12000       24000       12712
        12000

(1 row affected)
Configuration option changed. The SQL Server must be rebooted before
the change
in effect since the option is static.
(return status = 0)
```

Ilustración 10-12. Uso de sp_configure para configurar t o t a l m e m o r y agregando un número calculado de páginas perdidas del cache.

2. Cerrar y reiniciar SQL Server

3. Verificar la memoria asignada ejecutando **`dbcc memusage`**

```
1> dbcc memusage
2> go
Memory Usage:
```

	Meg.	2K Blks	Bytes
Configured Memory:	24.8281	12712	26034176
Code size:	5.7627	2951	6042600
Kernel Structures:	2.6616	1363	2790872
Server Structures:	5.4238	2777	5687248
Cache Memory:	8.6465	4427	9066496
Proc Buffers:	0.0760	39	79704
Proc Headers:	2.2539	1154	2363392

Ilustración 10-13. Verificación de la memoria asignada usando dbcc memusage en SQL Server.

Discutiendo otras opciones de configuración

Otras opciones que requieren memoria, entre las variables de configuración que requieren memoria destacan:

- **`number of open databases`** (aproximadamente 35K cada una)
- **`number of devices`** (aproximadamente 512 bytes cada una)
- **`number of open objects`** (< 1000 bytes cada una)
- **`number of locks`** (<100 bytes cada una)

Calculando el cache

- Configuración del Kernel y el servidor.
- Calcular el tamaño total del cache de procedimientos necesario.
- Determinar el tamaño total del cache de datos necesario.
- Fijar el parámetro de configuración **`total memory`** correctamente.
- Fijar el parámetro de configuración **`procedure cache size`** correctamente.
- Cerrar y reiniciar SQL Server.

Requerimientos para el cache de procedimientos

Calcular el cache del procedimiento que sea por lo menos una copia de cada procedimiento principal que pueda adecuarse en el cache.

El siguiente algoritmo es un buen comienzo:

```
procedure cache size = max(# of concurrent users) * (size of largest plan) * 1.25
```

Código 10-5. Algoritmo base para calcular el cache de procedimiento en SQL Servers.

Después, modificar el cálculo usando el tamaño promedio del plan de todos los procedimientos almacenados:

```
Minimum procedure cache needed = (# of main procedures) * (avg. plan size) * 1.25
```

Código 10-6. Algoritmo para calcular el cache de procedimientos mínimo necesitado en SQL Server.

Determinar el tamaño del plan de procedimientos

- Configurar el cache de procedimientos para 40 usuarios; siguiendo lo siguiente:
  - Usar **`dbcc memusage`** para determinar el tamaño del plan de los procedimientos almacenados más grandes.

```
Database Id: 6
Object Id: 1232008685
Object Name: upd_custord
Version: 1
Uid: 1
Type: stored procedure
Number of trees: 0
Size of trees: 0.000000 Mb, 0.000000 bytes, 0 pages
Bytes lost for alignment 0 (Percentage of total: 0.000000)
Number of plans: 1
Size of plans: 0.171157 Mb, 179471.000000 bytes, 88 pages
Bytes lost for alignment 22830 (Percentage of total: 12.720718)
```

Ilustración 10-14. Uso de dbcc memusage para determinar el tamaño del plan de procedimientos en SQL Server.

Calculando el cache de procedimientos

- Determinar el requerimiento del cache de procedimientos.

```
max(#users)*(size of largest plan)*1 .25
```

Código 10-7. Algoritmo para determinar el tamaño del cache de procedimientos en SQL Server.

```
(40 * 179,471 )*1 .25 = 8,973,550 bytes (4,382 2K pages)
```

Código 10-8. Uso del algoritmo para determinar el tamaño de cache de procedimientos en SQL Server.

- Determinar el requerimiento del cache total

```
Procedure cache = 4,382 2K pages (20%) - (1% =(4,382 / 20) = 219.1)
Data cache = 17,528 2K pages (80%) - (80% = (80 *219.1) = 17,528)
Total cache = (4,382 + 17,528) = 21,910 2K pages
```

Código 10-9. Proceso para determinar el requerimiento de cache total.

- Determinar los requerimientos de memoria adicional.
  - Determinar el tamaño actual del cache configurado:

Cache Memory:	4.2520	2177	4458496
Proc Buffers:	0.0395	21	41420
Proc Headers:	1.1562	592	1212416
		2790	5,712,332

Ilustración 10-15. Determinar el tamaño del cache configurado.

```
Required additional memory = (required cache –current cache)
```
Código 10-10. Algoritmo para calcular la memoria adicional requerida en SQL Server.

```
Required additional memory = 19,120 2K pages (21,910 - 2,790)
```
Código 10-11. Uso de algoritmo para calcular la memoria adicional requerida en SQL Server.

- Calcular el requerimiento de memoria total para **sp_configure**

```
Total memory = additional memory + run value
```
Código 10-12. Algoritmo para calcular el nuevo valor de memoria total en SQL Server.

```
Total memory = 27,145 2K pages (19,120 + 8,025)
```
Código 10-13. Uso del algoritmo para calcular el nuevo valor de memoria total en SQL Server.

- Reconfigurar SQL Server.

```
1> sp_configure "total memory", 27145
2> go
Parameter Name   Default   Memory Used Config Value Run Value
---------------- --------- ----------- ------------ -----------
total memory     7500                0        27145        8025

Configuration option changed. The SQL Server must be rebooted before the
change in effect since the option is static.
```

Ilustración 10-16. Reconfigurando la memoria total en SQL Server.

- Cerrar y reiniciar SQL Server y ejecutar **dbcc memusage**.

Tamaño: sp_configure Procedure Cache

- El cache de procedimientos predeterminado es el 20% del cache disponible.
- Esto se cambia usando **sp_configure**.
- Usar esta característica si no es posible acomodar el 20/80.
- El tamaño del cache de procedimientos generalmente incrementa en sistemas de desarrollo.

```
sp_configure "procedure cache", 35
```
Código 10-14. Uso de sp_configure para cambiar el cache de procedimientos en SQL Server.

- Cerrar y reiniciar el servidor para activar el nuevo cache de procedimientos configurado.
- Revisar **dbcc memusage** para verificar la memoria asignada.

Resumiendo

- SQL Server preasigna memoria en el arranque.
- SQL Server primero asigna a los ejecutables, servidor, y estructuras del kernel.
- El tamaño de las estructuras del kernel y del servidor depende de la configuración de recursos, y conexiones de usuario.
- Lo que queda es dividido entre el cache de datos y de procedimientos.

- La división del cache es determinado por el parámetro de configuración **`procedure cache size`**.
- Se usa **`dbcc memusage`** para confirmar la asignación de memoria.

10.2 REFINACIÓN EN LA ASIGNACIÓN DE RECURSOS DE DISCO

Entender el proceso y estrategia para el manejo del almacenamiento, crear y usar segmentos de base de datos, crear y usar particiones en las tablas, estrategias e implementación de colocación de objetos.

Decisiones en el manejo del almacenamiento

Consideraciones importantes con respecto al manejo del almacenamiento:

- Recuperación:
  - Duplicado de discos y/o mantenimiento de logs en dispositivos físicos separados proporcionan dos mecanismos para una completa recuperación.
- Rendimiento:
  - Para tablas y base de datos donde la velocidad en lecturas y escrituras de disco es crucial.
  - Colocar apropiadamente objetos de base de datos en dispositivos físicos mejora el rendimiento.
  - El duplicado de disco alenta la velocidad de escritura del disco.

Bases de datos del sistema y predeterminadas

El programa de instalación inicializa el dispositivo **`master`** y prepara las siguientes estructuras:

- **`master`**, **`model`**, y **`tempdb`**. Las bases de datos **`master`**, **`model`**, y **`tempdb`** se instalan en el dispositivo **`master`**.
- **`sybsystemprocs`**. La base de datos **`sybsystemprocs`** es instalada en el dispositivo que se escoja.
- **`sybsecurity`***. **`sybsecurity`** es instalada sobre su propio dispositivo.
- **`dbccdb`***

Para cada base de datos, tres segmentos son creados: **`system`**, **`default`**, y **`logsegment`**

Discos, dispositivos, segmentos, y particiones

Un dispositivo o disco físico es el hardware actual que almacena datos. Un dispositivo de base de datos es una parte de un disco físico que ha sido inicializado para ser usado por el SQL Server. Es inicializado por el comando **`disk init`**.

Un **`fragmento`** es una asignación inmediata de espacio en un dispositivo de base de datos.

Un **`segmento`** es una etiqueta en una base de datos, que puede ser asociada con uno o más fragmentos de base de datos.

Una **`partición`** es una cadena de página para una tabla.

Determinando una estrategia

Puntos a considerar acerca de como colocar datos:

- Poner bases de datos con requerimientos de rendimiento crítico en dispositivos separados; poner las bases de datos de servidor (**`master`**, **`tempdb`**) en dispositivos separados.
- Poner tablas con uso pesado en discos separados.
- Usar segmentos como necesidad para tablas críticas.
- Usar particiones como necesidad para consultas paralelas.
- Separar datos almacenados del log almacenado.
- Dispositivos duplicados en discos separados.
- Usar segmentos para poner tablas e índices en sus propios discos.
- Las tablas particionadas equilibran el número de dispositivos físicos en un segmento.

Uso de Segmentos

Una base de datos puede estar sobre varios dispositivos.

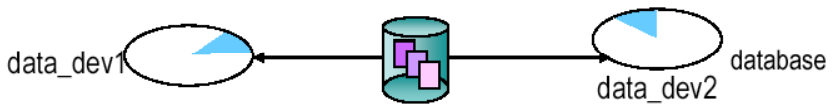


Ilustración 10-17. Esquema de una base de datos almacenada en dos dispositivos en SQL Server.

Una recomendación es poner una tabla de uso pesado en un dispositivo y otra tabla grande en otro dispositivo.

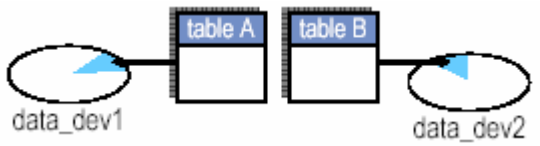


Ilustración 10-18. Esquema de distribución de tablas en diferentes dispositivos en SQL Server.

Para hacer esto, se usan segmentos. Se crean Segmentos de espacio en uno o más dispositivos lógicos para una base de datos.

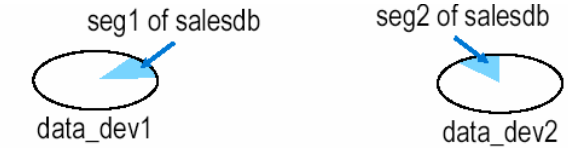


Ilustración 10-19. Representación de segmentos en dispositivos lógicos en SQL Server.

Cuando un segmento ha sido creado, se pueden poner tablas, o índices en ese segmento.

```
create table tableA(. . .) on seg1
create table tableB(. . .) on seg2
```

Código 10-15. Uso de create table para asignar una tabla en un segmento específico.

Para controlar la ubicación de las tablas, se pueden colocar tablas activas e índices para ser cubiertas a través de todo el disco.

¿Por qué usar segmentos?

- Se usa segmentos para controlar la colocación de los objetos de la base de datos y dispositivos.
- Cuando se crea un objeto sobre un segmento, el objeto puede usar todos los dispositivos de base de datos que estén disponibles en el segmento.
- Recordar mejorar el rendimiento con:
  - Dividir grandes tablas en discos.
  - Separar tablas y sus índices nonclustered en discos.
- Controlar el uso del espacio:
  - Usar segmentos para limitar el tamaño de la tabla; una tabla no puede crecer más que su asignación en el segmento.
  - Tomar ventaja del manejador Threshold para monitorear el uso del espacio.

Segmentos Definidos por el Sistema

Cuando una base de datos es creada, el espacio asignado para la base de datos genera tres segmentos: **`system`**, **`default`**, y **`logsegment`**.

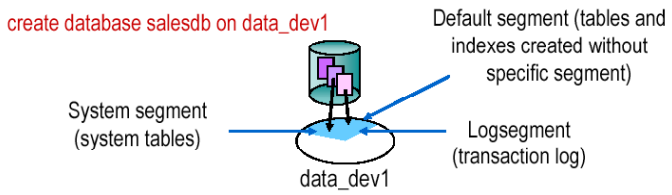


Ilustración 10-20. Esquema de los segmentos system, default y logsegment de una base de datos en un mismo dispositivo en SQL Server.

En el ejemplo anterior, tanto los datos como el log están en el mismo dispositivo

Ejemplo de base de datos: Datos y log en dispositivos separados

```
create database salesdb on data_dev1 = 5 log on log_dev2 = 2
```

Código 10-16. Uso de create database para separar el log de datos y el de dispositivos en dispositivos separados en SQL Server.

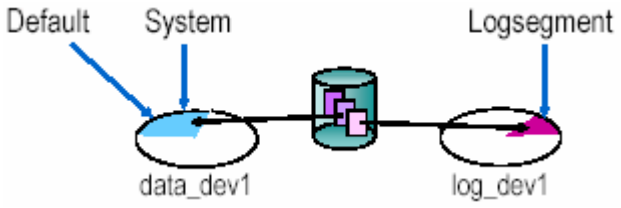


Ilustración 10-21. Esquema de los segmentos system, default y logsegment de una base de datos en dispositivos separados en SQL Server.

Agregando segmentos definidos por el usuario

Para definir un segmento, se ejecute **sp_addsegment**.

```
sp_addsegment segname, dbname, device_name
```

Código 10-17. Sintaxis de sp_addsegment para definir segmentos en SQL Server.

Ejemplo de secuencia:

```
disk init name = "data_dev3",...
go
alter database salesdb on data_dev3 = 1
go
use salesdb
go
sp_addsegment seg1, salesdb, data_dev3
go
```

Código 10-18. Secuencia para crear un segmento en un dispositivo nuevo de una base de datos en SQL Server.

El definir un segmento no asigna espacio adicional; etiqueta el espacio que ya ha sido asignado.

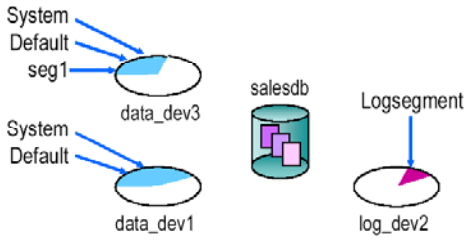


Ilustración 10-22. Esquema de segmentos una vez agregados tanto los segmentos como un nuevo dispositivo en una base de datos en SQL Server.

Después de los comandos **alter database** y **sp_addsegment**, el dispositivo **data_dev3** tiene los segmentos **system**, **default**, y **seg1** de la base **salesdb**.

Borrando segmentos de sistema y predeterminados

Para borrar segmentos hay que asegurarse que los objetos que ponga en un segmento de usuario no compitan con las tablas del sistema y otros objetos. Se ejecuta **sp_dropsegment** para borrar los segmentos **system** y **default**.

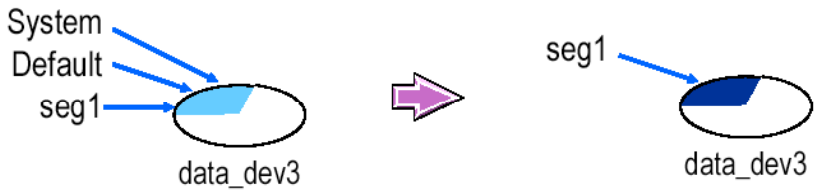


Ilustración 10-23. Esquema que ilustra la eliminación de los segmentos System y Default en SQL Server.

```
sp_dropsegment "system", salesdb, data_dev3
sp_dropsegment "default", salesdb, data_dev3
```

Código 10-19. Eliminación de los segmentos System y Default de un dispositivo en SQL Server.

Creando objetos sobre segmentos

Para colocar un objeto en un segmento, hay que especificar el segmento cuando se crea un objeto.

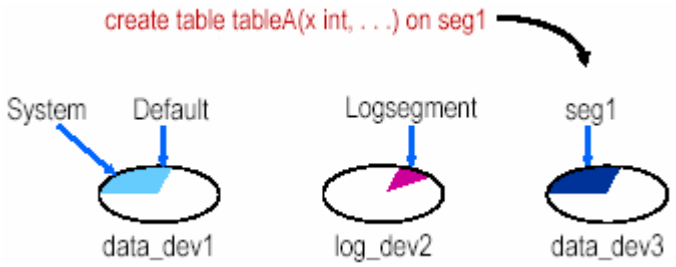


Ilustración 10-24. Ubicar una tabla en un segmento a la hora de su creación en SQL Server.

Si no especifica un nombre de segmento, la tabla es puesta en el segmento **`default`**. Los índices Clustered y sus tablas siempre están en el mismo segmento.

Desplegando información acerca de los segmentos

Se usa **`sp_helpsegment`** para listar todos los segmentos de la base de datos actual. Un ejemplo de salida de **`sp_helpsegment`** es el siguiente:

segment	name	status
0	system	0
1	default	1
2	logsegment	0
3	seg1	0

Ilustración 10-25. Salida de sp_helpsegment para listar todos los segmentos de la base de datos actual en SQL Server.

Los nombres de segmentos son específicos en las base de datos. No pueden ser confundidos con nombres de segmentos iguales en otras bases de datos. Los Segmentos son listados en orden de su creación.

Usando **sp_helpdb <databasename>** se pueden desplegar los segmentos de una base de datos.

```
name   db_size  owner   dbid   created     status
sales 4 MB    sa    5    Oct 16 1992 no options set
device_fragments     size      usage           free kbytes
-------------------- --------  --------------- ----------
data_dev1            2 MB      data only              1376
log_dev2             1 MB      log only               1008
data_dev3            1 MB      data only              1008

device                          segment
------------------------------- ------------------------------
data_dev1                       default
data_dev1                       system
log_dev2                        logsegment
data_dev3                       seg_1
```

Ilustración 10-26. Salida de sp_helpdb que muestra los segmentos por dispositivo de una base de datos en SQL Server.

Desplegando información acerca de objetos en segmentos.

Ejecutando **sp_helpsegment [segname]** lista los objetos en el segmento y muestra el o los dispositivos relacionados.

```
1> sp_helpsegment seg1
2> go
segment name         status
-------------------- ----------
seg1                          0
device                          size           free-pages
------------------------------- -------------- --------------------
data_dev3                       1.0MB                       430
table_name                      index name     indid
------------------------------- -------------- ----------
tableA                          x_tableA                1
```

Ilustración 10-27. Uso de sp_helpsegment para desplegar los objetos en un segmento en SQL Server.

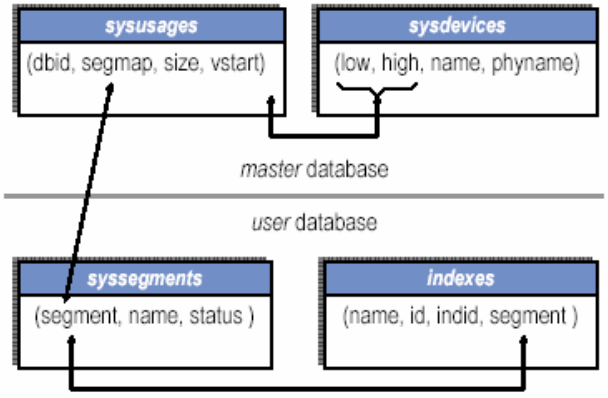


Ilustración 10-28. Tablas del sistema relacionadas a segmentos

`Sysusages.segmap` es una representación decimal de un mapa de bits.

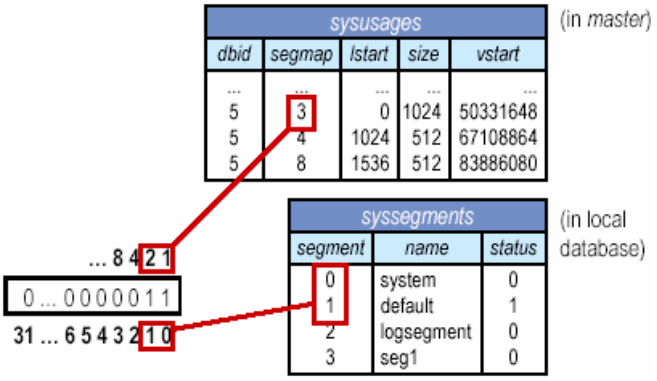


Ilustración 10-29. Relación de sysusages y sysegments en la representación decimal del campo segmap de un mapa de bits.

Tres dispositivos para la base de datos **`salesdb`** tienen segmentos definidos como sigue:

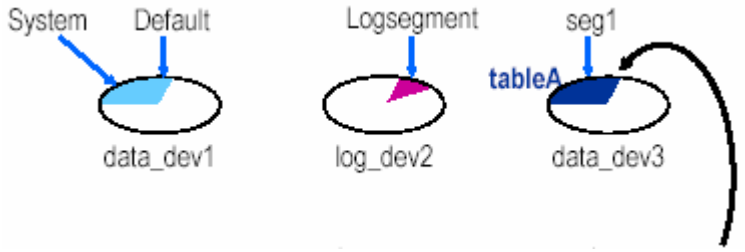


Ilustración 10-30. Distribución de dispositivos y segmentos en una base de datos en SQL Server.

Grande y de uso pesado la tabla A ha sido puesta en seg1. ¿Qué se puede hacer si ya hay otra tabla grande y de uso pesado? ¿Qué puede hacer si la tabla A crece demasiado en seg1? La solución es crear un nuevo segmento en un dispositivo nuevo como se ilustra en el siguiente esquema.

Ilustración 10-31. Diagrama resultante de agregar un nuevo segmento en un nuevo dispositivo en SQL Server.

```
disk init; alter database; sp_addsegment;
sp_dropsegment (system and default);
create table...on seg2
```

Código 10-20. Comandos para agregar un nuevo segmento en SQL Server.

¿Qué se hace si no hay suficiente espacio en disco, si la tabla A crece demasiado que espacio de seg1 se usa? Solución, extender seg1 a otro dispositivo.

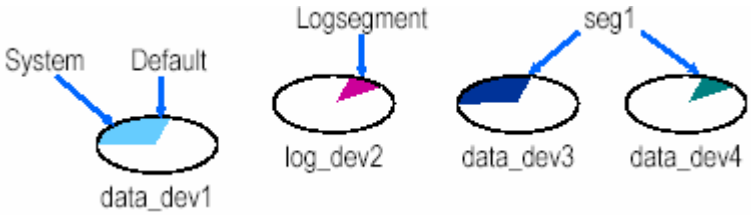


Ilustración 10-32. Extender un segmento a otro dispositivo en SQL Server.

```
disk init;
alter database;
sp_extendsegment (onto dev4);
sp_dropsegment (system and default)
```

Código 10-21. Comandos para extender un segmento a otro dispositivo en SQL Server.

Cambiando segmentos para objetos existentes. Hay que dirigirse a las tablas o índices que crecen para ejecutar: **`sp_placeobject segname, object`**

```
sp_placeobject seg2, tableA
```

Código 10-22. Uso de sp_placeobject en SQL Server.

Ilustración 10-33. Dirigir objetos a otro segmento en SQL Server.

Cuando la **`tablaA`** necesita más espacio, SQL Server utiliza **`seg2`**.

Divida una tabla grande hacia dos segmentos o dispositivos.

Usando Particiones

Al particionar una tabla se crean cadenas multipágina para una tabla. Al particionar se agrega un grado de paralelismo a los sistemas configurados para realizar procesamientos de consulta paralelo. Cada proceso lee una partición por separado. El particionar hace posible llenar una tabla en paralelo con **`bulk copy`**. El particionar hace posible distribuir el I/O de una tabla a través de múltiples dispositivos de base de datos. El particionar proporciona muchos puntos de inserción para varias tablas.

Claves para el uso de particiones

Un diseñador de base de datos escoge una o más tablas que pueden tomar ventaja de la partición de datos en una base de datos.

Un candidato es una tabla "**`append only`**" en la que en cada transacción debe escribir.

Las tablas que proporcionan una historia o lista de auditoria de actividades son también candidatas para la partición de datos. Por ejemplo, una compañía de teléfonos debe registrar cada llamada que llega durante las horas laborales para procesos históricos. Esto causa una alta disputa en la tabla.

Inserciones hacia una single page chain

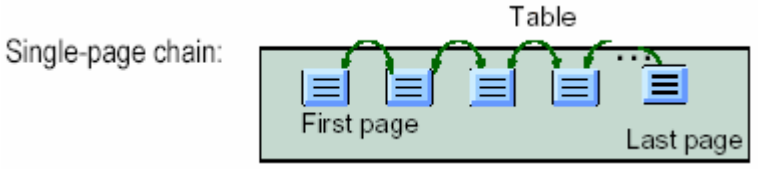


Ilustración 10-34. Esquema de inserciones hacia una Single – page chain.

Los objetos no particionados son implementados como una partición sencilla. Todos los nuevos registros van a la última página. Si está llena, una nueva es asignada y los nuevos registros van ahí.

La última página llega a tener un hot spot. Una transacción de inserción mantiene un bloqueo exclusivo hasta el fin de la transacción.

Inserts con datos particionados

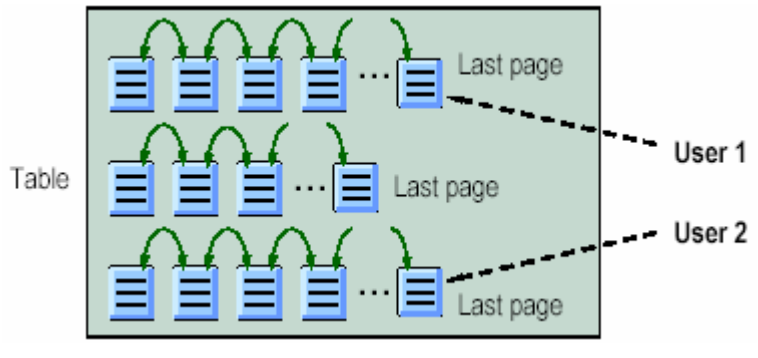


Ilustración 10-35. Manejo de inserts con datos particionados.

Dividir la tabla en múltiples particiones. Cada inserción es realizada en una de las cadenas de página. Se puede asignar las particiones en las que se hacen inserciones, con **bcp**, o el sistema los selecciona al azar. Todas las inserciones de una transacción dada van hacia la misma partición.

Inserciones usando bulk copy paralelo

Las tablas que son particionadas tienen mejor rendimiento cuando las inserciones son hechas mediante un **bulk copy** paralelo (**bcp**). Las inserciones son asignadas a particiones específicas usando **bcp** y ejecutadas concurrentemente.

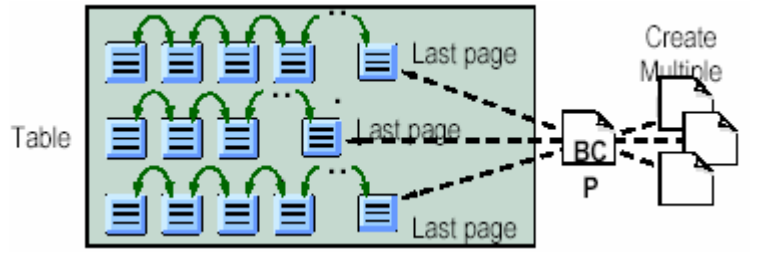


Ilustración 10-36. Esquema del tratamiento que hace SQL Server al usar bcp.

Tablas particionadas y procesamiento de consulta paralelo

Si las tablas son particionadas, el procesamiento de consulta paralelo puede potencialmente producir mejoras en la realización de consultas. Las particiones incrementan el número de cadenas de páginas que son accesadas simultáneamente por procesos. Cuando las particiones son distribuidas a través de discos físicos, la disputa reducida de I/O acelera la velocidad del procesamiento de consulta paralelo y logra un alto nivel de paralelismo. El optimizador puede escoger usar el procesamiento de consulta paralelo para una consulta contra una tabla particionada cuando el procesamiento de consulta paralelo es habilitado.

Equilibrio de partición / partición inclinada

Una distribución equitativa de páginas a través de particiones (llamada equilibrio de partición) incrementa la efectividad en búsquedas paralelas. A veces los datos en una tabla particionada pueden llegar a ser distribuidas en forma irregular, o inclinados dados los siguientes aspectos:

- Un gran número de registros son insertados en un subconjunto de una tabla particionada.
- Un gran número de registros son borrados de un subconjunto de una tabla particionada.
- Las actualizaciones ocurren sobre un subconjunto de una tabla particionada usando algún otro modo de actualización que in-place.
- **`bcp`** copia un gran número de registros hacia un subconjunto de una tabla particionada o inserta múltiples archivos aleatoriamente.

Distribuyendo datos a través de particiones con índices clustered

Creando un índice clustered sobre una tabla particionada se distribuyen los datos de la tabla sobre las particiones. SQL Server determina los rangos de la llave del índice para cada partición así que puede distribuir los registros equitativamente en la partición. Cada partición es asignada a por lo menos un dispositivo exclusivo si el número de dispositivos en el segmento es igual o mayor que el número de particiones.

Particionando una tabla sobre un dispositivo sencillo

Si no se puede crear una tabla como particionada. Primero se debe crear la tabla, entonces usar **`alter table`** para particionarla.

```
#Crear la tabla
create table salesdetail
(ord_no char(10),
prod_no char(25)....
)
#Especificar el número de particiones para la tabla
alter table salesdetail partition 4
```

Código 10-23. Proceso para particionar una tabla en SQL Server.

Particionando una tabla que contiene datos

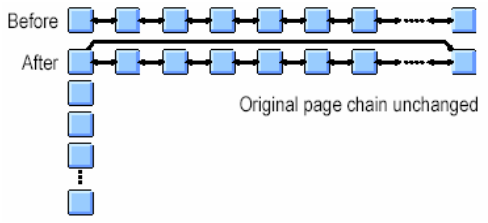


Ilustración 10-37. Esquema del proceso de partición de una tabla que contiene datos.

La primera partición es muy grande. Da beneficios de tener múltiples últimas páginas. El comando **`alter table .. partition`** se ejecuta rápidamente

Proceso para el particionado a través de múltiples dispositivos de Bases de Datos:

- Establecer un segmento que tenga múltiples fragmentos de datos:

```
sp_addsegment seg1, salesdb, data_dev1
sp_extendsegment seg1, salesdb, data_dev2
```

Código 10-24. Establecer un segmento que con múltiples fragmentos de datos en SQL Server.

- Poner la tabla en el segmento:

```
create table salesdetail (column1 numeric(3,0), column2 char(25)) on seg1
```

Código 10-25. Crear una tabla en un segmento específico en SQL Server.

O usar **`sp_placeobject`** después de crear un objeto:

```
sp_placeobject seg1, salesdetail
```

Código 10-26. Colocar una tabla en un segmento en SQL Server.

- Especifique el número de particiones para la tabla:

```
alter table salesdetail partition 4
```

Código 10-27. Uso de alter table para colocar una tabla en una partición específica en SQL Server.

```
disk init NAME='datadev1',PHYSNAME='/curdev1/server11/datadev1.dat', VDEVNO=5, SIZE=1024
disk init NAME='logdev1', PHYSNAME='/curdev1/server11/logdev1.dat', VDEVNO=6, SIZE=512
disk init NAME='datadev2', PHYSNAME='/curdev1/server11/datadev2.dat',VDEVNO=7, SIZE=512
disk init NAME='datadev3', PHYSNAME='/curdev1/server11/datadev3.dat',VDEVNO=8, SIZE=512
disk init NAME='datadev4', PHYSNAME='/curdev1/server11/datadev4.dat',VDEVNO=9, SIZE=512
create database testdb on datadev1=2, datadev2=1, datadev3=1, datadev4=1 log on logdev1=1
use testdb
exec sp_addsegment 'tableseg' , 'testdb' , 'datadev2'
exec sp_extendsegment 'tableseg','testdb','datadev3'
exec sp_extendsegment 'tableseg','testdb','datadev4'
exec sp_dropsegment 'system','testdb','datadev2'
exec sp_dropsegment 'system','testdb','datadev3'
exec sp_dropsegment 'system','testdb','datadev4'
exec sp_dropsegment 'default','testdb','datadev2'
exec sp_dropsegment 'default','testdb','datadev3'
exec sp_dropsegment 'default','testdb','datadev4'
create table a (column1 int, column2 char(30)) on tableseg
alter table a partition 3
```

Código 10-28. Proceso para particionar a través de múltiples dispositivos en SQL Server.

Mapeo de particiones a partes de un segmento

Si cuatro particiones han sido requeridas, y el segmento toma cuatro fragmentos de dispositivo, cada partición es mapeada a uno de los fragmentos de dispositivo del segmento.

Partition	4 Fragments	10 Fragments	3 Fragments
1	F1	F1, F5, F9	F1
2	F2	F2 , F6, F10	F2
3	F3	F3 , F7, F1	F3
4	F4	F4 , F8, F2	F1

Ilustración 10-38. Mapeo de particiones en SQL Server.

Restricciones en particiones

Las tablas que no pueden ser particionadas son:

- Todas las tablas del sistema en la base de datos **`master`** y base de datos de usuario.
- Tablas de trabajo.
- Tablas temporales.
- Tablas que ya están particionadas.

Operaciones que *no* están permitidas sobre tablas particionadas:

- **`truncate table`**
- **`sp_placeobject`**
- **`alter table .. partition`**

Otros comandos sobre objetos particionados

Para desparticionar una tabla particionada:

```
alter table table_name unpartition
```

Código 10-29. Uso de alter table para desparticionar en SQL Server.

Borrar una tabla particionada:

```
drop table table_name
```

Código 10-30. Uso de drop table para borrar una tabla particionada en SQL Server.

Para usar particiones, se usa la sintaxis normal de **`insert`**, **`update`**, **`delete`**, y **`select`**

- La disputa entre un **`insert`** y un **`update`** es reducido.
- Use **`sp_helpartition`** para obtener información acerca de las tablas con múltiples particiones.

Noticias para el uso de particiones

Usualmente sólo un pequeño número de tablas en una aplicación se benefician de la partición. El mejoramiento del rendimiento puede ser significativo. Se usa particiones de datos cuando se tenga una tabla con muchas inserciones. Si hay una pequeña disputa sobre una tabla, el particionarla afectará mínimamente el rendimiento.

Colocación de Datos

En un pequeño sistema o en un entorno de desarrollo, se pone toda una base de datos en el mismo disco físico, especialmente si es fácil de volver a crear.

Ventajas:

- Simple.
- Usa recursos mínimos.

Desventajas:

- No hay una recuperación completa en caso de fallo en el medio (Solo el último respaldo).
- El rendimiento se afecta si las tablas son grandes y la actividad es pesada.

Con tablas grandes, por las cuales la recuperación y el rendimiento son críticos. En base de datos extendidos uniformes a través del disco, minimiza la contención:

- Poner base de datos en discos separados.
- Poner tablas grandes en discos separados usando segmentos.

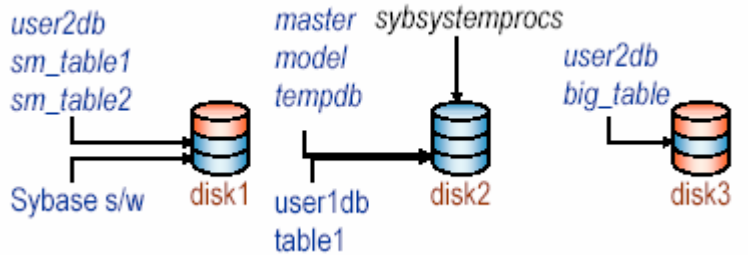


Ilustración 10-39. Esquema de bases de datos en discos separados en SQL Server.

Colocando logs

En sistemas pequeños o en un entorno de desarrollo, almacenar el log y los datos en un mismo disco físico.

Ventajas:

- Simple.
- Usa recursos mínimos.

Desventajas:

- No se pueden realizar respaldos de log a menos que la base sea creada con la opción **`with override`**.
- En caso de un fallo en el medio, no se puede recuperar hasta el último minuto; solamente es bueno en el último respaldo completo.
- Con gran actividad, el rendimiento baja, porque las escrituras de el log compite con las escrituras de datos

Con grandes tablas cuando la recuperación y rendimiento son críticos, las opciones siguientes son posibles:

- Poner el log en un disco físico separado de los datos.
- Poner el log en un disco que no este ocupado.
- Poner el log en su propio segmento en el que se pueda usar el threshold manager para monitorear su uso.
- Duplicar del dispositivo del log para una completa recuperación.
- Log en separado, disco relativamente inactivo, duplicado.

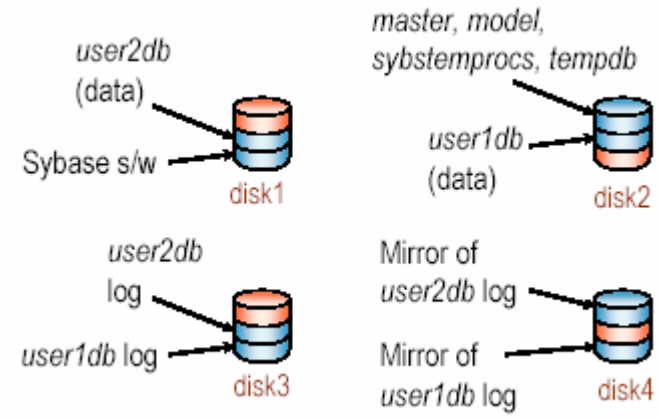


Ilustración 10-40. Esquema optimizando el rendimiento y recuperación críticas en SQL Server.

Colocando tempdb

La base de datos **`tempdb`** es instalada automáticamente en el dispositivo **`master`**. Si se necesita más espacio, se puede extender **`tempdb`** a otros dispositivos.

Ilustración 10-41. Colocación de tempdb en un dispositivo diferente al master en SQL Server.

Si se espera que **`tempdb`** tenga mucha actividad, se debiera de colocar en un disco que no este ocupado.

Colocando Respaldos, `sybsecurity`

Los respaldos son generalmente puestos en archivos de disco o cintas. La base de datos **`sybsecurity`** contiene la tabla **`sysaudits`**.

Si se espera una gran actividad auditable, hay que poner **`sybsecurity`** en:

- En un disco que no este ocupado.

- En su propio segmento.

Reuniéndolo todo

En producción, se equilibra la carga de acceso al disco con los discos disponibles. En el desarrollo, solo hay que poner todo en un disco como se muestra a continuación:

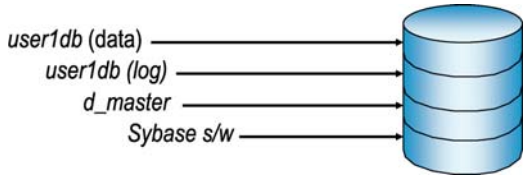


Ilustración 10-42. Distribución recomendada para el entorno de desarrollo en SQL Server.

Para el sistema en producción se deben de usar al menos 2 discos:

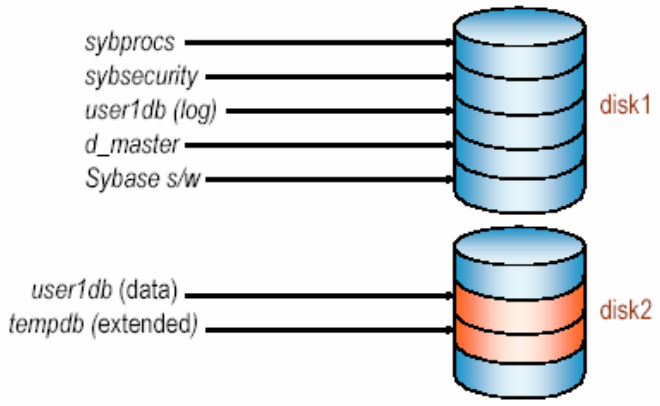


Ilustración 10-43. Distribución recomendada para el entorno en producción en 2 discos SQL Server.

Para un sistema grande de cuatro discos en el sistema de producción se recomienda tener distribuido de la siguiente forma:

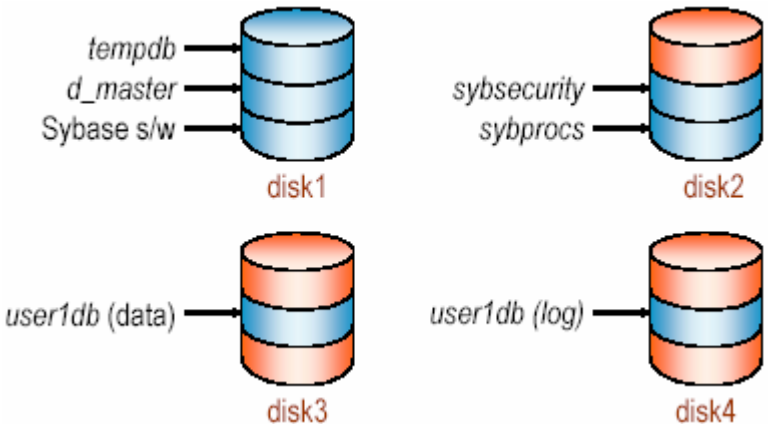


Ilustración 10-44. Distribución recomendada para el entorno en producción de 4 discos SQL Server.

Para sistemas muy grandes de 8 discos el sistema de producción multidisco recomendado se distribuye como sigue, donde user2db tiene baja actividad:

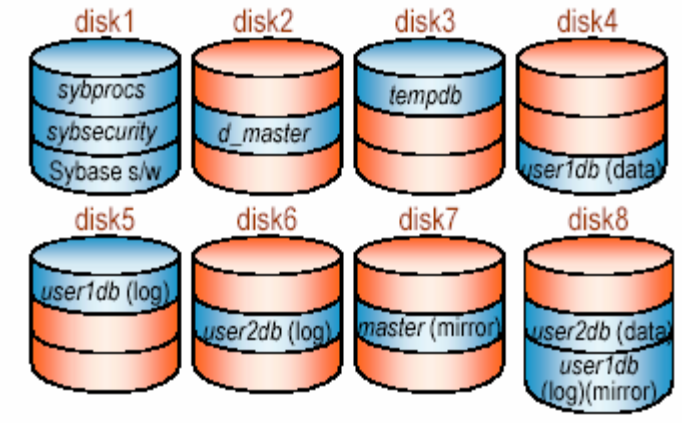


Ilustración 10-45. Distribución recomendada para el entorno en producción multidisco de 8 discos SQL Server.

Resumen

Cuando se da el tamaño de una base de datos, se debe considerar tablas, índices y log de transacciones. Las bases de datos son fáciles de extender (usando **alter database**), pero imposibles de encoger. Para máximo uptime y completa recuperación, hay que colocar logs en discos separados y duplicar dispositivos de log. Para mejor rendimiento, hay que repartir el trabajo uniformemente por los discos que tiene y minimizar la contención. Para bases de datos que tengan tres discos, usar segmentos para controlar la colocación de objetos grandes y de uso pesado. Para reducir contención de last-page en tablas estructuradas, usar tablas particionadas.

10.3 CONFIGURANDO CACHES Y E/S GRANDES

Introducir a las características de activación del manejador del SQL Server - caches y grandes E/S's; crear, borrar, y modificar caches; vincular objetos a caches; configurar caches de metadatos; crear bancos de buffer y activar grandes E/S's; verificar qué bancos del buffer son usados por las consultas, son características de configuración importantes en el tratamiento de caches y E/S grandes.

Apreciación: caches y grandes E/S's

Cuando se reinicia un SQL Server por primera vez, sólo existe un cache, el default data cache.

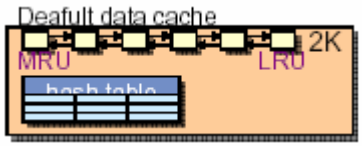


Ilustración 10-46. Configuración inicial del default data cache en SQL Server.

Si se quiere usar caches hay que crearlos, especificando:

- La cantidad de memoria.
- Un nombre.
- Un estado (mixed, log only).

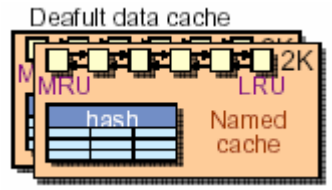


Ilustración 10-47. Esquema de un default data cache creado en SQL Server.

Se pueden vincular objetos a los caches o al default data cache.

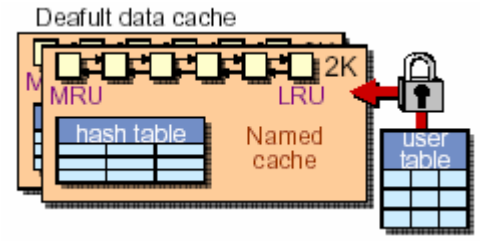


Ilustración 10-48. Esquema que ejemplifica la vinculación de un objeto a los caches en SQL Server.

También se pueden crear, modificar y eliminar bancos de buffer para los caches creados. Los bancos de Buffer son listas ligadas de páginas de 2K que pueden tener I/O de 2K, 4K, 8K, o 16K.Proporcionando un mecanismo para realizar grandes I/O's.

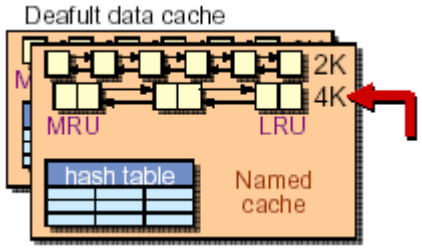


Ilustración 10-49. Esquema de bancos de buffer creados en un cache creado en SQL Server.

Caches

- Reducen la contención de los recursos del manejador de buffer en un entorno SMP[1].
- Permite un mejor uso de memoria al particionar la memoria a través de líneas.

---

[1] SMP - Symmetric Multi-Processing, Multi-Procesamiento Simétrico para multi-procesadores o de varios núcleos.

- Minimiza las I/O en el disco a través de un mejor manejo de memoria.

Tamaño del Banco del Buffer (grandes I/Os)

- Reduce el costo de I/O leyendo y escribiendo múltiples páginas de una base de datos en un sencillo I/O
- Hay que usar I/Os de 2K, 4K, 8K, y 16K.

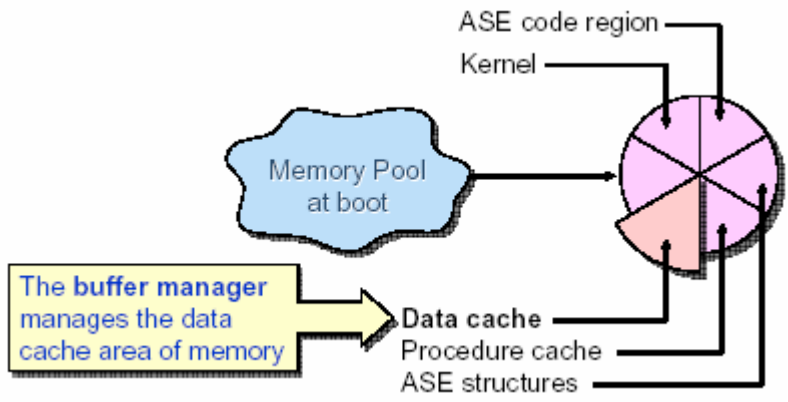


Ilustración 10-50. Asignación de memoria para caches y grandes I/O's.

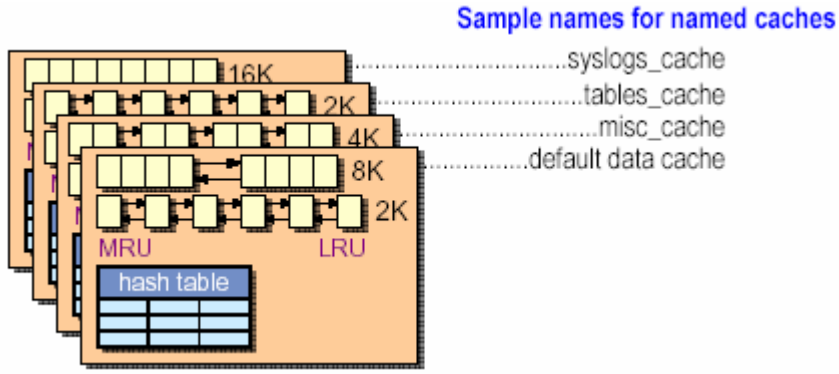


Ilustración 10-51. Manejador del Buffer del SQL Server.

Default data cache

Siempre hay un default data cache. No se necesita agregar caches.

El default data cache es usado:

- Cuando no se han especificado caches para objetos.
- Para la recuperación, **`load database`** y **`load transaction`**.

El **`default data cache`** es creado cuando SQL Server reinicia y es un cache mixto.

No se puede eliminar el **`default data cache`**, pero si se permite cambiar su tamaño.

El **`default data cache`** siempre tiene un banco del buffer donde se realizan I/O's de 2K; el tamaño mínimo de este banco es de 512K.

```
1> sp_helpcache
2> go
Cache Name          Config Size     Run Size    Overhead
------------------- --------------- ----------- -----------
default data cache   0.00 Mb         3.99 Mb     0.25 Mb

Memory Available For        Memory Configured
Named Caches                To Named Caches
--------------------------- --------------------------
        4.00 Mb             0.00 Mb

There is 4.00 Mb of memory left over that will be allocated to the default cache
---------------------------- Cache Binding Information: ----------------------------
Cache Name          Entity Name     Type        Index Name
------------------- --------------- ----------- ----------------
(return status = 0)
1>
```

Ilustración 10-52. Verificando caches usando sp_helpcache en SQL Server.

Creando un cache

Para crear un cache se puede usar **`isql`**:

```
1> sp_cacheconfig pub_cache, "10M"
2> go
```

Código 10-31. Uso de sp_cacheconfig para crear un cache de 10 Mb en SQL Server.

Usando el archivo de configuración:

```
[Named Cache: pub_cache]
cache size = 10M
cache status = mixed cache
```

Código 10-32. Uso del archivo de configuración para crear un cache de 10Mb en SQL Server.

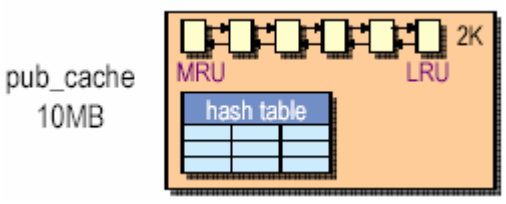


Ilustración 10-53. Esquema de un cache de 10Mb en SQL Server.

Modificando un cache

Usando **`isql`**:

```
1> sp_cacheconfig pub_cache, logonly
2> go
```

Código 10-33. Uso de sp_cacheconfig para modificar un cache en SQL Server.

Usando el archivo de configuración:

```
[Named Cache: pub_cache]
cache status = log only cache
```

Código 10-34. Uso del archivo de configuración para modificar un cache en SQL Server.

Para cambiar el tamaño del cache porque es muy pequeño.

Para alterar el tipo del cache de un cache mixto a un **`log only cache`**.

Vinculando y desvinculando objetos a caches

Datos de la vinculación:

- Puede vincular bases de datos, tablas, índices, y logs a caches.
- Un índice (**`clustered`** y **`nonclustered`**) puede vincularse a un cache diferente de los datos de ese índice.
- Una entidad puede vincularse sólo a un cache.
- Un cache puede tener varios objetos vinculados a él.
- Una columna de texto puede vincularse a su propio cache.

Debe estar en **`master`** para vincular y desvincular bases de datos.

Usando **`isql`**:

```
1> sp_bindcache "pub_cache","pubs2","authors"
2> go

1> sp_bindcache "pub_cache","pubs2",
2> "titles","titleind"
3> go

1> sp_unbindcache "pubs2","authors"
2> go

1> sp_unbindcache_all "pub_cache"
2> go
```

Código 10-35. Vinculando objetos a caches en SQL Server.

Las I/O's posteriores serán realizadas en el cache predeterminado.

La acción toma efecto inmediatamente sin requerir la reinicialización.

Se usa **`sp_helpcache`** para determinar objetos que están vinculados.

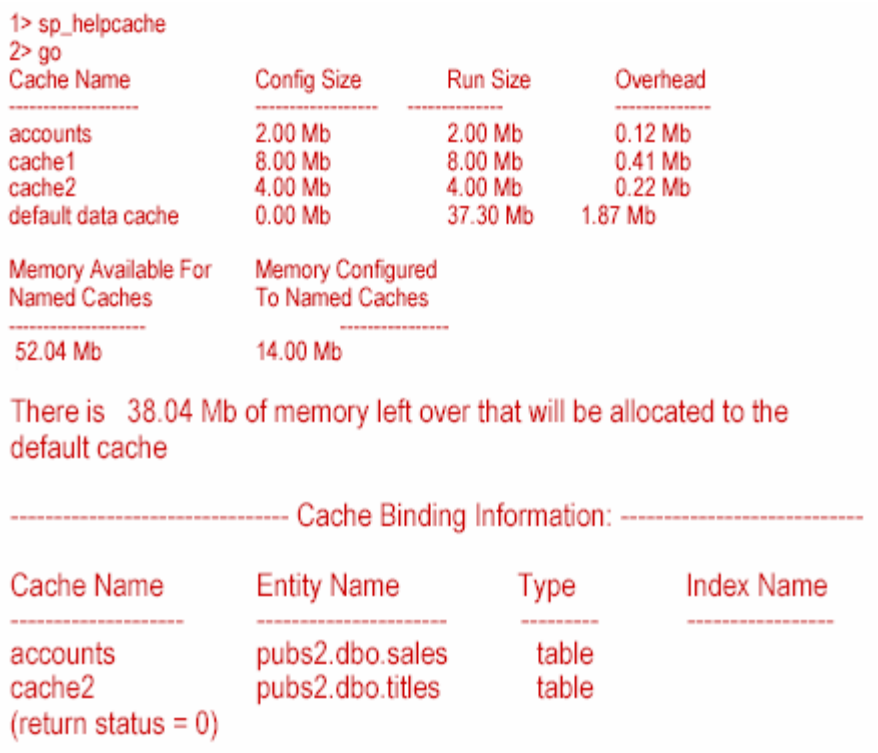


Ilustración 10-54. Uso de sp_helpcache para determinar objeto vinculados en SQL Server.

sp_helpcache

```
1> sp_helpcache "20M"
2> go
1.00Mb of overhead memory will be needed to manage a cache of size 20M
(return status = 0)
```

Código 10-36. Uso de sp_helpcache en SQL Server.

sp_help

```
1> sp_help pubs2..authors
2> go
...
buffer manager cache binding 1 cache_test1 NULL
...
```

Código 10-37. Uso de sp_help para vinculara caches a objetos en SQL Server.

Eliminando un cache

Al eliminar un cache se tiene un cache de memoria disponible primero hay que desvincular objetos antes de eliminar un cache.

Usando **isql**:

```
1> sp_cacheconfig "pub_cache", "0"
2> go
```

Código 10-38. Uso de sp_cacheconfig para eliminar un cache en SQL Server.

Usando el archivo de configuración:

Borrar el registro del banco del buffer en el archivo de configuración.

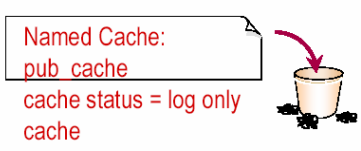


Ilustración 10-55. Uso del archivo de configuración para eliminar un cache en SQL Server.

Implicaciones de la configuración del cache para la recuperación

Sólo el **`default data cache`** esta disponible para la recuperación. El más pequeño **`default data cache`**, tomará más tiempo en recuperar.

Consideraciones en el tamaño del **`default data cache`**:

- Mezcla de transacciones.
- Intervalo de recuperación.
- Tamaño total de todos los caches configurados.

Para aplicaciones con un alto grado de transacciones, hay que asegurarse que el tamaño de el **`default data cache`** sea lo suficientemente grande para que la recuperación se haga dentro de un tiempo razonable.

Caches Metadatos

SQL Server 11.5 elimina la necesidad de caches para:

- **`sysindexes`**
- **`sysobjects`**
- **`sysdatabases`**

SQL Server 11.5 introduce tres nuevas estructuras de configuración de memoria en el servidor y el kernel que son usados para el manejo de los tipos de metadatos necesarios.

Configurando caches metadatos

Se puede configurar el tamaño de una nueva vía de metadatos con los siguientes parámetros en el **`sp_configure:`**

- **`number of open indexes`** fija el número máximo de índices abiertos que pueden estar activos en un momento.
- **`number of open objects`** fija el número máximo de objetos abiertos que pueden estar activos en un momento.
- **`number of open databases`** fija el número máximo de bases de datos abiertos que pueden estar activos en un momento.

Hay que asegurarse de dejar suficiente espacio para el crecimiento y una alta actividad no anticipada. Estos parámetros de configuración no son estáticos; por consiguiente, se deberá restaurar el servidor antes de que los cambios tomen efecto.

`sp_countmetadata`

- Se usa para buscar el número total de objetos que corresponden a cada tipo de metadatos.
- Despliega el número de objetos del tipo especificado así como la cantidad de memoria que requiere el cache metadato.

`sp_monitorconfig`

- Usado para reportar el número de descriptores del objeto actualmente en uso y libre, así como el número máximo usado desde la última reinicialización del servidor.

`sp_helpconfig`

- Despliega cuánta memoria es requerida para un número de objetos dado.

Grandes I/Os

Grandes I/Os minimizan las I/O fisicas reduciendo el número de veces que el SQL Server obtenga del disco páginas de datos.

Grandes I/Os dan mejores resultados tomando ocho páginas de 2K en una sola operación, en lugar de tomar páginas de 2K, que hacen la operación ocho veces separadas

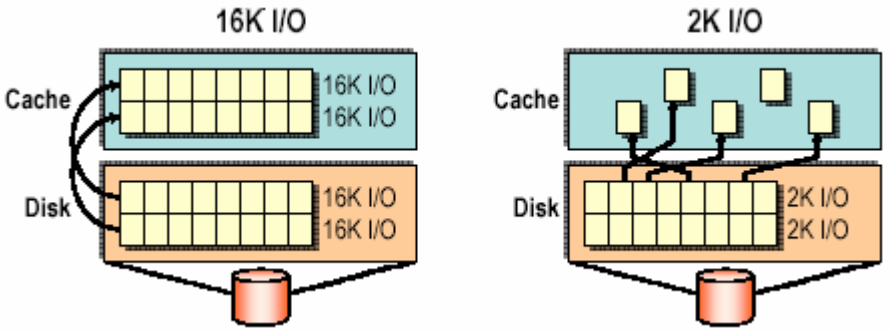


Ilustración 10-56. Comparación de acceso al cache tomando ocho páginas de 2K en una sola operación, en lugar de tomar páginas de 2K, que hacen la operación ocho veces separadas en SQL Server.

¿Porqué tener muchos tamaños de bancos de buffer?

Múltiples tamaños de bancos de buffer son usados para guardar grandes I/O's. Típicamente, grandes I/O's de datos son más útiles en aplicaciones DSS como:

- Realizar tareas de mantenimiento regulares como **`bcp`** y **`dbcc`**.
- Actualizar un millón de registros en un proceso.
- Unir tablas grandes.
- Cargar datos en una tabla usando sentencias **`insert`**.
- Buscando en una tabla de 100,000-registros.

- Realizar un rango de búsqueda en una tabla.

Grandes I/O's son útiles cuando el disco del log es un cuello de botella, entonces, se cambia el tamaño de las I/O's del log.

Bancos de buffer de 2K I/O obligatorios.

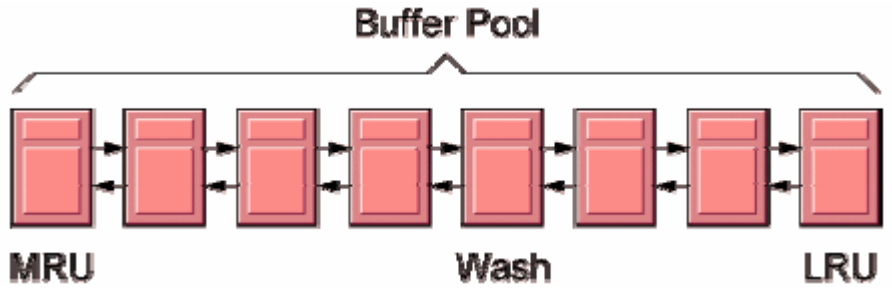


Ilustración 10-57. Esquema de un banco de buffer en SQL Server.

Cada cache contiene un banco del buffer de 2K I/O.

El banco de 2K I/O es usado para utilidades internas.

El tamaño mínimo para el banco de buffer obligatorio de 2K I/O es de 512 K.

Creando un banco del buffer

Crear el banco del buffer en un cache o en el **`default data cache`** de 4K I/O, con un total de 4 Mb dentro de un cache se hace como ilustra a continuación:

```
1> sp_poolconfig pub_cache, "4M", "4K"
2> go
```

Código 10-39. Uso de sp_poolconfig para crear un banco de buffer en SQL Server.

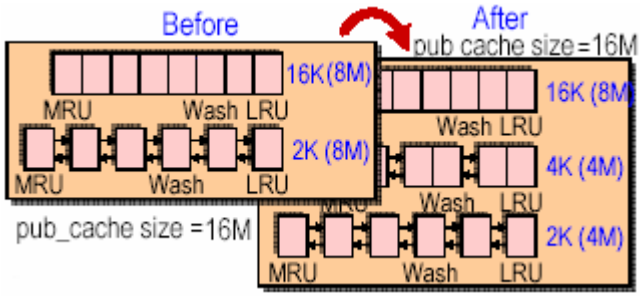


Ilustración 10-58. Esquema de un banco de buffer creado en SQL Server.

Modificando un banco de buffer

```
1> sp_poolconfig pub_cache, "5M", "4K", "16K"
2> go
```

Código 10-40. Uso de sp_poolconfig para modificar un banco del buffer.

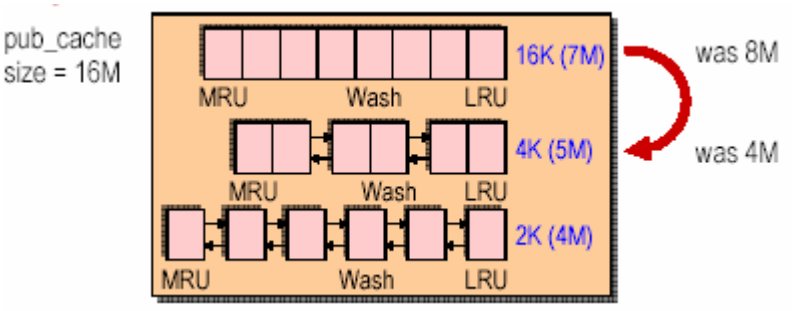


Ilustración 10-59. Resultado de modificar un banco del buffer.

¿Por qué modificar un banco del buffer?

- El banco del buffer es demasiado pequeño y las consultas deben esperar o usar diferentes tamaños de bancos.
- El banco del buffer es demasiado pequeño; y puede haber más I/Os físicas debido a que las páginas están siendo limpiadas del cache.
- El banco del buffer es demasiado grande, y la memoria puede ser utilizada mejor en otra parte.
- Se tiene una aplicación que requiere el proceso OLTP durante el día y proceso DSS durante la noche.
    - Esto requiere diferentes configuraciones de bancos de buffer— 16K I/O para DSS y 2K I/O para OLTP.
- Crear dos archivos de configuración separados con la correcta configuración del banco del buffer y cargarlo en el SQL Server como sea necesario.

Borrando un banco del buffer

Se debe Borrar un banco del buffer que fue usado sólo para dar una solución temporal en un proceso por lote de 16K I/O y que no sea necesario en un futuro.

Usando **`isql`**:

```
1> sp_poolconfig pub_cache, "0", "16K"
2> go
```

Código 10-41. Uso de sp_poolconfig para borrar un banco del buffer en SQL Server.

Usando el archivo de configuración:

- Eliminar el registro del banco del buffer en el archivo de configuración.

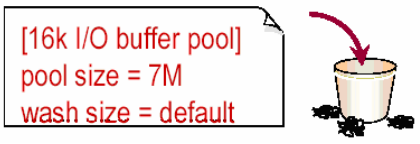


Ilustración 10-60. Fragmento del archivo de configuración a borrar para eliminar un banco del buffer en SQL Server.

Restricciones para crear, modificar y borrar bancos de buffer

- La suma de todos los buffers en el banco debe ser menor que el tamaño del cache en la memoria total.
- No se puede borrar el banco del buffer de 2K en ningún cache.
- La I/O para un banco del buffer puede ser especificado en incrementos de 2K, 4K, 8K, o 16K.
- Un nuevo banco es creado usando memoria del banco predeterminado de 2K I/O pool, a menos que se especifique otro.
- El tamaño mínimo de un banco es de 512K.
- El banco del buffer máximo para cualquier cache es de 16K.

Configurando caches desde el archivo de configuración

Los siguientes procedimientos del sistema configuran caches y grandes I/Os:

- **sp_bindcache**, **sp_unbindcache**, **sp_unbindcache_all**, **sp_poolconfig**, **sp_cacheconfig**, **sp_helpcache**.

Estos procedimientos almacenados escriben los cambios en el conjunto de parámetros en el archivo de configuración.

El archivo de configuración editable:

- Mantiene memoria de caches y bancos de buffer para el manejador del buffer. Reside en el directorio **$SYBASE**.

