UNIVERSIDAD NACIONAL AUTÓNOMA DE MÉXICO

FACULTAD DE ESTUDIOS SUPERIORES ARAGÓN

ADMINISTRACIÓN DE MÚLTIPLES BASES DE DATOS DESDE UNA INTERFAZ ÚNICA

T E S I N A

QUE PARA OBTENER EL TÍTULO DE:

INGENIERO EN COMPUTACIÓN

P R E S E N T A:

ANGEL CANALIZO MONROY

ASESOR: ING. SILVIA VEGA MUYTOY

NOVIEMBRE 2010

AGRADECIMIENTOS

Dedico este trabajo a la memoria de mi mamá, Carmen Monroy Santos, por todas las enseñanzas que me dejo, empezamos juntos este camino, y aunque ya no lo esta, en mi mente y mi corazón es y siempre será la luz que guie mi camino.

A mis hermanas, Norma y Laura que en todo momento me apoyan, son parte de todos mis triunfos y parte fundamental en mi vida.

A una persona muy especial, Yazmín, por el apoyo incondicional brindado, por crecer conmigo y creer en mi.

A todos los que directa o indirectamente han formado parte de mi crecimiento profesional, gracias.

ÍNDICE

INTRODUCCIÓN

El manejo de información es vital en la vida del ser humano, tanto para dejar huella de lo que se sucede día a día a las futuras generaciones, como para recuperar datos relevantes para las personas que dependen de la misma.

Desde que el ser humano ha tratado de explicarse su origen y ha buscado datos que lo guíen a descubrir y entender el pasado, ha encontrado pistas que al unirlas como rompecabezas, generan cierta información que ayudan a descifrar esos enigmas.

Los antepasados de una u otra manera, ingeniaron mecanismos para informar sobre lo que ocurría en sus tiempos; aunque, las formas de almacenamiento de información no eran las de hoy en día, fueron utilizadas pinturas en rocas o piedras talladas; y es precisamente este el inicio del almacenamiento de información. A través del tiempo fueron creadas las bases de datos, las cuales también han evolucionado a la par del ser humano.

A partir de la década de los 70's, con la evolución que han tenido tanto las computadoras como el internet, la información fluye de lado a lado en el mundo en cuestión de minutos, se puede hacer una transacción de dinero de un país a otro sin ningún problema, una reservación de un hotel o de un boleto de avión o concierto desde la comodidad del hogar, y todo eso es información que circula momento a momento por la red.

En el caso de los bancos y las transacciones de dinero, es importante resaltar que el manejo de la información se lleva a cabo desde una base de datos, quien accede a ella es sólo gente autorizada, las diferentes opciones que un sistema toma son casi siempre con respecto a información que se obtiene de la misma, existen mecanismos que aseguran la total privacidad de los datos que se les confían a esas empresas, las cuales dependen de un manejador de base de datos y del administrador de la misma, las consecuencias que tendría la violación de la privacidad de ésta, como el robo de cuentas bancarias o de datos personales serían extremadamente peligrosas.

Por otro lado, muchas veces se tiene la necesidad de tratar con este tipo de mecanismos sin ser un experto y siempre queda la duda de saber si se está haciendo correctamente un proceso que tal vez genere un resultado positivo a lo que se busca, pero que no necesariamente puede ser la mejor forma de resolverlo o manejarlo.

El presente trabajo, se divide en dos capítulos, en el primero se presenta un informe general de cada uno de los módulos del diplomado de Administración de Bases de Datos, se encuentra dividido en 12 temas, colocados en el orden en el que se encuentran en el temario; en el segundo capitulo se muestra el desarrollo de un sistema, donde se aplican los conocimientos adquiridos durante el diplomado.

I.- INFORME GENERAL DEL DIPLOMADO

I.I.- SISTEMA DE INFORMACIÓN Y MODELADO DE DATOS RELACIONAL

Hablar de información es común en la actualidad, pero pocas veces se analiza a fondo lo que este concepto significa y sobre todo lo que vale, y para ello es necesario conocer algunos conceptos básicos e importantes.

El dato, es la forma mínima en la que se representa algo que se quiere expresar, el cual puede estar conformado de uno o más caracteres.

La información también es un conjunto de caracteres, pero a diferencia del dato tiene sentido o dice algo que se puede asociar con un conocimiento previo. Ésta para ser considerada como tal, debe de cumplir con las siguientes características:

- Útil: Ya que si no es de utilidad entonces pierde su valía.
- Accesible: Debe poderse consultar todo el tiempo y con rapidez.
- Precisa: Debe ser lo más apegada a la realidad.
- Oportuna: Debe tenerse en el momento que se le necesite, antes o después ya no sirve.
- Clara: Debe ser entendible.
- Segura: Que sólo sea accedida por la gente adecuada.[1]

I.I.1.- BASES DE DATOS

"*Una base de datos es un deposito o contenedor de una colección de datos computarizados*"[2], es decir, es un lugar donde se almacena información de manera persistente, para poder ser accedida en el momento que se necesite, siendo su uso principal la toma de decisiones.

Se componen por:

- Hardware: Son los dispositivos físicos donde se almacena la información; por ejemplo los discos duros, las tarjetas perforadas, las memorias, etcétera, también los dispositivos físicos que interactúan con las bases de datos (PC, laptop, teléfonos, etcétera).

[1] Notas diplomado de Administracion de Bases de datos, DGSCA UNAM.
[2] Date, C. J. *Introducción a los sistemas de bases de datos.* Prentice Hall

- ❖ Software: Son los programas que ayudan a interactuar con las bases de datos.

- ❖ Datos: La información a guardar.

- ❖ Usuarios: Las personas que interactúan con la base de datos.

I.I.2.- MODELOS DE DATOS

"Es un conjunto de herramientas conceptuales que sirven para describir los datos, sus relaciones, su semántica y sus limitantes (Rango de valores permitido)"[1].

Los modelos mas usados son el jerárquico, de red, relacional y orientado a objetos.

- Modelo de datos jerárquico

  En este modelo se utilizan arboles para representar lógicamente los datos, por lo que no pueden existir ciclos, cada nodo depende de otro, a excepción del nodo raíz, del cual dependen todos.

  Otra de sus características es que sólo se pueden hacer relaciones uno a uno o uno a muchos y cuando se elimina un nodo padre, todos sus hijos se pierden.

- Modelo de datos de red

  Este modelo utiliza una estructura no lineal, es decir, un nodo padre puede tener muchos hijos y un nodo hijo puede tener muchos nodos padre.

  Se les conoce, al padre como propietario y al hijo como miembro.

- Modelo de datos relacional

  "La representación lógica de las entidades y sus relaciones se representa en tablas bidimensionales"[2]

  Utiliza conceptos importantes como son:

  - ❖ Atributo o campo: Así se le llama a cada uno de los campos que componen una tabla.

  - ❖ Registro: Es un conjunto de atributos, se les conoce también como tuplas.

[1] Los sistemas de información en empresa, Alberto Gómez Gómez y Nicolás de Abajo Martínez
[2] Diseño y programación de bases de datos, Ángel Covo Yera, Colección didáctica escolar

- ❖ Relación: Esta compuesta por un conjunto de registros y un conjunto de atributos.

- ❖ Clave: Es el identificador único de una tupla, se compone por 1 o mas atributos, por lo que en el modelo de datos relacional no puede haber tuplas repetidas.

  En este modelo se hace uso de operaciones de la teoría de conjuntos, para el manejo de las tablas (unión, intersección, etcétera), así como operaciones propias del modelo de datos relacional (selección, proyección, etcétera).

- Modelo de datos orientado a objetos

  En este tipo de modelo se encapsulan datos en un objeto, el cual a su vez interactúa con el resto del sistema a través de mensajes, los datos son guardados en variables y los mensajes son implementados por métodos que contiene el objeto.

En la actualidad el modelo más usado (aunque no es el más reciente) es el modelo relacional

I.I.3.- BASES DE DATOS RELACIONALES

Las bases de datos tienen que cumplir con ciertas características para poderse considerar relacionales; cuando se crean, se establecen mecanismos (reglas) para garantizar su buen funcionamiento, la integridad es respetar las reglas establecidas por el DLL (Lenguaje de definición de datos), y si ésta es cumplida en su totalidad, entonces la base de datos es consistente. "*La redundancia de datos se refiere, a la existencia de información repetida o duplicada innecesaria para la base de datos*"[1].

Las reglas de integridad se dividen en:

- ❖ Integridad de campo: Con esta regla se establece que cada campo que contenga la base de datos debe de tener valores atómicos, esto quiere decir que la información contenida en un campo sea indivisible.

- ❖ Integridad de entidad: Todas las entidades o tablas deben se tener un nombre único, así como cada uno de sus registros debe tener un identificador único.

[1] Notas diplomado de Administracion de Bases de datos, DGSCA UNAM.

- Integridad de clave: Los identificadores de registros deben de ser no nulos, no se deben cambiar nunca y no se deben duplicar.

- Integridad referencial: Si existe una relación entre 2 tablas A y B, el dato que relaciona a B debe existir en A. Cuando se quiere eliminar un registro en A, se puede indicar que se borren todos los registros en B que se relacionan con el elemento borrado en A (on delete cascade u on update cascade), o que se deje vacio el campo (on delete restict u on update restict).

- Integridad de usuario: Define qué usuario tiene permiso para manipular el RDBMS, quien entra a qué base datos y qué usuario tiene permiso de ver qué objetos.

Una base de datos debe tener independencia lógica de datos; es decir, que cualquier manejador pueda interpretarla independientemente de la forma en cómo se creó. Y también independencia física de datos, que significa que no importa si se cambia de lugar, aun así se debe poder acceder a sus datos.

Los identificadores de renglón son conocidos como llaves primarias, para poder identificarlas se pasa por todo un proceso de análisis. De todos los campos de una tabla se toman todos los que cumplen con ser no nulas, no se vayan a cambiar y no estén duplicadas y a esos campos se les conoce como llaves candidatas.

Una vez que se tienen las llaves candidatas se analizan cuál de ellas cumpliría mejor con el objetivo de ser llave primaria, realmente al ser candidatas todas podrían serlo, pero se recomienda que sea numérica y que sea lo más corta posible, las que no fueron elegidas como llave primaria se les denomina llaves secundarias, y finalmente las llaves foráneas son las que van a servir para relacionar tablas.

Un campo va a contener diferentes valores entre más crezca la base de datos; a ese catálogo de datos que contiene se le conoce como dominio.

I.I.4.- REGLAS DE CODD

Cuando nació el modelo relacional, Edgar Frank Codd definió 12 reglas a seguir para la creación de bases de datos relacionales:

1.- Regla de la información: Todo lo que se encuentre en una base de datos que se diga relacional debe tener forma de relación.

2.- Regla de acceso garantizado: Todo dato o valor en una base de datos relacional debe poder ser accedido mediante el nombre de su relación, el nombre de su atributo y el identificador de la tupla.

3.- Regla del tratamiento sistemático del valor nulo: Define la forma en que es tratado un dato nulo en una operación; cabe mencionar que independientemente de que operación se hable, si existe un operando nulo, el resultado también lo será.

4.- Regla del catálogo dinámico basado en línea: Se debe definir un catálogo que contenga la estructura y características de la base de datos.

5.- Regla del sublenguaje de datos completo: Todo manejador de bases de datos relacional debe de utilizar, para el manejo a datos, un lenguaje estándar; se le llama sublenguaje porque no hace uso de memoria física.

6.- Regla de vistas teóricamente actualizables: Cuando se hace constantemente una consulta que implica un join muy complicado y se quiere facilitar el acceso, o cuando se requiere que algunos usuarios no puedan ver todas las columnas de una tabla, se crea una vista; la cual, directamente no puede actualizarse ya que sólo es una ventana de acceso a datos contenidos en otro lugar; sin embargo, si la tabla origen de la vista es actualizada, el cambio se verá reflejado en la misma.

7.- Regla de inserción, actualización y eliminación de alto nivel: Se refiere a que con una sola instrucción se puedan afectar muchos registros.

8.- Regla de la independencia física de datos: No importa el medio físico donde se encuentre la información, se deben poder obtener sus datos.

9.- Regla de la independencia lógica de datos: No importa con que manejador fue creada una base de datos, cualquiera puede interpretarla.

10.- Regla de independencia de integridad: Las reglas de integridad pueden cambiar, no importando qué datos estén en las tablas.

11.- Regla de independencia de distribución: No importa cuántas veces esta fragmentada la información, tampoco si esas partes estén sobre dispositivos diferentes, se debe ser capaz de ver la información como si estuviera junta.

12.- Regla de no subversión: Estas reglas deben ser acatadas por todo lenguaje de programación.[1]

Una vez que se tiene clara la idea de las bases de datos relacionales y se conocen las reglas a seguir para que se considere relacional, se puede empezar a pensar en el diseño de una base de datos.

El diseño se hace en varias etapas, las cuales se representan por medio de los siguientes modelos:

- Modelo conceptual: El primer paso para diseñar una base de datos relacional es representar el problema, se identifican las entidades y resaltan las propiedades y atributos más básicos.

[1] Notas diplomado de Administracion de Bases de datos, DGSCA UNAM.

- Modelo lógico: Una vez que se sabe qué entidades se necesitan y algunas de sus características, se modela cómo esas entidades se relacionan entre sí.

- Modelo físico: Finalmente se definen la información detallada de cada una de las propiedades de los atributos.

Cada uno de los tres modelos es importante, ya que si no se identifican bien las entidades y sus atributos se pueden estar perdiendo datos importantes a guardar.

De los conceptos más importantes que se definen al hacer el modelo lógico, está la cardinalidad, donde se define como va a ser la relación entre dos entidades, hay tres opciones: uno a uno, uno a muchos y muchos a muchos. Esta información es clave, ya que ayudara en un futuro a definir las llaves foráneas.

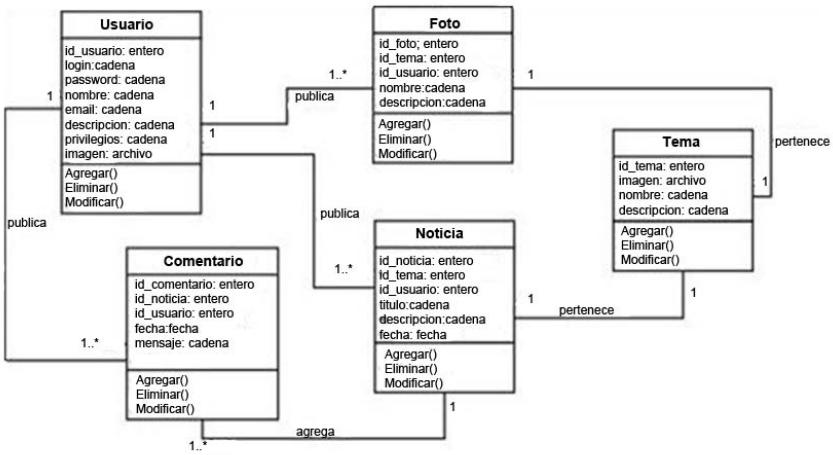

Figura I.1: Modelo lógico

En la figura I.1 se puede ver la relación uno a uno entre Foto y Tema, ya que una foto pertenece a un tema y viceversa, entre Usuario y Comentario existe una relación de uno a muchos, ya que un usuario puede publicar uno o muchos comentarios, mientras que un comentario solo puede ser publicado por un usuario.

Para el modelo físico, ya que se tienen bien identificadas todas las partes que compondrán la base de datos, se definen las características como: el tipo de dato a guardar en los atributos y que tamaño tendrán; si aceptará que el dato quede vacio o se pondrá un valor default; si será llave primaria o foránea o ambas.

Una vez que se diseñó la base de datos aún queda un último paso que debe hacerse para asegurar su óptimo y buen funcionamiento; éste proceso se conoce como normalización.

"El principio de normalización indica que las tablas de las bases de datos eliminaran las incoherencias y redundancias, y minimizaran la ineficacia"[1].

Básicamente se tienen tres reglas conocidas como formas normales:

- Primer forma normal: Todos los atributos deben guardar valores que sean atómicos; esto quiere decir, que no se puedan dividir.

- Segunda forma normal: Indica que se debe estar en primera forma normal y además que cada atributo no clave, dependa del atributo clave de forma irreducible.

- Tercera forma normal: Ningún atributo debe depender de otro.

I.II.- SISTEMAS MANEJADORES DE BASES DE DATOS RELACIONALES (RDBMS)

"*Un RDBMS (Relational Database Managment System) o sistema administrador de bases de datos relacional, es el conjunto de programas que permiten la definición, manipulación y control de acceso para una o varias bases de datos*"[2].

El RDBMS provee de varios beneficios, entre los que destacarían: un acceso fácil, seguridad de acceso a los datos y la mayoría de ellos contemplan varias reglas de integridad que generan consistencia en las bases de datos.

Existen muchos RDBMS, entre los que se encuentran: MS_SQL SERVER, ORACLE, SYBASE, MYSQL y POSTGRESQL entre otros.

La mayoría de los RDBMS manejan el log de transacciones, que no es más que un archivo que guarda las instrucciones que se van indicando; esto, con el objetivo de tener un histórico de instrucciones y se pueda volver a ellas en cualquier instante. Uno de los principales usos que se le da es el de monitorear lo que se ha hecho.

Los RDBMS tratan de seguir un estándar para el manejo y acceso de los datos, conocido como ANSI.

[1] Bases de datos relacionales, Fray León Osorio Rivera, Instituto Tecnológico Metropolitano.
[2] Notas diplomado de Administracion de Bases de datos, DGSCA UNAM.

I.II.1.- FUNCIONES DE UN RDBMS

- Facilitar la integridad
- Proveer seguridad.
- Dar acceso de los datos.
- Gestionar que los datos se almacenen con mínima redundancia.

I.II.2.- CARACTERÍSTICAS DE ALGUNOS RDBMS

- *MSSQL SERVER*

  Este RDBMS es comercial, por lo que se debe pagar una cierta cantidad por hacer uso de él, está desarrollado por Microsoft (lo que lo limita, ya que sólo se puede implementar sobre tecnología Windows) y permite la conexión vía JDBC y ODBC.

- *SYBASE*

  Este manejador está en dos versiones, tanto comercial como libre, la primera tiene la ventaja de que se cuenta con un soporte, así como la adquisición de manuales técnicos del producto y manuales de uso, además de tener conexiones ilimitadas para los usuarios, provee de soporte a los administradores del manejador. La versión libre está limitada por el número de usuarios que pueden conectarse, en ambas versiones permite conectividad ODBC y JDBC; internamente la información se divide por páginas de 2, 4, 8 y 16 kilobytes, permite el bloqueo a varios niveles (tabla, columna y páginas de datos), y a diferencia de MSSQL SERVER, éste es multiplataforma.

- *ORACLE*

  Es multiplataforma, tiene su propio lenguaje de programación llamado PL/SQL, tiene protección de datos extremadamente fuerte y entre sus mejores características está que permite redefinición de la estructura de una tabla en línea, sin pérdida de datos.

