

Curso de introducción a las bases de datos y SQL

Manual del alumno



Indice

1 Introducción al curso.....	5
1.1 Objetivo de este curso.....	5
1.2 Manual del alumno.....	5
1.3 Ejercicios prácticos.....	5
1.4 Requisitos para atender a este curso.....	5
2 Introducción a las bases de datos.....	6
2.1 Objetivo del capítulo.....	6
2.2 Que es una base de datos.....	6
2.3 Componentes de una base de datos.....	6
2.4 Cliente/Servidor.....	6
3 Terminología y conceptos.....	7
3.1 Objetivo del capítulo.....	7
3.2 Términos específicos.....	7
3.2.1 Tabla.....	7
3.2.2 Registro.....	7
3.2.3 Campo.....	7
3.2.4 Vista.....	7
3.2.5 Procedimiento.....	7
3.2.6 Función.....	7
3.2.7 Paquete.....	7
3.2.8 Secuencia.....	7
3.2.9 Disparador.....	7
3.2.10 Esquema.....	8
3.2.11 Sinónimo.....	8
3.2.12 Indice.....	8
3.3 Abreviaciones.....	8
3.3.1 DBMS.....	8
3.3.2 RDBMS.....	8
3.3.3 SQL.....	8
3.3.4 DDL.....	8
3.3.5 DML.....	8
3.3.6 ODBC.....	8
3.3.7 ADO.....	8
3.3.8 JDBC.....	8
3.4 Conceptos.....	9
3.4.1 OLTP.....	9
3.4.2 DWH.....	9
3.4.3 Replicación.....	9
3.4.4 Multi-tiers.....	9
4 Servidores SQL.....	10
4.1 Objetivo del capítulo.....	10
4.2 Servidores SQL comerciales.....	10
4.2.1 Oracle.....	10
4.2.2 MS SQL Servidor.....	10
4.2.3 Teradata.....	10
4.2.4 DB2 y Sybase.....	10
4.3 Servidores SQL libres.....	11
4.3.1 MySQL.....	11
4.3.2 PostGreSQL.....	11
4.3.3 Firebird.....	11
4.4 Herramientas de desarrollo.....	11

4.5 Instalación de MySQL.....	12
5 <i>Análisis de requisitos</i>	13
5.1 Objetivo del capítulo.....	13
5.2 Tipo de base de datos.....	13
5.3 Implementación de la base de datos.....	13
6 <i>Creación del modelo de datos basados en las necesidades</i>	14
6.1 Objetivo del capítulo.....	14
6.2 Análisis de requisitos.....	14
6.3 Diagramas Entidad Relación.....	14
6.4 Normalización del modelo de datos.....	15
6.4.1 Primer forma normal.....	15
6.4.2 Secunda forma normal.....	15
6.4.3 Tercera forma normal.....	16
6.5 Integridad referencial.....	16
6.5.1 Clave primaria.....	16
6.5.2 Clave secundaria.....	16
6.6 Restricciones de dominio.....	16
6.6.1 Not null.....	16
6.6.2 Check constraints.....	16
6.6.3 Unique constraints.....	16
6.7 Tipos de datos.....	17
6.7.1 Tipos generales.....	17
6.7.2 Tipos disponibles en MySQL.....	17
6.7.2.1 Principales tipos de caracteres.....	17
6.7.2.2 Principales tipos numéricos.....	17
6.7.2.3 Principales tipos de fecha.....	18
6.7.2.4 Información completa.....	18
7 <i>DDL – Sentencias de definición de datos</i>	19
7.1 Objetivo del capítulo.....	19
7.2 Create table.....	19
7.3 Alter table.....	19
7.4 Drop table.....	19
8 <i>DML - Sentencias de manipulación de datos</i>	20
8.1 Objetivo del capítulo.....	20
8.2 Insert.....	20
8.3 Update.....	20
8.4 Delete.....	20
8.5 Commit y rollback.....	20
8.6 Select.....	21
8.7 Where.....	21
8.8 Count.....	22
8.9 Sum, avg, min, max.....	22
8.10 Distinct.....	22
8.11 Order by.....	22
8.12 Uniones	22
8.13 Subconsultas.....	22
8.14 Agrupaciones.....	23

<u>9 Rendimiento.....</u>	<u>24</u>
<u>10 Ejercicios.....</u>	<u>25</u>

www.SolucionJava.com

1 Introducción al curso

1.1 Objetivo de este curso

En este curso vamos a ver una introducción a las bases de datos. Esta introducción les va a servir por cualquier tipo de base de datos que desea utilizar, pero como solo es una introducción, no será suficiente para administrar estas bases de datos, si no para hacer encuestas básicas en ellas.

1.2 Manual del alumno

Este manual del alumno es una ayuda para el alumno, para tenga un recuerdo del curso. Este manual contiene un resumen de las materias que se van a estudiar durante el curso, pero el alumno debería de tomar notas personales para completas este manual.

1.3 Ejercicios prácticos

Para captar mejor la teoría, se harán ejercicios con los alumnos, para probar la teoría y verificar la integración de la materia.

También, el alumno podrá copiar sus códigos en un disquete al fin del curso para llevarse, con fin de seguir la práctica en su hogar.

Como base de datos de prueba utilizaremos Firebird, porque es facil de instalar, es libre, y no requiere muchos recursos.

También existe la posibilidad de utilizar otra base de datos (MySQL, Oracle,...) si se notifica antes empezar el curso.

1.4 Requisitos para atender a este curso

No hay requisitos para atender a este curso.

Si el alumno tiene dificultades en un u otro capitulo, el debe sentirse libre de pedir explicaciones adicionales al profesor.