```
...
[Named Cache:accounts]
cache size = 2M
cache status = mixed cache

[Named Cache:cache1]
cache size = 8M
cache status = mixed cache

[2K I/O Buffer Pool]
pool size = 2048K
wash size = 1534 K

[4K I/O Buffer Pool]
pool size = 512K
wash size = 100 K
...
```

Código 10-42. Fragmento de un archivo de configuración donde muestra que hay dos caches: accounts y cache1 en SQL Server.

Verificando que un banco del buffer esta siendo usado

Se usa **set showplan y select ... prefetch** para verificar que un banco del buffer esta siendo usado.

```
1> set showplan on
2> go
1> select * from cachtb42
2> (index cacha_ind prefetch 8)
3> go
QUERY PLAN FOR STATEMENT 1 (at line 1).
STEP 1
        The type of query is SELECT.
        FROM TABLE cachtb42
```

```
Nested iteration.
Index : cacha_ind
Ascending scan.
Positioning by key.
Keys are: a
Using I/O Size 8 Kbytes.
With LRU Buffer Replacement Strategy
```

Código 10-43. Proceso para verificar que si un banco del buffer esta siendo ocupado en SQL Server.

También usando el comando **`set statistics io`**:

```
1> set statistics io on
2> go
1> select count(a) from cachtb42
2> (index cacha_ind prefetch 8)
3> go

--------------

                 100

Table: cachtb42 scan count 1, logical reads: 1, physical reads: 0
Total writes for this command: 0
(100 rows affected)
```

Código 10-44. Uso de set statistics io para verificar si un banco de buffer esta siendo utilizado en SQL Server.

Resumen

- Múltiples caches eliminan la contención spinlock en un entorno SMP.
- Múltiples bancos del buffer ayudan a controlar recursos de memoria en aplicaciones OLTP, DSS, y mixtas.
- Tomando las páginas frecuentemente más usadas en memoria incrementa el rendimiento, porque el acceso a memoria es más rápido que el acceso al disco.
- Los caches Metadato eliminan la necesidad de los caches en **`sysindexes`**, **`sysdatabases`**, y **`sysobjects`**.
- Grandes I/Os reducen costos de I/O leyendo y escribiendo muchas páginas de la base de datos en una sencilla I/O.
- Se usa **`set showplan`**, **`set statistics io`** para verificar que banco del buffer esta siendo utilizado.

10.4 INTRODUCCIÓN AL ACCESO PARALELO

Describir nuevos métodos de acceso paralelo, Usar herramientas del SQL Server que identifiquen y rastreen procesos paralelos, Usar herramientas del SQL Server que determinen que es más óptimo si el acceso serial o el acceso paralelo.

Métodos de acceso

Los métodos de Acceso son mecanismos de búsqueda de datos en tablas de base de datos relacionales para satisfacer consultas. Para muchas consultas, diferentes métodos de acceso son posibles. Por ejemplo, los datos que se necesitan por una consulta a una tabla pueden ser leídos:

- Buscando en toda la tabla.
- Buscando en índices **`clustered`**.
- Buscando en uno o más índices **`nonclustered`**

El optimizador de consulta es responsable de estimar los costos de cada método de acceso con relación a otros, y al final escoge el método más óptimo.

Accesando a datos sin un índice

SQL Server almacena tablas sin un índice **`clustered`** como cadenas de páginas directas.

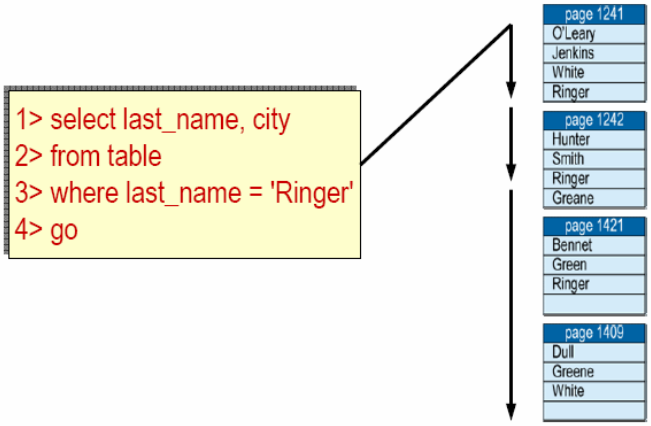


Ilustración 10-61. Esquema de acceso en cadenas de páginas directas en SQL Server.

La tabla, la cuál consiste en páginas ligadas, es llamada comúnmente una tabla **heap**.

Accesando a datos con un índice nonclustered

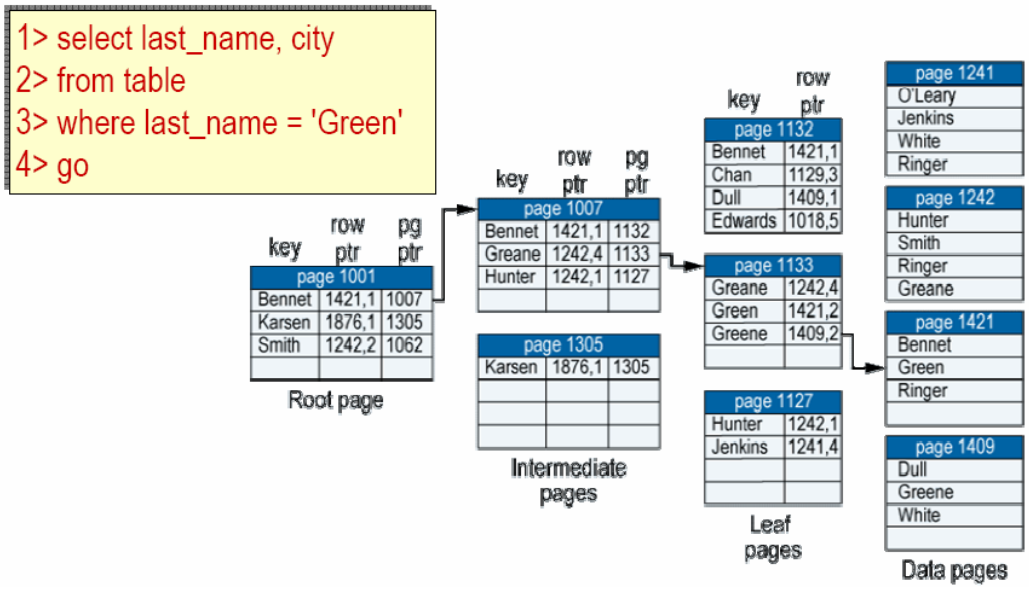


Ilustración 10-62. Accesando a datos con un índice nonclustered en SQL Server.

Accesando a datos con un índice clustered

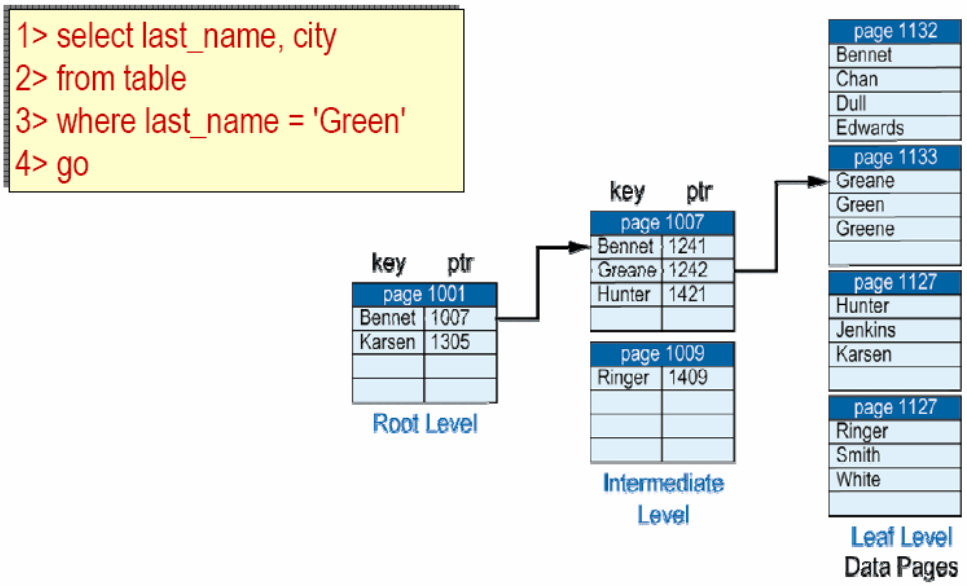


Ilustración 10-63. Accesando a datos con un índice clustered en SQL Server.

Acceso paralelo

SQL Server deja que cada usuario use el acceso paralelo, o más de un proceso, para una consulta sencilla:

- Búsqueda en particiones.
- Búsqueda en Hash.

Búsqueda en particiones

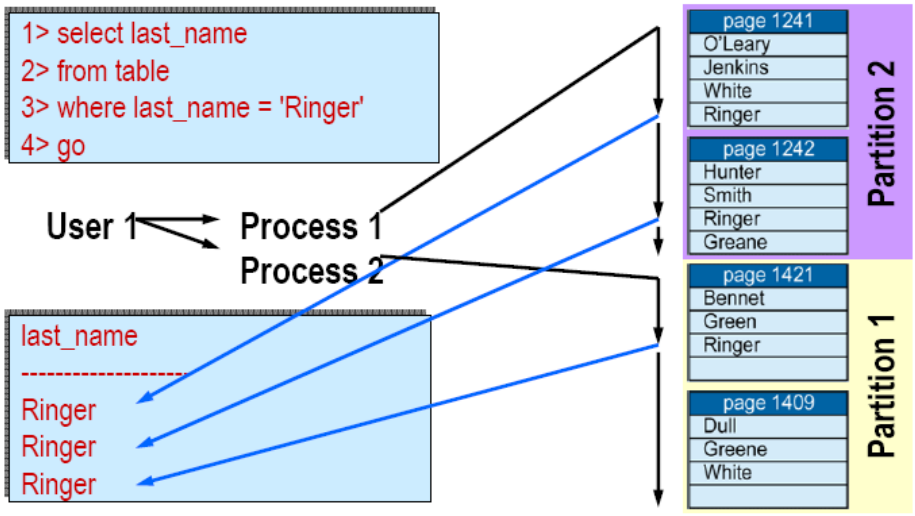


Ilustración 10-64. Método de acceso en la búsqueda en particiones en SQL Server.

Búsquedas basadas en hash

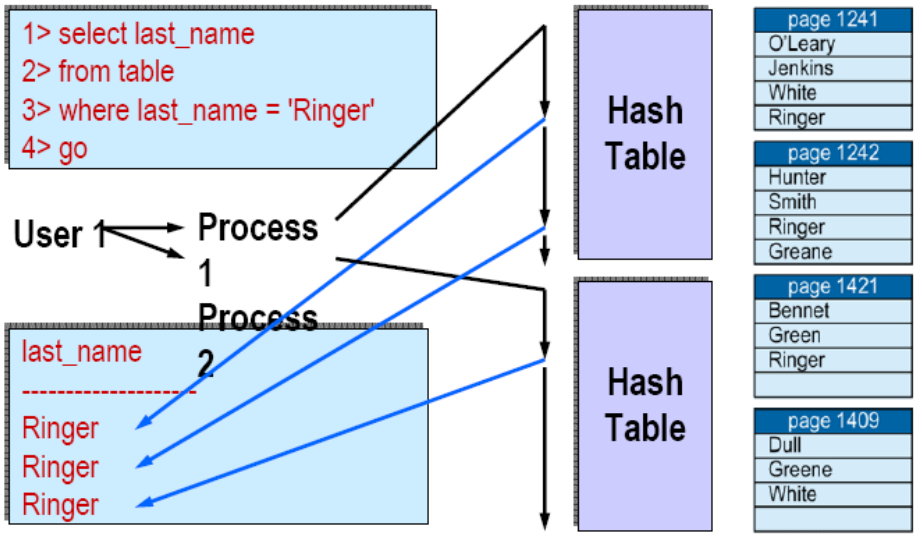


Ilustración 10-65. Método de acceso en búsquedas basadas en hash en SQL Server.

Búsqueda paralela

Un vistazo a la búsqueda en paralelo

Fase de acceso

1. Ejecuta procesos en paralelo.
2. Crea un conjunto de resultados, o **encapsulado**, para cada proceso.

Fase de unión

3. Une las cápsulas hacia un solo conjunto de resultados.
4. Envía el conjunto de resultados al cliente.

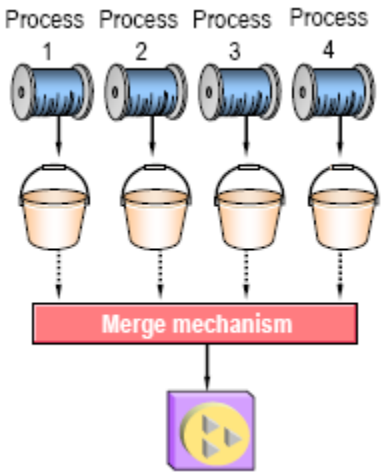


Ilustración 10-66. 4 etapas en el proceso de acceso paralelo en SQL Server.

Búsqueda paralela worker processes

En la versión 11.5 se introduce un nuevo tipo de tarea del SQL Server llamada **`worker process`**. Un **`worker process`** es un nuevo tipo de recurso disponible en el SQL Server.

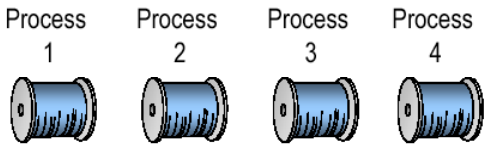


Ilustración 10-67. Procesos en el worker process en SQL Server.

Todos los procesos en el diagrama son **`worker process`**. Cada **`worker process`** tiene un ID de proceso único (**`spid`**).

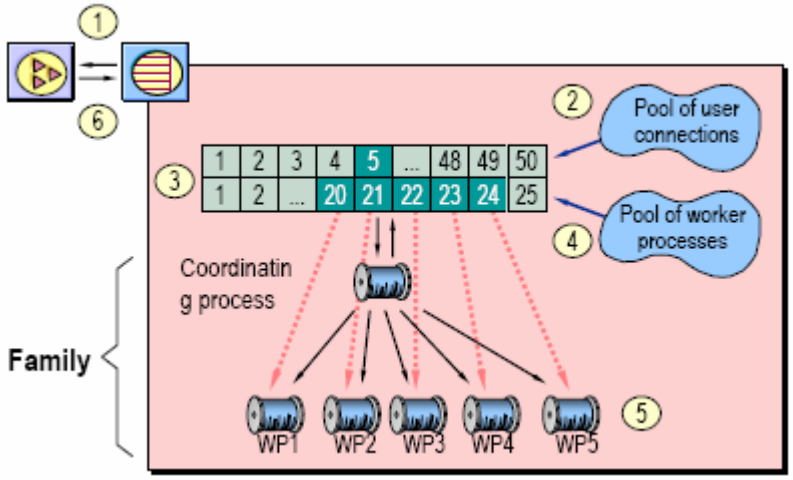


Ilustración 10-68. Familia de procesos en SQL Server.

Combinación de búsqueda paralela

Un **`merge`** es el proceso de combinar múltiples conjuntos de resultados de una consulta paralela hacia un solo conjunto de resultados. Hay dos tipos de combinación paralela:

- **`One-time-only merge`**.
- **`Continuos merge`**.

Algunas consultas paralelas pueden regresar registros en un orden diferente de una ejecución a otra.

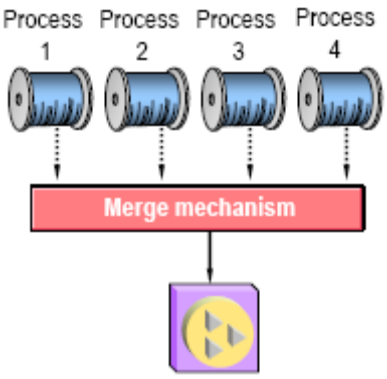


Ilustración 10-69. Esquema de un merge mechanism en la combinación de búsquedas paralelas en SQL Server.

Activando el acceso paralelo

Para que optimizador considere algún método de acceso paralelo. El procesamiento paralelo debe estar activado en el servidor vía **`sp_configure`** y los parámetros:

- **`number of worker processes`**.
- **`max scan parallel degree`**.
- **`max parallel degree`**.

Cada condición siguiente debe ser verdadera:

- El parámetro **`max parallel degree`** debe ser 2 o mayor.
- El parámetro **`parallel_degree`** debe ser 2 o mayor.
- El parámetro **`number of worker processes`** debe ser 2 o mayor.
- **`Select into/bulk copy/pllsort`** debe ser verdadera.

Identificando worker process

Usando **`sp_who`** se identifican los **`worker process`**.

```
1>sp_who
2> go
fid         spid status                                ...
            cmd
----        ------------------------------             ------------
NULL        1 runnable...                              SELECT
NULL        2 sleeping...                              NETWORK HANDLER
NULL        3 sleeping...                              DEADLOCK TUNE
NULL        4 sleeping...                              MIRROR HANDLER
NULL        5 runnable...                              HOUSEKEEPER
NULL        6 sleeping...                              CHECKPOINT SLEEP
NULL        7 running...                               SELECT
8               8 lock sleep...                        SELECT
8               9 lock sleep...                        WORKER PROCESS
8               10lock sleep...                        WORKER PROCESS
8               11send sleep...                        WORKER PROCESS
8               12lock sleep...                        WORKER PROCESS
8               13lock sleep...                        WORKER PROCESS

(13 rows affected, return status = 0)
```

Ilustración 10-70. Uso de sp_who para identificar los worker processes en SQL Server.

Búsqueda paralela y rendimiento

Las búsquedas paralelas permiten a varios **worker processes** ejecutar la consulta simultáneamente, lo cuál incrementa tanto la utilización del CPU como las I/Os.

Los factores que mejoran el tiempo de ejecución paralela incluyen:

- Múltiples CPU's.
- Múltiples dispositivos de I/O.

Múltiples CPU's, múltiples dispositivos de I/O

La siguiente tabla muestra la escala del CPU en la búsqueda de una tabla de 800 Mb con 30 particiones, usando 16 K I/O.

Eng	Elapsed Time (secs)	CPU Utilization	I/O Saturation	Throughput per Device
1	207	100%	Not saturated	.13 MB/sec
2	100	98.7%	Not saturated	.27 MB/sec
4	50	98.0%	Not saturated	.53 MB/sec
8	27	93.0%	100% saturated	.99 MB/sec

Tabla 10-1. Escala de uso de CPU en una consulta a una tabla de 800 Mb en SQL Server.

Acceso paralelo y sentencias SQL

El proceso de consulta paralelo mejora potencialmente el rendimiento de los siguientes tipos de consulta:

- Sentencias **select** que buscan muchas páginas pero regresan pocos renglones, como buscar en tablas o en índices **clustered**.
- Sentencias **select** que incluyan **group by**.
- Sentencias **select** que incluyan funciones agregadas.
- Sentencias **select** que incluyan **union**, **order by**, o **distinct**.
- Sentencias **select** las cuáles el optimizador escoja en la estrategia cambiada.
- Consultas select into.
- Sentencias **create index**.

Determinando el método de acceso usando showplan

```
1> select * from table1
2> go

QUERY PLAN FOR STATEMENT 1 (at line 1).
Executed in parallel by coordinating process and 4 worker processes.
        STEP 1
        The type of query is SELECT.
                Executed in parallel by coordinating process and 4 worker processes.
                FROM TABLE
                        table1
                Nested iteration.
```

```
        Table Scan.
        Ascending scan.
        Positioning at start of table.
        Executed in parallel with a 4-way partition scan.
        Using I/O Size 2 Kbytes.
                With LRU Buffer Replacement Strategy.

Parallel network buffer merge.
```

Código 10-45. Uso de showplan para determinar el método de acceso en SQL Server.

Resumiendo

SQL Server soporta dos métodos de búsqueda paralela: búsqueda basada en particiones y búsqueda basada en hash.

Como un resultado de una búsqueda paralela, SQL Server tiene cuatro métodos de acceso disponibles para optimizar consultas:

- Búsqueda en la tabla basada en particiones.
- Búsqueda en la tabla basada en Hash.
- Búsqueda en índices **`nonclustered`** basada en Hash.
- Búsqueda en índices basada en particiones.

Se usa **`sp_who`** para identificar y rastrear procesos paralelos.

10.5 PARALLEL DBCC

Uso del nuevo comando **`dbcc checkstorage`** eficazmente. Hay que preparar el Adaptive Server para usar **`dbcc checkstorage`**. Explicar el funcionamiento interno del comando **`checkstorage`** y cómo funciona y lo que verifica. Manejar corrupción en la base de datos con **`dbcc`** así como otro mantenimiento con respecto al **`dbcc checkstorage`**. Interpretar errores informadas por **`checkstorage,`** incluso una cartografía de nuevos errores a viejos.

Revisión de características

Verificando consistencia de datos

La integridad de los datos no sólo puede ser comprometida por el DBMS también por el sistema operativo y el hardware. El performance del **`checkdb`** y **`checkalloc`** varía grandemente y generalmente está alrededor de una a cinco horas por gigabyte de datos. Cuando esta proporción puede ser bastante para las bases de datos de 1 a 2 Gb, el costo para las bases de datos más grandes a menudo:

- Impide estos chequeos en todo el sistema.
- Fuerza a medidas extremas para trabajar alrededor de este problema.

Otro problema con estos comandos es la información falsa, o rara, errores debido a la actividad de actualización. Esto deja a menudo a los usuarios inseguros sobre el nivel de integridad de los datos después de ejecutar los **`dbcc checks`**.

dbcc checkstorage

Apreciación global

El comando **dbcc checkstorage** identifica los problemas de:

- Bajo performance en el chequeo de Consistencia.
- Informa Errores esporádicos.

El comando **checkstorage** se proporciona como una adición además de los **dbcc** existentes en la versión 11.0.

El comando **checkstorage** difiere de los chequeos anteriores:

- Requiere una base de datos especial.
- Ejecuta en paralelo usando **worker processes**.
- Minimiza bloqueos en tablas durante el proceso.
- La integridad se reporta en una base de datos en lugar de un archivo de salida.
- No usa números de error.

Antes de que usted pueda usar los comandos **checkstorage**, usted debe poner apropiadamente al Adaptive Server vía:

- el procedimiento **sp_plan_dbccdb** y **sp_addsegment**
- y los comandos **disk init**, **create database**
- los procedimientos **sp_configure**, **sp_cacheconfig**, **sp_poolconfig**
- la ejecución del script **installdbccdb**
- los procedimientos de **dbcc sp_dbcc_createws** y **sp_dbcc_alterws**, **sp_dbcc_updateconfig**

El resultado de esto es:

- Una base de datos **dbccdb** instalada en los dispositivos apropiados.
- El tamaño correcto **scans** y **text workspaces**.
- El tamaño correcto de cache

Uso de dbcc checkstorage

Ejecutando los chequeos

La utilidad de **checkstorage** puede invocarse en cualquiera de estas maneras:

- Vía el comando **dbcc checkstorage**.
- Vía el procedimiento **sp_dbcc_runcheck dbcc**.

Analizando resultados

Una vez que todo está completo y el **dbcc checkstorage** se corre, pueden analizarse los resultados del chequeo vía estos procedimientos:

- **sp_dbcc_faultreport**
- **sp_dbcc_fullreport**
- **sp_dbcc_statisticsreport**
- **sp_dbcc_summaryreport**
- **sp_dbcc_differentialreport**

Mantenimiento de la base de datos dbccdb

Usando estos procedimientos **dbcc** para el mantenimiento requerido:

- **sp_dbcc_deletedb**
- **sp_dbcc_deletehistory**
- **sp_dbcc_configreport**
- **sp_dbcc_updateconfig**

Se pueden usar otros comandos en el Adaptive Server para dar mantenimiento a **dbccdb**. Éstos incluyen:

- comando **alter database**
- **dump transaction** y **dump database**
- **load transaction** y **load database**

Preparando el Adaptive Server para usar checkstorage

Antes de que el dbcc checkstorage pueda ejecutarse

Al contrario del comando **dbcc** anterior, el comando de **dbcc checkstorage** requiere:

- Recursos especializados.
- Una serie de pasos completados.

Para preparar el Adaptive Server para usar el comando **dbcc checkstorage** involucra completar las dos fases siguientes:

- Fase 1: Planeación de requerimientos de recursos.
- Fase 2: Crear e inicializar la base de datos **dbccdb**.

Fase 1: Planeando requerimientos de recursos

El comando **checkstorage** requiere varios tipos de recursos que necesitan ser asignados correctamente en términos de tamaño y localización. La asignación inapropiada de recursos puede causar que el **checkstorage** falle con un error o llevar a un performance pobre.

Entre los recursos que necesitan ser configurados apropiadamente están:

- **Logical devices**.
- **Databases**.
- **Workspaces**.

- **Named caches**.
- **Worker processes**.

Usando el procedimiento sp_plan_dbccdb

El Adaptive Server provee el procedimiento **sp_plan_dbccdb** para ayudar al usuario recomendando los valores para:

- Qué dispositivos son convenientes para la base de datos **dbccdb**.
- El tamaño de base de datos **dbccdb**, incluyendo tamaño para datos y log.
- Tamaño para el **scan** y **text workspace**.
- Tamaño del cache.
- El número del máximo de **worker processes** por **checkstorage**.
- El procedimiento **sp_plan_dbccdb** basa sus recomendaciones en información de las tablas **sysdatabases**, **sysusages** y **sysdevices**.

Usando el procedimiento sp_plan_dbccdb

Este es un ejemplo de la ejecución de **sp_plan_dbccdb**:

```
1> sp_plan_dbccdb db1
2> go
Recommended size for dbccdb database is 9MB
(data = 7MB, log = 2MB)
Recommended devices for dbccdb are:
Logical Device Name                  Device Size (KB)
---------------------------------    -------------------------
dbdev1                               4000
dbdev2                               8000
Recommended values for workspace size, cache size and process count are:
dbname  scan ws        text ws        cache   process count
db1     96K            48K            1280K   2
```

Código 10-46. Uso de sp_plan_dbccdb para encontrar los valores recomendados para el workspace.

El **sp_plan_dbccdb** debe ejecutarse contra cada base de datos en la que se planee ejecutar **dbcc_checkstorage**.

Planeando recursos: Physical Devices

Los dispositivos físicos seleccionados para la base de datos **dbccdb** son críticos con respecto a la actuación del **checkstorage**. Idealmente, Los dispositivos (físicos) escogidos no deben compartirse con cualquier otra base de datos en el Adaptive Server. Si todos los dispositivos físicos existentes están en uso, el procedimiento **sp_plan_dbccdb** especifica que ningún dispositivo existente es conveniente.

Cualquier dispositivo puede usarse, pero la falla de no usar un dispositivo físico especializado puede producir una disminución del performance. Si el administrador de bases de datos no puede dedicar a un dispositivo físico al **dbccdb**, se sugiere crear la base de datos en dispositivo(s) con la menor actividad de I/O.

El dispositivo **master** ***no debe usarse*** para la base de datos del **dbccdb**.

Planeando recursos: Database Size

La base de datos **dbccdb** necesita ser bastante grande para guardar:

- **Scan** y **text workspaces** para ejecuciones concurrentes de **checkstorage**.
- Errores y datos de estadística informados por **checkstorage**.

El procedimiento **sp_plan_dbccdb** calcula el tamaño de la base de datos recomendado que usan las siguientes funciones:

- No más de dos **checkstorage** funcionando al mismo tiempo.
- Guarda los **workspaces** para las dos bases de datos más grandes.
- La base de datos **dbccdb** guarda datos de un promedio de cinco ejecuciones.
- Los datos **Fault** promedian alrededor de 50K por la ejecución.
- Los datos de estadística promedia alrededor de 500K por la ejecución.

El tamaño mínimo para la base de datos del **dbccdb** es 4MB.

Ajustando el tamaño de la base de datos

Los tamaños de base de datos de **dbccdb** reales pueden ser más altos si:

- Más de cinco ejecuciones se guardan en **dbccdb**.
- Más de dos concurrentes **checkstorage** se ejecutan.
- Una cantidad grande de datos **default** se encuentran.

Para ambientes que están cortos en espacio de disco, se puede usar un tamaño la base de datos **dbccdb** más pequeño que el tamaño recomendado por **sp_plan_dbccdb** si:

- Sólo una ejecución de **checkstorage** se ejecuta a un momento.
- Menos de cinco ejecuciones y valores de **Fault** y datos de estadísticas se salvan.

Planeando recursos: Transaction Log Size

El procedimiento **sp_plan_dbccdb** siempre recomienda el tamaño del log de transacciones a 2 MB. Mientras este valor es suficiente en la mayoría de las situaciones, como todos los logs de transacción, puede llenarse. Las únicas actividades que los logs del **checkstorage** son las inserciones de estadísticas y los datos **Fault** relacionados. Si un número grande de **Faults** se descubre encima de un corto intervalo de tiempo, el log de transacción puede llenarse.

Si el log se llena, el **checkstorage** se suspende hasta el espacio del log sea suficiente. Si el trabajo suspendido es terminado por el usuario, antes de que escribiera a la base de datos la **dbccdb** será accesible.

Planeando recursos: Workspace Sizes

Scan workspace

Los **scan workspace** requieren 18 bytes de almacenamiento para cada página en la base de datos designado (aproximadamente 1 % del tamaño de la base de datos).

El procedimiento **sp_plan_dbccdb** da una holgura de 10% para el crecimiento en el tamaño de la base de datos designado.

Por consiguiente, el tamaño del **scan workspace** se calcula como 1.2% del tamaño total de la base de datos.

Text workspace

El procedimiento **sp_plan_dbccdb** calcula el **text workspace** como el 25% del tamaño del **scan workspace**.

Ejemplo de asignación **workspace**

Dada la base de datos **db1** que tiene asignada **750MB** de disco y contiene **100MB** de datos

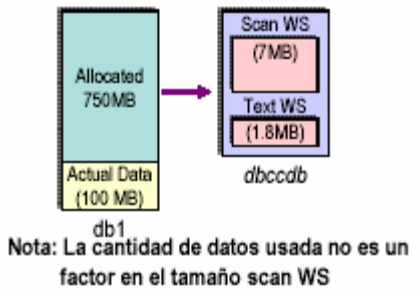


Ilustración 10-71. Esquema del dbccdb para el Workspace en SQL Server.

Ajustando el tamaño de text workspace

Para ambientes que usan alguno o ninguna columna del texto, se pueden hacer los **workspace** del texto más pequeño que lo recomendado por **sp_plan_dbccdb**. El tamaño mínimo, en páginas, para un **workspace** del texto es de acuerdo al siguiente algoritmo:

```
(number of worker processes used + 1) * 8
```

Código 10-47. Algoritmo para calcular el tamaño mínimo en páginas para un workspace de texto en SQL Server.

Para una base de datos de 1GB que no usa datos del tipo **text/image** hacer el tamaño de **workspace** del texto al mínimo y así ahorrar alrededor de 2.6MB.

Se puede necesitar aumentar el tamaño del **workspace** del texto más allá que el **sp_plan_dbccdb** recomendó. La primera vez que el **checkstorage** se usa, despliega un mensaje si el tamaño del **workspace** del texto necesita ser aumentado.

Planeando recursos: Named Caches

Durante una ejecución del **checkstorage**, los **workspace** se ligan dinámicamente al cache configurado para la base de datos. Para la actuación ideal, el cache usado debe dedicarse a la operación del **checkstorage** (aunque esto no se requiere). El cache usado debe configurarse con un pool de 16K.

El tamaño del cache recomendado por **sp_plan_dbccdb** aplica al buffer pool de 16K y es igual a:

- El 20% del tamaño del **workspace** para la base de datos designada.
- 640K por **worker process** usado.

Asignando más memoria al buffer pool de 16K que el recomendado por **sp_plan_dbccdb** puede aumentar el performance de **checkstorage**.

El comando **checkstorage** sólo usa el buffer pool de 16K en el cache asignado a él. Por consiguiente, si un cache especialmente se usa y tiene algo más que:

- 0.5MB 2K buffer pool
- 16K buffer pool

¡Entonces la memoria no usada se limpia!

Si el cache usado es compartido por otras aplicaciones, el **checkstorage** puede experimentar una degradación de la actuación mientras está corriendo.

Planeando recursos: Worker Processes

Considere los siguientes cuatro conceptos:

- **Ws**: El número total de **worker processes** configurado en el Adaptive Server.
- **Wp**: El número de **worker processes** sugerido por **sp_plan_dbccdb** uno por dispositivo en el que la base de datos designada reside.
- **Wc**: El número de **workers** configurado para una base de datos particular.
- **Wu**: El número de **workers** realmente usado por **checkstorage**.

El número de **worker processes** configurado para el **checkstorage** no es el número necesariamente usado. Los **checkstorage** pueden usar menos **worker processes** que **Wp** o **Wc**.

Fase 2: Inicializando la base de datos dbccdb

Después de que la fase de planeación de recursos está completa, hay que hacer los siguientes pasos de instalación:

1. Inicializar los dispositivos (optativo).
2. Crear la base de datos **dbccdb**.
3. Configurar el log de **dbccdb** (optativo).
4. Crear y mapear los segmentos de la base de datos (optativo).
5. Crear e inicializar las tablas del **dbccdb**.
6. Crear el **workspaces**.
7. Configurar **dbccdb** para las bases de datos designado.

Paso 1: Inicialice los dispositivos usando el comando **disk init**. Estar seguro de usar dispositivos físicos particulares cuando sea posible.

Paso 2: Crear la base de datos **dbccdb**. Estar seguro de poner los datos y el segmento de log para el **dbccdb** en dispositivos diferentes, para evitar problemas y performance bajo.

Paso 3: Configurar la **dbccdb** para el manejo del log que usa uno de los métodos siguientes:

- Habilitar la opción **truncate log on chkpt** de la base de datos **dbccdb**.

- Agregar un umbral de última oportunidad en **dbccdb** para que el log de transacciones se respalde automáticamente.

El fracaso para realizar estos pasos puede producir que el log se llene y el **checkstorage** se suspende o es terminado por el usuario.

Paso 4: Crear y mapear segmentos de la base de datos para el **workspaces**.

- Este paso es optativo; sólo aplicable cuando se esperan que las bases de datos analizándose crecerán con el tiempo.
- Si los tamaños de la base de datos designada son estáticos, saltarse este paso.
- Usar segmentos para asegurar crecimientos futuros del **workspaces**.
- Ejemplo sobre el uso de segmentos:

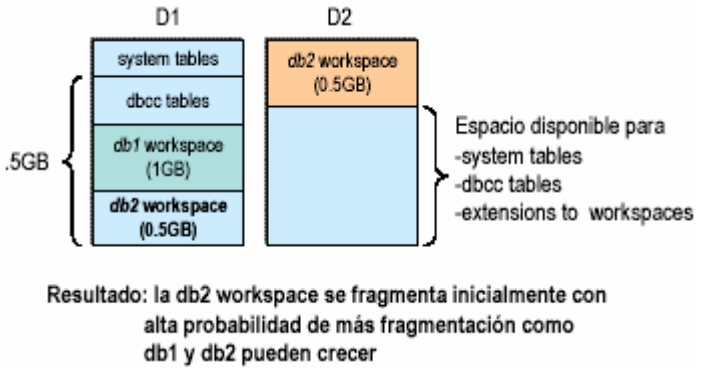


Ilustración 10-72. Asignación de espacio: no hay segmentos usados.

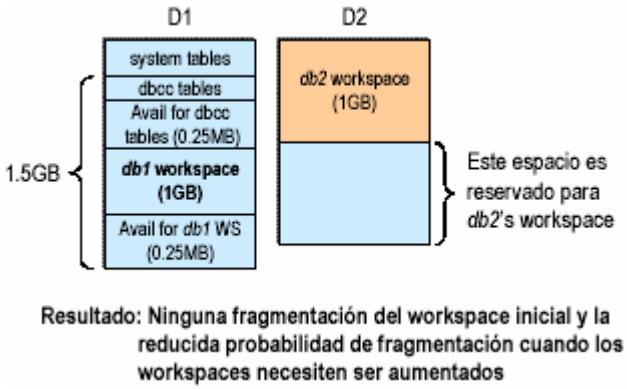


Ilustración 10-73. Asignación de espacio: con segmentos usados.

Paso 5: Crear e inicializar las tablas del **dbccdb**.

- El script **installdbccdb** es proporcionado para:
  - Instalar las ocho tablas de control de **dbcc** (es necesario).
  - Instala los procedimientos del **dbcc**.
  - Inicializa la tabla **dbcc_types** insertando 80 filas.

- Si una tabla ya existe, se borra y sólo se recrea si una tabla de la que depende no existe

Paso 6: Crear el **workspace**

- El procedimiento **sp_dbcc_createws** se usa para crear los dos **scan** y **workspaces** del texto (vía el **dbcc createws**).
- Un nombre del segmento debe proporcionarse si el usuario no definió segmentos, entonces el segmento predefinido debe especificarse.

Paso 7: Configurar las bases de datos fuentes en **dbccdb**

- Antes de que puedan ejecutarse **checkstorage**, debe configurarse información sobre la base de datos a analizar explícitamente en la base de datos **dbccdb**.
- El procedimiento **sp_dbcc_updateconfig** es usado para configurar atributos de base de datos especificados en **dbccdb**.
- Pueden configurarse siete atributos diferentes cuatro obligatorios debe ponerse de la base de datos a analizarse antes del **checkstorage** correrá contra él.
- Estos atributos se describen en la tabla siguiente:

Nombre del Atributo	¿Requerido? / Default	Descripción
max worker processes	**Yes/NA**	Máximo número de worker processes que pueden ser usados.
dbcc named cache	**Yes/NA**	Nombre y tamaño del cache ligado.
scan workspace	**Yes/NA**	Nombre de scan workspace.
text workspace	**Yes/NA**	Nombre de text workspace.
OAM count threshold	No / 2%	Si la página de contadores en el OAM de un objeto difiere de la página de contadores real por más de este porcentaje, entonces los chequeos del objeto se descontinuará.
IO error abort	No / 10	El número de errores de I/O permitido en un dispositivo particular antes del chequeo de objetos en ese dispositivo se discontinuará
linkage error abort	No / 10	El número de errores de la unión permitido en un objeto particular antes del chequeo de ese objeto se discontinuará

Tabla 10-2. Atributos del sp_dbcc_updateconfig a analizarse antes del checkstorage en SQL Server.

Configurando dbccdb para una base de datos fuente

El procedimiento **sp_dbcc_updateconfig** debe ejecutarse cuatro veces por lo menos para configurar una base de datos analizada particular. Por ejemplo, los siguientes cuatro comandos se necesitan para configurar la base de datos **db1**:

```
sp_dbcc_updateconfig db1, 'max worker processes', '2'
sp_dbcc_updateconfig db1,'dbcc named cache','cache1','8M'
sp_dbcc_updateconfig db1,'scan workspace', scan_ws
sp_dbcc_updateconfig db1,'text workspace', text_ws
```

Código 10-48. Uso de sp_dbcc_updateconfig para configurar una base de datos en SQL Server.

Evaluando la configuración de la base de datos fuente

El **checkstorage** graba información considerando:

- Cualquier error que descubrió en la base de datos.
- Las estadísticas características de los datos en la base de datos designado.

El procedimiento **sp_dbcc_evaluatedb** usa datos estadísticos para:

- Reevaluar la configuración de una base de datos designado, y recomendar valores más exactos para:

  Named cache size.

  Scan workspace size.

  Text workspace size.

  Process count.

Usar el procedimiento **sp_dbcc_evaluatedb** para determinar por qué los **checkstorage** no correrán en algunas bases de datos.

Usando sp_dbcc_evaluatedb

Para debugear problemas de **checkstorage**

- Si los **checkstorage** no corren contra una base de datos particular, se puede usar el procedimiento **sp_dbcc_evaluatedb** (en algunos casos) ayuda a analizar el problema.

Los datos de configuración no existen.

- Si el **sp_dbcc_evaluatedb** se corre contra una base de datos que no contiene ningún dato de configuración en la tabla **dbccdb..dbcc_config**, fallará con el error siguiente:

```
Msg 18474
Sales database is not configured for DBCC in the dbcc_config table. Use
sp_dbcc_updateconfig to configure it.
```

Código 10-49. Salida de error del uso de sp_dbcc_evaluatedb sin datos de configuración en SQL Server.

En este escenario, los usuarios deben:

- Ejecutar **sp_plan_dbccdb** para conseguir estimaciones.
- Entonces, ejecutar **sp_dbcc_updateconfig** para insertar los datos de configuración.

Los datos de configuración no existen

- Si el **sp_dbcc_evaluatedb** se corre contra una base de datos con por lo menos una fila correspondiente en la tabla del **dbcc_config**:

  Examina la configuración actual.

  Examina datos del **checkstorage** previamente ejecutado.

  Despliega la configuración actual.

  Despliega la configuración recomendada.

- Para situaciones en las que los **checkstorage** no corren, usar la configuración actual y la configuración recomendada para poner a punto.
- Los ejemplos siguientes ilustran algunos usos de este procedimiento.

Poniendo a punto problemas de instalación.

Incluido el bajo rendimiento de ejecución se usa el **sp_dbcc_evaluatedb** contra una base de datos con extrañas opciones de configuración:

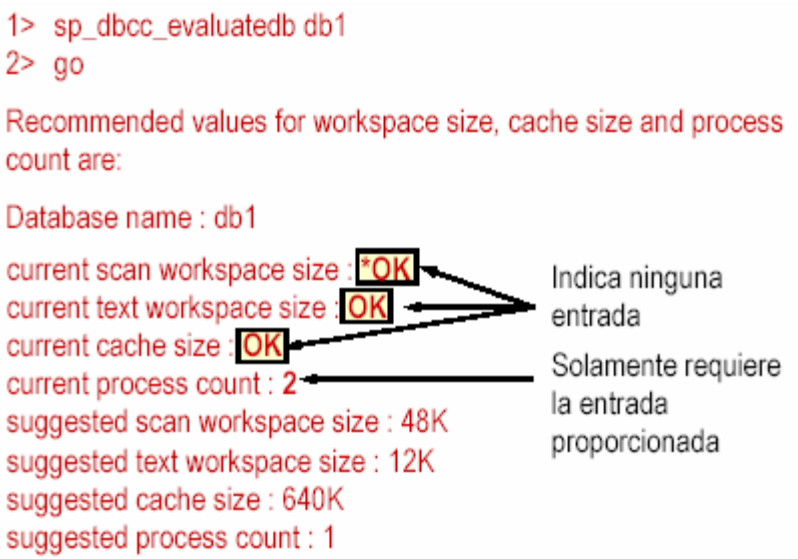


Ilustración 10-74. Uso de sp_dbcc_evaluated en SQL Server.

Si desplegó valores actuales menores a valores sugeridos, los **checkstorage** no correrán. ***OK (ceros K)** indica ninguna entrada o una entrada de **0**.

Trampas potenciales de la instalación

Los **checkstorage** no correrán.

Algunas situaciones impiden al **checkstorage** correr y producen errores:

Situación	Error
La base de datos dbccdb no encontrada	9964
La tabla dbcc no existe	208 & 9965
La configuración de la tabla dbcc_config no es correcta	9965
El número de worker processes disponible es menor que el número configurado	9961
El scan workspace no existe	9966
El scan workspace existe, pero el ID del objeto no es el mismo en dbcc_config	9966
El scan workspace es demasiado pequeño	9952
El text workspace no existe	9967
El text workspace existe, Pero el ID del objeto no es el mismo del ID en dbcc_config	9967
El cache referido no existe	9977
El buffer pool de 16K en el cache no existe o es muy corto	9978

Tabla 10-3. Errores al ejecutar checkstorage en SQL Server.

Funcionamiento interno de checkstorage

Arquitectura del dbcc checkstorage

La operación del **checkstorage** incluye 5 fases:

- Inicialización.
- Búsqueda de base de datos.

- Chequeos de ligado de páginas.
- Chequeo de OAM.
- Terminación.

Cada fase lee y escribe a la base de datos de **dbcc**.

Antes de discutir los detalles de cada fase, una vista previa rápida del esquema del **dbcc** base de datos es necesaria.

Fase database scan

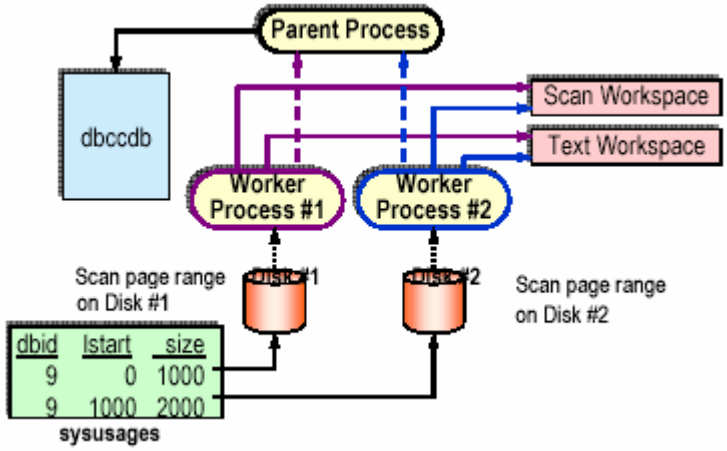


Ilustración 10-75. Apreciación global del database scan usando dos worker processes.

Manejador de errores informados por checkstorage

Manejando faults

Examinando resultados

El primer paso en el manejo de **Faults** descubiertos por **checkstorage** es comprender por que ellos existen. El **checkstorage** no escribe sus resultados a un archivo, graba sus resultados en la base de datos **dbccdb**. Por consiguiente, se exigen a los usuarios que examinen los resultados del chequeo explícitamente. Lo que el **checkstorage** hace, despliega un mensaje sumario a la sesión en la que fue invocado que incluye:

- Número de la secuencial (opid) de la ejecución descrita.
- Número actual (hard) de **Faults** y **Faults** sospechosos.
- El número a nivel objeto de chequeos abortados.

Resumen de mensajes del checkstorage

El **checkstorage** típicamente despliega cualquiera de los siguientes dos mensajes a la sesión en la que se invocó:

- **Storage checks for 'ChrisDB' are complete. DBCC is now recording the results in the dbccdb database.**

- **DBCC CHECKSTORAGE for database 'ChrisDB' sequence 1 completed at May 20 1997 8:29AM. 3 faults and 0 suspect conditions were located. 0 checks were aborted.You should investigate the recorded faults, and plan a course of action that will correct them.**

El Adaptive Server provee dos mecanismos para examinar los resultados del **checkstorage**:

- Usando el procedimientos **dbcc**.
- Accediendo a la base de datos de **dbcc** directamente.

Procedimientos del sistema **dbcc**. Estos procedimientos se proporcionan con la base de datos **dbccdb**:

- **sp_dbcc_summaryreport**: Despliega un resumen de los datos como el número de errores duros y suaves informado contra la base de datos especificada para cada ejecución del **checkstorage** guardada en **dbccdb**.
- **sp_dbcc_faultreport**: Despliega el nombre del objeto, ID del índice, número de la página, y descripción de cada error encontrado en la última ejecución del **checkstorage**.
- **sp_dbcc_differentialreport**: Compara valores de contadores desde dos casos de **checkstorage** en la base de datos **dbccdb** y reporta sólo valores que cambiaron (los contadores aplican a faltas y estadísticas).
- **sp_dbcc_statisticsreport**: Despliega los valores del contador actuales para cada objeto en la base de datos designado; su ejecución puede ser muy larga.
- **sp_dbcc_configreport**: Despliega la configuración actual para la base de datos designada; similar al **sp_dbcc_evaluatedb**.
- **sp_dbcc_fullreport**: Despliega la configuración, estadísticas y datos de faltas para una base de datos dada u objeto.

Usando **dbcc stored procedures**

sp_dbcc_summaryreport. Se usa este procedimiento para desplegar resultados sumarios sobre las bases de datos en **dbccdb**. Este procedimiento despliega esta información para cada ejecución grabada de **checkstorage**; por consiguiente, el rendimiento puede abarcar mucho tiempo. A continuación se muestra la salida de este procedimiento:

```
1> sp_dbcc_summaryreport
2> go
```

Database	Date	Start time	End Time	Hard	Soft	Text	Abort Count	
db1	05/02/97	10:01:47	10:02:25	1	4	0		0
db2	05/02/97	12:10:02	12:10:08	0	9	0		0
db2	05/02/97	12:45:11	12:45:16	2	7	0		1

Ilustración 10-76. Salida de sp_dbcc_summaryreport en SQL Server.

Usando **dbcc stored procedures**. Se usa esta información para determinar rápidamente:

- El número de faltas en una base de datos.
- Número de chequeos abortados (da el número de objeto con problemas en su unión de página).

- Promedio del tiempo de la ejecución.
- Número de valores no-nulos tipo texto.

Usando procedimientos de Sybase

`sp_dbcc_faultreport`: Reporte corto.

- Este procedimiento despliega información adicional sobre las faltas particulares descubiertas por la más reciente ejecución de **`checkstorage`**.
- Por defecto, este procedimiento se ejecuta en el modo corto:

```
1> sp_dbcc_faultreport sybsystemprocs
2> go

Database Name: sybsystemprocs

Table Name         Index     Type Code Description        Page Number
-----------------  --------- --------- -----------------  -----------
sysprocedures      0         100031    page not allocated 5702
sysprocedures      1         100031    page not allocated 14151
syslogs            0         100022    chain start error  24315
syslogs            0         100031    page not allocated 24315
```

Ilustración 10-77. Uso de sp_dbcc_faultreport en el modo corto en SQL Server.

Se usa esta información para determinar rápidamente:

- Los **`objects`**/**`indid`** específicos involucrados.
- Los números de la página específicos involucrados.
- La descripción breve de la falta que descubrió.

Esta salida no indica qué faltas son duras y qué es suave.

`sp_dbcc_faultreport`: Reporte largo

- El informe largo de este procedimiento se invoca vía la sintaxis siguiente:

```
sp_dbcc_faultreport long, [database name]
```

Código 10-50. Sintaxis de sp_dbcc_faultreport de reporte largo en SQL Server.

- Este reporte es más extenso que para el informe corto, como el mostrado:

```
1> sp_dbcc_faultreport long, jakedb
2> go
Generating 'Fault Report' for object mytab in database jakedb.
Fault Code: 100022; Hard fault
chain start reference disagrees with page linkage.
page id: 313
page header: 0x0000013900000000000013900F430100000056F00...
Header for 313, next 0, previous 313, id = 16003088:0
        TS = 0x00010000056F, next row = 0, level = 0
        free offset = 32, minlen = 70, status = 129(0x0081)
Generating 'Fault Report' for object mytab in database jakedb.
Fault Code: 100002; Soft fault, possibly spurious
page free offset value in header is invalid.
```

```
page id: 313
page header: 0x0000013900000000000013900F430100000056F00...
Header for 313, next 0, previous 313, id = 16003088:0
        TS = 0x00010000056F, next row = 0, level = 0
        free offset = 0, minlen = 70, status = 129(0x0081)
```

Código 10-51. Uso del reporte largo de sp_dbcc_faultreport en SQL Server.

Directamente vía **dbccdb**. Dependiendo de los requisitos específicos de un usuario y nivel especializado, hay que considerar hacer consultas directamente a la tabla del **dbccdb**.

Resumiendo

El comando **checkstorage** introduce un nuevo paradigma para inspeccionar la consistencia del Adaptive Server. Se requiere trabajo adicional para:

- Preparar la base de datos de **dbcc**.
- Examinar los resultados del **checkstorage**.

El resultado es que los clientes que inspeccionaban la consistencia de sus bases de datos en el Sistema 11 pueden hacerla así en Adaptive Server 11.5.

10.6 RESOURCE GOVERNOR

Aplicación, características, rasgos, habilitar, cree límites activos del administrador de recursos.

Descripción

El Administrador de recursos provee una manera para que el administrador del sistema pueda poner límites en recursos de ASE. El tipo de límites incluye:

- **I/O cost** (estimado y actual)
- **Elapsed time** (de un bach o transacción)
- **Number of rows**

Acciones para asumir cuando ocurra una violación al límite:

- **Warning**
- **Aborting the query batch**
- **Aborting transaction**
- **Killing user session**

Los límites del recurso pueden estar activos en momentos diferentes.

Beneficios

Las restricciones se pueden fijar a logins individuales o aplicaciones. Tiene la habilidad de crear nombres para rangos de tiempo durante los cuales las restricciones están en efecto.

Habilitando límites de recurso en ASE

El ASE debe configurarse para permitir límites de recurso. Se usa la opción **`allow resource limits`** de la configuración.

```
1>  sp_configure "allow resource limits",1
2>  go
Parameter Name                  Default     Memory Used  Config Value Run Value
------------------------------- ----------- ------------ ------------ -----------
allow resource limits                     0            0            1           0
Configuration option changed. ASE must be rebooted before the change in effect since the
option is static.

(return status = 0)
```

Ilustración 10-78. Uso de sp_configure para permitir límites de recurso en SQL Server.

Creando un límite de recurso

Un límite de recurso se crea usando el procedimiento **`sp_add_resource_limit`**

```
sp_add_resource_limit  name,
                       appname,
                       rangename,
                       limittype,
                       limit_value,
                       enforced,
                       action,
                       scope
```

Código 10-52. Sintaxis de sp_add_resource para poner límite a un recurso.

Límite de recurso (**`name y appname`**)

> **`name`**. El login al cual se le aplicara el límite de recurso.
>
> **`appname`**. El nombre de la aplicación al que el límite de recurso aplica.

Uno de los dos parámetros puede ser **`NULL`**

- Para crear un límite que aplica a todos los usuarios de una aplicación particular, se especifica el **`name`** de **`NULL`**.
- Para crear un límite que aplica a todas las aplicaciones usada por un login del ASE, se especifica un **`appname`** de **`NULL`**.

Ambos **`name`** y **`appname`** pueden ser no **`NULL`**. Para crear un límite que aplica a un login específico que usa una aplicación específica, se indica ambos **`name`** y **`appname`**. Por lo menos uno debe ser no **`NULL`**.

Creando una jerarquía de límite de recurso

Una jerarquía de límite de recurso puede ser creada asignando límites separadamente a la aplicación y a login.

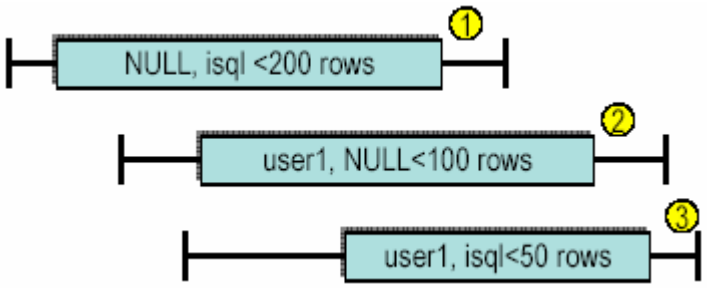


Ilustración 10-79. Esquema de una jerarquía poniendo límites de recursos.

Rango Time para el límite de recurso (rangename)

El rango de tiempo durante el cual el límite esta aplicando es creado por el procedimiento **sp_add_time_range**.

El rango de tiempo predefinido se llama **at all times**.

- Trabaja todo el tiempo, desde el primer día de la semana hasta el último, desde 00:00 a 23:59.
- No puede ser modificado o borrado.

Tipos de límite de recurso (limittype)

I/O cost (**io_cost**)

- **showplan** proporciona información que estima el costo I/O.
- **statistics io** proporciona información del costo real de I/O.
- Tiempo transcurrido de ejecución (**elapsed_time**).

El número de filas devueltas (**row_count**)

Valor de límite de recurso (limit_value)

El valor se expresa como un valor del tipo **integer**

El valor especifica:

- **I/O cost** en suma total.
- **Elapsed time** en segundos.
- El número de renglones.

Para tener idea de posibles valores limites hay que usar io_cost, ejecutando la opción **set statistics io** o **set showplan**

- Porque es basado en las estadísticas del índice, cláusulas del query, y la suposición de I/O físicos, el estimado para un query devuelta por **showplan** puede ser más alto que el costo real vuelto por **statistics io,** particularmente si las páginas ya están en cache.

Un límite en **io_cost** controla la cantidad de I/O que un query puede hacer. Este tipo del límite puede usarse para controlar I/O de queries intensivos, pero no controla queries simples que devuelven un juego del resultado grande.

Un límite en **row_count** controla el número de filas devuelto a un usuario desde un query. No controla preguntas complejas que devuelven un juego del resultado pequeño. No incluye filas generadas en tablas temporales de **temdb**.

Un límite en **elapsed_time** controla la cantidad de tiempo que toma para devolverle resultados al usuario. Hay que tener presente que una pregunta puede tomar un tiempo largo debido a su complejidad, la carga del servidor, la espera por bloqueos, y otros.

Momento del límite de recurso (enforced)

Pueden forzarse los límites de recurso (1) a priori o (2) durante la ejecución del query. La siguiente tabla lista los valores válidos por cada tipo de límite:

Enforced	Descripción	Tipo de Limite
1	La acción se toma cuando el costo estimado de ejecución excede el límite especificado.	io_cost
2	La acción se toma cuando el contador de fila real, tiempo transcurrido, o el costo de ejecución excede el límite especificado.	row_count elapsed_time io_cost
3	La acción se toma cuando el costo estimado o el costo real excede el límite especificado.	io_cost

Tabla 10-4. Valores validos (enforced) por tipo de límite.

Acciones a tomar cuando hay una violación al limite (action)

Especifica la acción para tomar cuando el límite se excede.

Los códigos de acción siguientes son válidos para todos los tipos del límite:

Acción	Descripción
1	Emite una advertencia
2	Aborta el query batch
3	Aborta la transaction
4	Elimina la sesión

Tabla 10-5. Códigos de acción validos cuando un límite se excede.

Alcances de límite de recurso (scope)

El uso del recurso acumulativo puede calcularse encima de las agrupaciones de declaraciones de SQL.

```
Query (1)
Select * from titles
Go

Query batch (2)
Select * from titles
Select * from publishers
Go
```

```
Transaction (4)
Begin tran
Insert
Update
Commit tran
```
Código 10-53. Agrupaciones de declaraciones SQL alcanzadas por el límite de recurso (scope).

Límite de recurso (row_count)

A continuación se crea un límite de recurso que es activo **at all times** (default) para prevenir que aplicación **isql** no retorne más de 200 renglones; la violación se produce terminando el batch:

```
1> sp_add_resource_limit
2> NULL,'isql','at all times','row_count',200,
3> 2, 2, 1
4> go
New resource limit created.
(return status = 0)
1>
```
Código 10-54. Uso de sp_add_resource_limit para limitar isql a 200 filas en SQL Server.

Violación del recurso

Perspectiva del usuario

```
1> select * from test_table
2> exec sp_who
2> go
.       .       .
198 Testing
199 Testing
200 Testing
Row count exceeded limit of 200.
Command batch has been aborted.
```
Código 10-55. Perspectiva del usuario al realizar una violación del recurso en SQL Server.

Perspectiva del SA

Perspectiva del SA de la violación del recurso (errorlog):

```
edeme2% tail $SYBASE/install/errorlog_SYBASE
.
00:0000:96/08/02 15:14:53.27 server User 'user1', application 'isql' exceeded
row count limit of 200.
```
Código 10-56. Perspectiva de SA al realizarse una violación del recurso en SQL Server.

Límite de Recurso (elapsed_time)

A continuación se cree un límite de recurso que es activo **at all times** (default) para impedir al **isql** pasar más de 120 segundos para cada transacción; la violación produce el abortar la transacción

```
1> sp_add_resource_limit
2> NULL,'isql','at all times','elapsed_time',
3> 120, 2, 3, 4
4> go
New resource limit created.
```
Código 10-57. Creación de un límite de recurso para isql a impedir pasar más de 120 segundos para cada transacción en SQL Server.

Ejemplo de límite de recurso (*elapsed_time*)

```
1> begin tran
2> insert into test_table values (1, "Insert")
3> go
1> /*Waiting 120 seconds*/
Elapsed time exceeded limit of 120.
Transaction has been aborted.
1>
```

Código 10-58. Perspectiva del usuario al realizar una violación del recurso elapsed_time en SQL Server.

spt_limits_types

La tabla de sistema **spt_limit_types** mantiene los tipos de límite.

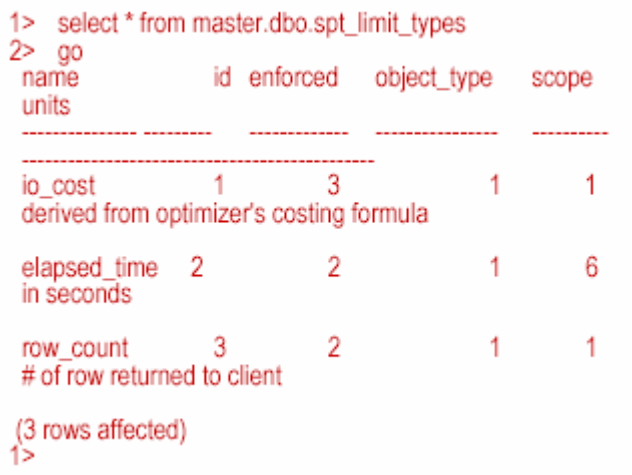


Ilustración 10-80. Contenido de la tabla de sistema spt_limit_types en SQL Server.

sysresourcelimits

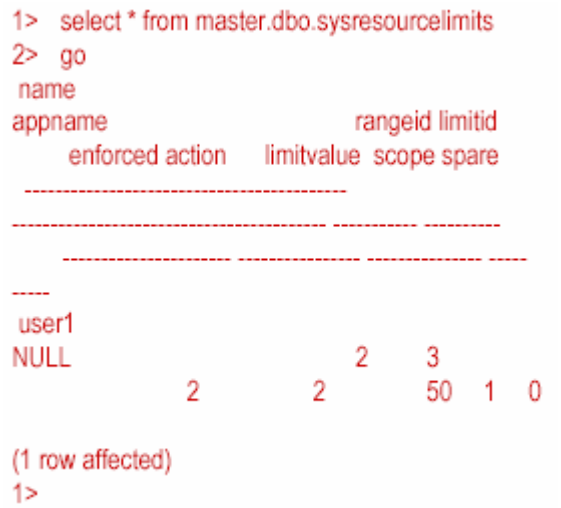


Ilustración 10-81. Contenido de la tabla de sistema sysresourcelimits en SQL Server.

- Por lo menos uno de los atributos **name** y **appname** debe ser **non NULL**.
- Un límite se liga a la sesión del usuario en momento de entrar a sesión.
- Se informan errores para los nombres del login inválidos, el rango de tiempo, o los tipos del límite no encontrados en **syslogins**, **systimeranges**, y **spt_limit_types**.
- Los **limitvalue** deben ser positivos.
- Solamente pueden limitarse los **io_cost** antes de la ejecución.
- Los tipos **row_count** y **elapsed_time** limitan durante la ejecución.
- Esta predefinida como acción de la violación para todos los límites el aborte del lote.

Modificando límites de recurso

Se modifica los valores del recurso de límite usando **sp_modify_resource_limit**:

```
sp_modify_resource_limit
{ name , appname }, rangename, limittype
[, limitvalue] [, enforced] [, action]
[, scope]
```

Código 10-59. Sintaxis de sp_modify_resource_limit para modificar límites de recurso.

Pueden modificarse sólo el valor del límite y la acción a tomar. Hay que borrar y recrear el límite si se requieren modificación otros parámetros.

Ejemplo de modificación de límite de recurso a cambiar **@limitvalue** de **200** a **50**:

```
1> sp_modify_resource_limit
2> NULL, 'isql','at all times','row_count',50,
3> 2, 2, 1
4> go
(return status = 0)
```

Código 10-60. Cambio de un límite de recurso @limitvalue en SQL Server.

Probando el nuevo límite del recurso:

```
1> select * from test_table
2> go
...
48 Testing
49 Testing
50 Testing
Row count exceeded limit of 50.
Command batch has been aborted.
```

Código 10-61. Salida de un límite de recurso excedido en SQL Server.

Consiguiendo información sobre los límites de recurso

Se usa el procedimiento de sistema **sp_help_resource_limit** para determinar qué límites del recurso aplican a un usuario dado, aplicación, o tiempo del día.

