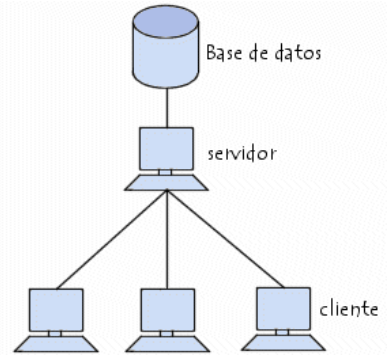
¿Qué es una base de datos?

Una base de datos (cuya abreviatura es BD) es una entidad en la cual se pueden almacenar datos de manera estructurada, con la menor redundancia posible. Diferentes programas y diferentes usuarios deben poder utilizar estos datos. Por lo tanto, el concepto de base de datos generalmente está relacionado con el de red ya que se debe poder compartir esta información. De allí el término base. "Sistema de información" es el término general utilizado para la estructura global que incluye todos los mecanismos para compartir datos que se han instalado.



¿Por qué utilizar una base de datos?
Una base de datos proporciona a los usuarios el acceso a datos, que pueden visualizar, ingresar o actualizar, en concordancia con los derechos de acceso que se les hayan otorgado. Se convierte más útil a medida que la cantidad de datos almacenados crece. Una base de datos puede ser local, es decir que puede utilizarla sólo un usuario en un equipo, o puede ser distribuida, es decir que la información se almacena en equipos remotos y se puede acceder a ella a través de una red.
La principal ventaja de utilizar bases de datos es que múltiples usuarios pueden acceder a ellas al mismo tiempo.

LENGUAJE DE CONSULTA ESTRUCTURADO (SQL)

El SQL es un lenguaje universal que se emplea en cualquier sistema gestor de bases de datos relacional. Tiene un estándar definido, a partir del cual cada sistema gestor ha desarrollado su versión propia.

EL SQL en principio es un lenguaje orientado únicamente a la definición y al acceso a los datos por lo que no se puede considerar como un lenguaje de programación como tal ya que no incluye funcionalidades como son estructuras condicionales, bucles, formateo de la salida, etc. (aunque veremos que esto está evolucionando).

Se puede ejecutar directamente en modo interactivo, pero también se suele emplear embebido en programas escritos en lenguajes de programación convencionales. En estos programas se mezclan las instrucciones del propio lenguaje (denominado anfitrión) con llamadas a procedimientos de acceso a la base de datos que utilizan el SQL como lenguaje de acceso.
Las instrucciones SQL se clasifican según su propósito en tres grupos:

- **El DDL** (Data Description Language) Lenguaje de Descripción de Datos. El DDL, es la parte del SQL dedicada a la definición de la base de datos, consta de sentencias para definir la

estructura de la base de datos, permiten crear la base de datos, crear, modificar o eliminar la estructura de las tablas, crear índices, definir reglas de validación de datos, relaciones entre las tablas, etc. Permite definir gran parte del nivel interno de la base de datos. Por este motivo estas sentencias serán utilizadas normalmente por el administrador de la base de datos

- **El DCL** (Data Control Language) Lenguaje de Control de Datos.
- **El DML** (Data Manipulation Language) Lenguaje de Manipulación de Datos. El DML se compone de las instrucciones para el manejo de los datos, para insertar nuevos datos, modificar datos existentes, para eliminar datos y la más utilizada, para recuperar datos de la base de datos. Veremos que una sola instrucción de recuperación de datos es tan potente que permite recuperar datos de varias tablas a la vez, realizar cálculos sobre estos datos y obtener resúmenes. El DML interactúa con el nivel externo de la base de datos por lo que sus instrucciones son muy parecidas, por no decir casi idénticas, de un sistema a otro, el usuario sólo indica lo que quiere recuperar no cómo se tiene que recuperar, no influye el cómo están almacenados los datos.

I. Componentes del SQL

El lenguaje SQL está compuesto por comandos, cláusulas, operadores y funciones de agregado. Estos elementos se combinan en las instrucciones para crear, actualizar y manipular las bases de datos.

a. Comandos

Existen dos tipos de comandos SQL:

- DLL que permiten crear y definir nuevas bases de datos, campos e índices.
- DML que permiten generar consultas para ordenar, filtrar y extraer datos de la base de datos.

Comandos DLL	
Comando	**Descripción**
CREATE	Utilizado para crear nuevas tablas, campos e índices
DROP	Empleado para eliminar tablas e índices
ALTER	Utilizado para modificar las tablas agregando campos o cambiando la definición de los campos.
Comandos DML	
Comando	**Descripción**
SELECT	Utilizado para consultar registros de la base de datos que satisfagan un criterio determinado
INSERT	Utilizado para cargar lotes de datos en la base de datos en una única operación.
UPDATE	Utilizado para modificar los valores de los campos y registros especificados
DELETE	Utilizado para eliminar registros de una tabla de una base de datos

b. Cláusulas

Las cláusulas son condiciones de modificación utilizadas para definir los datos que desea seleccionar o manipular.