2 Introducción a las bases de datos

2.1 Objetivo del capítulo

Al fin de este capítulo el alumno tendrá una idea de lo que es una base de datos y de sus componentes.

2.2 Que es una base de datos

Una base de datos es uno o varios archivos a donde la información está registrada de forma estructurada, en tablas. Estas tablas contienen registros. Los registros están compuesto de campos bien identificados.

La base de datos más simple es un archivo texto, correspondiendo a una tabla, a donde los campos son delimitados por un carácter (como una coma) o por posición (tamaño fijo).

Existen varios programas profesionales que permiten manejar de bases de datos que contienen varias tablas relacionadas. Existen programas autónomos, que se pueden ejecutar en una misma computadora, como MS Access, y otros que funcionan como cliente-servidor, como MySQL, Oracle, Firebird, DB2, MS SQL Servidor,...

Existen también bases de datos 'embarcadas', que son base de datos incluidas en una aplicación, sin servidor.

2.3 Componentes de una base de datos

Los componente de base de una base de datos son las tabla, que contienen registros (líneas) hechos de campos (columnas). Cada campo es de un tipo definido, y todos los registros de una misma tabla tienen los mismos campos con valores propias.

Las tablas están guardadas en archivos. Dependiendo de la base de datos, un archivo puede contener varias tablas o no, y/o una tabla puede extender sobre varios archivos o no.

Otros componentes que pueden hacer parte de una base de datos, pero que no son presente en todas (y que no vamos a ver en detalle en este curso) , son entre otras las vistas, las funciones, las procedimientos, las secuencias, los disparadores, etc...

Con los servidores corren también uno o varios servicios, dependiendo de la base de datos.

2.4 Cliente/Servidor

Ciertas base de datos, como MS-Access, están diseñadas para funcionar sola, o con pocos clientes. Los archivos de la base de datos pueden ser compartido entre varios clientes, pero cada cliente tiene que manejar los archivos enteros. Si se hace una encuesta sobre una tabla, tienen que leer toda la tabla y luego filtrar, lo que ocasiona mucho trafico de red, y muchos IO en el disco.

Con las bases de datos de tipo servidor, todos los clientes piden los datos al servidor que les regresa solo el resultado de la encuesta. El servidor hace el trabajo en local y devuelve solo la respuesta, lo que es mucho mas eficiente. También se pueden utilizar pooles des conexiones para disminuir la necesidad de abrir y cerrar conexiones, y disminuir el numero de conexiones concurrentes necesarias.

Podemos clasificar las bases de datos en tres categorias:

- Las pequeñas, sin servidor. Para max 10 clientes concurrentes. Ejemplo: archivo texto, MS-Acces,...
- Las medianas, con servidor. Para hasta 50 clientes concurrentes. Ejemplo: Firebird
- Las grandes, con servidor. Para hasta miles de clientes concurrentes. Ejemplo: MySQL, Oracle, SQL Server.

3 Terminología y conceptos

3.1 Objetivo del capítulo

Al fin de este capítulo el alumno tendrá una base de términos y conceptos que se usan con frecuencia en las bases de datos.

3.2 Términos específicos

3.2.1 Tabla

Conjunto de registros que contienen los mismos campos, es decir el mismo tipo de información).

3.2.2 Registro

Conjunto de campos que pertenecen a un mismo dato.

3.2.3 Campo

Detalle de un dato, que es de un tipo específico.

3.2.4 Vista

Una vista es una definición lógica de un parte y/o conjunto de tablas. El objetivo de una vista puede ser varios: esconder las tablas originales, limitar el acceso a los datos (seguridad), simplificar y/o optimizar el extracto de datos, ...

3.2.5 Procedimiento

Un procedimiento es un código compilado que permite ejecutar acciones sobre la base de datos (servidor). El procedimiento puede tener cero o varios parámetros de entrada y/o salida. El objetivo de una vista puede ser varios: esconder las tablas originales, limitar el acceso a los datos (seguridad), simplificar y/o optimizar operaciones sobre los datos, ...

3.2.6 Función

Una función es como un procedimiento si no tiene siempre uno (y solo uno) parámetro de salida.

3.2.7 Paquete

Una paquete es un conjunto de procedimientos y/o funciones.

3.2.8 Secuencia

Las secuencias son contadores que son manejados por el servidor, que les incrementa y se arregla para evitar doble valores. Ciertos servidores como MySQL o SQL Server no tienen secuencias pero tienen una opción de incremento automático de un campo numérico de una tabla.

3.2.9 Disparador

Un disparador es un código compilado en el servidor que se ejecuta cuando se hacen ciertas acciones sobre una tabla (inserción, modificación, ...). Pueden disparar antes o después del cambio. Si dispara antes del cambio permite cambiar el valor se que va agregar/modificar. Si dispara después puede modificar otras tablas.

3.2.10 Esquema

Un esquema es el conjunto de todos los objetos de la base de datos que pertenecen a un mismo usuario.

3.2.11 Sinónimo

Un sinónimo es un nombre que se refiere a un objeto de la base de datos. Esto permite de crear atajos para ciertos objetos de la base de datos.

3.2.12 Indice

Un índice es un pequeño archivo de uno o varios campos ordenados de una tabla que permite aumentar el rendimiento de las encuestas utilizando estos campos como filtro .

3.3 Abreviaciones

3.3.1 DBMS

'Database Management System'. Es un sistema que está desarrollado para manejar una o varias bases de datos, a lo contrario de un simple archivo texto, que no necesita un sistema específico para manejarlo.

3.3.2 RDBMS

'Relational Database Management System'. Sistema de base de datos relacionadas. Es decir una base de datos que contiene varias tablas relacionadas entre ellas, a lo contrario de las bases de datos que contienen una o varias tablas sin relación entre ellas.