```
sp_help_resource_limit [name [, appname [, limittime [, limitday [,scope
[,action]]]]]]
```

Código 10-62. Sintaxis de sp_help_resource_limit para determinar los límites de recursos por usuario en SQL Server.

```
sp_help_resource_limit NULL, my_app
sp_help_resource_limit NULL, NULL, "09:00"
sp_help_resource_limit joe_user, NULL, NULL, Sunday
```
Código 10-63. Uso de sp_help para determinar los límites de recurso en SQL Server.

Borrando límites de recursos

Para borrar uno o más recursos se usa el procedimiento **sp_drop_resource_limit** (debe ser sa).

```
sp_drop_resource_limit {name, appname} [,rangename,
limittype, enforced, action, scope]
```
Código 10-64. Sintaxis de sp_drop_resource_limit en SQL Server.

name. El ASE fuerza al login para que el límite aplique:

- Especificar a un usuario para borrar límites de recurso de un login.
- Para Borrar límites de recurso que aplican a todos los usuarios de una aplicación en particular, especifique un nombre de **NULL**.

appname. La aplicación a la que el límite aplica:

- Especifique un nombre de la aplicación para borrar los límites de recurso de esa aplicación.
- Para borrar límites de recurso que aplican a todas las aplicaciones usadas por el login especificado, especifique un appname de **NULL**.

rangename. El rango de tiempo durante el que el límite trabaja:

- Éste debe ser un rango de tiempo existiendo o **NULL** para anular todos los límites del recurso para el nombre especificado, **appname**, **limittype**, acción, y alcance, sin tener en cuenta el **rangename**.

limittype. El tipo de recurso que está limitado, este debe ser uno de los siguientes

Tipo de Límite	Descripción
row_count	Borra sólo límites que restringen el número de filas que un query pueden devolver.
elapsed_time	Borra sólo límites que restringen el número de segundos que un query batch o la transacción puede correr.
io_cost	Borra sólo límites de costo que restringen real o estimado del query que procesa.
NULL	Borra todos los límites del recurso con el nombre especificado, appname, rangename, acción, y alcance, sin tener en cuenta el limittype

Tabla 10-6. Tipo de limite (limittype) en SQL Server.

action. La acción tomada cuando el límite se excede:

action	Descripción
1	Borra sólo límites que emiten una advertencia.
2	Borra sólo límites que abortan el query batch.
3	Borra sólo límites que abortan la transacción.
4	Borra sólo límites que eliminan la sesión.
NULL	Borra todos los límites de recurso con el nombre especificado, appname, rangename, limittype, y alcance, sin tener en cuenta la acción que toman.

Tabla 10-7. Opciones de acción a tomar cuando el límite se excede en SQL Server.

scope. El alcance del límite este debe ser uno de los siguientes:

scope	Descripción
1	Borra sólo límites a los que aplican a queries.
2	Borra sólo límites a los que aplican a query batches.
4	Borra sólo límites a los que aplican a transactions.
6	Borra sólo límites a los que aplican a ambos query batches y transactions.
NULL	Borra todos los límites de recurso con el nombre especificado, appname, rangename, limittype, y acción, sin tener en cuenta su alcance.

Tabla 10-8. Opciones en los límites en SQL Server.

Ejemplo. Borrar un solo límite del recurso que mata la sesión siempre que **joe** usa la aplicación de nómina los viernes por la tarde y tienen resultados en costo de I/O excesivo:

```
sp_drop_resource_limit joe, payroll, friday_afternoon, io_cost, NULL, 4, 1
```

Código 10-65. Uso de sp_drop_resource_limit en SQL Server.

Ejemplo. Borra todos los límites que aplican al uso de **joe** de la aplicación de la nómina:

```
sp_drop_resource_limit joe, payroll, NULL, NULL, NULL, NULL, NULL
```

Código 10-66. Uso de sp_drop_resource_limit en SQL Server.

Ejemplo. Borra todos los límites que aplican al usuario **joe**.

```
sp_drop_resource_limitjoe, NULL, NULL, NULL, NULL, NULL, NULL
```

Código 10-67. Uso de sp_drop_resource_limit en SQL Server.

Ejemplo. Borra todos los límites de recurso que aplican a la aplicación de la nómina:

```
sp_drop_resource_limit NULL, payroll, NULL, NULL,NULL, NULL, NULL
```

Código 10-68. Uso de sp_drop_resource_limit en SQL Server.

Ejemplo. Borra todos los límites de recurso en la aplicación de la nómina cuya acción es matar la sesión:

```
sp_drop_resource_limit NULL, payroll, NULL, NULL, 4, NULL, NULL
```

Código 10-69. Uso de sp_drop_resource_limit en SQL Server.

Se usa el procedimiento de sistema **sp_help_resource_limit** para determinar qué límites del recurso aplican a un usuario dado, aplicación, o tiempo de día.

Cuando se usa **sp_droplogin** para borrar un login de ASE, se borran todos los límites de recurso asociados con ese login automáticamente.

Creando un nombre de Time Range

Un límite de recurso activo es determinado a través de un rango de tiempo nombrado. Se crean rangos de tiempo usando el procedimiento **sp_add_time_range**.

```
sp_add_time_range       name,
                        startday,
                        endday,
                        starttime,
                        endtime
```

Código 10-70. Sintaxis de sp_add_time_range en SQL Server.

```
1> sp_add_time_range
2> "evenings", "Monday","Friday",
3> "18:00", "00:00"
4> go
```

Código 10-71. Uso de sp_add_time_range en SQL Server.

Crea un **`time range`** llamado **`evenings`**, que esta activo de lunes a viernes de las 6 p.m. a medianoche.

La tabla de sistema systimeranges

La tabla **`systimeranges`** mantiene todos los rangos de tiempo conocidos. El rango de tiempo predefinido es el de **`"at all times"`**.

```
1>  select * from systimeranges
2>  go
name          id  startday endday starttime  endtime
------------- ---- ---------- ---------- ---------- ----------
at all times   1         1       7    00:00      00:00

(1 row affected)
1>
```

Ilustración 10-82. Contenido de la tabla systimeranges en SQL Server.

`ERROR`: El rango de tiempo que se sobreponen.

¿Se Puede limitar el recurso por rangos de tiempo? No, el recurso limita por usuario, nombre de la aplicación, el **`limittype`** no se puede sobreponer.

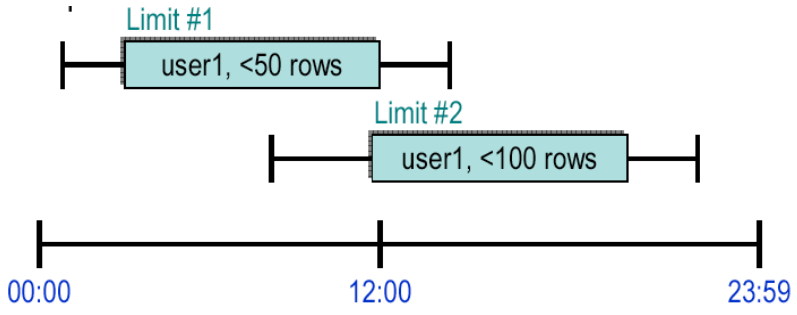


Ilustración 10-83. Uso incorrecto del rango de tiempo sobrepuesto.

Impacto del recurso

Rendimiento

El administrador de recurso causa menos del 1% de impacto del performance en el ASE en medidas de transacciones por segundo (TPC-B y TPC-C).

Memoria

El administrador de recurso requiere algunas estructuras de memoria interiores adicionales.

Cambios después de permitir límites de recurso:

- el cache del procedimiento disminuyó de 1028 a 1026 (KB).
- el número de alarmas aumentó de 2 a 4 (KB).

Administrador de recurso mantiene una manera para que el administrador del sistema ponga límites en recursos del ASE.

Los tipos del límite incluyen:

- **`I/O cost`** (estimado y actual).
- **`Elapsed time`**.
- **`Number of rows`**.

Las acciones de una violación incluyen:

- **`Warning`**.
- **`Aborting the query batch`**.
- **`Aborting transaction`**.
- **`Killing user session`**.

Los límites del recurso pueden estar activos en momentos diferentes.

Capítulo 11

MODELO ORIENTADO A OBJETOS

Principales características de la metodología orientada a objetos, como lo son: su modelado y su enfoque dentro de las empresas y organizaciones.

11.1 INTRODUCCIÓN

¿Qué es un sistema de información?

Un sistema de información lo componen los siguientes elementos básicos:

- Software.
- Hardware.
- Personas.
- Documentación.
- Procesos (reglas de negocio).
- Bases de Datos (información).

¿Qué es el software?

Las características del software incluyen que es:

- Intangible.
- Complejo.
- No se gasta pero llega a ser obsoleto.
- Es la parte más importante de los sistemas e inclusive la más costosa.
- La industria del software sigue en crecimiento.

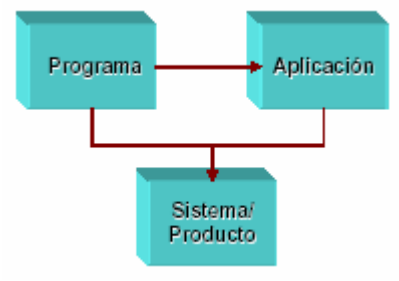


Ilustración 11-1. Esquema de la complejidad del software.

Problemática en el desarrollo de software

- Mala estimación y dimensionamiento de los proyectos.
- Fallas en el manejo de riesgos.
- Complejidad del software.
- Los ciclos de negocio son más cortos.
- La definición de requerimientos es mala y no se documentan.
- Los sistemas legados deben ser mantenidos por otras personas.
- Diseño y arquitectura inadecuados.
- No se realiza un modelado detallado de las características y componentes del software.

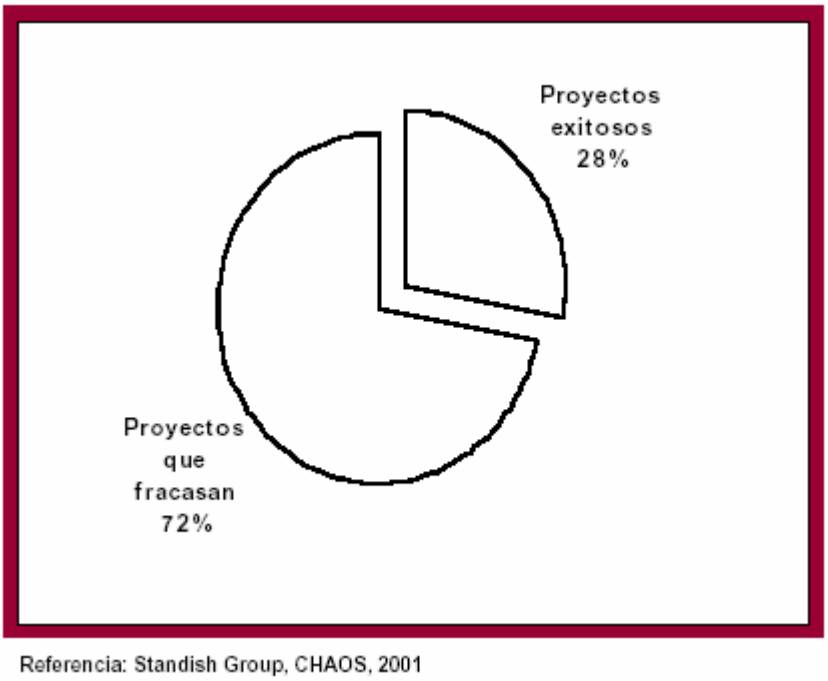


Ilustración 11-2. Comparativa de los proyectos exitosos y los que fracasan en el desarrollo de software.

El promedio del costo sobrepasado es de 189%.

El tiempo de retraso promedio es de 222%.

El promedio de fallas según las expectativas iniciales es de 61%.

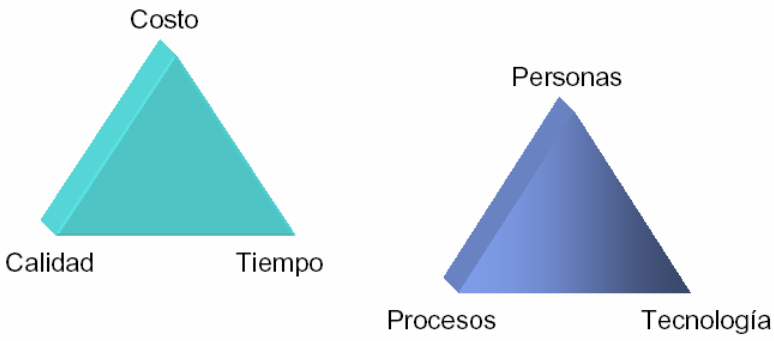


Ilustración 11-3. Triángulos en el desarrollo de software.

¿Qué es un modelo?

- Es una representación de algo real.
- Un modelo capta los aspectos importantes de lo que estamos representando, desde cierto punto de vista, y simplifica u omite el resto.
- Los modelos pueden ser físicos, gráficos, matemáticos

¿Por qué modelar?

- Es difícil comprender el todo.
- Permite dividir un problema complejo en problemas menores.
- Visualizar un sistema desde varias perspectivas.
- Entender y dimensionar el problema.
- Comprender y probar la solución.
- Abstraer las características, componentes y estructura de algo por construir.
- Detectar fallas, inconsistencias y prever cambios.
- Documentación del proyecto.
- Es menos costoso construir un modelo que un sistema.
- Comunicación entre el equipo de desarrollo y con los usuarios.

Ilustración 11-4. Ejemplo de un modelo y de su estructura final.

Aplicaciones del modelado

Construcción  , automotriz  , aeronáutica  , electricidad y electrónica, telecomunicaciones, cinematográfica, administración, software, etc.

UML permite que los usuarios, analistas, diseñadores y programadores de software comprendan un lenguaje de modelado estándar.

¿Con qué modelamos el software (si es que lo modelamos)?

- Análisis Estructurado (Yourdon).
- Metodología Jackson.
- Técnicas de Diseño y Análisis Estructurado (SADT).
- UML (Orientación a objetos).

UML

El Lenguaje de Modelado Unificado (UML) es un lenguaje de modelado visual que se usa para especificar, visualizar, construir y documentar artefactos de un sistema de software.

Fue desarrollado en un esfuerzo para simplificar y consolidar las notaciones de desarrollo orientado a objetos que habían surgido.

Apoyado por el Object Management Group (OMG), Rational Software, Microsoft, Hewlett-Packard, Oracle, Texas Instruments, MCI Systemhouse y otros.

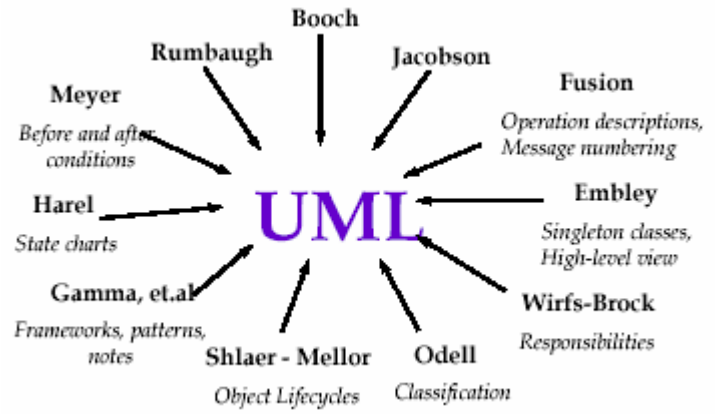


Ilustración 11-5. Autores del UML.

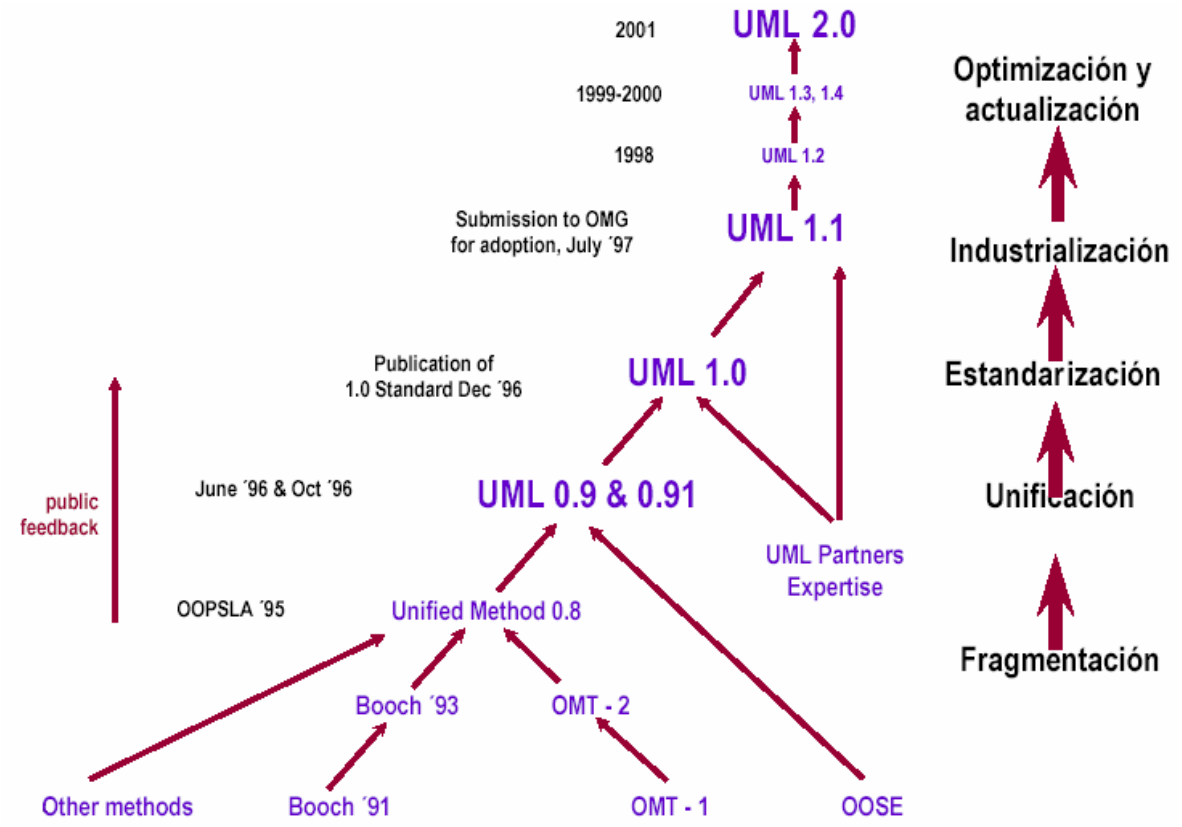


Ilustración 11-6. Evolución del UML.

Siglas UML

Lenguaje. Mediante la notación permite expresar y comunicar conocimiento.

Unificado. Integra lo mejor de varios autores, notaciones y técnicas.

Modelado. Permite representar de manera abstracta aspectos reales.

Características

- UML es un lenguaje (de modelado) y no un método.
- No es un proceso de software.
- Incluye una serie de diagramas; especifica la notación para representarlos pero no describe cómo crearlos.
- Organización del UML:

  Elementos.

  Relaciones.

  Diagramas.
- Define una notación expresiva y consistente.

- Facilita la comunicación con otros.
- Permite detectar omisiones o inconsistencias.
- Es aplicable a sistemas sencillos y complejos.
- Es un estándar en la industria de construcción de software.
- Existen herramientas en el mercado para modelar y generar código a partir de UML.

11.2 Generalidades del Paradigma Orientado a Objetos

Evolución

1967 - Se desarrolla Simula 67, el primer lenguaje reconocido como OO.

70' s - Métodos de desarrollo para lenguajes de programación como Cobol y Fortran.

80' s - Análisis estructurado y diseño estructurado (Yourdon, Ward, DeMarco, etc.). Surgen y evolucionan otros lenguajes como: Smalltalk, Objetive C, C++, Eiffel, CLOS. Primeros métodos de desarrollo orientado a objetos.

90' s - Unificación de métodos orientados a objetos.

2000 – Desarrollo por componentes, reutilización, integración e interoperabilidad de los sistemas.

Tecnología de objetos

El paradigma orientado a objetos es una filosofía para el desarrollo de sistemas basado en el modelado del mundo real a través de la identificación de objetos.

Surgió inicialmente como un enfoque para la programación pero se ha extendido a todo el ciclo de desarrollo de sistemas:

- POO: Método de implementación en el que los programas se organizan como colecciones cooperativas de objetos.
- DOO: Método de diseño que abarca el proceso de descomposición OO y una notación para describir los modelos lógico, físico, estático y dinámico del sistema.
- AOO Método de análisis que examina los requisitos desde la perspectiva de los objetos que se encuentran en el dominio del problema.

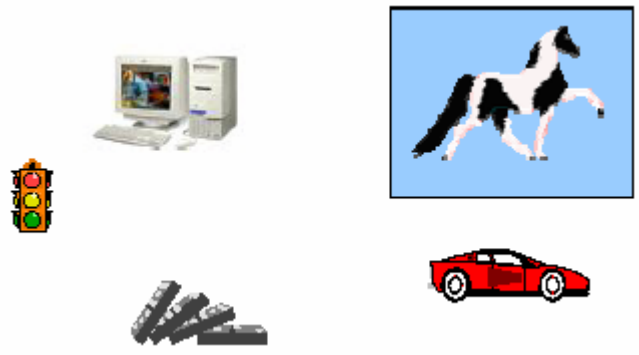


Ilustración 11-7. Ejemplos de objetos cotidianos.

Un objeto representa un elemento identificable con ciertas características (atributos) y que puede realizar un conjunto de acciones (operaciones).

¿Qué es un objeto?

De manera informal, un objeto representa una entidad, ya sea física, conceptual o de software.

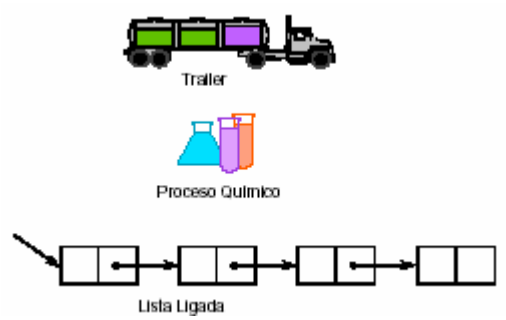


Ilustración 11-8. Aplicaciones física, conceptual o de software de la entidad de un Objeto.

En la vida diaria, un objeto es una cosa tangible o conceptual. → En análisis, un objeto es una entidad conceptual del dominio del problema. → En diseño, un objeto es una entidad detallada que representa una parte de la solución. → En código, un objeto es un bloque de código.

Los objetos son:

- La pieza fundamental que combina tanto estructura como comportamiento.
- En contraste con la programación convencional, el elemento donde la estructura de datos y el comportamiento están estrechamente relacionados.
- Un elemento con estado, comportamiento e identidad.
- Una instancia de una clase.
- Un bloque de código.

Una clase es un conjunto de objetos que comparten una estructura común y un comportamiento común.

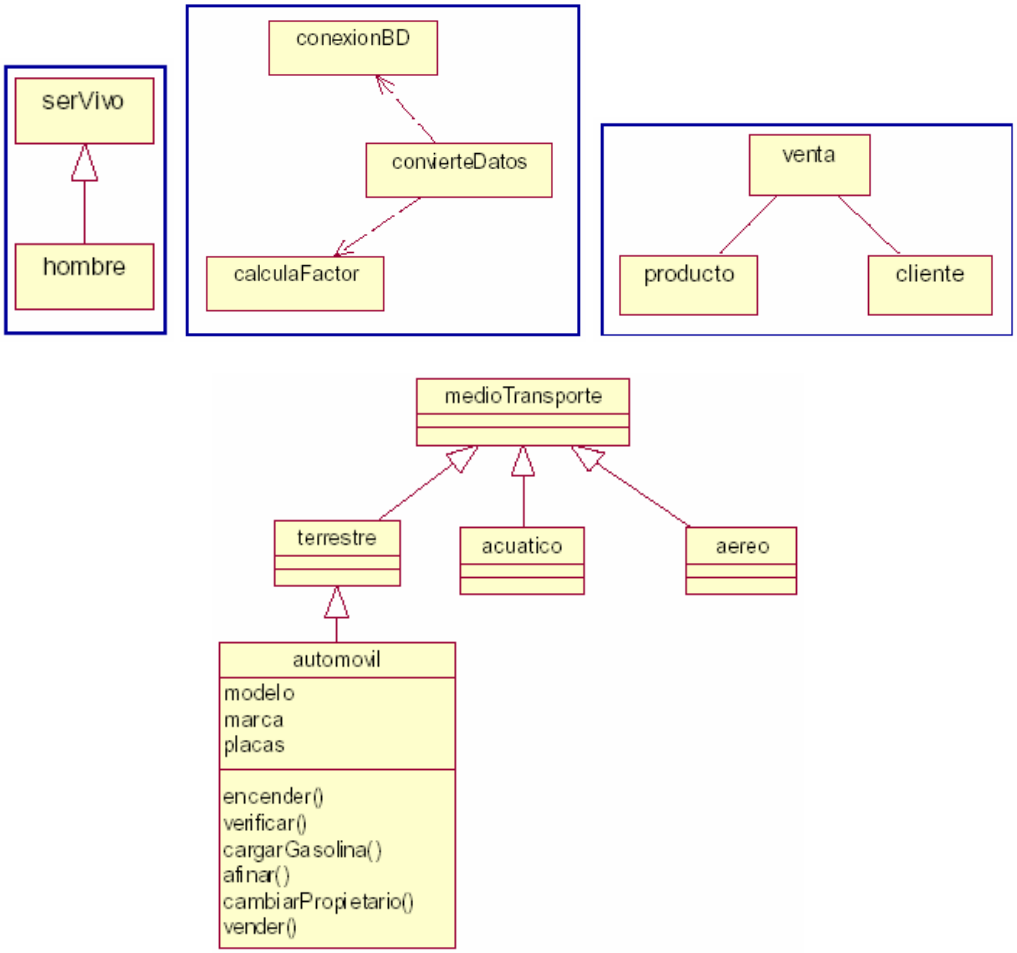


Ilustración 11-9. Ejemplos de clases.

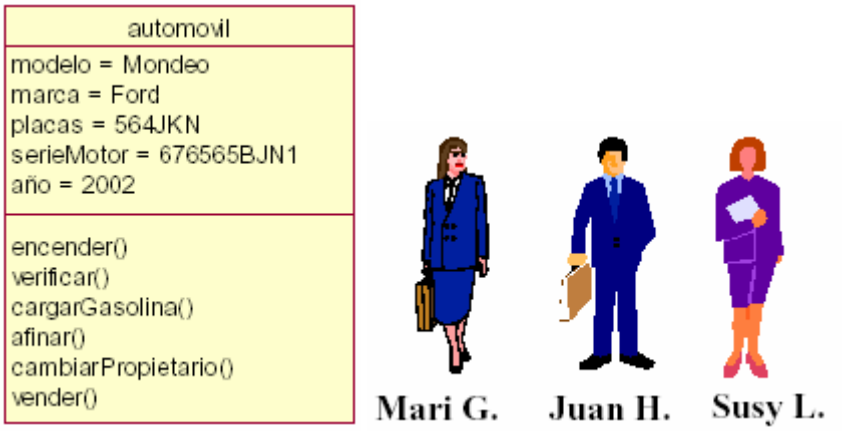


Ilustración 11-10. Ejemplos de objetos.

Los objetos

Los objetos son:

- La pieza fundamental que combina tanto estructura como comportamiento.
- En contraste con la programación convencional, el elemento donde la estructura de datos y el comportamiento están estrechamente relacionados.
- Un elemento con estado, comportamiento e identidad.
- Una instancia de una clase.
- Un bloque de código.

Conceptos

- Objeto

  Un objeto es un concepto, abstracción, o cosa con límites bien definidos y significado para una aplicación.

  Un objeto es algo que tiene:

  Estado.

  Comportamiento.

  Identidad.

  Un objeto es una instancia de una clase.

- Clase

  Una clase es una descripción de un grupo de objetos con propiedades comunes (atributos), comportamiento común (operaciones) y relaciones comunes con otros objetos (asociaciones y agregaciones).

  Una clase es una abstracción en la que ella:

  Enfatiza características relevantes.

  Suprime otras características.

  Una clase es una definición abstracta de un objeto.

  Define la estructura y comportamiento de cada objeto en la clase.

  Sirve como una plantilla para crear objetos.

- Abstracción

  Omitir las propiedades y acciones de un objeto y dejar sólo aquellas que nos interesan.

  Por ejemplo: cuando se estudia el cuerpo humano podemos abstraer diversos subsistemas (respiratorio, óseo, nervioso, etc.).

- Clasificación

  Los objetos con la misma estructura y comportamiento son agrupados en clases.

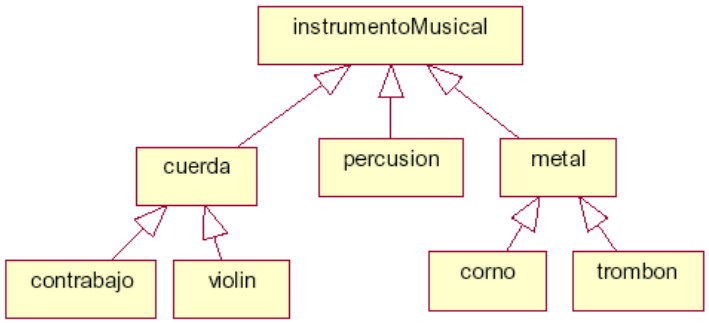


Ilustración 11-11. Esquema de clasificación de objetos en clases.

- Jerarquía

  Un sistema se compone de subsistemas (más pequeños) relacionados que tienen a su vez propios subsistemas, y así sucesivamente.

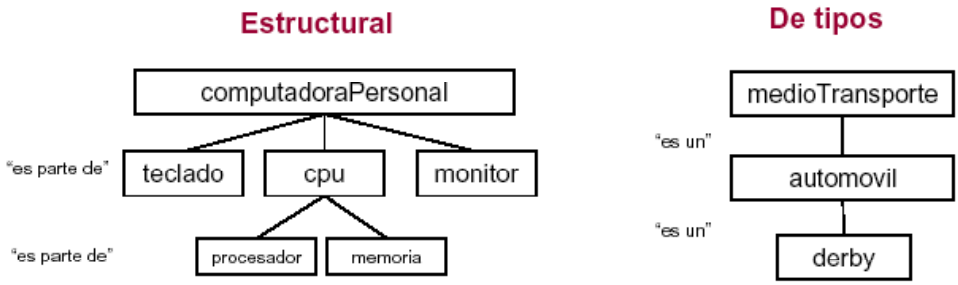


Ilustración 11-12. Esquema de jerarquía estructural y de tipos.

- Mensajes

  Para que los objetos de un sistema trabajen en conjunto, un objeto (cliente) envía a otro un mensaje para realizar una operación y el objeto receptor (proveedor) ejecutará la operación.

  Un mensaje es la petición de un servicio.

  La invocación de una operación es el tipo de mensaje más común.

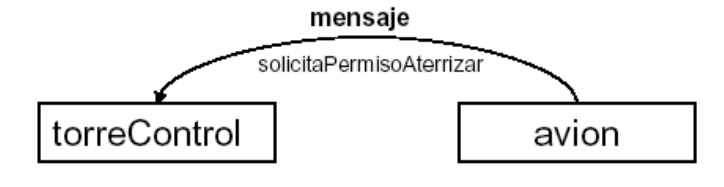


Ilustración 11-13. Esquema de un mensaje.

Un mensaje es una función de llamada (con nombre, parámetros y un resultado).

El nombre, tipos de parámetros y tipo del resultado se denomina firma (**`signature`**).

El conjunto de firmas de todas las operaciones sobre un objeto se denomina **`protocolo`**.

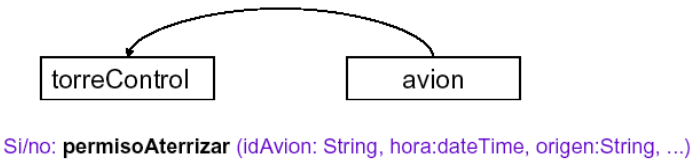


Ilustración 11-14. Esquema de un mensaje con firma y protocolo.

- Herencia

  Las clases son organizadas jerárquicamente. Las clases hijas conservan la estructura y comportamiento de las clases padres.

  La clase general es llamada superclase; la especialización subclase.

  La superclase define un comportamiento (protocolo) que todas las subclases heredan.

  Las subclases pueden agregar nuevos atributos y operaciones.

  Las subclases pueden sobre-escribir operaciones (conservando la firma pero proporcionando distintas implementaciones).

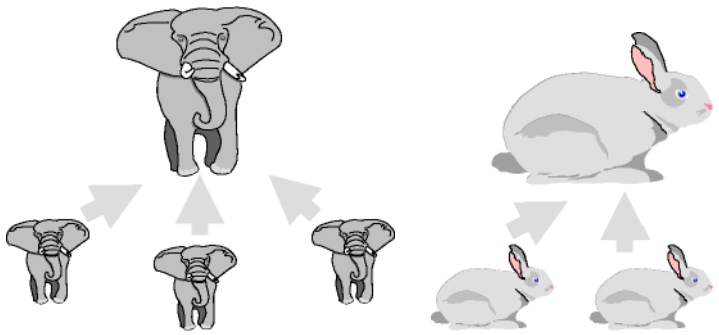


Ilustración 11-15. Ejemplo de herencia.

- Polimorfismo

  Diferentes objetos pueden responder a un mismo mensaje de diferentes maneras. El polimorfismo permite a los objetos interactuar entre ellos sin necesidad de conocer previamente a que tipo pertenecen.

- Encapsulamiento

  La única forma de acceder a los datos (estructura) de un objeto es a través de las operaciones.

  Permite ocultar los detalles de la implementación de un objeto.

  Los objetos ocultan la funcionalidad interna de sus operaciones, de otros objetos y del mundo exterior, pero brindan los mecanismos para accederla.

Ilustración 11-16. Esquema de encapsulamiento de un objeto.

Ventajas

- Facilita la re-utilización de la arquitectura y el código.
- Los modelos reflejan de manera más cercana el mundo real.
- Describe con mayor exactitud los procesos y datos incorporados.
- Descomposición basada en partición natural.
- Más "fácil" de entender y mantener.
- Estable.
- Un cambio pequeño en requerimientos no significa cambios masivos en el sistema en desarrollo.

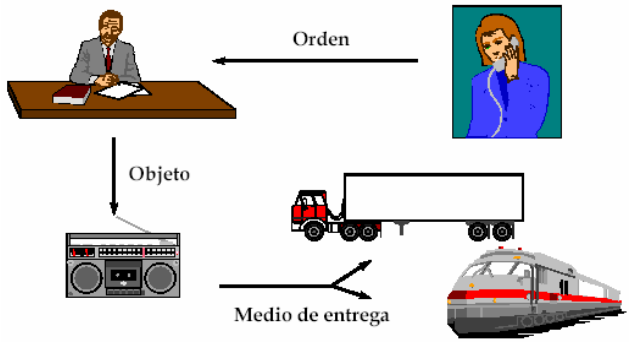


Ilustración 11-17. Ventajas en el uso orientado a objetos.

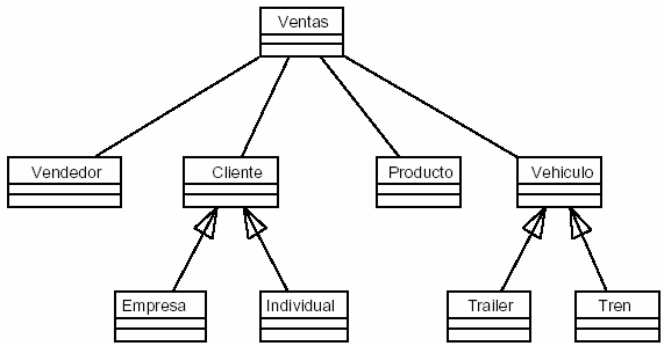


Ilustración 11-18. Ventajas en la organización orientada a objetos.

Aplicaciones

- Sistemas basados en GUI.

  La orientación a objetos facilita el diseño de sistemas con Interfaces Gráficas de Usuario (GUI).

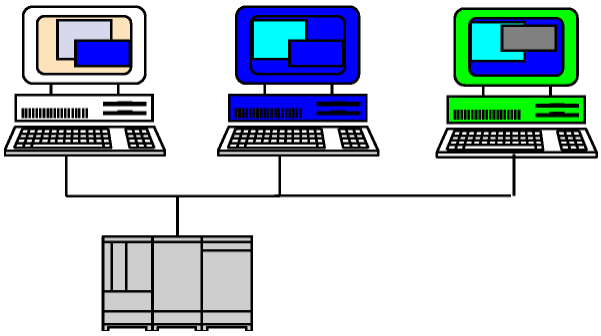


Ilustración 11-19. Simplificación de los sistemas basados en GUI.

- Sistemas inmersos.

  Los métodos orientados a objetos permiten desarrollar sistemas inmersos[1] y de tiempo real con mayor calidad y flexibilidad.

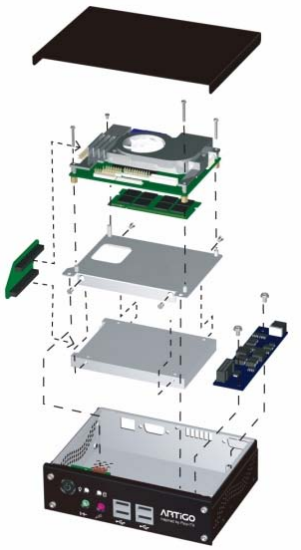


Ilustración 11-20. Sistema inmerso optimizado por los métodos orientados a objetos.

[1] Un sistema inmerso es un sistema de computación especial diseñado para llevar a cabo una o unas cuantas funciones dedicadamente, a menudo en tiempo real. Es normalmente parte de un dispositivo completo incluyendo hardware y piezas mecánicas. Dado que el sistema inmerso se dedica a tareas específicas, los ingenieros de diseño pueden optimizar, reducir el tamaño y el costo del producto, o aumentar la fiabilidad y el rendimiento.

- Comercio electrónico.

  Las aplicaciones de comercio electrónico requieren robustez y estabilidad que la tecnología orientada a objetos proporciona.

  Reutilización de las aplicaciones.

11.3 PROCESO DE DESARROLLO ITERATIVO E INCREMENTAL

¿Qué es un proceso?

Un proceso es un conjunto de actividades, métodos y prácticas para desarrollar y mantener el software y los productos asociados. Un proceso define qué hacer, cuándo y cómo hacerlo, y quién(es) deben hacerlo.

Ilustración 11-21. Proceso de una aplicación.

Rational Unified Process RUP es un proceso de desarrollo de software y junto con el Lenguaje Unificado de Modelado UML, constituye la metodología estándar más utilizada para el análisis, implementación y documentación de sistemas orientados a objetos. RUP es en realidad un refinamiento realizado por Rational Software del más genérico Proceso Unificado. El RUP no es un sistema con pasos firmemente establecidos, sino un conjunto de metodologías adaptables al contexto y necesidades de cada organización.

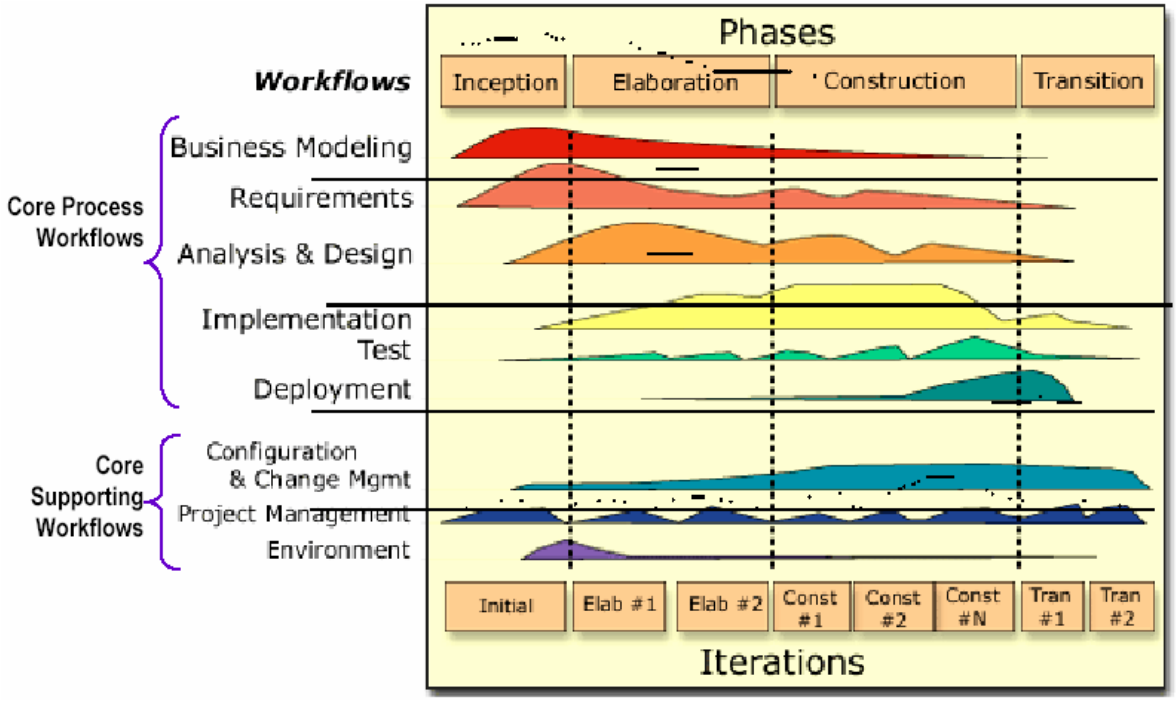


Ilustración 11-22. Uso de RUP en flujo de trabajo de un sistema.

11.4 ANÁLISIS Y DISEÑO ORIENTADO A OBJETOS CON UML

Análisis vs. diseño

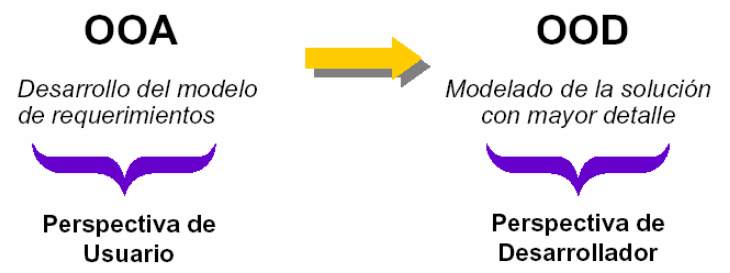


Ilustración 11-23. Pasar del análisis orientado a objetos al diseño orientado a objetos.

Análisis

- Determinar el qué se va a hacer.
- Obtener, analizar y especificar requerimientos.
- Modelar el sistema desde la perspectiva del usuario.
- "Problema".

Diseño

- Determinar el cómo se va a hacer (conceptualmente).
- Realizar especificaciones de software.
- Modelar la solución desde la perspectiva del desarrollador.
- "Solución".

UML

UML es el lenguaje de modelado estándar para crear planos de software. Permite:

- Visualizar la solución y facilitar la comunicación.
- Especificar modelos más precisos.
- Construir código base mediante herramientas como Rational Rose en varios lenguajes de programación.
- Documentar arquitectura, requerimientos, pruebas, procesos de negocio, entre otros.

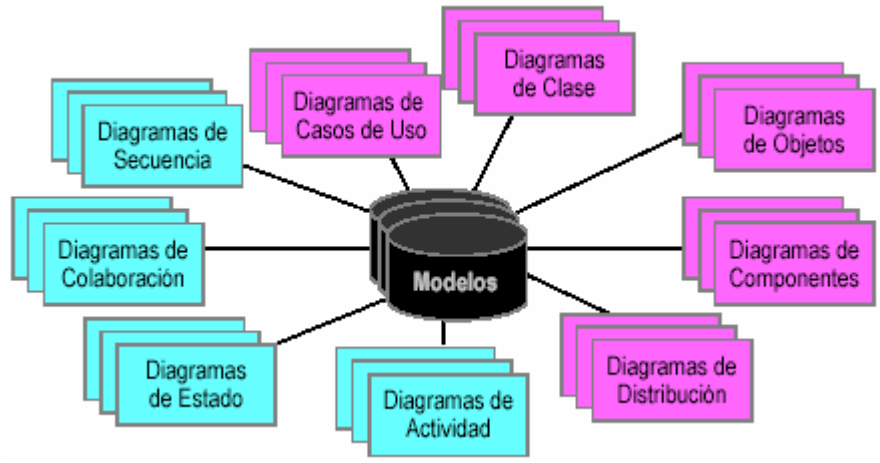


Ilustración 11-24. Diagramas disponibles con el UML.

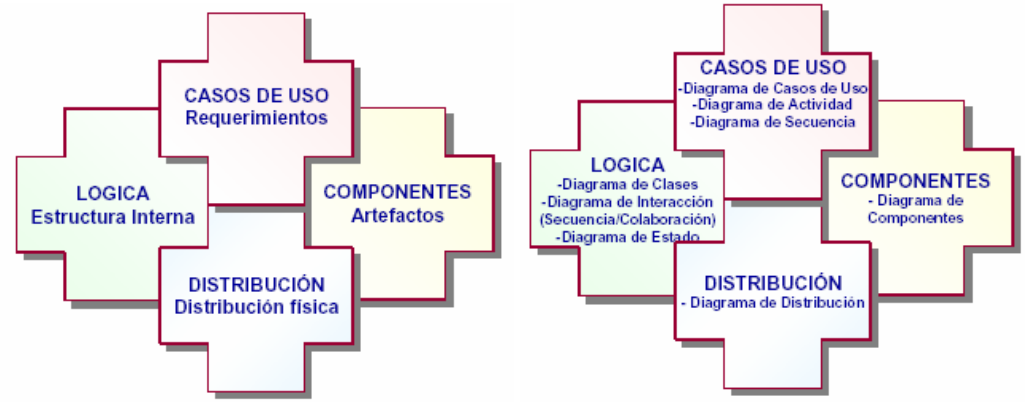


Ilustración 11-25. Perspectivas de una aplicación.

Casos de uso

- Especifican el comportamiento del sistema y sus requerimientos.
- El comportamiento del sistema se refiere a cómo actúa y reacciona el sistema: la actividad visible y comprobable.
- Los casos de uso describen al sistema, su ambiente, y las relaciones entre estos.
- Un caso de uso es una interacción típica entre un usuario (actor) y un sistema.

Requerimientos

Entendiendo como requerimiento como una condición o característica que se debe satisfacer se puede definir como sigue:

Ilustración 11-26. Posición de los requerimientos de software en el desarrollo de sistemas.

La vista de casos de uso captura el comportamiento de un sistema

- Un caso de uso representa por lo tanto una secuencia de acciones que un sistema lleva a cabo para ofrecer algún resultado de valor para un Actor.
- Los casos de uso representan toda la funcionalidad del sistema.

El modelo de casos de uso es una especificación completa de todas las formas posibles de utilizar un sistema entendido como requerimientos funcionales.

Fue un concepto aportado por Jacobson en 1994.

Elementos

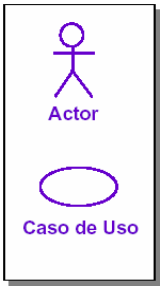


Ilustración 11-27. Representación de un actor y un caso de uso en UML.

Actor

- Un actor representa cualquier cosa que interactúe con el sistema.
- Los Actores no son parte del sistema; representan los roles que pueden jugar los usuarios del sistema.
- Un Actor puede intercambiar información activamente con el sistema.
- Un Actor puede ser un receptor pasivo de información. Un Actor puede ser una persona, una máquina u otro sistema.
- Cada Actor participa en uno o más Casos de Uso.

Actores - instancias

Ilustración 11-28. Representación de un rol de un actor y un caso de uso.

Un rol puede ser realizado por distintos usuarios.

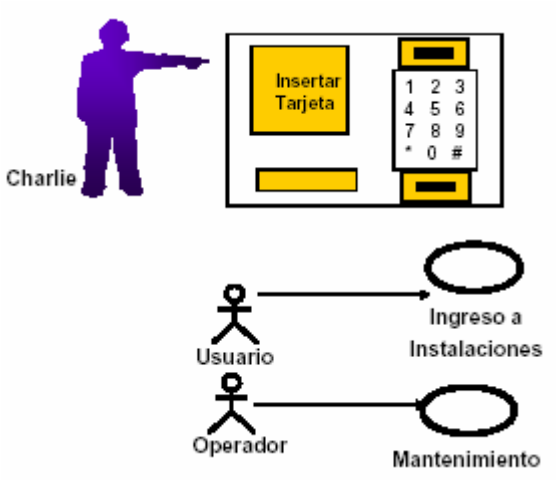


Ilustración 11-29. Representación de varios roles de un actor y un caso de uso.

Un usuario puede desempeñar varios roles

Identificar a los actores

- ¿Quién esta interesado en cierto requerimiento? ¿Qué papel desempeña?
- ¿Dónde se usará el sistema dentro de la organización? ¿En qué áreas?
- ¿Quién proveerá de información al sistema? ¿Quién consultará la información? ¿Quién borrará la información?
- ¿Quién usará determinada función? ¿Quién le dará soporte y mantenimiento al sistema?
- ¿El sistema usa una fuente externa?
- ¿Qué actores necesitan los casos de uso?
- ¿Puede un actor desempeñar varios roles diferentes?
- ¿Varios actores desempeñan el mismo rol?

Casos de uso

- Un Caso de Uso es una unidad coherente de funcionalidad.
- Un Caso de Uso especifica una secuencia de acciones, incluyendo variantes, que el sistema puede llevar a cabo, y que producen un resultado observable de valor para un actor.
- Es una toma instantánea de algún aspecto del sistema.
- Los Casos de Uso en conjunto representan toda la funcionalidad del sistema.

Fuentes

Fuentes de información para obtener los Casos de Uso

- Conversación con el cliente/usuario.

- Dominio del problema.
- Especificación del sistema.
- Literatura relevante del dominio.
- Entrevistas con expertos.
- Conocimiento personal del dominio.
- Sistemas legados.
- Experiencia de sistemas anteriores

Identificar los casos de uso

- ¿Cuáles son las tareas de este actor?
- ¿El actor creará, almacenará, cambiará, borrará o leerá información en el sistema?
- ¿Qué caso de uso creará, almacenará, cambiará, borrará o leerá determinada información?
- ¿El actor necesita recibir información acerca de ciertas ocurrencias en el sistema?
- ¿El sistema proporciona al negocio el comportamiento correcto?
- ¿Qué casos de uso van a darle soporte y mantenimiento al sistema?
- ¿Los requerimientos funcionales están cubiertos por los casos de uso? ¿Es todo lo que debe hacer el sistema?

Consideraciones

- Cada Caso de Uso representa una forma de usar el sistema (dan soporte a un usuario durante un proceso de negocio).
- Los mejores Caso de Uso son aquellos que añaden el mayor valor al negocio que implanta el sistema.

  Casos de Uso vs. "Casos de" Abuso.

  Casos de Uso vs. "Casos sin" Uso.
- ¿Cuántos Casos de Uso? Es necesario ajustar la granularidad de acuerdo al problema:

  Muchos Casos de Uso son abrumantes

  Pocos Casos de Uso pueden ocultar detalles importantes.

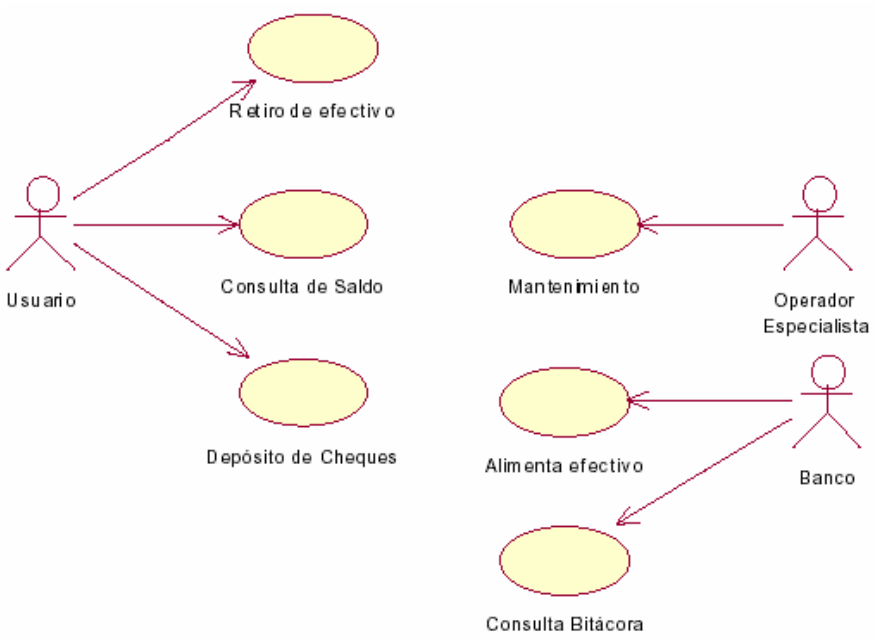


Ilustración 11-30. Ejemplo de casos de uso y actores.

Relaciones "Uses" y "Extends"

Las relaciones uses ocurren cuando se tiene una porción de comportamiento que es similar en más de un caso y no se quiere duplicar la descripción de tal conducta. **`Uses (include)`** permite incluir la misma funcionalidad en dos o más. Caso de Uso separados sin necesidad de repetir los detalles.

Se utiliza la relación extends cuando se tiene un Caso de Uso que es similar a otro, pero que hace un poco más. **`Extends`** describe una variación de la conducta normal.

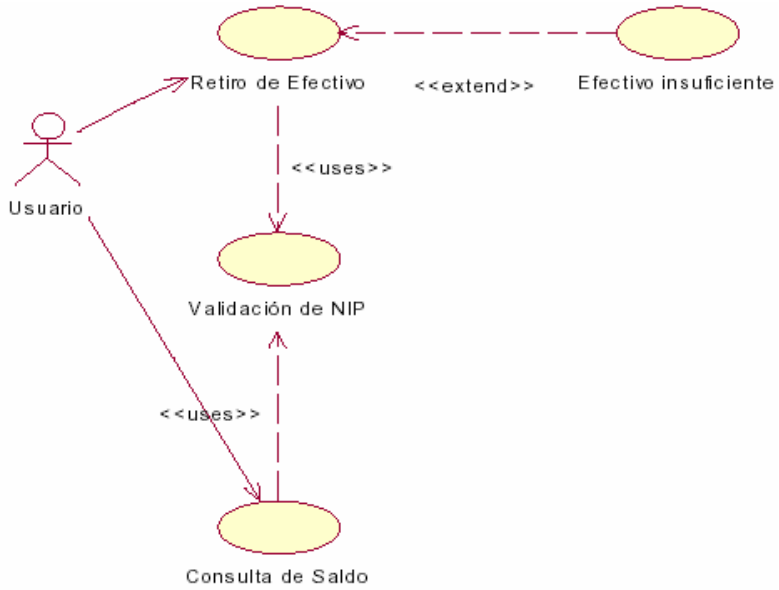


Ilustración 11-31. Uso de extend de UML.

Documentación de los Casos de Uso

Breve Descripción. El propósito es explicar en pocas líneas el Caso.

Pre-condiciones. Requisitos necesarios para que el Caso de Uso se lleve a cabo.

Flujo Principal. Secuencia de acciones que se dan normalmente.

Flujos Alternos. Secuencias de acciones que detallan las alternativas del Caso.

Flujos de Excepción. Eventos extraordinarios del Caso.

Post-condiciones. Requisitos que deben cumplirse posteriormente.

Describe sólo los eventos que pertenecen al caso de uso, y no lo que pasa en otros casos de uso.

Utilizar un lenguaje comprensible para el cliente.

Evitar terminología vaga como "por ejemplo", "etc." y "los datos".

Deberá describir:

- Cómo y cuándo inicia y termina el caso de uso.
- Cuando interactúa el caso de uso con los actores.
- Qué información se intercambia entre un actor y el caso de uso.
- No describe los detalles de la interfaz de usuario, describe las acciones.

Utilidades de los casos de uso

Clientes. Aprueban lo que el sistema debe hacer.

Usuarios. Ganan entendimiento del sistema.

Analistas. Proporciona las bases para el análisis y diseño.

Desarrolladores. Comportamiento del documento del sistema.

Probadores. Se usan como base para las pruebas del sistema.

Líder del proyecto. Planeación de proyectos.

Documentador. Base para la guía de usuario.

Diagramas de actividad

- Representan secuencia y acciones paralelas.
- El diagrama de actividad combina ideas de varias técnicas: el diagrama de eventos de John Odell, las técnicas de modelado de estados de SDL y las redes de Petri.
- Son útiles para la descripción del comportamiento que tiene una gran cantidad de procesos paralelos.

- Permite modelar los procesos reales de una organización humana y modelar actividades de software.

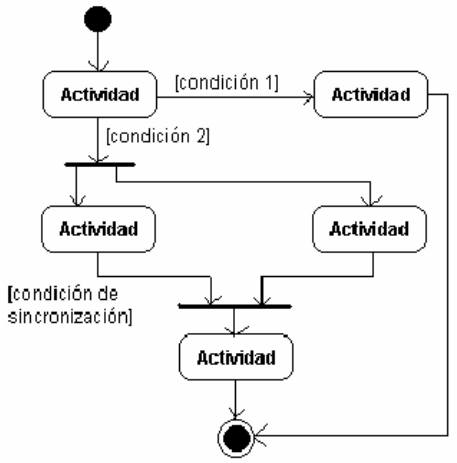


Ilustración 11-32. Diagramas de actividad en UML.

Notación

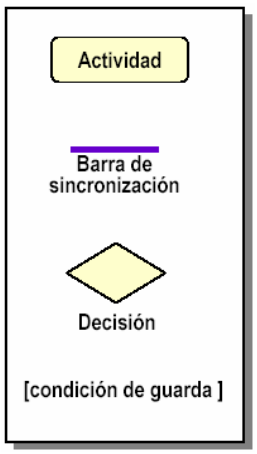


Ilustración 11-33. Representación de actividad, barra de sincronización, decisión y condiciones.

Una actividad es una acción a realizar.

La barra de sincronización permite iniciar acciones una vez que se han realizado actividades concurrentes.

La decisión es un punto en el que se pueden seguir alternativas distintas de acuerdo al resultado de la actividad anterior.

Las condiciones de guarda son los posibles resultados de una acción que servirán como condición para la realización de otra.

Swimlanes

Los diagramas de actividad mediante los "**`swimlanes`**" muestra además de qué pasa, quién lo hace. Se indican con una línea vertical que separa el diagrama en zonas. Cada zona representa una clase, persona o departamento en particular.

Utilidad: Desde la perspectiva de la implementación, permite representar qué clase es responsable de qué actividad. Desde el punto de vista de dominio, permite representar qué persona o departamento es responsable de qué actividad.

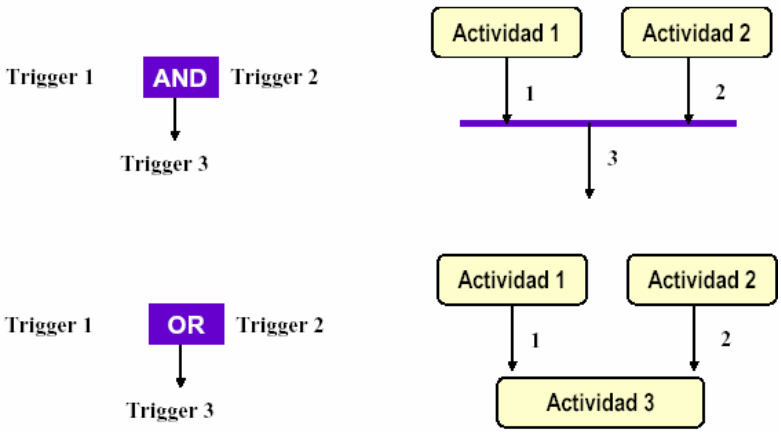


Ilustración 11-34. Diagramas de actividad AND/OR en UML.

Diagrama de Clases

- Representan las características estructurales.
- El diagrama de clases muestra las clases que hay en el sistema y las relaciones estáticas (asociación y agregación) entre ellas.
- Se consideran la columna vertebral de los métodos orientados a objetos.

¿Qué es una clase?

Una clase es una descripción de un grupo de objetos con propiedades comunes (atributos), comportamiento común (operaciones) y relaciones comunes con otros objetos (asociaciones y agregaciones).

- En análisis, una entidad conceptual del dominio del problema.
- En diseño, una entidad detallada que representa una parte de la solución.
- En código, es un bloque de código.

Perspectivas

Una clase es una descripción de un grupo de objetos con los diagramas de clases se pueden construir desde tres perspectivas:

- Conceptual. Representa los conceptos del dominio que se está estudiando. Se dibujan sin importar el software con que se implementarán por lo que son independientes del lenguaje. Pueden ubicarse en el contexto del negocio (modelado de negocio) o del sistema.
- Especificación. Enfocados al software en su filosofía pero no en su implementación.
- Implementación. Se expone por completo la implementación.

Identificar las clases en un problema

Las clases surgen de la terminología de un área de conocimiento. Hay que prestar atención a:

- Los sustantivos (cosas, personas, hechos) ya que pueden convertirse en clases del modelo.
- Verbos que pueden ser operaciones de las clases o en clases por sí mismas.
- Los atributos de una clase también se pueden identificar por sustantivos específicos que pueden tomar algún valor.

Designar un nombre a las clases

El nombre de una clase debe ser un nombre singular que caracterice de la mejor forma a la abstracción.

- La dificultad en el nombramiento de una clase puede indicar que una abstracción está pobremente definida.
- Los nombres deben venir directamente del vocabulario del dominio.

Una guía de estilo debe dictar convenciones de nombres para clases. Ejemplo:

- Las clases se nombran usando sustantivos singulares.
- Los nombres de clases empiezan con una letra minúscula.
- No se usan palabras subrayadas. Los nombres compuestos de palabras múltiples se ponen juntos y la primera letra de cada palabra adicional se escribe en mayúscula. Ejemplo: **`alumno`**, **`profesor`**, **`sistemaCobro`**.

Notación

Una clase se comprende de tres secciones

- La primera sección contiene el nombre de la clase.
- La segunda sección muestra la estructura (atributos).
- La tercera sección muestra el comportamiento (operaciones).
- Las secciones segunda y tercera pueden suprimirse si no es necesario que sean visibles en el diagrama.

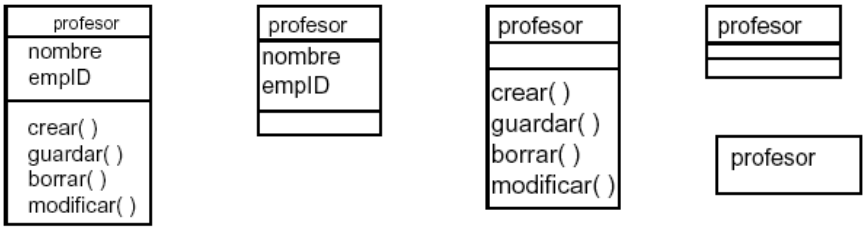


Ilustración 11-35. Secciones de una clase.

Una clase es una descripción de un conjunto de objetos que comparten los mismos atributos, relaciones y semántica.

Un atributo es una propiedad de una clase identificada con un nombre, que describe un rango de valores que pueden tomar las instancias de la propiedad.

- Una abstracción de un tipo de dato o estado que puede incluir un objeto de la clase.

Una operación es la implementación de un servicio que puede ser requerido a cualquier objeto de la clase para que muestre un comportamiento.

- Es una abstracción de algo que se puede hacer a un objeto y que es compartido por todos los objetos de la clase.

Ilustración 11-36. Ejemplificación de un diagrama de clase.

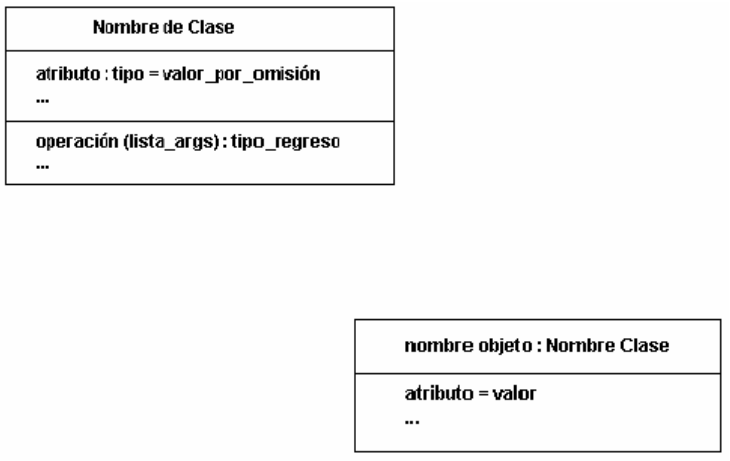


Ilustración 11-37. Notación de los diagramas de clase.

Estereotipos

Un estereotipo es un elemento que extiende las características. Cada clase puede tener como máximo un estereotipo.

Estereotipos comunes:

- **`Entity`**.
- **`Boundary`**.
- **`Control`**.

Los Estereotipos se muestran en la parte donde se escribe el nombre de la clase entre **`<< >>`**. Se pueden definir los estereotipos según convenga. Otros ejemplos: **`Table`**, **`Page`**, **`HTML`**.

Entity

Una clase **`entity`** (entidad) modela información y comportamiento asociado que es generalmente de larga vida (persistente). Son de alto nivel (perspectiva conceptual) pero también puede necesitarse para las tareas internas del sistema ya que corresponden a las estructuras de datos (mapeo a BD). Representan los objetos de negocio (análisis).

Ejemplos de clases **`entity`**:

- **`Curso`**.
- **`Horario`**.
- **`Catálogo de asignaturas`**.
- **`Producto`**.
- **`Cliente`**.

Boundary

Las clases boundary se utilizan para modelar la interacción entre el sistema y sus actores. Esta interacción a menudo implica recibir (y presentar) información y peticiones de (y hacia) los usuarios y los sistemas externos.

Representan a menudo abstracciones de ventanas, formularios, paneles, interfaces de comunicaciones, impresoras, sensores, terminales.

Ejemplos:

- **`frmInscripcion`**
- **`formaDatosPersonales`**

Control

Las clases de control representan coordinación, secuencia, transacciones y control de otros objetos y se usan con frecuencia para encapsular el control de un caso de uso en concreto.

Las clases de control también se utilizan para representar derivaciones y cálculos completos, como lógica del negocio que no pueden asociarse con ninguna información concreta, de larga duración, almacenada por el sistema.

Relaciones

Todos los sistemas contienen varias clases y objetos.

Los objetos contribuyen al comportamiento de sistema colaborando con unos y otros. La colaboración se realiza a través de las relaciones.

La existencia de una relación entre dos clases denota una vía de comunicación (enlace) entre instancias de las clases, por las que un objeto puede enviar mensajes a otro.

Asociación. Nombre, rol, multiplicidad.

Agregación (pertenencia). Relación de tipo tiene-un.

Generalización (herencia). Relación de tipo es-un.

Dependencia.

Asociación

Una asociación es una conexión semántica bi-direccional entre clases. Esto implica que hay una liga entre objetos en las clases asociadas.

Las asociaciones se representan en diagramas de clase por una línea que conectada a las clases asociadas.

La información puede fluir en cualquier dirección o en ambas direcciones a través de la liga.

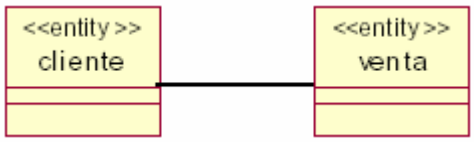


Ilustración 11-38. Relación entre dos clases.

Para clarificar su significado, se puede nombrar una asociación. El nombre se representa como una etiqueta que se pone a lo largo de la línea de asociación, a mitad de camino entre los iconos de clases. Un nombre de asociación es usualmente un verbo o una frase con verbo.

Ilustración 11-39. Nombramiento de una relación entre dos clases.

Un rol denota el propósito o capacidad en la que una clase se asocia con otra.

Los nombres de roles son típicamente sustantivos o frases con sustantivo.

Un nombre de rol se pone a lo largo de la línea de asociación cerca de la clase que modifica. Uno o ambos finales de una asociación pueden tener nombres de roles.

Ilustración 11-40. Roles en una asociación.

Multiplicidad

Multiplicidad es el número de instancias de una clase relacionada a una instancia de la otra clase.

Para cada asociación, hay dos decisiones de multiplicidad que tomar: una por cada final de la asociación.

- Por ejemplo, en la conexión entre "**`Persona`**" jugando el rol de "**`Profesor`**" con "**`Curso`**".
- Para cada instancia de **`Persona`**, varios (i.e., cero o más) **`Cursos`** deben impartirse.
- Para cada instancia **`Curso`**, exactamente una "**`Persona`**" es el **`Profesor`**.

Cada final de una asociación contiene una multiplicidad de indicadores

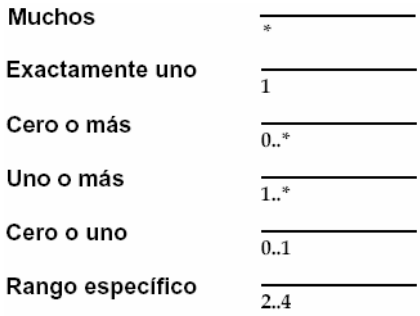


Ilustración 11-41. Indicador del número de objetos que participan en la relación.

La multiplicidad expone varias hipótesis ocultas acerca del problema que se está modelando.

- ¿Puede estar un profesor en sabático? ¿Puede tener un curso dos profesores?

Ilustración 11-42. Multiplicidad en una asociación.

La multiplicidad responde a dos preguntas.

- ¿La asociación es obligatoria u opcional? ¿Cuál es el número mínimo y máximo de instancias que pueden ligarse a una instancia?

Ilustración 11-43. Asociación opcional.

Agregación

La agregación es una forma especializada de asociación en la que un todo se relaciona con su parte o sus partes. La agregación es conocida como "parte de" o relación que contiene.

Una agregación se representa como una asociación con un diamante al lado de la clase denotando el agregado (todo).

La multiplicidad se representa de la misma manera que otras asociaciones.

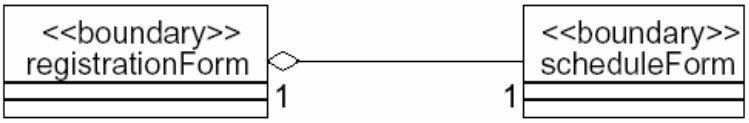


Ilustración 11-44. Esquema de la agregación.

¿Se usa la frase "parte de" para describir relaciones? Una Puerta es "parte de" un Carro.

¿Se aplican algunas operaciones en el todo automáticamente a sus partes? Mover el Carro, Mover la Puerta.

¿Se propagan algunos valores de atributos del todo a todas o algunas de sus partes? El Carro es azul, La Puerta es azul.

¿Hay una asimetría intrínseca a la relación donde una clase se subordina a la otra? Una Puerta Es parte de un Carro, un Carro No Es parte de una Puerta.

Clase asociación

Si quisiéramos rastrear los grados para todos los cursos que un alumno ha tomado. La relación entre **`Alumno`** y **`Curso`** es una relación de muchos-a-muchos. ¿Dónde se podría el atributo de calificación?

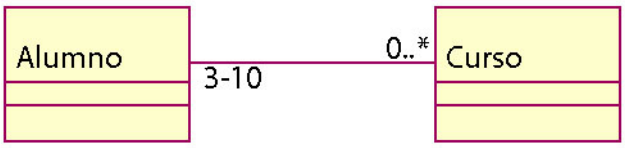


Ilustración 11-45. Creación de una clase de asociación.

Crear una clase de asociación usando el icono clase. Conectar el icono clase a la línea de asociación usando una línea punteada. La clase de asociación puede incluir múltiples propiedades de la asociación. Sólo se permite una clase de asociación por cada asociación.

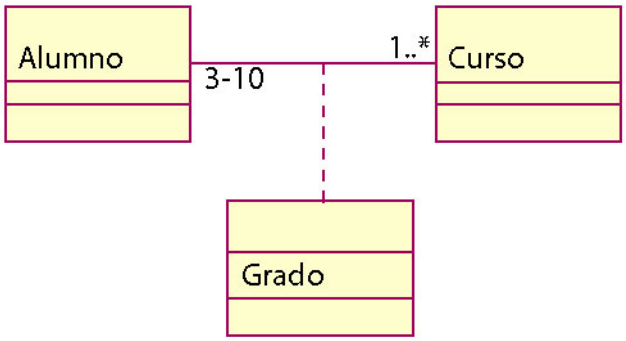


Ilustración 11-46. Conexión a la línea de asociación.

Generalización

La generalización permite conectar clases generales con otras más especializadas. Se conocen como **`subclase/superclase`** o **`hijo/padre`**. Es una relación de "**`es-un-tipo-de`**". Una clase hija hereda las propiedades de sus clases padres, especialmente sus atributos y operaciones.

Ilustración 11-47. Conector de generalización.

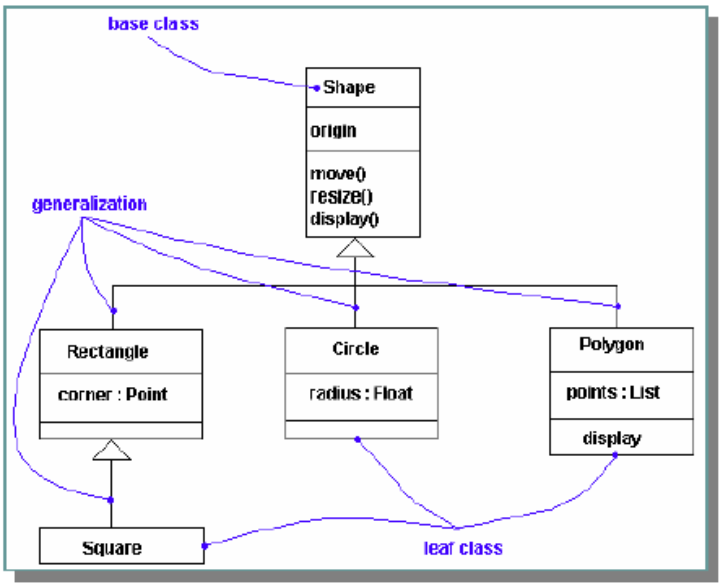


Ilustración 11-48. Diagrama UML usando conectores de generalización.

Dependencia

Las dependencias son relaciones de uso. Indican que un cambio en la especificación de un elemento puede afectar a otro elemento que la utiliza. Se representa con una línea discontinua dirigida hacia el elemento del cual se depende. Se utilizan generalmente en el contexto de las clases o entre paquetes.

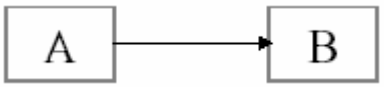


Ilustración 11-49. Esquema de dependencias de clases.

Paquetes

Un paquete es un mecanismo de propósito general para organizar elementos en grupos. Se visualizan como carpetas.

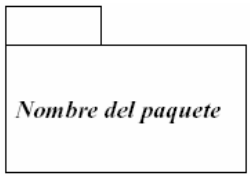


Ilustración 11-50. Notación de paquetes en UML.

Relaciones entre paquetes

Los paquetes se relacionan unos a otros usando una relación de dependencia.

Si una clase en un paquete "habla" con una clase en otro paquete entonces se agrega una relación de dependencia en el nivel de paquete.

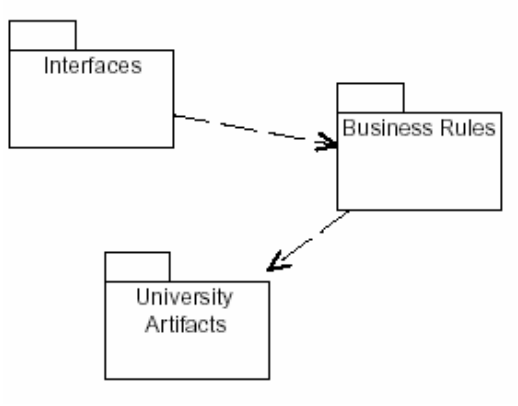


Ilustración 11-51. Relación de dependencia a nivel de paquete.

Visibilidad

`Public`. Accesible a todas las clases.

`Protected`. Accesible sólo a las subclases y a la clase misma.

`Private`. Accesible sólo a la clase.

`Implementation`. Accesible únicamente por la implementación del paquete.

Evolución en análisis y diseño

Durante el análisis:

- Se establecen clases (principalmente **`entity`**) y conexiones básicas (asociaciones, agregaciones y generalizaciones). Estas conexiones existen debido a la naturaleza de las clases, y no debido a una implementación específica.
- Se hace una estimación inicial de multiplicidad para exponer hipótesis ocultas

Durante el diseño:

- Se consideran otros estereotipos de clase (**`boundary`** y **`control`**).
- Se refinan y actualizan las estimaciones de multiplicidad.
- Se evalúan y refinan las relaciones.

- Se detallan los tipos y firmas (**`signature`**).

Diagramas de interacción

Un diagrama de interacción es una representación gráfica de interacciones entre objetos Hay dos tipos de diagramas de interacción:

- Diagramas de secuencia.
- Diagramas de colaboración.

Cada uno provee un punto de vista diferente de la misma interacción.

- Los diagramas de secuencia están ordenados de acuerdo al tiempo.
- Los diagramas de colaboración pueden incluir flujo de datos.

Diagrama de secuencia

Un diagrama de secuencia muestra interacciones de objetos ordenados en secuencia de tiempo El diagrama muestra:

- Los objetos que participan en la interacción.
- La secuencia de mensajes intercambiados.

Un diagrama de secuencia contiene:

- Objetos con sus "líneas de vida".
- Mensajes intercambiados entre objetos en orden secuencial.
- Enfoque de control (opcional).

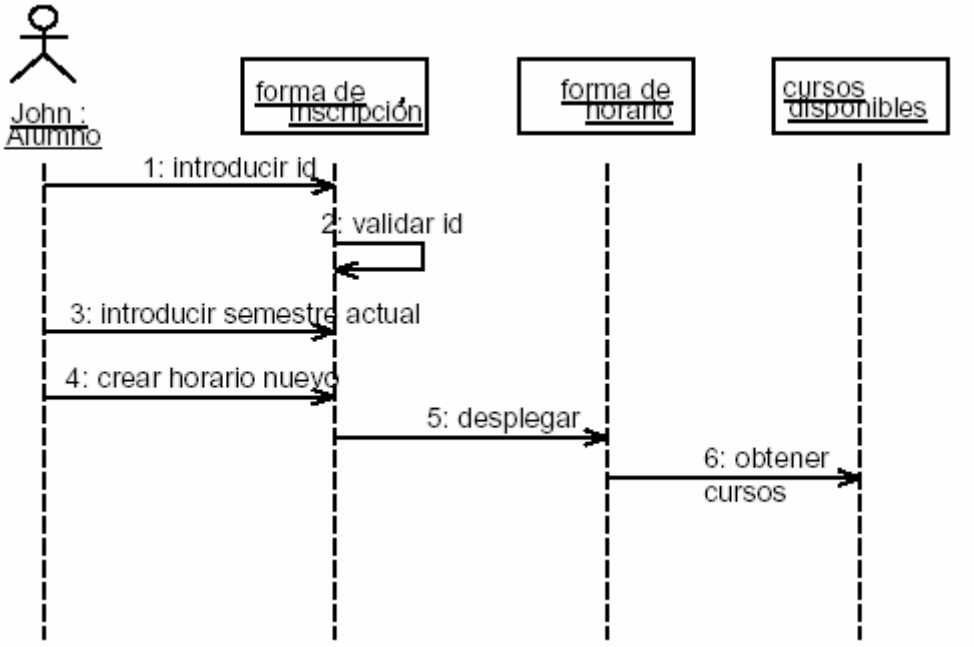


Ilustración 11-52. Diagrama de secuencia.

Diagrama de colaboración

Un diagrama de colaboración es una forma alternativa de representar los mensajes intercambiados por un conjunto de objetos. El diagrama muestra interacciones de objeto organizadas alrededor de los objetos y sus ligas a cada uno.

Un diagrama de colaboración contiene:

- Objetos.
- Ligas entre objetos.
- Mensajes intercambiados entre objetos.
- Flujo de datos entre objetos, si hay alguno.

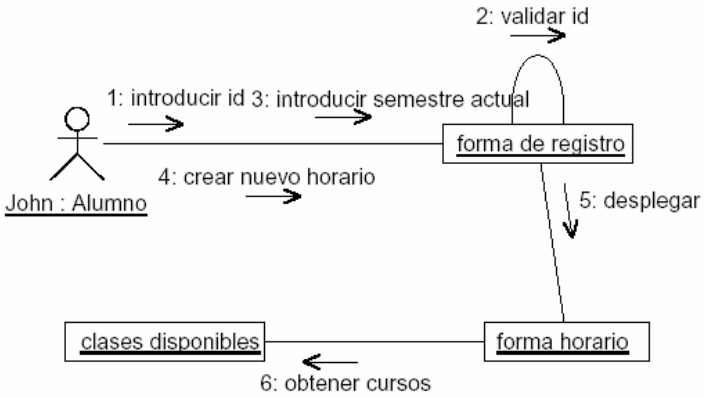


Ilustración 11-53. Diagrama de colaboración.

Diagrama de estado

Los diagramas de clase representan la estructura y el comportamiento estático de los objetos.

Los diagramas de estado se usan para mostrar la historia de vida de una clase dada, los eventos que causan una transición de un estado a otro, y las acciones que resultan de un cambio de estado.

El estado de un objeto es una de las condiciones posibles en las que puede existir un objeto.

Notación

Ilustración 11-54. Representación de estados.

Un estado es una de las condiciones posibles en las que puede existir un objeto.

Un evento es la ocurrencia de alguna situación que sucede en un punto del tiempo.

Una transición es un cambio de un estado original a un estado sucesor como resultado de algunos estímulos.

Una acción es una operación que se asocia a una transición.

Una actividad es una operación que toma tiempo para completarse. Las actividades se asocian con un estado.

Una condición de guarda es una expresión booleana de valores de atributos que permiten una transición solo si la condición es verdadera.

Estados

Los estados pueden distinguirse por los valores de ciertos atributos.

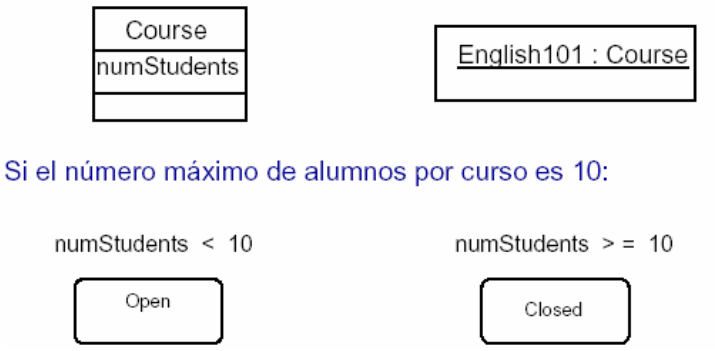


Ilustración 11-55. Representación de los estados.

Los estados también pueden identificarse por la existencia de ciertas ligas.

Las instancias de la clase Profesor puede tener dos estados:

- Impartir cuando existe una liga a un curso.
- En sabático cuando no existe liga.

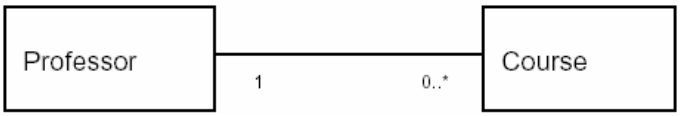


Ilustración 11-56. Instancias de una clase con varios estados.

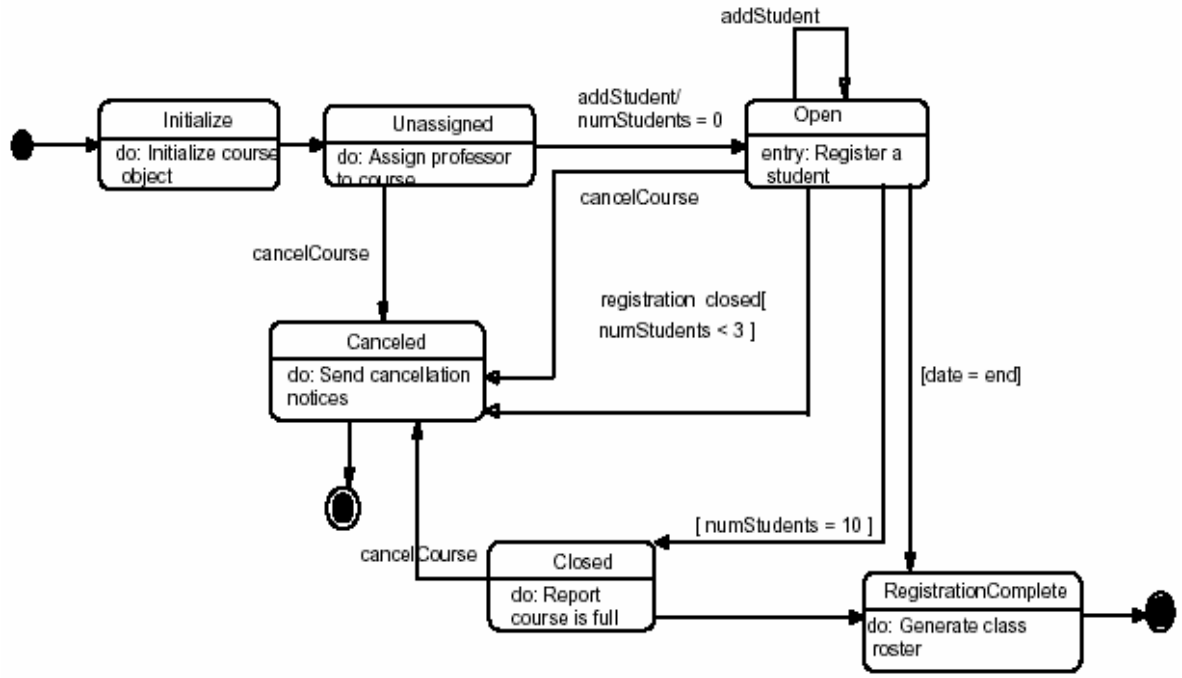


Ilustración 11-57. Diagrama de estados de un aplicación para el registro de alumnos y profesores a cursos.

Diagrama de componentes

Un componente es una unidad de código fuente que sirve como bloque constructor para la estructura física de un sistema. Un componente pertenece al mundo material de los bits. Los estereotipos (con iconos alternos) pueden usarse para definir tipos de componentes específicos: ejemplos pueden ser ejecutables EXE, bibliotecas DLL, programas principales, headers, módulos, formas, tablas, archivos y documentos.

Notación

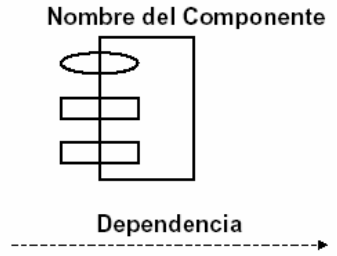


Ilustración 11-58. Notación de los componentes.

Diagrama de distribución

Los diagramas de distribución son creados para mostrar los diferentes nodos (procesadores y dispositivos) en el sistema. Los nodos, al igual que los componentes, pertenecen al mundo material y modelan el aspecto físico de un sistema. Los nodos modelan la topología del hardware sobre el que se ejecuta el sistema. Los elementos esenciales de un diagrama de distribución son los nodos y sus conexiones.

Notación

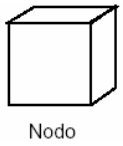


Ilustración 11-59. Representación de nodo en UML.

Un **nodo** es un objeto físico en tiempo de ejecución que representa recursos de cómputo.

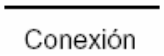


Ilustración 11-60. Representación de una conexión en UML.

Una **conexión** indica comunicación, usualmente la relación directa entre hardware.

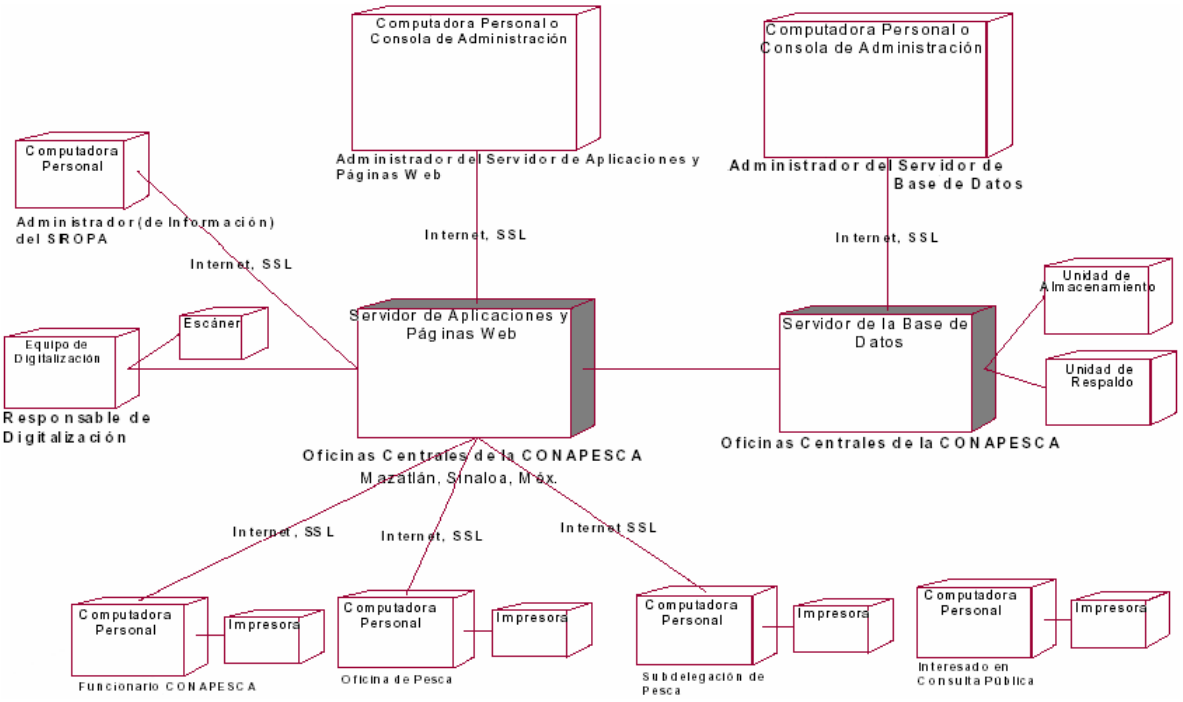


Ilustración 11-61. Diagrama de distribución.

11.5 ARQUITECTURA

Perspectivas diferentes para perfiles diferentes.

- Usuario final, cliente, administrador de proyecto.
- Ingeniero de sistema, desarrollador, arquitecto, evaluador.

Las perspectivas múltiples requieren múltiples vistas.

- Los diagramas de clase no muestran los mapas del sistema al hardware.
- Los diagramas de distribución no describen la estructura del sistema

4+1 Vistas

4 Vistas

Para describir completamente una arquitectura, el UML propone 4 vistas:

- La vista lógica para proporcionar una imagen estática de las clases primarias y sus relaciones.
- La vista de componente para mostrar como está el código organizado en paquetes y librerías.
- La vista de proceso para mostrar los procesos y tareas.
- La vista de distribución para mostrar los procesadores, dispositivos y ligas en el ambiente operacional.

+1 Vistas

- Finalmente, una vista de escenario explica como trabajan juntas las otras cuatro vistas.

Ilustración 11-62. Esquema de las 4 + 1 vistas.

Arquitectura multinivel

La arquitectura es la estructura organizativa de un sistema, que incluye su descomposición en partes, conectividad, mecanismos de interacción y principios de guía que proporcionan información sobre el diseño del mismo.

La arquitectura es el espacio en donde trabajan los objetos

- ¿Cómo reparto mi sistema?
- ¿Qué herramientas se ajustan mejor?

Actualmente se definen dos estilos principales de arquitectura para las aplicaciones distribuidas:

- Arquitectura de 2 capas (Cliente/Servidor).
- Arquitectura de 3 capas (n capas).

Arquitectura Cliente/Servidor

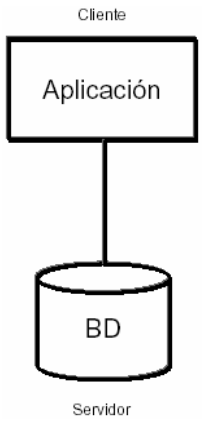


Ilustración 11-63. Esquema de la relación Cliente / Servidor.

Puede haber mucha carga en el cliente. Mucho tráfico en la red. Mantenimiento costoso, en cada cliente. Posibilidad de clientes desfasados.

Preguntas que debe contestarse al aplicar la arquitectura Cliente/Servidor ¿Se usa una sola base de datos? ¿La base de datos está situada en un único host? ¿El tamaño de la BD será similar con el paso del tiempo? ¿La cantidad de usuarios es similar a los largo del tiempo? ¿Los requisitos son fijos o con muy poca posibilidad de cambiar? ¿Se espera un mantenimiento mínimo de la aplicación?

Arquitectura de 3 capas

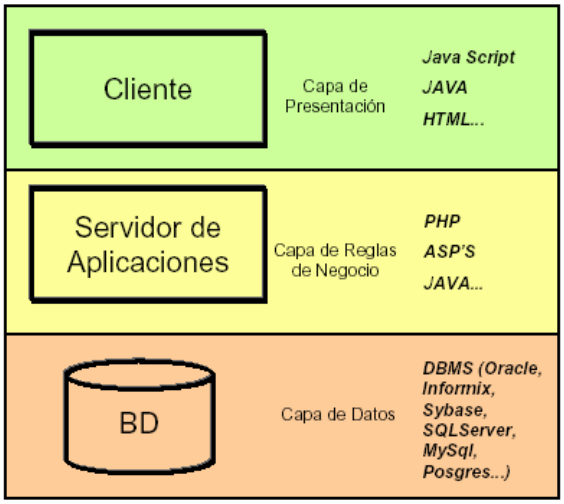


Ilustración 11-64. Esquema de la arquitectura de 3 capas.

Una arquitectura de 3 capas:

- Maximiza la reutilización de objetos.
- Facilita el mantenimiento.
- Distribuye el procesamiento.

El principal inconveniente es el nivel de complejidad que da al sistema.

Capa de datos. La capa de datos comprende lo relativo al almacenamiento lógico y físico de la información, es decir, las estructuras de datos así como los mecanismos de acceso, recuperación, procesamiento, seguridad y administración pura de la información (por ejemplo: base de datos relacional centralizada y gestionada por un Manejador de Bases de Datos X).

Capa de reglas de negocio. La capa de reglas de negocio tiene como objetivo especificar de manera formal los requerimientos funcionales propios del sistema, mediante la construcción de los componentes en alguna herramienta de desarrollo. Es intermediaria entre la capa de datos y la capa de presentación.

Capa de presentación. La capa de presentación es la que permite la interacción del usuario con el sistema; comprende por lo tanto las interfaces gráficas, la obtención y la presentación de información estática y dinámica al usuario, así como el procesamiento de la información a nivel del cliente, principalmente validaciones. Bajo esta arquitectura de tres capas en Web, si bien en la capa de

presentación se realiza cierto procesamiento que permite agilizar y mejorar el desempeño del sistema, requiere recursos mínimos en la máquina cliente.

11.6 TEORÍA DEL DISEÑO DE BASES DE DATOS

La teoría del diseño de bases de datos es una de las áreas de cómputo que más lentamente ha cambiado.

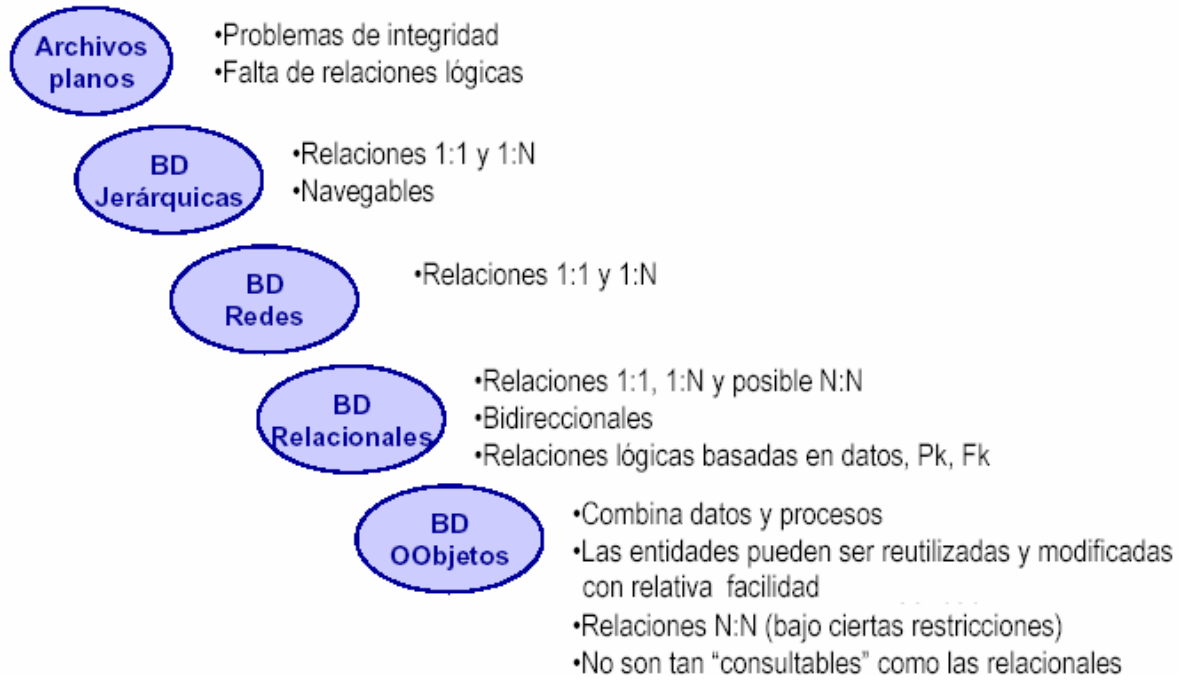


Ilustración 11-65. Evolución de las bases de datos.

El modelado de datos es el arte de identificar las entidades y sus relaciones que deben ser representadas en una base de datos.

Modelo relacional vs. modelo de objetos

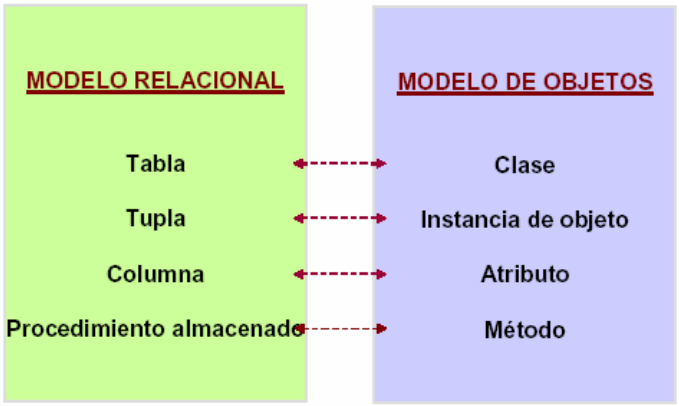


Ilustración 11-66. Comparación del modelo relacional vs. modelo de objetos.

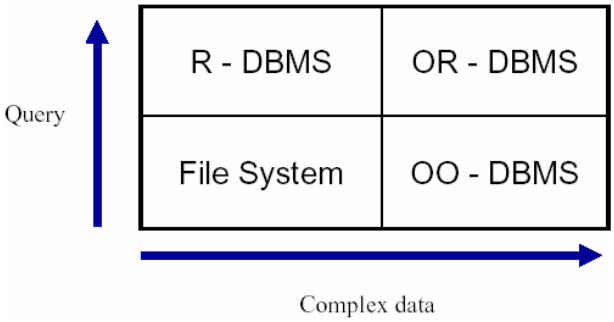


Ilustración 11-67. Aplicación de consultas vs. datos complejos.

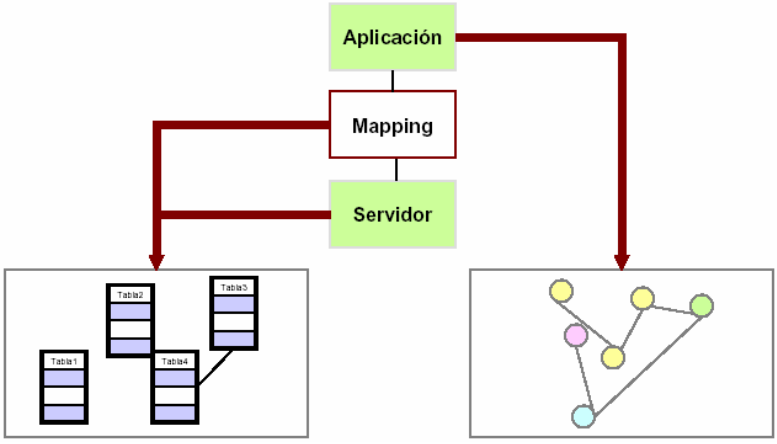


Ilustración 11-68. Mapeo de una base de datos en tablas a una salida de objetos en una aplicación.

Objetos "Entity"

Los objetos "**entity**" usados por los programas orientado a objetos son directamente análogos a las entidades de la base de datos orientado a objetos. Los objetos definidos en un programa desaparecen una vez que el programa termina mientras que los objetos de una base de datos se conservan (son persistentes).

Modelo objeto relacional

Bases de datos objeto relacionales

Son bases de datos relacionales pero soportan algunas características de las orientadas a objetos, por ejemplo:

- Tipos de objetos.
- Colecciones.
  - **VARRAYS**.
  - **Nested Tables.**
- **LOBs**.

- **BLOBs** (Binary Large Object).
- **CLOB** (Character Large Object).

Entre las bases de datos objeto relacionales destacan:

- Oracle 8, 9.
- Informix.
- DB2.
- Postgress.

Object-relational mapping

Entre el software de mapeo de bases de datos objeto relacionales destacan:

- Hibernate.
- JDO Engine.
- DADO Database Mapper.
- PowerTier.
- Castor.
- ObjectRelationalBridge
- OR Mapper
- SourcePro DB
- PowerDesigner

Modelo orientado a objetos

Bases de datos orientados a objetos

Permiten el almacenamiento de estructuras datos complejos que no pueden ser almacenadas fácilmente en bases de datos convencionales. No sólo almacenan tablas, columnas y renglones. Soportan toda la persistencia necesaria cuando se trabaja con lenguajes orientados a objetos. Fueron diseñadas para ser integradas de forma transparente con la programación orientada a objetos. Sistema administrador de bases de datos que soporta el modelado y creación de datos como objetos.

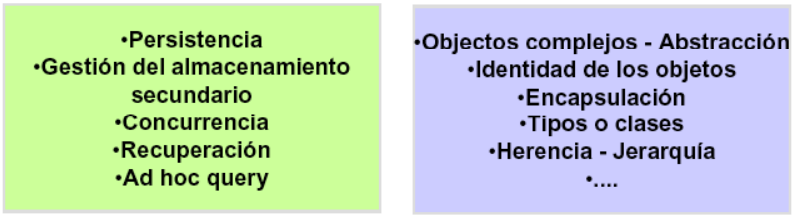


Ilustración 11-69. Ventajas de las bases de datos orientadas a objetos.

Persistencia. Conservación de los datos; permanencia.

Concurrencia. Interacción de muchos usuarios.

Recuperación. En caso de fallas de HW o SW, retroceder a un estado coherente de los datos.

Almacenamiento. Gestión del almacenamiento secundario. Mecanismos no visibles al usuario para mantener la independencia lógica y física del sistema.

Consultas. Facilidad de consultas

Características de interés para el modelado

OID. Existencia de un identificador de objetos. No se requieren llaves primarias explícitas en algunos casos.

Colecciones. Posibilidad de almacenar varios objetos y valores en un mismo atributo, mediante "SET" (conjuntos).

Navegabilidad.

Una base de datos relacional representa las relaciones mediante valores (PKs - FKs)	Una base de datos OO incluye identificadores de objetos dentro de un objeto indicando otros objetos al cual está relacionado

Ilustración 11-70. Diferencia entre las relaciones de las RDBMS vs. OODBMS.

Un OID es un identificador interno para cada objeto individual, los cuales nunca son manipulados por el "**`usuario`**". El OID es independiente de cualquiera de sus características. Los OID's son asignados y usados sólo por el DBMS.

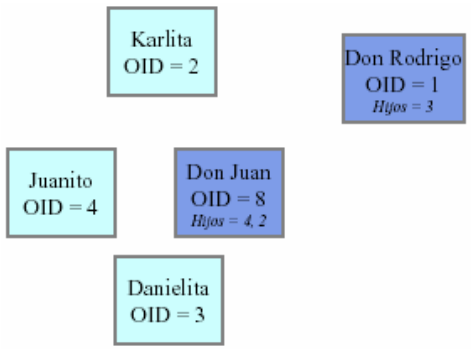


Ilustración 11-71. Ejemplos de OID's.

Relaciones 1:N

En contraste con el modelo relacional, el orientado a objetos permite almacenar valores múltiples. Esta característica es básica para cualquier tipo de relaciones "N".

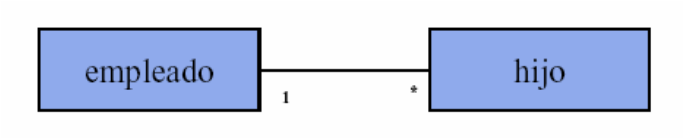


Ilustración 11-72. Relaciones 1:N en OODBMS.

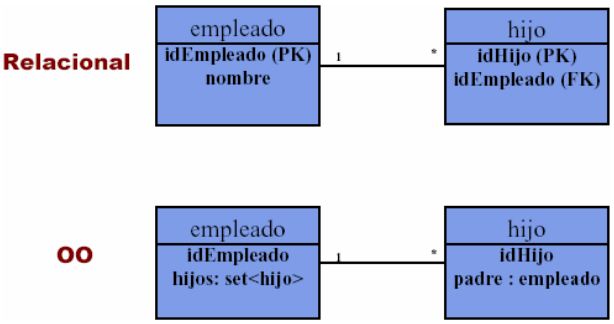


Ilustración. Comparación de las relaciones 1:N entre el modelo relacional y el orientado a objetos.

Relaciones N:N

Un OODB permite que los objetos tengan atributos de conjuntos de objetos por lo que las relaciones N:N pueden ser representadas directamente sin necesidad de entidades transitivas (aplica restricciones).

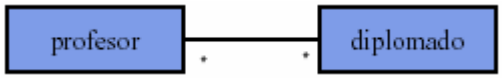


Ilustración 11-73. Relaciones N:N en OODBMS.

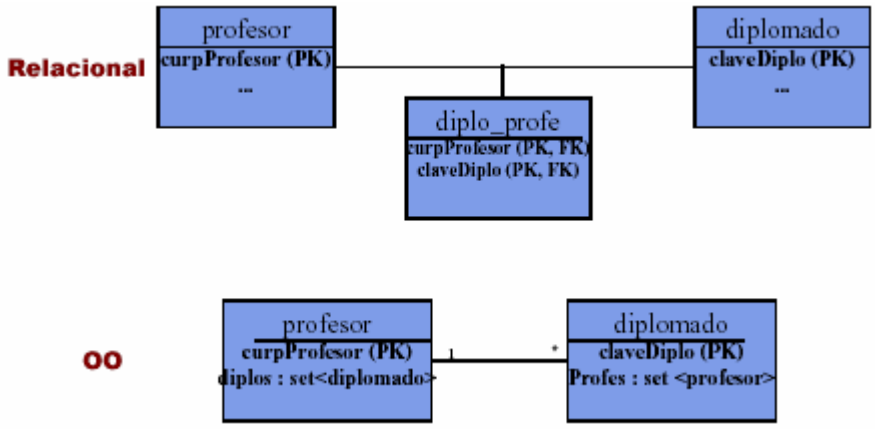


Ilustración 11-74. Comparación de las relaciones N:N entre el modelo relacional y el orientado a objetos.

- ¿Indicar o no la navegabilidad?
- Se utiliza la navegabilidad en una relación no inversa entre dos objetos, donde solamente uno de los objetos contiene el OID del objeto relacionado.

Integridad de relaciones

Se debe asegurar de alguna forma que los OID's referenciados en un objeto coincidan con su contraparte.

Al definir empleado:

```
hijos : set <hijo>

        inverse is  hijo.padre
```

Al definir hijo:

```
padre : empleado

        inverse is empleado.hijos
```

Cuando se inserta o se borra un OID, el DBMS debe actualizar automáticamente la referencia a los objetos.

Navegabilidad

¿Indicar o no la navegabilidad? Se utiliza la navegabilidad en una relación no inversa entre dos objetos, donde solamente uno de los objetos contiene el OID del objeto relacionado.

Ilustración 11-75. Indicación de la navegabilidad en un objeto relacionado.

Colecciones

Las colecciones de objetos permiten a un objeto contener valores múltiples de una propiedad.

El estándar del ODMG propone:

- **SET**. Grupo no ordenado de objetos del mismo tipo. No acepta duplicados.
- **BAG**. Grupo no ordenado de objetos del mismo tipo. Acepta duplicados.
- **LIST**. Grupo ordenados de objetos del mismo tipo.

- **`ARRAY`**. Grupo ordenado de objetos del mismo tipo que pueden ser accedidos por posición.

Algunas de las operaciones que el ODMG establece para las colecciones:

- **`Cardinality`**. Retorna el número de elementos en un colección.
- **`Is_empty`**. Retorna "true" si una colección está vacía.
- **`Is_ordered`**. Retorna "true" si los elementos están ordenados de alguna manera.
- **`Insert_element`**. Agrega un elemento a la colección.
- **`remove_element`**. Remueve un elemento.

ODMG

OMG – Object Management Group. ODMG – Object Data Management Group. ODMG 3.0

ODL (Object Definition Language). Es un lenguaje declarativo. "What you want to know"… "how you get to the result".

OQL (Object Query Language). Es muy parecido al SQL-92. Con extensión para soportar conceptos de objetos.

OML (Object Manipulation Language).

Una de las razones del amplio uso de las bases de datos relacionales es la existencia de un lenguaje estándar de manipulación de datos (SQL).

ODL

El ODL es usado para declarar la estructura de clases incluyendo propiedades y signature de las operaciones. La implementación de las operaciones requiere de un lenguaje de programación específico y por lo tanto no es parte del ODL.

```
CREATE TYPE POINT_TYPE AS OBJECT ( x int y int)
CREATE TABLE CIRCLES ( RADIUS NUMBER CENTER POINT_TYPE)
```

Código 11-1. Uso del ODL.

```
Select e.nombre from Empleado e where e.nombre = "Carlos"
Select struct(n: e.nombre, s: e.sexo) from Empleado as e where e.edad < 18
```

Código 11-2. Uso del OQL.

Beneficios

- Modelado más apegado a la realidad. Se identifican objetos, no renglones y columnas.
- Transparencia en la programación OO.
- No es necesario traducir de OO a SQL, ODBC.
- Desempeño en la administración de objetos complejos. No es necesario traducir de OO a SQL, ODBC o JDBC.

Desventajas

- Complejidad.

- Inmadurez.
- Falta de estandarización.

Sistemas de bases de datos orientadas a objetos

Entre el software que destaca de las OO-DBMS se encuentra:

- Versant – Versant.
- POET.
- Objectivity DB – Objectivity.
- ObjectStore – Object Design.
- Jasmine – Computer Associates.
- Gemstone – Servio Corp..
- Ontos – Ontos.

Ejemplos

```
CREATE TYPE direccion AS
( calleNo      String;
  colonia      String;
  ciudad       String;
  estado       String;
  cp           String;)

CREATE TYPE proveedor AS
( nombre       String;
  dir          direccion
  tel          telefono
  www          String;
)

SELECT struct( n: p.name, s: p.sex)
FROM People as p
WHERE p.age < 18

SELECT SName: p.name
FROM p in People
WHERE p.age > 26

SELECT s.name
FROM Tutors t, t.students s

SELECT c.address
FROM Persons p, p.children c
WHERE   p.address.street="Main Street" AND
        count(p.children) >= 2 AND
        c.address.city != p.address.city