- *INFORMIX y DB2*

  Este manejador es el más seguro del mercado, además de que soporta bases de datos enormes (se habla de más de 4 terabytes), permite replicación de bases de datos y conexiones paralelas al mismo objeto; Por estas características es que es el más utilizado por los bancos.

- *POSTGRESQL*

  Este manejador es de distribución libre, lo que implica que se puede copiar en las máquinas que sean necesarias sin problema alguno, es muy veloz y fácil de utilizar.

- *MYSQL*

  Al igual que PostgreSql es libre, maneja 2 motores (innoDB y MyIsam), la diferencia es que innoDB es utilizado para cuando las bases de datos van a ser altamente transaccionales, y las MyIsam es para cuando no están en constante crecimiento. Existe una gran comunidad en la Web que utiliza este manejador de bases de datos, por lo que es fácil resolver problemas que se presenten ya que la gran mayoría en algún momento ya se resolvieron y los mismos usuarios se encargan de compartir sus experiencias.

RDBMS	MYSQL	POSTGRESQL	SQL Server	ORACLE
Tipo de licencia	GNU GPL(General Public Licence)	BSD	Datacenter, standard, enterprise, workgroup, web y developer	Enterprise Edition, Standard E, SEOne
Aplicaciones gráficas de administración	Mysql Administrator, mysql query, mysql migrator.	PgAdmin, PgExplorer, PgAdmin 3	Service Manage, Enterprise Manager y Query Analyzer	Oracle Raptor 6, SQL Developer
Aplicación de consola	mysql	psql	isql	sql*plus
Arquitectura	Intel x86, IA32, IA64, AMD Optaron, IBM PowerPC, Sun SPARC	AMD Opteron, AMD Athlon 64, Intel Xeon compatible con Intel EM64T, Intel Pentium IV, IBM PowerPC, Sun SPARC	AMD Opteron, AMD Athlon 64, Intel Xeon compatible con Intel EM64T, Intel Pentium IV compatible con EM64T	Intel x86, IA32, IA64, AMD Optaron, IBM PowerPC, Sun SPARC
Sistemas operativos en los que corre.	Linux: RedHat, SuSE, Fedora, Debian. Unix: Solaris, HP-UX, AIX, SCO. Windows: 2000, 2003, XP. Novell: NetWare.	Disponible para cualquier versión de UNIX y Windows 2000, xp y vista.	Windows Server, Windows NT, XP y vista.	Linux, Unix, Windows: 2000, 2003, XP. Novell: NetWare. Apple: Mac OS.

	Apple: Mac OS.			
Lenguaje procedural		PL/pgSQL	Transact-SQL o T-SQL	PL/SQL
Velocidad	Posee una gran velocidad sobre todo en bases de datos pequeñas.	Es 3 o 4 veces más lento que MySQL.	Es lento en comparación con MySQL.	Es lento con respecto a los demás.
Conectividad	JDBC, ODBC y sockets TCP/IP	JDBC, ODBC y TCP/IP	JDBC, ODBC, TCP/IP.	JDBC y ODBC.
Puerto por default	3306	5432	1433	1521
Versión actual	MySQL 6.2	Postgres 8.3	SQL Server 2008	Oracle Database 11g
Version ANSI SQL	SQL-92	SQL-92	SQL-92	SQL-92
Costo aproximado	Sin costo	Sin costo	Datacenter por procesador $600,000, enterprice por procesador $300,00, estándar por procesador $80,000, workgroup por procesador $40,000, web por procesador $40,000 o $2000 por mes, developer por usuario $500, express gratuito.	Esta entre $80,000 y $700,000 dependiendo del tipo de licencia.

Tabla I.1: Tabla comparativa entre algunos manejadores.

I.III. -SQL (Structured Query Language)

El SQL es un lenguaje de consulta, que es utilizado para manipular la información contenida en una base de datos, es considerado un lenguaje de alto nivel, ya que utiliza palabras comunes que describen lo que se está haciendo, por lo que es muy fácil de aprender.

SQL está compuesto de 3 sublenguajes:

- *DDL:* Es el que se encarga de la definición o creación de los objetos.
- *DML:* Permite la manipulación de datos.
- *DCL:* Es aquel que permite controlar los accesos a los diferentes objetos definidos.

"*El estándar SQL es adoptado como tal en el año de 1986, aunque el que más se utiliza es la versión 92*"[1].

- *ANSI 89*: Éste estándar, implementa las reglas de Codd, los elementos de las bases de datos como son los DDL, DML y DCL, y las funciones de agregado.
- *ANSI 92*: En esta versión del estándar se le añaden los esquemas, el manejo de las vistas, y admite el uso de la palabra **distinct** en las consultas (la cual indica que se extraigan sólo registros no repetidos).
- *ANSI 99*: Además de implementar lo de los estándares anteriores, agrega nuevos tipos de datos como el clob y el blob que permiten guardar información en forma de objetos.

El objetivo de los estándares ANSI es que los RDBMS lo implementen, para que de esta forma el usuario final no tenga ningún problema en pasar de uno a otro; sin embargo, no todos lo hacen, ya que la mayoría en su afán de ganar terreno en el mercado, implementa funcionalidades extra; lo que significa que cuando se quiera migrar de un manejador a otro, haya problemas de compatibilidad.

I.III.1.- DDL O LENGUAJE DE DEFINICIÓN DE DATOS

- ***CREATE***

Bases de datos

Todo lo que se encuentra dentro de una base de datos es un objeto incluyendo a la propia base de datos, y como tal, se debe crear para después agregarle otros objetos.

[1] Notas diplomado de Administracion de Bases de datos, DGSCA UNAM.

La sintaxis básica para la creación de una base es:

CREATE DATABASE nombre;

Tablas

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

CREATE TABLE nombre_tabla (nombre_columna tipo_dato NULL | NOT NULL DEFAULT 'valor_default' constraint, … n)

Es recomendable que una tabla tenga un nombre descriptivo a lo que va a almacenar. Por ejemplo, si va a almacenar la información de los empleados de una empresa, un nombre adecuado sería "*empleado*", además, debe cumplir con otras reglas, como que no inicie con caracteres diferentes al alfabeto y que no sea demasiado largo, en este sentido depende del manejador, algunos aceptan nombres largos y otros no, pero se debe seguir un estándar y no irse al límite para no tener problemas después.

AnsiSQL92
Tinyint
Int, integer
Smallint
Numeric(p,d)
Float
Real
Doublé
Carácter varying(n)
Boolean
Date
Time
Timestamp
Char, carácter(n)
Blob
Clob

Tabla I.2: Tipos de datos definidos por el estándar ANSI SQL92

Como se ve en la sintaxis, cada tabla existente en la base de datos contendrá a su vez campos, los cuales tendrán que ser definidos con el tipo de dato que van a almacenar, esto es importante ya que cada campo tiene que coincidir con la información que en un futuro va a contener. Si se necesita almacenar un texto, se tiene que buscar qué tipo de dato es el adecuado; si se define un tipo de dato diferente, el dato no podrá ser almacenado.

Existe una cantidad extensa en cuanto a tipos de datos; sobre todo, porque algunos manejadores definen de forma diferente, algunos les varían el nombre como es el caso del tipo de dato numeric, el cual manejadores como el SQL

server y Sybase lo manejan como decimal. Otro de los casos son los tipo real y float que manejadores como MySql y PostgreSql no los implementan, en la tabla I,2 se pueden ver algunos tipos de datos que son parte del estándar ANSI SQL 92.

Algo a tomar en consideración al momento de definir tipos de datos, es que hay ocasiones donde la información esperada no es proporcionada; en tales casos es necesario saber que los valores nulos, el cero y una cadena vacía son diferentes, y que si el tipo de dato es numeric o int es más conveniente colocar un cero que un null ya que si se quisieran realizar operaciones se tendrían serios problemas , un valor nulo es un vacio, sin dato; TODA OPERACIÓN DONDE INTERVENGA UN VALOR NULO DA COMO RESULTADO NULO, es importante manejar un default 0 para los campos de tipo numérico.

Reglas

Además de definir tablas, se pueden crear reglas, las cuales son restricciones que se ponen a ciertos campos. Si se quisiera que en un campo se almacene un dato numérico, pero que este no sobrepase el número 100, se crearía la siguiente regla:

CREATE RULE nombre AS campo1<100

No es muy común la definición de reglas en una base de datos, dado que disminuirían mucho su rendimiento porque no serían flexibles. Además de que no todos los manejadores las implementan.

Existen más objetos que pueden ser creados en una base de datos, todos utilizan la misma sintaxis para su creación, entre ellos están los mencionados a continuación:

- Triggers: "*Es un procedimiento almacenado que es invocado cuando un evento en particular ocurre. Por lo que se puede ejecutar cada vez que se borre, actualice o inserte un registro*"[1].

- Cursores: "*Es un área de memoria de trabajo en la que se almacena la información de salida de una sentencia SQL y que es necesario declarar para posteriormente acceder a las tuplas extraídas en la consulta*".

- Vistas: "*Es una tabla lógica que permite acceder a la información de otras tablas o vistas*".

- Procedimientos almacenados:" *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*".[2]

[1] Notas diplomado de Administracion de Bases de datos, Modulo 2 'SQL', DGSCA UNAM.

[2] Irene Luque Ruiz, Miguel Ángel Gómez-Nieto, Bases de Datos desde Chen hasta Codd, Alfaomega

- ***DROP***

Cuando un objeto ya no es necesario en el esquema debe ser eliminado, el proceso que se lleva a cabo para eliminarlos es muy simple y se lleva a cabo con la siguiente instrucción:

DROP objeto nombre

- ***ALTER***

Una vez definido un objeto, este puede necesitar ser modificado por alguna razón, ya sea que se definió mal su nombre o alguno de sus campos faltó o esta de más o algún tipo de dato no es el correcto, etcétera. En el caso de una tabla si aun no almacena nada sería mejor borrarla y crear una nueva. Si ya se cuenta con información es indispensable hacer un respaldo de la misma antes de intentar cualquier cambio, ya que no siempre responde de la manera en que se desea y podría ser muy desagradable la experiencia en el caso de pérdida de información.

No todos los manejadores permiten la alteración de una tabla, por ejemplo PostgreSql y Sybase no permiten eliminar campos de una tabla y tampoco renombrarlos, sin embargo Oracle, MySql y SQL Server sí.

Para modificar la estructura se utiliza la sentencia:

ALTER objeto nombre.

Llaves

Las llaves también se definen mediante SQL, ya sea al momento de la creación de la tabla o modificándola cuando ya está creada.

I.III.2.- DML O LENGUAJE DE MANIPULACIÓN DE DATOS

Una vez que se definió la estructura de la base de datos, se puede ingresar, manipular y sobre todo extraer información almacenada. Para ello es que se definió el DML, que son el conjunto de instrucciones para realizar estas tareas.

Para insertar registros en una tabla se utiliza el comando INSERT, no es necesario introducir todos los campos definidos, pero si los definidos como NOT NULL, ya que como se está indicando no es posible introducir valores nulos en ese campo y se debe definir un valor, a menos que tenga un valor default.

La sintaxis para insertar registros es:

INSERT INTO nombre_tabla (nombre_campo, … n) VALUES (‘valor_campo’… n)

Una nota importante, es que cuando se inserta información los tipos de datos diferentes al numérico deben ir entrecomillados.

Para actualizar información, en caso de ser necesario, se utiliza la sentencia UPDATE. No es necesario alterar todos los campos de un registro, se puede hacer campo por campo o varios a la vez. Se debe tener demasiado cuidado en este caso, ya que si no se especifica una condición, la modificación afectará a todos los registros existentes, éste es uno de los errores comunes.

La sintaxis para actualizar registros es:

UPDATE nombre_tabla SET nombre_campo='valor_campo', … n CONDICIÓN

Para eliminar registros se utiliza DELETE que, al igual que UPDATE si no se da una condición, borra todos los registros.

La sintaxis para eliminar registros es:

DELETE FROM nombre_tabla CONDICIÓN

Finalmente, existe SELECT, que es la instrucción que se utiliza para extraer información de la base de datos, es la más utilizada por administradores y programadores, ya que constantemente se necesita hacer uso de la información para la toma de decisiones.

La sintaxis para extraer registros es:

SELECT *|nombre_columnas FROM nombre_tabla

Join

Cuando se diseñan bases de datos, lo común, y hasta cierto punto efectivo, es que la información no se encuentre toda en una tabla, pero… ¿Qué pasa si se quiere obtener información que se encuentra en 2 o más tablas? No es necesario hacer 2 o más consultas, existe un concepto llamado join que, como su nombre lo indica sirve para unir y extraer información de varias tablas, éstas tienen que tener una columna en común, que servirá para hacer la condición para la extracción; de no poner esta condición, se estará haciendo un producto cartesiano, lo que significaría, cada uno de los renglones de la tabla1 combinado con cada uno de los renglones de la tabla2. Por ejemplo, si la tabla 1 contiene 5 renglones y la tabla 2 contiene 6 y se hace un join de ambas tablas sin condición, se obtendrían 30 renglones que seguramente no será el resultado esperado, algo a tomar en cuenta es que si la tabla 1 tuviera 2000 registros y tabla 2 150000, un join sin condición correría el riesgo de parar la base de datos.

La sintaxis de un join es:

SELECT *|nombre_columnas FROM nombre_tabla1, nombre_tabla2,… nombre_tablaN CONDICIÓN

I.III.3.- DCL O LENGUAJE DE CONTROL DE DATOS

El DCL es el encargado de la seguridad de la base de datos, es un conjunto de sentencias que permiten dar o quitar permisos sobre los objetos existentes. Para otorgar permisos se utiliza la palabra reservada GRANT, y la sintaxis es:

GRANT ALL ON baseDeDatos.objeto TO usuario IDENTIFIED BY ‘contraseña’;

Se puede dar permiso de acceder a una base de datos completa o a ciertas tablas de la misma, también se pueden dar permisos por usuario o a un grupo de usuarios.

Para quitar los permisos se utiliza la instrucción REVOQUE, la sintaxis es similar a la mencionada para GRANT.

I.III.4.- OBJETOS Y FUNCIONES EN UNA BASE DE DATOS

Los objetos que se pueden encontrar en una base de datos son bastantes, uno de ellos y de los más utilizados son las vistas, las cuales son como un acceso directo a toda o parte de una tabla o vista; por ejemplo, si se tiene la tabla empleados la cual contiene todos los datos del empleado incluyendo su sueldo y se necesita otorgarle permisos sobre la información contenida ahí a algunos usuarios, pero ocultándoles el sueldo del empleado, se crea una vista hacia todos los datos que se desea que vean los usuarios y se dan permisos sobre la vista y no sobre la tabla; entonces, se puede decir que la vista es una ventana de acceso a información real que no puede ser actualizada desde la misma, pero que si se actualiza la tabla de donde provienen los datos, se verá afectada.

También existen las funciones, las cuales sirven para ejecutar procesos de manera más ágil, si bien están definidas en el estándar sql92 un grupo de funciones que todo RDBMS debe tener, no todos lo siguen y algunos tienen funciones implementadas por ellos mismos.

Entre las funciones más importantes que existen, se encuentran las siguientes:

substr	Extrae parte de una cadena.
upper	Devuelve una cadena en mayúsculas.
lower	Devuelve una cadena en minúsculas.
trim	Quita los espacios en blanco en los extremos de una cadena.
case when	Evalúa una expresión.

Tabla I.3: Funciones de cadena.

abs	Sirve para obtener el valor absoluto de un número.
ceiling	Obtiene el entero más próximo hacia arriba de un valor con decimal.
cos	Da como resultado el coseno de un número dado.
floor	Obtiene el entero más próximo hacia abajo de un valor con decimal.
Log	Saca el logaritmo de un número.
Log10	El logaritmo base 10 de un número.
pi	Da el valor de pi.

Tabla I.4: Funciones matemáticas.

Current_date / Current_date()	Regresa la fecha actual en formato 'YYYY-MM-DD' o YYYYMMDD
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 'YYYYMM-DD HH:MM:SS' o YYYYMMDDHHMMSS

Tabla I.4: Funciones de fecha.

I.III.5.- COMMIT Y ROLLBACK

"*Una transacción es una unidad de trabajo lógica que comprende por lo regular varias operaciones de base de datos*"[1].

Cuando se realizan operaciones de inserción, actualización y borrado sobre los datos es posible deshacer los cambios si se inició una transacción, la forma de iniciarla es **begin** o **begin transaction**(dependiendo del manejador); una vez que se ha efectuado el inicio de la transacción, las instrucciones son llevadas a cabo; sin embargo, en memoria se van guardando éstas, de tal forma, que si no se finaliza con éxito, se pueda volver al estado inicial. Cuando se desea que los cambios sean permanentes se da la instrucción **commit**, y cuando se desea que los cambios no se efectúen, **rollback**.

[1] Date, C. J. *Introducción a los sistemas de bases de datos.* Prentice Hall

I.IV.- ACCESO A DATOS A TRÁVES DE LA PROGRAMACIÓN DE CLIENTES

La información contenida en una base de datos es inservible si no es utilizada, uno de los grandes objetivos del almacenamiento de información es extraerla para tomar decisiones en base a su contenido.

El acceso a los datos no siempre es desde el cliente del propio RDBMS; la mayoría de las ocasiones, el acceso se da desde aplicaciones que necesitan de esa información para su funcionamiento.

La mayoría de los lenguajes de programación permiten la conexión a las bases de datos, el uso de uno u otro depende del tipo de aplicación que se necesite crear; por ejemplo, si el acceso se quiere de forma local en el equipo, se podría hacer uso del lenguaje c o lenguaje java para crear las aplicaciones; si se desea crear una aplicación con la cual se pueda acceder a los datos desde otro punto en la red, lo más conveniente sería utilizar lenguajes de script del lado del servidor como php, asp y jsp.

I.IV.1.- PHP

El uso de PHP para el desarrollo de aplicaciones web que interactúan con bases de datos es enorme, ya que éste es un lenguaje muy simple, además de que es muy parecido al lenguaje C, otra de sus características es que es un lenguaje multiplataforma además de que es interpretado, por lo que no necesita de una compilación previa.

PHP permite la conexión a múltiples manejadores de bases de datos como PostgreSql, MySql, Oracle, SQL Server, Sybase, etcétera. Esta conexión se puede hacer por ODBC en tecnologías de Microsoft o por medio de algún conector creado por la compañía del propio manejador.

Las características del lenguaje PHP son similares a las de otros lenguajes, tiene estructuras de control (if, for, while, switch), declaración de variables (aunque no es necesaria), incrementos y decrementos, redirección, lectura y escritura de archivos, manejo sesiones y cookies.

Una página web, hasta hace unos años era totalmente estática; una máquina cliente hacia una petición al servidor, el cual localizaba una página web HTML ya creada, y la enviaba como respuesta al cliente.

Actualmente y gracias a lenguajes como PHP ya no es así, la máquina cliente hace la petición al servidor, éste busca la información, crea dinámicamente en el servidor la página en base a lo que se encuentra en una base de datos y lo solicitado y responde enviando la página generada.