3.3.3 SQL

'Simple Query Language'. Lenguaje utilizado para hacer encuestas y acciones sobre base de datos. Existe un lenguaje SQL estándar, pero cada base de datos tiene sus excepciones cuanto a este estándar.

3.3.4 DDL

'Data Definition Language'. Instrucciones SQL que permiten definir y modificar la estructura de los datos.

3.3.5 DML

'Data Modeling Language'. Instrucciones SQL que permiten modificar y visionar los datos.

3.3.6 ODBC

'Open Database Connectivity'. API de Windows que permite conectarse a una base de datos de una manera estándar. El driver ODBC funciona como un puente entre la aplicación llamando y la base de datos.

3.3.7 ADO

Mejoramiento del ODBC, para un acceso mas directo y con mejor eficiencia a las base de datos.

3.3.8 JDBC

'Java Database Connector'. Driver Java que permite conectarse a una base de datos de una manera estándar. El driver JDBC funciona como un puente entre la aplicación llamando y la base de datos.

3.4 Conceptos

3.4.1 OLTP

'Online Transaction Processing'. Sistema de base de datos diseñado para soportar muchas transacciones conjuntas, que modifican los datos (adjunto, modificación, borrado), con fin de soportar la utilización de la aplicación en línea.

3.4.2 DWH

'Data Warehouse'. Sistema de base de datos diseñado para soportar transacciones que requieren agregaciones de datos, con fin de crear reportes. Un DWH es normalmente una copia parcial de los datos de un OLTP, con modificaciones para optimizar las encuestas esperadas.

3.4.3 Replicación

La replicación es la acción de copiar datos de una base de datos a otra. La replicación puede ser parcial o completa, y ejecutarse de manera por incrementa, diferencial o completa.

3.4.4 **Multi-tiers**

Repartición de la carga de una aplicación entre varios sistemas. De costumbre, hay dos o tres partes: cliente / servidor, o cliente / machina de gestión de conexiones / servidor. Cuarta parte podría ser un servidor web.

www.SolucionJava.com

4 Servidores SQL

4.1 Objetivo del capítulo

Al fin de este capítulo el alumno tendrá una vista de las principales bases de datos que existen en el mercado. Como existen miles de bases de datos diferentes, solo vamos a ver algunas. Para mas información sobre estas bases de datos y las otras, puede consultar los sitios Internet de los vendedores o creadores.

4.2 Servidores SQL comerciales

Existen muchos servidores SQL comerciales. Los principales son Oracle, MS SQL Servidor, DB2, y Sybase.

4.2.1 Oracle

Oracle es una la mayor base de datos utilizadas en el mundo.

Los servidores de Oracle son muy poderosos y pueden funcionar sobre varias plataformas, entre otras Windows, Linux, y Unix.

Aquí en Nicaragua se utiliza mucho la versión 8i de Oracle, aún que no es más soportada por Oracle (porque ya tienen dos nuevas versiones). La versión 8i tiene como ventaja sobre las últimas versiones que ya esta muy poderosa, y no pregunta una machina demasiada poderosa para ejecutarse.

Después de la versión 8i, existe la versión 9i, y la versión 10g, que es la versión actual de Oracle.

Oracle es una base de datos que soporta miles de usuarios conjuntos, y archivos de datos muy grande.

Existe una verión gratis de Oracle 10 que es la versión Oracle XE. Tiene algunas limitaciones (max 4GB de datos,...) pero permite desarrollar un sistema pequeños sin costo de licencia. El objetivo de Oracle es que cuando su sistema crece, te quedas con Oracle, pero con la versión pagada.

Sitio web: <http://www.oracle.com>

4.2.2 MS SQL Servidor

El servidor SQL de Microsoft es un competidor para Oracle. Tiene el ventaja que es más fácil de administrar que Oracle, pero solo funciona sobre servidores Windows.

Existe una versión gratis de SQL Server que es la versión SQL Server Express. Tiene algunas limitaciones (max 4GB de datos,...) pero permite desarrollar un sistema pequeños sin costo de licencia. Como Oracle, el objetivo es que cuando su sistema crece, te quedas con Microsoft, pero con la versión pagada.

Sitio web: <http://www.microsoft.com>

4.2.3 Teradata

Teradata es un base de datos que es especialmnete desarrollada para manejar cantidades muy grande de datos (un terabyte=1000 GB).

4.2.4 DB2 y Sybase

DB2 es la base de datos de IBM, y Sybase es un base de datos de una empresa independiente. Funcionan, como Oracle, sobre varias plataformas, pero tienen una parte del mercado mas pequeñas que Oracle y MS SQL servidor. Sitio web: <http://www.ibm.com> y <http://www.sybase.com>

4.3 Servidores SQL libres

Existen muchos servidores SQL gratuitos. Los principales son MySQL, PostGreSQL, y Firebird.

4.3.1 **MySQL**

MySQL es una base de datos libre que ha sido muy utilizada en el Internet. Es un servidor muy rápido y soportando miles de usuarios concurrente, y tamaños de datos muy grande.

Al MySQL le hacia faltas muchas opciones mas avanzadas disponibles en otras bases de datos libre como vistas, procedimientos, etc...

La última versión de MySQL (V. 5) corrige en gran parte estas faltas.

En el curso vamos a utilizar MySQL en su versión 5.

Sitio web: <http://www.mysql.com>

4.3.2 **PostgreSQL**

Como MySQL, PostgreSQL ha sido muy utilizada en el Internet. Tenía como ventaja sobre MySQL las opciones como vistas y procedimientos.