#Constructors
Object: Employee(name: “John”, salary: 15000)
```

Código 11-3. Ejemplo de uso del ODL y OQL.

11.7 Modelado en UML del Sistema de Acopio de Información de la UNAM. Caso práctico

El Sistema de Acopio de Información (SAI) es una aplicación cliente-servidor que opera a través de Internet. El servidor de la base de datos se encuentra en la Dirección General de Planeación y cada una de las entidades y dependencias establece su conexión a través de un navegador de WWW (cliente), una cuenta y una contraseña autorizadas.

Los objetivos del SAI son registrar las principales acciones realizadas durante el año por todas las entidades y dependencias de nuestra Institución, proporcionando así un panorama completo del quehacer universitario y responder a los diversos requerimientos internos y externos sobre el estado, actividades y trabajo académico la UNAM.

Para realizar estos objetivos se necesita el modelado del sistema y acceso a la base de datos.

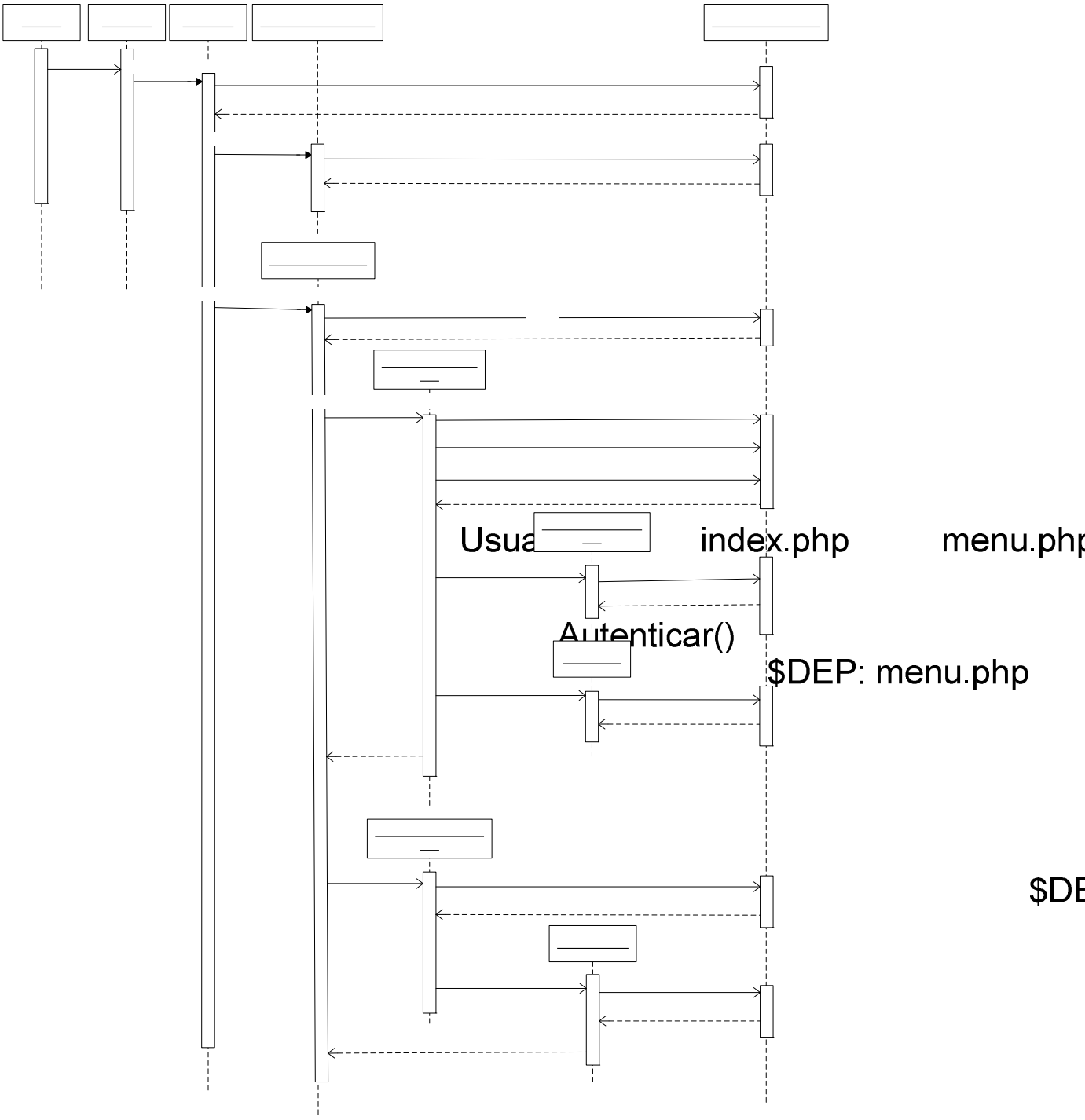


Ilustración 11-76. Uso de UML para el modelado del sistema SAI y su acceso a la base de datos para la memoria de la UNAM.

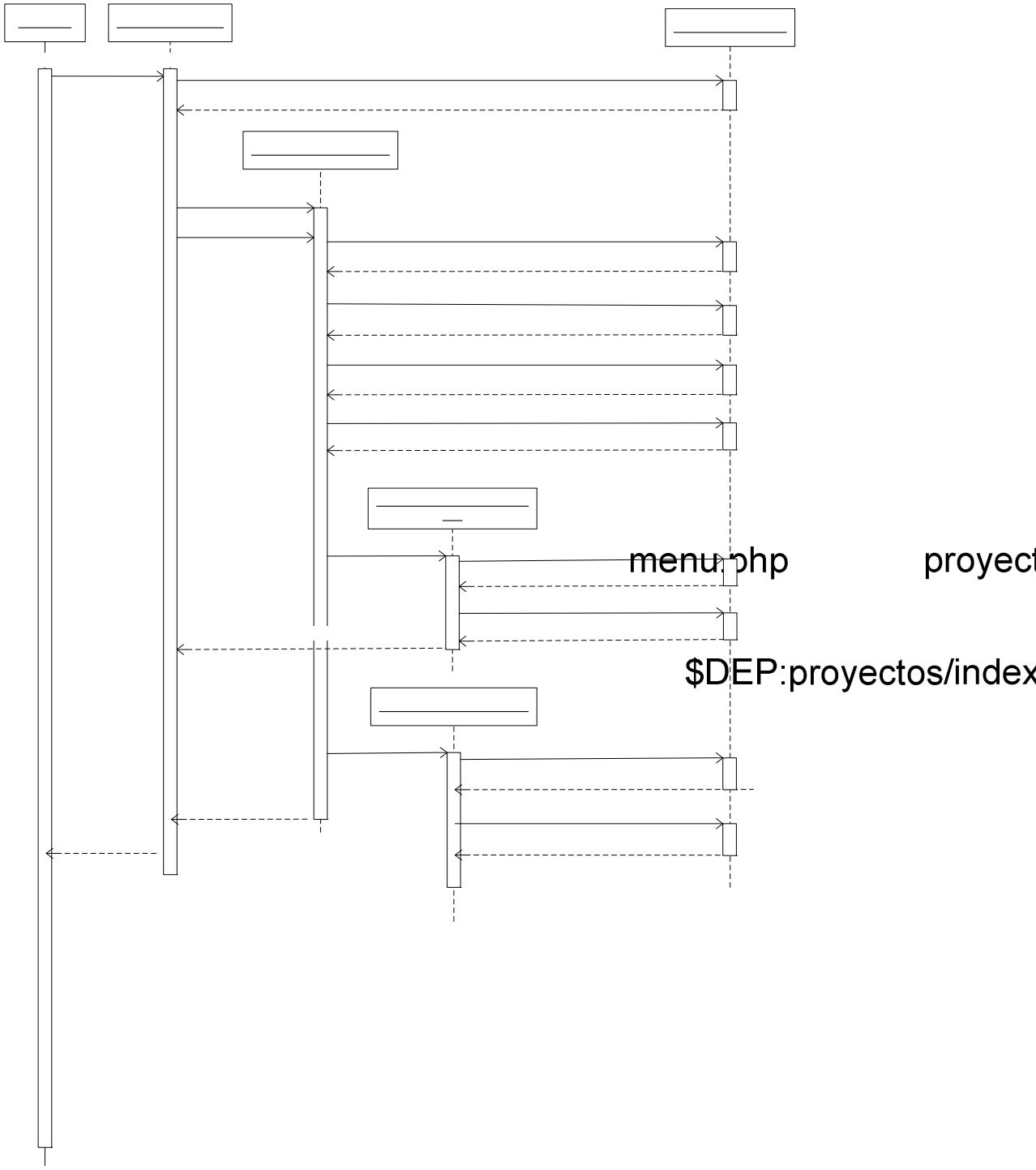


Ilustración 11-77. Uso de UML para el modelado del sistema SAI y su acceso a la base de datos para proyectos de investigación de la UNAM.

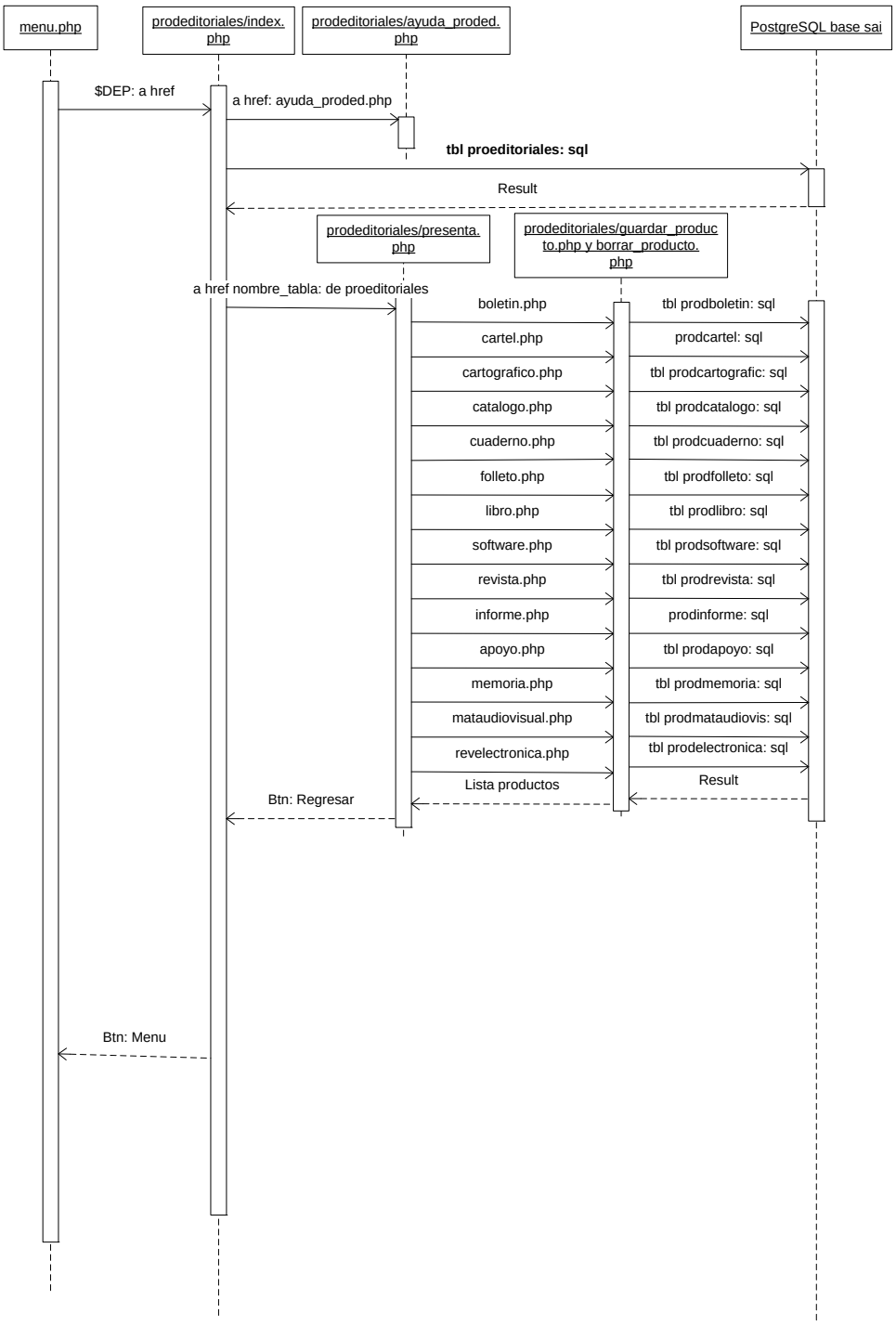


Ilustración 11-78. Uso de UML para el modelado del sistema SAI y su acceso a la base de datos para la producción editorial de la UNAM.

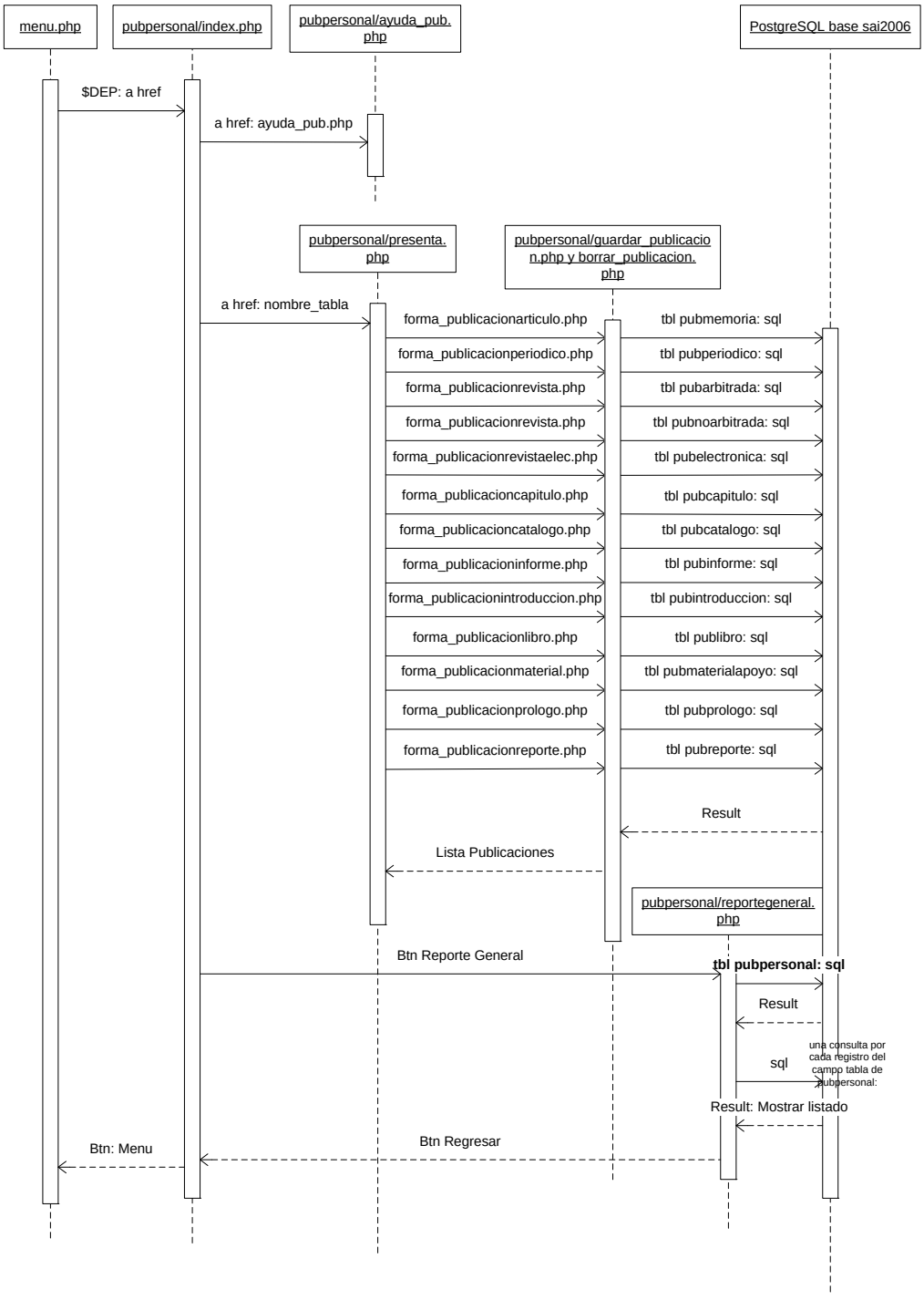


Ilustración 11-79. Uso de UML para el modelado del sistema SAI y su acceso a la base de datos para las publicacionesde la UNAM.

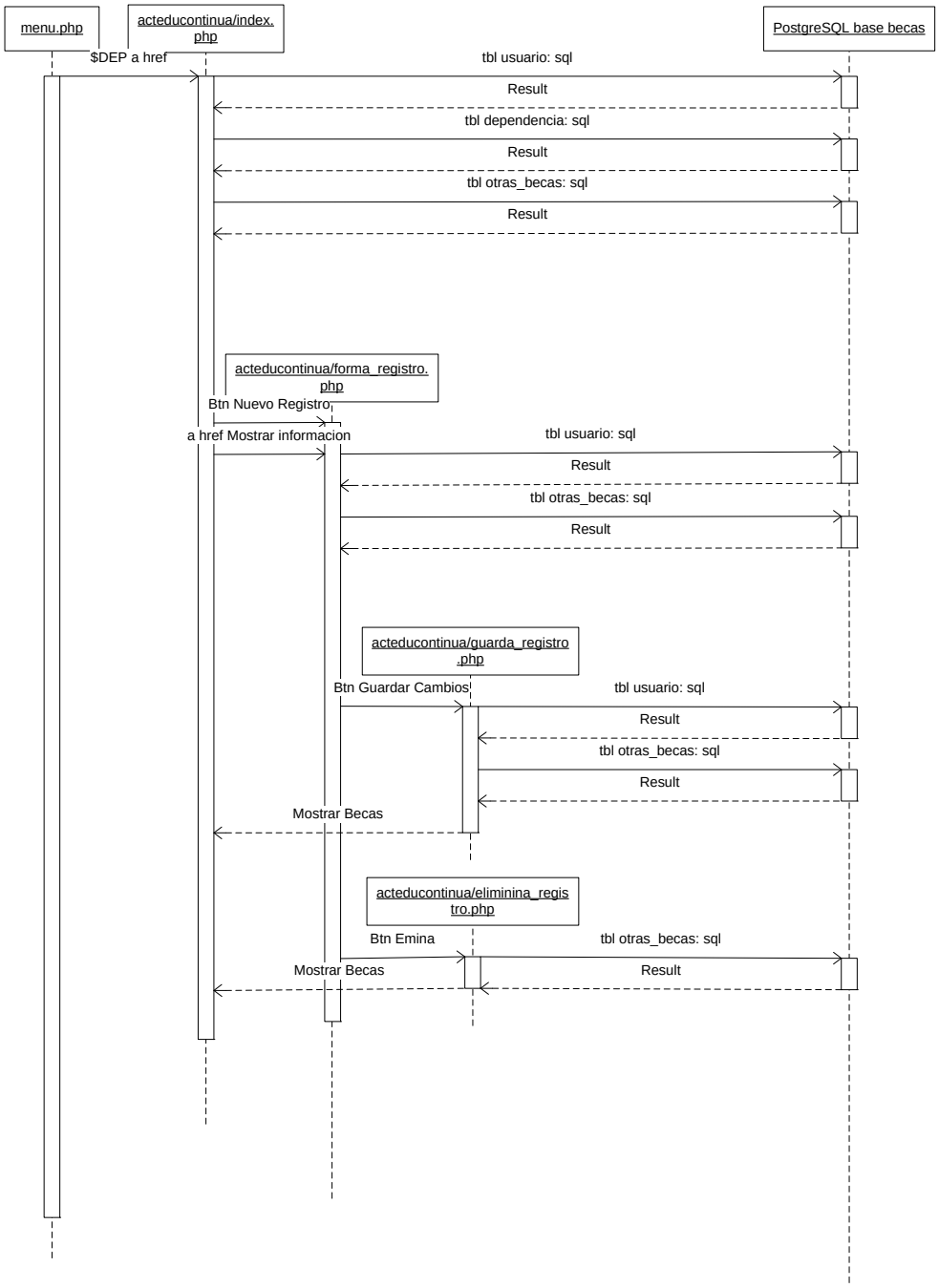


Ilustración 11-80. Uso de UML para el modelado del sistema SAI y su acceso a la base de datos para las becas de la UNAM.

Capítulo 12

TÓPICOS AVANZADOS DATA MINING

¿Cómo puede entenderse un fenómeno sobre la base de la interpretación de grandes volúmenes de datos? ¿De qué manera puede utilizarse la información para la toma de decisiones?, son algunos ejemplos de interrogantes comunes.

La respuesta a estas preguntas es el objetivo del data mining, un conjunto de técnicas agrupadas con el fin de crear mecanismos adecuados de dirección, entre ellas puede citarse la estadística, el reconocimiento de patrones, la clasificación y la predicción.

12.1 DATA MINING

La tecnología informática constituye la infraestructura fundamental de las grandes organizaciones y permite, hoy, registrar múltiples detalles de la vida de las empresas. Las bases de datos posibilitan almacenar cada transacción, así como otros muchos elementos que reflejan la interacción de la organización con otras organizaciones, clientes, o internamente, entre sus divisiones y empleados, etcétera.

Es imprescindible convertir los grandes volúmenes de datos existentes en experiencia, conocimiento y sabiduría, formas que atesora la humanidad para que sea útil a la toma de decisiones, especialmente en las grandes organizaciones y proyectos científicos. La búsqueda de información relevante siempre es útil a la administración empresarial: el control de la producción, el análisis de los mercados, el diseño en ingeniería y la exploración científica, porque pueden ofrecer las respuestas más apropiadas a las necesidades de información. Varias preguntas se relacionan frecuentemente con los datos, la información y el conocimiento. Su respuesta, demanda la participación de varios especialistas. ¿Cómo puede entenderse un fenómeno sobre la base de la interpretación de grandes volúmenes de datos? ¿De qué manera puede utilizarse la información para la toma de decisiones?, son algunos ejemplos de interrogantes comunes.

La respuesta a estas preguntas es el objetivo del DM, un conjunto de técnicas agrupadas con el fin de crear mecanismos adecuados de dirección, entre ellas puede citarse la estadística, el reconocimiento de patrones, la clasificación y la predicción.

Para descubrir patrones de relaciones útiles en un conjunto de datos se empezaron a utilizar métodos que fueron denominados de diferente forma. El término data mining, en inglés, no era, al principio, del agrado de muchos estadísticos, porque sus investigaciones estaban dirigidas a procesar y reprocesar suficientemente los datos, hasta que confirmasen o refutasen las hipótesis planteadas. Desde este ángulo, la minería de datos aplica una dinámica que se mueve en sentido contrario al método científico tradicional.

Con frecuencia, el investigador formula una hipótesis; luego, diseña un experimento para captar los datos necesarios y realizar los experimentos que confirmen o refuten la hipótesis

planteada. Este es un proceso, que realizado de forma rigurosa, debe generar nuevos conocimientos.

En la minería de datos, por el contrario, se captan y procesan los datos con la esperanza de que de ellos surja una hipótesis apropiada. Se desea que los datos nos describan o indiquen el porqué presentan determinada configuración y comportamiento.

El DM es un mecanismo de explotación, consistente en la búsqueda de información valiosa en grandes volúmenes de datos. Está muy ligada a los Data Warehousing que proporcionan la información histórica con la cual los algoritmos de minería de datos tienen la información necesaria para la toma de decisiones.

La minería de datos (DM) y el descubrimiento de conocimientos en bases de datos (KDD).

Existe cierta tendencia a identificar como sinónimos a la minería de datos y el descubrimiento de conocimientos en bases de datos, que de forma abreviada se refiere con las siglas KDD (del inglés *Knowledge Discovery in Data Bases*), la convergencia del aprendizaje automático, la estadística, el reconocimiento de patrones, la inteligencia artificial, las bases de datos, la visualización de datos, los sistemas para el apoyo a la toma de decisiones, la recuperación de información y otros muchos campos.

El KDD es el proceso completo de extracción de conocimientos, no trivial, previamente desconocidos y potencialmente útil a partir de un conjunto de datos, mientras que la minería de datos es una compilación de técnicas reunidas para crear mecanismos adecuados para la toma de decisiones. Entre estas técnicas se pueden citar la estadística, el reconocimiento de patrones, la clasificación y la predicción, la excavación de información relevante de la administración empresarial, el control de la producción, el análisis de los mercados, el diseño en ingeniería y la exploración científica. En otras palabras, el concepto minería de datos se asocia al proceso de construcción de reglas a partir de colecciones de datos con una finalidad previamente determinada y para su uso en la toma de decisiones con respecto a dicha finalidad. El concepto de KDD no comprende necesariamente esta segunda parte. Esta diferencia, muchas veces inadvertida, puede ser la causa de que ambos conceptos se utilicen indistintamente en gran parte de la literatura.

KDD = *proceso completo:* Extracción no trivial de conocimiento implícito, previamente desconocido y potencialmente útil, a partir de una base de datos. (Frawley *et al.,* 1991).

DM = *etapa de descubrimiento* en el proceso de KDD: Paso consistente en el uso de algoritmos concretos que generan una enumeración de patrones a partir de los datos preprocesados. (Fayyad *et al.,* 1996).

Etapas principales del proceso de data mining

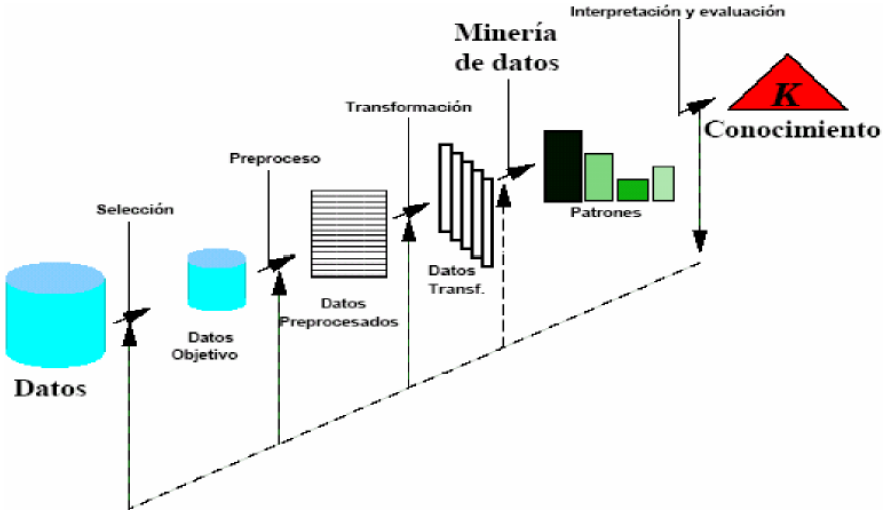


Ilustración 12-1. Etapas principales del proceso de data mining.

1. Determinación de los objetivos: delimitar los objetivos que el cliente desea bajo la orientación del especialista en DM.

2. Preprocesamiento de los datos: se refiere a la selección, la limpieza, el enriquecimiento, la reducción y la transformación de las bases de datos. Esta etapa consume generalmente alrededor del setenta por ciento del tiempo total de un proyecto de DM.

3. Determinación del modelo: se comienza realizando un análisis estadístico de los datos, y después se lleva a cabo una visualización gráfica de los mismos para tener una primera aproximación. Según los objetivos planteados y la tarea que debe llevarse a cabo, pueden utilizarse algoritmos desarrollados en diferentes áreas de la Inteligencia Artificial.

4. Análisis de los resultados: verifica si los resultados obtenidos son coherentes y los coteja con los obtenidos por el análisis estadístico y de visualización gráfica. El cliente determina si son novedosos y si le aportan un nuevo conocimiento que le permita considerar sus decisiones.

Fundamentos del data mining

Las técnicas de DM son el resultado de un largo proceso de investigación y desarrollo de productos. Esta evolución comenzó cuando los datos de negocios fueron almacenados por primera vez en computadoras, y continuó con mejoras en el acceso a los datos, y más recientemente con tecnologías generadas para permitir a los usuarios navegar a través de los datos en tiempo real. DM toma este proceso de evolución más allá del acceso y navegación retrospectiva de los datos, hacia la entrega de información prospectiva y proactiva. DM está listo para su aplicación en la comunidad de negocios porque está soportado por tres tecnologías que ya están suficientemente maduras:

- Recolección masiva de datos.
- Potentes computadoras con multiprocesadores.
- Algoritmos de data mining.

Los componentes esenciales de la tecnología de DM han estado bajo desarrollo por décadas, en áreas de investigación como estadística, inteligencia artificial y aprendizaje de máquinas. Hoy, la madurez de estas técnicas, junto con los motores de bases de datos relacionales de alta performance, hizo que estas tecnologías fueran prácticas para los entornos de Data Warehouse actuales.

El DM envuelve muchos diferentes algoritmos para resolver diferentes tareas. Todos estos algoritmos intentan estar en un modelo de datos. El algoritmo examina los datos y determina un modelo que es cerrado. Los algoritmos de DM pueden ser caracterizados consistiendo en tres partes:

- **Modelo**. El propósito del algoritmo es el de colocar el dato en un modelo.
- **Preferencia**. Algún criterio debe ser usado para colocar un modelo en otro.
- **Buscar**. Todos los algoritmos requieren alguna técnica para buscar el dato.

Modelos del data mining

El modelo que es creado puede ser predictivo o descriptivo.

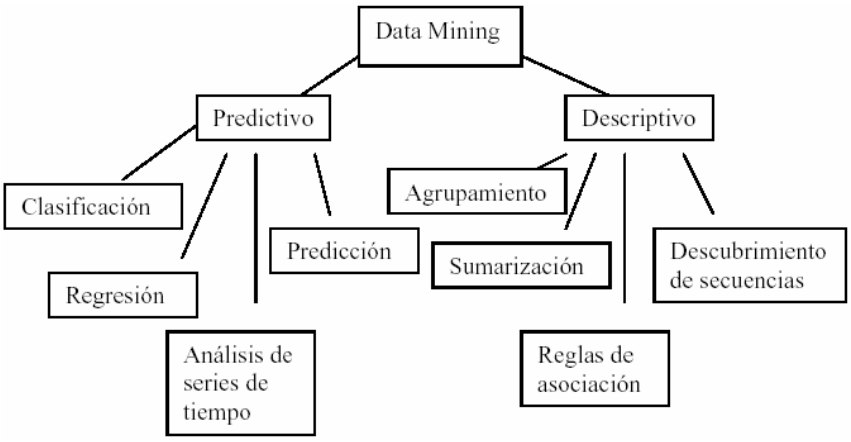


Ilustración 12-2. Modelos en los que se divide el data mining.

Modelo predictivo

Hace una predicción acerca de los valores de los datos usando la compresión de los resultados encontrados desde diferentes datos. Un modelo predictivo puede hacerse basado sobre el uso de otro historial de datos. Las tareas de este modelo incluyen la clasificación, regresión, análisis de series de tiempo y predicción.

Modelo descriptivo.

Éste identifica patrones o relaciones en los datos. A diferencia del modelo predictivo, sirve como un camino para explorar las propiedades de los datos examinados, no para predecir nuevas propiedades. El agrupamiento (clustering), sumarización, reglas de asociación y descubrimiento de secuencias son vistas como tareas del modelo descriptivo.

Tareas básicas del data mining.

Clasificación

Traza mapas de datos dentro de grupos o clases predefinidos. Esto se obtiene refiriéndose a como supervisar el aprendizaje, porque las clases son determinadas antes de examinar los datos. Los algoritmos de clasificación requieren que las clases sean definidas basadas sobre los valores de los atributos de los datos. Ellos obtienen la descripción de estas clases mirando las características de los datos conocidos existentes que pertenecen a las clases.

Regresión

Ésta se usa para un mapa de un detalle de un dato para la predicción de un valor real de una variable. En la actualidad, la regresión envuelve el aprendizaje de la función que hace este mapeo. La regresión asume que la meta del dato se encuentre dentro de un tipo de función conocida (ejemplo: lineal, logística, etc.) y entonces determina la mejor función de este tipo que los modelos dan al dato.

Análisis de series de tiempo.

Con el análisis de series de tiempo, el valor de un atributo es examinado de como varia éste sobre el tiempo. Los valores usualmente son obtenidos como sucesos separados por puntos en el tiempo (diariamente, semanalmente, etc.).

Predicción.

Muchas aplicaciones de DM del mundo real se pueden ver, prediciendo el dato futuro basado sobre los estados pasado y actual del dato. La predicción puede ser vista como un tipo de clasificación. Cabe notar que ésta es una tarea de DM que es diferente del modelo predictivo, aunque la tarea de predicción es un tipo del modelo predictivo.

Las aplicaciones de predicción incluyen el conocimiento del habla, aprendizaje de la máquina, reconocimiento de patrones, etc.

Agrupamiento (Clustering)

Es un proceso de dividir un conjunto de datos en grupos mutuamente excluyentes de tal manera que cada miembro de un grupo esté lo "más cercano" posible a otro, y grupos diferentes estén lo "más lejos" posible uno del otro, donde la distancia está medida con respecto a todas las variables disponibles.

Sumarización.

Son mapas de datos dentro de subseries o subconjuntos con simples descripciones asociadas. Ésta es llamada también caracterización o generalización. Ésta extrae o deriva información descriptiva acerca de la base de datos.

Reglas de asociación.

Establece asociaciones en base a los perfiles de los clientes sobre los cuales se está realizando el DM. Las reglas de Asociación están siempre definidas sobre atributos binarios. No es muy complicado generar reglas en grandes bases de datos. El problema es que tal algoritmo eventualmente puede dar información que no es relevante. DM envuelve modelos

para determinar patrones a partir de los datos observados. Los modelos juegan un rol de conocimiento inferido. Diciendo cuando el conocimiento representa conocimiento útil o no, esto es parte del proceso de extracción de conocimiento en bases de datos (Knowledge Discovery in Databases-KDD).

Descubrimiento de secuencias.

El descubrimiento de secuencias o análisis secuencial es usado para determinar patrones secuenciales en los datos. Estos patrones son basados sobre una secuencia de acciones en el tiempo. Los patrones son similares a las asociaciones, los datos o eventos son encontrados para ser relatados, pero la relación es basada en el tiempo.

Técnicas del data mining

Redes neuronales artificiales

Modelos predecibles no-lineales que aprenden a través del entrenamiento y semejan la estructura de una red neuronal biológica.

Árboles de decisión

Estructuras de forma de árbol que representan conjuntos de decisiones. Estas decisiones generan reglas para la clasificación de un conjunto de datos. Métodos específicos de árboles de decisión incluyen Árboles de Clasificación y Regresión (CART: Classification And Regression Tree) y Detección de Interacción Automática de Chi Cuadrado (CHAI: Chi Square Automatic Interaction Detection).

CART Árboles de clasificación y regresión.

Técnica usada para la clasificación de un conjunto da datos. Provee un conjunto de reglas que se pueden aplicar a un nuevo (sin clasificar) conjunto de datos para predecir cuáles registros darán un cierto resultado. Segmenta un conjunto de datos creando 2 divisiones. Requiere menos preparación de datos que CHAID.

CHAID Detección de interacción automática de Chi cuadrado.

*T*écnica similar a la anterior, pero segmenta un conjunto de datos utilizando tests de Chi cuadrado para crear múltiples divisiones.

Algoritmos genéticos

Técnicas de optimización que usan procesos tales como combinaciones genéticas, mutaciones y selección natural en un diseño basado en los conceptos de evolución.

Método del vecino más cercano

Una técnica que clasifica cada registro en un conjunto de datos basado en una combinación de las clases del(os) **k** registro(s) más similar(es) a él en un conjunto de datos históricos (donde **k 1**). Algunas veces se llama la técnica del **vecino** ≥ **k-más cercano**.

Regla de inducción.

La extracción de reglas **if-then** de datos basados en significado estadístico.

Extensiones del data mining

Web mining.

Consiste en aplicar las técnicas de minería de datos a documentos y servicios del Web (Kosala y otros, 2000). Todos los que visitan un sitio en Internet dejan huellas digitales (direcciones de IP, navegador, etc.) que los servidores automáticamente almacenan en una bitácora de accesos (Log). Las herramientas de Web mining analizan y procesan estos logs para producir información significativa. Debido a que los contenidos de Internet consisten en varios tipos de datos, como texto, imagen, vídeo, metadatos o hiperligas, investigaciones recientes usan el término multimedia data mining (minería de datos multimedia) como una instancia del Web mining (Zaiane y otros, 1998) para tratar ese tipo de datos. Los accesos totales por dominio, horarios de accesos más frecuentes y visitas por día, entre otros datos, son registrados por herramientas estadísticas que complementan todo el proceso de análisis del Web mining.

Text mining.

Dado que el ochenta por ciento de la información de una compañía está almacenada en forma de documentos, las técnicas como la categorización de texto, el procesamiento de lenguaje natural, la extracción y recuperación de la información o el aprendizaje automático, entre otras, apoyan al text mining (minería de texto). En ocasiones se confunde el text mining con la recuperación de la información (Information Retrieval o IR) (Hearst, 1999). Esta última consiste en la recuperación automática de documentos relevantes mediante indexaciones de textos, clasificación, categorización, etc. Generalmente se utilizan palabras clave para encontrar una página relevante. En cambio, el text mining se refiere a examinar una colección de documentos y descubrir información no contenida en ningún documento individual de la colección; en otras palabras, trata de obtener información sin haber partido de algo (Nasukawa y otros, 2001).

12.2 DATAWAREHOUSING

El DataWarehouse (DW) no es un producto que pueda ser comprado en el mercado, sino más bien un concepto que debe ser construido. DW es una combinación de conceptos y tecnología que cambian significativamente la manera en que es entregada la información a la gente de negocios. El objetivo principal es satisfacer los requerimientos de información internos de la empresa para una mejor gestión, con eficiencia y facilidad de acceso.

El DW puede verse como una bodega donde están almacenados todos los datos necesarios para realizar las funciones de gestión de la empresa, de manera que puedan utilizarse fácilmente según se necesiten. El contenido de los datos, la organización y estructura son dirigidos a satisfacer las necesidades de información de analistas.

El DW intenta responder a la compleja necesidad de obtención de información útil sin el sacrificio del rendimiento de las aplicaciones operacionales, debido a lo cual se ha convertido actualmente en una de las tendencias tecnológicas más significativas en la administración de información.

Los almacenes de datos (o Datawarehouse) generan bases de datos tangibles con una perspectiva histórica, utilizando datos de múltiples fuentes que se fusionan en forma congruente. Estos datos se mantienen actualizados, pero no cambian al ritmo de los sistemas transaccionales. Muchos DW se

diseñan para contener un nivel de detalle hasta el nivel de transacción, con la intención de hacer disponible todo tipo de datos y características, para reportar y analizar. Así un DW resulta ser un recipiente de datos transaccionales para proporcionar consultas operativas, y la información para poder llevar a cabo análisis multidimensional. De esta forma, dentro de un almacén de datos existen dos tecnologías complementarias, una relacional para consultas y una multidimensional para análisis.

Existen muchas definiciones para el DW, la más conocida fue propuesta por Inmon (considerado el padre de las Bases de Datos) en 1992: "Un DW es una colección de datos orientados a temas, integrados, no-volátiles y variante en el tiempo, organizados para soportar necesidades empresariales". En 1993, Susan Osterfeldt publica una definición que sin duda acierta en la clave del DW: "Yo considero al DW como algo que provee dos beneficios empresariales reales: Integración y Acceso de datos. DW elimina una gran cantidad de datos inútiles y no deseados, como también el procesamiento desde el ambiente operacional clásico". Esta última definición refleja claramente el principal beneficio que el DW aporta a la empresa, eliminar aquellos datos que obstaculizan la labor de análisis de información y entregar la información que se requiere en la forma más apropiada, facilitando así el proceso de gestión.

DW se sustenta en un procesamiento distinto al utilizado por los sistemas operacionales, OLAP (Procesamiento Analítico En Línea), el cual surge como un proceso para ser usado en el análisis de negocios y otras aplicaciones que requieren una visión flexible del negocio.

Datamart

El concepto DataMart es una extensión natural del DW, y está enfocado a un departamento o área especifica, como por ejemplo los departamentos de Finanzas o Marketing.

Permitiendo así un mejor control de la información que se está abarcando.

OLAP, R-OLAP y M-OLAP.

Un sistema OLAP se puede entender como la generalización de un generador de informes. Las aplicaciones informáticas clásicas de consulta, orientadas a la toma de decisiones, deben ser programadas. Atendiendo a las necesidades del usuario, se crea una u otra interfaz. Sin embargo, muchos desarrolladores se dieron cuenta de que estas aplicaciones eran susceptibles de ser generalizadas y servir para casi cualquier necesidad, esto es, para casi cualquier base de datos. Los sistemas OLAP evitan la necesidad de desarrollar interfaces de consulta, y ofrecen un entorno único valido para el análisis de cualquier información histórica, orientado a la toma de decisiones. A cambio, es necesario definir dimensiones, jerarquías y variables, organizando de esta forma los datos.

Para los desarrolladores de aplicaciones acostumbrados a trabajar con bases de datos relacionales, el diseño de una base de datos multidimensional puede ser complejo o al menos, extraño. Pero en general, nuestra experiencia nos dice que el diseño de dimensiones y variables es mucho más sencillo e intuitivo que un diseño relacional. Esto es debido a que las dimensiones y variables son reflejo directo de los informes en papel utilizados por la organización.

Una vez que se ha decidido emplear un entorno de consulta OLAP, se ha de elegir entre R-OLAP y M-OLAP. R-OLAP es la arquitectura de base de datos multidimensional en la que los datos se encuentran almacenados en una base de datos relacional, la cual tiene forma de estrella (también llamada copo de nieve o araña). En R-OLAP, en principio la base de datos

sólo almacena información relativa a los datos en detalle, evitando acumulados (evitando redundancia).

En un sistema M-OLAP, en cambio, los datos se encuentran almacenados en archivos con estructura multidimensional, los cuales reservan espacio para todas las combinaciones de todos los posibles valores de todas las dimensiones de cada una de las variables, incluyendo los valores de dimensión que representan acumulados. Es decir, un sistema M-OLAP contiene precalculados (almacenados) los resultados de todas las posibles consultas a la base de datos.

M-OLAP consigue consultas muy rápidas a costa de mayores necesidades de almacenamiento, y retardos en las modificaciones (que no deberían producirse salvo excepcionalmente), y largos procesos *batch* de carga y cálculo de acumulados. En R-OLAP, al contener sólo las combinaciones de valores de dimensión que representan detalle, es decir, al no haber redundancia, el archivo de base de datos es pequeño. Los procesos *batch* de carga son rápidos (ya que no se requiere agregación), y sin embargo, las consultas pueden ser muy lentas, por lo que se aplica la solución de tener al menos algunas consultas precalculadas.

En M-OLAP, el gran tamaño de las variables multidimensionales o el retardo en los procesos *batch* puede ser un inconveniente.

Características de un Datawarehouse

Entre las principales se tiene:

- Orientado al tema.
- Integrado.
- De tiempo variante.
- No volátil.

Orientado a temas.

Una primera característica del DW es que la información se clasifica en base a los aspectos que son de interés para la empresa. Siendo así, los datos tomados están en contraste con los clásicos procesos orientados a las aplicaciones.

Integración.

El aspecto más importante del ambiente DW es que la información encontrada al interior está siempre integrada.

La integración de datos se muestra de muchas maneras: en convenciones de nombres consistentes, en la medida uniforme de variables, en la codificación de estructuras consistentes, en atributos físicos de los datos consistentes, fuentes múltiples y otros.

De tiempo variante.

Toda la información del DW es requerida en algún momento. Esta característica básica de los datos en un depósito, es muy diferente de la información encontrada en el ambiente operacional. En éstos, la información se requiere al momento de acceder. En otras palabras, en el ambiente operacional, cuando se accede a una unidad de información, se espera que los valores requeridos se obtengan a partir del momento de acceso.

Como la información en el DW es solicitada en cualquier momento (es decir, no "ahora mismo"), los datos encontrados en el depósito se llaman de "tiempo variante".

Los datos históricos son de poco uso en el procesamiento operacional. La información del depósito por el contraste, debe incluir los datos históricos para usarse en la identificación y evaluación de tendencias.

El tiempo variante se muestra de varias maneras:

1. La más simple es que la información representa los datos sobre un horizonte largo de tiempo - desde cinco a diez años. El horizonte de tiempo representado para el ambiente operacional es mucho más corto - desde valores actuales hasta sesenta a noventa días. Las aplicaciones que tienen un buen rendimiento y están disponibles para el procesamiento de transacciones, deben llevar una cantidad mínima de datos si tienen cualquier grado de flexibilidad. Por ello, las aplicaciones operacionales tienen un corto horizonte de tiempo, debido al diseño de aplicaciones rígidas.

2. La segunda manera en la que se muestra el tiempo variante en el DW está en la estructura clave. Cada estructura clave en el DW contiene, implícita o explícitamente, un elemento de tiempo como día, semana, mes, etc. El elemento de tiempo está casi siempre al pie de la clave concatenada, encontrada en el DW. En ocasiones, el elemento de tiempo existirá implícitamente, como el caso en que un archivo completo se duplica al final del mes, o al cuarto.

3. La tercera manera en que aparece el tiempo variante es cuando la información del DW, una vez registrada correctamente, no puede ser actualizada. La información del DW es, para todos los propósitos prácticos, una serie larga de "snapshots" (vistas instantáneas).

No volátil

La información es útil sólo cuando es estable. Los datos operacionales cambian sobre una base momento a momento. La perspectiva más grande, esencial para el análisis y la toma de decisiones, requiere una base de datos estable.

Estructura del Datawarehouse

Los DW tienen una estructura distinta. Hay niveles diferentes de esquematización y detalle que delimitan el DW.

Detalle de datos actuales

En gran parte, el interés más importante radica en el detalle de los datos actuales, debido a que:

- Refleja las ocurrencias más recientes, las cuales son de gran interés.
- Es voluminoso, ya que se almacena al más bajo nivel de granularidad.
- Casi siempre se almacena en disco, el cual es de fácil acceso, aunque su administración sea costosa y compleja.

Detalle de datos antiguos.

La data antigua es aquella que se almacena sobre alguna forma de almacenamiento masivo. No es frecuentemente su acceso y se almacena a un nivel de detalle, consistente con los datos detallados actuales. Mientras no sea prioritario el almacenamiento en un medio de almacenaje alterno, a causa

del gran volumen de datos unido al acceso no frecuente de los mismos, es poco usual utilizar el disco como medio de almacenamiento.

Datos ligeramente resumidos

La data ligeramente resumida es aquella que proviene desde un bajo nivel de detalle encontrado al nivel de detalle actual. Este nivel del DW casi siempre se almacena en disco. Los puntos en los que se basa el diseñador para construirlo son:

- Que la unidad de tiempo se encuentre sobre la esquematización hecha.
- Qué contenidos (atributos) tendrá la data ligeramente resumida.

A veces se encuentra en el ambiente de DW y en otros, fuera del límite de la tecnología que ampara al DW. (De todos modos, los datos completamente resumidos son parte del DW sin considerar donde se alojan los datos físicamente).

Metadata

El componente final del DW es el de la metadata. De muchas maneras la metadata se sitúa en una dimensión diferente al de otros datos del DW, debido a que su contenido no es tomado directamente desde el ambiente operacional. La metadata juega un rol especial y muy importante en el DW y es usada como:

- Un directorio para ayudar al analista a ubicar los contenidos del DW.
- Una guía para la trazabilidad de los datos, de cómo se transforma, del ambiente operacional al de DW.
- Una guía de los algoritmos usados para la esquematización entre el detalle de datos actual, con los datos ligeramente resumidos y éstos, con los datos completamente resumidos, etc.

La metadata juega un papel mucho más importante en un ambiente DW que en un operacional clásico.

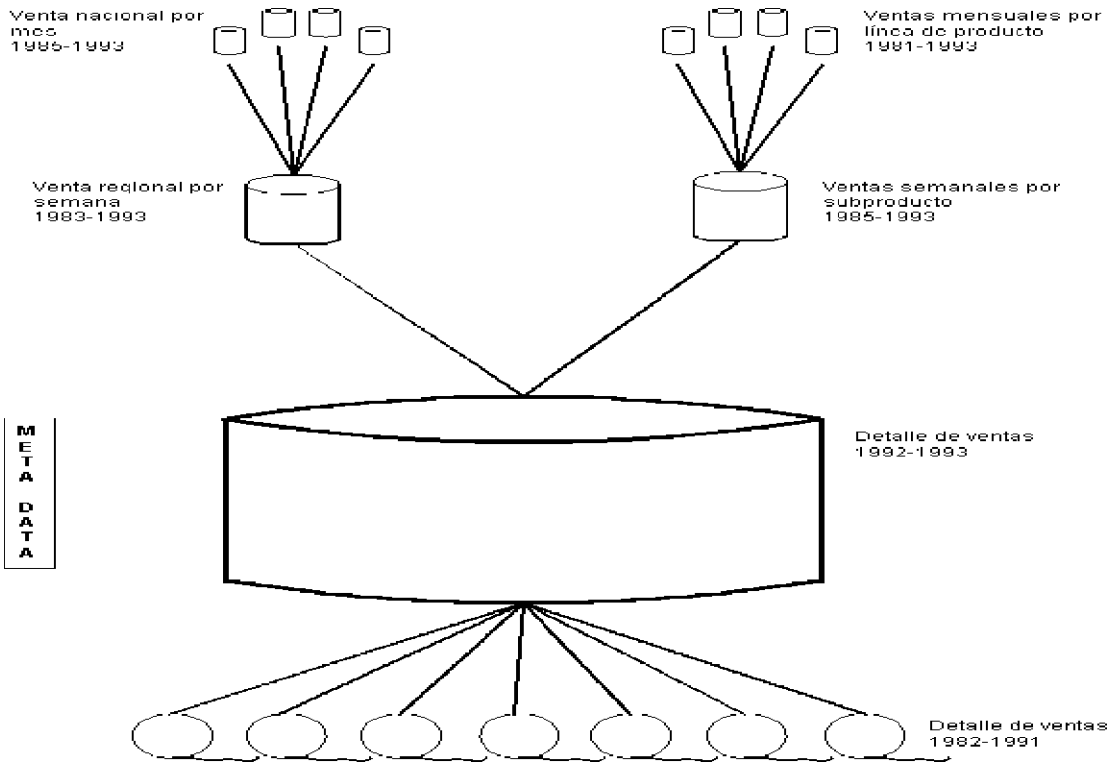


Ilustración 12-3. Segmentación de los metadatos.

Arquitectura de un Datawarehouse

Componentes y estructuras.

El término DataWarehouse se utiliza indistintamente para hablar de la arquitectura en sí como también para uno de los componentes que la conforman, específicamente el que tiene relación con el almacenamiento físico de los datos.

La estructura básica de la arquitectura DW incluye:

1. Datos operacionales: un origen de datos para el componente de almacenamiento físico DW.

2. Extracción de Datos: selección sistemática de datos operacionales usados para poblar el componente de almacenamiento físico DW.

3. Transformación de datos: Procesos para sumarizar y realizar otros cambios en los datos operacionales para reunir los objetivos de orientación a temas e integración principalmente.

4. Carga de Datos: inserción sistemática de datos en el componente de almacenamiento físico DW.

5. Datawarehouse: almacenamiento físico de datos de la arquitectura DW.

6. Herramientas de Acceso al componente de almacenamiento físico DW: herramientas que proveen acceso a los datos.

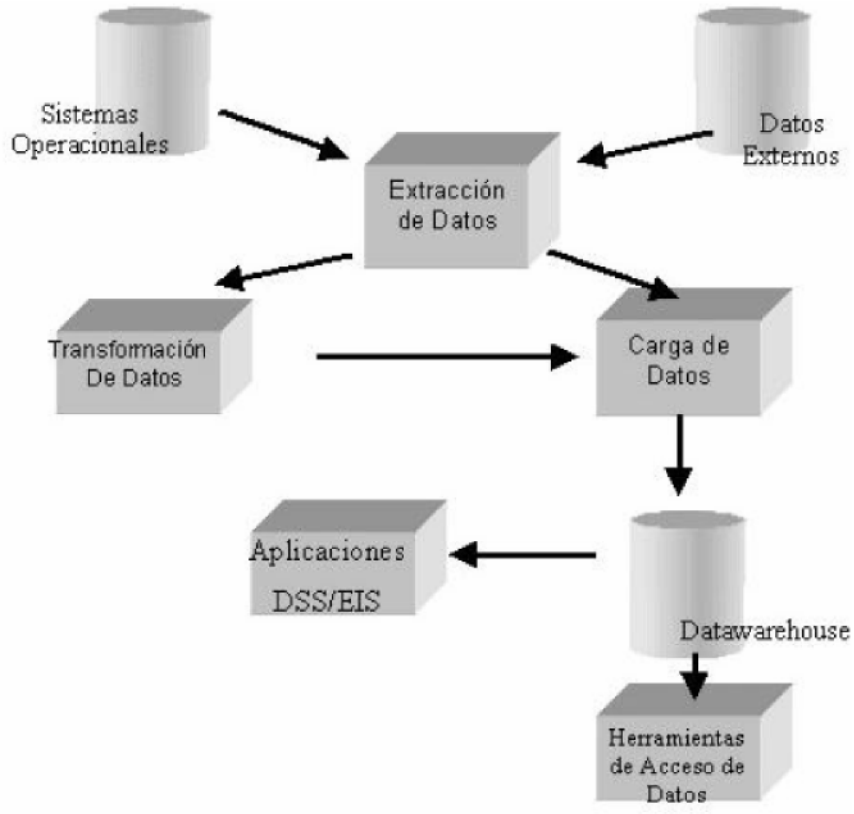


Ilustración 12-4. Esquema de la relación entre componentes y estructuras del Datawarehouse.

Transformación de datos y metadata

Transformación de datos

Uno de los desafíos de cualquier implementación de DW, es el problema de transformar los datos. La transformación se encarga de las inconsistencias en los formatos de datos y la codificación, que pueden existir dentro de una base de datos única y que casi siempre existen cuando múltiples bases de datos contribuyen al DW.

Metadata

Otro aspecto de la arquitectura de DW es crear soporte a la metadata. Metadata es la información sobre los datos que se alimenta, se transforma y existe en el DW. Metadata es un concepto genérico, pero cada implementación de la metadata usa técnicas y métodos específicos.

Estos métodos y técnicas son dependientes de los requerimientos de cada organización, de las capacidades existentes y de los requerimientos de interfaces de usuario. Hasta ahora, no hay normas para la metadata, por lo que la metadata debe definirse desde el punto de vista del software DW, seleccionado para una implementación específica.

Típicamente, la metadata incluye los siguientes puntos:

1. Las estructuras de datos que dan una visión de los datos al administrador de datos.

2. Las definiciones del sistema de registro desde el cual se construye el DW.

3. Las especificaciones de transformaciones de datos que ocurren tal como la fuente de datos se replica al DW.

El modelo de datos del DW (es decir, los elementos de datos y sus relaciones). Un registro de cuando los nuevos elementos de datos se agregan al DW y cuando los elementos de datos antiguos se eliminan o se resumen. Los niveles de sumarización, el método de sumarización y las tablas de registros de su DW.

Algunas implementaciones de la metadata también incluyen definiciones de la(s) vista(s) presentada(s) a los usuarios del DW. Típicamente, se definen vistas múltiples para favorecer las preferencias variadas de diversos grupos de usuarios. En otras implementaciones, estas descripciones se almacenan en un Catálogo de Información.

Los esquemas y subesquemas para bases de datos operacionales, forman una fuente óptima de entrada cuando se crea la metadata. Hacer uso de la documentación existente, especialmente cuando está disponible en forma electrónica, puede acelerar el proceso de definición de la metadata del ambiente DW.

La metadata sirve, en un sentido, como el corazón del ambiente DW. Crear definiciones de metadata completa y efectiva puede ser un proceso que consuma tiempo, pero lo mejor de las definiciones y si se usa herramientas de gestión de software integrado, son los esfuerzos que darán como resultado el mantenimiento del DW.

Flujo de datos

Existe un flujo de datos normal y predecible dentro del DW.

Los datos ingresan al datawarehouse desde el ambiente operacional. (Hay pocas excepciones a esta regla).

Al ingresar al datawarehouse, la información va al nivel de detalle actual. Se queda allí y se usa hasta que ocurra uno de los tres eventos siguientes:

- Sea eliminado.
- Sea resumido.
- Sea archivado.

Con el proceso de desactualización en un DW se mueve el detalle de la data actual a data antigua, basado en el tiempo de los datos. El proceso de esquematización usa el detalle de los datos para calcular los datos en forma ligera y completamente resumidos.

Hay pocas excepciones al flujo mostrado. Sin embargo, en general, para la mayoría de datos encontrados en un DW, el flujo de la información es como se ha explicado.

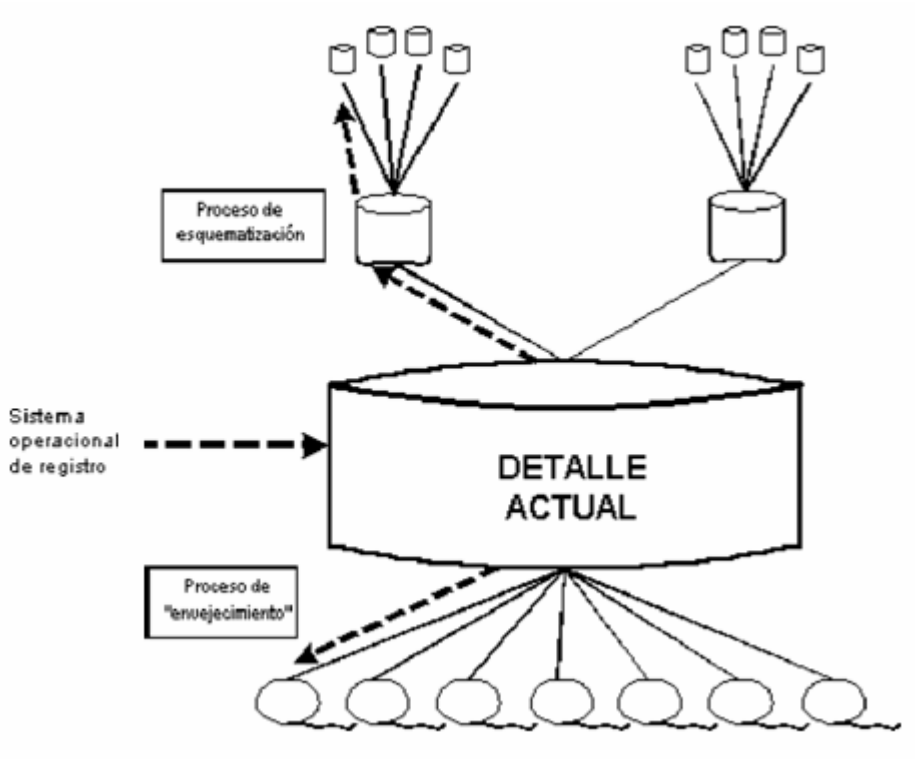


Ilustración 12-5. Esquema del flujo de datos en Datawarehouse.

Medios de almacenamiento para información antigua

Hay una amplia variedad de medios de almacenamiento que deben considerarse para almacenar datos más antiguos.

Dependiendo del volumen de información, la frecuencia de acceso, el costo de los medios y el tipo de acceso, es probable que otros medios de almacenamiento sirvan a las necesidades del nivel de detalle más antiguo en el DW.

- Almacenamiento foto óptico.
- Raid.
- Microficha.
- Cinta magnética.
- Almacenamiento en masa.

12.3 BASE DE DATOS INTELIGENTES

Las bases de datos inteligentes representan una tecnología para la administración de la información que fue envuelta como resultado de la integración de las bases de datos tradicionales con otros campos como lo son:

- Programación orientada a objetos.
- Sistemas expertos.
- Hypermedia.
- Recepción de información en línea.

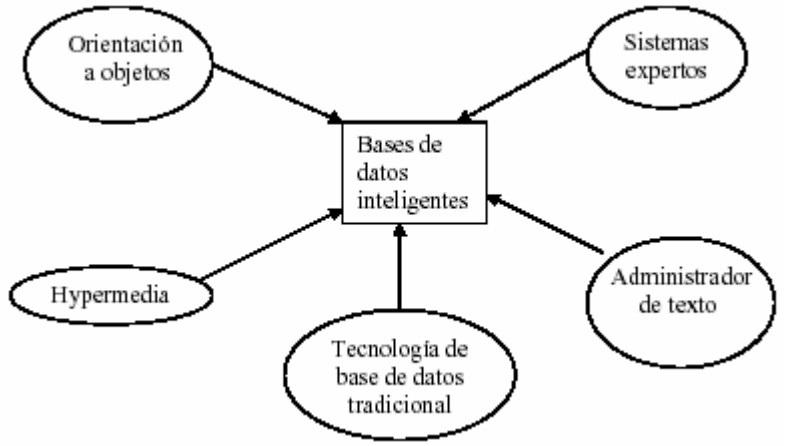


Ilustración 12-6. Campos que se integran en las base de datos inteligentes.

La arquitectura de las bases de datos inteligentes consta de tres niveles, los cuales son:

- High-level tools.
- High-level user interface
- Intelligent database engine

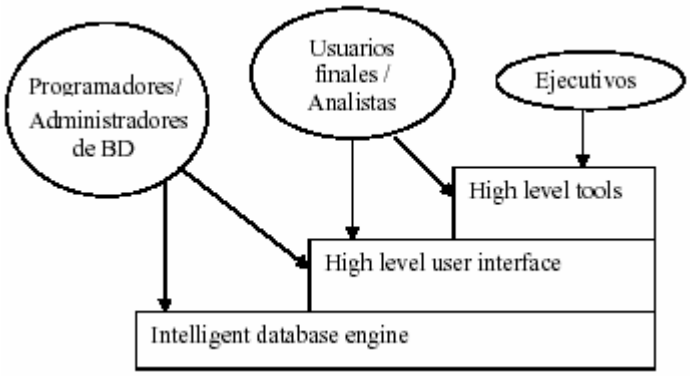


Ilustración 12-7. Niveles de las bases de datos inteligentes.

High-level tools

Estas herramientas proveen al usuario muchas facilidades como son búsquedas inteligentes, calidad de los datos y control de integridad, además de descubrimiento autómata. Estas herramientas representan una biblioteca externa de herramientas poderosas que algunos usuarios quizás encuentren útiles y otros no. Muchas de éstas pueden ser clasificadas como técnicas de administración para la información.

Se pueden obtener siete tipos de herramientas High-level tools, las cuales son:

- Herramientas de descubrimiento de conocimiento.
- Herramientas de integridad de datos y control de calidad.
- Herramientas de administración de hypermedia.
- Herramientas de presentación y despliegue de los datos.
- Herramientas de análisis de escenarios y de soporte de decisiones.
- Herramientas de administración de formato de datos.
- Herramientas de diseño de sistemas inteligentes.

Herramientas de descubrimiento de conocimiento.

Esta categoría incluye herramientas para el análisis de datos, aprendizaje de la máquina, y análisis estadísticos. Estas herramientas representan una nueva y excitante frontera en la tecnología de base de datos inteligentes que nos permite extraer conocimiento desde datos.

Herramientas de integridad de datos y control de calidad.

Las herramientas en esta segunda categoría son necesarias debido a los no deseados efectos del crecimiento en número y tamaño de las bases de datos.

Herramientas de administración de hypermedia.

Las herramientas de esta tercera categoría permiten a los desarrolladores y usuarios construir sistemas de información hypermedia que combinan una forma libre de colocar texto, dato, imágenes, sonido, etc. Esta categoría refleja el hecho que la información quizás será expresada en muchas diferentes formas o medios y que métodos son necesarios para organizar y acceder a estas diferentes formas de información.

Herramientas de presentación y despliegue de los datos.

Dando una gran base de datos, muchos usuarios desean ver y desplegar alguno de los datos o resúmenes. Así las herramientas de presentación y despliegue de la categoría cuatro proveen gráficos, formas y otros tipos de presentación de datos.

Herramientas de análisis de escenarios y de soporte de decisiones.

Las herramientas de la categoría cinco permiten una fusión uniforme entre las hojas de cálculo, bases de datos, sistemas de modelado financiero, etc.

Herramientas de administración de formato de datos.

Las herramientas de la categoría seis permiten a los usuarios la transformación entre los formatos de datos, por ejemplo, la fusión de un archivo ASCII con un dato en formato dBASE y datos obtenidos desde una base de datos DB2 generados automáticamente con un query SQL. Estas herramientas son indispensables para aplicación que confíe sobre el análisis real de datos.

Herramientas de diseño de sistemas inteligentes.

Las herramientas de la categoría siete proveen facilidades para diseñar bases de datos inteligentes. Aunque el campo del diseño de base de datos, diseño de sistemas de información y diseño de sistemas expertos tiene separado su mantenimiento en el pasado, su integración en una base de datos inteligente es esencial. Así, estas herramientas permiten a los desarrolladores y administradores de sistemas un mejor diseño y mantenimientos de la base de datos inteligente.

High-level user interface.

En este nivel es en el cual los usuarios interactúan directamente. Este nivel crea el modelo de la tarea y ambiente de base de datos con el que el usuario interactuará.

La interfaz del usuario es presentada en dos aspectos. Éste es un modelo central que es presentado al usuario. Este modelo central consiste de la representación orientada a objetos de la información con una colección de herramientas integradas para crear nuevos tipos de objetos, buscando y respondiendo preguntas. En adición, estás son una colección de herramientas de alto nivel, el cual aunque no es una parte esencial del modelo central, da un realce a la funcionalidad del sistema de base de datos inteligente para ciertas clases y usuarios.

Normalmente se pueden distinguir dos niveles de la interfaz del usuario, las cuales son:

- El nivel físico.
- El nivel cognitivo.

El nivel físico de la interfaz es típicamente más obvio, consistiendo de entradas y salidas de dispositivos, semejantes al ratón, teclado, monitor. En contraste, el nivel cognitivo de la interfaz es más difícil para describir., consistiendo del modelo subyacente usado para presentar la información, la interpretación que el usuario entonces hace, y las intensiones que entonces el usuario formula.

Intelligent database engine.

Este es el nivel base. El motor de base de datos inteligente incorpora un modelo que permite hacer una deductiva orientación a objetos representación de la información, que puede ser expresada y operada sobre una variedad de caminos. El motor incluye procedimientos de inferencia de encadenamiento hacia delante y hacia atrás. Muchas de estas características del motor integrado dependerán sobre las especificaciones del ambiente de hardware y software en el cual la base de datos inteligente es implementada.

La interfaz del usuario es soportada por una colección de capacidades de la base de datos. Estas capacidades son el mecanismo que permite un sistema de administración de base de datos o aplicaciones de administración de información su funcionamiento. Ejemplo de estas capacidades incluye el procesamiento de query's y la habilidad para llevar fuera del razonamiento deductivo.

La inteligencia de la interfaz del usuario esta en gran parte determinada por la inteligencia de la aplicación subyacente. Las siguientes son algunas características de un sistema de base de datos y este nivel que realza el total de inteligencia del sistema:

- Modelo de datos basado en el conocimiento y orientado a objetos.
- Bases de datos integradas y motoras de inferencia.
- Búsquedas sensitivas al contexto y/o de estructura sensitiva.
- Soporte de múltiples medios de almacenamiento.
- Administración inteligente de versión, recuperación y resistencia.
- Soporte de transacciones y concurrencia.
- Optimización de query's.

Un modelo de datos basado en el conocimiento y orientado a objetos permite la representación de información en una forma la cuál refleja lo más fácilmente posible la percepción de los usuarios del mundo real.

Las bases de datos integradas y los motores de inferencia siguen después del uso del modelo de datos basado en el conocimiento. Éstos permiten la recuperación deductiva para ser llevado en un ambiente fuera donde la búsqueda e inferencia de la información son integradas.

La búsqueda estructura sensitiva envuelve la recuperación del conocimiento basado sobre la forma. La búsqueda sensitiva envuelve sabiendo donde buscar la información relevante basado en el contexto.

Los medios múltiples de almacenamiento permiten una variedad de tipos (gráficas, sonido, etc.) para ser eficientemente almacenados y recuperados dentro de la base de datos.

Administración inteligente de versión se asegura que las versiones previas y la actual de las bases de datos de desarrollo, sean recuperadas eficientemente. La recuperación y resistencia son las ediciones que tratan el grado a el cual la base de datos maneja las demandas operacionales que experimenta, incluyendo simulación de acceso de muchos usuarios y fallas de equipo. Así como muchas convencionales bases de datos, las bases de datos inteligentes soportan transacciones atómicas concurrentes. En adición, el motor subyacente de una base de datos inteligente ejecuta extensiva optimización de query para proveer adecuadas respuestas de tiempo real para complejos queries envolviendo bases de conocimiento orientadas a objetos.

12.4 BASE DE DATOS MULTIDIMENSIONALES

El Modelamiento Dimensional es una técnica para modelar bases de datos simples y entendibles al usuario final. La idea fundamental es que el usuario visualice fácilmente la relación que existe entre los distintos componentes del modelo.

Consideremos un punto en el espacio. El espacio se define a través de sus ejes coordenados (por ejemplo X, Y, Z). Un punto cualquiera de este espacio quedará determinado por la intersección de tres valores particulares de sus ejes.

Si se le asignan valores particulares a estos ejes. Digamos que el eje X representa Productos, el eje Y representa el Mercado y, el eje Z corresponde al Tiempo. Se podría tener por ejemplo, la siguiente combinación: producto = madera, mercado = Concepción, tiempo = diciembre-1998. La intersección de estos valores nos definirá un solo punto en nuestro espacio. Si el punto que buscamos, lo definimos como la cantidad de madera vendida, entonces se tendrá un valor específico y único para tal combinación.

En el modelo multidimensional cada eje corresponde a una dimensión particular. Entonces la dimensionalidad de nuestra base estará dada por la cantidad de ejes (o dimensiones) que le asociemos.

Cuando una base puede ser visualizada como un cubo de tres o más dimensiones, es más fácil para el usuario organizar la información e imaginarse en ella cortando y rebanando el cubo a través de cada una de sus dimensiones, para buscar la información deseada.

Por ejemplo: La descripción de una organización típica es: "Nosotros vendemos productos en varios mercados, y medimos nuestro desempeño en el tiempo": Un diseñador dimensional lo verá como: "Nosotros vendemos productos en varios mercados, y medimos nuestro desempeño en el tiempo. Donde cada palabra subrayada corresponde a una dimensión.

Esto puede visualizarse como un cubo, donde cada punto dentro del cubo es una intersección de coordenadas definidas por los lados de éste (dimensiones). Ejemplos de medidas son: unidades producidas, unidades vendidas, costo de unidades producidas, ganancias ($) de unidades vendidas, etc.

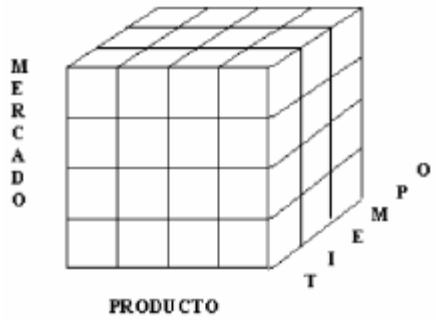


Ilustración 12-8. Ejemplo de dimensiones a definir en una base de datos multidimensional.

Características del modelo multidimensional

En general, la estructura básica de un DW para el Modelo Multidimensional está definida por dos elementos: esquemas y tablas.

Tablas DW*:* como cualquier base de datos relacional, un DW se compone de tablas. Hay dos tipos básicos de tablas en el Modelo Multidimensional:

- **Tablas Fact:** contienen los valores de las medidas de negocios, por ejemplo: ventas promedio en dólares, número de unidades vendidas, etc.
- **Tablas Lock_up:** contienen el detalle de los valores que se encuentran asociados a la tabla Fact.

Esquemas DW: la colección de tablas en el DW se conoce como Esquema. Los esquemas caen dentro de dos categorías básicas: esquemas estrellas y esquemas snowflake.

Esquema de Estrella

En general, el modelo multidimensional también se conoce con el nombre de esquema estrella, pues su estructura base es similar: una tabla central y un conjunto de tablas que la atienden radialmente.

El esquema estrella deriva su nombre del hecho que su diagrama forma una estrella, con puntos radiales desde el centro. El centro de la estrella consiste de una o más tablas fact, y las puntas de la estrella son las tablas lock_up. Este modelo entonces, resulta ser asimétrico, pues hay una tabla dominante en el centro con varias conexiones a las otras tablas. Las tablas lock-up tienen sólo la conexión a la tabla fact y ninguna más.

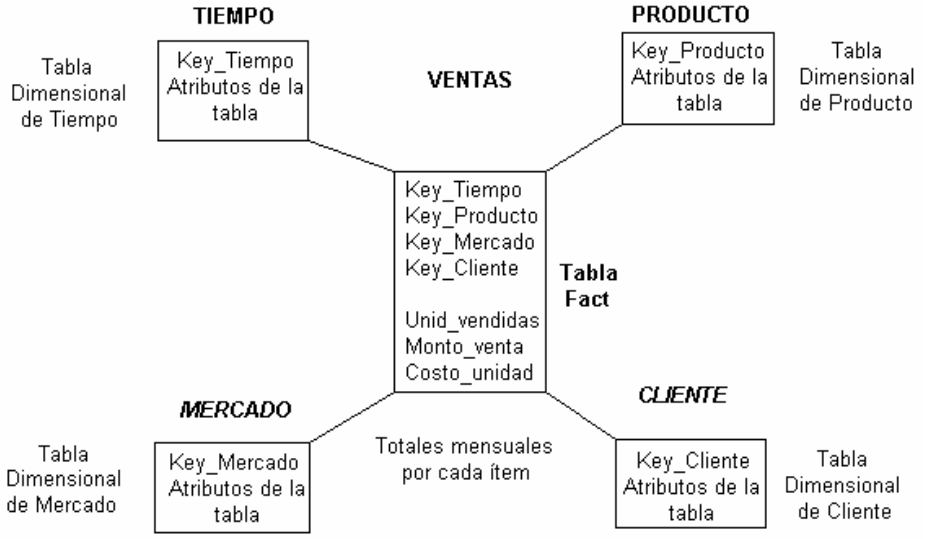


Ilustración 12-9. Ejemplo de un esquema estrella.

Esquema Snowflake

La diferencia del esquema snowflake comparado con el esquema estrella, está en la estructura de las tablas lock_up: las tablas lock_up en el esquema snowflake están normalizadas. Cada tabla lock_up contiene sólo el nivel que es clave primaria en la tabla y la foreign key de su parentesco del nivel más cercano del diagrama.

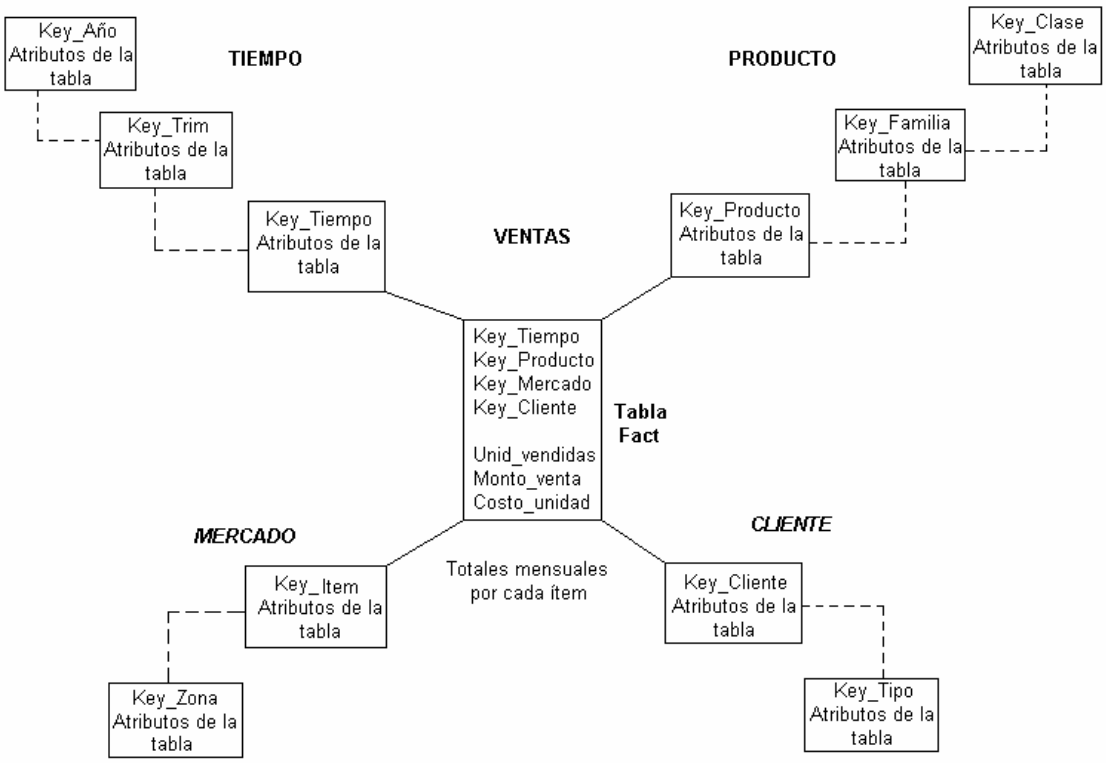


Ilustración 12-10. Esquema Snowflake.

Tabla Fact o de Hechos

Es la tabla central en un esquema dimensional. Es en ella donde se almacenan las mediciones numéricas del negocio. Estas medidas se hacen sobre el grano, o unidad básica de la tabla.

El grano o la granularidad de la tabla queda determinada por el nivel de detalle que se almacenará en la tabla. Por ejemplo, para el caso de producto, mercado y tiempo, el grano puede ser la cantidad de madera vendida 'mensualmente'. El grano revierte las unidades atómicas en el esquema dimensional.

Cada medida es tomada de la intersección de las dimensiones que la definen. Idealmente está compuesta por valores numéricos, continuamente evaluados y aditivos. La razón de estas características es que así se facilita que los miles de registros que involucran una consulta sean comprimidos en unas pocas líneas en un set de respuesta.

La clave de la tabla fact recibe el nombre de clave compuesta o concatenada debido a que se forma de la composición (o concatenación) de las llaves primarias de las tablas dimensionales a las que está unidad.

Así entonces, se distinguen dos tipos de columnas en una tabla fact: columnas fact y columnas key.

Donde la columna fact es la que almacena alguna medida de negocio y una columna key forma parte de la clave compuesta de la tabla.

Tablas Lock-up o Dimensionales

Estas tablas son las que se conectan a la tabla fact, son las que alimentan a la tabla fact. Una tabla lock_up almacena un conjunto de valores que están relacionados a una dimensión particular. Tablas lock_up no contienen hechos, en su lugar los valores en las tablas lock_up son los elementos que determinan la estructura de las dimensiones. Así entonces, en ellas existe el detalle de los valores de la dimensión respectiva.

Una tabla lock_up está compuesta de una primary key que identifica unívocamente una fila en la tabla junto con un conjunto de atributos, y dependiendo del diseño del modelo multidimensional puede existir una foreign key que determina su relación con otra tabla lock_up.

Para decidir si un campo de datos es un atributo o un hecho se analiza la variación de la medida a través del tiempo. Si varía continuamente implicaría tomarlo como un hecho, caso contrario será un atributo.

Los atributos dimensionales son un rol determinante en un DDW. Ellos son la fuente de todas las necesidades que debieran cubrirse. Esto significa que la base de datos será tan buena como lo sean los atributos dimensionales, mientras más descriptivos, manejables y de buena calidad, mejor será el DDW.

Pasos básicos del modelamiento multidimensional

1. Decidir cuáles serán los procesos de negocios a modelar, basándose en el conocimiento de éstos y de los datos disponibles. Ejemplo: Gastos realizados por cada mercado para cada ítem a nivel mensual. Productos vendidos por cada mercado según el precio en cada mes.

2. Decidir el Grano de la tabla Fact de cada proceso de negocio. Ejemplo: Producto x mercado x tiempo. En este punto se debe tener especial cuidado con la magnitud de la base de datos, con la información que se tiene y con las preguntas que se quiere responder. El grano decidirá las dimensiones del DDW. Cada dimensión debe tener el grano más pequeño que se pueda, puesto que las preguntas que se realicen necesitan, cortar la base en caminos precisos (aunque las preguntas no lo pidan explícitamente).

3. Decidir las dimensiones a través del grano. Las dimensiones presentes en la mayoría de los DDW son: tiempo, mercado, producto, cliente. Un grano bien elegido determina la dimensionalidad primaria de la tabla fact. Es posible usualmente agregar dimensiones adicionales al grano básico de la tabla fact, donde estas dimensiones adicionales toman un solo valor para cada combinación de las dimensiones primarias. Si se reconoce que una dimensión adicional deseada viola el grano por causar registros adicionales a los generados, entonces el grano debe ser revisado para acomodar esta dimensión adicional.

4. Elegir las mediciones del negocio para la tabla fact. Se deben establecer los ítems que quedarán determinados por la clave compuesta de la tabla fact.

Profundizaciones de diseño

La dimensión tiempo.

Virtualmente se garantiza que cada DDW tendrá una tabla dimensional de tiempo, debido a la perspectiva de almacenamiento histórica de la información. Usualmente es la primera dimensión en definirse, con el objeto de establecer un orden, ya que la inserción de datos en la base de datos multidimensional se hace por intervalos de tiempo, lo cual asegura un orden implícito.

Dimensiones que varían lentamente en el tiempo

Son aquellas dimensiones que se mantienen "casi" constantes en el tiempo y que pueden preservar la estructura dimensional independiente del tiempo, con sólo agregados menores relativos para capturar la naturaleza cambiante del tiempo.

Cuando se encuentra una de estas dimensiones se está haciendo una de las siguientes fundamentales tres elecciones. Cada elección resulta en un diferente grado de seguimiento sobre el tiempo:

- Tipo 1: Sobrescribir el viejo valor en el registro dimensional y por lo tanto perder la capacidad de seguir la vieja historia.
- Tipo 2: Crear un registro dimensional adicional (con una nueva llave) que permita registrar el cambio presentado por el valor del atributo. De esta forma permanecerían en la base tanto el antiguo como el nuevo valor del registro con lo cual es posible segmentar la historia de la ocurrencia.
- Tipo 3: Crear un campo "actual" nuevo en el registro dimensional original el cual almacene el valor del nuevo atributo, manteniendo el atributo original también. Cada vez que haya un nuevo cambio en el atributo, se modifica el campo "actual" solamente. No se mantiene un registro histórico de los cambios intermedios.

Niveles

Un nivel representa un nivel particular de agregación dentro de una dimensión; cada nivel sobre el nivel base representa la sumarización total de los datos desde el nivel inferior. Por ejemplo: consideremos una dimensión Tiempo con tres niveles: Mes, Semestre, Año. El nivel Mes representa el nivel base, el nivel Semestre representa la sumarización de los totales por Mes y el nivel Año representa la sumarización de los totales para los Semestres.

Agregar niveles de sumarización otorga flexibilidad adicional a usuarios finales de aplicaciones EIS/ DSS para analizar los datos.

Sobre Jerarquías

A nivel de dimensiones es posible definir jerarquías, las cuales son grupos de atributos que siguen un orden preestablecido.

Una jerarquía implica una organización de niveles dentro de una dimensión, con cada nivel representando el total agregado de los datos del nivel inferior. Las jerarquías definen cómo los datos son sumarizados desde los niveles más bajos hacia los más altos. Una dimensión típica soporta una o más jerarquías naturales. Una jerarquía puede pero no exige contener todos los valores existentes en la dimensión.

Se debe evitar caer en la tentación de convertir en tablas dimensionales separadas cada una de las relaciones muchos-a-uno presentes en las jerarquías. Esta descomposición es irrelevante en el planeamiento del espacio ocupado en disco y sólo dificulta el entendimiento de la estructura para el usuario final, además de destruir el desempeño del browsing.

12.5 ANÁLISIS DEL COMPORTAMIENTO DE LAS INGENIERÍAS DE LA FES ARAGÓN. CASO PRÁCTICO.

En la Facultad de Estudios Superiores Aragón de la UNAM se imparten las ingenierías de:

- Ingeniería Civil.
- Ingeniería Eléctrica y Electrónica.
- Ingeniería en Computación.
- Ingeniería Industrial.
- Ingeniería Mecánica.
- Ingeniería Mecánica Eléctrica.

Se desea saber el comportamiento de estas ingenierías con respecto al género.

Usando el Sistema OLAP del Sistema Dinámico de Estadísticas universitarias se obtuvo un filtro de las carreras y de las generación de 1980 al 2007 mostrados en la siguiente tabla 12-1.

Generación	Carrera	Género	Tipo de Ingreso	Población
1980	Ingeniería Civil	Hombres	Primer Ingreso	167
1980	Ingeniería Civil	Hombres	Reingreso	603
1980	Ingeniería Civil	Mujeres	Primer Ingreso	8
1980	Ingeniería Civil	Mujeres	Reingreso	23
1980	Ingeniería en Computación	Hombres	Primer Ingreso	0
1980	Ingeniería en Computación	Hombres	Reingreso	2
1980	Ingeniería en Computación	Mujeres	Primer Ingreso	0
1980	Ingeniería en Computación	Mujeres	Reingreso	1
1980	Ingeniería Mecánica Eléctrica	Hombres	Primer Ingreso	301
1980	Ingeniería Mecánica Eléctrica	Hombres	Reingreso	900
1980	Ingeniería Mecánica Eléctrica	Mujeres	Primer Ingreso	10
1980	Ingeniería Mecánica Eléctrica	Mujeres	Reingreso	17
1981	Ingeniería Civil	Hombres	Primer Ingreso	177
1981	Ingeniería Civil	Hombres	Reingreso	675
1981	Ingeniería Civil	Mujeres	Primer Ingreso	5
1981	Ingeniería Civil	Mujeres	Reingreso	24
1981	Ingeniería en Computación	Hombres	Primer Ingreso	46
1981	Ingeniería en Computación	Hombres	Reingreso	2
1981	Ingeniería en Computación	Mujeres	Primer Ingreso	10
1981	Ingeniería en Computación	Mujeres	Reingreso	1
1981	Ingeniería Mecánica Eléctrica	Hombres	Primer Ingreso	302
1981	Ingeniería Mecánica Eléctrica	Hombres	Reingreso	1074
1981	Ingeniería Mecánica Eléctrica	Mujeres	Primer Ingreso	11
1981	Ingeniería Mecánica Eléctrica	Mujeres	Reingreso	23
1982	Ingeniería Civil	Hombres	Primer Ingreso	163
1982	Ingeniería Civil	Hombres	Reingreso	714
1982	Ingeniería Civil	Mujeres	Primer Ingreso	3
1982	Ingeniería Civil	Mujeres	Reingreso	24
1982	Ingeniería en Computación	Hombres	Primer Ingreso	99
1982	Ingeniería en Computación	Hombres	Reingreso	43
1982	Ingeniería en Computación	Mujeres	Primer Ingreso	30
1982	Ingeniería en Computación	Mujeres	Reingreso	10
1982	Ingeniería Mecánica Eléctrica	Hombres	Primer Ingreso	343
1982	Ingeniería Mecánica Eléctrica	Hombres	Reingreso	1192
1982	Ingeniería Mecánica Eléctrica	Mujeres	Primer Ingreso	12
1982	Ingeniería Mecánica Eléctrica	Mujeres	Reingreso	32
1983	Ingeniería Civil	Hombres	Primer Ingreso	172
1983	Ingeniería Civil	Hombres	Reingreso	689
1983	Ingeniería Civil	Mujeres	Primer Ingreso	7
1983	Ingeniería Civil	Mujeres	Reingreso	24
1983	Ingeniería en Computación	Hombres	Primer Ingreso	140
1983	Ingeniería en Computación	Hombres	Reingreso	126
1983	Ingeniería en Computación	Mujeres	Primer Ingreso	33
1983	Ingeniería en Computación	Mujeres	Reingreso	32
1983	Ingeniería Mecánica Eléctrica	Hombres	Primer Ingreso	332
1983	Ingeniería Mecánica Eléctrica	Hombres	Reingreso	1257
1983	Ingeniería Mecánica Eléctrica	Mujeres	Primer Ingreso	15
1983	Ingeniería Mecánica Eléctrica	Mujeres	Reingreso	41
1984	Ingeniería Civil	Hombres	Primer Ingreso	162
1984	Ingeniería Civil	Hombres	Reingreso	660
1984	Ingeniería Civil	Mujeres	Primer Ingreso	12
1984	Ingeniería Civil	Mujeres	Reingreso	22
1984	Ingeniería en Computación	Hombres	Primer Ingreso	179

1984	Ingeniería en Computación	Hombres	Reingreso	238
1984	Ingeniería en Computación	Mujeres	Primer Ingreso	50
1984	Ingeniería en Computación	Mujeres	Reingreso	60
1984	Ingeniería Mecánica Eléctrica	Hombres	Primer Ingreso	329
1984	Ingeniería Mecánica Eléctrica	Hombres	Reingreso	1233
1984	Ingeniería Mecánica Eléctrica	Mujeres	Primer Ingreso	16
1984	Ingeniería Mecánica Eléctrica	Mujeres	Reingreso	46
1985	Ingeniería Civil	Hombres	Primer Ingreso	161
1985	Ingeniería Civil	Hombres	Reingreso	614
1985	Ingeniería Civil	Mujeres	Primer Ingreso	23
1985	Ingeniería Civil	Mujeres	Reingreso	25
1985	Ingeniería en Computación	Hombres	Primer Ingreso	175
1985	Ingeniería en Computación	Hombres	Reingreso	347
1985	Ingeniería en Computación	Mujeres	Primer Ingreso	47
1985	Ingeniería en Computación	Mujeres	Reingreso	94
1985	Ingeniería Mecánica Eléctrica	Hombres	Primer Ingreso	323
1985	Ingeniería Mecánica Eléctrica	Hombres	Reingreso	1175
1985	Ingeniería Mecánica Eléctrica	Mujeres	Primer Ingreso	19
1985	Ingeniería Mecánica Eléctrica	Mujeres	Reingreso	50
1986	Ingeniería Civil	Hombres	Primer Ingreso	174
1986	Ingeniería Civil	Hombres	Reingreso	606
1986	Ingeniería Civil	Mujeres	Primer Ingreso	14
1986	Ingeniería Civil	Mujeres	Reingreso	37
1986	Ingeniería en Computación	Hombres	Primer Ingreso	172
1986	Ingeniería en Computación	Hombres	Reingreso	457
1986	Ingeniería en Computación	Mujeres	Primer Ingreso	45
1986	Ingeniería en Computación	Mujeres	Reingreso	132
1986	Ingeniería Mecánica Eléctrica	Hombres	Primer Ingreso	319
1986	Ingeniería Mecánica Eléctrica	Hombres	Reingreso	1186
1986	Ingeniería Mecánica Eléctrica	Mujeres	Primer Ingreso	17
1986	Ingeniería Mecánica Eléctrica	Mujeres	Reingreso	59
1987	Ingeniería Civil	Hombres	Primer Ingreso	164
1987	Ingeniería Civil	Hombres	Reingreso	604
1987	Ingeniería Civil	Mujeres	Primer Ingreso	11
1987	Ingeniería Civil	Mujeres	Reingreso	42
1987	Ingeniería en Computación	Hombres	Primer Ingreso	171
1987	Ingeniería en Computación	Hombres	Reingreso	551
1987	Ingeniería en Computación	Mujeres	Primer Ingreso	50
1987	Ingeniería en Computación	Mujeres	Reingreso	150
1987	Ingeniería Mecánica Eléctrica	Hombres	Primer Ingreso	342
1987	Ingeniería Mecánica Eléctrica	Hombres	Reingreso	1166
1987	Ingeniería Mecánica Eléctrica	Mujeres	Primer Ingreso	16
1987	Ingeniería Mecánica Eléctrica	Mujeres	Reingreso	56
1988	Ingeniería Civil	Hombres	Primer Ingreso	149
1988	Ingeniería Civil	Hombres	Reingreso	573
1988	Ingeniería Civil	Mujeres	Primer Ingreso	7
1988	Ingeniería Civil	Mujeres	Reingreso	44
1988	Ingeniería en Computación	Hombres	Primer Ingreso	161
1988	Ingeniería en Computación	Hombres	Reingreso	619
1988	Ingeniería en Computación	Mujeres	Primer Ingreso	60
1988	Ingeniería en Computación	Mujeres	Reingreso	165
1988	Ingeniería Mecánica Eléctrica	Hombres	Primer Ingreso	391
1988	Ingeniería Mecánica Eléctrica	Hombres	Reingreso	1125
1988	Ingeniería Mecánica Eléctrica	Mujeres	Primer Ingreso	13
1988	Ingeniería Mecánica Eléctrica	Mujeres	Reingreso	59
1989	Ingeniería Civil	Hombres	Primer Ingreso	128
1989	Ingeniería Civil	Hombres	Reingreso	542
1989	Ingeniería Civil	Mujeres	Primer Ingreso	10
1989	Ingeniería Civil	Mujeres	Reingreso	35
1989	Ingeniería en Computación	Hombres	Primer Ingreso	201
1989	Ingeniería en Computación	Hombres	Reingreso	648
1989	Ingeniería en Computación	Mujeres	Primer Ingreso	67
1989	Ingeniería en Computación	Mujeres	Reingreso	190
1989	Ingeniería Mecánica Eléctrica	Hombres	Primer Ingreso	426
1989	Ingeniería Mecánica Eléctrica	Hombres	Reingreso	1263
1989	Ingeniería Mecánica Eléctrica	Mujeres	Primer Ingreso	21
1989	Ingeniería Mecánica Eléctrica	Mujeres	Reingreso	60
1990	Ingeniería Civil	Hombres	Primer Ingreso	128
1990	Ingeniería Civil	Hombres	Reingreso	490
1990	Ingeniería Civil	Mujeres	Primer Ingreso	12
1990	Ingeniería Civil	Mujeres	Reingreso	33
1990	Ingeniería en Computación	Hombres	Primer Ingreso	192
1990	Ingeniería en Computación	Hombres	Reingreso	700
1990	Ingeniería en Computación	Mujeres	Primer Ingreso	82
1990	Ingeniería en Computación	Mujeres	Reingreso	204
1990	Ingeniería Mecánica Eléctrica	Hombres	Primer Ingreso	428
1990	Ingeniería Mecánica Eléctrica	Hombres	Reingreso	1332
1990	Ingeniería Mecánica Eléctrica	Mujeres	Primer Ingreso	27
1990	Ingeniería Mecánica Eléctrica	Mujeres	Reingreso	68
1991	Ingeniería Civil	Hombres	Primer Ingreso	99
1991	Ingeniería Civil	Hombres	Reingreso	475
1991	Ingeniería Civil	Mujeres	Primer Ingreso	17
1991	Ingeniería Civil	Mujeres	Reingreso	32
1991	Ingeniería en Computación	Hombres	Primer Ingreso	185
1991	Ingeniería en Computación	Hombres	Reingreso	691
1991	Ingeniería en Computación	Mujeres	Primer Ingreso	81
1991	Ingeniería en Computación	Mujeres	Reingreso	232
1991	Ingeniería Mecánica Eléctrica	Hombres	Primer Ingreso	421
1991	Ingeniería Mecánica Eléctrica	Hombres	Reingreso	1384
1991	Ingeniería Mecánica Eléctrica	Mujeres	Primer Ingreso	33
1991	Ingeniería Mecánica Eléctrica	Mujeres	Reingreso	75

1992	Ingeniería Civil	Hombres	Primer Ingreso	138
1992	Ingeniería Civil	Hombres	Reingreso	448
1992	Ingeniería Civil	Mujeres	Primer Ingreso	20
1992	Ingeniería Civil	Mujeres	Reingreso	35
1992	Ingeniería en Computación	Hombres	Primer Ingreso	199
1992	Ingeniería en Computación	Hombres	Reingreso	689
1992	Ingeniería en Computación	Mujeres	Primer Ingreso	88
1992	Ingeniería en Computación	Mujeres	Reingreso	248
1992	Ingeniería Mecánica Eléctrica	Hombres	Primer Ingreso	421
1992	Ingeniería Mecánica Eléctrica	Hombres	Reingreso	1407
1992	Ingeniería Mecánica Eléctrica	Mujeres	Primer Ingreso	40
1992	Ingeniería Mecánica Eléctrica	Mujeres	Reingreso	86
1993	Ingeniería Civil	Hombres	Primer Ingreso	153
1993	Ingeniería Civil	Hombres	Reingreso	443
1993	Ingeniería Civil	Mujeres	Primer Ingreso	10
1993	Ingeniería Civil	Mujeres	Reingreso	41
1993	Ingeniería en Computación	Hombres	Primer Ingreso	180
1993	Ingeniería en Computación	Hombres	Reingreso	716
1993	Ingeniería en Computación	Mujeres	Primer Ingreso	87
1993	Ingeniería en Computación	Mujeres	Reingreso	279
1993	Ingeniería Mecánica Eléctrica	Hombres	Primer Ingreso	418
1993	Ingeniería Mecánica Eléctrica	Hombres	Reingreso	1496
1993	Ingeniería Mecánica Eléctrica	Mujeres	Primer Ingreso	35
1993	Ingeniería Mecánica Eléctrica	Mujeres	Reingreso	92
1994	Ingeniería Civil	Hombres	Primer Ingreso	132
1994	Ingeniería Civil	Hombres	Reingreso	482
1994	Ingeniería Civil	Mujeres	Primer Ingreso	12
1994	Ingeniería Civil	Mujeres	Reingreso	43
1994	Ingeniería en Computación	Hombres	Primer Ingreso	236
1994	Ingeniería en Computación	Hombres	Reingreso	730
1994	Ingeniería en Computación	Mujeres	Primer Ingreso	110
1994	Ingeniería en Computación	Mujeres	Reingreso	306
1994	Ingeniería Mecánica Eléctrica	Hombres	Primer Ingreso	406
1994	Ingeniería Mecánica Eléctrica	Hombres	Reingreso	1529
1994	Ingeniería Mecánica Eléctrica	Mujeres	Primer Ingreso	33
1994	Ingeniería Mecánica Eléctrica	Mujeres	Reingreso	106
1995	Ingeniería Civil	Hombres	Primer Ingreso	139
1995	Ingeniería Civil	Hombres	Reingreso	491
1995	Ingeniería Civil	Mujeres	Primer Ingreso	26
1995	Ingeniería Civil	Mujeres	Reingreso	46
1995	Ingeniería en Computación	Hombres	Primer Ingreso	180
1995	Ingeniería en Computación	Hombres	Reingreso	817
1995	Ingeniería en Computación	Mujeres	Primer Ingreso	73
1995	Ingeniería en Computación	Mujeres	Reingreso	356
1995	Ingeniería Mecánica Eléctrica	Hombres	Primer Ingreso	445
1995	Ingeniería Mecánica Eléctrica	Hombres	Reingreso	1592
1995	Ingeniería Mecánica Eléctrica	Mujeres	Primer Ingreso	27
1995	Ingeniería Mecánica Eléctrica	Mujeres	Reingreso	121
1996	Ingeniería Civil	Hombres	Primer Ingreso	152
1996	Ingeniería Civil	Hombres	Reingreso	476
1996	Ingeniería Civil	Mujeres	Primer Ingreso	23
1996	Ingeniería Civil	Mujeres	Reingreso	56
1996	Ingeniería en Computación	Hombres	Primer Ingreso	180
1996	Ingeniería en Computación	Hombres	Reingreso	838
1996	Ingeniería en Computación	Mujeres	Primer Ingreso	58
1996	Ingeniería en Computación	Mujeres	Reingreso	339
1996	Ingeniería Mecánica Eléctrica	Hombres	Primer Ingreso	464
1996	Ingeniería Mecánica Eléctrica	Hombres	Reingreso	1624
1996	Ingeniería Mecánica Eléctrica	Mujeres	Primer Ingreso	38
1996	Ingeniería Mecánica Eléctrica	Mujeres	Reingreso	122
1997	Ingeniería Civil	Hombres	Primer Ingreso	170
1997	Ingeniería Civil	Hombres	Reingreso	488
1997	Ingeniería Civil	Mujeres	Primer Ingreso	25
1997	Ingeniería Civil	Mujeres	Reingreso	54
1997	Ingeniería en Computación	Hombres	Primer Ingreso	192
1997	Ingeniería en Computación	Hombres	Reingreso	854
1997	Ingeniería en Computación	Mujeres	Primer Ingreso	62
1997	Ingeniería en Computación	Mujeres	Reingreso	337
1997	Ingeniería Mecánica Eléctrica	Hombres	Primer Ingreso	406
1997	Ingeniería Mecánica Eléctrica	Hombres	Reingreso	1669
1997	Ingeniería Mecánica Eléctrica	Mujeres	Primer Ingreso	30
1997	Ingeniería Mecánica Eléctrica	Mujeres	Reingreso	125
1998	Ingeniería Civil	Hombres	Primer Ingreso	155
1998	Ingeniería Civil	Hombres	Reingreso	484
1998	Ingeniería Civil	Mujeres	Primer Ingreso	20
1998	Ingeniería Civil	Mujeres	Reingreso	56
1998	Ingeniería en Computación	Hombres	Primer Ingreso	194
1998	Ingeniería en Computación	Hombres	Reingreso	846
1998	Ingeniería en Computación	Mujeres	Primer Ingreso	58
1998	Ingeniería en Computación	Mujeres	Reingreso	312
1998	Ingeniería Mecánica Eléctrica	Hombres	Primer Ingreso	427
1998	Ingeniería Mecánica Eléctrica	Hombres	Reingreso	1685
1998	Ingeniería Mecánica Eléctrica	Mujeres	Primer Ingreso	20
1998	Ingeniería Mecánica Eléctrica	Mujeres	Reingreso	116
1999	Ingeniería Civil	Hombres	Primer Ingreso	160
1999	Ingeniería Civil	Hombres	Reingreso	486
1999	Ingeniería Civil	Mujeres	Primer Ingreso	25
1999	Ingeniería Civil	Mujeres	Reingreso	56
1999	Ingeniería Eléctrica y Electrónica	Hombres	Primer Ingreso	0
1999	Ingeniería Eléctrica y Electrónica	Hombres	Reingreso	0
1999	Ingeniería Eléctrica y Electrónica	Mujeres	Primer Ingreso	0

1999	Ingeniería Eléctrica y Electrónica	Mujeres	Reingreso	0
1999	Ingeniería en Computación	Hombres	Primer Ingreso	172
1999	Ingeniería en Computación	Hombres	Reingreso	847
1999	Ingeniería en Computación	Mujeres	Primer Ingreso	63
1999	Ingeniería en Computación	Mujeres	Reingreso	269
1999	Ingeniería Industrial	Hombres	Primer Ingreso	0
1999	Ingeniería Industrial	Hombres	Reingreso	0
1999	Ingeniería Industrial	Mujeres	Primer Ingreso	0
1999	Ingeniería Industrial	Mujeres	Reingreso	0
1999	Ingeniería Mecánica	Hombres	Primer Ingreso	0
1999	Ingeniería Mecánica	Hombres	Reingreso	0
1999	Ingeniería Mecánica	Mujeres	Primer Ingreso	0
1999	Ingeniería Mecánica	Mujeres	Reingreso	0
1999	Ingeniería Mecánica Eléctrica	Hombres	Primer Ingreso	418
1999	Ingeniería Mecánica Eléctrica	Hombres	Reingreso	1642
1999	Ingeniería Mecánica Eléctrica	Mujeres	Primer Ingreso	18
1999	Ingeniería Mecánica Eléctrica	Mujeres	Reingreso	105
2000	Ingeniería Civil	Hombres	Primer Ingreso	93
2000	Ingeniería Civil	Hombres	Reingreso	430
2000	Ingeniería Civil	Mujeres	Primer Ingreso	5
2000	Ingeniería Civil	Mujeres	Reingreso	54
2000	Ingeniería Eléctrica y Electrónica	Hombres	Primer Ingreso	0
2000	Ingeniería Eléctrica y Electrónica	Hombres	Reingreso	0
2000	Ingeniería Eléctrica y Electrónica	Mujeres	Primer Ingreso	0
2000	Ingeniería Eléctrica y Electrónica	Mujeres	Reingreso	0
2000	Ingeniería en Computación	Hombres	Primer Ingreso	162
2000	Ingeniería en Computación	Hombres	Reingreso	750
2000	Ingeniería en Computación	Mujeres	Primer Ingreso	62
2000	Ingeniería en Computación	Mujeres	Reingreso	238
2000	Ingeniería Industrial	Hombres	Primer Ingreso	0
2000	Ingeniería Industrial	Hombres	Reingreso	0
2000	Ingeniería Industrial	Mujeres	Primer Ingreso	0
2000	Ingeniería Industrial	Mujeres	Reingreso	0
2000	Ingeniería Mecánica	Hombres	Primer Ingreso	0
2000	Ingeniería Mecánica	Hombres	Reingreso	0
2000	Ingeniería Mecánica	Mujeres	Primer Ingreso	0
2000	Ingeniería Mecánica	Mujeres	Reingreso	0
2000	Ingeniería Mecánica Eléctrica	Hombres	Primer Ingreso	253
2000	Ingeniería Mecánica Eléctrica	Hombres	Reingreso	1449
2000	Ingeniería Mecánica Eléctrica	Mujeres	Primer Ingreso	21
2000	Ingeniería Mecánica Eléctrica	Mujeres	Reingreso	92
2001	Ingeniería Civil	Hombres	Primer Ingreso	88
2001	Ingeniería Civil	Hombres	Reingreso	395
2001	Ingeniería Civil	Mujeres	Primer Ingreso	10
2001	Ingeniería Civil	Mujeres	Reingreso	37
2001	Ingeniería Eléctrica y Electrónica	Hombres	Primer Ingreso	0
2001	Ingeniería Eléctrica y Electrónica	Hombres	Reingreso	0
2001	Ingeniería Eléctrica y Electrónica	Mujeres	Primer Ingreso	0
2001	Ingeniería Eléctrica y Electrónica	Mujeres	Reingreso	0
2001	Ingeniería en Computación	Hombres	Primer Ingreso	180
2001	Ingeniería en Computación	Hombres	Reingreso	677
2001	Ingeniería en Computación	Mujeres	Primer Ingreso	73
2001	Ingeniería en Computación	Mujeres	Reingreso	221
2001	Ingeniería Industrial	Hombres	Primer Ingreso	0
2001	Ingeniería Industrial	Hombres	Reingreso	0
2001	Ingeniería Industrial	Mujeres	Primer Ingreso	0
2001	Ingeniería Industrial	Mujeres	Reingreso	0
2001	Ingeniería Mecánica	Hombres	Primer Ingreso	0
2001	Ingeniería Mecánica	Hombres	Reingreso	0
2001	Ingeniería Mecánica	Mujeres	Primer Ingreso	0
2001	Ingeniería Mecánica	Mujeres	Reingreso	0
2001	Ingeniería Mecánica Eléctrica	Hombres	Primer Ingreso	288
2001	Ingeniería Mecánica Eléctrica	Hombres	Reingreso	1274
2001	Ingeniería Mecánica Eléctrica	Mujeres	Primer Ingreso	15
2001	Ingeniería Mecánica Eléctrica	Mujeres	Reingreso	95
2002	Ingeniería Civil	Hombres	Primer Ingreso	105
2002	Ingeniería Civil	Hombres	Reingreso	372
2002	Ingeniería Civil	Mujeres	Primer Ingreso	17
2002	Ingeniería Civil	Mujeres	Reingreso	39
2002	Ingeniería en Computación	Hombres	Primer Ingreso	186
2002	Ingeniería en Computación	Hombres	Reingreso	716
2002	Ingeniería en Computación	Mujeres	Primer Ingreso	62
2002	Ingeniería en Computación	Mujeres	Reingreso	239
2002	Ingeniería Mecánica Eléctrica	Hombres	Primer Ingreso	317
2002	Ingeniería Mecánica Eléctrica	Hombres	Reingreso	1200
2002	Ingeniería Mecánica Eléctrica	Mujeres	Primer Ingreso	21
2002	Ingeniería Mecánica Eléctrica	Mujeres	Reingreso	83
2003	Ingeniería Civil	Hombres	Primer Ingreso	97
2003	Ingeniería Civil	Hombres	Reingreso	340
2003	Ingeniería Civil	Mujeres	Primer Ingreso	15
2003	Ingeniería Civil	Mujeres	Reingreso	39
2003	Ingeniería en Computación	Hombres	Primer Ingreso	202
2003	Ingeniería en Computación	Hombres	Reingreso	696
2003	Ingeniería en Computación	Mujeres	Primer Ingreso	59
2003	Ingeniería en Computación	Mujeres	Reingreso	241
2003	Ingeniería Mecánica Eléctrica	Hombres	Primer Ingreso	266
2003	Ingeniería Mecánica Eléctrica	Hombres	Reingreso	1193
2003	Ingeniería Mecánica Eléctrica	Mujeres	Primer Ingreso	22
2003	Ingeniería Mecánica Eléctrica	Mujeres	Reingreso	80
2004	Ingeniería Civil	Hombres	Primer Ingreso	104
2004	Ingeniería Civil	Hombres	Reingreso	331

2004	Ingeniería Civil	Mujeres	Primer Ingreso	16
2004	Ingeniería Civil	Mujeres	Reingreso	43
2004	Ingeniería en Computación	Hombres	Primer Ingreso	207
2004	Ingeniería en Computación	Hombres	Reingreso	753
2004	Ingeniería en Computación	Mujeres	Primer Ingreso	44
2004	Ingeniería en Computación	Mujeres	Reingreso	257
2004	Ingeniería Mecánica Eléctrica	Hombres	Primer Ingreso	266
2004	Ingeniería Mecánica Eléctrica	Hombres	Reingreso	1156
2004	Ingeniería Mecánica Eléctrica	Mujeres	Primer Ingreso	23
2004	Ingeniería Mecánica Eléctrica	Mujeres	Reingreso	77
2005	Ingeniería Civil	Hombres	Primer Ingreso	111
2005	Ingeniería Civil	Hombres	Reingreso	332
2005	Ingeniería Civil	Mujeres	Primer Ingreso	11
2005	Ingeniería Civil	Mujeres	Reingreso	48
2005	Ingeniería en Computación	Hombres	Primer Ingreso	211
2005	Ingeniería en Computación	Hombres	Reingreso	786
2005	Ingeniería en Computación	Mujeres	Primer Ingreso	52
2005	Ingeniería en Computación	Mujeres	Reingreso	233
2005	Ingeniería Mecánica Eléctrica	Hombres	Primer Ingreso	252
2005	Ingeniería Mecánica Eléctrica	Hombres	Reingreso	1092
2005	Ingeniería Mecánica Eléctrica	Mujeres	Primer Ingreso	18
2005	Ingeniería Mecánica Eléctrica	Mujeres	Reingreso	81
2006	Ingeniería Civil	Hombres	Primer Ingreso	96
2006	Ingeniería Civil	Hombres	Reingreso	342
2006	Ingeniería Civil	Mujeres	Primer Ingreso	30
2006	Ingeniería Civil	Mujeres	Reingreso	47
2006	Ingeniería en Computación	Hombres	Primer Ingreso	231
2006	Ingeniería en Computación	Hombres	Reingreso	811
2006	Ingeniería en Computación	Mujeres	Primer Ingreso	49
2006	Ingeniería en Computación	Mujeres	Reingreso	223
2006	Ingeniería Mecánica Eléctrica	Hombres	Primer Ingreso	266
2006	Ingeniería Mecánica Eléctrica	Hombres	Reingreso	1110
2006	Ingeniería Mecánica Eléctrica	Mujeres	Primer Ingreso	19
2006	Ingeniería Mecánica Eléctrica	Mujeres	Reingreso	75
2007	Ingeniería Civil	Hombres	Primer Ingreso	118
2007	Ingeniería Civil	Hombres	Reingreso	354
2007	Ingeniería Civil	Mujeres	Primer Ingreso	13
2007	Ingeniería Civil	Mujeres	Reingreso	65
2007	Ingeniería en Computación	Hombres	Primer Ingreso	267
2007	Ingeniería en Computación	Hombres	Reingreso	857
2007	Ingeniería en Computación	Mujeres	Primer Ingreso	44
2007	Ingeniería en Computación	Mujeres	Reingreso	206
2007	Ingeniería Mecánica Eléctrica	Hombres	Primer Ingreso	280
2007	Ingeniería Mecánica Eléctrica	Hombres	Reingreso	1030
2007	Ingeniería Mecánica Eléctrica	Mujeres	Primer Ingreso	21
2007	Ingeniería Mecánica Eléctrica	Mujeres	Reingreso	73

Tabla 12-1. Población escolar de las carreras de ingeniería de la Facultad de Estudios Superiores Aragón 1980-2007.

Usando las la combinación y graficación mediante cubos OLAP de la información de la Tabla 12-1 se obtiene el comportamiento de la población por género de las ingenierías de la FES Aragón mostradas en la Ilustración 12-11.

Pudiendo destacar que la población masculina de la carrera de Ingeniería Mecánica Eléctrica ha visto un aumento de población hasta llegar a 1999 donde comienza su declive, sin ver un aumento en su población femenina.

Para la carrera de ingeniería en computación, la población masculina se ve con incremento sostenido con un ligero decaimiento en 1999 y siguiendo su crecimiento para los años posteriores.

En lo que respecta a la carrera de Ingeniería Civil, la población se encuentra en una caída sostenida estabilizándose en el 2003 y para el 2007 muestra signos de crecimiento.

La asociación con el paro estudiantil de 1999 en la UNAM es lo que se puede asociar con la curva en las diferentes carreras, pero no tuvo el mismo efecto en todas como se ve en la Ilustración 12-11.

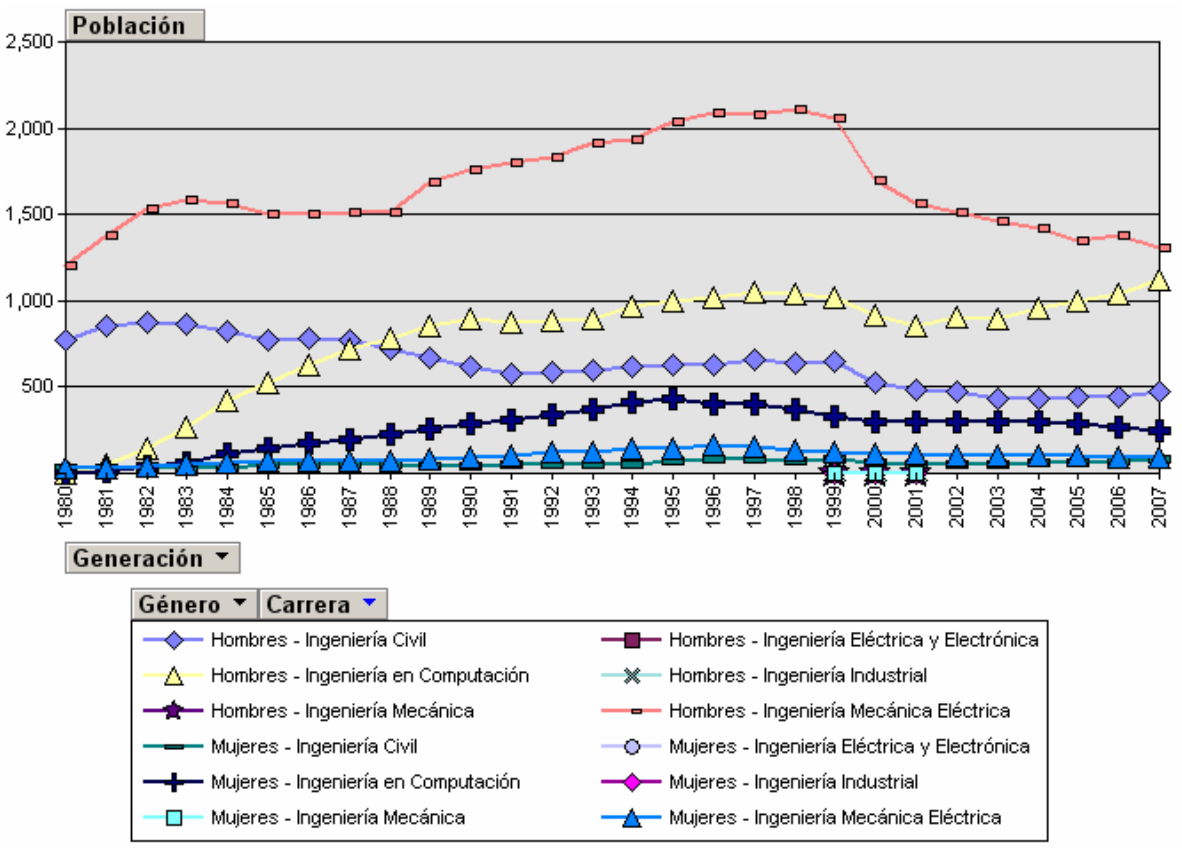


Ilustración 12-11. Comportamiento de la población escolar de ingenierías de la FES Aragón.

Capítulo 13

SISTEMA DE SEGUIMIENTO Y CONTROL DE ERRORES DEL MANEJADOR DE BASES DE DATOS SYBASE

Con el objetivo de ayudar, a un bajo costo, a disminuir el tiempo que se lleva encontrar una solución al ocurrir un error en el Sistema Manejador de Bases de Datos Relacional Sybase, se llevó a cabo la programación de una aplicación para documentar el error, darle seguimiento y tenerlo como referencia para un caso similar en el futuro.

Esta aplicación tiene considerado en su diseño y desarrollo el fin de aterrizar en un caso práctico la teoría y los conceptos de la administración de Bases de Datos vistos en los capítulos 1 al 12 de este documento, enfocándose principalmente en SQL del capítulo 3, el acceso a datos a través de la programación de clientes del capítulo 4, fundamentos de sistemas operativos capítulo 5, segregación de funciones del capítulo 8, entre otros conceptos.

13.1 PROBLEMÁTICA EN LA RESOLUCIÓN DE ERRORES EN EL MANEJADOR DE BASES DE DATOS SYBASE

Características del Manejador de Bases de Datos Sybase.

El Sistema Manejador de Bases de Datos Relacional Sybase en su versión "*Adaptive Server Enterprise*" es una potente base de datos que requiere del pago de licencias de uso. Estas licencias pueden ser adquiridas mediante algún distribuidor autorizado o realizar su compra desde su sitio en Internet www.sybase.com.

Tipos de soluciones

El costo de las licencias varía de acuerdo a la solución que sea requerida.

Sybase IQ	Motor de bases de datos altamente optimizado para inteligencia empresarial. Diseñado específicamente para entregar los resultados más rápido en solución de inteligencia empresarial analítica de misión crítica, almacenes de datos y generación de reportes. Sybase IQ combina velocidad y agilidad, con un bajo costo total de propiedad, lo que permite a las empresas llevar a cabo análisis de datos y generación de reportes antes impensables, imprácticos o costosos. La más reciente versión de Sybase IQ es la 12.6.

Adaptive Server Enterprise (ASE)	Motor de bases de datos insignia de la compañía Sybase. ASE es un sistema de gestión de datos, altamente escalable, de alto rendimiento, con soporte a grandes volúmenes de datos, transacciones y usuarios, y de bajo costo, que permite: almacenar datos de manera segura, tener acceso y procesar datos de manera inteligente, además de movilizar los datos. La versión más reciente de ASE es la 15.0.
Sybase Adaptive Server Anywhere (ASA)	Sistema administrador de bases de datos relacionales de alto rendimiento, que dentro de su funcionalidad incluye la gestión de transacciones, un optimizador de consultas auto-afinable, integridad referencial, procedimientos almacenados Java y SQL, triggers, bloqueo a nivel de registro, programación de eventos y recuperación automática. ASA es desarrollado por iAnywhere, subsidiaria de Sybase.

Tabla 13-1. Principales soluciones de bases de datos relacionadas ofertadas por Sybase.

Además del tipo de solución, el costo de las licencias cambia debido a que están relacionadas con el número de procesadores o núcleos que contenga cada procesador en la computadora en que se vaya a instalar.

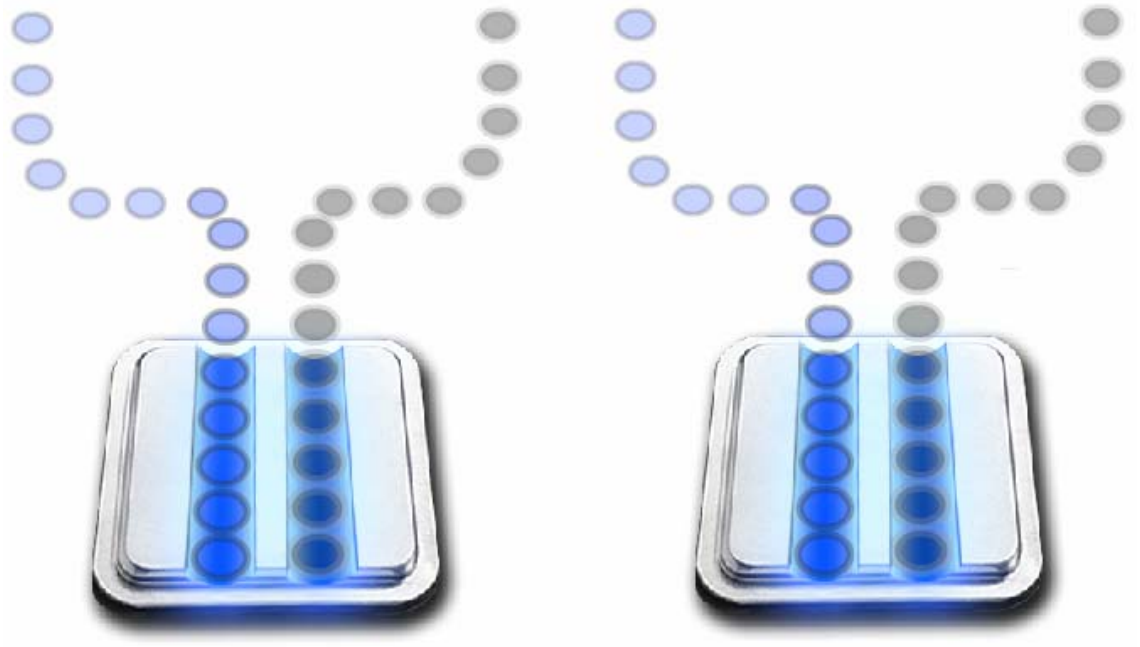


Ilustración 13-1. Si la computadora tiene 2 procesadores de doble núcleo, el costo sería lo de 4 licencias, esto por ser 4 núcleos al sumar los 2 de cada procesador.

Otro costo a considerar es el de las conexiones a la base de datos, debido a que mientras más conexiones se tengan, se requerirá de licencias que soporten esas conexiones, y desde luego, mientras más conexiones soporte mayor será el costo de la licencia.

Costo de las licencias

También el costo de las licencias aumenta en la selección del tipo de soporte técnico que se elija. A continuación se presenta algunos de los costos para la versión "ASE Enterprise Edition 12.5.4".

ASE Enterprise Edition 12.5.4 Idioma: Inglés Plataforma: Linux x86-64

Opciones de licencia:

Licencia de CPU	US $32,137.00 /CPU
Soporte Básico	US $ 6,426.00
Soporte Extendido	US $ 7,068.00
CPU en espera	US $ 8,035.00 /CPU
Soporte Básico	US $ 1,606.00
Soporte Extendido	US $ 1,768.00
Desarrollo y pruebas	US $ 9,643.00
Soporte Básico	US $ 1,928.00
Soporte Extendido	US $ 2,120.00

US $ = Precios en Dólares

Información de la Licencia:

Licencia de CPU - Usuarios limitados a la capacidad de usuarios que soporte el procesador. La cantidad suministrada en la compra así como por licencia deberá reflejar el número de procesadores por cada licencia requerida. No es requerido el comprar una licencia basándose en el número máximo de procesadores que se pueda configurar un sistema. En vez de esto, se puede licenciar únicamente los procesadores que están actualmente instalados en la(s) máquinas donde se quiera instalar el software y escalarlos cuando se incremente la capacidad de cómputo. El acceso a Internet esta incluido en la licencia del CPU.

CPU en espera - Permite la instalación de una copia de seguridad del programa licenciado bajo una licencia del CPU.

Desarrollo y pruebas - Con esta licencia el programa debe de ser usado únicamente para propósitos de pruebas y desarrollo sin un ambiente paralelo. Basado en la licencia de CPU. Puede ser instalada en un servidor, en vez de una estación de trabajo. La licencia de "desarrollo y pruebas" y la "licencia de CPU" son precios por cada procesador. El precio mostrado es por un único procesador. La licencia de "desarrollo y pruebas" está limitada a una configuración de 4 procesadores, a menos que se espejee un sistema de despliegue. Si se esta espejeando un sistema de despliegue, se deberá comprar licencias de "desarrollo y pruebas" para cuantos procesadores como se tenga en sistemas de despliegue.

Precios considerados de la tienda en línea de Sybase disponibles para las siguientes naciones

AU BE CA DK FR DE HK IT LU NL

NZ NO PT ES SE CH GB US

Para su compra en México hay que dirigirse a:

Sybase de México, S. de R.L. de C.V. Torre Altiva - 6to piso Blvd. Manuel Ávila Camacho # 138 Col. Lomas de Chapultepec México, D.F. C.P. 11000 Teléfono: +50938500

Fuente: http://eshop.sybase.com

Sistema de almacenamiento de información

Un sistema de almacenamiento de información requiere de al menos de 4 equipos.

Ilustración 13-2. Esquema de un sistema de almacenamiento de información usando licencias de Sybase.

- Equipo en producción para sistemas en producción necesita una licencia de CPU.
  - Se ejecutan las aplicaciones previamente verificando su funcionamiento y estabilidad en el equipo de pruebas.
  - Se almacena la información que se ocupa diariamente.
  - Se actualiza los datos cotidianamente en una empresa.
- Equipo para respaldos y copias de seguridad necesita una licencia de CPU en espera.
  - Se almacena la información de respaldo del equipo en producción.
- Equipo de desarrollo para sistemas en desarrollo necesita una licencia de desarrollo y pruebas.
  - Se diseñan sistemas.
  - Se programan las actualizaciones de los sistemas en producción.

- Se elaboran nuevos sistemas.

- Equipo de pruebas requiere de una licencia de desarrollo y pruebas.
  - Se prueban los sistemas desarrollados.
  - Se experimenta si los parches de seguridad no afectan al sistema en producción.
  - Se verifica si el instalar un nuevo software no afecta a los sistemas en producción.

Considerando que los cuatro equipos sean computadoras de un solo procesador y de un solo núcleo, el costo de las licencias, sin soporte, para un sistema de almacenamiento de información usando licencias de Sybase sería de:

Tipo de equipo	Tipo de licencia	Costo (en dólares)
Equipo en producción	Licencia de CPU	US $ 32,137.00
Equipo de respaldo	CPU en espera	US $ 8,035.00
Equipo de desarrollo	Desarrollo y pruebas	US $ 9,643.00
Equipo de pruebas	Desarrollo y pruebas	US $ 9,643.00
	Total	US $ 49,458.00

Tabla 13-2. Costo de un sistema de almacenamiento de información cotizando licencias de ASE Enterprise Edition 12.5.4

Manejo de errores

La bitácora de errores

Una vez instalado el Adaptive Server Enterprise[1] - ASE se crea un archivo en la ruta