Cláusula	**Descripción**
FROM	Utilizada para especificar la tabla de la cual se van a seleccionar los registros
WHERE	Utilizada para especificar las condiciones que deben reunir los registros que se van a seleccionar
GROUP BY	Utilizada para separar los registros seleccionados en grupos específicos
HAVING	Utilizada para expresar la condición que debe satisfacer cada grupo
ORDER BY	Utilizada para ordenar los registros seleccionados de acuerdo con un orden específico

c. Operadores Lógicos

Operador	**Uso**
AND	Es el "y" lógico. Evalúa dos condiciones y devuelve un valor de verdad sólo si ambas son ciertas.
OR	Es el "o" lógico. Evalúa dos condiciones y devuelve un valor de verdad si alguna de las dos es cierta.
NOT	Negación lógica. Devuelve el valor contrario de la expresión.

d. Operadores de Comparación

Operador	Uso
<	Menor que
>	Mayor que
<>	Distinto de
<=	Menor o igual que
>=	Mayor o igual que
=	Igual que
BETWEEN	Utilizado para especificar un intervalo de valores.
LIKE	Utilizado en la comparación de un modelo
In	Utilizado para especificar registros de una base de datos

e. Funciones de Agregado

Las funciones de agregado se usan dentro de una cláusula SELECT en grupos de registros para devolver un único valor que se aplica a un grupo de registros.

Función	Descripción
AVG	Utilizada para calcular el promedio de los valores de un campo determinado
COUNT	Utilizada para devolver el número de registros de la selección
SUM	Utilizada para devolver la suma de todos los valores de un campo determinado
MAX	Utilizada para devolver el valor más alto de un campo especificado
MIN	Utilizada para devolver el valor más bajo de un campo especificado

II. Orden de ejecución de los comandos

Dada una sentencia SQL de selección que incluye todas las posibles cláusulas, el orden de ejecución de las mismas es el siguiente:

1. Cláusula FROM
2. Cláusula WHERE
3. Cláusula GROUP BY
4. Cláusula HAVING
5. Cláusula SELECT
6. Cláusula ORDER BY

III. Características del lenguaje

Una sentencia SQL es como una **frase** (escrita en **inglés**) con la que decimos **lo que queremos obtener y de donde obtenerlo**. Todas las sentencias empiezan con un **verbo** (palabra reservada que indica la acción a realizar), seguido del resto de **cláusulas**, algunas **obligatorias** y otras **opcionales** que completan la frase. Todas las sentencias siguen una **sintaxis** para que se puedan ejecutar correctamente.

IV. ¿Cómo se interpretaría el diagrama sintáctico de la figura?

Hay que empezar por la palabra **SELECT**, después puedes poner **ALL** o bien **DISTINCT** o nada, a continuación un nombre de columna, o varios separados por comas, a continuación la palabra **FROM** y una expresión-tabla, y por último de forma opcional puedes incluir la cláusula **WHERE** con una condición-de-búsqueda.

Por ejemplo:

```
SELECT ALL col1,col2,col3 FROM mitabla
SELECT col1,col2,col3 FROM mitabla
SELECT DISTINCT col1 FROM mitabla
SELECT col1,col2 FROM mitabla WHERE col2 = 0
```

V. Consultas de Selección

Las consultas de selección se utilizan para indicar al motor de datos que devuelva información de las bases de datos, esta información es devuelta en forma de conjunto de registros que se pueden almacenar en un objeto recordset. Este conjunto de registros es modificable.

5.1 Consultas básicas:
La sintaxis básica de una consulta de selección es la siguiente:

```
SELECT Campos FROM Tabla;
```

En donde campos es la lista de campos que se deseen recuperar y tabla es el origen de los mismos, por ejemplo:

```
SELECT Nombre, Telefono FROM Clientes;
```

Esta consulta devuelve un recordset con el campo nombre y teléfono de la tabla clientes.

Ordenar los registros
Adicionalmente se puede especificar el orden en que se desean recuperar los registros de las tablas mediante la claúsula ORDER BY Lista de Campos. En donde Lista de campos representa los campos a ordenar. Ejemplo:

```
SELECT CodigoPostal, Nombre, Telefono FROM Clientes ORDER BY Nombre;
```

Esta consulta devuelve los campos CodigoPostal, Nombre, Teléfono de la tabla Clientes ordenados por el campo Nombre. Se pueden ordenar los registros por más de un campo, como por ejemplo:

```
SELECT CodigoPostal, Nombre, Telefono FROM Clientes ORDER BY      CodigoPostal, Nombre;
```

Incluso se puede especificar el orden de los registros: ascendente mediante la claúsula (ASC -se toma este valor por defecto) ó descendente (DESC)

```
SELECT CodigoPostal, Nombre, Telefono FROM Clientes ORDER BY  CodigoPostal DESC , Nombre ASC;
```

