COMANDOS SQL

Crear una tabla

- Vamos a crear una tabla llamada empleado en alguna base de datos.

- CREATE TABLE empleado (
- nombre VARCHAR(20),
- apellido VARCHAR(20),
- sexo CHAR(1),
- edad INTEGER,
-)

Insertar datos

- INSERT INTO empleado (nombre, apellido, sexo, edad)
- VALUES ('Javier', 'Ramírez', 'M', 23,);

Seleccionar

- Obtener toda la tabla empleado
- **SELECT * FROM empleado;**

- Obtener algunas columnas y todos los renglones de la tabla empleado
- **SELECT apellido, sueldo, ingreso FROM empleado;**

Seleccionar renglones específicos

- En SELECT se usa la cláusula WHERE, donde se especifican los renglones que se quieren obtener.
- Ejemplo: el nombre y apellido de los empleados cuya edad sea mayor ó igual a 25 años.
- **SELECT nombre, apellido FROM empleados WHERE edad >=25;**

- O bien, los que su apellido sea Ramírez

- **SELECT nombre, apellido FROM empleado**
- **WHERE apellido = 'Ramírez';**

- Como se trata de columnas tipo caracter, se usan apóstrofes y debe escribirse exactamente como esté en la base de datos.

Removiendo datos con DELETE

- Con DELETE podemos mover uno ó bien todos los renglones de una tabla, por ejemplo **DELETE FROM empleado;** eliminaría todos los datos de la tabla empleado.

- Si utilizamos la cláusula WHERE se eliminan los renglones que cumplan la condición, por ejemplo:

DELETE FROM empleado WHERE apellido = 'Ramirez';

Modificando datos con UPDATE

- En una base de datos además de insertar y eliminar datos hay que actualizar.
- Ejemplo: la edad del empleado Javier Pérez es de 35 años.
- `UPDATE empleado SET edad = 35 WHERE apellido = 'Pérez';`

- La cláusula WHERE controla los renglones en los que se llevará a cabo la modificación. Si no se pone esta cláusula, se cambiarían la edad de todos los empleados a 35.

Ordenando datos con ORDER BY

- Cuando hacemos un SELECT, los renglones se despliegan con un orden no determinado. Si se quieren obtener los renglones en un orden específico, es necesario aumentar la cláusula ORDER BY al final del SELECT.

```
SELECT * FROM empleado ORDER BY apellido;
```

- Para invertir el orden se usa DESC

```
SELECT * FROM empleado ORDER BY edad DESC;
```

Se puede ordenar por distintas columnas, si en la primera hay dos valores iguales, se usa la siguiente para ver cual va primero.

Destruyendo tablas con DROP

- Cuando queremos eliminar completamente la tabla escribimos:

```
DROP TABLE empleado;
```

- Si queremos eliminar todos los datos de una tabla pero conservar su estructura utilizamos (no ejecutarla):

```
DELETE FROM empleado;
```

Caracteres especiales

- Si el empleado que vamos a insertar tiene un apóstrofe en el nombre, por ejemplo:

```
INSERT INTO empleado (nombre, apellido,, sexo, edad,)
VALUES ('Jack', 'O'Donnell', 'M', 23, ');
```

- Me marca un error:

Para resolverlo:

'O''Donnell',se ponen dos apóstrofes ó

'O\'Donnell' se pone una diagonal invertida

Etiquetado de columnas y Comentarios

- El nombre que aparece en la salida de un SELECT hasta arriba es la etiqueta de la columna que corresponde al nombre del atributo. Se puede cambiar usando la palabra clave AS, por ejempo:

```
SELECT nombre AS Razón_Social FROM cuenta;
SELECT 1 + 3 AS total;
```

- Con dos guiones -- indica que hasta el final del renglón es un comentario
- Con /* al principio y con */ al final indica que todo lo que está contenido es un comentario

```
/*Esto es un comentario y no ejecuta  nada*/;
```

Uso de AND/OR

- AND y OR se usan para conectar condiciones simples.