```
/opt/sybase-11.9.2/logs/SQLSRV.log
```

Código 13-1. Ruta en la que se almacena el log de errores.

Donde **sybase-11.9.2** es la versión del ASE, mientras **SQLSRV** es el nombre que se le asignó al servidor de bases de datos a la hora de su instalación.

El archivo **SQLSRV.log** almacena la bitácora de errores que se van generando con el uso de la base de datos. La bitácora de errores es más conocida como log de errores.

```
00:00000:00000:2007/03/17 13:40:37.93 kernel  Using config area from primary master device.
00:00000:00000:2007/03/17 13:40:37.95 kernel  Configuration Error: Configuration file,
'/opt/sybase-11.9.2/SQLSRV.cfg', does not exist.
00:00000:00000:2007/03/17 13:40:37.95 kernel  Warning: A configuration file was not specified
and the default file '/opt/sybase-11.9.2/SQLSRV.cfg' does not exist. SQL Server creates the
default file with the default configuration.
00:00000:00000:2007/03/17 13:40:37.96 kernel  Using 1024 file descriptors.
```

Código 13-2. Breve contenido del log de errores.

En el log de errores se almacenará, entre otra información, los errores que deben ser informados al administrador de Sybase. Se almacenan sólo los errores que sean mayores de 16 en severidad. Donde el personal administrador de Sybase deberá monitorearlos y resolverlos.

[1] Para el presente trabajo se usa la versión 11.9.2 del Adaptive Server Enterprise (ASE).

Hay errores que provocan que no se pueda usar la base de datos. Errores como cuando se llena un dispositivo[2].

```
1> insert into almacena (x1, x2) values (2,3)
2> go 20000
```

Código 13-3. Ingreso de 20,000 registros en una tabla que saturan el dispositivo.

Cuando esto pasa, en el log de errores queda registrado cada intento en que se trata de ingresar nueva información a la tabla, registrando un error que indica que el dispositivo esta lleno esperando que se tenga espacio para continuar con la inserción de datos.

