UNIDAD 3. SQL (DML)

1. Introducción

Recordaremos una serie de conceptos vistos anteriormente:

- Un SGBD (Sistema Gestor de Bases de Datos) es un conjunto de programas, procedimientos, lenguajes, herramientas, etc.., que suministra a los usuarios los medios necesarios para crear y manipular los datos manteniendo la integridad, confidencialidad y seguridad.

- Los SGBD deben ofrecer lenguajes e interfaces para trabajar con datos de forma adecuada al tipo de usuario que tengamos (administrador, operarios, diseñadores…)

Pues bien, estos lenguajes pueden ser:

- DDL → Lenguajes de definición de datos → Son los que permiten definir las tablas, dominios, vistas y …etc.
- DML → Lenguaje de manipulación de datos → Son los que permiten realizar consultas, insertar o borrar datos (registros o tuplas),…etc.
- DCL → Son aquellos lenguajes que permiten gestionar privilegios, autorizaciones y permisos.

El lenguaje más utilizados es el SQL, el cual incorpora funciones de DDL, DML y DCL. Se emplea en la mayoria de los SGBD.

2. SQL (Structured Query Language)

SQL, es un lenguaje de consultas estructuradas que los diferentes motores de bases de datos utilizan para realizar determinadas operaciones sobre los datos o sobre la estructura de los datos.

SQL esta compuesto por comandos, claúsulas, operadores y funciones de agregado:

1. Comandos → Es una palabra reservada que indica la acción que vais a realizar . (Ej: Select → Seleccionar).
2. Claúsulas → Son condiciones utilizadas para definir los datos que deseamos seleccionar o manipular. (Ej: From → De).
3. Operadores → Son elementos que permiten especificar las características de cada una de las condiciones existentes y posibilitan el enlace de todas ellas en SQL. (Ej: (And → Y); (>, <, =)).
4. Funciones de agregado → Se utilizan sobre todo dentro de comando “Select” para devolver un único valor que se aplica a un conjunto de registros o tuplas. (Ej: Mínimo, máximo)

Sentencias o comandos		
DML	Select	Recupera datos de la BD
	Insert	Inserta datos en la BD
	Delete	Borra datos de la BD
	Update	Modifica datos en una BD
DDL	Create Table	Crea una nueva tabla
	Drop table	Borrar una tabla en una BD
	Alter table	Modifica la estructura de una tabla
	Create database	Crea una nueva BD
	Drop database	Elimina una BD
DCL	Grant	Concede privilegios de acceso
	Revoke	Suprime privilegios

Operadores	**Significado**
>	Mayor que
<	Menor que
=	Igual que
>=	Mayor o igual que
<=	Menor o igual que
<>	Distinto de
Between	Entre “se utiliza para especificar un intervalo de valores”
In	En “se emplea para comparar con otra consulta
Like	Como “se utiliza para comparar de forma incompleta”
And	Y es el “y” lógico
Or	O es el “o” Lógico
Not	No, es el “no” Lógico

Funciones	**Significado**
SUM	Calcula la suma de valor de un atributo en un conjunto de registros
AVG	Calcula la media aritmética del valor de un atributo en un conjunto de registros
MAX	Calcula el máximo del valor de un atributo en un conjunto de registros
MIN	Calcula el mínimo del valor de un atributo en un conjunto de registros.
COUNT	Calcula el nº total de registros que cumplen una condición

3. Consulta de SQL

Las consultas más básicas en SQL utilizan "Select" y "From". Estas son las fáciles, pero luego hay muchas más complejas.

- "Select" se utiliza para referirnos al campo.
- "From" lo utilizamos para referimos a las tablas.

Si queremos que nos salgan todos los campos, en vez de poner uno por uno podemos utilizar "*"

Ejemplos:

"Quiero saber el DNI de todos los alumnos":

SELECT dni FROM alumnos;

"Quiero saber el DNI , apellidos y nombre de todos los alumnos":

SELECT *dni, apellidos, nombre* FROM *alumnos*;

Añadimos más claúsulas:

- **WHERE:**

  La cláusula WHERE se utiliza para las condiciones que deben cumplir los campos a mostrar.