El código HTML del que inicialmente se hacia uso para crear páginas estáticas se sigue utilizando y el código PHP se integra dentro de éste, las marcas de

código PHP son <?php, <?, <?=, ?>, así que todo lo que se encuentra dentro de éstas es código PHP.

De los elementos del lenguaje HTML se sigue utilizando todo, párrafos, títulos, tablas, frames y formularios, estos últimos muy importantes en el desarrollo de sistemas de acceso a datos.

El servidor más utilizado para este tipo de sistemas es Apache, al cual sólo se le activa un modulo para que interprete PHP.

Para acceder desde el script a una base de datos en el servidor se necesita de un host, un puerto, nombre de la base de datos, nombre de usuario y contraseña.

La configuración de php se hace en un archivo llamado php.ini. Ahí se definen algunas características como si se admite el uso de sesiones, el uso de cookies, si se permite el uso de variables globales, etcétera.

I.IV.2.- JSP

JSP al igual que PHP es un lenguaje de script, es tecnología desarrollada por Sun y muy similar a java, de hecho se conoce como Java para Web. Al ser Java posee todas sus características como: portabilidad, simpleza, seguridad, y a diferencia de PHP requiere una declaración forzosa de una variable, por lo que es conocido como un lenguaje fuertemente tipado.

También se integra en las páginas web dentro de las líneas HTML entre los signos <% %>, además de poder realizar la conexión por ODBC a la base de datos, lo puede hacer por medio de JDBC que es un conector desarrollado por la empresa propietaria del manejador para conectar con la tecnología Java.

JSP también es interpretado como PHP, aunque los archivos jsp la primera vez que se corren son compilados y convertidos a un servlet, lo que se traduce en tiempo; una vez que se ejecutó por primera vez, las siguientes serán interpretadas. JSP, al ser descendiente de java, es orientado a objetos por naturaleza.

Java cuenta con toda una plataforma de desarrollo, y ha evolucionado la creación de sus sistemas de sólo JSP y servlets a frameworks que integran éstas y más tecnologías para el buen desarrollo de un sistema, uno de ellos es struts que implementa el patrón de diseño MVC (Modelo Vista Controlador) para la creación de sistemas, lo que este framework implementa es una separación de tareas; utiliza los servlets para la lógica de negocio, los JavaBeans para el acceso a datos y los JSP para la presentación de los mismos. No sólo es struts, también existe spring, java server faces e hibérnate.

I.IV.3.- ASP

Este lenguaje de script, a diferencia de los otros, es tecnología pura de Microsoft por lo que los sistemas desarrollados con ASP son ejecutados en un servidor IIS (Internet Information Server), es posible correr estos scripts sobre Unix pero se necesita de un programa que interprete los datos.

I.V.- FUNDAMENTOS DE SISTEMAS OPERATIVOS

Éste modulo del diplomado está orientado a explicar de manera general como se administra un sistema operativo, en aras de que el administrador de bases de datos conozca las tareas básicas del sistema operativo en el que va a instalar su manejador de base de datos, así como métodos que le permitan mantener un nivel de seguridad alto desde el sistema operativo (configuración y manejo de firewall, monitoreo de logs, etcétera).

Un sistema operativo es un programa que se encarga de administrar los recursos con lo que cuenta la computadora; al haber muchos procesos haciendo peticiones constantemente, él es el encargado de repartir los recursos como los tiempos de memoria para cada proceso.

Los sistemas operativos, al igual que las computadoras, han evolucionado; antes era una pantalla negra donde únicamente se escribían instrucciones que sólo algunas personas eran capaces de interpretar, fue a partir del Windows 3.1 que se dio la revolución del entorno gráfico, aunque no necesariamente fue el primer sistema operativo con éste, ya que se habla de que la empresa Apple había desarrollado antes un sistema operativo gráfico.

En la actualidad, los sistemas operativos existentes son muchísimos, por un lado Windows y sus diferentes versiones desde la 3.11 hasta Windows Seven, Linux y sus diferentes presentaciones (Mandrake, Red Hat, CentOs, Fedora, Suse) y UNIX (Solaris, aix, iris).

Uno de los conceptos más importantes dentro de cualquier sistema operativo son los procesos, que es un programa en ejecución que puede pasar por varios estados: en ejecución, listo, bloqueado. Cuando está en ejecución, el proceso está corriendo y realizando las tareas para lo que fue lanzado; cuando está en estado listo, el proceso está en estado óptimo para ser ejecutado pero no hay CPU disponible para él; y bloqueado, es cuando está parado por haber terminado su ciclo o porque está en espera de algún evento para lanzarse.

El sistema operativo cuenta con 4 partes capaces de desarrollar las tareas de mando, que son las siguientes:

- *Administrador de procesos:* Su función es la de administrar los tiempos del procesador, qué proceso se lanza primero y por cuánto tiempo.

- *Administrador de E/S:* Administra los dispositivos de entrada y salida (mouse, escáner, monitor, teclado, lápiz óptico, etc.)
- *Administrador de memoria:* Cuanta memoria es designada para cada tarea.
- *Manejo de sistema de archivos:* Cómo se estructuran los archivos dentro del árbol de directorios.

I.V.1.- LINUX

Uno de los sistemas operativos más utilizados como servidores de aplicaciones web es Linux, y una de las características principales para el logro de esto es la seguridad.

Sus características principales son:

- Es multitarea, puede ejecutar muchas tareas al mismo tiempo.
- Es multiusuario, pueden ocuparlo muchos usuarios a la vez sin afectar el rendimiento.

La instalación de los Linux en la actualidad es muy simple, casi en todas la versiones es totalmente en modo gráfico, en lo que si se debe tener mucho cuidado es al momento de particionar el disco duro, ya que se deben dejar particiones específicas para un uso posterior, también se debe apartar una partición específica para utilizarla como memoria virtual, llamada swap.

Una vez instalado el sistema operativo, si se desea utilizar como servidor web, se necesitará instalar un programa que haga esa función, uno de los más conocidos es Apache, la instalación de este tipo de programas es recomendable hacerla vía comandos, ya que de otra forma la configuración es menos controlada, aunque para ello hay que conocer los comandos correspondientes.

En las versiones que son variantes de Red hat existe el comando **rpm,** el cual ayuda a instalar y desinstalar paquetes; uno de los problemas es que si el paquete a instalar tiene alguna dependencia con otro no se instalara, y para ello se debe instalar con el comando **yum** que instala el paquete y sus librerías necesarias para funcionar.

Los comandos similares para sistemas Debian son **dpkg** y **apt-get**.

Una vez instalado el servidor, se debe configurar; para ello cuenta con un archivo de configuración, el cual se edita usando alguno de los editores de texto disponibles en Linux, el más utilizado es el **vi**. También deberá ser protegido dando permisos de lectura, escritura y ejecución a las personas indicadas, eso se hace con el comando **chmod**.

Cuando se inicia el servidor, éste no es más que un proceso que se está ejecutando, así que se podría necesitar monitorearlo de vez en cuando, eso se hace con el comando **ps**.

I.V.2.- WINDOWS SERVER

Microsoft por su parte cuenta con el sistema operativo Windows Server, el cual viene con muchas utilidades para administración. Este servidor es comercial, por lo que se debe pagar una licencia por su uso, la licencia puede ser adquirida por servidor o por usuario, según convenga.

La instalación de un sistema operativo Windows es muy simple, ya que todo es gráfico. Una vez instalado se procede a administrar los usuarios, para ello cuenta con la herramienta MMC (Management Microsoft Control), él cual es un entorno para la administración de usuarios y grupos. Los grupos son el mismo concepto que los roles en Linux.

Para monitorear los procesos se utiliza el programa **taskmgr**, e igual de manera gráfica se podrán ver los procesos y suspenderlos o matarlos, si se cuenta con los permisos necesarios.

Existen muchísimas utilidades, pero de las más importantes esta: **backup,** que apoya en el uso y manejos de respaldos; así como **scheduled task,** que sirve para programar tareas.

I.V.3.- MÁQUINAS VIRTUALES

El hecho de tener uno u otro sistema operativo instalado no significa que ha de trabajarse únicamente con éste; existen las máquinas virtuales, que son aplicaciones que emulan todo un sistema en una porción del disco duro; se puede tener Windows instalado e internamente, y con la máquina virtual, tener un Linux instalado y ejecutándose al mismo tiempo.

I.VI.- HABILIDADES DIRECTIVAS PARA ADMINISTRADORES

Este modulo ésta orientado a formar una profesional con un perfil de líder, que conozca las diferentes formas de comunicarse con las personas, así como aplicarlas para un mejor desempeño del equipo de trabajo.

Para ser un administrador, primero se tiene que ser un buen líder, y para ser un buen líder es muy importante la comunicación con las personas. Existen 3 tipos de comunicación, la visual, la cenestésica y la auditiva.

La comunicación visual: Es aquella que lleva a cabo a través de imágenes, toda persona que utiliza este tipo de comunicación asocia todo por medio de efectos visuales y le es más fácil aprender si observa las cosas.

La comunicación Cenestésica: En este tipo de comunicación la persona utiliza su subconsciente para identificar sensaciones o sentimientos. Está asociado con el olfato y el tacto, una persona que le es más fácil asociar ideas por medio de contacto físico con objetos es Cenestésica.

La comunicación auditiva: Los sonidos son parte esencial para los individuos auditivos, normalmente les es más fácil aprender las cosas si las escuchan.

El buen comunicador debe saber utilizar los 3 tipos de comunicación cuando expone una idea, tratar de que las personas que lo escuchan lo entiendan independientemente de que tipo de comunicación utilicen.

Cuando se tiene personal a cargo, se debe aprender a comunicar lo que se necesita, liderear un equipo es complicado, sobretodo si no se tiene la experiencia necesaria, no es lo mismo ser jefe que ser un líder, existen diferencias muy marcadas entre ambos; por ejemplo, un jefe ordena, un líder pide; un jefe se concentra totalmente en las tareas, un líder le da el valor a las personas para que realicen bien sus tareas; un jefe da una conferencia tipo monólogo, un líder conversa con sus empleados y hace preguntas para escucharlos.

Un administrador además de ser buen comunicador y un líder debe de tener bien claros sus objetivos, ponérselos a corto plazo para así poder ver si está cumpliéndolos en los tiempos deseados, esos objetivos deben de ser alcanzables, en el proceso de crear los objetivos entra el proceso PCM, este consiste en determinar si el objetivo es posible, si se es capaz de alcanzarlo y si se merece en caso de lograrlo.

Cuando se está en una junta de negocio se querrán transmitir ideas, y normalmente eso se hace en base a presentaciones, para asegurar que éstas cumplan eficazmente con el objetivo de comunicar bien las ideas, deben de cubrir algunos requisitos como los siguientes:

- Una diapositiva sólo debe tener palabras clave para no saturarla de información que haga que las personas por leerla se distraigan y no pongan atención a lo que explica el expositor.
- Además debe tener colores claros, que no sean muy contrastantes.
- No saturarla de imágenes.
- Usar una letra legible y grande.
- Usar viñetas cuando sean necesarias para el mejor entendimiento por parte de la audiencia.

Pero no sólo la presentación de las diapositivas juega un papel importante, también se debe cuidar un aspecto tan importante como la voz del expositor, ya que un tono muy grave o muy agudo puede llegar a molestar tanto a las personas que escuchan, que preferirán irse.

El expositor debe cuidar detalles como: no estar tenso, tratar de ser seguro de sí mismo, mirar a las personas a los ojos, tener una buena postura, no caminar demasiado.

El lenguaje corporal es muy importante, las manos son muy expresivas. Cuando se hace una presentación se debe de tomar en cuenta el tiempo, no administrar bien los tiempos, llevaría a terminar antes de que concluya la sesión y no tener más de que hablar, por lo que se debe procurar terminar justo en el momento que finaliza la sesión.

I.VII.- ADMINISTRACIÓN DE BASE DE DATOS.

La administración de una base de datos es uno de los puntos medulares en la actualidad en el rubro de las tecnologías de la información.

Administrar una base de datos significa: crear usuarios, darle permisos a esos usuarios, crear respaldos. Pero no sólo está ligado a la seguridad de los datos, sino también a asegurar el correcto guardado de la información y que la base de datos este siempre al cien por ciento disponible para su uso eficaz, por lo que un administrador de base de datos también tiene que: estar al tanto de los dispositivos físicos en los que se almacenará, monitorear el sistema operativo para así detectar cualquier posible falla, y en caso de tener que hacer una migración total o parcial, encargarse de verificar que haya una elección correcta del sistema operativo y del hardware que ha de utilizarse.

I.VII.1.- ADMINISTRACIÓN DE SYBASE

Cuando se instala una base de datos Sybase en una máquina, se toma una porción del disco duro; ésta, a su vez, se divide en 3 dispositivos al menos.

- *El primer dispositivo (Master):* Va a contener las bases de datos útiles para el manejo de la información del sistema: **master**, la cual contiene las tablas del sistema; **model**, que es la que se toma como modelo para la creación de las demás; y **tempdb** que almacena datos temporales.
- *El segundo dispositivo (sysprocs):* Contendrá la base de datos **sybsystemprocs,** que almacenará a su vez todos los procedimientos almacenados que se cargan de inicio al instalar Sybase.
- *El tercer dispositivo (Dev-dat1):* Almacena las bases de datos de los usuarios, a menos de que se indique lo contrario en la configuración.

Opcionalmente, se crearán más dispositivos, es común que se desee crear el dispositivo **dev-log1** el cual se puede utilizar para separar los logs de las bases de datos; de lo contrario, el log estará en dev-dat1 y se corre el riesgo de que el espacio del dispositivo dev-dat1 se llene muy rápido.

Para crear los dispositivos es importante calcular previamente cuánto espacio es necesario a un tiempo estimado. Sybase cuenta con un procedimiento para hacer ese cálculo, invocado al ejecutar **sp_estspace**.

Sybase cuenta con otros procedimientos almacenados para realizar múltiples tareas; por ejemplo: listar los dispositivos existentes **sp_helpdevice**, mostrar una ayuda **sp_help**, etcétera.

Este manejador cuenta con 3 capas de seguridad:

- La capa uno es la de acceso y se encarga de que sólo acceda al manejador las personas permitidas.
- La segunda capa es la de base, sólo accede a una base de datos las personas autorizadas ya sea por el administrador o por el propietario de esa base de datos.
- La tercera capa es la de objeto, que verifica los permisos de los objetos existentes en la base de datos como tablas, vistas, procedimientos almacenados, etcétera.

Para definir esos permisos se necesitan crear roles, existen 3 roles principales y uno extra.

- Sa_role

  Este rol es el de administrador del sistema, el cual tiene permisos para todo dentro de la base de datos, es el superusuario.

- Sso_role

  Este es el rol de operador de seguridad del sistema, el cual permite crear logins y realizar auditorías, todo lo que tenga que ver con permisos de acceso es tarea del sso_role.

- Oper_role

  Este role está definido únicamente para tener permiso de ejecutar respaldos de la base de datos o de las tablas.

El rol extra es el DBO, el cual se asigna al usuario que crea una base de datos; ese usuario, al ser el propietario de esa base de datos, tiene permiso de hacer todo con ella, desde asignarle usuarios, crear tablas, etcétera.

También se pueden definir grupos de usuarios, los cuales ayudarán a dar permisos por grupo de una manera más rápida y controlada; por ejemplo: si se tienen 50 usuarios y 20 pertenecen a un grupo y a todos los que pertenecen a ese grupo se les quisiera dar acceso a la base de datos prueba, con una sola

instrucción se podría otorgar permiso a todo ese grupo, de no existir este concepto tendría que haber indicado 20 instrucciones, una por usuario.

Además, se puede configurar una base de datos por medio de un procedimiento almacenado que al ejecutarlo permite alterar algunas opciones de configuración; el procedimiento es **sp_dboption**. De las opciones a configurar están: el definir si la base de datos es de sólo lectura o no, si acepta nulos, el número de usuarios conectados al mismo tiempo, etcétera.

El comando **dbcc** permite hacer una desfragmentación de los datos, por lo que es una buena opción ejecutarlo para optimizar la base de datos.

I.VIII.- MEJORES PRÁCTICAS EN LA FUNCIÓN DE LA ADMINISTRACIÓN.

Como ya se mencionó existe un grupo de tareas que un administrador de la base de datos debe hacer; pero no se trata de sólo hacer por hacer, se debe tener un buen control de cómo se realizan éstas, y para ello existen algunas normas o estándares que van a ayudar a dar un buen seguimiento a estos procedimientos con la finalidad de asegurar su buen funcionamiento.

Pocas dependencias implementan estándares en sus procedimientos y es que suele pensarse que es muy costoso, tanto en tiempo como en dinero, y si lo es, ya que es posible que lo que se puede hacer en dos meses se haga en cinco, lo que implica perder tres meses de trabajo y el sueldo de los empleados en esos meses, pero lo que pocas dependencias consideran es que es más costoso aun no seguir esas normas y sobre todo no estar certificado, ya que si son objeto de una auditoria informática y no cumplen con los estándares permitidos la multa podría ser muy costosa, y sobre todo la desconfianza de los clientes que al ver que es una dependencia que no cumple con los estándares de calidad buscan una alternativa, que termina en muchos casos con la quiebra de esa empresa.

Actualmente existen dos procesos de buenas prácticas que se pueden implementar en una dependencia.

- *COBIT:* El cual está más orientado a la implementación de procedimientos para prevenir fallos futuros, los cuales son procesos de control.
- *ITIL:* Que está muy orientado a los servicios y sus objetivos, así como la reducción de costos.

Existe un número importante de normas ISO encargadas de apoyar en estas tareas; es importante aclarar que no es más que tomar un documento e ir siguiendo los pasos. Para certificarse se debe de pagar una cantidad importante a un certificador en el estándar y pasar por ciertas pruebas que él mismo determinará.

I.IX.- SEGURIDAD EN BASE DE DATOS.

La seguridad de las bases de datos es uno de los temas que más preocupan, el objetivo es mantenerla fuera del alcance de personas no autorizadas, en el mundo informático robar información de una base de datos es una de las prácticas comunes de las personas dedicadas a violar la seguridad de sistemas.

La información, que es lo más valioso que existe, es constantemente atacada y por ello los administradores de bases de datos deben estar alerta en todo momento.

El porqué alguien intentaría acceder a los datos, es algo que puede tener muchos orígenes, desde únicamente hacerlo por la satisfacción de decir que lo hizo, hasta por intereses por la información que podrían servir para llegar a causar un daño físico como un secuestro o el robo de dinero.

Por ejemplo, lo que pasaría si la seguridad de la base de datos de un banco fuera violada sería desastroso, ya que quien haya accedido a esa información podría hacer lo que quisiera con los números de cuenta de las tarjetas. Es por eso que empresas bancarias no escatiman esfuerzo ni dinero en cuanto a la seguridad de sus sistemas se refiere y, sobre todo, de las bases de datos.

La tarea de la seguridad es compartida, ya que no depende de una sola persona ni de un solo procedimiento, es por ello que se trata en capas.

- Seguridad de usuario

  Este tipo de seguridad se refiere a definir usuarios con privilegios a ciertos objetos, así se asegura que sólo los usuarios indicados tengan acceso a cierto contenido dependiendo de sus necesidades.