Sitio web: <http://www.postgresql.com>

4.3.3 **Firebird**

Firebird es la versión libre de la base de datos Interbase (versión 6), de Borland. Firebird es una base de datos muy completas y con instrucciones SQL muy parecidas a Oracle.

Firebird, como MySQL y PostgreSQL, funciona bajo Windows, Linux, Unix, y también Mac.

Firebird tiene una emprenta muy pequeña, y existe también sobre la forma de base de datos embarcada.

Si necesitas una base de datos para un sistema pequeño o mediano (≥ 50 conexiones concurrentes), te aconsejo Firebird, porque es muy completa y cumple muy bien con los estándar del SQL. También no pregunta muchos recursos al nivel del PC, lo que les permitirá instalarlo en su machina a la casa, si tienen una, para seguir practicando después de este curso. Para un sistema grande (mas de 50 conexiones concurrentes) te aconsejo MySQL.

Sitio web: <http://www.firebirdsql.com>

4.4 **Herramientas de desarrollo**

Existen muchas herramientas de desarrollo para base de datos. Algunos son gratuitos, otros no.

Para Firebird, les aconsejo el entorno IBManager Lite, que es gratis, pero que solo funciona bajo Windows. Existen otros entornos, que funcionan también bajo Linux. Mas información en www.ibphoenix.com

Para MySQL, vamos a utilizar el entorno desarrollado por MySQL mismo, así como phpMyAdmin, un sitio web de manejo de MySQL escrito en PHP (necesita un servidor con capacidad PHP para usarlo). Más información en www.mysql.com y www.phpmyadmin.net.

4.5 **Instalación de MySQL**

En este curso vamos a utilizar la base de datos MySQL, en su versión 5.0.24a. No vamos a ver su instalación, porque dilatara un poco, pero abajo esta descrito como instalarla bajo SuSE Linux. La base de datos ya esta instalada en su computadora. El CD trae la versión de MySQL para Linux y para Windows.

Para instalar MySQL vamos primero a entrar como el usuario Root (o usar su).

Luego abrimos una ventana de consola, introducemos el CD del curso, y vamos a instalar la version de MySQL que esta en el CD lanzando desde el CD la instrucción:

```
cd /media/BD
```

```
rpm -iv MySQL-server-5.0.24a-0.i386.rpm para instalar el servidor
```

```
rpm -iv MySQL-client-5.0.24a-0.i386.rpm para instalar el cliente
```

Eso installo MySQL bajo /usr/bin.

Vamos a crear una carpeta /mysql conteniendo los atajos hacia programas de MySQL.

```
. createMySQLlinks.sh
```

Vamos ahora a cambiar la clave del usuario root. Para cambiar la clave, entra en /mysql y ejecuta :

```
/usr/bin/mysqladmin -u root password 'SolJava'. La nueva clave sera 'SolJava'.
```

Para verificar que MySQL esta bien instalado y se inicia, ejecuta 'rcmysql restart' como Root.

Y ahora vamos a crear la base de datos del curso:

```
cd /media/BD
/mysql/mysql -u root -pSolJava
create database curso;
exit;
cd /media/BD
/mysql/mysql -u root -pSolJava curso < curso.sql
```

E instalar un entorno de desarrollo (en PHP!) para poder visualizar la base de datos. Se supone que un servidor con capacidad PHP corre y trata las paginas debajo de /workspace

```
cd /
md workspace
cd /media/BD/tools
cp phpMyAdmin.tar.gz /workspace
cd /workspace
tar -xvf phpMyAdmin.tar.gz
```

Y el entorno de desarrollo de MySQL:

```
cd /media/BD/tools
rpm -ivh mysql-gui-tools-5.0r4-1suse10.i586.rpm
rpm -ivh mysql-query-browser-5.0r4-1suse10.i586.rpm
rpm -ivh mysql-workbench-5.0r4-1suse10.i586.rpm
rpm -ivh mysql-administrator-5.0r4-1suse10.i586.rpm
```

www.SolucionJava.com

5 Análisis de requisitos

5.1 Objetivo del capítulo

Al fin de este capítulo el alumno será capaz de entender los requisitos que se tienen que analizar para elegir una base de datos y dibujar su implementación.

5.2 Tipo de base de datos

Para elegir el tipo de base de datos, hay que tomar en cuenta varios parámetros, de los cuales:

- ¿Hay un presupuesto para comprar una base de datos comercial y/o soporte? Mesmo si hay un presupuesto para eso, hay que considerar siempre las oportunidades 'libres'.
- ¿Ya tenemos otras bases de datos administradas en la empresa? ¿Cuál experiencia tienen los desarrolladores y administradores?
- ¿Cuál sistema operativo? ¿Windows u otro?
- Cuantos usuarios tendrán que trabajar juntos en la base de datos
- Tipo de aplicación: ¿OLTP, DWH, o uso general?
- Cuales opciones son deseadas (replicación, interconexión con otros sistemas (web, BD, GUI), objetos binarios, multi-tier, embarcado...)
- ¿Cuantos datos están esperadas?
- ¿Cuántos objetos?
- ¿Existen limitaciones por el producto comprado (paquete de programa cliente como SAP,...)?

La lista no es exhaustiva, y no podemos ver todas las posibilidades para elegir la base de datos más apropiada, pero con la información del capítulo anterior ya podrán tener una pista.

5.3 Implementación de la base de datos

El objetivo de planificar la implementación es de permitir el mantenimiento de la base de datos, como su crecimiento, con costos mínimos.

La implementación dependerá más o menos de los mismos parámetros que los del punto anterior. Al nivel del modelo de datos, se va a ver en el próximo capítulo.

6 Creación del modelo de datos basados en las necesidades

6.1 Objetivo del capítulo

Al fin de este capítulo el alumno tendrá una base para crear un modelo de datos según las necesidades de su aplicación.