Insertaremos más datos en empleado:

```
INSERT INTO empleado (nombre, apellido, sexo, edad)
VALUES ('María', 'Pérez, 'F', 23, ');
```

- AND se usa para combinar las dos comparaciones que conecta, ejemplo:

```
SELECT * FROM empleado WHERE nombre='María' AND
  apellido='Pérez';
```

- OR verifica que se cumpla una de las dos comparaciones al menos, ejemplo:

```
SELECT * FROM empleado WHERE sexo = 'F' OR edad = 24;
```

- Si se combinan ANDs y ORs en un mismo query es mejor agrupar los ANDs y los ORs usando paréntesis. Los ANDs se evalúan primero. Por ejemplo, si se quieren los de apellido Pérez y que sean hombres ó mujeres si no usamos paréntesis el resultado es incorrecto, ya que se evalúa el AND primero:

```
SELECT * FROM empleado WHERE apellido='Pérez' AND sexo='M' OR sexo='F';
```

Me da los de apellido Pérez Y son de sexo M pero también todos los de sexo F´, ya que primero se ejecuta el AND y después el OR.

- Lo correcto es:

```
SELECT * FROM empleado WHERE apellido='Pérez' AND (sexo='M' OR sexo='F');
```

Este me da los que son de apellido Pérez y que son de sexo M ó F.

Between y Like

- Si queremos aquéllos empleados que están entre 24 y 30 años de edad:

```
SELECT * FROM empleado WHERE edad >= 24 AND edad <= 30;
SELECT * FROM empleado WHERE edad BETWEEN 24 AND  30;
```

- Para obtener los datos de los empleados cuyo apellido inicia con la letra R:

```
SELECT * FROM empleado WHERE apellido LIKE 'R%';
```

- Obtener los que ingresaron en 2008:

```
SELECT * FROM empleado WHERE ingreso LIKE '2008%';
```

Comparaciones con LIKE

Empieza con una D	LIKE ‘D%’
Contiene una D	LIKE ‘%D%’
Tiene D en 2da posición	LIKE ‘_D%’
Empieza con D y tiene una e	LIKE ‘D%e%’
No empieza con D	NOT LIKE ‘D%’

AGREGACIÓN

- A veces es necesario resumir cierta información. En lugar de ver renglones, solo se necesita saber cuántos son. Para esto se usan las siguientes palabras clave:

COUNT(*) cuenta renglones
SUM(nombre_col) total
MAX(nombre_col) máximo
MIN(nombre_col) mínimo
AVG(nombre_col) promedio

Ejemplos

- Contar número de renglones

```
SELECT COUNT(*) FROM empleado;
```

- Obtener la suma de los sueldos

```
SELECT SUM(sueldo) FROM empleado;
```

- Obtener el sueldo máximo

```
SELECT MAX(sueldo) FROM empleado;
```

- Obtener la edad mínima

```
SELECT MIN(edad) FROM empleado;
```

- Obtener el sueldo promedio

```
SELECT AVG(sueldo) FROM empleado;
```

GROUP BY

- En los ejemplos anteriores nos regresó un renglón como resultado y sólo se usó una columna con el agregado.
- Usando los agregados con GROUP BY se tendrá la aplicación del agregado en una columna, en los renglones agrupados por otra columna.

SELECT COUNT(*) FROM empleado, regresa el número de renglones en la tabla.

- Si queremos contar cuántos son sexo M y cuantos F:

SELECT sexo, COUNT(*) FROM empleado GROUP BY sexo;

- Si por sexo se quiere saber datos del salario y de la edad:

SELECT sexo,MIN(edad),MAX(edad),AVG(sueldo) FROM empleado GROUP BY sexo ORDER BY 4 DESC;

HAVING

- Permite probar condiciones en los valores agregados. A menudo se usa con GROUP BY. Con HAVING se pueden incluir o excluir grupos basados en el valor de agregación para ese grupo.

En el ejemplo de contar los empleados por sexo, podemos limitar para que despliegue solo aquéllos que sean más de 3, el query que cuenta empleados por sexo es:

```
SELECT sexo, COUNT(*) FROM empleado GROUP BY sexo;
```

- Sólo los que sean más de 3 empleados de ese sexo:

```
SELECT sexo, COUNT(*) FROM empleado GROUP BY sexo HAVING COUNT(*)>3;
```