- Seguridad de aplicaciones

  Las aplicaciones desarrolladas en cualquier lenguaje de programación por los desarrolladores, sólo debe permitir el acceso a la información necesaria.

- Seguridad de RDBMS

  Como se vio en el módulo de Administración de bases de datos (pág. 29), existen 3 capas de seguridad para el acceso a las bases de datos, “acceso, base y objeto”, cada capa orientada a permitir o restringir acceso al sistema manejador dependiendo sus permisos.

- Seguridad de sistema operativo

  El sistema operativo por su parte, también debe de implementar seguridad, ya que de nada serviría restringir acceso a la base de datos si alguien roba el archivo que contiene los datos, es por ello que se recomienda uso de antivirus y firewall.

- Seguridad de red

  También es necesario asegurar que ninguna persona ajena pueda violar el acceso, para lo cual se puede denegar el acceso desde la red.

- Seguridad física

  Ésta es de las más difíciles de implementar, ya que se debe mantener toda evidencia física que pudiera en algún momento provocar que la seguridad de las bases de datos o el sistema operativo sean violados. Desafortunadamente, hay muchas personas que para no olvidar alguna contraseña, la escriben sobre algo y lo dejan a la vista de las personas que pudieran aprovechar el descuido.

De las capas descritas, las que son de interés del administrador de bases de datos son: La capa de seguridad de aplicación, la capa de seguridad de RDBMS (al cien por ciento) y la capa de sistema operativo, por lo que un administrador de bases de datos debe conocer algo de la administración de sistemas operativos, sobretodo como configurar un firewall, como instalar un antivirus y como quitar u otorgar permisos a las carpetas a usuarios no autorizados.

I.IX.1.- METODOLOGÍA PARA LA SEGURIDAD

Este es un conjunto de pasos que debe seguir un administrador de bases de datos como una tarea común.

Evaluación: Se debe evaluar que tan importante es la información manejada para que la protección implementada sea adecuada al nivel de seguridad que se necesita.

Protección: Implementar los mecanismos necesarios para cubrir las necesidades identificadas al evaluar.

Detección: Un administrador de bases de datos debe tener claro que no solo debe implementar protección, sino también estar alerta por cualquier posible ataque no contemplado.

Respuesta: Una vez detectado el ataque, deben existir los mecanismos necesarios para que en el menor tiempo posible se pueda solventar el problema y dejar la base de datos 100% segura nuevamente y estable.

Todo esto en aras de mantener siempre la base de datos confidencial, íntegra y disponible.

Los errores de una empresa son:

- No elegir bien al personal que va a administrar la base de datos.

- No darle el valor real a la información.

- Depender de un solo método de protección; ejemplo, Antivirus o firewall.

- No hacer caso a tiempo de posibles violaciones de acceso.

Algunos de los consejos más comunes para la implementación de protección de una base de datos son:

1.- Solo tener una cuenta root para el acceso al RDBMS

2.- Configurar el sistema operativo, para que sólo sea accedido al RDBMS desde ciertas IP's.

3.- No permitir el acceso remoto con el súper usuario.

4.- Deshabilitar los servicios que no se usen, de tal forma que no queden puertos abiertos para un posible ataque.

5.- Implementar políticas a los empleados, en donde ellos se comprometan a no facilitar las cuentas que se les asignen a personas ajenas a la institución.

I.X. - PERFORMANCE AND TUNNING.

El performance de una base de datos se refiere a monitorear constantemente que esté trabajando como debe, siguiendo las reglas.

Cuando se instala un Manejador de Bases de Datos, éste tiene una configuración inicial que únicamente sirve para iniciar el Manejador, pero que de ninguna manera son las configuraciones óptimas, sobretodo porque cada usuario tiene sus propias necesidades.

Hay muchos aspectos configurables en un RDBMS, de los principales están la memoria, el manejo de la cache, la red y el acceso.

Normalmente se tiene un archivo de configuración, en el cual se editan ciertos parámetros, los cuales pueden ser dinámicos que significa que no es necesario reiniciar el servidor para ver su efecto y los estáticos que si requieren reiniciar el servidor. El archivo de configuración en Sybase tiene la extensión **cfg** y se encuentra en el path del servidor.

Existen dos formas de editar los parámetros, la primera de ellas es desde un cliente utilizando alguno de los comandos, y la segunda es directamente en el archivo de configuración editándolo con un editor de textos como el block de notas, vi, emacs, etcétera. Es hasta cierto punto más fácil configurar desde el cliente, aunque a veces sucede que una configuración mal hecha no deja reiniciar el servidor por lo que no hay otra forma de reconfigurar, sólo editando el archivo.

Cada que hay un cambio a la configuración se crea un nuevo archivo, esto es así por si se desea regresar a una configuración previa al iniciar el servidor, únicamente se indica que se tome la configuración del archivo que contiene esa configuración, en Sybase esto se efectúa con el comando RUNSERVER.

I.X.1. - MEMORIA

Uno de los aspectos más cuidados al momento de la configuración es la memoria, un mal uso de ésta puede traer consigo muchos problemas, como que suceda un desbordamiento de memoria y el manejador ya no responda o definitivamente se detenga su funcionamiento. Esto puede muy costoso sobre todo si se habla de bases de datos altamente transaccionales.

El total de memoria física que se asigna a un RDBMS es dividido internamente dependiendo del manejador, parte de esa memoria es asignada a los ejecutables SQL, otra parte se queda como memoria estática y lo demás es asignada al caché de datos y el caché de procedimientos.

Si la memoria asignada al caché es suficiente, las consultas SQL que pudieran ser requeridas, con frecuencia se almacenarán y el servidor no tendrá necesidad de extraer los datos nuevamente, lo que le ahorraría tiempo que pudiera estar siendo requerido por otro proceso.

Hay que tomar en cuenta que si la configuración de la memoria no puede ser tomada por el manejador, éste no arrancará; una opción cuando esto sucede es revisar los archivos log que seguramente contendrán una descripción del problema. Cuando se requiere saber cuánta memoria asignar al RDBMS de la memoria total disponible en el sistema, hay que tomar en cuenta cual es la memoria requerida por el sistema operativo y por otras aplicaciones que también estén consumiendo memoria, esto es importante para saber si se cuenta con la memoria suficiente para levantar el servicio.

El número de usuarios conectados a la vez también se deberá tomar en cuenta al asignar memoria, ya que cada usuario conectado consumirá una cantidad de ésta. Si el número de conexiones sobrepasa la memoria disponible, el servidor se volverá lento y seguramente no se pueda realizar ni una tarea más.

Si se necesita incrementar memoria por usuario, hay que tomar en cuenta que la memoria que se había asignado a lo demás también debe de ser modificada, por lo que la memoria total utilizada por el servidor deberá ser también reconfigurada de modo que los cachés vuelvan a tomar sus valores óptimos.

I.X.2. - ALMACENAMIENTO

Dónde almacenar la información es una de las decisiones importantes a tomar, ya que se podría llenar el disco duro con facilidad, si no se toman las precauciones necesarias.

Las bases de datos distribuidas tienen la cualidad de poderse dividir en diferentes dispositivos físicos; también depende de las necesidades de la base de datos. Por ejemplo, si es una base de datos que contiene tablas que constantemente son accedidas, se podría optar por colocar en algún dispositivo diferente, los catálogos y así, cuando se quiera consultar un catálogo se asegura mayor velocidad, ya que la carga de peticiones estará en el otro dispositivo.

Cuando se va a dividir una base de datos, hay que tomar en cuenta muchos aspectos, entre ellos que la mayor cantidad de espacio es consumida por los logs, por lo que una de las cosas más recomendables es mandar el log a un dispositivo especialmente creado para él.

I.X.3. - CACHÉ

Un caché es un espacio de la memoria total del RDBMS, sirve para almacenar temporalmente datos que son solicitados por el usuario, utiliza bancos de buffer de mínimo 512K que es compuesto por páginas ligadas de 2K, 4K, 8K, 16K o 32K, el tamaño de página dependerá de la velocidad que se quiera alcanzar, entre más grande sea la página ligada más veloz será la consulta.

Las tablas o índices que utilizan estos cachés tienen que ser vinculados desde el dispositivo master. Un caché no puede ser eliminado a menos que no tenga objetos vinculados, a excepción del caché de datos que por ninguna razón puede ser eliminado.

I.XI.- MODELADO ORIENTADO A OBJETOS.

Cuando se inicia un proyecto, se deben considerar muchos factores, por ejemplo: si el tiempo estimado para el término del proyecto es suficiente, si se cuenta con las herramientas necesarias para realizar el proyecto, si se cuenta con el material humano debidamente capacitado para llevarlo a cabo adecuadamente; desafortunadamente, éstos y otros aspectos son poco tomados en cuenta por los desarrolladores de software, por lo que existe un número elevadísimo de proyectos que no se concluyen.

Uno de los factores principales es no tomar correctamente los requerimientos del cliente y no hacer un modelado detallado de las características del proyecto (Un modelo es una representación de algo real tomando aspectos importantes y dejando de lado los demás).

Cuando se realiza un modelo se tienen muchas ventajas, entre las que destaca la detección de fallas, el usuario puede ver como va a quedar y así determinar si realmente lo quiere o no y sobre todo permite documentar el proyecto para así tener un plan a seguir.

Existen algunas técnicas para el modelado de software, entre ellas UML (Unified ModelingLanguaje) el cual se utiliza para crear modelos visuales y

documentar los proyectos. Fue realizado por un grupo de desarrolladores con la finalidad de crear un estándar para el modelado; al ser un estándar, se asegura que sea entendible por la mayoría de los desarrolladores, no es un método de solución de un problema.

UML permite representar la posible solución de un problema, cuenta con varios tipos de diagramas. Estos diagramas se dividen en dos tipos: de estructura y de comportamiento; Los de estructura representan los componentes de lo que se quiere modelar, los de comportamiento representan las acciones que debe de seguir para cumplir el objetivo. No es forzoso implementar el proyecto con un paradigma orientado a objetos; sin embargo, UML es un excelente complemento de la programación orientada a objetos.

Dentro de los diagramas de estructura más importantes, están los diagramas de clases, componentes y los de objetos y dentro de los diagramas de comportamiento están los de actividades, casos de uso, estados y secuencia.

I.XI.1.- CASOS DE USO

Este tipo de diagramas se encargan de representar el comportamiento de un sistema; para ello, se representan los actores, que son los objetos que interactúan con el sistema y cada uno de las acciones que va a realizar, que son llamados casos de uso. Cada actor puede interactuar con uno o más casos de uso.

En la figura I.2 se muestra un sistema de venta en línea. Como se puede observar el actor se representa con una figurita como de una persona, aunque puede ser una base de datos o hasta otro sistema.

El ovalo representa el caso de uso y describe una funcionalidad en el sistema.

Figura I.2: Ejemplo de un diagrama de casos de uso

La forma en la cual se identifican los casos de uso para poder hacer una representación total del proyecto es:

- Teniendo conversaciones con el usuario final (La persona que requiere el sistema).
- Entrevistando a algunos expertos en la materia.
- Por experiencia en otros proyectos similares.

No hay un límite para el número de casos de uso que compondrán un sistema, si bien muchos casos de uso podrían hacer demasiado complicado entender el sistema, pocos podrían ocultar muchos detalles importantes.

Existen dos componentes llamados relaciones, que son muy importantes al realizar un modelo representado con diagrama de casos de uso. Estos son la relación **uses** y la relación **extends**. La relación **uses** (include) se utiliza cuando varios casos de uso van a hacer uso de una misma funcionalidad, esto ayuda a no tener que repetir muchas veces un mismo proceso. La relación **extends** se utiliza cuando un caso de uso desea aumentar la funcionalidad de otro. Aquí es cuando se hace uso de un paradigma orientado a objetos ya que se esta haciendo uso de la reutilización y la herencia que son pilares fundamentales de ese paradigma.

En la figura I.2 se tiene al actor cliente, el cual puede realizar ciertas acciones; entre ellas: consultar información, buscar artículos, llevar el articulo al carrito, etcétera. A su vez, al momento de efectuar un pago, el actor cliente puede interactuar con el banco si ese pago se efectúa con tarjeta de crédito; así mismo un tercer actor aparece, en este caso es el responsable de la tienda, que deberá registrar el depósito, cancelar la orden en caso de ser necesario y/o emitir una factura.

Los casos de uso se pueden documentar, lo que se llama descripción de casos de uso, y normalmente contienen una breve descripción, pre-condiciones, flujo principal, flujo alterno, flujos en caso de excepciones y post-condiciones.

Sin duda, este diagrama es el más importante cuando se modela por medio de UML, ayuda al usuario final a determinar si se aprueba o no lo que el sistema hará; ayuda a los desarrolladores a entender mejor el problema; al tester a probar si el producto final hace lo que en un inicio se planteo; y al líder de proyecto a manejar los tiempos de desarrollo.

I.XI.2.- DIAGRAMAS DE ACTIVIDAD

En este tipo de diagramas se representan las acciones o actividades que se habrán de efectuar durante todo el funcionamiento del sistema, explica cada

una de esa acciones de manera detallada y en algunas ocasiones se necesitará que se efectúen actividades al mismo tiempo; es por ello que se pensó en los diagramas de actividad como una forma de representar esas actividades asíncronas, además de que también puede representar actividades humanas; es decir, quien lo hace.

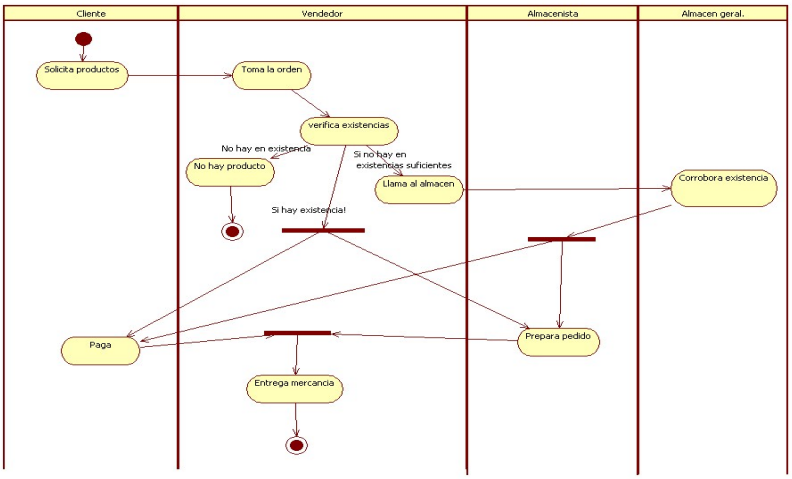

Figura I.3: Ejemplo de un diagrama de actividad

Siguiendo con un ejemplo muy similar al de diagrama de casos de uso, en la figura I.3 se puede ver que se inicia con la entidad cliente que lleva a efecto la actividad de solicitar un producto, el vendedor toma la orden y verifica existencias de ese producto, en dado caso de no haber termina el proceso; si no hay, llama al almacén general el cual corrobora existencias; si hay o existe en el almacén, se prepara el pedido a su vez que el cliente paga estando éstas dos últimas actividades sincronizadas; finalmente, se hace entrega del producto por parte del vendedor.

I.XI.3.- DIAGRAMAS DE CLASES

Este tipo de diagramas representan las clases que modelarán los objetos dentro del sistema, es por ello que se dice que los diagramas de clases son los que le dan la estructura a los sistemas desarrollados con la metodología de orientación a objetos.

Una clase es un modelo que definirá atributos y métodos. Cuando se desarrolla el sistema se crean objetos de esa clase; tales objetos tendrán esos atributos y esos métodos que se definieron en la clase, pero cada objeto se comportará de una forma independiente a los demás.

Existen tres tipos de clases que pueden ser definidas:

- Las clases de entidad, que son usadas cuando se quiere crear un objeto de esa clase y que este objeto tenga vida a lo largo de todo tiempo de vida del sistema (persistente).
- Las clases boundary, son las que se utilizan para mostrar información a los usuarios, normalmente imprimen formularios.
- Las clases de control, que son las encargadas de definir toda la lógica de negocio del sistema es decir que se hace con cada acción y como.

En la figura I.4 se observa las clases formAcceso, inicio y paginaWeb, éstas serian las clases boundary de el sistema; cuando se codifique éste, tanto formAcceso como inicio serán paginas HTML, mientras que paginaWeb sí será una clase que herede a todas las pantallas del sistema y la cual, como se observa, tiene todas las propiedades y métodos para definir una página.

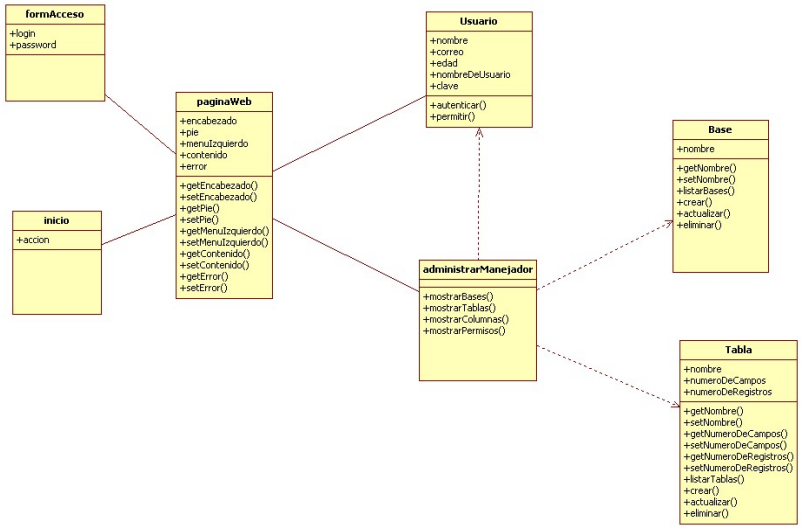

Figura I.4: Ejemplo de diagrama de clases

Las clases Base y Tabla son las clases entidad ya que con ellas se podrá manipular información y hacerla persistente durante el sistema.

Finalmente, la clase AdministrarManejador es una clase de control que únicamente recibe las peticiones de las diferentes vistas y decide qué clase de entidad llamar para generar una respuesta.

I.XI.4.- DIAGRAMAS DE SECUENCIA

Este tipo de diagramas representan las tareas tal como van sucediendo de forma secuencial, así como su línea de vida y los mensajes que se intercambian entre una actividad y otra.

En la figura I.5 se pone como ejemplo el acceso a un sistema, en el cual un usuario a través de un formulario de ingreso coloca su login y su contraseña, al tratar de enviarlo el sistema valida la sintaxis, de no serlo, se muestra el error en la misma pantalla y de serlo se hace la petición a administrarUsuario requiriendo el acceso, el cual verifica si éste existe mediante usuarioRegistrado.