6.2 Análisis de requisitos

Antes de empezar a diseñar un modelo de datos, el arquitecto tiene que conocer muy bien los requisitos de la aplicación futura.

Hay que poder identificar cada entidad necesaria, con sus atributos, las relaciones que existen entre estas entidades, y las reglas de integridad necesarias.

Hay también que tomar en cuenta el destino de la aplicación (¿más DWH o OLTP?), y las evoluciones esperadas (¿modelo fijo, o aplicación evolutiva?).

Son más que todos la practica y la experiencia que pueden ayudar a diseñar un modelo de datos adecuado, porque no existe una regla fija por eso.

6.3 Diagramas Entidad Relación

Para diseñar el modelo de datos, utilizaremos un diagrama de entidades relacionadas.

Existen varios tipos de diagramas, con diferente signos para representar las relaciones.

Aquí vamos a utilizar diagramas con la notación IE (Crow's Feet).

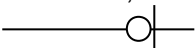
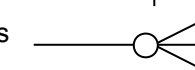
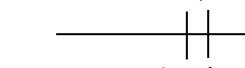
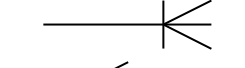
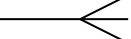
Cada entidad corresponde a una tabla. La entidad puede enseñar diferente niveles de información: de solamente el nombre de la tabla hasta la tabla con todos sus campos, con el tipo de cada campo mencionado y sus restricciones.

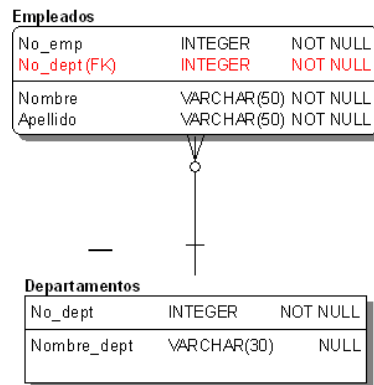
Entre dos entidades pueden existir diferente tipos de relaciones. La relación esta determinada por las posibilidades de conexiones directas entre las dos entidades.

La relación siempre tiene dos partes (de <parte 1> a <parte 2>), de las cuales cada parte es una de las siguientes opciones:

- Cero o uno
- Cero o varios
- Uno
- Uno o varios
- Varios

Para representar las relaciones, cada opción tiene su propio signo:

- Cero o uno 
- Cero o varios 
- Uno 
- Uno o varios 
- Varios 



Ejemplo de relación uno a cero o varios entre las tablas departamentos y empleados.

Existen varios entornos que permiten diseñar diagramas de entidad relación. Ciertos vienen incluidos con entornos de desarrollo, otros son entornos aparte. Ciertos están ligada con un tipo de base de datos, otros soportan varias tipos de base de datos, y otro solo permiten dibujar las tablas, pero no son ligado con ninguna base de datos.

Unos ejemplos de entornos de diseño: Embarcadero ER Studio, Visio, Oracle Designer, Rational Rose,...

Entornos libres: DBDesigner4 (multiples BD), MyWorkbrench (MySQL).

6.4 Normalización del modelo de datos

Existen tres niveles de normalización. La normalización intenta reducir la duplicación de la información y estructurarla en entidades lo más específico posible.

6.4.1 Primer forma normal

Para cumplir con la primer forma normal, una tabla no debe tener dos columnas con la misma información. Columnas que son duplicadas deben ser eliminadas.

Ejemplo:

Sin primer forma normal

No_cliente
Nombre
Apellido
Edad_visita1
Edad_visita2
Fecha_visita1
Fecha_visita2

Con primer forma normal

No_cliente
Nombre
Apellido

No_cliente
No_visita
Edad
Fecha

6.4.2 Segunda forma normal

La segunda forma normal solo se verifica para tablas que tienen una clave primaria de más de un campo. Para cumplir con la segunda forma normal, todos los atributos de una tabla deben de depender de toda la clave primaria. Atributos que no dependen de toda la clave primaria deben ser movido por otra tabla.

Ejemplo:

Sin segunda forma normal

No_cliente
Nombre
Apellido

No_cliente
No_visita
Edad
Fecha
Tipo

Con segunda forma normal

No_cliente
Nombre
Apellido

No_cliente
No_visita
Edad

No_visita
Fecha
Tipo

6.4.3 Tercera forma normal

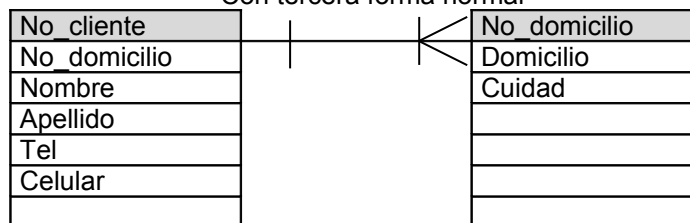
Para cumplir con la tercera forma normal, una tabla no debe contener campos que no son directamente dependiendo de la clave primaria.

Ejemplo:

Sin tercera forma normal

No cliente
Nombre
Apellido
Domicilio
Cuidad
Tel
Celular

Con tercera forma normal



6.5 Integridad referencial

6.5.1 Clave primaria

La clave primaria es un campo o conjunto de campo que identifican de manera única a los registros de una tabla.

Esta clave puede ser utilizada como relación con clave secundaria.

6.5.2 Clave secundaria

Una clave secundaria es un campo de una tabla que se refiere a la clave primaria de otra tabla. El valor de la clave secundaria debe existir como clave primaria de la otra tabla para que sea válido.