```
00:00000:00005:2007/03/27 20:08:50.37 server  1 task(s) are sleeping waiting for space to
become available in the log segment for database db1.
00:00000:00005:2007/03/27 20:09:50.96 server  1 task(s) are sleeping waiting for space to
become available in the log segment for database db1.
00:00000:00005:2007/03/27 20:10:51.55 server  1 task(s) are sleeping waiting for space to
become available in the log segment for database db1.
```

Código 13-4. Registro del log de errores en que indica que el servidor tiene una tarea esperando espacio en el segmento del log de transacciones para la base de datos db1.

Tiempo de respuesta en el manejo de errores

Debido a que en el log de errores se van almacenando los incidentes de las bases de datos, el archivo se va incrementando.

Pero cuando ocurre un error que necesite que el administrador realice una acción para que se siga otorgando el servicio. El administrador intentará descifrar lo que le dice el log de errores.

El explorar el log de errores tiene la dificultad de que un programa como el **`less`** de Linux, sólo permite la visibilidad de 80 líneas por página. Y si el log de errores almacena 80,000 líneas esto requerirá un gran esfuerzo para encontrar el error. Y si se requieren 5 segundos para examinar cada página entre 80 mil líneas serían 1,600 segundos o 26 minutos en tan sólo examinar el log de errores.

Costo al no documentar las soluciones a errores pasados

El log de errores, como es una bitácora, seguirá almacenando las incidencias que vayan ocurriendo y no borrará los errores que hayan sido resueltos. Suponiendo que en la línea 40,000 se encuentre un error que no se sabía que ya fue resuelto con anterioridad y que se suponga que este error es el que ocasiona el mal funcionamiento en la base. Al comprobar que ese no es el error que ocasiona el problema, generará una gran perdida de tiempo en descubrirlo, perdiendo al menos 20 minutos.

Veinte minutos que sumados a los 26 de la inspección del log de errores, generan prácticamente una hora en la caída de servicio. Lo que hace ver que es necesario un mecanismo que lleve a cabo el seguimiento y el control de los errores pasados.

---

[2] Un dispositivo en Sybase es el espacio reservado para almacenar bases de datos.

13.2 USO DEL SOFTWARE LIBRE PARA DESARROLLAR UN SISTEMA DE SEGUIMIENTO Y CONTROL DE ERRORES DEL MANEJADOR DE BASES DE DATOS SYBASE

Como el objetivo es desarrollar una aplicación que ayude a encontrar los errores a un bajo costo, el usar software libre es la alternativa. Por lo que hay que pensar en qué Sistema Operativo, qué lenguaje de programación, el sistema de almacenamiento de información y de qué interfase de usuario se va a usar.

Sistema operativo

Se eligió como Sistema Operativo Community Enterprise Operating System (CentOS) porque es un clon a nivel binario de la distribución Red Hat™ Enterprise Linux (RHEL), compilado por voluntarios a partir del código fuente liberado por Red Hat™. Además que CentOS, al igual que el RHEL, las actualizaciones de este Sistema Operativo se lleva a cabo cada 3 años y no cada 6 u 8 meses como lo hacen otros sabores de Linux, por lo que ayuda a que se pueda programar la migración de los sistemas en producción.

CentOS está enfocado para el uso en servidores en producción. Tiene el Kernel 2.6.18, soporta las plataformas i386 y x86_64:

- Procesadores i386:
    - AMD (K6, K7, Thunderbird, Athlon, Athlon XP, Sempron).
    - Pentium (Classic, Pro, II, III, 4, Celeron, M, Xeon).
    - VIA (C3, Eden, Luke, C7).
- Procesadores x86_64:
    - AMD 64 (Athlon 64, Opteron).
    - Intel Pentium (Xeon EM64T).

Infraestructura del kernel de Linux 2.6:

- Planificador genérico de CPU's lógicas: gestiona las CPU multi-núcleo e hyperthreaded.
- Máquina virtual con asignación inversa basada en objetos: rendimiento mejorado en sistemas con restricciones de memoria.
- Optimización del algoritmo SMP para estructuras de datos del sistema operativo.
- Compatibilidad con SMP y NUMA[3]: rendimiento y escalabilidad en grandes servidores.
- Mitigación de la interrupción de la red (NAPI[4]): alto rendimiento para grandes cargas de red.
- CentOS incorpora escalabilidad y rendimiento de los subsistemas de almacenamiento de datos con el del sistema de archivos.
- Rendimiento Ext3, las reservas de bloque y directorios de árboles "hash[5]" mejoran el rendimiento de las operaciones de lectura y escritura, E/S y exploración de directorios. Admite

[3] NUMA- Non-Uniform Memory Access. Acceso a memoria no uniforme.

[4] NAPI - Network Application Programming Interface. Interfase programada de aplicación de red.

la extensión dinámica del sistema de archivos y tamaños de archivo de hasta 8 Tb, esto es importante porque en sistemas operativos como Windows sólo se maneja hasta 2 Gb.

- Gestión de LUNs[6] de almacenamiento hace posible configurar subsistemas de almacenamiento mucho mayores.
- Automontaje: la incorporación de AutoFSv4 proporciona un sofisticado control de acceso a los dispositivos, con características tales como montajes examinables y servidores replicados.
- Compatibilidad con el almacenamiento en disco Serial ATA proporciona un incremento del rendimiento, mayores densidades y un menor costo por megabyte con respecto a los dispositivos IDE tradicionales.

La provisión de características de seguridad:

- Control de acceso obligatorio: Security Enhanced Linux (SELinux) proporciona una infraestructura MAC[7] que complementa las características de seguridad de control de acceso discrecional disponibles en el entorno Linux estándar. En un entorno basado en MAC, las características y privilegios de las aplicaciones se fijan por medio de reglas definidas y su cumplimiento es forzado por el kernel. De esta manera, se evita que aplicaciones erráticas puedan comprometer la seguridad del sistema.
- En la gestión de memoria: diversas características, como Exec Shield o los ejecutables de posición independiente PIE[8], se combinan para impedir que las aplicaciones sufran ataques tales como los desbordamientos de buffer.
- Comprobación de la coherencia en tiempo de compilación y de ejecución: las nuevas técnicas de validación de los buffers del compilador GCC y la librería Glibc reducen en gran medida el riesgo de que las aplicaciones defectuosas comprometan la seguridad.
- En su instalación incluye el servidor de Web Apache, el lenguaje de programación PHP y el manejador de bases de datos PostgreSQL.

Lenguaje de programación

Se escogió PHP porque es un lenguaje de programación usado frecuentemente para la creación de contenido para sitios Web con los cuales se puede programar las páginas html. PHP es un acrónimo recursivo que significa "PHP Hypertext Pre-processor", se trata de un lenguaje interpretado usado para la creación de aplicaciones para servidores, o creación de contenido dinámico para sitios Web.

Por el fácil uso y la similitud con los lenguajes más comunes de programación estructurada, como C y Perl, permiten a la mayoría de los programadores crear aplicaciones complejas con una curva de aprendizaje muy suave. También permite involucrarse con aplicaciones de contenido dinámico sin tener que aprender todo un nuevo grupo de funciones y prácticas.

---

[5] HASH se refiere a una función o método para generar claves o llaves que representen de manera casi unívoca a un documento, registro, archivo, etc.

[6] LUN- Logical Unit Number. Número de unidades lógicas.

[7] MAC - Mandatory Access Control. Control de acceso mandatario.

[8] PIE - Position Independent Executable. Ejecutable independiente posición.

Porque su interpretación y ejecución se da en el servidor Web, en el cual se encuentra almacenado el script, y el cliente sólo recibe el resultado de la ejecución. Cuando el cliente hace una petición al servidor para que le envíe una página Web, el servidor ejecuta el intérprete de PHP, el cual procesa el script solicitado que generará el contenido de manera dinámica, pudiendo modificar el contenido a enviar, y regresa el resultado al servidor, el cual se encarga de regresárselo al cliente.

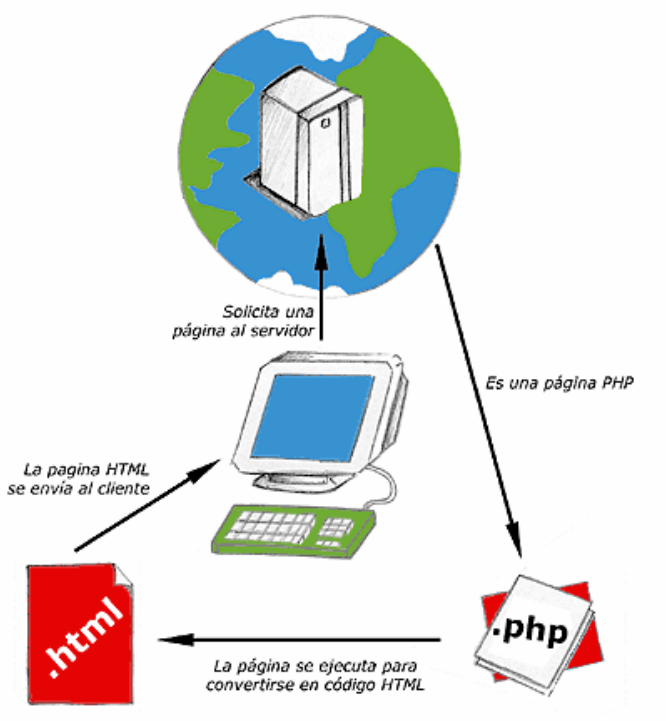


Ilustración 13-3. Flujo de despliegue de una página Web usando PHP.

Permite la conexión a diferentes tipos de servidores de bases de datos tales como MySQL, PostgreSQL, Oracle, ODBC, DB2, Microsoft SQL Server, Sybase, Firebird y SQLite; lo cual permite la creación de Aplicaciones Web muy robustas.

En forma conjunta con el servidor de aplicaciones Apache y el sistema operativo Linux, PHP está entre las tecnologías de código abierto más desarrolladas y usadas. PHP según lo comenta NetCraft ha sobrepasado a Microsoft ASP, convirtiéndose en el lenguaje de desarrollo para Internet mas popular, siendo utilizado en mas de 19 millones de sitios hoy en día.

PHP esta diseñado para el desarrollo de aplicaciones Web que sean accedidas por un gran número de usuarios. PHP es estable y seguro, además de suficientemente robusto para soportar aplicaciones de misión crítica de los negocios que requieran estar constantemente disponibles y seguras.

Fácilmente integrable con ambientes y sistemas empresariales heterogéneos.

PHP es ínter operable con otros lenguajes, protocolos, sistemas, bases de datos, incluyendo C/C++, Java, Perl, COM/.NET, XML/Web services, LDAP, ODBC, Oracle y MySQL. Debido a que PHP es un producto de código abierto (open source), puede ser implementado en cualquier lugar, instalado en cualquier plataforma, con cualquier servidor Web y con cualquier base de datos, sin estar restringido a ninguna plataforma o tecnología.

Almacenamiento de la información

Para almacenar la información, en vez de elegir depositarlas en archivos de texto plano, se eligió un sistema manejador de bases de datos. Pero no debía de ser almacenado en el mismo Sybase, por ser precisamente el que se monitoreará.

Para el almacenamiento de la información se eligió el manejador de bases de datos PostgreSQL porque:

- Soporta transacciones, vistas, triggers, joins, llaves foráneas y procedimientos almacenados, además que permite la programación de estos procedimientos almacenados en distintos lenguajes.
- Soporta el almacenamiento de objetos de gran tamaño.
- Se destaca en ejecutar consultas complejas, consultas sobre vistas, subconsultas y joins[9] de gran tamaño.
- Permite la definición de tipos de datos personalizados.
- Incluye un modelo de seguridad completo con base a permisos de acceso y manipulación de usuarios, bases de datos, hasta del acceso a tablas y vistas.
- Permite distribuir una base de datos en distintos discos para el almacenamiento de grandes cantidades de información.
- Es altamente escalable tanto en la cantidad de datos que puede manipular como en la cantidad de usuarios concurrentes que puede atender.
- Ha sido lanzado bajo licencia BSD que lo hace libre para cualquier propósito, lo que justifica el uso de una herramienta de almacenamiento de información a bajo costo.

Interfase de visualización del usuario

Para la visualización de la aplicación se decidió usar el entorno gráfico del navegador de Internet. Recomendando el navegador Mozilla Firefox por ser software libre y poder ejecutarse bajo:

- Windows: Windows 98, Windows 98 SE, Windows ME, Windows NT 4.0, Windows 2000, Windows XP, Windows Vista.
- Mac OS: Mac OS X 10.2.x o superior.
- Linux: Linux kernel - 2.2.14 o superior.

Dado que el navegador se ha convertido en una herramienta de trabajo que permite el acceso a otros sistemas, y va sustituyendo a herramientas que estaban diseñadas para ambientes específicos. El navegador se va convirtiendo en algo similar a un Sistema Operativo. Un entorno desde donde el usuario accede de forma transparente a multitud de aplicaciones. El navegador y su capacidad de integración de diversas aplicaciones, le permite a las empresas desde servir contenidos y herramientas personalizadas, hasta servicios de servidores, impactando en la eficiencia de las organizaciones. La comunicación servidor/usuario se realiza mediante el navegador usando un servidor Web que transforma la información del servidor en formatos visualizables de XML, HTML, etc.

---

[9] Los joins fueron vistos en el capítulo 3.10 y 3.11.

Mozilla Firefox, como cualquier navegador, interpreta el texto que se le envía del servidor Web (en este caso el servidor Web es apache), lo interpreta y da al usuario una salida gráfica.

El uso de Internet y de un navegador Web como entorno de visualización al usuario, permite que la distribución de la aplicación no se vea afectada por tener que ser compilada para alguna plataforma en particular.

13.3 METODOLOGÍA DE PROGRAMACIÓN EN ESPIRAL

Determinar objetivos

Los objetivos planteados para integrar el Sistema de Seguimiento y Control de Errores (SISCOE) se plantearon bajo el siguiente marco:

- Ayudar, a un bajo costo, a disminuir el tiempo en que se lleva encontrar una solución al ocurrir un error en el Sistema Manejador de Bases de Datos Relacional Sybase.
- Llevar a cabo la programación de una aplicación que pueda documentar el error almacenando la información preferentemente en una base de datos.
- Usar la aplicación para realizar el seguimiento a los errores y tener como referencia la documentación del error ya resuelto para responder un caso similar en el futuro.

Análisis del riesgo

Se requirió hacer un análisis de alternativas, identificación y resolución de riesgos que se describe a continuación.

Al ser una herramienta de autoconsumo, el riesgo se limita en crear una aplicación que no genere costos, por lo que significa que:

- No se tendrá asistencia técnica.
- No se tendrá colaboración en su desarrollo.
- No se contará con ayuda en la instalación.
- No habrá el apoyo en su mantenimiento.

Actividades que indican que se requiere personal con los conocimientos de programación. Siendo el lenguaje PHP el elegido por tener poca curva de tiempo para aprendizaje.

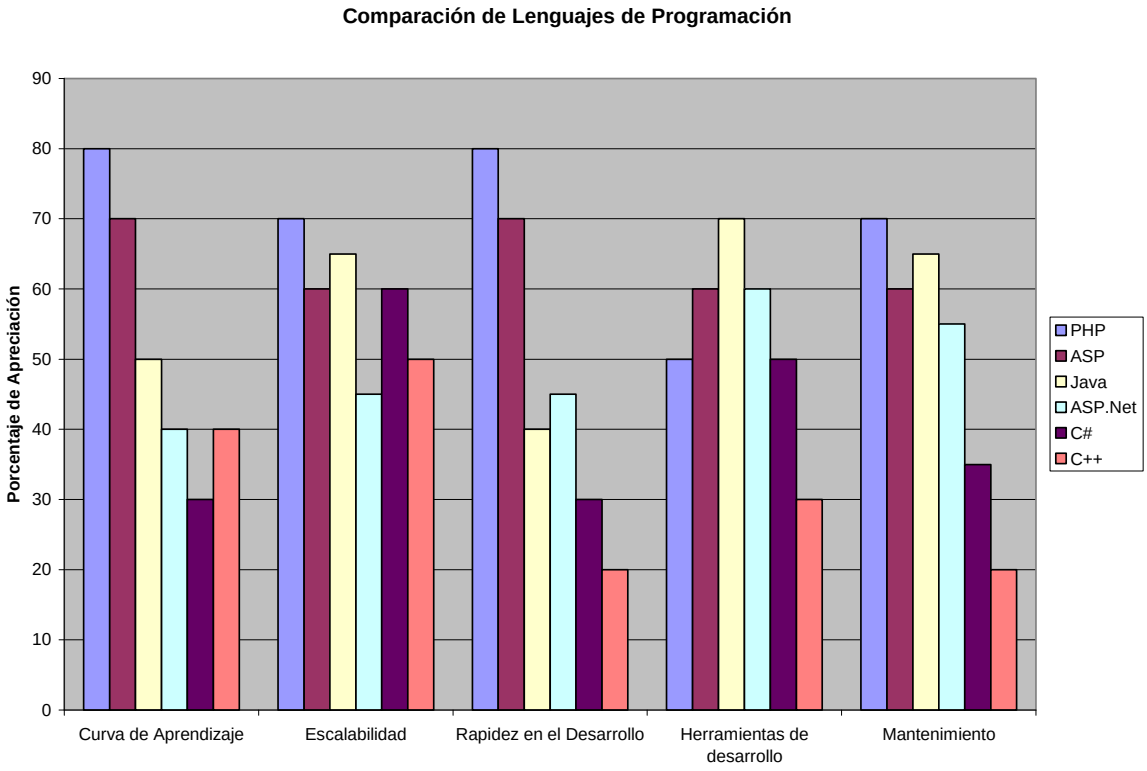


Ilustración 13-4. Apreciación de elementos a comparar al elegir un lenguaje de programación para su implementación en Internet.

Comparar otros sistemas similares

Actualmente existen herramientas que revisan las bitácoras del sistema. Los analizadores de bitácoras, como lo es LogWatch, proporcionan el análisis de lo que pasa con las bitácoras en sistemas operativos tipo Linux.

El caso del LogWatch es una herramienta de Software Libre que revisa las bitácoras del sistema y genera un resultado en modo texto que puede ser usado para su inspección ayudando a resumir la información de las bitácoras del sistema, como es la frecuencia del acceso o espacio en disco.

Como el objetivo es que a un bajo costo se revise la bitácora de Sybase se puede usar Software Libre para reducir los costos justificando su uso como se demostró en el capitulo 13.2.

Desarrollar, verificar, validar y probar

Se estuvo desarrollando con datos en un ambiente controlado, en un servidor de pruebas, donde no afectara a los servicios en producción. Usando una bitácora de errores de Sybase 11.9.2 para el desarrollo, verificación y aprobación. Pasando las pruebas del porcentaje de espacio en disco y el suministro de información al realizarse una falla en los archivos de configuración de Sybase.

Planificar

En base en la observación del funcionamiento del SISCOE versión 1. La planificación para un futuro se pretende llevar a cabo el uso de las sesiones de PHP y contar con opciones de configuración para una segunda versión del SISCOE.

13.4 USO DEL SISTEMA SISCOE

Instalación

Para la instalación se requiere:

- Descomprimir los archivos fuente de PHP.
- Apuntar la ruta del archivo del log a donde este el archivo de la bitácora de errores de Sybase.
- Restaurar en la base de datos de PostgreSQL la base en donde se almacenan los datos.

El objetivo es ofrecer la solución idónea para elevar la competitividad de la empresa a un costo mínimo.

El Sistema de Seguimiento y Control de Errores es una aplicación Web que esta desarrollada con Software Libre, esta orientada hacia empresas que deseen disminuir la inversión en herramientas de monitoreo de sus Sistemas de Gestión de Bases de Datos Relacionales en Sybase y del espacio en disco duro en Sistemas Operativos tipo Linux.

La funcionalidad de este software abarca desde el monitoreo de espacio en disco duro hasta el monitoreo y seguimiento de errores en la RDBMS de Sybase.

Su desarrollo está basado en la arquitectura de 3 Capas:

- El Lenguaje PHP como lenguaje de programación instalado sobre el servidor de Internet Apache.
- El servidor de bases de datos PostgreSQL en una plataforma tipo Linux.
- Como Front-End Cliente el navegador de Internet Mozilla.

Requerimientos

Tanto Apache, Mozilla, PostgreSQL, PHP, Linux se pueden obtener gratuitamente. Pueden instalarse sin requerir comprar una licencia de uso y están basados en los principios del Software Libre.

Simbología

ICONOS Y BOTONES			
Monitoreo de Errores	Ingresar al monitoreo de usuarios		Modifica usuario
Administrar Usuarios	Ingresar a la actualización de usuarios		Elimina un usuario
Administrar Errores	Ingresa a la administración de errores		

Ingresar	Ingresa el password		Status incorrecto
Regresar	Regresar a la página anterior		Status correcto
Agregar	Agrega un usuario		
Salir	Salir		Guardados los cambios
Marcado =Solucionado; Sin Marca =No Solucionado	Marca si esta solucionado o no un error		

Tabla 13-3. Simbología de los iconos y botones

Acceso al sistema

Una vez instalado el SISCOE, para poder ingresar a los distintos módulos del sistema debe autentificarse en el Sistema de Seguimiento y Control de Errores (SISCOE) del Manejador de Base de Datos Sybase.

Usando un navegador Web como Mozilla Firefox se ingresa en la dirección Web en donde fue configurado el SISCOE, accediendo a:

`http://mi.direccion.ip.com/siscoe/`

Una vez accediendo lo primero que hay que hacer es autentificarse.

Usuario inicial

El software de monitoreo SISCOE cuenta con un usuario inicial con el rol de administrador de usuarios (**`sso_role`**), llamado **`adminuser`**. Este usuario puede realizar las siguientes funciones:

- Crear de usuarios.
- Asignar y cambiar contraseñas a usuarios.
- Definir y cambiar tipos de usuario.
- Eliminar usuarios.

Administración de usuarios

Para acceder con el usuario **`adminuser`** hay que:

1. Ingresar el Login: **`adminuser`**
2. Ingresar el Password: **`adminuser`**
3. Dar clic en el botón: Ingresar

4. Dar clic en el botón: 

Tipos de Usuarios

Para poder cumplir con la segregación de funciones dentro de una empresa es primordial crear usuarios que puedan desarrollar las diversas actividades que contiene el sistema. Siendo los siguientes tipos de usuario los que le proporciona el SISCOE:

`sso_role`. Permiso para administrar de usuarios. Tiene el privilegio de administración de usuarios. El usuario al tener asignado el rol **`sso_role`** puede crear, actualizar o eliminar usuarios.

`sa_role`. Permiso para administrar errores. Tiene el privilegio de administrar el histórico de errores. El sistema almacena el histórico del log de errores de Sybase y este usuario puede editar la información relacionada al error para documentar la solución del problema

`oper_role`. Usuario para monitorear errores. Tiene los privilegios de monitorear el log de errores de Sybase, así como de visualizar el espacio disponible en las particiones del Sistema Operativo.

Creación de usuarios

La primera vez que ingrese al módulo de **`Administrador de Usuarios`** ingresando con el usuario **`adminuser`** verá la pantalla mostrada en la Ilustración 13-5.

Ilustración 13-5. Pantalla inicial del Administrador de Usuarios.

Para agregar un usuario hay que dar clic en el botón 

Una vez que se seleccionó **`Agregar`** se desplegará la pantalla mostrada en la Ilustración 13-6 en la cual es importante llenar los campos **`Usuario`**, **`Contraseña`** y **`Tipo de Usuario`**.

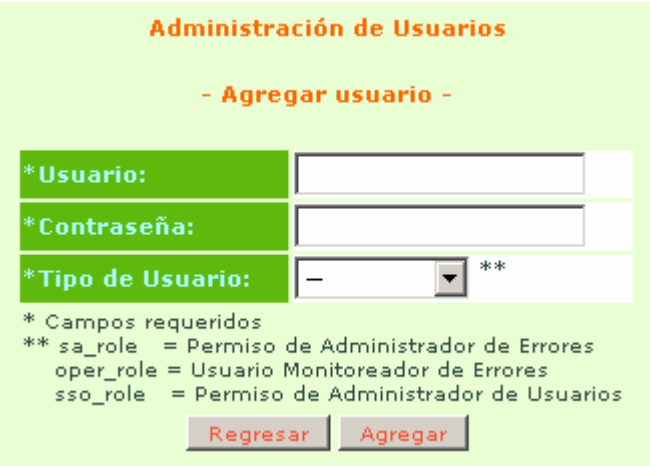


Ilustración 13-6. Pantalla de llenado de datos de identificación del usuario.

El campo **Usuario** debe tener un máximo de 10 caracteres.

Contraseña puede ser alfanumérica y debe tener una longitud no mayor a 10 caracteres. Distingue entre mayúsculas y minúsculas.

Tipo de usuario asigna el rol que cumpla con las actividades a desarrollar.

- Una vez ingresados los campos dar clic en el botón 

- El sistema muestra el mensaje **`Cambios Efectuados`** indicando que los cambios fueron efectuados como se ve en la Ilustración 13-7.

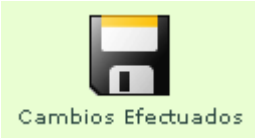


Ilustración 13-7. Mensaje que indica que los cambios fueron efectuados con éxito.

Actualización de usuarios

Ingresar a la aplicación con el usuario **`adminuser`** o con algún usuario creado con el rol de **`sso_role`** y seleccionar el botón 

En la actualización de datos pueden ser modificados el password o cambiar el rol al usuario.

- Seleccionar el icono  sobre el usuario que se desea actualizar.

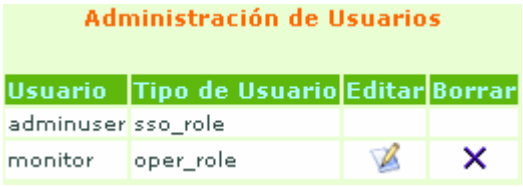


Ilustración 13-8. Lista de usuarios de la Administración de Usuarios donde se seleccionar la edición de los datos de algún usuario.

- Actualizar la contraseña y/o el tipo de usuario deseado.

Ilustración 13-9. Pantalla Cambiar datos.

- Guardar los cambios dando clic en el botón "**Guardar**".

Eliminar usuarios

Ingresar a la aplicación con el usuario **adminuser** o con algún usuario creado con el rol de **sso_role** y seleccionar 

esto para acceder al apartado de "**Administración de Usuarios**".

Ilustración 13-10. Pantalla de Administración de Usuarios con varios usuarios ingresados.

Para borrar los usuarios deseados del Sistema.

- Seleccionar el icono ✕ sobre el usuario que se desee borrar.

- Aceptar el cambio.

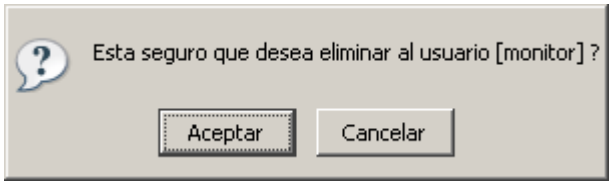


Ilustración 13-11. Caja de diálogo pidiendo la confirmación para poder eliminar un usuario, en este caso el usuario monitor.

Nota:

Por cuestiones de seguridad de la aplicación el usuario **`adminuser`** no tiene la opción para ser actualizado o eliminado del sistema. La contraseña se proporciona con la instalación del SISCOE recomendando que sea diferente por cada instalación.

Monitoreo

El uso adecuado de esta aplicación proporcionará que el Sistema sea confiable, seguro y estable gracias al monitoreo que proporciona.

- Ingresar con un usuario que tenga el rol con permisos de **`oper_role`**.

Ilustración 13-12. Pantalla de inicio del SISCOE.

Este módulo cuenta con dos funciones:

1. Monitoreo de errores.
2. Monitoreo del espacio en disco.

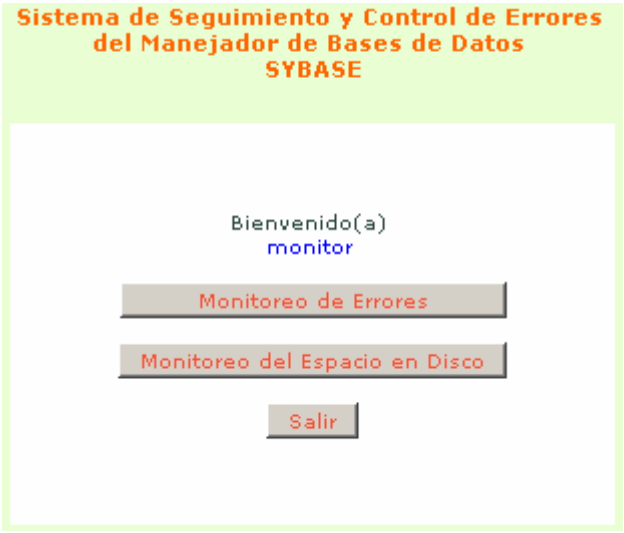


Ilustración 13-13. Pantalla de bienvenida al usuario con permisos de oper_role con las opciones que este tipo de usuario tiene permitido visualizar.

Monitoreo de errores

- Seleccionar el botón 

    - Al seleccionar "**Monitoreo de Errores**" la aplicación revisará el Log de errores del RDBMS Sybase y sólo recuperará para su despliegue los errores que no hayan sido solucionados ya sean de tipo **kernel** o de tipo **server**.

Errores recuperados del log de errores

Si el monitor de errores encuentra algún error se despliega la siguiente información:

Ilustración 13-14. Pantalla del monitor de errores que muestra los errores encontrados en el RDBMS Sybase.

Fecha La fecha en la cual el error fue generado.

Hora Hora en que ocurrió el error.

Tipo Están clasificados los errores en:

- Errores de **server** y
- Errores de **kernel**

Error En el caso de un error de **kernel** mostrará de manera general de que se trata el error.

#error En el caso de ser un error de **server** desplegará el número de error, el cual posteriormente se puede utilizar para buscar las posibles soluciones.

Severidad En el nivel de severidad mientras menor es el número el error es más grave.

Estado Para cada número de error se tiene diferentes estados del error.

Ejemplo:

Para el número de error 216 se tienen los siguientes estados

1. Durante la fase de renormalización, si SYBASE no puede eliminar una tabla temporal creada, ocurre el error 216 con Estado número 1. Estos son tiempo de definición en tablas temporales.

2. Durante la fase de normalización, si SYBASE no puede eliminar una tabla temporal creada durante la fase de análisis, ocurre el error 216 con Estado número 2.

3. Durante un aborto de la fase de normalización, si SYBASE no puede eliminar una tabla temporal, ocurre el error 216 con Estado número 3.

4. Cuando un proceso es eliminado, si SYBASE falla al limpiar una tabla temporal, el error 216 ocurre con Estado número 4.

Información En el caso de error de **kernel** el campo información dará una explicación más a detalle sobre el problema.

Nota: La combinación del número de error y el estado es lo que debe buscarse en el manual para los errores de **server** de la documentación del manejador Sybase.

Base de datos sin errores

Si la aplicación no encuentra ningún error, o fueron marcados como corregidos por el usuario con el rol de administrador de errores, el sistema mandará el siguiente mensaje.

Ilustración 13-15. Pantalla de confirmación de que el manejador de Sybase no tiene errores.

Monitoreo del espacio en disco

Si se selecciona la opción de "**Monitoreo de Espacio en Disco**" se podrá visualizar gráficamente el espacio disponible que se encuentre en cada una de las particiones.

- Ingresar al **monitoreo del espacio en disco** seleccionando el botón 

- Seleccionar la partición en la que se desea visualizar el espacio libre.

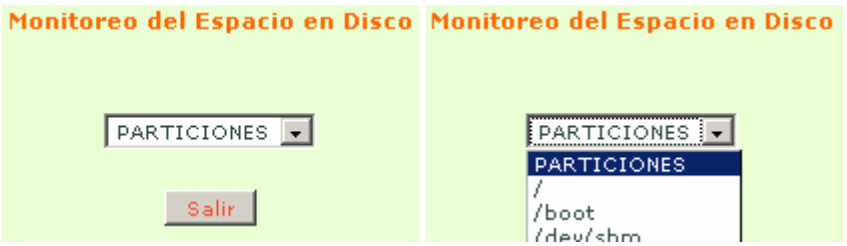


Ilustración 13-16. Selección de la partición para monitorear su espacio libre.

- **Status correcto.** En el caso de que la partición seleccionada tenga un espacio ocupado menor al 80% se mostrará un mensaje con el status correcto mostrando la proporción en una barra de porcentaje con la cantidad de espacio ocupado.

Ilustración 13-17. Pantalla que muestra un Status correcto de la partición seleccionada, mostrando el porcentaje de uso de la partición.

- **Status incorrecto**. En el caso de que la partición seleccionada tenga un espacio ocupado mayor al 80% se mostrará un mensaje con el status de incorrecto mostrando la proporción en una barra de porcentaje con la cantidad de espacio ocupado.

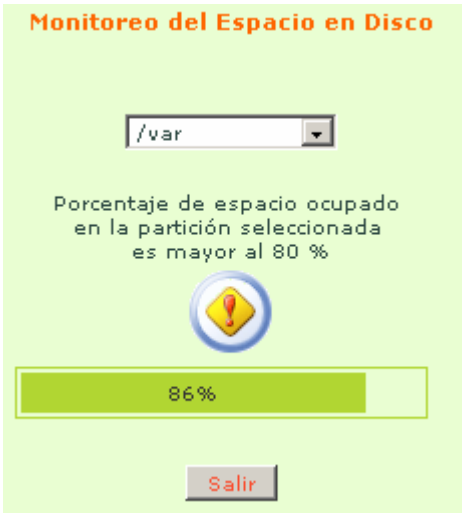


Ilustración 13-18. Pantalla que muestra un Status incorrecto de la partición seleccionada, mostrando el porcentaje de uso de la partición.

Status incorrecto

En cualquier momento que aparezca algún error será más fácil su visualización al encontrarse con el icono de **`Status incorrecto`**.

Ilustración 13-19. Icono de Status incorrecto.

Este icono también sirve para mandar un mensaje de correo electrónico al administrador de la base de datos para informarle sobre el error.

Al darle clic sobre el icono de **`Status incorrecto`** se ejecuta el cliente de correo electrónico configurado para escribir el mensaje informando al administrador sobre el error.

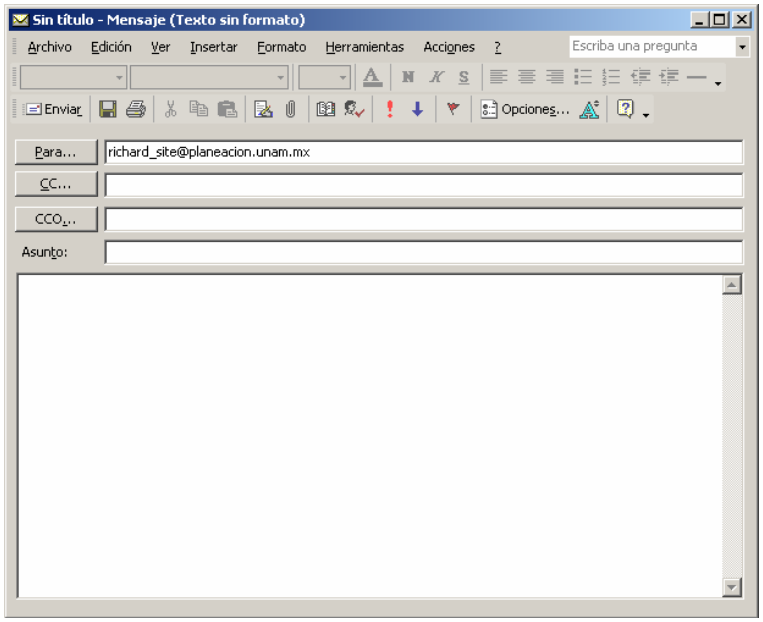


Ilustración 13-20. Operación de la aplicación al seleccionar el icono de Status incorrecto.

Administración de Errores

Histórico de Errores

La aplicación al mismo tiempo que realiza la lectura del Log de errores de la Base de Datos va guardando un histórico, el cual puede ser modificado por un usuario que tenga el rol **`sa_role`** para llevar una documentación adecuada de las modificaciones o cambios que se han realizado en el RDBMS.

Acceso a Administrar Errores

- Ingresar con un usuario que tenga el rol con permisos de **`sa_role`**. Como se muestra en la Ilustración 13-12.

- Dar clic en el botón de "**`Administrar Errores`**".

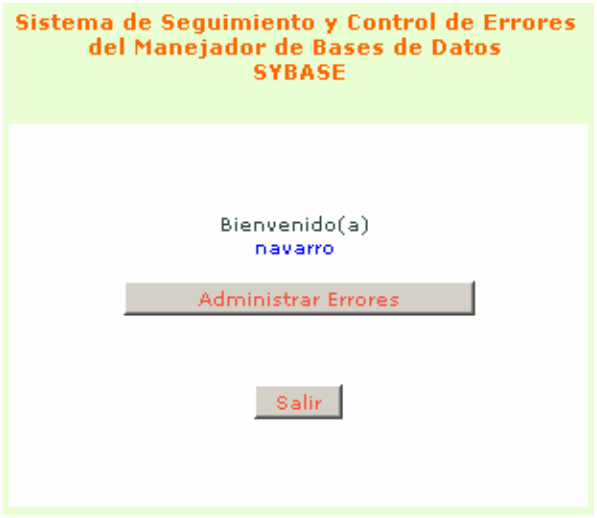


Ilustración 13-21. Pantalla de bienvenida al usuario con permisos de sa_role con las opciones que este tipo de usuario tiene permitido visualizar

La aplicación ingresará a la bitácora del Log de errores, en la cual el administrador de errores validará cuando el error ya este solucionado y podrá documentar la solución.

Ilustración 13-22. Pantalla de Administración de Errores con la bitácora del Log de errores.

Documentar el Error

- Seleccionar el icono  sobre el error que se desea documentar.
- Activar el botón de selección ☑ del campo **`Status Revisión`** para marcarlo como **`Error solucionado`**.
    - ☑ Error solucionado
    - ☐ Error no solucionado

Ilustración 13-23. Sección para marcar el Status de la Revisión del Error.

- Si se desea se puede documentar el error en el campo **`Descripción Solución`**. Esto es útil cuando se genere el mismo error, poder ver en la documentación la solución con la cual se resolvió el problema.

Ilustración 13-24. Sección para llenar la Descripción de la Solución del error.

- Dar clic en el botón 

para archivar los cambios.

Status de Revisión

Una vez guardados los cambios, si el **`Status de Revisión`** quedó registrado como **`Error solucionado`** entonces en la bitácora de errores se modificará y en la columna **`Revisado`** aparecerá el mensaje **`[OK]`**.

Ilustración 13-25. Pantalla de Administración de Errores con la bitácora del Log de errores con el status de Revisado..

Vista Monitoreo

Posteriormente cuando un usuario con permisos para monitorear el log de errores ingresa (**`oper_role`**) ya no visualizará los errores marcados previamente como corregidos.

Ilustración 13-26. Pantalla del monitor de errores que muestra sólo los errores no solucionadols del RDBMS Sybase.

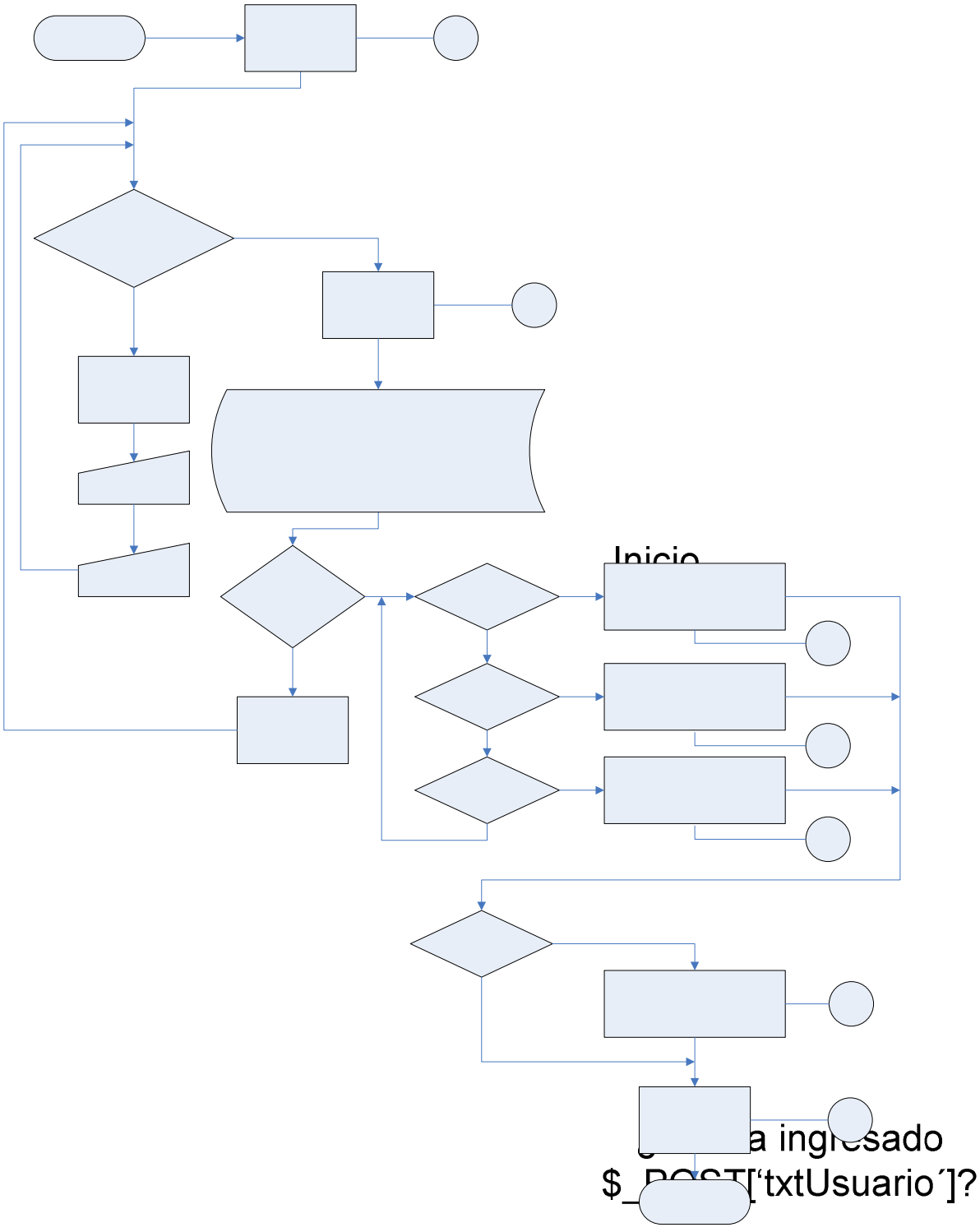


Ilustración 13-27. Archivo index.php Se encarga de ingresar a algún apartado dependiendo del rol del usuario al autenticarse.

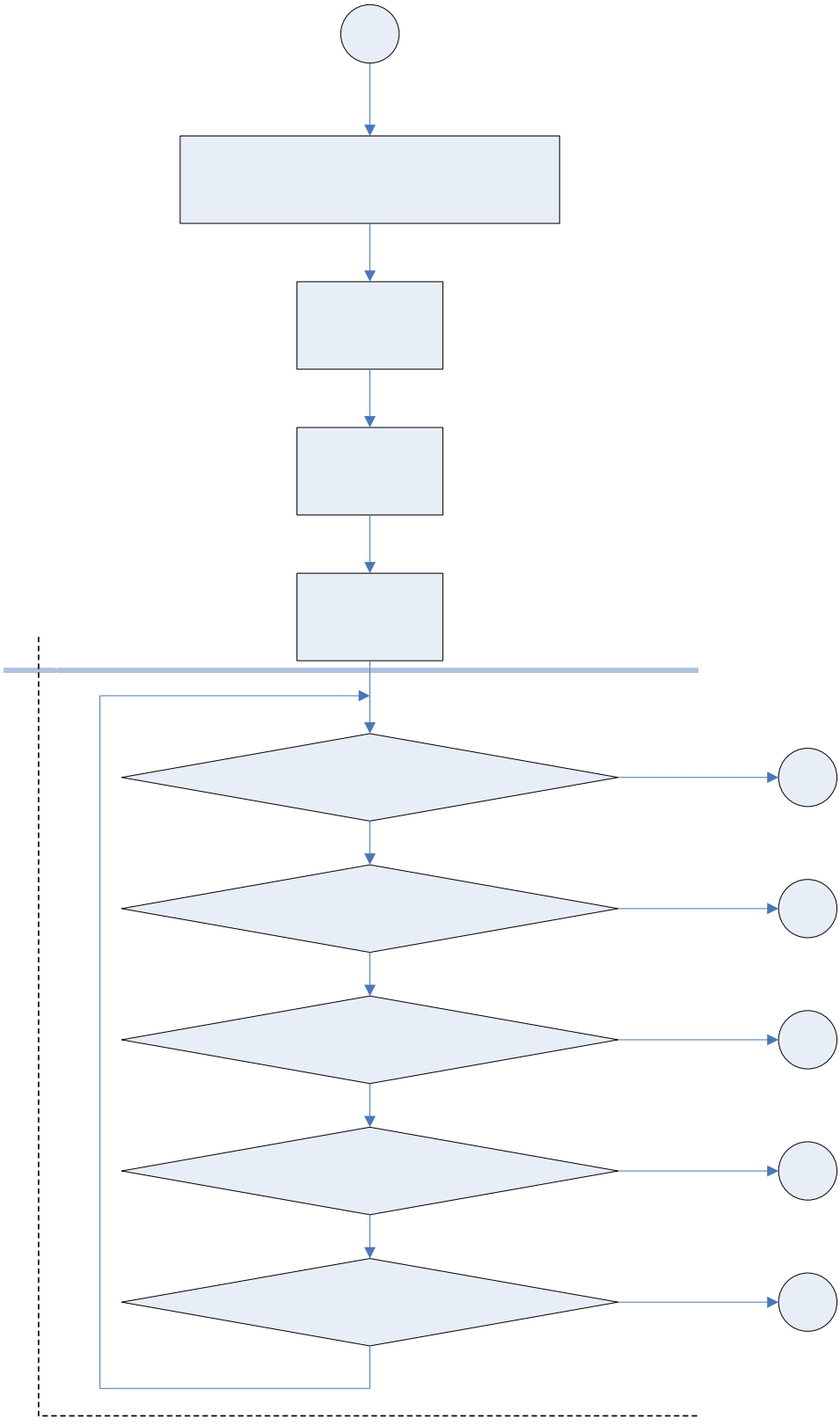


Ilustración 13-28. Archivo: encabezado.php (1). Se encarga de mostrar el menú en el encabezado de la página.

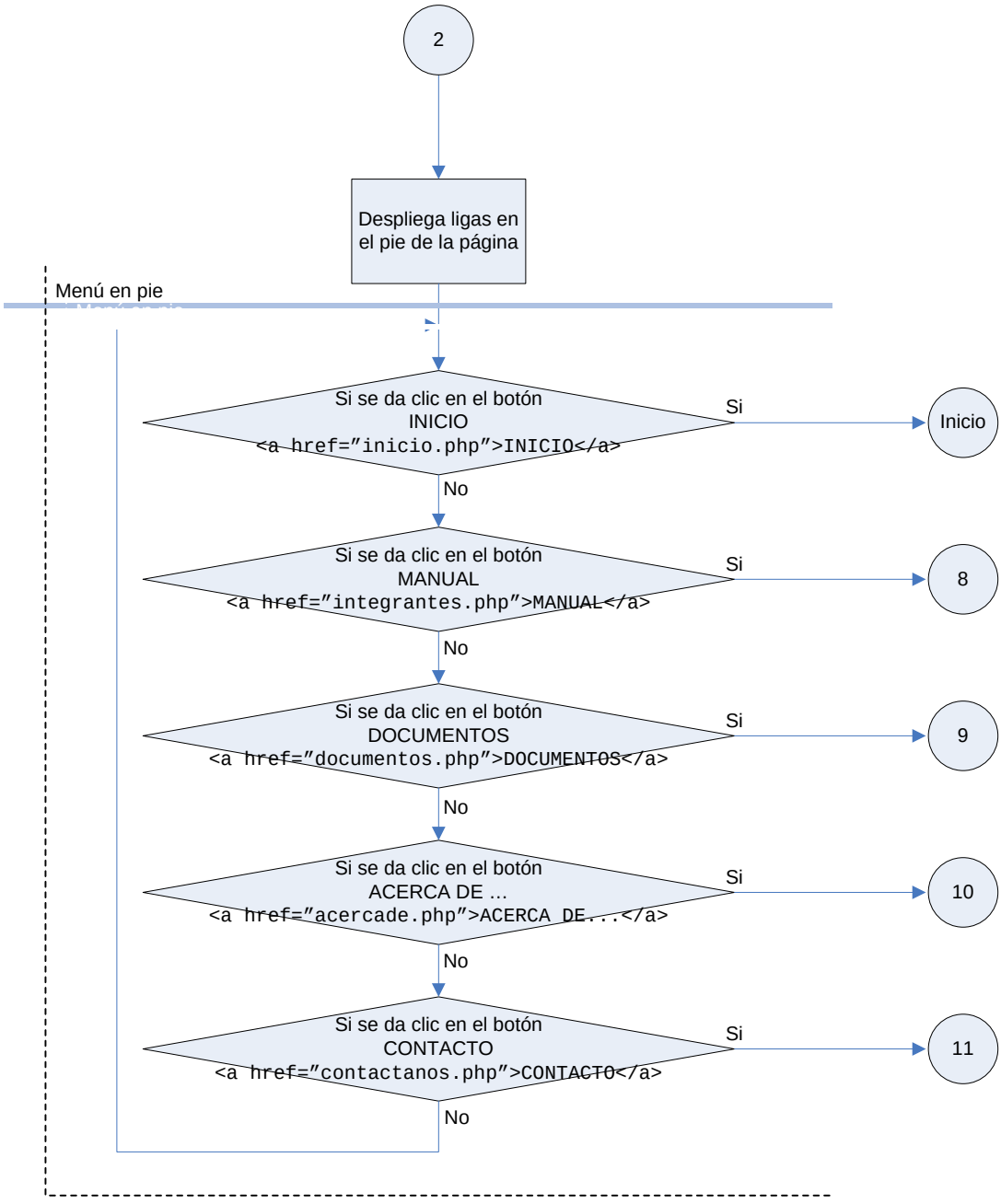


Ilustración 13-29. Archivo: pie.php (2). Se encarga de mostrar información al pie de la página.

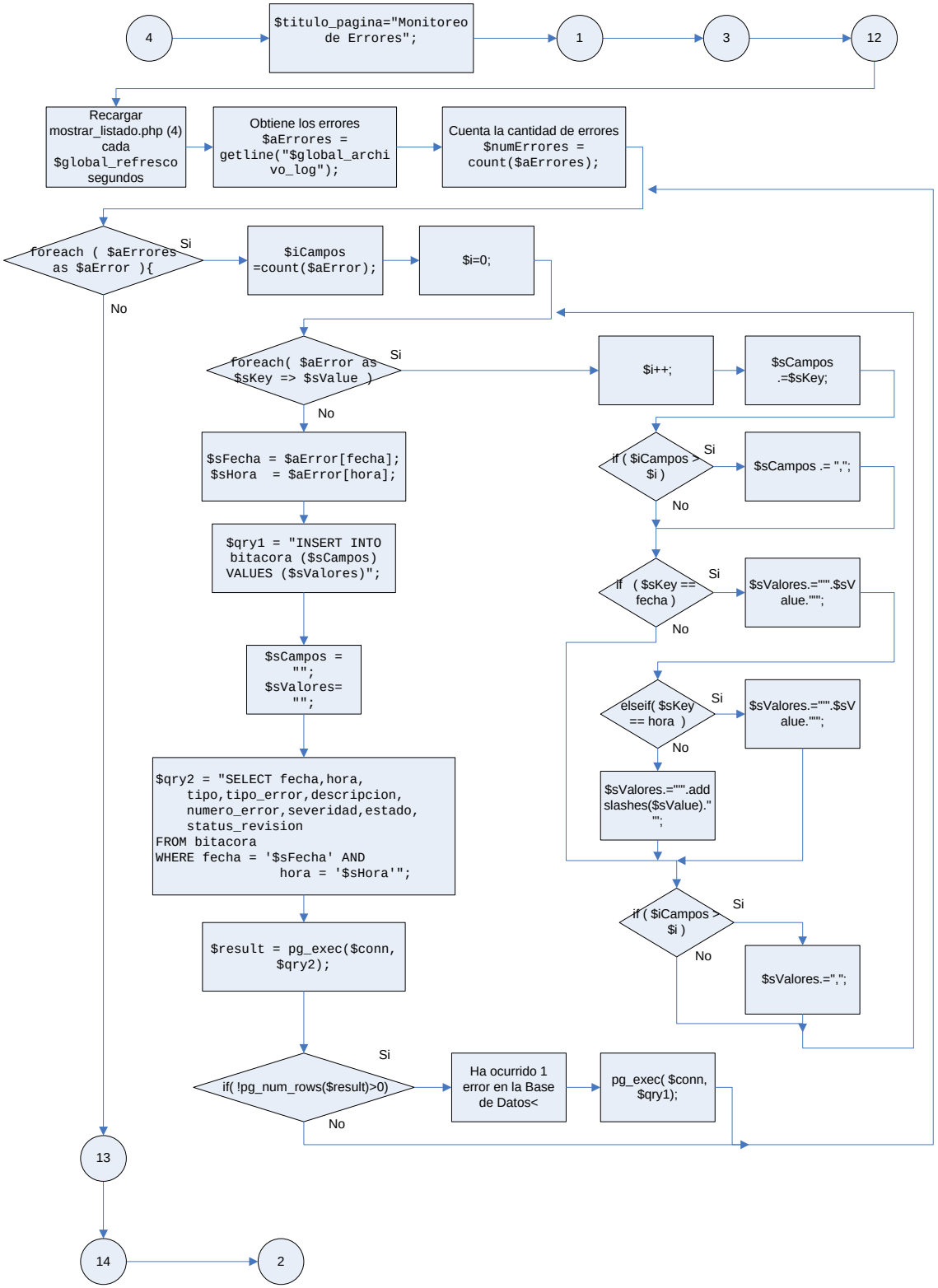


Ilustración 13-30. Archivo: mostrar_listado.php (4). Se encarga de mostrar el listado de errores que genere la base de datos de Sybase almacenando la información en la tabla bitácora en PostgreSQL.

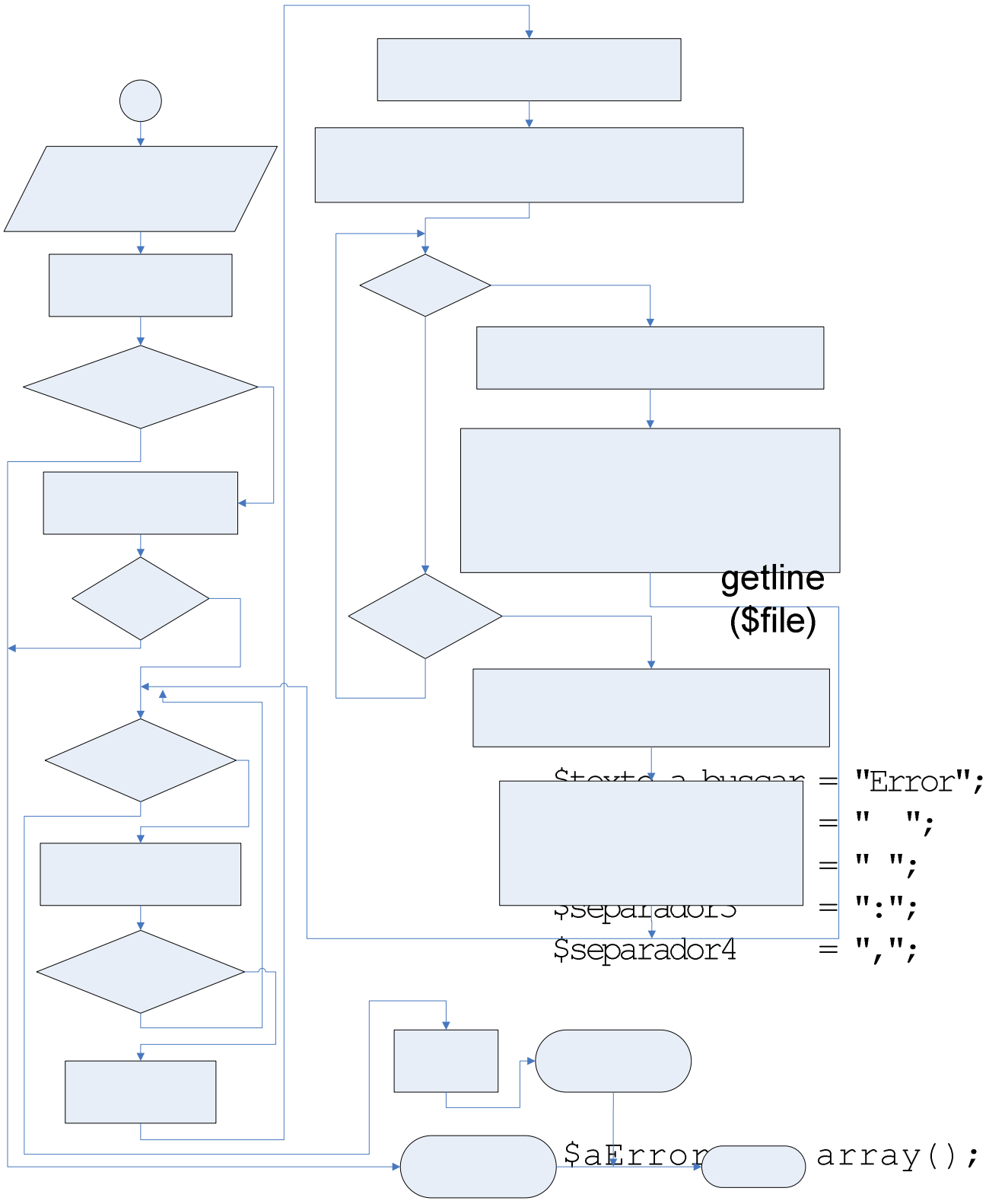


Ilustración 13-31. Archivo: funciones.php (Función: getline($file)). La función getline se encarga de obtener la línea del error y la devuelve en forma de arreglo.

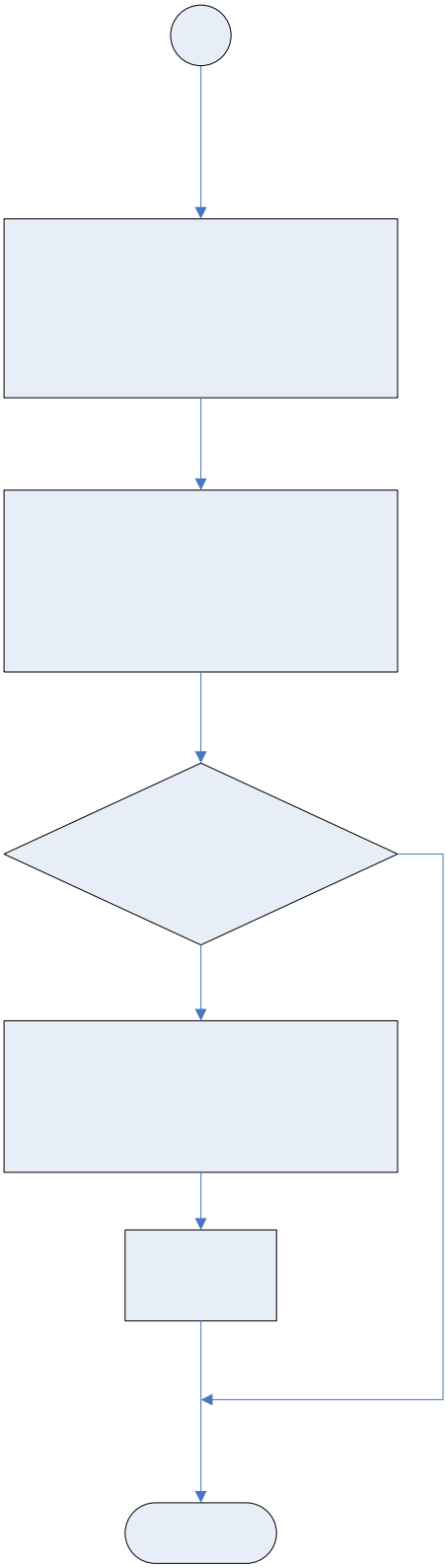


Ilustración 13-32. Archivo: conectar.php (3). Se encarga de realizar el proceso de conexión a la base de datos de PostgreSQL.

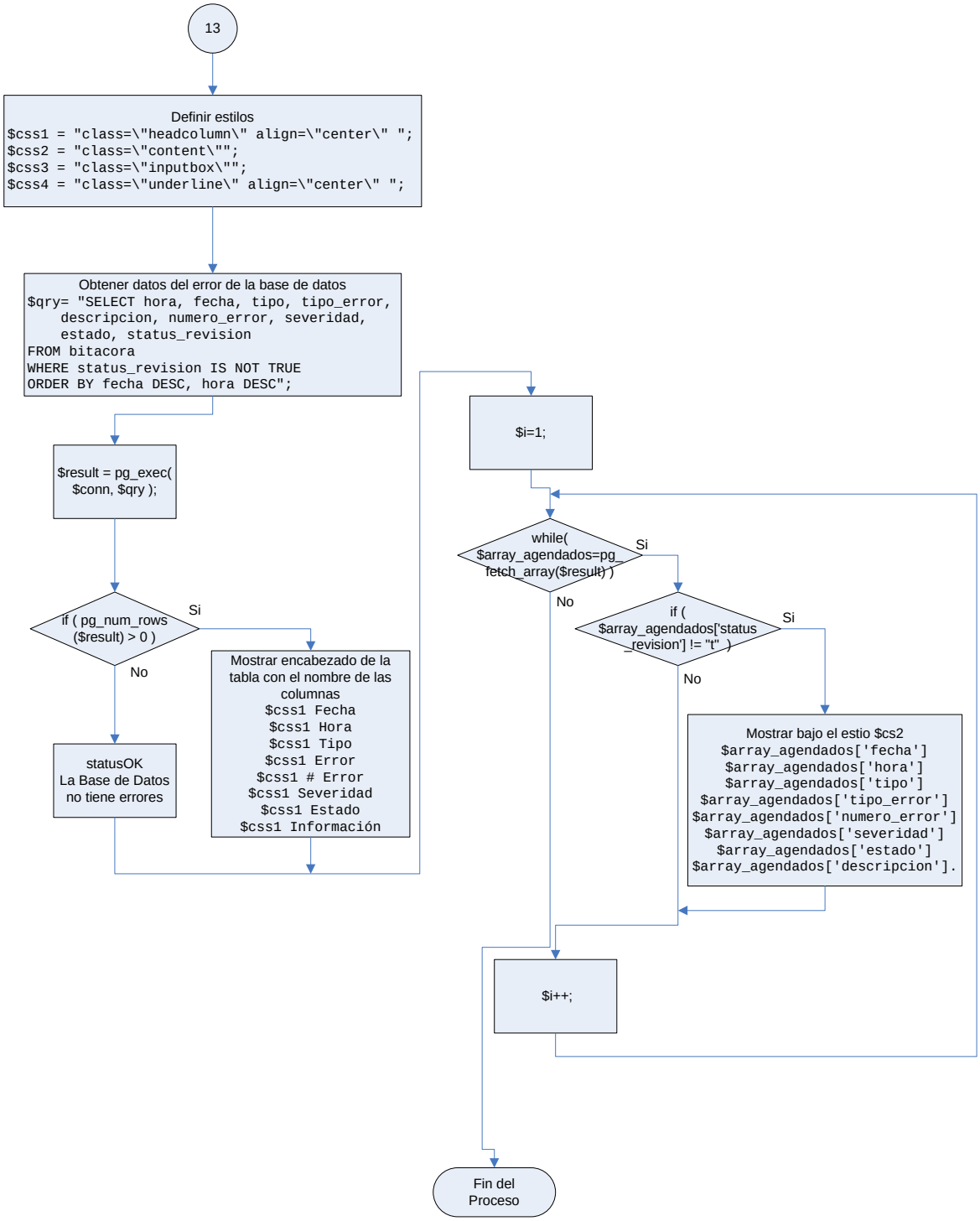


Ilustración 13-33. Archivo: mostrar.php (13). Es la encargada de obtener de la base de datos de PostgreSQL la información almacenada del error y mostrarla en pantalla.

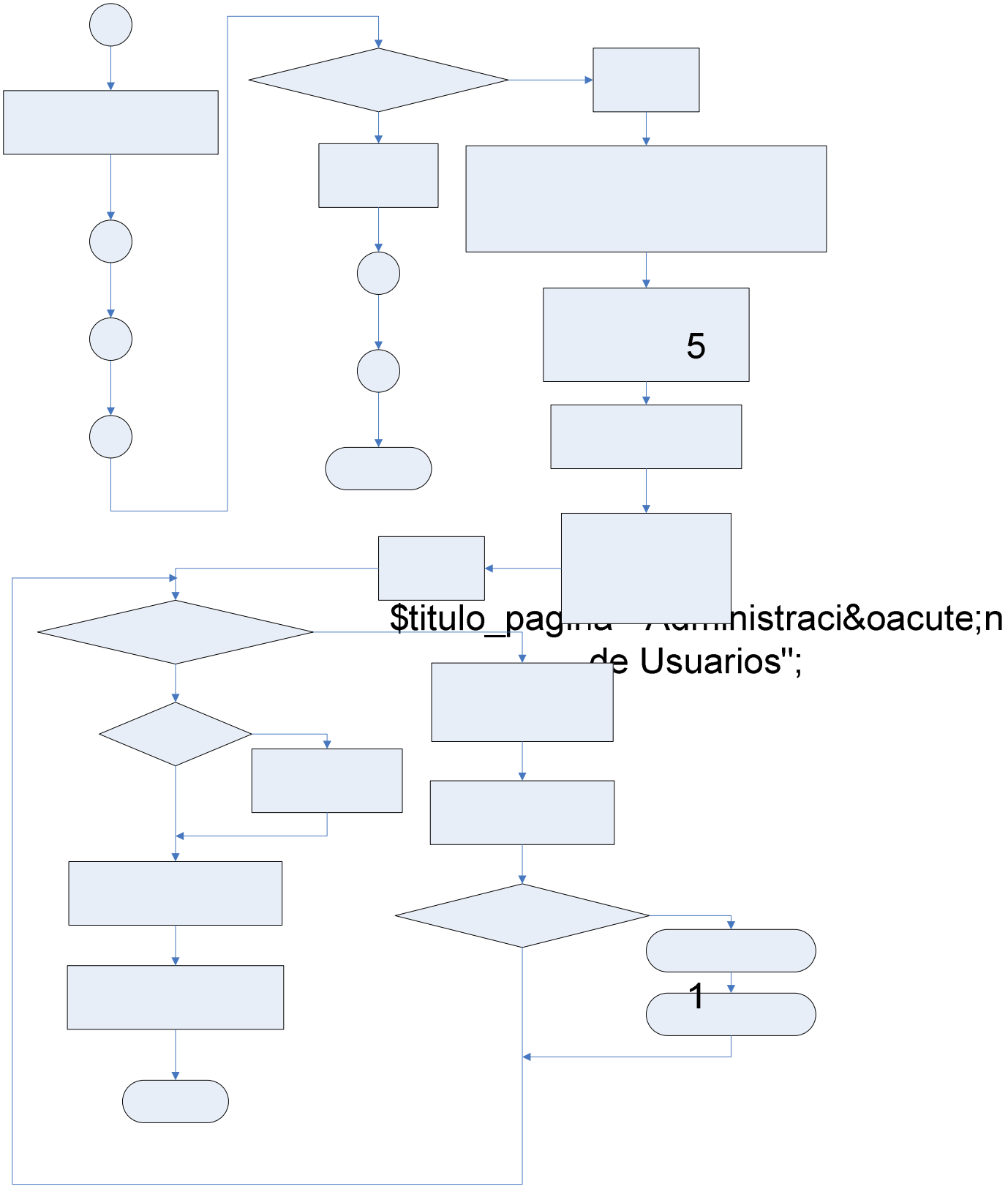


Ilustración 13-34. Archivo: listar_usuarios.php (5). Muestra la información de los usuarios para su administración.

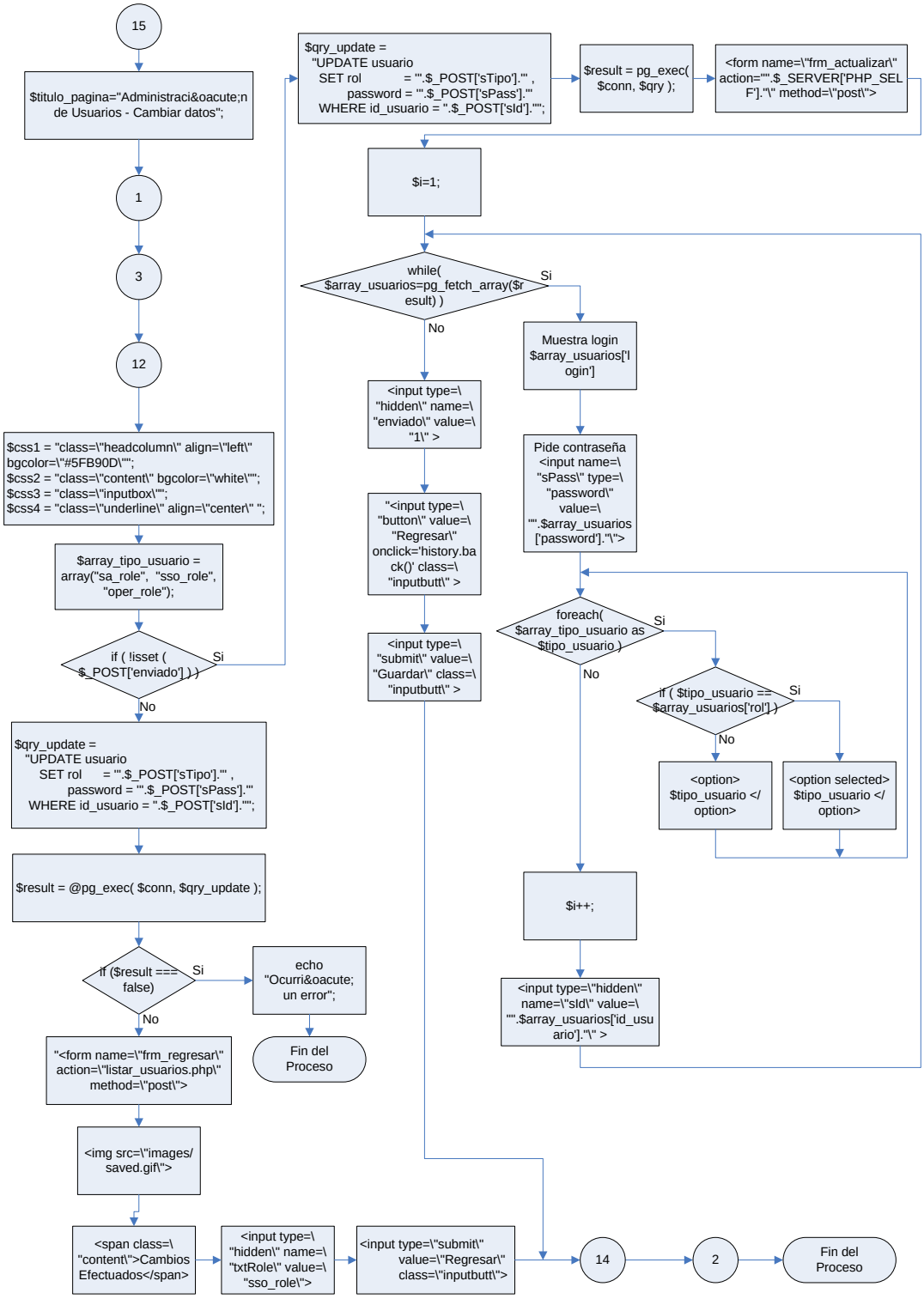


Ilustración 13-35. Archivo: editar_usuario.php (15). Edita la información del usuario para cambiar la contraseña o el rol que desempeña

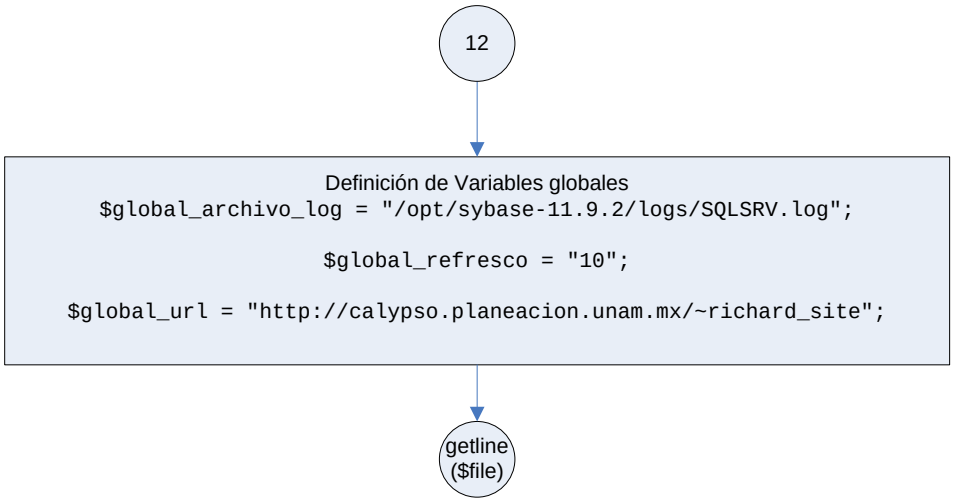


Ilustración 13-36. funciones.php (12). Se encarga de definir las funciones que se van a usar en el programa como lo es la función getline.

Referencias:

(1) encabezado.php

(2) pie.php

(3) conectar.php

(4) mostrar_listado.php

(5) listar_usuarios.php

(6) mostrar_y_cambiar.php

(7) checkdiskspace.php

(8) integrantes.php

(9) documentos.php

(10) acercade.php

(11) contactanos.php

(12) funciones.php

(13) mostrar.php

(14) desconectar.php

(15) edtar_usuarios.php

13.6 CÓDIGO

A continuación se muestra el código fuente del Sistema de Seguimiento y Control de Errores (SISCOE 1.1).

acercade.php

```
<?php
$titulo_pagina="Acerca de ...";
include ("encabezado.php");
?>

<br>
<center><span class="logtxt">Acerca de ...</span></center>
<br><br>

<table align="center" border="0">
 <tr>

  <td>

   <center>
   <span class="content">
   &copy;2008<br><br>
   <b>Si</b>stema de <b>S</b>eguimiento y <b>C</b>ontrol de <b>E</b>rrores<br>
   del Manejador de Bases de Datos<br>
   Sybase<br>
   Versi&oacute;n 1.1<br><br>
   ( SISCOE 1.1 )<br><br><br>
   Universidad Nacional Aut&oacute;noma de M&eacute;xico<br><br>
   Facultad de Estudios Profesionales<br>
   Arag&oacute;n<br><br>
   Direcci&oacute;n General de Servicios de C&oacute;mputo Acad&eacute;mico<br><br>
   Centro Coapa de Extensi&oacute;n en C&oacute;mputo y Telecomunicaciones<br><br>
   Diplomado Administraci&oacute;n de Bases de Datos<br>
   Sexta Generaci&oacute;n<br><br><br><br>
   Programador<br>
   Ricardo Navarro L&oacute;pez<br>
   richard_site @ planeacion.unam.mx<br><br><br>
   2008<br>
   Copyright<br><br>
   &Uacute;ltima actualizaci&oacute;n<br>Junio 2008
   </span>
   </center>
   <br>