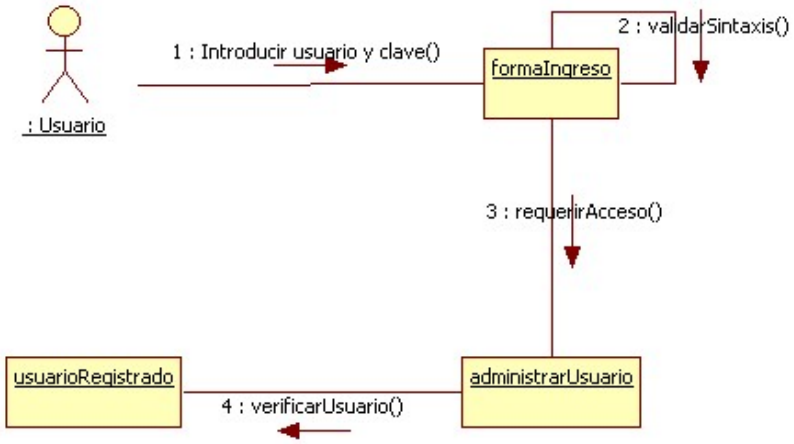

Figura I.5: Diagrama de secuencia

I.XI.5.- DIAGRAMAS DE COLABORACIÓN

Este tipo de diagramas, al igual que los de secuencia, representan tareas, sólo que a diferencia de ellos se hace de forma lineal, lo cual le deja claro a quien esta interpretando el diagrama qué tarea continua y en que momento. En la figura I.6 se puede ver la línea de pasos que se ejecutan uno tras de otro.

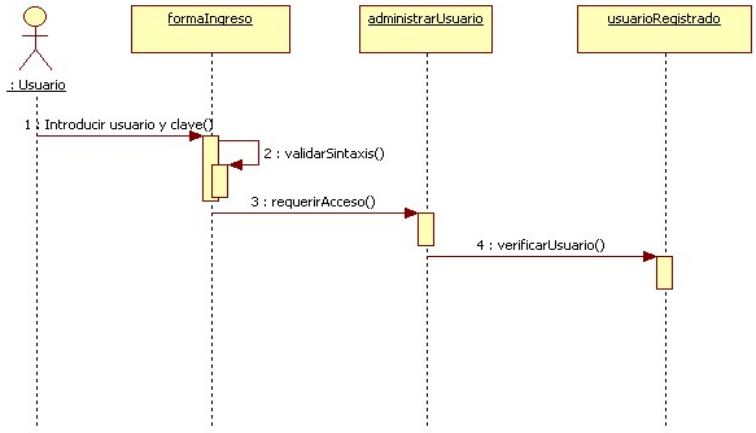

Figura I.6: Diagrama de colaboración.

I.XI.6.- DIAGRAMAS DE COMPONENTES

Sirven para mostrar todas las piezas que forman parte del proyecto en cuanto a software, éstos pueden ser librerías, documentos, aplicaciones, etcétera.

En la figura I.7 se muestra un ejemplo de diagrama de componentes; en este caso, la tecnología utilizada es Java. Haciendo uso del patrón de diseño MVC se puede observar que por un lado la vista estará compuesta de archivos HTML y JSP, por otro lado el control con los servlets y las diferentes API's utilizadas en el desarrollo del sistema, y el modelo que son todas las clases java y cualquier otra herramienta usada para comunicarse con las bases de datos, también se menciona la parte de base de datos, en este caso el uso del manejador PostgreSql.

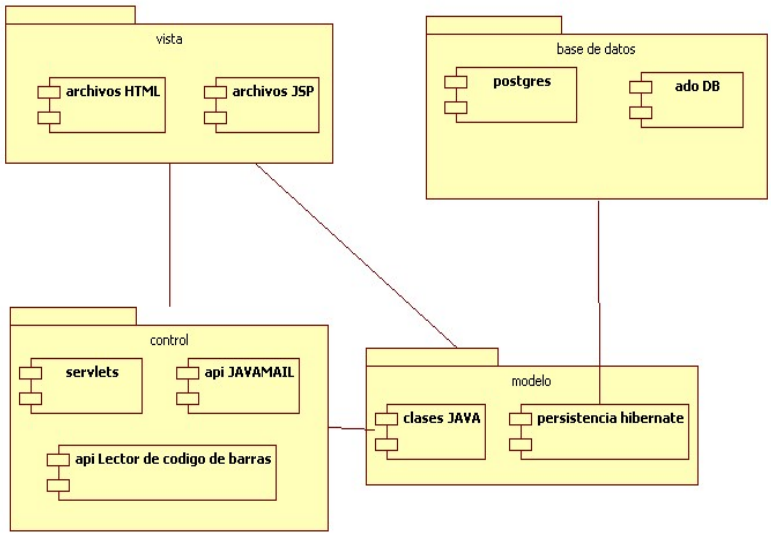

Figura I.7. Diagrama de componentes.

I.XI.7.- DIAGRAMAS DE DISTRIBUCIÓN

Este tipo de diagramas representa los nodos existentes en el sistema en el aspecto físico (hardware) y sus conexiones entre ellos.

Muestra la conexión entre un servidor de aplicaciones y un servidor de base de datos y qué dispositivos se conectan a ellos, si hay una impresora o un escáner conectado a una computadora también debe colocarse en el diagrama.

I.XI.8.- ARQUITECTURA UML

El lenguaje de modelado (UML) propone una arquitectura de trabajo para el desarrollo de un sistema, esta arquitectura es llamada Arquitectura de 4 + 1 vistas.

- *La vista lógica:* Muestra las clases en su nivel más básico, para este tipo de vista se usan los diagramas de clases, secuencia, colaboración y estado.
- *La vista de componentes:* Muestra como se distribuyen esas clases en paquetes, usando los diagramas de componentes.
- *La vista de proceso*: Muestra como se van a efectuar los procesos y tareas dentro del sistema, usa los diagramas de clases y de distribución.
- *La vista de distribución:* Muestra físicamente la conexión entre componentes físicos necesarios para el desarrollo y operación del mismo, sólo se apoya en los diagramas de distribución.
- La vista extra sólo explica como deben operar las otras 4 vistas

En este tipo de arquitectura no es necesario que todas las personas que integran el equipo de desarrollo conozcan las 4 vistas, la vista lógica sólo le será útil al arquitecto del sistema, la de componentes al programador, la de proceso al diseñador y la de distribución al administrador del sistema.

I.XI.9.- BASES DE DATOS ORIENTADAS A OBJETOS (OODB)

Este tipo de bases de datos utilizan una estructura similar a la de la programación orientada a objetos; de hecho, una clase entidad se convierte en una entidad que almacena datos para una base de datos orientada a objetos.

También los tipos de datos que se manejan son similares, ya que las OODB soportan colecciones, arreglos y objetos tipo CLOB (Objetos de caracteres) Y BLOB (Objetos binarios como imágenes o videos).

En la actualidad, el 95 por ciento de las empresas utilizan bases de datos relacionales, la negativa al cambio a bases de datos orientadas a objetos radica en la complejidad y costo económico y de tiempo que esto significaría, por lo que hay proyectos que se han desarrollado que hacen sencillo el proceso de cambiar el manejo de los objetos de la programación a lo relacional de la base de datos, evitando todo ese proceso al desarrollador, estos son llamados mapeadores objeto relacionales, de los más conocidos son hibernate. JDO Engine y PowerTier.

La ventaja al hacer uso de bases de datos orientadas a objetos es que este paso no es necesario, ya que el almacenamiento de objetos a bases de datos orientadas a objetos es transparente, una ventaja más es que cuando se desea

guardar valores múltiples no es necesario crear una tabla transitiva como en el modelo relacional.

I.XII.- TÓPICOS AVANZADOS DE BASES DE DATOS.

En la actualidad el manejo que se le da a la información almacenada en una base de datos es cada vez mayor, lo que hace que el rendimiento de la misma disminuya de manera importante, y en bases de datos altamente transaccionales es vital tener un buen rendimiento en todo momento, es por ello que se han desarrollado algunas herramientas y técnicas que proveen cierto beneficio en el manejo de la bases de datos.

I.XII.1.- MINERÍA DE DATOS

Consiste en un conjunto de algoritmos orientados a procesar datos, para así obtener un modelo que permita generar un análisis de la información.

Esto se realiza en 4 pasos:

- *Determinación de objetivos:* El cliente junto con algún especialista en minería de datos fijan los objetivos que se perseguirán; por ejemplo, el objetivo del IFE podría ser determinar por medio de la información que contienen en sus bases de datos, cuál será la tendencia en las próximas elecciones.

- *Pre procesamiento de los datos:* Se recopilan los datos necesarios, se hace una selección de los que realmente son importantes y necesarios para la investigación, se limpian los datos, esto quiere decir quitar información basura que no aportara nada a el proceso, esta etapa es muy costosa en cuanto a tiempo, ya que se lleva alrededor del 70 por ciento de tiempo de todo el proceso.

- *Se determina el modelo a seguir:* Después de haber procesado los datos se determina que modelo es mejor seguir para aprovechar más la información, estos modelos se determinan por medio de estadísticas que se representan gráficamente y permiten a su vez decidir el modelo el cual comúnmente es un algoritmo relacionados con inteligencia artificial.

- *Análisis de los resultados:* El cliente analiza si los resultados obtenidos son coherentes y si le aportan cosas nuevas a su investigación.

La minería de datos puede tomar dos caminos para determinar los modelos a seguir, el predictivo y el descriptivo.

En el predictivo, se supone un comportamiento dependiendo de datos recopilados; por ejemplo, si los votos a algún partido político han subido en 3%

cada año y el año anterior ese mismo partido obtuvo un 23% de votos, se puede predecir que tendrá un 26%.

En el descriptivo se hace un análisis minucioso de esa información y se toman en cuenta otros factores que pudieran influir en el resultado. Tomando en cuenta el ejemplo anterior, se diría que efectivamente ha ido creciendo la votación para ese partido, pero que por datos estadísticos anteriores y tomando en cuenta la cantidad de partidos, no es posible que en este año se llegue al 26%, ya que es un partido relativamente joven, y la interacción con las personas no fue buena en el último año. Ahí ya se están tomando en cuenta más factores que podrían afectar directa o indirectamente, pero que de ninguna forma pueden ser descartados.

Entre los algoritmos que son utilizados comúnmente para realizar la minería de datos se encuentran las redes neuronales, los algoritmos genéticos, árboles de decisión, reglas de inducción y la regla del vecino más cercano.

I.XII.2.- DATAWAREHOUSING

El datawarehousing no es más que una bodega que almacena datos representativos para obtener estadísticas que sean importantes para una empresa, por lo que cuando se crea el datawarehousing se eliminan los datos basura y se tiene una base de datos limpia y enfocada a un tema especifico.

Es decir, si una empresa necesita obtener diferentes reportes de ventas del último año, sería muy costoso estar haciendo eso sobre la base de datos que está en operación, por lo que se crea una copia y se extraen los datos más necesarios, y entonces a ese conjunto de datos históricos que se tiene se le puede consultar las veces que se quiera.

La desventaja del datawarehousing es que los datos no son totalmente actuales, ya que los nuevos registros y actualizaciones se seguirán efectuando sobre la base de datos operacional, y esos cambios no afectarán al datawarehousing hasta crear uno nuevo.

I.XII.3.- BASES DE DATOS MULTIDIMENSIONALES

Este tipo de bases de datos se almacenan en 3 dimensiones, normalmente una de esas variables es el tiempo, son popularmente conocidas como cubos de datos y son de gran utilidad, porque en un solo cubo de datos está contenida toda la información de la base de datos.

Por ejemplo, se quiere saber qué Productos habrá disponibles en el supermercado el 5 de Marzo, en el cubo buscaría esas coordenadas, productos * mercado * tiempo = punto x, este punto es único en el espacio del

cubo y tendría exactamente la información que se necesita de la manera más fácil y natural.

CONCLUSIÓN DEL CAPITULO

Esta es un breve informe de lo que se ve en cada uno de los módulos del diplomado de Administración de Bases de Datos, en cada uno de ellos se explica de manera resumida los aspectos más importantes y los términos manejados con mayor frecuencia en el curso y en el mundo de las bases de datos.

El orden en el que se colocó cada uno de los capítulos es importante ya que se inicia con una explicación a fondo de los conceptos mas básicos, después una breve introducción a lo que es el modelado, el manejo del lenguaje sql, entrando de manera importante a la parte de administración y seguridad de las bases de datos y finalmente, explicación de otros conceptos que se relacionan de manera directa con éstas, como la programación y técnicas avanzadas.

II.- PROYECTO DE ADMINISTRACIÓN DE MÚLTIPLES BASES DE DATOS

OBJETIVO

El desarrollo de este sistema tiene el objetivo de facilitar la tarea de los usuarios inexpertos en la administración de bases de datos que por algún motivo tienen la necesidad de interactuar con diferentes bases de datos; No es un sistema para expertos en la materia.

PROBLEMATICA

En muchas dependencias u organizaciones se carece de un equipo completo que gestione las tareas informáticas; normalmente, el desarrollador tiene que hacer funciones de analista, diseñador de sistemas, tester, diseñador gráfico y en la mayoría de las ocasiones de administrador de bases de datos.

En el desarrollo de software y sobre todo sistemas web se ha hecho imprescindible el manejo de información contenida en bases de datos, es por ello que los programadores tienen la necesidad de manejar esa información, que en el mejor de los casos le es solicitada al administrador de bases de datos que se encargará de crearle un usuario y darle una cuenta de usuario con permisos a la información que necesita. Desafortunadamente no siempre es así, empresas y dependencias de todo tipo y niveles carecen de un especialista en bases de datos, por lo que dejan la tarea de administración de sus bases de datos a personas que aun perteneciendo al rubro informático, no tienen el conocimiento suficiente para administrar una base de datos al cien por ciento.

El problema crece cuando la persona que desea realizar únicamente tareas básicas de administración, al no saber, recurre a buscar una solución rápida que en la mayoría de los casos es buscar una solución en internet. Esta no es una mala idea, el problema es que aun cuando existe un lenguaje estándar para el manejo de bases de datos (ANSI SQL), las sentencias para creación de objetos y permisos de usuarios no siempre son iguales en todos los manejadores de bases de datos.

VENTAJAS

En este proyecto, se realiza un entorno gráfico, que nos simplifica las tareas básicas de administración, tratando éstas de una manera similar, aun cuando sean sistemas manejadores de bases de datos diferentes; de esta forma, aquel usuario que necesite realizar tareas de administración básicas en uno de estos manejadores, le sea más fácil, intuitivo y no tenga que instalar los clientes de cada uno de ellos que a la larga terminarán haciendo lento su sistema y que no explotará en toda su capacidad; por otro lado, si el usuario posee una máquina con pocos recursos en cuanto a memoria, espacio en disco duro y un procesador no tan potente, puede aun así administrar sus bases de datos, ya que el trabajo pesado se le carga en su totalidad al servidor web.

En este caso el sistema se enfoca en 2 manejadores:

1.- MySql
2.- PostgreSql

La estructura del sistema es la siguiente:

Se tienen 5 módulos:

- Usuarios: Consiste en una interfaz para crear, modificar y eliminar usuarios.
- Objetos: Interfaz para crear, modificar y eliminar objetos, en este caso sólo tablas y vistas (a manera de ejemplo).
- Respaldos: Modulo con el cual se generan backups y restauran bases de datos desde un restore.
- Reportes: Únicamente muestra un reporte detallado de que permisos tiene cada usuario creado.
- Bases de datos: Interfaz para crear, modificar y eliminar bases de datos.

El sistema está programado con el lenguaje de programación Java en su versión para web, en este caso haciendo uso de servlets y jsp’s, además se hace uso del patrón de diseño Modelo-Vista-Controlador (MVC), ya que éste nos permite separar el código haciéndolo mas limpio e intuitivo, Java permite a su vez hacer mejor manejo de las excepciones que se pudieran tener, ya que se cuenta con los bloques try y catch, que se encargan de gestionarlas.

Diagrama de casos de uso

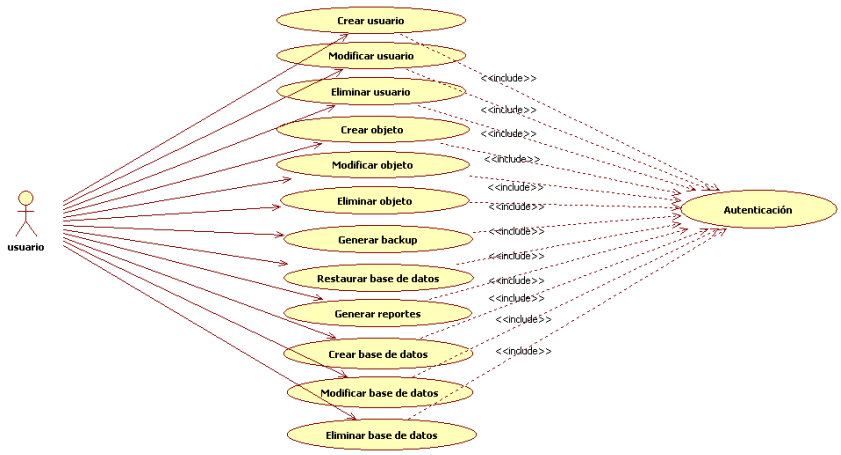

Figura II.1: Diagrama de casos de uso del proyecto.

Diagrama de clases

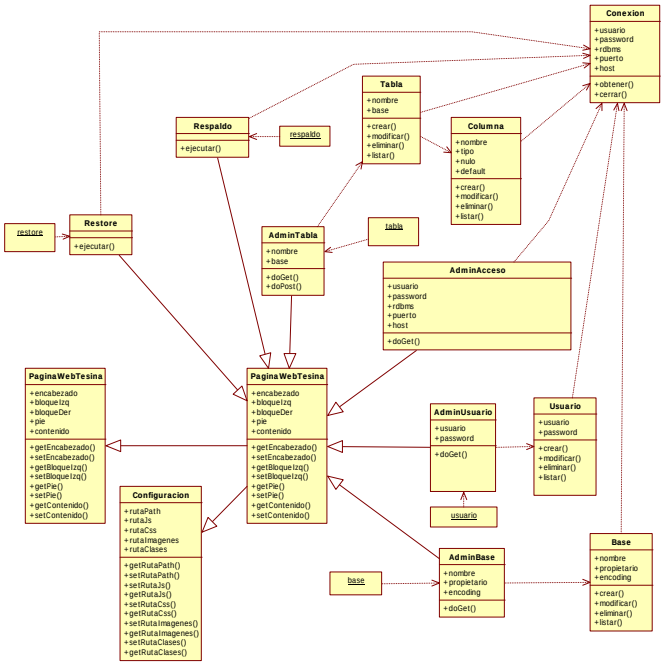

Figura II.2: Diagrama de clases del proyecto.

No se tiene un diagrama entidad-relación, ya que el sistema hace uso de las tablas de configuración del sistema al que se conecta.

FUNCIONAMIENTO

Como ya se menciono anteriormente, se hace uso del MVC, las clases se dividen en la siguiente forma.