6.6 Restricciones de dominio

Además la las claves primaria y segunda, existen otros métodos para restringir los valores que pueden ser utilizados para un cierto campo. Estas posibilidades no existen en todas bases de datos, y depende de la base de datos utilizadas. Aquí siguen varias posibilidades de restringir los valores entradas.

6.6.1 Not null

Un campo definido como 'not null' no permite que el campo no sea nulo.

6.6.2 Check constraints

Son verificaciones que son declarada al nivel del campo, y que restringe los valores validas.

6.6.3 Unique constraints

'Unique constraints' no permite que un mismo campo de dos registros diferentes de la tabla tengan el mismo valor.

6.7 Tipos de datos

6.7.1 Tipos generales

Aún que existen tipos definidos por la nomenclatura SQL-92, cada tipo de base de datos implementa parte de los tipos estándar y también otros tipos propios, que corresponden a un tipo estándar pero con otro nombre, o que es un nuevo tipo.

Básicamente, existen las categorías siguiente de tipos:

- Los booleanos. No presente en todas base de datos.
- Los caracteres. Hay que mencionar la longitud de la cadena máxima. Por ejemplo CHAR(2) o VARCHAR(250).
- Los numéricos. A veces hay que mencionar la longitud (binaria) del tipo. Por ejemplo INTEGER , o DECIMAL(8,3).

- Las fechas. Por ejemplo DATE o TIMESTAMP.
- Los binarios. Por ejemplo BLOB.

Cada tipo ocupe un espacio propio en el archivo de la base de dato. Así no sirve definir un tipo CHAR(8) si el campo solo guardará 3 caracteres máximo.

Consulte la documentación de su base de datos para escoger el tipo más adecuado.

6.7.2 Tipos disponibles en MySQL

6.7.2.1 Principales tipos de caracteres:

Los tipos CHAR y VARCHAR son similares, pero difieren en cómo se almacenan y recuperan.

Los tipos CHAR y VARCHAR se declaran con una longitud que indica el máximo número de caracteres que quiere almacenar. Por ejemplo, CHAR(30) puede almacenar hasta 30 caracteres.

La longitud de una columna CHAR se fija a la longitud que se declara al crear la tabla. La longitud puede ser cualquier valor de 0 a 255. Cuando los valores CHAR se almacenan, se añaden espacios a la derecha hasta la longitud específica. Cuando los valores CHAR se recuperan, estos espacios se borran.

Los valores en columnas VARCHAR son cadenas de caracteres de longitud variable. En MySQL 5.0, la longitud puede especificarse de 0 a 65,535 en 5.0.3 y versiones posteriores.

El tipo TEXT [(M)] es una columna con longitud máxima de 65,535 ($2^{16} - 1$) caracteres.

El tipo ENUM es un objeto de cadenas de caracteres con un valor elegido de una lista de valores permitidos que se enumeran explícita mente en la especificación de columna en tiempo de creación de la tabla.

El tipo SET ('value1' , 'value2' , ...) es un conjunto. Un objeto de cadena de caracteres que puede tener cero o más valores que deben pertenecer a la lista de valores 'value1' , 'value2' , ... Una columna SET puede tener un máximo de 64 miembros. Los valores SET se representan internamente como enteros.

6.7.2.2 Principales tipos numéricos:

Tipo	Bytes	Valor Mínimo	Valor Máximo
		(Con signo/Sin signo)	(Con signo/Sin signo)
TINYINT	1	-128	127
		0	255
SMALLINT	2	-32768	32767
		0	65535
MEDIUMINT	3	-8388608	8388607
		0	16777215
INT	4	-2147483648	2147483647
		0	4294967295
BIGINT	8	-9223372036854775808	9223372036854775807
		0	18446744073709551615

6.7.2.3 Principales tipos de fecha:

Tipo de Columna	"Cero" Valor
DATETIME	'0000-00-00 00:00:00'
DATE	'0000-00-00'
TIMESTAMP	0000000000000000
TIME	'00:00:00'

YEAR	0000
------	------

6.7.2.4 Información completa

Para una información completa sobre los tipos de datos disponible en MySQL, refiérase a la documentación de MySQL disponible en el CD del curso o en línea.

www.SolucionJava.com

7 DDL – Sentencias de definición de datos

7.1 Objetivo del capítulo

Al fin de este capítulo el alumno será capaz de crear, modificar, y borrar tablas de la base de datos. No vamos a ver la creación de otros objetos de la base de datos (vistas, procedimientos, secuencias,...) porque eso sale del cuadro de este curso de iniciación.

Existen muchas opciones para cada tipo de acción (en cual archivo se va a crear, clave primaria, check constraint,...) pero estas opciones dependen de la base de datos utilizadas. Solo vamos a ver las sentencias básicas.

Existen muchos entornos que simplifican la definición de datos.

7.2 Create table

Esta sentencia permite de crear una tabla.

Para crear una tabla hay que mencionar como mínimo el nombre de la tabla y un campo.

```
CREATE TABLE <nombre_de_tabla> (  
<campo_uno> <tipo_campo_uno>,  
<campo_dos> <tipo_campo_dos>,  
...);
```

También se puede crear una tabla sobre la base de una o varias tablas existentes.

```
CREATE TABLE <nombre_de_tabla> AS <SELECT STATEMENT>;
```

7.3 Alter table

Esta sentencia permite de modificar una tabla existente.

Para modificar una tabla hay que mencionar como mínimo el nombre de la tabla, el tipo de modificación a ejecutar, y el campo afectado.

```
ALTER TABLE <nombre_de_tabla> <acción> <campo> <tipo_campo>;
```

Las acciones posibles son 'modify' and 'add'.

Ciertas bases de datos permiten de borrar campos de una tabla, pero no todas.