La cláusula WHERE
La cláusula WHERE puede usarse para determinar qué registros de las tablas enumeradas en la cláusula FROM aparecerán en los resultados de la instrucción SELECT. Si no se emplea esta cláusula, la consulta devolverá todas las filas de la tabla. WHERE es opcional, pero cuando aparece debe ir a continuación de FROM.

```
SELECT Apellidos, Salario FROM Empleados WHERE Salario > 21000;
SELECT Id_Producto, Existencias FROM Productos
WHERE Existencias <= Nuevo_Pedido;
```

5.2 Criterios de Selección

```
SELECT * FROM Empleados WHERE Edad > 25 AND Edad < 50;
SELECT * FROM Empleados WHERE (Edad > 25 AND Edad < 50) OR Sueldo = 100;
SELECT * FROM Empleados WHERE NOT Estado = 'Soltero';
```

```
SELECT * FROM Empleados WHERE (Sueldo > 100 AND Sueldo < 500) OR
(Provincia = 'Madrid' AND Estado = 'Casado');
```

VI. Consultas de Acción

Las consultas de acción son aquellas que no devuelven ningún registro, son las encargadas de acciones como añadir y borrar y modificar registros.

6.1 DELETE

Crea una consulta de eliminación que elimina los registros de una o más de las tablas listadas en la cláusula FROM que satisfagan la cláusula WHERE. Esta consulta elimina los registros completos, no es posible eliminar el contenido de algún campo en concreto. Su sintaxis es:

```
DELETE Tabla.* FROM Tabla WHERE criterio
```

DELETE es especialmente útil cuando se desea eliminar varios registros. En una instrucción DELETE con múltiples tablas, debe incluir el nombre de tabla (Tabla.*). Si especifica más de una tabla desde la que eliminar registros, todas deben ser tablas de muchos a uno. Si desea eliminar todos los registros de una tabla, eliminar la propia tabla es más eficiente que ejecutar una consulta de borrado.

```
DELETE * FROM Empleados WHERE Cargo = 'Vendedor';
```

6.2 INSERT INTO

Agrega un registro en una tabla. Se la conoce como una consulta de datos añadidos. Esta consulta puede ser de dos tipos: Insertar un único registro ó Insertar en una tabla los registros contenidos en otra tabla.

Para insertar un único Registro: En este caso la sintaxis es la siguiente:

```
INSERT INTO Tabla (campo1, campo2, .., campoN)  VALUES (valor1, valor2, ..., valorN)
```

Esta consulta graba en el campo1 el valor1, en el campo2 y valor2 y así sucesivamente.

6.3. UPDATE

Crea una consulta de actualización que cambia los valores de los campos de una tabla especificada basándose en un criterio específico. Su sintaxis es:

```
UPDATE Tabla SET Campo1=Valor1, Campo2=Valor2, ... CampoN=ValorN  WHERE Criterio;
```

UPDATE es especialmente útil cuando se desea cambiar un gran número de registros o cuando éstos se encuentran en múltiples tablas. Puede cambiar varios campos a la vez. El ejemplo siguiente incrementa los valores Cantidad pedidos en un 10 por ciento y los valores Transporte en un 3 por ciento para aquellos que se hayan enviado al Reino Unido.:

```
UPDATE Pedidos SET Pedido = Pedidos * 1.1, Transporte = Transporte * 1.03
WHERE PaisEnvío = 'ES';
```

Ejercicios

EMPLEADO : Tabla

codigo_c	nombre	edad	oficio	dir	fecha_alt	salario	comision	depto_no
281-160483-0005F	Rocha Vargas Hector	27	Vendedor	Leon	12/05/1983	12000	0	40
281-040483-0056P	López Hernandez Julio	27	Analista	Chinandega	14/07/1982	13000	1500	20
081-130678-0004S	Esquivel José	31	Director	Juigalpa	05/06/1981	16700	1200	30
281-160473-0009Q	Delgado Carmen	37	Vendedor	Leon	02/03/1983	13400	0	40
281-160493-0005F	Castillo Montes Luis	17	Vendedor	Masaya	12/08/1982	16309	1000	40
281-240784-0004Y	Esquivel Leonel Alfonso	26	Presidente	Nagarote	12/09/1981	15000	0	30
281-161277-0008R	Perez Luis	32	Empleado	Managua	02/03/1980	16890	0	10

1. Mostrar los nombres de los empleados ordenados alfabéticamente (Z...A)
2. Listar los nombres de los empleados cuyo nombre termine con la letra ‘o’.
3. Listar todos los empleados que viven en la dirección “León”
4. Mostrar todos los empleados que tengan un salario mayor o igual a 13400
5. Mostrar todos los empleados que no tengan comisión
6. Mostrar todos los empleados cuya edad este entre 25 y 30 años
7. Mostrar todos los empleados cuyo oficio sea Analista

Fuente:

http://personal.lobocom.es/claudio/sql001.htm
http://www.aulaclic.es/sql/t_1_1.htm
http://wiki.elhacker.net/bases-de-datos/introduccion
http://proton.ucting.udg.mx/tutorial/sqltut/sql0.html