Modelo

- Conexión
- Driver propietario (Dependiendo del RDBMS elegido)
- Base
- Tabla
- Usuario
- Columna

Las ultimas cuatro clases son las encargadas de la interacción del controlador con el modelo, reciben una petición del controlador, usan a la clase conexión que a su vez usa un driver para lograr la conexión y efectuar la tarea requerida, tanto base, tabla, usuario y columna se conocen como entity beans, tienen métodos get y set de cada atributo y estas clases finalmente servirán para manipular cada entidad como objeto.

Vista

- base
- restore
- respaldo
- tabla
- usuario
- PaginaWeb
- PaginaWebTesina
- Configuracion

Son las interfaces con las que interactúa el usuario, cada vista es un archivo jsp y cada una de ellas internamente hace uso de 3 clases que la forman, que son las siguientes: PaginaWeb que es la estructura general de una pagina, PaginaWebTesina que implementa las pantallas especificas del sistema y Configuracion que se encarga de definir las rutas de los componentes del sistema, esto con el fin de que no se definan rutas dentro de los archivos jsp, y así facilitar su manejo.

Controlador

- AdminAcceso
- AdminUsuario
- AdminTabla
- AdminBase
- Restore
- Respaldo

En esta parte encontramos a los Servlets, que son los que se encargan de la administración y determinar el flujo de trabajo del sistema (lógica de negocio), normalmente reciben una petición de la vista y en base a la información obtenida determinan la acción, es entonces cuando hacen una petición a alguna clase modelo y quedan en espera de respuesta para también ellos dar una respuesta a la petición inicial normalmente a una vista.

II.I.- ACCESO

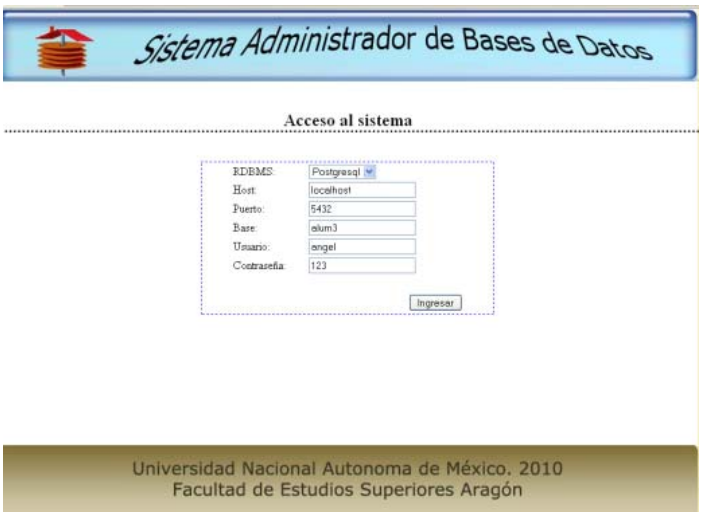

Figura II.3: Pantalla de logeo al manejado por medio del sistema

Para acceder al sistema, se solicita el host del servidor, el puerto, el usuario, el password, el nombre de la base de datos y el manejador deseado, figura II.3.

Si alguno de los datos no son correctos, saldrá un mensaje de error indicándolo, como lo muestra la figura II.4.

Para hacer una conexión a una base de datos PostgreSql, mediante una terminal tecleamos:

psql -d nombre_base -U usuario -p puerto -h host -w password

Y para conectarse a MySql, es de la siguiente manera:

mysql -h host -u usuario -p password

Esto es transparente para el sistema, ya que al utilizar java únicamente se requiere de un driver, ya sea odbc o jdbc, en este caso se hace uso de jdbc para ambos manejadores.

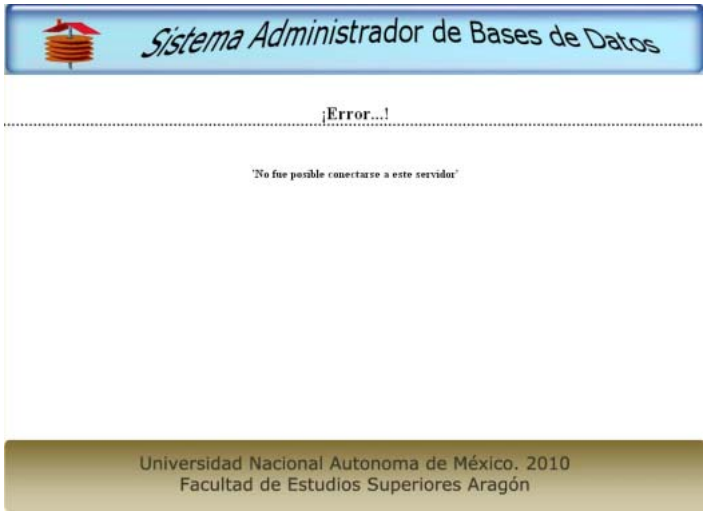

Figura II.4. Mensaje de error al intentar ingresar.

Utilizando el driver jdbc de PostgreSql, se hace de la siguiente manera, Class.forName, sirve para registrar el driver, esto se hace para indicarle al sistema que se hará uso de él, y con el objeto DriverManager, se obtiene la conexión utilizando la cadena de conexión, dependiendo de la base de datos a conectar.

```
String driver = "org.postgresql.Driver";
String connectString = "jdbc:postgresql://"+host+":"+puerto+"/"+base;
Connection con = null;
Class.forName(driver);
con = DriverManager.getConnection(connectString, user , pass);
```

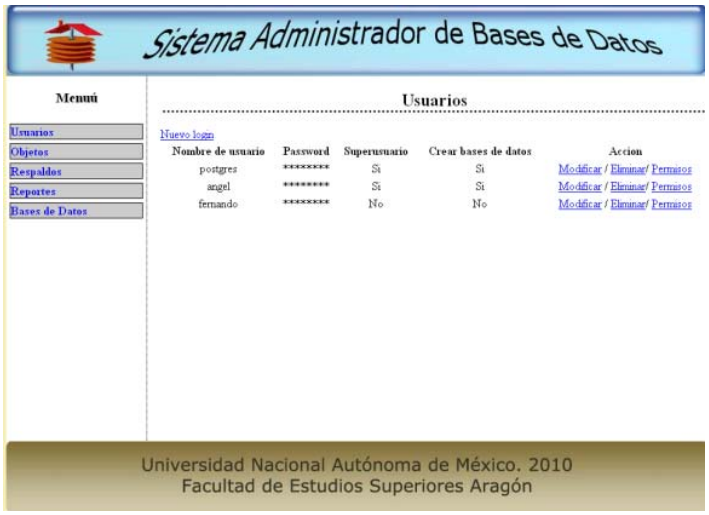

Figura II.5. Acceso exitoso al sistema.

Para MySql seria algo muy parecido, lo único que cambia es el driver, que como se ve a continuación es com.mysql.jdbc.Driver, y la cadena de conexión, lo demás es exactamente igual.

```
String driver = "com.mysql.jdbc.Driver";
String connectString = "jdbc:mysql://"+host+":"+puerto+"/"+base;
```

Esta conexión, es la utilizada durante todo el sistema. Si los datos ingresados son los correctos, éste se conecta y se crean 5 variables en sesión, para no volver a preguntar esa información al usuario hasta terminar la sesión, después mostrará un menú con 5 módulos, como se puede observar en la figura II.5.

II.II.- USUARIOS

La administración de usuarios es la primera opción del menú, una vez que se está logueado, al dar click en esa opción, se muestra un listado de los usuarios existentes en ese momento.

II.II.1.- CREACIÓN DE USUARIOS

La creación de usuarios es una de las tareas más comunes que se presentan, ya que cada usuario que pretenda acceder a la base de datos le debe de ser asignado un nombre de usuario y un password.

PostgreSql

CREATE USER *usuario* WITH *opciones*

Las opciones son las siguientes:

```
CREATEDB | NOCREATEDB
   CREATEUSER | NOCREATEUSER
   IN GROUP groupname [, ...]
PASSWORD 'password'
VALID UNTIL 'tiempo'
```

La sentencia más sencilla es sólo creando el usuario sin asignarle ningún tipo de permiso:

CREATE USER usuario WITH PASSWORD ‘password’

Si se quisiera dar el permiso de creación de bases de datos se le agrega CREATEDB al final a la sentencia, CREATEUSER para dar permiso de creación de usuario, agregarlo a un grupo o grupos usando IN GROUP e indicarle hasta que fecha estará valida esa cuenta.

MySql

Existen 3 formas de crear usuarios en MySql:

La primera de ellas es utilizando la sentencia GRANT, con la cual se puede a la vez que se crea el usuario asignarle permisos sobre los diferentes objetos o sobre toda la base de datos.

GRANT acciones ON nombre_base.objetos TO 'usuario'@'host' IDENTIFIED BY 'password';

Las acciones, son lo que puede el usuario ejecutar (select, insert, delete, drop, etcétera.), los objetos son las tablas, vistas, cursores, etcétera.

La segunda forma es utilizando la instrucción create user, ésta a diferencia de la anterior sólo permite crear al usuario, la tarea de otorgar permisos queda pendiente por el momento.

CREATE USER *usuario* [IDENTIFIED BY [PASSWORD] '*password*']

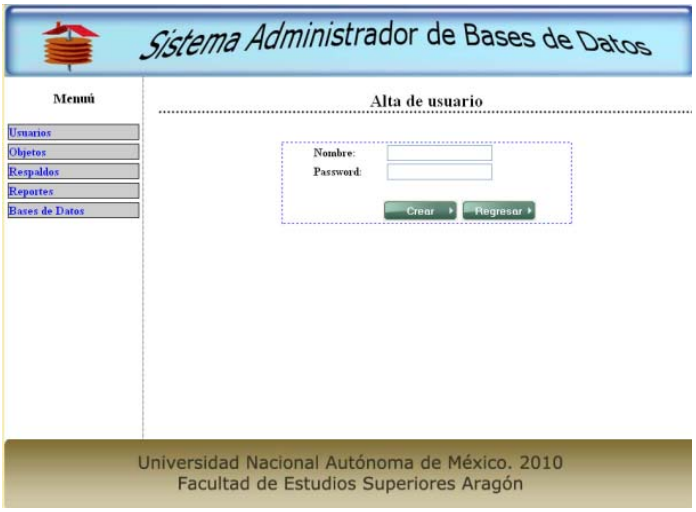

Figura II.6. Pantalla de creación de usuario.

La tercer forma es insertando un registro directamente sobre la tabla user de la base de datos mysql, que es la que controla el acceso de los usuarios.

INSERT INTO user VALUES('host','usuario', PASSWORD('password') ,
'Y','Y','N','N','N','N','N','N','N','N','N','N','N','N','N','N',
'N','N','N','N','N','N','N','N','N','','','','',0,0,0,0);

Con esta instrucción se crea el usuario y se puede a su vez, otorgar permisos generales a éste, al ser una instrucción tan larga, se vuelve un poco complicado su manejo, el nombre de las columnas no es muy intuitivo sobre lo que hacen o afectan al usuario creado.

En el sistema:

Para crear un usuario, se pide el nombre de usuario y el password, como se puede ver en la figura II.6.

Como se puede observar, la manera en que se crea el usuario por medio del sistema es la más simple de todas, es sólo crear un usuario y asignarle un password; realmente, este es el objetivo del sistema, ser simple, que el usuario final no tenga que involucrarse en detalles que tal vez por el momento no le interesen o que ni siquiera conozca y sólo le causen confusión.

II.II.2.- ACTUALIZACIÓN DE USUARIOS

De igual forma que en la creación de usuarios, ésta es una tarea donde cada manejador implementa su propia manera para hacerlo.

PostgreSql

ALTER USER *usuario* WITH *opciones*

Las opciones son las mismas que las empleadas en la creación del usuario.

MySql

En este manejador no existe como tal un procedimiento para actualizar un usuario creado, esto se divide en 2 comandos, la primera de ellas es:

RENAME USER antiguo_nombreusuario TO nuevo_nombreusuario

Que únicamente sirve para cambiar el nombre de usuario, y el segundo comando es:

SET PASSWORD = PASSWORD('nuevo_password')

Con lo cual se cambia el password al usuario, aunque ejecutar este comando alteraría el password del usuario conectado en este momento, por lo que se utiliza una forma completa de esa instrucción, quedando de esta forma:

UPDATE mysql.user SET Password=PASSWORD('nuevo_password') WHERE User='nombre_usuario';

Esto lo que hace es modificar directamente la tabla user, que es la encargada del manejo de los usuarios, y por lo que se tendría que tener tanto permiso de crear usuarios como sobre la base de datos mysql para poder ejecutar este procedimiento.

En el sistema:

En ésta opción, el sistema muestra en un formulario, el usuario y un campo vacío para la contraseña, si el usuario es alterado o el campo de contraseña se rellena con algún texto, entonces procederá a hacer la actualización de datos. Como ya se mencionó, el manejo que se le da a esta tarea por cada manejador es totalmente diferente, sin embargo, el sistema realiza este trabajo por debajo, para que el usuario no tenga que preocuparse por ello.

II.II.3.- ELIMINAR USUARIOS

Contrario a la creación de usuarios y a la actualización, esta acción tiene una sentencia común para ambos manejadores, con algunas consideraciones:

PostgreSql

La instrucción para borrado de objetos (incluyendo usuarios) es DROP, por lo que queda de la siguiente forma:

DROP user nombre_usuario

MySql

En MySql es igual solo que hay que tomar en cuenta que hasta antes de la versión 5.0.2 sólo se podían borrar usuarios que no tuvieran permisos sobre otros objetos, por lo que antes de ejecutar el drop había que eliminarle esos permisos con la instrucción revoque.

En el sistema:

Aunque el usuario en sesión pueda ser un inexperto en el rubro de la administración de bases de datos, se le da la opción de eliminar usuario por que puede ser una tarea necesaria en algún momento, aunque al momento de elegir la opción se le alerta de que es una operación riezgosa y que no podrá resuperar esos datos.

Al dar clic sobre la liga eliminar se pide una confirmación, si ésta es positiva se procede a eliminar el usuario y listar nuevamente los usuarios existentes, se observa de manera clara en la figura II.7.

Manejo interno

La vista usuario.jsp es la encargada de recibir los datos de la acción solicitada, mediante un formulario, una vez que se hace la petición, el servlet AdminUsuario recibe los parámetros y determina la acción, en este caso hace la petición al bean Usuario, que se encarga por medio de su método correspondiente a la acción y al manejador en el que se esta logueado de

efectuar la tarea, esto lo hace utilizando a la clase Conexion; el servlet recibe respuesta y determina si manda un mensaje de error a la vista error.jsp o a usuario.jsp indicando que la acción solicitada fue exitosa. En las tres acciones anteriores se efectúa el mismo manejo interno del sistema, la única diferencia es que el método ejecutado del bean cambia dependiendo de la acción.

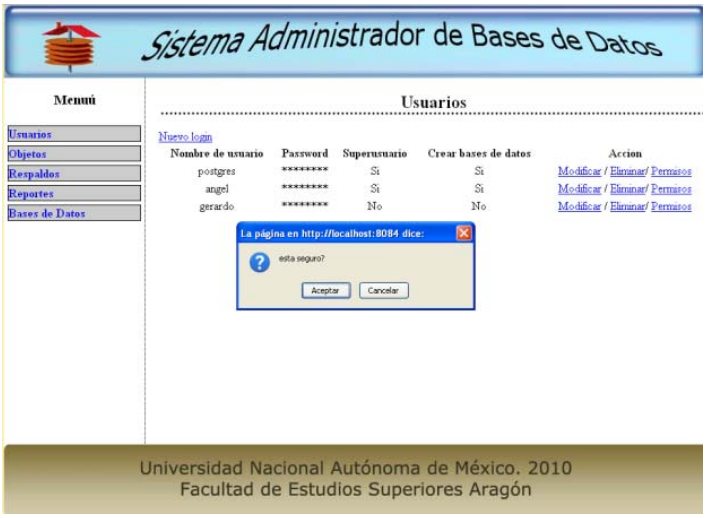

Figura II.7. Eliminar un usuario.

II.III.- OBJETOS

Un objeto es una entidad dentro del manejador, las tablas, índices, vistas, triggers, procedimientos almacenados son ejemplos de objetos.

Para almacenar información primero se debe crear el contenedor de esa información, o tal vez adecuarlo a las necesidades de nuestra información, y muchas ocasiones un objeto es necesario eliminarlo para mantener limpia la base de datos.

Administrar los permisos a un objeto no es tan simple como pareciera serlo, el ejemplo más común es cuando una persona necesita acceder a información en una base de datos, pero no se quiere que acceda a toda la información por lo que únicamente se le da permiso de acceso a los objetos que realmente necesita y no a toda la base de datos.

En otro ejemplo sería, que el mismo usuario necesita ver cierta información de una tabla pero no de todos los campos, se crea para ello una vista con sólo los campos necesarios para el usuario y se le da permiso sobre la vista no sobre la tabla, por lo que los campos que no están contenidos en la vista el usuario no podrá ver.

En este módulo se pueden crear objetos, eliminarlos, modificar algunas de sus características, siempre que el usuario con el cual se ingreso al sistema tenga permisos para realizar estas tareas.

Para ello se cuenta con un submenú que lista los objetos; una vez elegido el objeto, se mostrará un listado de tablas o vistas, según sea el caso , y desde ahí se podrán elegir alguna de las opciones sobre el objeto en turno, si la opción es borrar se listan los objetos del tipo seleccionado, para elegir cual se eliminará; y una vez elegido se pide confirmación, si la opción es modificar se listan los objetos del tipo seleccionado y una vez elegido se carga un formulario con los datos del objeto con opción a modificar.

Para la elaboración del proyecto sólo se contemplan las tablas y las vistas, ya que son los objetos más comunes a administrar.

II.III.1.- TABLAS

Para poner un listado de tablas, como se observa en la figura II.8, se utiliza el estándarSQL, ya que en ambos manejadores se puede consultar de:

```
SELECT * FROM INFORMATION_SCHEMA.TABLES
WHERE TABLE_TYPE='BASE TABLE'
```

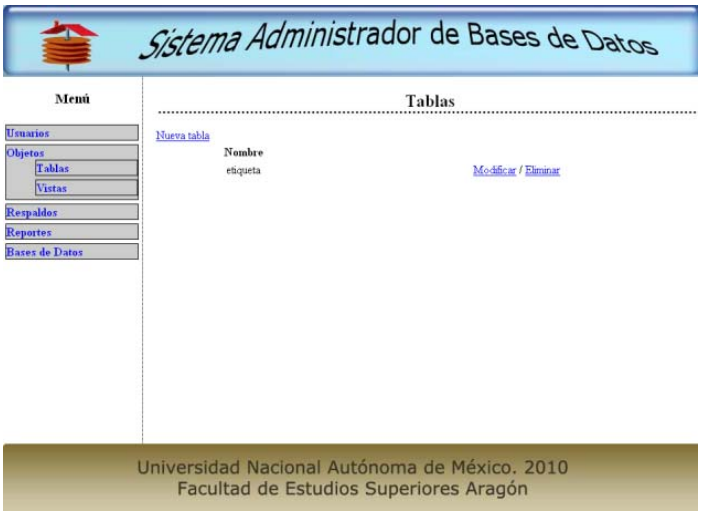

Figura II.8. Listado de tablas.