  Ejemplos:

  "Quiero saber los apellidos y el nombre de todos los alumnos que han aprobado":

  SELECT apellidos, *nombre* FROM *alumnos* WHERE *nota* >=*5*;

- **ORDER BY:**

  Es para ordenar por un campo de forma ascendente o descendente. Se coloca detrás del "Where", y si no hay "Where" detrás del "From".

  Ejemplos:

  "Quiero saber los apellidos y nombre de todos los alumnos aprobados ordenados por apellidos":

  SELECT *apellidos, nombre* FROM *alumnos* WHERE *nota* >=*5*
  Order by *apellidos*;

- **GROUP BY:**

  Para agrupar las filas o registros de una tabla. Se pone detrás de la claúsula "From" o "Where" y siempre delante del "Order by".

  Ejemplos:

  "Quiero saber los apellidos, nombre, curso de todos los alumnos aprobados agrupados por curso":

  SELECT *apellidos, nombre, curso* FROM *alumnos* WHERE *nota >=5* Group by *curso*;

- **HAVING:**

  Representa las condiciones que debe cumplir la cláusula "Group by" para agrupar. Por lo tanto siempre va con "Group by" y nunca nos lo podremos encontrar solo.

RESUMIENDO la estructura queda:

Select * From * Where * Group by * Having * Order by *;

3.1. Ejemplos de consulta simple con una tabla

Partimos de las siguientes tablas:

Motos (#modelo, potencia, fecha, precio, observaciones) Alumnos (#dni, nombre, apellidos, edad, calificaciones, localidad)

Pues bien:

---

1) Seleccionar todos los campos y registros de una tabla.

a. Crear una consulta que muestre todos los datos de la tabla motos.

b. Muestra la información que poseemos de todos los alumnos almacenados en la tabla alumnos.

---

2) Seleccionar uno o varios campos de una tabla

a. Crea una consulta que muestre el modelo y la potencia de las motos que tenemos.

b. Muestra el nombre de nuestros alumnos por orden alfabético descendente.

---

3) Seleccionar uno o varios campos de una tabla con una condición.

a. Crea una consulta que muestre el modelo y precio de las motos cuyo precio sea mayor que 2000 €.

b. Muestra los apellidos y nombre de los alumnos mayores de edad.

---

4) Seleccionar uno o varios campos con una o varias condiciones.

a. Crea una consulta que muestre el modelo y la potencia de las motos cuya potencia sea superior de 250 y el precio menor de 2000 €.

b. Muestra los apellidos y el nombre de los alumnos que viven en Marbella o Estepona.

* Primero se hacen los and y luego los or. Se pueden colocar paréntesis igual que en matemáticas.

* Las comillas al escribir " a mano" se suelen utilizar las dobles ", pero las que realmente se utilizan el SQL son las simples '.

---

5) Seleccionar uno o varios campos de una tabla con condiciones haciendo uso del operador LIKE.

a. Crea una consulta que muestra el modelo y el precio de aquellas motos que sean modelo ¿? (No se recuerda el nombre del modelo, solo que empieza por F).

b. Mostrar los apellidos, el nombre y la edad de los alumnos que su apellido empiece por "me".

* Los comodines para muchos caracteres y uno solo son: % y _

---

6) Seleccionar uno o varios campos de una o varias tablas haciendo uso del operador Between.

a. Crear una consulta que muestre el modelo de aquellas motos cuyo precio esté entre 20.000 € y 30.000€

b. Muestra los apellidos y nombre de los alumnos de edad comprendida entre los 12 y 16 años.

---

7) Seleccionar uno o varios campos de una tabla haciendo uso del all/distinct

a. Crea una consulta que muestre todas las diferentes potencias de las motos que tenemos.

b. Muestra las diferentes edades de los alumnos que tenemos por orden ascendente.

---

8) Uso del Null

a. Crea una consulta que muestre las motos que no tienen precio.

b. Mostrar los alumnos que no tienen introducida la edad en la B.D.

   * No es posible para comprobar valores NULL con operadores de comparación, tales como =, <, o <>.

---

9) Uso del in.

a. Crear una consulta que muestre todos los datos de las motos cuya potencia sea 125, 250, 300 o 400.

b. Mostrar los alumnos que tengan como calificación 0,1, 5, 9 ó 10.

   * La sintaxis es WHERE column_name IN (value1,value2,...)