7.4 Drop table

Esta sentencia permite de borrar una tabla existente. Para borrar una tabla hay que mencionar el nombre de la tabla.

```
DROP TABLE <nombre_de_tabla>;
```

¡ CUIDADIO ! De costumbre, esa acción es inmediata y no recuperable sin una restauración de la BD.

8 DML - Sentencias de manipulación de datos

8.1 Objetivo del capítulo

Al fin de este capítulo el alumno será capaz de hacer encuestas de la base de datos. No vamos a ver todas las opciones, ni las encuestas de otros objetos de la base de datos (vistas, funciones, secuencias,...) porque eso sale del cuadro de este curso de iniciación.

Existen muchas opciones (top, exists, cube,...) para cada tipo de acción, pero estas opciones dependen de la base de datos utilizadas y/o de su versión. Solo vamos a ver las sentencias básicas.

Existen muchos entornos que simplifican las encuestas sobre los datos.

8.2 Insert

La sentencia 'Insert' permite de insertar datos en una tabla.

```
INSERT INTO <nombre_de_tabla> (<campo_1>,<campo_2>,<...>) VALUES  
(<valor_campo_1>,<valor_campo_2>,<valor_...>);
```

También existe:

```
INSERT INTO <nombre_de_tabla> (<campo_1>,<campo_2>,<...>) <SELECT STATEMENT>;
```

8.3 Update

La sentencia 'Update' permite de modificar el valor de uno o varios datos en una tabla.

```
UPDATE <nombre_de_tabla> SET <campo_1>=<valor_campo_1>,<campo_2>=<valor_campo_2>,<...>;
```

De costumbre se limita el cambio a ciertos registros, mencionados utilizando una cláusula WHERE.

```
UPDATE <nombre_de_tabla> SET <campo_1>=<valor_campo_1>,<campo_2>=<valor_campo_2>,<...>  
WHERE <cláusula_where>;
```

8.4 Delete

La sentencia 'Delete' permite de borrar un uno o varios registros en una tabla.

```
DELETE FROM <nombre_de_tabla> ;
```

De costumbre se limita el borrado a ciertos registros, mencionados utilizando una cláusula WHERE.

```
DELETE FROM <nombre_de_tabla> WHERE <cláusula_where>;
```

8.5 Commit y rollback

Si la base de datos permite la gestión de transacciones, se puede utilizar 'Commit' para confirmar una 'Insert', 'Update', o 'Delete', o 'Rollback' para cancelarlos. Ciertas base de datos pueden ser confiuradas para autocommit, que hace un commit automáticamente despues de cada instrucción, a menos que se ha iniciado una transacción de manera explícita (con 'begin transaction xxx;').

Hasta que el 'Commit' está ejecutado, las modificaciones no están inscritas de manera permanente en la base de datos, y sólo son visible para la sesión en curso del usuario autor de las acciones. Después del 'Commit', los cambios son definitivos y visible para todos.

Cuidado que ciertos objetos pueden quedar bloqueados (bloqueando otros usuarios) hasta que el commit sea hecho.

El commit/rollback permite confirmar o de hacer un lote de transacción, para que si una falle, todas las anteriores se anulan también. Cuando se necesita una integridad de transacción, se utiliza en commit/rollback.

Ejemplo:
SELECT emp_no,job_grade FROM employee where emp_no=45;
START TRANSACTION;
update employee set job_grade=5 where emp_no=45;
SELECT emp_no,job_grade FROM employee where emp_no=45;
rollback;
SELECT emp_no,job_grade FROM employee where emp_no=45;
START TRANSACTION;
update employee set job_grade=5 where emp_no=45;
SELECT emp_no,job_grade FROM employee where emp_no=45;
commit;
SELECT emp_no,job_grade FROM employee where emp_no=45;

8.6 Select

El 'Select' permite de seleccionar datos en la base de datos, y visualizarlos.

Se puede utilizar un alias para que el campo se pueda llamar con otro nombre.

```
SELECT <campo_1>,<campo_2>,<...> FROM <nombre_tabla>;  
SELECT <campo_1> as <alias1>,<campo_2>,<...> FROM <nombre_tabla>;
```

Para seleccionar todos los campos de la tabla, se utiliza el asterisco en vez de los nombres de campo.

```
SELECT * FROM <nombre_tabla>;
```

Ejemplo:
SELECT emp_no,job_grade as nivel FROM employee;
SELECT * FROM employee;

8.7 Where

La cláusula 'Where' permite de limitar la encuesta a ciertos datos.

Se utiliza evaluando un campo versus una condición. Se pueden utilizar varias condiciones, con el uso de 'Or', 'And', y/o paréntesis.

Para compara números, se utiliza el signo '=', o '<', o '>', o '<=', o '>=', o 'between ... and ...'.

Para comparar caracteres se utiliza la palabra 'like'. El wildcard es '%'.

Para compara fecha, se utiliza el signo '=', o '<', o '>', o '<=', o '>=', o 'between ... and ...'.