En la tabla “TABLES” se puede obtener información sobre las tablas existentes, los campos importantes que dan esta información son: table_schema que indica a que esquema pertenece la tabla, table_name que muestra el nombre de la tabla y table_type que indica de que tipo es, también se puede sacar ésta información directamente en la consola, para PostgreSql es con \dt y para MySql con show tables, a pesar de que MySql tiene un information_schema éste no es un esquema, ya que MySql no los soporta (es una base de datos), no así en PostgreSql, ahí si es un esquema.

II.III.1.1.- CREACIÓN DE TABLAS

PostgreSql

Para crear una tabla en Postgres se utiliza la siguiente instrucción:

CREATE TABLE *nombre_tabla* ([
  { *nombre_columna tipo_dato* [DEFAULT *valor_default*] [*constraint_columna* [
, ...]]
    [*constraint_tabla]*
  }
])

Los constraint de columna se definen de la siguiente forma:

[CONSTRAINT*nombre*]
{ NOT NULL | NULL | UNIQUE |AUTO_INCREMENT|PRIMARY KEY |
REFERENCES *tabla_referencia* [(*columna_referencia*)] [ON DELETE *action*
] [ON UPDATE *action*] }

Las acciones pueden ser:

NO ACTION

> Indica si la actualización o borrado del registro produce una violación de referencia a una llave foránea, a excepción de que se indique que la tabla es defarrable, esto quiere decir que se estaría checando en cierto tiempo si la columna aún esta relacionada.

RESTRICT

> Es igual que no action, sin excepciones.

CASCADE

> Borra o actualiza las columnas relacionadas.

SET NULL

> Las pone con un valor nulo.

SET DEFAULT

> Les pone su valor por default en caso de tener.

Los constraints de tabla se especifican de la siguiente manera:

[CONSTRAINT *nombre*]
{ UNIQUE (*nombre_columna* [, ...]) |
  PRIMARY KEY (*nombre_columna* [, ...]) |

FOREIGN KEY (*nombre_columna* [, ...])
REFERENCES *tabla_referencia* [(columna_*referencia* [, ...])]
[ON DELETE *action*] [ON UPDATE *action*]
}

MySql

CREATE [TEMPORARY] TABLE [IF NOT EXISTS] nombre_tabla(
nombre_columna tipo_dato NOT NULL | NULL valor_default
[AUTO_INCREMENT] [UNIQUE [KEY] | [PRIMARY] KEY], … n) |
[CONSTRAINT [symbol]] PRIMARY KEY [tipo_constrain]
(nombre_columna,...)REFERENCES nombre_tabla_referenciada
(columna_referenciada,...)[ON DELETE RESTRICT | CASCADE | SET NULL |
NO ACTION][ON UPDATE RESTRICT | CASCADE | SET NULL | NO ACTION]

Como se ve en la instrucción anterior, la creación de un tabla es similar para ambos manejadores, existiendo pocas diferencias, de lo que se observa que difiere una de otra es que en MySql se puede indicar que se cree la tabla de forma temporal y sólo si no esta creada, además de que también se puede indicar algunas configuraciones que se quieren que esa tabla tenga, por ejemplo la codificación que manejara, donde estará físicamente en el árbol de directorios, un password, etcétera.

En el sistema:

Como se ve en la figura II.9, la interfaz pide el nombre de la tabla, los campos y sus tipos de datos, si aceptara valores nulos o no, así como marcar si alguno o algunos de esos campos actuaran como llave primaria de la tabla, ésta es la forma más básica de crear una tabla.

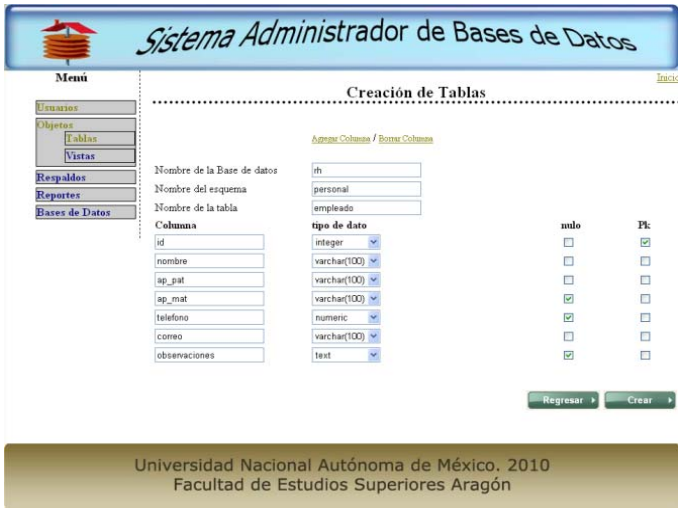

Figura II.9. Creación de tablas.

En un principio, sólo habrá un cuadro para ingresar una columna, y dependiendo de cuantas se necesiten dando clic en agregar columna irán apareciendo.

Nota: En este caso, para MySql no se pide el nombre del esquema, ya que éste no maneja esquemas

II.III.1.2.- MODIFICAR TABLAS

Para modificar la definición de una tabla se utiliza la instrucción sql:

ALTER TABLE [nombreTabla] [opción];

La opción es lo que se va a modificar a la tabla, dentro del sistema se encuentran 4 posibles modificaciones que son muy comunes, ellas se ilustran en la siguiente figura II.10.

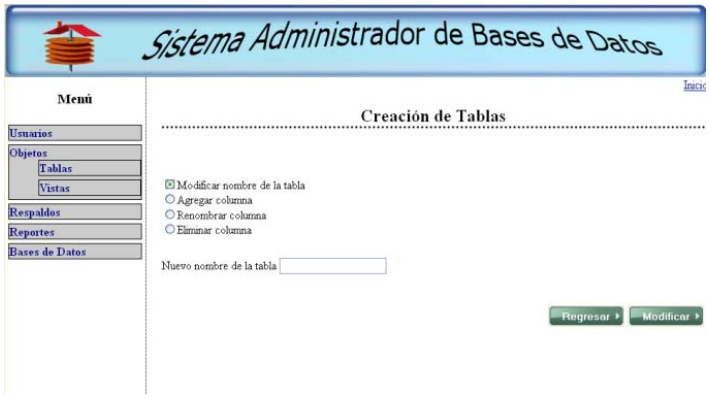

Figura II.10. Modificación de tablas.

La primera de ellas es modificar el nombre de una tabla, en la cual sólo se solicita el nuevo nombre, internamente tanto PostgreSql como MySql hacen uso de la instrucción estándar, la cual es:

RENAME TO [nuevoNombre].

En las otras tres opciones ocurre algo similar, agregar columna solicita una nueva definición de la misma (nombre, tipo de dato y si es nulo o no), al dar clic sobre el botón modificar, ésta nueva columna será agregada a la tabla ya definida, renombrar columna le redefine el nombre y borrar columna la elimina de manera definitiva de la definición de la tabla.

Manejo interno

La vista tabla.jsp, es la encargada de recibir los datos de la acción solicitada mediante un formulario, una vez que se hace la petición, el servlet AdminTabla recibe los parámetros y determina la acción, en este caso hace la petición al

bean Tabla, que se encarga por medio de su método correspondiente a la acción y el manejador al que se esta logueado de efectuar la tarea, esto lo hace utilizando a la clase Conexion; el servlet recibe respuesta y determina si manda un mensaje de error a la vista error.jsp o a tabla.jsp, indicando que la acción solicitada fue exitosa.

II.III.2.- VISTAS

La lista de vistas tanto en PostgreSql como en MySql se encuentra en la base de datos information_schema, tabla views, por lo que con la siguiente instrucción se puede acceder a esa información:

SELECT * FROM INFORMATION_SCHEMA.VIEWS

En la figura II.11 se puede observar el listado de todas las vistas definidas por el usuario, no importando en que esquema o base de datos se encuentran.

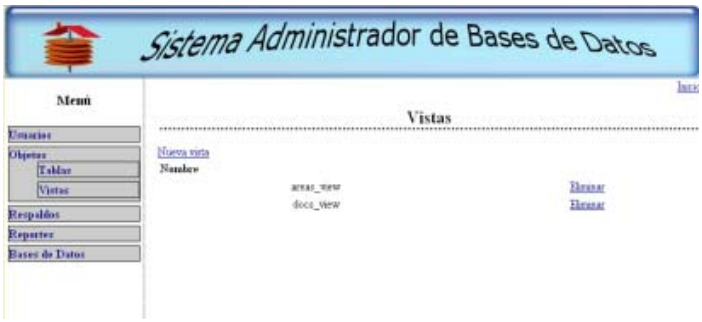

Figura II.11. Listado de vistas.

II.III.2.1.- CREACIÓN DE VISTAS

PostgreSql

CREATE [OR REPLACE] [TEMP | TEMPORARY]
VIEW nombre_vista AS *consulta*

La opción or_replace significa que si ya existe la vista se reemplaza por la nueva definición. La de temp o temporary se colocan si se desea que la vista sólo exista durante la sesión actual, de esta forma una vez que termina la sesión la vista es borrada.

MySql

CREATE [OR REPLACE]
VIEW nombre_vista
AS consulta

Para MySql la instrucción para crear una vista es igual, con excepción que aquí no se pueden definir vistas temporales.

En el sistema:

Como se observa en la figura II.12, la creación de una vista implica elegir una o varias tablas y las columnas a las que se desea acceder desde la vista.

Se listan las tablas existentes, una vez elegida la o las tablas se abre una pestaña inmediatamente abajo con las columnas que pertenecen a esa tabla, al dar clic en crear, la vista será generada y se redirecciona al listado de vistas (figura II.11).

Figura II.12: Creación de vistas.

II.III.3.1.- ELIMINAR VISTAS

Para eliminar vistas se hace uso del estándar SQL para ambos manejadores, DROP VIEW nombre_vista, dentro del sistema la forma de eliminar una vista es igual a la de las tablas, en el listado aparecerá el link eliminar, y una vez seleccionado solo se pedirá confirmación.

Manejo interno

En este caso es un poco diferente a los casos de usuario y tabla, aquí se tiene una interfaz llamada vista.jsp, la cual manda el nombre de tabla o tablas, columnas y condición, al servlet AdminVista, éste a su vez recibe la petición y determina la acción que va en uno de los parámetros recibidos, forma el query y directamente efectúa la acción usando la clase Conexión.

II.IV.- RESPALDOS

Los respaldos son esenciales al realizar una buena práctica de administración, motivos existen muchos pero los principales son:

II.IV.1.- PÉRDIDA INVOLUNTARIA DE INFORMACIÓN

El hecho de estar almacenando información importante en una base de datos, conlleva el riesgo de que en cualquier momento éstos puedan perderse por algún factor ajeno a nosotros y que no este contemplado; por ejemplo, un terremoto, inundaciones, incendios, etcétera. Estos podrían dañar el dispositivo físico en el cual se almacenan los datos y por consecuencia su pérdida total.

Es por ello, que constantemente las bases de datos deben ser respaldadas, dependiendo de cual es el uso de esas bases de datos, de esta forma, en un caso de emergencia como el ya mencionado pueda recuperarse la información en su totalidad o lo más cercano a su totalidad.

Ya se ha hecho hincapié en el concepto de información y el valor que ésta tiene o adquiere, y es precisamente este motivo por el cual respaldar las bases de datos resulta esencial, si la base de datos se perdiera por cualquier razón el costo seria altísimo, especialmente en donde el manejo de información es vital para el desarrollo y mantenimiento de grandes negocios.

II.IV.2.- TRASLADO DE INFORMACIÓN

Cuando se desea llevar una base de datos de un lugar a otro, se efectúa haciendo un respaldo (backup) de la base de datos y cuando se ha trasladado el archivo al lugar destino, realizar un restore.

PostgreSql.

Para crear un backup en una base de datos, PostgreSql utiliza el comando pg_dump, el cual tiene varias opciones; por ejemplo, se puede indicar si el backup se generara en un tar o en texto plano, si se guardara la estructura de la base de datos y los datos o solo estos últimos.

MySql.

Para el caso de MySql, se efectúa algo similar, a excepción de que el programa ejecutado es mysqldump.exe, las opciones son las mismas.

En el sistema:

Dentro del sistema, como se ve en el siguiente código, con el comando exec de la clase Runtime se ejecuta el programa pg_dump.exe únicamente pasándole las opciones –U para indicar al usuario, -E para indicar la codificación, -d para pasarle el nombre de la base de datos y –W para indicar la contraseña.

```
String commandShell = "C:\\Veramex Technologies\\Wappo
1.0\\PostgreSQL\\bin\\pg_dump.exe -U " + usuario + " -E " + codificacion + " -d "
+ base + " –W "+password;

Process child = rt.exec(commandShell);

String commandShell = "C:\\Archivos de programa\\MySQL\\MySQL Server
5.1\\bin\\mysqldump.exe -U " + usuario + " -E " + codificacion + " -d " + base + "
";

Process child = rt.exec(commandShell);
```

Visualmente, al acceder a la administración de respaldos se muestra una interfaz, donde únicamente se indica nombre de usuario y clave, adicionalmente se puede indicar que codificación será utilizada para guardar los datos y la ruta del archivo, una vez ingresada esta información se generara el archivo con extensión .sql, que será almacenado en la raíz de nuestro sistema (en el caso de Windows será C:/, como se observa en la figura II.13.

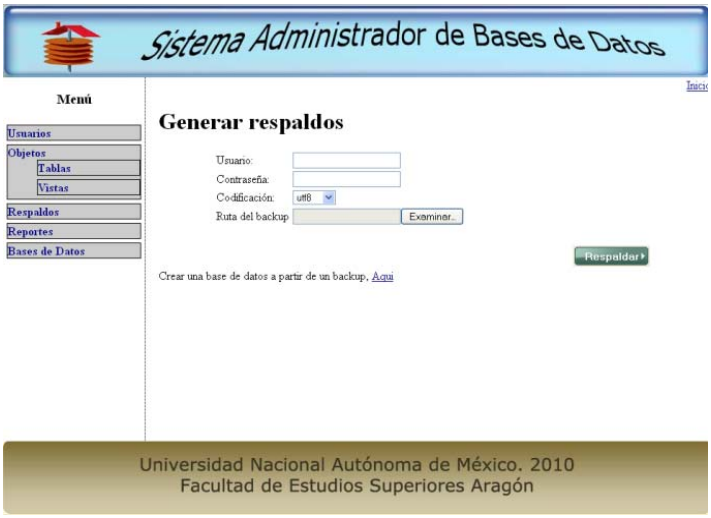

Figura II.13: Pantalla de creación de backups.

Manejo interno

La interfaz respaldo.jsp manda mediante un formulario los datos para la generación de un respaldo (backup), el servlet AdminRespaldo recibe la petición, mediante su método ejecutar y la clase Conexión efectúan la consulta para la generación del backup, el cual se guarda en la ruta especificada por el usuario en la interfaz.

Para la generación del backup se hace uso de la clase de escritura de archivos, fileOutputStream como se ve en el siguiente código:

```
FileOutputStream fout = new FileOutputStream(backupNombre);
```

```
OutputStreamWriter writer = new OutputStreamWriter(fout, "utf8");
writer.write(outStr);
writer.flush();
writer.close();
fout.close();
```

Donde:

fout es la instancia de creación de archivo, en ese momento únicamente se genera, después, se abre para poder escribir en él, instanciando entonces el objeto writer de la clase OutputStreamWriter, con el método writer se inserta al archivo todo el código sql que forma parte del backup, y finalmente se cierra y guarda.

También, se puede entrar en el mismo modulo de administración de respaldos a la opción cargar respaldo (Restore), que lleva a una interfaz donde se indicara la ruta del archivo .txt o .sql y en ese momento se efectúa la acción de cargar la base de datos respaldada, figura II.14.

El respaldo de información es un tema muy delicado, ya que puede ser objeto de robo de información si no se maneja con el suficiente cuidado, es por ello que para ver la opción de administración de respaldos se requiere una cuenta de administrador, sino no aparece en el menú.

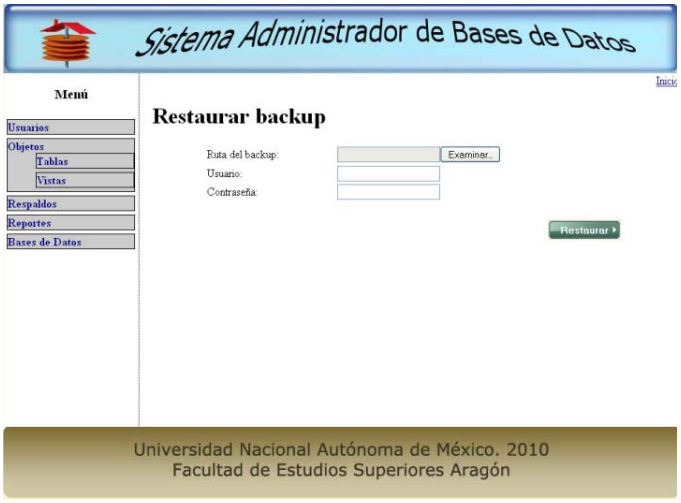

Figura II.14: Restaurar bases de datos desde un backup.

Manejo interno

La interfaz restore.jsp, manda mediante un formulario los datos para restaurar una base de datos desde un backup, el servlet AdminRestore recibe la petición, mediante su método ejecutar, la clase Conexión y el propio backup efectúan la consulta para restaurar la base de datos.

En este caso especifico, se hace uso de una clase llamada ProccessBuilder, que se encarga de generar una consola para efectuar la acción, como se muestra en la siguiente línea de código.

```
pb = new ProcessBuilder("pg_restore", "-v", "-D", "-f", "backup.sql", "-U", user, dbase);
```

II.V.- REPORTES

Este módulo sólo es de consulta de información de los permisos con los que cuentan los usuarios para acceder a los objetos existentes en el servidor, se pueden generar dos tipos de reportes los cuales son mandados en un archivo de Excel (xls).

II.V.1.- REPORTES POR USUARIO

Muestra un listado de usuarios y sus permisos que tiene dentro del manejador de bases de datos.

PostgreSql

Dentro de cada una de las bases de datos creadas se genera el esquema pg_catalog, el cual a su vez tiene un conjunto de tablas y vistas que contienen la información con la que opera el manejador.

Cuando un usuario se conecta o cada que quiere ejecutar alguna acción, el sistema manejador de bases de datos (en este caso PostgreSql) verifica si tiene permisos para realizar la acción solicitada, esto internamente lo hace realizando una consulta a la vista pg_shadow, que a su vez extrae los datos de la tabla pg_authid. El reporte generado contiene el nombre de usuario, su id de usuario, si puede crear o no bases de datos, si es o no un superusuario, si puede o no actualizar las tablas del system catalog, el password encriptado y la fecha de caducidad de el password, como se observa en la figura II.15, en este ejemplo, se puede ver que la contraseña es una cadena de números y letras revueltas, esto es por que así la guarda el propio sistema manejador por seguridad, por lo que ni aun el administrador podría saber la contraseña de algún usuario, la opción en caso de que el usuario la pierda seria que el administrador la cambie. En el campo de fecha de caducidad de la contraseña si dice infinity significa que la cuenta no tiene una fecha de expiración, por lo que siempre estará activa, hasta que el administrador la elimine.