---

10) Uso de los alias.

a. Crea una consulta que muestre los modelos pero haciendo uso de alias para la tabla (alias de tablas).

b. Mostrar los apellidos y el nombre de todos mis alumnos pero mostrando como cabecera “sus apellidos” y “su nombre” (Alias de columnas)

Por último decir, que en lugar de seleccionar o mostrar un campo, también se podría introducir funciones. Recordaremos que estamos hablando para lo que colocamos tras el SELECT y antes del FROM.

Ejemplos:

- "Mostrar la máxima nota sacada por lo alumnos"
  Select Max (calificaciones) From alumnos;
- "Mostrar la nota media del los alumnos"
  Select AVG (Calificaciones) From alumnos;

Además de las funciones ya existentes se pueden crear nuevas funciones en base a operaciones matemáticas. Ejemplo:

Select (nota1+nota2+nota3)/3 From alumnos;

3.2. Consultas con varias tablas

Las consultas más complejas son la que hacen uso de dos o más tablas. Para ello:

- Se añaden los nombres de las tablas en la clausula FROM separados por ",".

- Si existieran campos con igual nombre en varias tablas a utilizar, es conveniente hacer uso del nombretabla.nombrecolumna para referenciar a los campos/atributos. Se pueden utilizar "alias" para no escribir todo el nombre de la tabla.
  Ejemplo con el mismo resultado:
  SELECT proveedores.dni,clientes.dni FROM clientes, proveedores …….
  SELECT p.dni,c.dni FROM clientes c, proveedores p …….

- Lo primero que debería aparecer en la clausula WHERE es condición que une ambas tablas por sus campos similares (clave ajena/clave primaria).
  Ejemplo:
  Select a.dni,nombre,tema,nota FROM alumnos a, calificaciones c WHERE a.dni=c.dni;

Pues, vamos a realizar ejemplos prácticos, para ello partiremos de las siguientes tablas:

Empleados(#emp_n, nombre, apellido, oficio, dirección, fecha_alt, salario, comisión, depto_n)
departamentos

Departamentos(#depto_n, dnombre, Loc)

Alumnos (#nif, apenom, direc, pobla, telef)
Asignaturas (#Cod, nombre, horas_semanales)
Notas(#nif, #cod, nota)
alumnos asignatura

Tenemos:

1. Obtener el apellido, oficio, nº empleado, nombre departamento, localidad departamento de todos los empleados.

2. Obtener apellidos, oficio y nombre del departamento de aquellos empleados que pertenezcan a los departamentos 10 o 20.

3. Obtener el nombre de los alumnos junto con sus notas.

4. Obtener el nombre de las asignaturas de cada alumno.

5. Obtener las notas de los alumnos en “FOL”.

6. Obtener el nombre de asignatura junto a su nota para el alumno “Ruiz, José”.

7. Obtener el nombre de alumnos suspensos en “AplicacionesWeb”.

8. Calcular el salario medio de los empleados.

9. Calcular el salario medio de los empleados del departamento “Ventas”.

10. Calcular el nº de registros donde la comisión no sea nula.

11. Calcular el máximo salario que pagamos en Marbella.

12. Calcular el total de gasto de la empresa en salarios.

13. Calcular el nº total de oficios en la empresa.

14. Obtener el nº de empleados en cada departamento para aquellos cuyo número sea mayor de 3 empleados.
    Saber : Las condiciones de un principio parecen ser del where y si en la consulta aparece un group by, hay que ponerlas con un having.

15. Lo mismo pero ordenando por el nº empleados de forma descendente.

16. Obtener el salario máximo que hay en cada departamento. Visualizar nº departamento y salario máximo.

17. Obtener el nº empleados en cada departamento visualizando nº departamento, nombre departamento y nº empleados.

    Nota: Las últimas versiones de MySQL, creo que no te obligan a hacer el group by por todos los campos que te aparezcan en el Select.

18. Obtener la media de los salarios por departamento, visualizando el dnombre y la media para aquellos que su media sea mayor de 1000 €.

3.3. SubConsultas (o sea, consultas dentro de consultas)

Una subconsulta es una sentencia "Select" que se específica dentro de otra sentencia "Select".

