MySQL

Erick Brito

MySQL

MySQL es un sistema de gestión de bases de datos relacional, multihilo y multiusuario desde enero de 2008 una subsidiaria de Sun Microsystems y ésta a su vez de Oracle Corporation desde abril de 2009 — desarrolla MySQL como software libre en un esquema de licenciamiento dual.

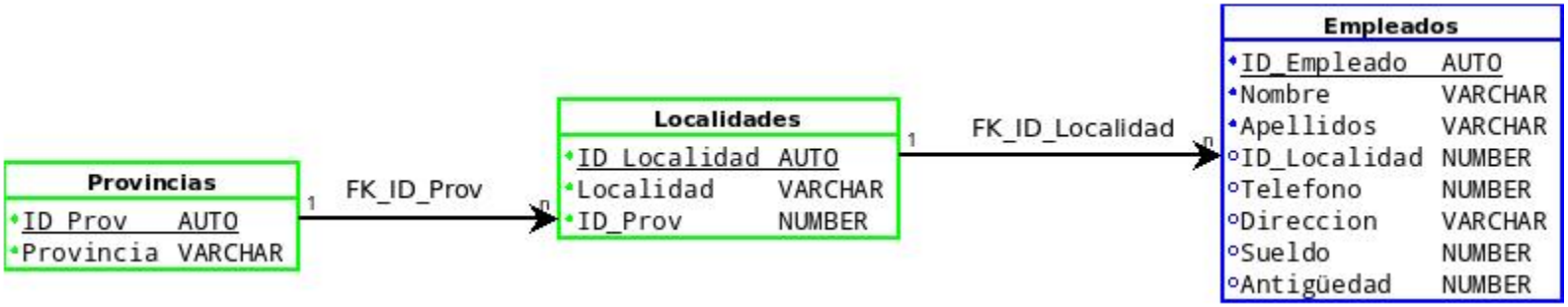
Características MySQL

MySQL es un sistema de administración relacional de bases de datos. Una base de datos relacional archiva datos en tablas separadas en vez de colocar todos los datos en un gran archivo. Esto permite velocidad y flexibilidad. Las tablas están conectadas por relaciones definidas que hacen posible combinar datos de diferentes tablas sobre pedido.

Una Base de Datos Relacional, es una base de datos que cumple con el modelo relacional, el cual es el modelo más utilizado en la actualidad para implementar bases de datos ya planificadas. Permiten establecer interconexiones (relaciones) entre los datos (que están guardados en tablas), y a través de dichas conexiones relacionar los datos de ambas tablas.

Características

- **Una Base de Datos se compone de varias tablas o relaciones.**
- **No pueden existir dos tablas con el mismo nombre ni registro.**
- **Cada tabla es a su vez un conjunto de registros (filas y columnas).**
- **La relación entre una tabla padre y un hijo se lleva a cabo por medio de las claves primarias y ajenas (o foráneas).**
- **Las claves primarias son la clave principal de un registro dentro de una tabla y éstas deben cumplir con la integridad de datos**
- **Las claves ajenas se colocan en la tabla hija, contienen el mismo valor que la clave primaria del registro padre; por medio de éstas se hacen las formas relacionales**



Las relaciones que almacenan datos son llamadas "*relaciones base*" y su implementación es llamada "*tabla*". Otras relaciones no almacenan datos, pero son calculadas al aplicar operaciones relacionales. Estas relaciones son llamadas "*relaciones derivadas*" y su implementación es llamada "*vista" o "consulta*".

Una **clave primaria** es una clave única elegida entre todas las candidatas que define unívocamente a todos los demás atributos de la tabla, para especificar los datos que serán relacionados con las demás tablas. La forma de hacer esto es por medio de claves foráneas.

Una **clave foránea** es una referencia a una clave en otra tabla, determina la relación existente en dos tablas. Las claves foráneas no necesitan ser claves únicas en la tabla donde están y sí a donde están referenciadas.

Relaciones Base y Derivadas

El primer paso para crear una base de datos, es planificar el tipo de información que se quiere almacenar en la misma, teniendo en cuenta dos aspectos: la información disponible y la información que necesitamos.

La planificación de la estructura de la base de datos, en particular de las tablas, es vital para la gestión efectiva de la misma. El diseño de la estructura de una tabla consiste en una descripción de cada uno de los campos que componen el registro y los valores o datos que contendrá cada uno de esos campos.

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

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

Diseño de las bases de datos relacionales

El **lenguaje de consulta estructurado** o **SQL** (por sus siglas en inglés *Structured Query Language*) es un lenguaje declarativo de acceso a bases de datos relacionales que permite especificar diversos tipos de operaciones en ellas.