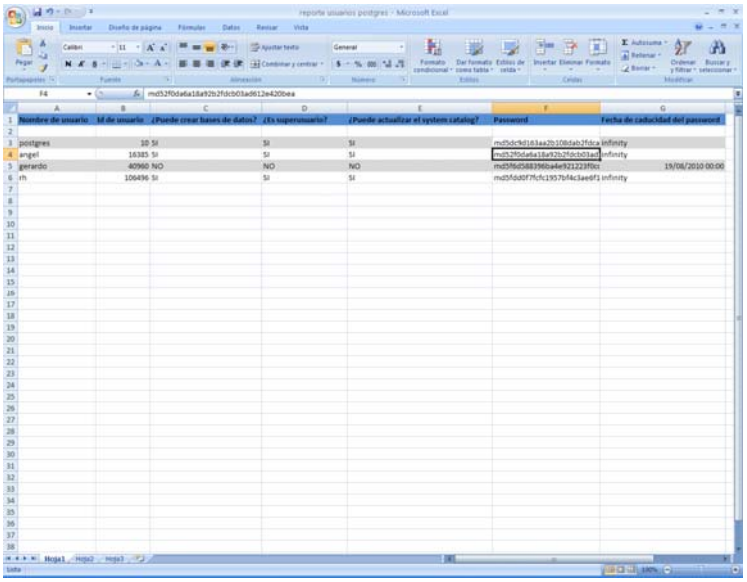

Figura II.15: Reporte de permisos por usuario de PostgreSql

En teoría un superusuario es un usuario que tiene todos los privilegios, en general esto aplica también para PostgreSql, aunque el permiso de actualizar el system catalog va en contra de ello, ya que si éste no lo tuviera un superusuario, no podría de ninguna manera modificar las tablas y vistas del esquema pg_catalog, lo que significa que no puede modificarle permisos a un usuario.

MySql

En este sistema manejador es mas fácil localizar las tablas que contienen la información de los permisos, cuando MySql es instalado se crea una base de datos llamada mysql, ésta contiene un conjunto de tablas que utiliza el propio manejador para determinar su operación.

La tabla específica de la cual podemos obtener los permisos de los usuarios se llama user, y la forma en la cual guarda esa información es un poco diferente a la de PostgreSql, como se ve en la figura II.16. De inicio está la columna host, que dictamina que el usuario que puede conectarse desde ese host al servidor, en el ejemplo que se ve en la figura se observa localhost, lo cual indica que el usuario root solo podrá conectarse al servidor desde la maquina que contiene el servidor, si se requiere que el usuario root se conecte desde otro u otros host, se necesitara dar permiso por cada uno de ellos y eso significaría un registro mas por cada uno en la tabla user de mysql, para indicar todos se utiliza el comodín %, aunque específicamente para el usuario root no es recomendable dar este permiso por seguridad.

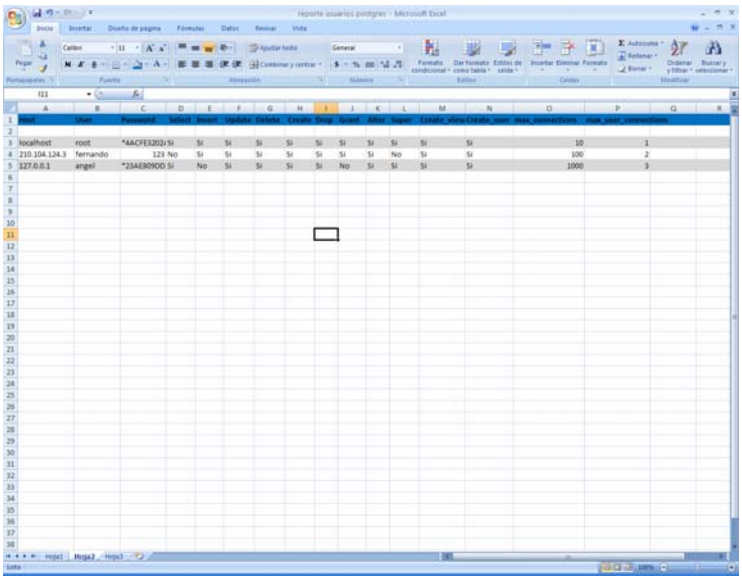

Figura II.16: Reporte de permisos por usuario de MySql.

También se puede ver que el password está encriptado y lo mas importante, que por cada comando o acción se tiene un permiso especifico, es decir, se puede indicar que el usuario angel tenga permiso de todo, excepto de hacer inserts y grant como se observa en la figura mencionada.

La dos ultimas columnas también son muy importantes, max_connections y max connections_user, indican cuantas veces el usuario podrá conectarse en una hora, si el usuario rebasa esa cantidad de conexiones no podrá acceder al sistema.

Figura II.17 Creación de reportes por usuario.

En el sistema:

Para obtener estos reportes desde el sistema solo se deberá dar clic en el menú reportes y elegir usuarios, lo que inmediatamente generara el reporte dependiendo del manejador al que se este conectado en ese momento, esta opción del menú solo aparece si la cuenta con la que se ingreso al sistema es de administrador.

II.VI.- BASES DE DATOS

Administrar una base de datos es la tarea más básica para un administrador, ya que es el contenedor sobre el cual se almacenan las tablas que contienen los datos, la información de los usuarios existentes y los permisos que poseen esos usuarios.

II.VI.1.- CREACIÓN DE BASES DE DATOS

PostgreSql

La sentencia SQL para la creación de bases de datos en el manejador PostgreSql es:

```
CREATE DATABASE nombre_baseDatos
    [ [ WITH ] [ OWNER [=] propietario ]
           [ TEMPLATE [=] nombre_baseTemplate ]
           [ ENCODING [=] codificación ] ]
```

Como se observa, lo único que se requiere es indicar un nombre para la nueva base de datos, los demás datos son opcionales, aunque es muy importante indicar el tipo de codificación que utilizará la base de datos, y el propietario de la misma; ya que de no ser así, por default será creada en codificación dada en la configuración inicial del manejador y el propietario o dueño de la base de datos será el usuario que la creó (en este caso el que ingreso en el sistema). La opción template se utiliza para indicar que se cree una copia de una base de datos específica, de no indicarse este dato se utiliza una base de datos que se crea al momento dela instalación de PostgreSql llamada `template1`.

Es importante mencionar que para poder crear bases de datos, se debe tener el permiso de createdb.

MySql

Esta sentencia es similar a la de Postgres, aunque hay algunas diferencias importantes, como se ve a continuación.

```
CREATE {DATABASE | SCHEMA} [IF NOT EXISTS] nombre_baseDatos
[DEFAULT] CHARACTER SET codificación
```

La condición if not exists es opcional, si se indica significaría que si la base de datos a crear no existe la cree; es una opción importante sobretodo para usuarios novatos, ya que de no ponerla y existir, la base de datos será borrada y creada una nueva. Por medio de esta misma consulta se pueden crear esquemas y tendría las mismas opciones.

En el sistema:

Gráficamente al ingresar al módulo de administración de bases de datos, se listan todas las bases de datos existentes en el servidor, como se muestra en la figura II.18, donde únicamente se puede ver el nombre de la base de datos y su codificación así como el botón de eliminar, lo cual será permitido sólo si el usuario con el que se ingreso es administrador o dueño de esa base de datos que se quiere eliminar.

Para crear bases de datos en la misma interfaz, se elige la liga crear base de datos, que lleva a una pagina donde sólo se muestran 2 campos donde se indicara el nombre, la codificación elegida, si el manejador en el que se esta en sesión es PostgreSql también se puede ingresar el nombre de una base de datos template y el dueño, esto se puede observar en la figura II.19.

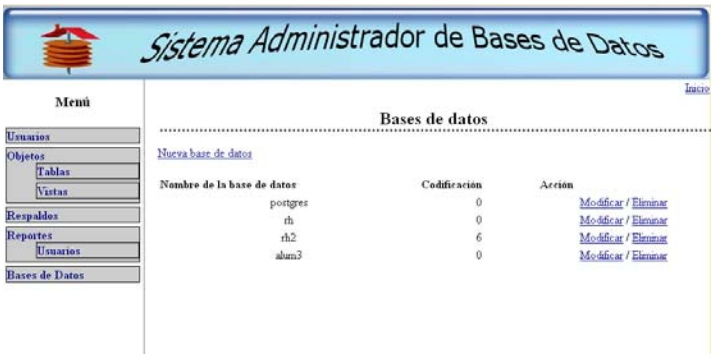

Figura II.18 Listado de las bases de datos existentes.

Figura II.19. Formulario de creación de bases de datos.

II.VI.2.- MODIFICACIÓN DE BASES DE DATOS

PostgreSql

```
ALTER DATABASE nombre_baseDatos RENAME TO nuevo_nombre

ALTER DATABASE nombre_baseDatos OWNER TO nuevo_propietario
```

En este manejador de bases de datos se permite modificar el nombre y el propietario, aunque con instrucciones separadas.

MySql

```
ALTER {DATABASE | SCHEMA} [nombre_baseDatos]
    [DEFAULT] CHARACTER SET codificación
```

El MySql a diferencia de PostgreSql se puede efectuar el cambio de nombre como de codificación en una sola instrucción.

En el sistema:

En el entorno gráfico se elige en el listado la liga modificar, entonces aparece una ventana como la que se ve en la figura II.20, donde se puede cambiar el nombre a la base de datos si es que se coloca texto en el campo "*nuevo nombre*", cambiar el encoding o ambos.

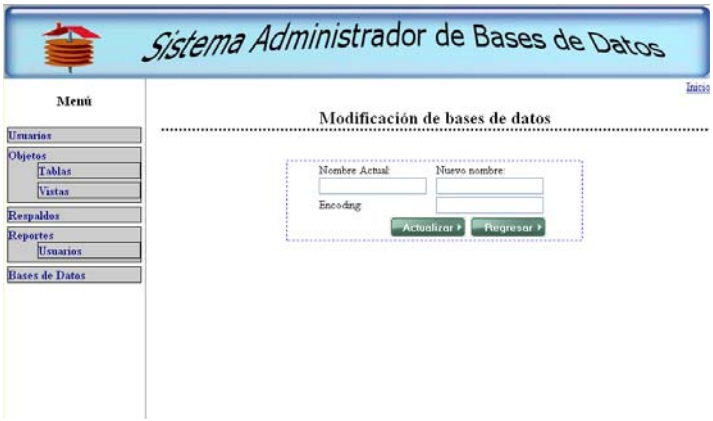

Figura II.20: Modificación de una base de datos.

II.VI.3.- ELIMINAR UNA BASE DE DATOS

Para borrar una base de datos únicamente desde el listado se da clic en la liga eliminar, dependiendo de la que se desee, tanto para PostgreSql como para MySql la sentencia es la misma.

Drop database nombre_baseDatos

El sistema alertará si realmente se está seguro de eliminar esa base de datos, ya que es una acción que no hay forma de cancelar ni mucho menos recuperar, a menos que se haya hecho un backup de esa base de datos.

Manejo interno

La interfaz base.jsp recibe los datos de la acción solicitada mediante un formulario, ya sea crear , modificar o eliminar una base de datos; una vez que se hace la petición, el servlet AdminBase recibe los parámetros y efectúa una petición al bean Base, que se encarga por medio de su método correspondiente a la acción y el manejador al que se esta logueado de realizar la tarea, esto lo hace utilizando a la clase Conexion; el servlet recibe respuesta y determina si manda un mensaje de error a la vista error.jsp o a base.jsp indicando que la acción solicitada fue exitosa.

CONCLUSIONES DEL INFORME DEL PROYECTO

En el proyecto, se contemplan sólo algunas de las acciones más comunes y básicas que son necesarias para un usuario inexperto, aquella persona que tiene la necesidad de realizar tareas de administración de una base de datos, sin conocer a fondo como se administra ésta.

Por supuesto que no son las únicas tareas que se pueden ejecutar; sin embargo, son suficientes para ejemplificar lo que se puede hacer desde un sistema web y como se pueden manejar los hilos de un manejador u otro de manera totalmente transparente para el usuario final.

Como se ve en el desarrollo del sistema, las diferencias entre manejador y manejador son bastantes, y se intentó que el sistema hiciera las tareas que ambos manejadores pueden realizar. Es cierto que MySql y PostgreSql no son los únicos manejadores de bases de datos; sin embargo, son los más comunes en el desarrollo de aplicaciones.

El objetivo de colocar la sintaxis de ambos manejadores para cada acción que efectúa el sistema, es para que se vean las diferencias entre los dos manejadores, que palabras reservadas utiliza un manejador que no reconoce el otro y viceversa, y hacer notar como el sistema maneja esas instrucciones por debajo sin que el usuario final note la diferencia, y es precisamente ahí donde se cumple con el objetivo trazado al inicio del proyecto.

CONCLUSIONES GENERALES

El estudio de las bases de datos confirma que, efectivamente, la información es muy valiosa, tanto que evitar su pérdida por el motivo que sea y mantener

su privacidad son procuradas en todo momento por especialistas en la administración.

Es muy difícil tratar de explicar todo lo que se relaciona con las bases de datos, ya que es un mundo enorme y para cada uno de los apartados que se tocaron aquí podría tomarse todo un curso y especializarse en ello.

También se observa que aún cuando existe el estándar SQL, cada manejador implementa ciertas sentencias de forma diferente, ya que tienen la libertad de poder agregar lo que quieran en cuanto a servicio para sus clientes, y es justo aquí cuando difícilmente un administrador experto en algún manejador de bases de datos podría serlo para otro.

Al ser tan importante y valiosa la información, los administradores de bases de datos que son expertos en la materia, obtienen muchos ingresos aunque el tiempo que invierten en ello es enorme, un administrador de bases de datos está prácticamente obligado a mantener a salvo esa información y a que permanezca disponible todo el tiempo, sobre todo si los datos que alimentan las bases de datos que administra se obtienen de algún sistema que opere las veinticuatro horas del día; al ser estas personas tan bien pagadas, sólo los grandes consorcios y las grandes empresas dedican ese dinero para alguien con ese perfil y los demás prefieren ya sea capacitar al personal que ya tienen o delegar la responsabilidad de administración a alguien que está ligado al mundo de la informática, aunque no sea un experto.

Por ello, para las personas que se dedican a la ingeniería en computación e informática, es necesario tener un conocimiento medio en estos temas, ser capaz de hacer tareas de administración, o en su defecto, tener conocimiento de lo que existe, para que en caso de ser necesario poder especializarse en ello.

Para el manejo de las bases de datos es necesario un experto en la administración de las bases de datos, dependiendo del manejador que se esté ocupando; una persona dedicada únicamente a esa tarea y que además tenga una muy buena comunicación con todas las personas que directa o indirectamente hagan uso de la información contenida en las bases de datos que administra.

Además, es recomendable que como buen administrador, haga respaldos en el tiempo y manera adecuados al uso de las bases de datos; que revise periódicamente los logs en busca de errores que pudieran estar sucediéndose o accesos o intentos de acceso no autorizado a la información; estar alerta sobre cualquier indicio de ataque y tener los mecanismos principalmente de prevención adecuados; así como estar prevenido para una recuperación inmediata de la información en caso de pérdida de datos.

Es de tomar en cuenta que los mecanismos de seguridad no sólo son a nivel lógico, sino también físico; es decir, no es suficiente tener un buen antivirus, un buen firewall y una contraseña de veinte caracteres numéricos, alfanuméricos y símbolos, si se tiene anotada la contraseña en un papel frente a la computadora en un lugar accesible por cualquier persona.

GLOSARIO

ANSI:

Instituto Nacional Americano de Estandarización, supervisa los estándares en procesos, productos y servicios y pertenece a la organización internacional de estándares ISO.

ATRIBUTO:

Es el nombre que se le da a las columnas de una tabla en las bases de datos relacionales.

BASE DE DATOS:

Es un repositorio de información, lugar donde se guarda información de manera persistente para ser accedida en cualquier instante.

CHAR:

Tipo de dato de carácter, solo acepta un carácter.

DBO:

Propietario o dueño de una base de datos.

DCL:

Lenguaje de Control de Datos, se encarga de los permisos a los usuarios sobre los objetos.

DD:

Diccionario de Datos, sirve para definir los componentes de un esquema de una base de datos.

DDL:

Lenguaje de definición de datos, parte del SQL para la creación y borrado de objetos.

DEFAULT:

Asignación de un valor en caso de no indicar uno.

DISPOSITIVO:

Físicamente es un espacio que se crea para asignar algo a él.

DML:

Lenguaje de manipulación de datos, también es parte del SQL y sirve para interactuar con los datos contenidos en los objetos.

EMACS:

Es un editor de texto de distribuciones Linux.

INT:

Tipo de dato entero.

IP:

Es la dirección de una computadora en internet, se compone por 4 octetos se parados por un punto.

JOIN:

Significa unir dos o más consultas para obtener un único resultado.

LOGS:

Son los archivos que se definen para guardar los errores que pudiera generar un manejador en cualquier momento.

MAPEADOR:

Se encarga de traducir los datos de una base de datos relacional a objetos y viceversa.

NULL:

Significa ausencia de información.

NUMERIC:

Tipo de dato numérico, acepta solo valores numéricos.

PARTICIONAMIENTO:

Seccionamiento de un disco duro, división en partes.

RDBMS:

Sistema administrador de bases de datos relacionales.

SQL:

Lenguaje Estructurado de Consultas, es un estándar proveído por el ANSI para la interacción con los manejos de bases de datos.

TIPADO:

Definido por un tipo de dato.

TRIGGER:

Lanzador, se efectúa cuando ocurre algún evento.

TUPLA:

Así se le conoce a los renglones de las tablas.

UML:

Lenguaje de Modelado Unificado, sirve para modelar sistemas.

VARCHAR:

Tipo de dato alfanumérico de tamaño variable, se ajusta al tamaño de la cadena que contiene.

BIBLIOGRAFIA

- Apuntes del diplomado de Administración de bases de datos
  Generación 12
  Noviembre 2008 – Agosto 2009

- Introducción a los sistemas de bases de datos.
  Date, C. J.
  Prentice Hall

- Los sistemas de información en empresa
  Alberto Gómez Gómez y Nicolás de Abajo Martínez

- Diseño y programación de bases de datos
  Ángel Covo Yera
  Colección didáctica escolar

- Bases de datos relacionales
  Fray León Osorio Rivera
  Instituto Tecnológico Metropolitano.

- Bases de Datos desde Chen hasta Codd
  Irene Luque Ruiz, Miguel Angel Gómez-Nieto
  Alfaomega

- http://www.postgresql.org/docs/8.1/static/app-psql.html
- http://dev.mysql.com/doc/refman/5.0/es/connecting-disconnecting.html
- http://dev.mysql.com/doc/refman/5.1/en/tables-table.html
- http://dev.mysql.com/doc/refman/5.1/en/information-schema.html
- http://troels.arvin.dk/db/rdbms/