Ejemplo:

```
Select * From coches Where modelo In (Select modelo From alquileres);
```

Select * From alumno Where DNI In (Select DNI From matrículas Where asignatura="BBDD");

Nosotros elegimos el que más nos convenga

Podemos poner operadores o predicados:
- Operadores relacionales: =, <, >, <=, >=,…
- Predicados Cuantificador: Any, Some, All.
- Predicados de inclusión: In
- Predicados existenciales: Exists.

¿Cuántos Select se pueden anidar?
Todos los que queramos siempre uno dentro del otro.

Select…(Select…(Select..(Select…(Select…))));
Ejemplo:

"Seleccionar aquellos registros de la tabla coches cuya potencia sea mayor que la media de las potencias de todos los coches"

SELECT * FROM coches WHERE potencia > (SELECT avg(potencia) FROM coches);

Nota: Cuando una subconsultas genera un resultado se podía utilizar operadores relacionales, pero cuando genera varios resultados tendremos que utilizar predicados.

Pues bien, partiremos de las mismas tablas que en el apartado anterior, no obstante recordaremos:

Empleados(#emp_n, nombre, apellido, oficio, dirección, fecha_alt, salario, comisión, depto_n)
departamentos

Departamentos(#depto_n, dnombre, Loc)

1. Obtener apellido, oficio y salario de los empleados que trabajan en el mismo oficio que "Salas".

2. Obtener el nº de empleado de aquellos que cobren más salario que la media de todos.

3. Obtener apellidos y salario de aquellos empleados que tengan el mismo oficio que alguno de los oficios del departamento número 20.

4. Obtener el apellido, salario y departamento de los empleados con el mismo departamento que "Arroyo" y con un salario de los empleados menor que el de "Arroyo".

5. Obtener los datos de los empleados que trabajen en el departamento de Ventas.

6. Obtener los datos de los empleados (Apellidos, oficio, salario, dept_no) que trabajen en los departamentos de "Ventas" o "Contabilidad".

7. Obtener los datos de los empleados (apellido, oficio, salario, dept_no) cuyo salario supera la media de los salarios de todos los empleados.

---

8. Obtener los datos (apellido, oficio y salario) del empleado cuyo apellido sea alfabéticamente el más pequeño. (***varias soluciones, una es:)*

---

9. Obtener los datos de los departamentos que no tengan empleados.

---

10. Obtener los datos de los departamentos que tengan empleados y cuyo salario sea menos que la media de todos.

---

11. Obtener todos los datos de empleados que trabajan en el departamento de contabilidad de Marbella.

---

12. Obtener el apellido, el salario y el nº de departamento de los empleados cuyo salario sea mayor que el máximo salario del departamento 20.

---

13. Igual que la consulta anterior, pero además de visualizar el apellido, el salario y el número de departamento, se ha de visualizar el dnombre.

---

4. Borrar, insertar y actualizar datos en SQL.

Nota → Estas operaciones solo sirven para modificar (manipulación = DML) datos dentro de la tabla, pero no la estructura de la tabla.

4.1 Insertar datos (registros o filas)

La sentencia a utilizar es "INSERT INTO" y permite añadir un registro o fila entera en una tabla. Su estructura es:

Insert Into tabla (Campo1, Campo2…) Values (Valor1, Valor2…);

Los campos de la tabla y que queríamos insertar.
* Si lo dejamos vacío los elige todo

Si falta algún valor podemos poner NULL

- Como mínimo tiene que estar la clave.
- Valores Alfanuméricos entre comillas. (Simples)
- Fechas entre 'x/x/x'
- Si un campo no tiene valor determinado, se pone null

Ejemplo:

"Insertar los datos de un nuevo coche en la tabla "Coches", la cual tiene la estructura (modelo, potencia, precio, fecha, observaciones)"

Ficha : Citroen C4 Picasso
Color Azul
Potencia 140 CV
Puertas 5
Precio =27.000 €

INSERT INTO COCHES (modelo, potencia, precio,fecha, observaciones) VALUES ('C4Picasso', 140, 27000, '31/01/2012',null);

Ejemplo:

"Tenemos una tabla llamada "actores" con los siguientes campos (Cod_actor, nombre, fecha_nac, lugar_nac, nacionalidad, fecha_muerte). Queremos introducir los datos del siguiente actor: Brad Pitt, nacido el 18/12/1963 en Oklahoma."