Para

```
SELECT * FROM <nombre_tabla>  
WHERE <campo_1> <operation> <condición> AND <campo_2> <operation> <condición>;
```

Ejemplo:
SELECT emp_no,job_grade FROM employee where emp_no>45;
SELECT emp_no,job_grade FROM employee where emp_no=45 or emp_no=41;
SELECT * FROM employee where emp_no between 40 and 45;
SELECT * FROM employee where last_name like 'P%';

8.8 Count

Para contar un numero de registros, se utiliza la palabra 'Count'.

```
SELECT COUNT(<campo_1>) FROM <nombre_tabla>;
```

Ejemplo:
SELECT count(*) FROM employee where job_grade=4;

8.9 Sum, avg, min, max

Para una suma, min, max,... de un campo, se utilizan la palabras 'Sum', 'Min', 'Max', 'Avg'.

```
SELECT SUM(<campo_1>) FROM <nombre_tabla>;
```

Ejemplo:
SELECT avg(salary) FROM employee where job_grade=2;

8.10 Distinct

Para tener la lista de valores distinguidos de un campo, se utiliza la palabra 'Distinct'.

```
SELECT DISTINCT(<campo_1>) FROM <nombre_tabla>;
```

Ejemplo:

```
SELECT distinct(job_grade) FROM employee;
```

8.11 Order by

Para ordenar los registros regresados, hay que utilizar la palabra 'Order by'.

```
SELECT * FROM <nombre_tabla>
ORDER BY <campo_1>,<...>;
```

Ejemplo:

```
SELECT first_name,last_name FROM employee order by first_name,last_name;
```

8.12 Uniones

Uniones permiten de unir los resultados de dos consultas. Para poder unirlos, tienen que tener los mismos campos.

```
SELECT <campo_1>,<campo_2>,<...> FROM <nombre_tabla_1>
UNION
SELECT <campo_1>,<campo_2>,<...> FROM <nombre_tabla_2>;
```

Ejemplo:

```
select t.first_name,t.last_name from employee t where job_grade=5
union
select t2.fname,t2.lname from usuario t2;
```

8.13 Subconsultas

Subconsultas son consultas sobre otras consultas. La subconsulta se puede utilizar en la cláusula 'From', o en la condición de la cláusula 'Where'. La subconsulta se pone entre paréntesis. En MySQL, las subconsultas deben tener sus propios alias.

```
SELECT t3.<campo_1>, t3.<campo_2> FROM (SELECT t.<campo_1>, t.<campo_2> FROM <nombre_tabla> t <where
clause>) t3
WHERE t3.<campo_1> IN (SELECT t2.<campo_1> FROM <nombre_tabla_2> t2);
```

Ejemplo:

```
SELECT t3.first_name,t3.last_name FROM
(
select t.first_name,t.last_name from employee t where job_grade=5
union
select t2.fname,t2.lname from usuario t2
) t3 where t3.last_name like '%%';
```

```
SELECT t3.first_name,t3.last_name FROM employee t3
where t3.job_country IN
(select t.country from country t where t.currency='Euro');
```

8.14 Agrupaciones

Las agrupaciones permiten agrupar datos y saber cuantos datos hay de cada valor.

```
SELECT <campo_1>,<campo_2>, COUNT(*) FROM <nombre_tabla>
GROUP BY <campo_1>,<campo_2>;
```

Las agrupaciones se pueden filtrar utilizando la cláusula HAVING.

Ejemplo:

```
SELECT job_grade, count(*) FROM employee
where emp_no>45
group by job_grade;
SELECT job_grade, sum(salary) FROM employee
where emp_no>45
group by job_grade
having sum(salary)<1000000;
```

9 Rendimiento

Un problema común en las encuesta a base de datos es el rendimiento.

Las causas de problema de rendimiento son numerosas.

Las más comunes son:

- Instrucción sin o con mala clausula WHERE
- Falta de indice sobre un campo utilizado como filtro
- Mal diseño de la base de datos
- Problema de hardware (falta de memoria, disco ocupado, cpu ocupado por otra aplicación,...)
- Mala configuración del servidor (mal uso de la memoria, disco, cpu,...)
- Mala programación en el cliente. Falta de commit, conexión no cerrada, ...
- Red sobrecargada o muy lenta

Cuando se enfrenta a un problema de rendimiento hay que probar primero de identificar la causa y los síntomas. Servidor sobrecargado en CPU, disco, memoria? Un cliente afectado o todos? Cuando aparece el problema?

Para ayudar a investigar estos problemas existen herramientas. Algunos vienen con la base de datos, otros están desarrollados aparte. Ver la documentación de su base de datos para mas información.

www.SolucionJava.com

10Ejercicios

Utiliza la base de datos 'curso' para MySQL, que se encuentra en el CD para practicar los ejercicios.

Para importarla, de desde la línea de comando en el servidor:

```
mysql -u root curso < curso.sql
```

1. Crea una tabla cliente que debe contener el nombre, apellido, fecha de nacimiento, y numero de cliente.
2. Inserta 3 clientes en la tabla cliente, y enseña el contenido de la tabla.
3. Pone al día el primer cliente, y enseña el contenido de la tabla.
4. Borra el segundo cliente, y enseña el contenido de la tabla.
5. Selecciona los empleados (tabla employee) cuyo first_name empieza con 'Ro'.
6. Selecciona los empleados (tabla employee) cuyo first_name contiene un 'g'.
7. Selecciona los empleados (tabla employee) cuyo job_grade es igual a 2.
8. Selecciona los empleados (tabla employee) cuyo job_grade es entre 3 y 5.
9. Selecciona los empleados (tabla employee) cuyo pais (country) tiene una moneda que contiene un 'r'.
10. Selecciona el salario promedio de los empleados (tabla employee) por job_grade.
11. Cuenta los pacientes (patient) nacidos después del 01/01/2000.
12. Cuenta los pacientes (patient) que tienen más de una prescripción (lab_prescripcion).
13. Cual es el max, min, y el promedio de pruebas por prescripción (mtm_lab_presc2lab_test)
14. Cuenta por año de nacimiento (patient) cuantas prescripciones se han hecho (lab_prescripcion)
15. Cuenta la repartición entre sexo de los pacientes (patient) que tuvieron un examen de hematocrito (lab_test y mtm_lab_presc2lab_test y lab_prescripcion).

www.SolucionJava.com