Lenguaje de definición de datos (DDL): es el que se encarga de la modificación de la estructura de los objetos de la base de datos

```
CREATE TABLE 'CUSTOMERS';
```

```
ALTER TABLE 'ALUMNOS' ADD EDAD INT UNSIGNED;
```

```
DROP TABLE 'NOMBRE_TABLA';.
```

```
TRUNCATE TABLE 'NOMBRE_TABLA'
```

Lenguaje de definición de datos (DDL)

Data Manipulation Language, es un lenguaje proporcionado por el sistema de gestión de base de datos que permite a los usuarios llevar a cabo las tareas de consulta o manipulación de los datos, organizados por el modelo de datos adecuado.

SELECT

La sentencia **SELECT** nos permite consultar los datos almacenados en una tabla de la base de datos.

```
SELECT [ALL | DISTINCT ]
        <nombre_campo> [{,<nombre_campo>}]
FROM <nombre_tabla>|<nombre_vista>
    [{,<nombre_tabla>|<nombre_vista>}]
[WHERE <condicion> [{ AND|OR <condicion>}]]
[GROUP BY <nombre_campo> [{,<nombre_campo >}]]
[HAVING <condicion>[{ AND|OR <condicion>}]]
[ORDER BY <nombre_campo>|<indice_campo> [ASC | DESC]
          [{,<nombre_campo>|<indice_campo> [ASC | DESC ]}]]
```

Lenguaje de manipulación de datos DML

SELECT

Palabra clave que indica que la sentencia de SQL que queremos ejecutar es de selección.

ALL

Indica que queremos seleccionar todos los valores.Es el valor por defecto y no suele especificarse casi nunca.

DISTINCT

Indica que queremos seleccionar sólo los valores distintos.

FROM

Indica la tabla (o tablas) desde la que queremos recuperar los datos. En el caso de que exista más de una tabla se denomina a la consulta "consulta combinada" o "join". En las consultas combinadas es necesario aplicar una condición de combinación a través de una cláusula **WHERE**.

WHERE

Especifica una condición que debe cumplirse para que los datos sean devueltos por la consulta. Admite los operadores lógicos **AND** y **OR**.

GROUP BY

Especifica la agrupación que se da a los datos. Se usa siempre en combinación con funciones agregadas.

HAVING

Especifica una condición que debe cumplirse para que los datos sean devueltos por la consulta. Su funcionamiento es similar al de **WHERE** pero aplicado al conjunto de resultados devueltos por la consulta. Debe aplicarse siempre junto a **GROUP BY** y la condición debe estar referida a los campos contenidos en ella.

ORDER BY

Presenta el resultado ordenado por las columnas indicadas. El orden puede expresarse con **ASC** (orden ascendente) y **DESC** (orden descendente). El valor predeterminado es **ASC.**

```
SELECT matricula,
       marca,
       modelo,
       color,
       numero_kilometros,
       num_plazas
FROM Coches
ORDER BY marca,modelo;

SELECT * FROM Coches ORDER BY marca,modelo;
```

Cláusula WHERE

La cláusula *WHERE* es la instrucción que nos permite filtrar el resultado de una sentencia **SELECT**. Habitualmente no deseamos obtener toda la información existente en la tabla, sino que queremos obtener sólo la información que nos resulte útil es ese momento. La cláusula **WHERE** filtra los datos antes de ser devueltos por la consulta. Cuando en la Cláusula WHERE queremos incluir un tipo texto, debemos incluir el valor entre comillas simples.

```
SELECT matricula,
           marca,
               modelo,
               color,
               numero_kilometros,
               num_plazas
FROM Coches
WHERE matricula = 'MF-234-ZD'
      OR matricula = 'FK-938-ZL' ;
```

Cláusula ORDER BY

La cláusula *ORDER BY* es la instrucción que nos permite especificar el orden en el que serán devueltos los datos. Podemos especificar la ordenación ascendente o descendente a través de las palabras clave **ASC** y **DESC**. El valor predeterminado es ASC si no se especifica al hacer la consulta.

```
SELECT matricula,
    marca,
    modelo,
    color,
    numero_kilometros,
    num_plazas
FROM  coches
ORDER BY marca ASC,modelo DESC;
```

INSERT

Una sentencia *INSERT* de SQL agrega uno o más registros a una (y sólo una) tabla en una base de datos relacional.

```
INSERT INTO agenda_telefonica (nombre, numero)
VALUES ('Roberto Jeldrez', 4886850);
```

UPDATE

Una sentencia *UPDATE* de SQL es utilizada para modificar los valores de un conjunto de registros existentes en una tabla.

```
UPDATE My_table SET field1 = 'updated value asd' WHERE field2 = 'N';
```

DELETE

Una sentencia *DELETE* de SQL borra uno o más registros existentes en una tabla.

```
DELETE FROM tabla WHERE columna1 = 'valor1'
```

INSERT, INSERTAR & UPDATE & DELETE

Fuente:
http://es.wikipedia.org/wiki/MySQL
http://es.wikipedia.org/wiki/SQL

PDF: http://responsive.mx/curso/mysql.pdf

erick@responsive.mx