INSERT INTO ACTORES (Cod_actor, nombre, fecha_nac, lugar_nac, nacionalidad, fecha_muerte) VALUES (25, 'BRAD PITT', '18/12/1968', 'Oklahoma', 'Estados Unidos', null);

También podemos insertar filas o registros procedentes de una tabla (diferente o la misma a la que vallamos a meter). Para ello necesitaremos esta orden orden (cambiando el Values y los valores por un select completo).

"INSERT INTO_____ SELECT_____"

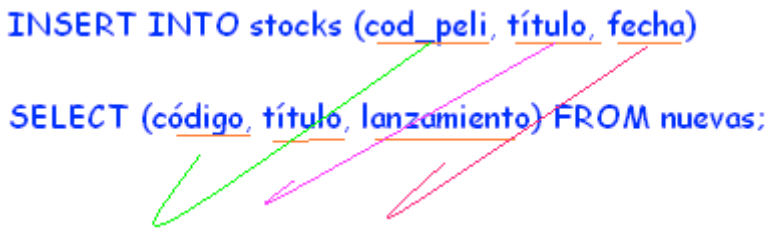
Ejemplo:

"Pasar las películas nuevas (tabla NUEVAS) a la tabla STOCKS."

INSERT INTO stocks SELECT * FROM nuevas;

Nota: Siempre tiene que ser del mismo tipo. No tiene por que llamarse igual, lo que cuenta es que están con el mismo tipo y el mismo orden.

Ejemplo:



4.2. Modificar datos de una tabla

La sentencia en "UPDATE" y nos permite modificar los valores de las tablas. Su estructura es:

UPDATE tabla SET Condición actualización WHERE condición de las filas a actualizar;

Ejemplos:

"Actualizar los datos del precio de los coches ya que hemos de sumar nuestra comisión de 500 €."

UPDATE coches SET precio=precio+500;

"Actualizar el precio en 60€ para aquellos coches con más de 140 CV."

UPDATE coches SET precio=precio+60 WHERE potencia>140;

4.3 Borrar datos en una tabla

La sentencia para esta acción es "DELETE" y permite borrar filas o registros de una tabla. La estructura es:

DELETE * FROM tabla WHERE condición;

No tiene por que ponerse

Ejemplos:

"Borra todos los datos de la tabla coches"

DELETE * FROM coches;

"Borra todas las piezas del modelo Mercedes"

DELETE * FROM COCHES WHERE modelo='Mercedes';

*Nota: En algunos SGBD no hace falta poner el **

Ejercicios sobre INSERT, UPDATE y DELETE:

1. Insertar un empleado que tendría los siguientes datos: nº de empleado 1000, apellido QUEVEDO, oficio ANALISTA, su dirección “MAR, 75”, salario 3000 €, la comisión es de 0, fecha de alta es la fecha actual y el departamento es el 40.

---

2. Insertar un empleado con los siguientes datos: nº de empleado 1001, apellido “RODRIGUEZ”, salario 3000 €, oficio ANALISTA y departamento 40.

---

3. Insertar un nuevo empleado del que solo tenemos los datos: Empleado 1002, apellido “PEREZ”.

---

4. Supongamos que tenemos una tabla de preinscripción de trabajo llamada “Emple30” con los mismos campos que la tabla empleados. Insertar todos los empleados de esa tabla a la tabla empleados asignándolos inicialmente al departamento 30.

---

5. Insertar un nuevo empleado con nº 1004 y apellido “LOPEZ” en el departamento que trabajó el empleado apellidado “CASAS” y con su mismo salario.

---

6. Cambiar el salario a la mitad y la comisión a 0 para aquellos empleados que pertenezcan al departamento de producción.

---

7. Igualar el salario y oficio de “FERNÁNDEZ” al salario y oficio de “ARROYO”.

   ****** hacer antes los select, anotar valores y ejecutar update, o sea, realizo el 1º select, luego el 2º select y con los valores obtenidos hago el update. ASUNTO : NO SE PUEDE METER SELECT EN LA PARTE DE SET.***

---

8. Borrar de la tabla Empleados los del departamento 30.

---

9. Borrar el contenido de la tabla EMPLE30.

RESUMIENDO:

Hemos visto 'DML' (Lenguaje de manipulación de datos) y dentro de esto, las siguientes comandos:

- Select → Para seleccionar
- Insertar Into → Para insertar
- Update → Para modificar
- Delete -> Para eliminar