  </td>

 </tr>
</table>

<br>

<?php include ('pie.php'); ?>
```

- -

agregar_usuario.php

```
<script type="text/javascript">
 function nospaces(t){
  if (t.value.match (/\s/g) ){
   alert('Los espacios no estan permitidos en el campo [Usuario]');
   t.value=t.value.replace(/\s/g,'');
  }
 }

 function validar(){
```

```
  t1 = document.frm_agregar.sUser.value;
  if(  t1 == '' ){
   alert('Campo [ Usuario ] es requerido');
   document.frm_agregar.sUser.focus();
   return false;
  }
  t2 = document.frm_agregar.sPass.value;
  if(  t2 == '' ){
   alert('Campo [ Contrasena ] es requerido');
   document.frm_agregar.sPass.focus();
   return false;
  }
  //Validacion solo sirve en Mozilla y no en Internet Explorer
  t3 = document.frm_agregar.sTipo.value;
  if(  t3 == '--' ){
   alert('Campo [ Tipo de Usuario ] es requerido');
   document.frm_agregar.sTipo.focus();
   return false;
  }
 }
</script>
<?php
$titulo_pagina="Administraci&oacute;n de Usuarios - Agregar Usuarios";
include ("encabezado.php");
include ("conectar.php");
include ("funciones.php");

$css1 = "class=\"headcolumn\" align=\"left\" bgcolor=\"#5FB90D\"";
$css2 = "class=\"content\" bgcolor=\"white\"";
$css3 = "class=\"inputbox\"";
$css4 = "class=\"underline\" align=\"center\" ";

$array_tipo_usuario = array("sa_role","sso_role", "oper_role");

if ( !isset ( $_POST['enviado'] ) ) {

        print "<br><center><span class=\"logtxt\">Administraci&oacute;n de
Usuarios</span><br><br>";
        print "<span class=\"logtxt\">- Agregar usuario -</span></center><br>";

        print "<table border=\"0\" align=\"center\">";

        print "<form name=\"frm_agregar\" action=\"".$_SERVER['PHP_SELF']."\" method=\"post\"
onSubmit=\"return validar()\">";

        print "<tr>";
        print "<td $css1>*Usuario:</td>";
        print "<td $css2><input name=\"sUser\" type=\"textbox\"
maxlength=\"10\"onkeyup=\"nospaces(this)\"></td>";
        print "</tr>";

        print "<tr>";
        print "<td $css1>*Contrase&ntilde;a:</td>";
        print "<td $css2><input name=\"sPass\" type=\"password\" maxlength=\"10\" ></td>";
        print "</tr>";

        print "<tr>";
        print "<td $css1>*Tipo de Usuario: </td>";
        print "<td $css2>";
        print "<select name=\"sTipo\">";
        print "<option> -- </option>";
        foreach(  $array_tipo_usuario as $tipo_usuario )
                print "<option> $tipo_usuario </option>";
        print "</select> <sup>**</sup>";
        print "</td>";
        print "</tr>";
        print "<tr><td colspan=2>
               <span class=\"content\">
                * Campos requeridos<br>
                ** sa_role   = Permiso de Administrador de Errores<br>
         oper_role = Usuario Monitoreador de Errores<br>
```

```
           sso_role   = Permiso de Administrador de Usuarios
               </span><br></td>
               </tr>";
        print "</table>";

        print "<center>";
        print "<input type=\"hidden\" name=\"enviado\" value=\"1\" >";
        print "<input type=\"hidden\" name=\"txtRole\" value=\"".$_POST['txtRole']."\">";
        print "<input type=\"button\" value=\"Regresar\" onclick='history.back()'
class=\"inputbutt\" > ";
        print "<input type=\"submit\"
                      value=\"Agregar\"
                      class=\"inputbutt\" > ";
        print "</center>";
        print "<br>";
}else{
        $qry_verifica = "SELECT * FROM usuario WHERE login = '".$_POST['sUser']."'";
        $result = @pg_exec ( $conn, $qry_verifica );
        $usuarios_iguales = pg_num_rows( $result );
        if ( $usuarios_iguales > 0 )
                print "<br><center>
                        <img src=\"images/warning.gif\"><br>
                        <span class=\"content\">
                         Ya existe ese nombre de usuario<br><br>
                         <a href=\"#\" onClick=\"javascript:history.back();\">Regresar</a>
                        </span>
                       </center><br>
                      ";
        else{
                $qry_update = "INSERT INTO usuario ( login, password, rol )
                               VALUES ( '".$_POST['sUser']."' ,
                                        '".$_POST['sPass']."' ,
                                        '".$_POST['sTipo']."'    )
                              ";

                $result = @pg_exec( $conn, $qry_update );

                if ($result === false){
                        echo "Ocurri&oacute; un error";
                        exit;
                }else{
                       print "<form name=\"frm_regresar\" action=\"listar_usuarios.php\"
method=\"post\">";
                        print "<br><br><center>";
                        print "<img src=\"images/saved.gif\"><br>";
                        print "<span class=\"content\">Cambios Efectuados</span><br><br>";
                       print "<input type=\"hidden\" name=\"txtRole\"
value=\"".$_POST['txtRole']."\">";
                       print "<input type=\"submit\"
                                     value=\"Regresar\"
                                     class=\"inputbutt\">";
                        print "</center><br><br>";
                       print "</form>";
                }
        }

}

include ("desconectar.php");
include ("pie.php");

?>
- - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
```

borrar_usuario.php

```
<?php
include ("conectar.php");

$qry_elimina = "DELETE FROM usuario
                WHERE id_usuario=$_GET[id]";

//print $qry;

$result = @pg_Exec($conn, "$qry_elimina");

if($result)
{
  header ("Location: listar_usuarios.php?ver=x");
  exit;
}
else{
        print "Ocurri&oacute; un error";
}
include ("desconectar.php");

?>
- - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
```

checkdiskspace.php

```
<?
$titulo_pagina="Monitoreo de Espacio en Disco";
include ("encabezado.php");
include ("funciones.php");
$particion_seleccionada = $_GET['part'];
?>

<html>
 <head>
  <META HTTP-EQUIV="REFRESH"
CONTENT="<?=$global_refresco?>;URL=<?=$_SERVER['PHP_SELF']?>?part=<?=$particion_seleccionada?>
">
 </head>
 <body>

<?
print "<br><center><span class=\"logtxt\">Monitoreo del Espacio en Disco</span></center><br>";

$porcentaje_valido = 80;
$separador = "!";

//Selecciona las particiones
$particion = exec ( "df | awk {'print $6'}" , $a_particiones );

//Muestra las particiones en un cuadro
print "<center>";
print "<br>";
print "<select onchange=\"if(options[selectedIndex].value){location =
options[selectedIndex].value}\" class=content>";
$i=0;
foreach ( $a_particiones as $particion ){
        if ($i == 0){
                print "<option value=\"checkdiskspace.php\">PARTICIONES</option>";
        }
        else{
                if ( $particion_seleccionada == $particion )
                        print "<option value=\"checkdiskspace.php?part=$particion\"
selected>$particion</option>";
                else
                        print "<option
value=\"checkdiskspace.php?part=$particion\">$particion</option>";
        }
        $i++;
```

```
}
print "</select>";

print "<br>";

$temp = exec ("df | awk {'print $5 \"$separador\" $6'} | grep $particion_seleccionada");

$a_temp = explode ( $separador, $temp );

$a_temp2 = explode ( "%", $a_temp[0] );

$porcentaje_espacio = $a_temp2[0];

if ( is_numeric($porcentaje_espacio) ){
        print "<br><span class=content>Porcentaje de espacio ocupado <br>en la
partici&oacute;n seleccionada </span><br>";
        if ( $porcentaje_espacio < $porcentaje_valido )
                print "<img src=images/statusOK.gif alt=\"OK\"><br><br>";
        else
                print "<span class=content>es mayor al $porcentaje_valido %</span><br><a
href=\"mailto:richard_site@planeacion.unam.mx\"><img src=images/warning.gif
border=\"0\"></a><br>";
        print "</center>";
        print "<table align=\"center\" width=\"100%\">";
        print "<tr><td width=\"35%\"> </td><td width=\"50%\">";

        //Barra de la grafica
        print "<div class=\"graph\">";
        print "<strong class=\"bar\" style=\"width:".$a_temp[0].";\">";
        print "".$a_temp[0]."";
        print "</strong>";
        print "</div>";
        //Barra de la grafica

        print "</td><td width=\"35%\"> ";
        print "</td></tr>";
        print "</table>";
}

print "<br>";
print "<center>";
print "<input type=\"button\" value=\"Salir\" onclick='document.location.href=\"index.php\"'
class=\"inputbutt\" ><br>";
print "</center>";
print "<br><br>";

include ("pie.php");
?>
- - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
```

conectar.php

```
<?
//Variables de conexion
$host     = 'localhost';
$puerto   = 5432;
$base     = 'diplomado_ricardo';
$usuario  = 'richard_site';
$password = 'diplo*2007';


//Conexion a PostgreSQL
$conn = pg_connect( " host=$host port=$puerto dbname=$base user=$usuario password=$password
");

//Verificar conexion
if($conn){
  //echo "Conexion establecida a: ". pg_host($conn)."<br>\n";
}else{
  echo pg_last_notice($conn);
```

```
  exit;
}


?>
```
- -

contactanos.php

```
<?php
$titulo_pagina="Cont&aacute;ctanos";
include ("encabezado.php");
?>

<table align="center">
 <tr WIDTH=100% HEIGHT=100%>
  <td align=center>
   <br>

   <span class="logtxt">Cont&aacute;ctanos</span>
   <br><br>

   <table>
    <tr>
     <td valign="top"><img src="images/sobre_contacto.gif"></td>
     <td><span class="content"><b>Direcci&oacute;n</b><br><br>Puerto Libertad #33 Col.
Jardines de Casanueva<br>Ecatepec de Morelos, Estado de M&eacute;xico<br><br><br></span></td>
    </tr>
    <tr>
     <td valign="top"><img src="images/tel_contacto.gif"></td>
     <td><span class="content"><b>Tel&eacute;fono</b><br><br>5776 2643<br><br><br></span></td>
    </tr>
    <tr>
     <td valign="top"><img src="images/arroba_contacto.gif"></td>
     <td><span class="content"><b>Email</b><br><br><a
href="mailto:richard_site@planeacion.unam.mx">richard_site@planeacion.unam.mx</a><br><br><br><
/span></td>
    </tr>
   </table>

  </td>
 </tr>
</table>

<br><br>

<?php
include ("pie.php");
?>
```
- -

content.css

```
.content {
        font-family: Verdana, Arial, Helvetica, sans-serif;
        font-size: 10px;
        font-weight: normal;
        color: #2D4842;
}

.inputbox {
        font-family: Verdana, Arial, sans-serif;
        font-size: 10px;
        color: #004040;
        background-color: #FFFFFF;
        border: 1px solid #AEE47C;
        font-style: normal;
        font-weight: normal;
        font-variant: normal;
```

```
}

.logtxt {
	font-family: Verdana, Arial, Helvetica, sans-serif;
	font-size: 11px;
	color: #FF6600;
	font-weight: bold;
}

.headcolumn {
	font-family: Verdana, Arial, Helvetica, sans-serif;
	font-size: 11px;
	color: #aaFFFF;
	font-weight: bold;
}

.inputbutt {
	font-family: Verdana, Arial, Helvetica, sans-serif;
	font-size: 10px;
	color: #F94F31;
}

.underline {
	font-family: Verdana, Arial, Helvetica, sans-serif;
	font-size: 10px;
	color: #EF4B05;
}

A:link {
	TEXT-DECORATION: none
}

A:active {
	TEXT-DECORATION: none
}

A:visited {
	TEXT-DECORATION: none
}

A:hover {
	TEXT-DECORATION: underline
}

.underline1 {
	font-family: Verdana, Arial, sans-serif;
	font-size: 10px;
	font-style: normal;
	font-weight: normal;
	font-variant: normal;
	color: #FFFFFF;
}

.underline2 {
	font-family: Verdana, Arial, sans-serif;
	font-size: 10px;
	font-style: normal;
	font-weight: normal;
	font-variant: normal;
	color: #000000;
}

.contact_box {
	font-family: Verdana, Arial, Helvetica, sans-serif;
	font-size: 10px;
	font-weight: normal;
	color: #154760;
	background-color: #FFFCF9;
	border: 1px solid #006F00;
}
```

```
/*Estilo para las graficas*/
.graph {
        position: relative; /* IE is dumb */
        width: 200px;
        border: 1px solid #B1D632;
        padding: 2px;
}

.graph .bar {
        display: block;
        position: relative;
        background: #B1D632;
        text-align: center;
        color: #333;
        height: 2em;
        line-height: 2em;
       font-family: Verdana, Arial, Helvetica, sans-serif;
       font-size: 10px;
       font-weight: normal;
}

.graph .bar span {
       position: absolute;
       left: 1em;
}
- - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
```

desconectar.php

```
<?
//Cerrar la conexion
pg_close($conn);
?>
- - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
```

documentos.php

```
<?php
$titulo_pagina="Documentos";
include ("encabezado.php");
?>

<br>
<center><span class="logtxt">Documentos</span></center>
<br><br>

<table align="center" border="0">
 <tr>

  <td >

   <br>
<ul>
<li><span class=content><b>Informe de Actividades y Contenidos del<br>Diplomado de
Administraci&oacute;n de Bases de Datos</b><br><br><center>
<a href="documentos/TrabajoDeTitulacionRicardoNavarroLopez.doc"><img src="images/word.gif"
border="0"><br>[49.00 MB]</a><br><br><br>
<a href="documentos/TrabajoDeTitulacionRicardoNavarroLopez.pdf"><img src="images/pdf.gif"
border="0"><br>[6.86 MB]</a></center>
</span>
<br><br>

<li>   <span class=content><b>C&oacute;digo del SISCOE Version 1.1</b><br><center>
<a href="documentos/siscoe.tgz"><img src="images/tgz.gif" border="0"></a><br>[16
MB]<br><br><br></center>
<br><br></span>

<li>   <span class=content><b>Reconstrucci&oacute;n de la Base de Datos</b><br><center>
```

```
<a href="documentos/siscoe.psql"><img src="images/txt.gif" border="0"><br>[0.21
MB]</a><br><br><br></center>
</span>
</ul>

   <p>

  </td>

  <td width="80"> </td>

  <td valign="top">
   <img src="images/documents.gif" width="80">
  </td>

 </tr>
</table>

<br><br><br>

<?php include ('pie.php'); ?>
- - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
```

editar.php

```
<?php
$titulo_pagina="Administraci&oacute;n de Errores - Soluci&oacute;n del Error";
include ("encabezado.php");
include ("conectar.php");
include ("funciones.php");

$css1 = "class=\"headcolumn\" align=\"left\" bgcolor=\"#5FB90D\"";
$css2 = "class=\"content\" bgcolor=\"white\"";
$css3 = "class=\"inputbox\"";
$css4 = "class=\"underline\" align=\"center\" ";

if ( !isset($_POST['enviado']) ){

$qry= "SELECT hora,
              fecha,
              tipo,
              tipo_error,
              descripcion,
              numero_error,
              severidad,
              estado,
              status_revision,
              solucion
       FROM bitacora
       WHERE id_bitacora=".$_GET['id'];

$result = pg_exec( $conn, $qry );

print "<br><center><span class=\"logtxt\">Administraci&oacute;n de Errores <br> -
Soluci&oacute;n del Error -</span></center><br>";
print "<form name=\"frm_agregar\" action=\"".$_SERVER['PHP_SELF']."\" method=\"post\">";
print "<table border=\"0\" align=\"center\">";

$i=1;
while( $array_agendados=pg_fetch_array($result) ){
        print "<tr>";
        print "<td $css1>Fecha</td>";
        echo "<td $css2>".$array_agendados['fecha']."</td></tr>";

        print "<tr>";
        print "<td $css1>Hora</td>";
        echo "<td $css2>".$array_agendados['hora']."</td></tr>";

        print "<tr>";
        print "<td $css1>Tipo</td>";
```

```
        echo "<td $css2>".$array_agendados['tipo']."</td></tr>";

        print "<tr>";
        print "<td $css1>Error</td>";
        echo "<td $css2>".$array_agendados['tipo_error']."</td></tr>";

        print "<tr>";
        print "<td $css1 width=\"60\"># Error</td>";
        echo "<td $css2>".$array_agendados['numero_error']."</td></tr>";

        print "<tr>";
        print "<td $css1>Severidad</td>";
        echo "<td $css2>".$array_agendados['severidad']."</td></tr>";

        print "<tr>";
        print "<td $css1>Estado</td>";
        echo "<td $css2>".$array_agendados['estado']."</td></tr>";

        print "<tr>";
        print "<td $css1>Informaci&oacute;n</td>";
        echo "<td $css2>".$array_agendados['descripcion']."</td></tr>";

        print "<tr>";
        print "<td $css1>Status Revisi&oacute;n</td>";
        echo "<td $css2 align=\"center\">";
        if ($array_agendados['status_revision']== "t")
                echo "<input type=\"checkbox\" name=\"bStatusRevision\" checked>";
        else
                echo "<input type=\"checkbox\" name=\"bStatusRevision\" >";
        echo "<br>(Marcado=Solucionado;Sin Marca=No Solucionado)</td></tr>";

        print "<tr>";
        print "<td $css1>Descripci&oacute;n Soluci&oacute;n</td>";
        echo "<td $css2><textarea cols=\"40\" rows=\"6\"
name=\"sSolucion\">".$array_agendados['solucion']."</textarea></td></tr>";

        $i++;
}
print "</table>";
print "<input type=\"hidden\" name=\"sId\" value=\"".$_GET['id']."\">";
print "<input type=\"hidden\" name=\"enviado\" value=\"1\">";
print "<center>";
print "<input type=\"button\" value=\"Regresar\"
onclick='document.location.href=\"mostrar_y_cambiar.php\"' class=\"inputbutt\" >";
print " <input type=\"submit\" class=\"inputbutt\" value=\"Guardar\">";
print "</center>";
print "</form>";
print "<br>";
}else{
        $qry_update = "UPDATE bitacora
                        SET solucion = '". $_POST['sSolucion'] ."' ";

        if ( $_POST['bStatusRevision'] == "on" )
                $qry_update .= ", status_revision = TRUE ";
        else
                $qry_update .= ", status_revision = FALSE ";

        $qry_update .= " WHERE id_bitacora = ". $_POST['sId'];

        $result = @pg_exec( $conn, $qry_update );

        if ($result === false){
                echo "Ocurri&oacute; un error";
                exit;
        }else{
                print "<br><br><center>";
                print "<img src=\"images/saved.gif\"><br>";
                echo "<span class=\"content\">Cambios Efectuados</span><br><br>";
                print "<input type=\"button\" value=\"Regresar\"
onclick='document.location.href=\"mostrar_y_cambiar.php\"' class=\"inputbutt\">";
                print "</center><br><br>";
```

```
        }

}

include ("desconectar.php");
include ("pie.php");

?>
- - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
```

editar_usuario.php

```
<?php
$titulo_pagina="Administraci&oacute;n de Usuarios - Cambiar datos";
include ("encabezado.php");
include ("conectar.php");
include ("funciones.php");

$css1 = "class=\"headcolumn\" align=\"left\" bgcolor=\"#5FB90D\"";
$css2 = "class=\"content\" bgcolor=\"white\"";
$css3 = "class=\"inputbox\"";
$css4 = "class=\"underline\" align=\"center\" ";

$array_tipo_usuario = array("sa_role","sso_role", "oper_role");

if ( !isset ( $_POST['enviado'] ) ) {

        $qry= "SELECT id_usuario,
                      login,
                      password,
                      rol
               FROM usuario
               WHERE id_usuario = ".$_GET[id]."";

        $result = pg_exec( $conn, $qry );

        print "<br><center><span class=\"logtxt\">Administraci&oacute;n de
Usuarios</span><br><br>";
        print "<span class=\"logtxt\">- Cambiar datos -</span></center><br>";

        print "<table border=\"0\" align=\"center\">";

        print "<form name=\"frm_actualizar\" action=\"".$_SERVER['PHP_SELF']."\"
method=\"post\">";

        $i=1;
        while( $array_usuarios=pg_fetch_array($result) ){
                print "<tr>";
                print "<td $css1>Usuario:</td>";
                print "<td $css2>".$array_usuarios['login']."</td>";
                print "</tr>";

                print "<tr>";
                print "<td $css1>Contrase&ntilde;a:</td>";
                print "<td $css2><input name=\"sPass\" type=\"password\"
value=\"".$array_usuarios['password']."\"></td>";
                print "</tr>";

                print "<tr>";
                print "<td $css1>Tipo de Usuario: </td>";
                print "<td $css2>";
                print "<select name=\"sTipo\">";
                foreach(  $array_tipo_usuario as $tipo_usuario ){
                        if ( $tipo_usuario == $array_usuarios['rol'] )
                                print "<option selected> $tipo_usuario </option>";
                        else
                                print "<option> $tipo_usuario </option>";
                }
                print "</select>";
                print "</td>";
```

```
            print "</tr>";

            $i++;
            print "<input type=\"hidden\" name=\"sId\"
value=\"".$array_usuarios['id_usuario']."\" >";
        }
        print "</table>";

        print "<center>";
        print "<input type=\"hidden\" name=\"enviado\" value=\"1\" >";
        print "<input type=\"button\" value=\"Regresar\" onclick='history.back()'
class=\"inputbutt\" > ";
        print "<input type=\"submit\" value=\"Guardar\" class=\"inputbutt\" > ";
        print "</center>";
        print "<br>";
}else{

        $qry_update = "UPDATE usuario
                       SET rol      = '".$_POST['sTipo']."' ,
                           password = '".$_POST['sPass']."'
                       WHERE id_usuario = ".$_POST['sId']."";

        $result = @pg_exec( $conn, $qry_update );

        if ($result === false){
                echo "Ocurri&oacute; un error";
                exit;
        }else{
                print "<form name=\"frm_regresar\" action=\"listar_usuarios.php\"
method=\"post\">";
                print "<br><br><center>";
                print "<img src=\"images/saved.gif\"><br>";
                print "<span class=\"content\">Cambios Efectuados</span><br><br>";
                print "<input type=\"hidden\" name=\"txtRole\" value=\"sso_role\">";
                print "<input type=\"submit\"
                              value=\"Regresar\"
                              class=\"inputbutt\">";
                print "</center><br><br>";
                print "</form>";
        }

}

include ("desconectar.php");
include ("pie.php");

?>
```

- -

encabezado.php

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
"http://www.w3.org/TR/html4/loose.dtd">
<html>
 <head>
  <title><?=$titulo_pagina?></title>
  <meta http-equiv="Content-Type" content="text/html; charset=iso-8859-1">
  <script language="JavaScript" src="mmswap.js"></script>
  <link href="content.css" rel="stylesheet" type="text/css">
  <style type="text/css">
  <!--
   .style5 {
          color: #FF6600
           }
   body {
          margin-left: 0px;
          margin-top: 0px;
          margin-right: 0px;
          margin-bottom: 0px;
        }
```

```
  -->
  </style>
 </head>

 <body onLoad="MM_preloadImages('images/ttd_comp_004_mo_08.jpg',
                                'images/ttd_comp_004_mo_09.jpg',
                                'images/ttd_comp_004_mo_10.jpg',
                                'images/ttd_comp_004_mo_11.jpg',
                                'images/ttd_comp_004_mo_12.jpg')" background="images/bg.gif">
  <table width="759"  border="0" align="center" cellpadding="0" cellspacing="0">

   <!-- Fila Titulo -->
   <tr align="left" valign="top">
    <td width="5" align="left" valign="top" background="images/ttd_comp_004_01.jpg"
scope="col"><img src="images/ttd_comp_004_01.jpg" width="5" height="47"></td>
    <td width="200" scope="col"><img src="images/ttd_comp_004_02.jpg" width="200"
height="47"></td>
    <td width="551" background="images/ttd_comp_004_03.jpg" scope="col"></td>
    <td width="4" background="images/ttd_comp_004_04.jpg" scope="col"></td>
   </tr>
   <!-- Termina Fila Titulo -->


   <!-- Fila Botones y Eslogan -->
   <tr>
    <td width="5" align="left" valign="top" background="images/ttd_comp_004_06.jpg"
scope="col"><img src="images/ttd_comp_004_06.jpg" width="5" height="99"></td>
    <td scope="col"><img src="images/ttd_comp_004_07.jpg" width="200" height="99"></td>
    <td width="551" align="left" valign="top" scope="col">

     <!-- Botones Menu-->
     <table width="100%"  border="0" align="center" cellpadding="0" cellspacing="0">
      <tr align="left" valign="top">
       <td width="111" align="center" scope="col"><a href="index.php"
onMouseOut="MM_swapImgRestore()"
onMouseOver="MM_swapImage('Image6','','images/ttd_comp_004_mo_08.jpg',1)"><img
src="images/ttd_comp_004_08.jpg" alt="Inicio" name="Image6" width="111" height="23"
border="0"></a></td>
       <td width="110" align="center" scope="col"><a href="integrantes.php"
onMouseOut="MM_swapImgRestore()"
onMouseOver="MM_swapImage('Image7','','images/ttd_comp_004_mo_09.jpg',1)"><img
src="images/ttd_comp_004_09.jpg" alt="Integrantes" name="Image7" width="110" height="23"
border="0"></a></td>
       <td width="110" align="center" scope="col"><a href="documentos.php"
onMouseOut="MM_swapImgRestore()"
onMouseOver="MM_swapImage('Image8','','images/ttd_comp_004_mo_10.jpg',1)"><img
src="images/ttd_comp_004_10.jpg" alt="Documentos" name="Image8" width="110" height="23"
border="0"></a></td>
       <td width="110" align="center" scope="col"><a href="acercade.php"
onMouseOut="MM_swapImgRestore()"
onMouseOver="MM_swapImage('Image9','','images/ttd_comp_004_mo_11.jpg',1)"><img
src="images/ttd_comp_004_11.jpg" alt="Acerca de" name="Image9" width="110" height="23"
border="0"></a></td>
       <td width="110" align="center" scope="col"><a href="contactanos.php"
onMouseOut="MM_swapImgRestore()"
onMouseOver="MM_swapImage('Image10','','images/ttd_comp_004_mo_12.jpg',1)"><img
src="images/ttd_comp_004_12.jpg" alt="Cont&aacute;ctanos" name="Image10" width="110"
height="23" border="0"></a></td>
      </tr>
      <tr align="left" valign="top" background="images/ttd_comp_004_15.jpg"><td
colspan="5"><img src="images/ttd_comp_004_15.jpg" width="551" height="76"></td>
      </tr>
     </table>
     <!-- Termina Botones Menu -->

    </td>
    <td width="13%" align="left" valign="top" background="images/ttd_comp_004_13.jpg"
scope="col"><img src="images/ttd_comp_004_13.jpg" width="4" height="99"></td>
   </tr>
   <!-- Termina Fila Botones y Eslogan -->
```

```
   <!-- Fila Barra separacion Menu -->
   <tr>
    <td background="images/ttd_comp_004_16.jpg" scope="col"><img
src="images/ttd_comp_004_16.jpg" width="5" height="31"></td>
    <td scope="col"><img src="images/ttd_comp_004_17.jpg" width="200" height="31"></td>
    <td colspan="2" align="left" valign="top" background="images/ttd_comp_004_19.jpg"
scope="col"><img src="images/ttd_comp_004_18.jpg" width="82" height="31"><img
src="images/ttd_comp_004_19.jpg" width="2" height="31"></td>
   </tr>
   <!-- Termina Fila Barra separacion Menu -->


   <!-- Fila Contenido -->
   <tr>
    <td colspan="4" scope="col">
     <table width="100%"  border="0" align="center" cellpadding="0" cellspacing="0">
      <tr>
       <td width="1" align="left" bgcolor="#008000" scope="col"><img src="images/spacer.gif"
width="1" height="1"></td>
       <td width="10" align="left" valign="top" bgcolor="#E9FFD2" scope="col"><p> </p>
        <p> </p>
        <p><img src="images/spacer.gif" width="5" height="1"></p></td>
       <td scope="col" bgcolor="#E9FFD2">
       <!--Contenido-->
```

- -

funciones.php

```
<?php

//Variables globales
$global_archivo_log = "/opt/sybase-11.9.2/logs/SQLSRV.log";
//$global_archivo_log = "/opt/sybase/install/errorlog";
$global_refresco = "10";
$global_url = "http://calypso.planeacion.unam.mx/~richard_site";

function getline( $file ){
        $texto_a_buscar = "Error";
        $separador1     = "  ";
        $separador2     = " ";
        $separador3     = ":";
        $separador4     = ",";

        $aErrores = array();

        //Checar si existe el archivo
        if ( !file_exists($file) )
                return '';

        //Abrir archivo para lectura
        $fp = @fopen($file, "rb");
        if (!$fp)
                return '';

        //Leer el archivo
        while (!@feof($fp)) {

                //Leer linea y almacenarla en $bufer
                $bufer = fgets($fp, 1024);


                //Si en la linea se encuentra el texto buscado
                if(substr_count($bufer,$texto_a_buscar)) {

                        $errores .= $bufer;

                        //Divide la linea por cada separador y lo almacena en $a_errores
                        $a_errores = explode ($separador1,$bufer);
```

```
                        //Divide la posicion 0 del arreglo a_errores por cada separador2 y lo
almacena en $a_codigoerror1
                        $a_codigoerror1 = explode ( $separador2  , $a_errores[0] );

                        //Selecciona un desplegado por cada tipo de error
                        //hasta ahora se han localizado los errores "server" y "kernel"
                        switch($a_codigoerror1[2]){

                                //Desplegado para el error tipo "server"
                                case "server":


                                        //Divide la fecha  del arreglo a_codigoerror1 por cada
$separador3
                                        //y lo almacena en $a_codigoerror0
                                        $a_codigoerror0 = explode ( $separador3  ,
$a_codigoerror1[0] );

                                        //Divide la posicion 1 del arreglo a_errores por cada
$separador2
                                        //y lo almacena en $a_codigoerror2
                                        $a_codigoerror2 = explode ( $separador2  ,
$a_errores[1] );

                                        $aErrores[]=array("prefijo_error"=> $a_codigoerror0[0],
                                                          "fecha"        => $a_codigoerror0[3],
                                                          "hora"         => $a_codigoerror1[1],
                                                          "tipo"         => $a_codigoerror1[2],
                                                          "severidad"    =>
number_format($a_codigoerror2[3]),
                                                          "numero_error" =>
number_format($a_codigoerror2[1],0,'',''),
                                                          "estado"       =>
$a_codigoerror2[5]);
                                break;

                        //Desplegado para el error tipo "kernel"
                                case "kernel":

                        //Divide la fecha  del arreglo a_codigoerror1 por cada $separador3
                        //y lo almacena en $a_codigoerror0
                                        $a_codigoerror0 = explode ( $separador3  ,
$a_codigoerror1[0] );

                        //Divide la posicion 3 del arreglo a_errores por cada $separador3
                        //y lo almacena en $a_codigoerror
                                        $a_codigoerror2 = explode ( $separador3  ,
$a_errores[1] );
                                        $aErrores[]=array("prefijo_error"=>$a_codigoerror0[0],
                                                          "fecha"        =>$a_codigoerror0[3],
                                                          "hora"         =>$a_codigoerror1[1],
                                                          "tipo"         =>$a_codigoerror1[2],
                                                          "tipo_error"   =>$a_codigoerror2[0],
                                                          "descripcion"  =>$a_codigoerror2[1]);
                                break;
                        }
                }
        }

        //Cierra el archivo
        @fclose($fp);

        //Regresa las lineas con el texto buscado
        return $aErrores;

}//Finaliza funcion getline

?>
```

index.php

```
<?php
$titulo_pagina="Inicio";
include ("encabezado.php");
?>


<!-- Titulo -->
<br>
<center>
 <span class="logtxt">
  Sistema de Seguimiento y Control de Errores <br>
  del Manejador de Bases de Datos <br>
  SYBASE
 </span>
</center>

<!-- Tabla Contenido -->
<table align="center" width=100% height=100% cellspacing=0 border=0>
 <tr>
  <td valign="top" align="center">

   <?php
    //Si no se ha ingresado el nombre de usuario
    if( !isset ($_POST['txtUsuario']) ){
   ?>

   <form name="forma" action="index.php"  method="post">

    <table border="0">
     <tr>
      <td colspan="2" align="center">
       <br><span class="content">Identif&iacute;cate</span><br><br>
      </td>
     </tr>

     <tr>
      <td align="right">
       <span class="logtxt">Login: </span>
      </td>
      <td>
       <input type="textbox" name="txtUsuario" class="inputbox">
      </td>
     </tr>

     <tr>
      <td align="right">
       <span class="logtxt">Password: </span>
      </td>
      <td>
       <input type="password" name="txtPassword" class="inputbox">
      </td>
     </tr>

     <tr>
      <td colspan="2" align="center">
       <input type="submit" class="inputbutt" value="   Ingresar   ">
      </td>
     </tr>

    </table>

   </form>

   <?php
    //Si el nombre de usuario ya fue ingresado
    }else {
   ?>

   <br>
```

```
   <table align="center" bgcolor="white" cellspacing="0" border="0" height="70%">
    <tr>
     <td Evalign="center" align="center" width="300" height="200">

      <?php
       include ("conectar.php");
       $query="SELECT id_usuario, login, password, rol FROM usuario
               WHERE login LIKE '".$_POST['txtUsuario']."' AND
                     password LIKE '".$_POST['txtPassword']."'";

       $result = pg_exec($conn, $query);

       //Si la autenticacion es exitosa
       if(pg_num_rows($result)>0){

        print "<span class=\"content\">
                Bienvenido(a)
                <br>
                <font color=blue> ".$_POST['txtUsuario']." </font>
                <br><br>
                <form name=\"ingresa\" action=\"";


        //Mostrar contenido dependiendo del rol del usuario
        $array_rol=pg_fetch_array($result);
        $rol = $array_rol['rol'];
        $id_usuario = $array_rol['id_usuario'];

        switch($rol){
         case "oper_role" :
          $mensajeboton = "Monitoreo de Errores";
          $mensajeboton2 = "Monitoreo de Espacio en Disco";
          print "mostrar_listado.php";
          break;

         case "sso_role"  :
          $mensajeboton = "Administrar Usuarios";
          print "listar_usuarios.php";
          break;

         case "sa_role"   :
          $mensajeboton = "Administrar Errores";
          print "mostrar_y_cambiar.php";
          break;

         default   :
          print "index.php";

        }
        print "\" method=\"post\">";
        print "  <input type=\"hidden\" name=\"txtUsuario\"
value=\"".$_POST['txtUsuario']."\">";
        print "  <input type=\"hidden\" name=\"txtRole\" value=\"$rol\">";
        print "  <input type=\"hidden\" name=\"txtUsuario\" value=\"$id_usuario\">";
        print "  <input type=\"submit\" class=\"inputbutt\" value=\"        $mensajeboton
\">";
        print "<br><br>";
        if ( $rol == "oper_role" )
               print "  <input type=\"button\" class=\"inputbutt\"
onclick='window.location=\"checkdiskspace.php\"' value=\"Monitoreo del Espacio en Disco\">";
        print "<br><br>";
        print "<input type=\"button\" value=\"Salir\"
onclick='document.location.href=\"index.php\"' class=\"inputbutt\" >";
        print " </span>";
        print "</form>";


       }else{ //Si la autenticacion no es exitosa
        print "<span class=\"content\">
                <font color=red>
                 Acceso Denegado
```

```
                 <br>
                 Verifica tus datos...
                </font>
                <br><br>
                <input type=\"button\" value=\"Regresar\" class=\"inputbutt\"
onclick=\"javascript:history.back()\">
              ";
       }
       include ("desconectar.php");
      ?>

     </td>
    </tr>
   </table>
   <br>

   <?php
    }
   ?>

  </td>
 </tr>
</table>
<!-- Termina Tabla Contenido -->
<br><br>

<?php include ("pie.php"); ?>
```

- -

integrantes.php

```
<?php
$titulo_pagina="Manual de Usuario";
include ("encabezado.php");
?>


<table align="center">
 <tr WIDTH=100% HEIGHT=100%>
  <td align=center>
   <br>

   <span class="logtxt">Manual de Usuario</span>
   <br><br><br>


<span class=content><b>Manual de Usuario del SISCOE</b><br><br><center>
<a href="documentos/Manual.doc"><img src="images/word.gif" border="0"><br>[6.86
MB]</a><br><br><br>
<a href="documentos/Manual.pdf"><img src="images/pdf.gif" border="0"><br>[ 0.64
MB]</a></center>
</span>
<br><br>

  </td>
 </tr>
</table>

<br><br>

<?php
include ("pie.php");
?>
```

- -

listar_usuarios.php

```
<?php
$titulo_pagina="Administraci&oacute;n de Usuarios";

include ("encabezado.php");
include ("conectar.php");
include ("funciones.php");

//Seguridad de la aplicacion
if( $_POST['txtRole'] == "sso_role" OR $_GET['ver'] == "x"){
        //print "Usuario valido";
}else{
        print "<center>
                <img src=\"images/warning.gif\"><br>
                <span class=\"content\">Error en la autenticaci&oacute;n</span>
               </center>";
        include ("desconectar.php");
        include ("pie.php");
        exit;
}

$css1 = "class=\"headcolumn\" align=\"left\" bgcolor=\"#5FB90D\"";
$css2 = "class=\"content\" ";
$css3 = "class=\"inputbox\"";
$css4 = "class=\"underline\" align=\"center\" ";

$qry= "SELECT id_usuario,
              login,
              password,
              rol
       FROM usuario";

$result = pg_exec( $conn, $qry );

print "<br><center><span class=\"logtxt\">Administraci&oacute;n de
Usuarios</span></center><br>";

print "<table border=\"0\" align=\"center\">";

print "<tr>";
print "<td $css1>Usuario</td>";
print "<td $css1>Tipo de Usuario</td>";
print "<td $css1>Editar</td>";
print "<td $css1>Borrar</td>";
print "</tr>";

$i=1;
while( $array_usuarios=pg_fetch_array($result) ){
        print "<tr onMouseOver=\"this.style.backgroundColor='#efefef'\"
                        onMouseOut=\"this.style.backgroundColor='#ffffff'\"
                        bgcolor=\"#ffffff\">";
        print "<td $css2>".$array_usuarios['login']."</td>";
        print "<td $css2>".$array_usuarios['rol']."</td>";
        print "<td $css2 align=\"center\">";
        if ( $array_usuarios['login'] != "adminuser" )
                print "                 <a
href=\"editar_usuario.php?id=".$array_usuarios['id_usuario']."\">
                  <img src=\"images/editar.gif\" border=\"0\" alt=\"Editar\" title=\"Editar\">
                 </a>";
        else
                print " ";
        print "
               </td>";
        print "<td $css2 align=\"center\">";

        if ( $array_usuarios['login'] != "adminuser" )
                print "<a href=\"borrar_usuario.php?id=".$array_usuarios['id_usuario']."\"
onClick=\"javascript:return confirm('Esta seguro que desea eliminar al usuario
[".$array_usuarios['login']."] ?')\">
                  <img src=\"images/delete.gif\" border=\"0\" alt=\"Editar\" title=\"Borrar\">
```

```
                </a>";
        else
                print " ";
        print "
                </td>";
        print "</tr>";
        $i++;
}
print "</table>";

print "<center>";
print "<form name=\"frm_agrega\" action=\"agregar_usuario.php\" method=\"post\">";
if ($_GET['ver'] == "x")
        print "<input type=\"hidden\" name=\"txtRole\" value=\"sso_role\" > ";
else
        print "<input type=\"hidden\" name=\"txtRole\" value=\"".$_POST['txtRole']."\" > ";

print "<input type=\"button\" value=\"Salir\" onclick='document.location.href=\"index.php\"'
class=\"inputbutt\" > ";
print "<input type=\"submit\" value=\"Agregar\" class=\"inputbutt\" >";
//print "<input type=\"button\" value=\"Agregar\"
onclick='document.location.href=\"agregar_usuario.php\"' class=\"inputbutt\" >";
print "</form>";
print "</center>";
print "<br>";

include ("desconectar.php");
include ("pie.php");
?>
- - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
```

mostrar2.php

```
<?php

$css1 = "class=\"headcolumn\" align=\"center\" ";
$css2 = "class=\"content\"";
$css3 = "class=\"inputbox\"";
$css4 = "class=\"underline\" align=\"center\" ";

$qry= "SELECT id_bitacora,
              hora,
              fecha,
              tipo,
              tipo_error,
              descripcion,
              numero_error,
              severidad,
              estado,
              status_revision
       FROM bitacora
       ORDER BY fecha DESC,
                hora DESC";

$result = pg_exec( $conn, $qry );

print "<br><center><span class=\"logtxt\">Administraci&oacute;n de
Errores</span></center><br>";

print "<table border=\"0\" >";

if ( pg_num_rows ($result) > 0 ){
        //Encabezado columnas
        print "<tr bgcolor=\"#5FB90D\">";
        print "<td $css1>Fecha</td>";
        print "<td $css1>Hora</td>";
        print "<td $css1>Tipo</td>";
        print "<td $css1>Error</td>";
```

```
        print "<td $css1 width=\"60\"># Error</td>";
        print "<td $css1>Severidad</td>";
        print "<td $css1>Estado</td>";
        print "<td $css1>Informaci&oacute;n</td>";
        print "<td $css1>Revisado</td>";
        print "<td $css1>Editar</td>";
        print "</tr>";
}else{
        print "<center>
               <img src=\"images/statusOK.gif\"><br>
               <span class=\"content\">La Base de Datos no tiene errores</span>
              </center>";
}

$i=1;
while( $array_agendados=pg_fetch_array($result) ){
        echo "<tr onMouseOver=\"this.style.backgroundColor='#efefef'\"
                  onMouseOut=\"this.style.backgroundColor='#ffffff'\"
                  bgcolor=\"#ffffff\">";
        echo "<td $css2>".$array_agendados['fecha']."</td>";
        echo "<td $css2>".$array_agendados['hora']."</td>";
        echo "<td $css2>".$array_agendados['tipo']."</td>";
        echo "<td $css2>".$array_agendados['tipo_error']."</td>";
        echo "<td $css2 align=\"center\">".$array_agendados['numero_error']."</td>";
        echo "<td $css2 align=\"center\">".$array_agendados['severidad']."</td>";
        echo "<td $css2 align=\"center\">".$array_agendados['estado']."</td>";
        echo "<td $css2>".$array_agendados['descripcion']."</td>";

        echo "<td $css2 align=\"center\">";
        if ( $array_agendados['status_revision'] == "t" )
                echo "OK";
        else
                echo " ";
        echo "</td>";
        echo "<td $css2 align=\"center\"> <a
href=\"editar.php?id=".$array_agendados['id_bitacora']."\"><img src=\"images/editar.gif\"
border=\"0\" alt=\"Editar\" title=\"Editar\"></a></td>";
        echo "</tr>";
        $i++;
}
print "</table>";
print "<br>";

?>
- - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
```

mostrar.php

```
<?php

$css1 = "class=\"headcolumn\" align=\"center\" ";
$css2 = "class=\"content\"";
$css3 = "class=\"inputbox\"";
$css4 = "class=\"underline\" align=\"center\" ";

$qry= "SELECT hora,
              fecha,
              tipo,
              tipo_error,
              descripcion,
              numero_error,
              severidad,
              estado,
              status_revision
       FROM bitacora
       WHERE status_revision IS NOT TRUE
       ORDER BY fecha DESC,
                hora DESC";
```

```
$result = pg_exec( $conn, $qry );

print "<br><center><span class=\"logtxt\">Monitoreo de Errores</span></center><br>";

print "<table border=\"0\" align=\"center\">";

if ( pg_num_rows ($result) > 0 ){
//Encabezado columnas
print "<tr><td colspan=8 align=center><a href=\"mailto:richard_site@planeacion.unam.mx\"><img
src=images/warning.gif border=0></a></td></tr>";
print "<tr bgcolor=\"#5FB90D\">";
print "<td $css1>Fecha</td>";
print "<td $css1>Hora</td>";
print "<td $css1>Tipo</td>";
print "<td $css1>Error</td>";
print "<td $css1 width=\"60\"># Error</td>";
print "<td $css1>Severidad</td>";
print "<td $css1>Estado</td>";
print "<td $css1>Informaci&oacute;n</td>";
print "</tr>";
}else{
        print "<center>
                <img src=\"images/statusOK.gif\"><br>
                <span class=\"content\">La Base de Datos no tiene errores</span>
               </center>";
}

$i=1;
while( $array_agendados=pg_fetch_array($result) ){
        if ( $array_agendados['status_revision'] != "t"  ){
                echo "<tr onMouseOver=\"this.style.backgroundColor='#efefef'\"
                        onMouseOut=\"this.style.backgroundColor='#ffffff'\"
                        bgcolor=\"#ffffff\">";
                echo "<td $css2>".$array_agendados['fecha']."</td>";
                echo "<td $css2>".$array_agendados['hora']."</td>";
                echo "<td $css2>".$array_agendados['tipo']."</td>";
                echo "<td $css2>".$array_agendados['tipo_error']."</td>";
                echo "<td $css2 align=\"center\">".$array_agendados['numero_error']."</td>";
                echo "<td $css2 align=\"center\">".$array_agendados['severidad']."</td>";
                echo "<td $css2 align=\"center\">".$array_agendados['estado']."</td>";
                echo "<td $css2>".$array_agendados['descripcion']."</td>";
                echo "</tr>";
        }
        $i++;
}
print "</table>";
print "<br>";

?>
- - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
```

mostrar_listado.php

```
<?
$titulo_pagina="Monitoreo de Errores";
include ("encabezado.php");
include ("conectar.php");
include ("funciones.php");
?>

<html>
 <head>
  <title>Leer LOG</title>
  <META HTTP-EQUIV="REFRESH"
CONTENT="<?=$global_refresco?>;URL=<?=$global_url?>/mostrar_listado.php">
 </head>
 <body>
```

```
<span class="logtxt">Hora:</span>
<span class="content">
 <script>
  miFecha = new Date()
  document.write(miFecha.getHours() + ":" + miFecha.getMinutes() + ":" + miFecha.getSeconds())
 </script>
</span>

<?php
$aErrores = getline("$global_archivo_log");
$numErrores = count($aErrores);

print "<br><br>\n";

foreach ( $aErrores as $aError ){
        $iCampos =count($aError);
        $i=0;
        foreach( $aError as $sKey => $sValue ){
                $i++;

                $sCampos .=$sKey;
                if ( $iCampos > $i )
                        $sCampos .= ",";

                if   ( $sKey == fecha )
                  //                  $sValores.="to_date('".$sValue."','YYYY/MM/DD')";
                        $sValores.="'".$sValue."'";

                elseif( $sKey == hora  )
                        $sValores.="'".$sValue."'";
                else
                        $sValores.="'".addslashes($sValue)."'";

                if ( $iCampos > $i )
                        $sValores.=",";

        }
        $sFecha = $aError[fecha];
        $sHora  = $aError[hora];

        $qry1 = "INSERT INTO bitacora ($sCampos) VALUES ($sValores)";
        //      print $qry1;
        $sCampos = "";
        $sValores= "";

        $qry2 = "SELECT fecha,
                        hora,
                        tipo,
                        tipo_error,
                        descripcion,
                        numero_error,
                        severidad,
                        estado,
                        status_revision
                 FROM bitacora
                 WHERE fecha = '$sFecha' AND
                       hora = '$sHora'";

        $result = pg_exec($conn, $qry2);

        //Si existe el registro
        if( !pg_num_rows($result)>0){
                print "<center>
                        <img src=\"images/alert.gif\"><br>
                        <span class=\"content\">Ha ocurrido 1 error en la Base de Datos</span>
                       <center>
                       <hr>";
                pg_exec( $conn, $qry1);
        }
```

```
}
        include ("mostrar.php");
print "<center>";
print "<input type=\"button\" value=\"Regresar\" onclick='history.back()'
class=\"inputbutt\">";
print "<input type=\"button\" value=\"Salir\" onclick='document.location.href=\"index.php\"'
class=\"inputbutt\" ><br>";
print "</center>";
print "<br><br>";
?>

 </body>
</html>

<?
include ("desconectar.php");
include ("pie.php");
?>
- - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
```

mostrar_y_cambiar.php

```
<?
$titulo_pagina="Administraci&oacute;n de Errores";
include ("encabezado.php");
include ("conectar.php");
include ("funciones.php");
?>

<html>
 <head>
  <title>Leer LOG</title>
  <META HTTP-EQUIV="REFRESH"
CONTENT="<?=$global_refresco?>;URL=<?=$global_url?>/mostrar_y_cambiar.php">
 </head>
 <body>

<span class="logtxt">Hora:</span>
<span class="content">
 <script>
  miFecha = new Date()
  document.write(miFecha.getHours() + ":" + miFecha.getMinutes() + ":" + miFecha.getSeconds())
 </script>
</span>



<?php
$aErrores = getline("$global_archivo_log");
print "<br><br>\n";

foreach($aErrores as $aError){
        $iCampos =count($aError);
        $i=0;
        foreach($aError as $sKey => $sValue){
                $i++;

                $sCampos .=$sKey;
                if ( $iCampos>$i )
                        $sCampos .= ",";

                if   ( $sKey == fecha )
                        $sValores.="to_date('".$sValue."','YYYY-MM-DD')";

                elseif( $sKey == hora  )
                        $sValores.="'".$sValue."'";
                else
                        $sValores.="'".addslashes($sValue)."'";
```

```
                if ( $iCampos>$i )
                        $sValores.=",";

        }
        $sFecha = $aError[fecha];
        $sHora  = $aError[hora];

        $qry1 = "INSERT INTO bitacora ($sCampos) VALUES ($sValores)";

        $sCampos = "";
        $sValores= "";

        $qry2 = "SELECT fecha,
                        hora,
                        tipo,
                        tipo_error,
                        descripcion,
                        numero_error,
                        severidad,
                        estado,
                        status_revision
                 FROM bitacora
                 WHERE fecha = '$sFecha' AND
                       hora = '$sHora'";

        $result = pg_exec($conn, $qry2);

        //Si existe el registro
        if( !pg_num_rows($result)>0){
                print "<center><img src=\"images/alert.gif\"><br><span class=\"content\">Ha
ocurrido 1 error en la Base de Datos</span><center><hr>";
                pg_exec( $conn, $qry1);
        }

}
        include ("mostrar2.php");

print "<center><input type=\"button\" value=\"Salir\"
onclick='document.location.href=\"index.php\"' class=\"inputbutt\" ></center><br>";

?>

 </body>
</html>

<?
include ("desconectar.php");
include ("pie.php");
?>
- - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
```

pie.php

```
        <!--Termina Contenido-->
        </td>
        <td bgcolor="#E9FFD2" scope="col"> </td>
        <td width="1" bgcolor="#008000" scope="col"><img src="images/spacer.gif" width="1"
height="1"></td>
       </tr>
      </table>
     </td>
    </tr>
    <tr bgcolor="#5FB90D" align="center">
     <td height="21" colspan="4" scope="col">
      <a href="index.php" class="underline1" title="Inicio">Inicio</a>
      <span class="underline">| </span>
      <a href="integrantes.php" class="underline1" title="Manual">Manual</a>
```

```
      <span class="underline">|</span>
      <a href="documentos.php" class="underline1" title="Documentos">Documentos </a>
      <span class="underline">| </span>
      <span class="underline1"><a href="acercade.php" class="underline1" title="Acerca
de">Acerca de ... </a> </span>
      <span class="underline">|</span>
      <a href="contactanos.php" title="Contacto" class="underline1">Contacto </a>
     </td>
    </tr>
   </table>
  <p> </p>
 </body>
</html>
```

1. La función de un administrador de bases de datos es tener las habilidades y conocimientos que van más allá del uso de un solo Sistema Manejador de Bases de Datos Relacional, es necesario lo siguiente:

   - Comprensión de cómo se integra el manejador de bases de datos con los sistemas de tecnologías de información y la comunicación (TIC).

   - Realizar la planeación de los recursos y capacidades del hardware en el que va a ser almacenada la información mediante un modelado de datos relacional.

   - Comparar los beneficios, restricciones y costos que existen entre los diferentes programas de administración de bases de datos.

   - Optimizar el acceso a la información mediante el óptimo manejo del lenguaje de consultas estructurado (SQL) para extraer sólo la información necesaria en el menor tiempo.

   - Manejar en sistemas de información automatizado, el control de acceso, la administración, el ingreso, horarios de acceso, desagregación de funciones y control en la modificación de la información.

   - Comprensión de que el software de administración de bases de datos estará sobre un sistema operativo que lo envuelve, integrado a sus virtudes y limitantes debiendo conocerse los aspectos de seguridad y administración del mismo.

   - Habilidades de liderazgo y comunicación en reuniones efectivas que permitan integrar grupos de trabajo de alto desempeño.

   - Administrar las bases de datos, su instalación, creación, arranque, conexión, control de acceso, monitoreo y arreglo de los problemas que se presenten.

   - Optimización de los procesos de almacenamiento, asignación de espacio y asignación de memoria

   - Contemplar los desastre y la recuperación de la información.

- Prever los procesos de auditoria que son marcados en los estándares como lo es el COBIT[1] de ISACA[2].

- Considerar la funcionalidad de las aplicaciones futuras como lo son las bases de datos orientadas a objetos.

- Aprovechar la información en algo más que las consultas a la base de datos, si no en el conocimiento que se puede extraer de la información almacenada ocupando la minería de datos.

2. El software propietario, a pesar de que cumpla con la funcionalidad que es ofrecida a la hora de su venta, requiere de extensiones del producto necesarios para proporcionar una fluidez en el servicio e incrementar la facilidad de uso.

   Estas extensiones del producto, pueden ser tan variadas como pueden ser:

   - Un entorno gráfico para la consulta y la administración.

   - Herramientas de respaldo.

   - Manejo de espejeo.

   - Soporte vía correo electrónico, chat, teléfono o en sitio ya sea para la instalación o corregir problemas en la administración.

   - Consultoría en la administración y puesta a punto.

   Las extensiones generalmente no son consideradas a la hora de la adquisición del Sistema Manejador de Bases de Datos y a pesar de tener un costo adicional por lo que se sugiere su adquisición.

3. El software libre a pesar de que es generalmente gratuito no debe de ser considerado como un elemento de poca calidad. Demostrándose con el Sistema de Seguimiento y Control de Errores

---

[1] COBIT – Control OBjetives for Information and related Technology. Objetivos de control de la información y tecnología relacionada.

[2] ISACA - Information Systems Audit and Control Association. Asociación para la auditoria y control de sistemas de información.

que permite el ahorro de trabajo al administrador de bases de datos en un 80% en la resolución de errores demostrando que el software libre por un costo prácticamente nulo resuelve una de las extensiones que son necesarias por el Manejador de Bases de Datos de Sybase específicamente. Cumpliéndose la hipótesis de que con el objetivo de ayudar, a un bajo costo, a disminuir el tiempo en que se lleva encontrar una solución al ocurrir un error en el Sistema Manejador de Bases de Datos Relacional Sybase para asegurar la disponibilidad de este RDBMS propietario de costo de licencia aproximado de US$32,0000 por CPU. Poder documentar en la aplicación el error, para darle seguimiento a éste y tenerlo como referencia para un caso similar en el futuro sólo que puede complementarse que con requiere más trabajo para el administrador de la base de datos el monitoreo sugiriendo delegar esta tarea permitiendo el acceso solo a un monitor con pocos privilegios sobre las bases de datos.

Glosario

ACM. Association for Computing Machinery. ACM publica un prestigioso periódico académico, el Journal of the ACM, además de revistas para profesionales en el área de cómputo conocida también como Communications of the ACM.

Atributo. Es una columna en una tabla.

Back-end. Es la parte que procesa la entrada desde un front-end.

Benchmark. El benchmark es una técnica utilizada para medir el rendimiento de un sistema o componente de un sistema, frecuentemente en comparación con algún parámetro de referencia. También puede encontrarse como benchmarking, el cual se refiere específicamente a la acción de ejecutar un benchmark. La palabra benchmark es un anglicismo traducible al castellano como comparativa. Más formalmente puede entenderse que un benchmark es el resultado de la ejecución de un programa informático o un conjunto de programas en una máquina, con el objetivo de estimar el rendimiento de un elemento concreto o la totalidad de la misma, y poder comparar los resultados con máquinas similares. En términos de ordenadores, un benchmark podría ser realizado en cualquiera de sus componentes, ya sea CPU, RAM, tarjeta gráfica, etc. También puede ser dirigido específicamente a una función dentro de un componente, por ejemplo, la unidad de coma flotante de la CPU; o incluso a otros programas.

BDOO. Bases de Datos Orientadas a Objetos.

C. Lenguaje de programación de propósito general que ofrece economía sintáctica, control de flujo y estructuras sencillas y un buen conjunto de operadores. Este lenguaje ha sido estrechamente ligado al sistema operativo UNIX, puesto que fueron desarrollados conjuntamente. Sin embargo, este lenguaje no está ligado a ningún sistema operativo ni a ninguna máquina concreta. Se le suele llamar lenguaje de programación de sistemas debido a su

utilidad para escribir compiladores y sistemas operativos, aunque de igual forma se puede desarrollar cualquier tipo de aplicación.

Cardinalidad. Es el número de tuplas en una tabla.

CSV. Comma-separated values. Archivo de texto separado por comas, formato de datos delimitando los campos por el carácter coma (,) y a cada registro separado por un carácter de nueva línea.

DBA. Data Base Administrator; Administrador de Bases de Datos es la persona o grupo de personas encargadas del control general del DBMS.

DBMS. Data Base Management System. Sistema Gestor de Bases de Datos también conocido en su acrónimo latino como SGBD.

DDL. Data Definition Language; Lenguaje de Definición de Datos. Permite crear, modificar y eliminar estructuras de datos como: tablas, bases de datos, índices, etc.

DML. Data Manipulation Language; Lenguaje de Manipulación de Datos.

Dominio. Es el conjunto de valores de los cuales los atributos obtienen sus valores.

DSL. Data Sub Language; Sub Lenguaje de Datos. Combinación de dos lenguajes DDL y DML.

Front-end. Es la parte del software que interactúa con el usuario.

Grado. Es el número de atributos en una tabla.

GUI. Graphical User Interface; Interfaz Gráfica de Usuario. En el contexto del proceso de interacción persona-ordenador es el artefacto tecnológico de un sistema interactivo que posibilita, a través del uso y la representación del lenguaje visual, una interacción amigable con un sistema informático.

Java. Lenguaje de programación orientado a objetos desarrollado por Sun Microsystems a principios de los 90. A diferencia de los lenguajes de programación convencionales, que generalmente están diseñados para ser compilados a código nativo, Java es compilado en un

bytecode que es ejecutado (usando normalmente un compilador JIT), por una máquina virtual Java.

JIT. Compilación JIT, una técnica para mejorar el rendimiento de un intérprete en informática.

Llave. Es un atributo con una característica de relevancia para identificar la tupla.

Llave primaria. Es una llave con valores únicos, es decir, no ocurren más de una vez en el atributo.

PC. Personal Computer; Computadora Personal.

PHP. Lenguaje de programación usado generalmente para la creación de contenido para sitios Web. PHP es un acrónimo recurrente que significa "PHP Hypertext Pre-processor" (inicialmente PHP Tools, o, Personal Home Page Tools), y se trata de un lenguaje interpretado usado para la creación de aplicaciones para servidores, o creación de contenido dinámico para sitios Web.

RDBMS. Relational Database Management System. Sistema Gestor de Bases de Datos Relacionales.

Relación. Una definición simple es que se corresponde con una tabla y en ocasiones es preferible pensarlo de esta manera. La definición canónica es que una relación es el producto cartesiano de dos o varios dominios. Se puede decir que un dominio es un conjunto de valores escalares del mismo tipo, dónde un valor escalar es la mínima unidad semántica de información en el sentido de que son valores atómicos.

RPC. Remote Procedure Call. Llamada de procedimiento remoto. Es un protocolo que permite a un programa ejecutar código en otra máquina remota sin tener que preocuparse por las comunicaciones entre ambos.

SGBD. Sistemas Gestores de Bases de Datos más conocido en su acrónimo en inglés como DBMS.

SQL. El Lenguaje de Consulta Estructurado (Structured Query Language) es un lenguaje declarativo de acceso a bases de datos relacionales que permite especificar diversos tipos de operaciones sobre las mismas.

Tabla base. Es una relación autónoma a diferencia de las vistas y las tablas intermedias construidas a partir de una consulta.

TIC. Tecnologías de la Información y la Comunicación. Se encarga del estudio, desarrollo, implementación, almacenamiento y distribución de la información mediante la utilización de hardware y software como medio de sistema informático.

Tuplas. Registros, renglones o filas de una tabla o resultado de una consulta.

Vista. Es una relación virtual, que se construye a partir de tablas base o incluso otras vistas, formada por atributos de estas otras tablas de forma directa o como resultado de una consulta.

B I B L I O G R A F Í A

SOLIS Montes, Gustavo Adolfo. *Reingeniería de la Auditoria Informática*, (ed. Trillas), México 2002, p. 39-49.

ISACA. *CISA. Review Manual 2005,* USA 2004, p. 77-79, 122-127, 184 y 185.

COBIT. *Control Objectives Management Guidelines Maturity Models, USA 2005, V 4.0,* pp. 30-40, 73-80, 97-102, 103-106, 111-122, 143-150.

TURNO, María Luisa. *La vida y la percepción visual*, (Librería Universitaria), Madrid, 1998.

BOOCH, Grady. *Análisis y diseño orientado a objetos*, 2ª Edición. México, (Addison-Wesley Iberoamericana). 1996.

JACOBSON, Ivar. *Object oriented software engineering.* EE.UU, (Addison-Wesley). 1994.

RUMBAUGH, James y otros. *Object oriented modeling and design.* EE.UU, (Prentice-Hall). 1991.

BOOCH, Grady. *El Lenguaje Unificado de Modelado.* España, (Addison-Wesley Iberoamericana). 1999.

BOOCH, Grady. *El Lenguaje Unificado de Modelado. Manual de Referencia.* España, (Addison-Wesley Iberoamericana). 1999.

FOWLER, Martin. *UML Gota a Gota.* México, (Addison-Wesley), 1999.

JACOBSON, Ivar. *El Proceso Unificado de Desarrollo de Software.* España, (Addison-Wesley Iberoamericana). 2000.

MACIASZEK, L.A. *Requirements Analysis and System Design. Developing. Information Systems with UML.* (Addison-Wesley) 2001.

GARCÍA Arenas, María Isabel – Curso Comercio Electrónico 2ª Edición con PHP [en línea] <http://geneura.ugr.es/~maribel/php/>

ASP Tutor – Guía de ASP [en línea] <http://www.asptutor.com/asp/default.asp>

Aula Digital – Tutoriales de ASP [en línea] <http://www.auladigital.com>

Programación en Castellano – Tutoriales de ASP, Java [en línea] <http://www.programacion.com>

JavaHispano – Programación en Java [en línea] <http://www.javahispano.org>

William Yañez – Java Server Pages [en línea] <http://mipagina.cantv.net/williamyanez/jsp/>

PHP Sitio Oficial – Manual de PHP [en línea] <http://mx.php.net>

Solo ASP [en línea] <http://www.soloasp.com.ar>

ObjetctManagementGroup – Documentación de UML [en línea] <http://www.uml.org>

Rational Software Corporation – Recursos de UML [en línea] <http://www.rational.com>

Cetus Links Object Orientation – Ligas a componentes y objetos [en línea] <http://www.cetus-links.org>

The Object Agency – Orientado a la Tecnología Orientada a Objetos [en línea] <http://www.toa.com>

Object Orientation Tips – Tips a la Tecnología Orientada a Objetos [en línea] <http://ootips.org/>