LENGUAJE DE MANIPULACIÓN DE DATOS (DML) Consultas SQL

- UT5 -

INTRODUCCIÓN

Dentro del lenguaje SQL existen varios tipos de sentencias (SELECT, INSERT, UPDATE, CREATE,..) las cuales podemos dividir en 2 grupos: Las **sentencias DDL** (Data Definition Language) y **DML** (Data Manipulation Language). Como añadido nos encontramos con el **DCL** (Data Control Language) que gestiona los privilegios de los usuarios respecto a la base de datos.

DDL

Las sentencias **DDL** son sentencias que nos permiten definir, alterar, borrar objetos dentro de la base de datos. Las sentencias más importantes son:

CREATE: Permite crear objetos dentro de la base de datos y la propia base de datos Estos objetos son: Procedimientos almacenados, tablas, base de datos, triggers (disparadores), funciones, vistas, índices, etc.

ALTER TABLE: Permite **modificar** la estructura de una tabla (añadir (ADD) o eliminar (DROP) columnas y sus tipos de datos, restricciones, etc)

DROP: La sentencia DROP me permite **eliminar** objetos de la base de datos (podríamos decir que es la operación contraria a CREATE).

TRUNCATE: Elimina todos los registros de una tabla.

COMMENT: Nos permite agregar comentarios al diccionario de datos de la BD.

RENAME

DML

Permiten **manipular**, consultar y/o eliminar los **datos** o registros de las tablas. Las sentencias del DML son:

SELECT: Permite recuperar información de una o más tablas en la base de datos, dependiendo de las condiciones que especifiquemos.

INSERT: Esta sentencia nos ayuda a insertar o ingresar filas de datos en las tablas de nuestra base de datos.

UPDATE: Actualiza o modifica los registros de una tabla dependiendo de la condición que especifiquemos.

DELETE: Elimina todos los registros de una tabla.

DCL

Permite crear roles, permisos e integridad referencial, así como el control al acceso a la base de datos.

Las sentencias del DCL son:

GRANT: Usado para otorgar privilegios de acceso de usuario a la base de datos.

REVOKE: Utilizado para retirar privilegios de acceso otorgados con el comando GRANT.

SENTENCIAS E INSTRUCCIONES PARA LA CONSULTA DE DATOS

SELECT

INSERT

UPDATE

DELETE

SELECT

SELECT
[ALL | DISTINCT | DISTINCTROW | TOP n [PERCENT]]
Campo1, campo2, max, min, sum ...
FROM tabla/s
[WHERE *expression*]
[GROUP BY {col_name | expr | position} [ASC | DESC]],
[HAVING expression]
[ORDER BY {col_name | expr | position} [ASC | DESC]];

Podríamos leer la sentencia como: seleccionar el campo o los campos ... de las tablas ... que cumplan la condición "expression", agrupados por ... en orden ascendente/descendente; y ordenados según ...

SELECT

- SELECT puede operar sobre una o varias tablas y sobre uno o varios campos.

- Cuando queremos establecer una condición usando registros individuales, se suele usar WHERE.

- Cuando se quiere establecer una condición sobre un grupo de registros normalmente emplearemos HAVING, además HAVING suele ir acompañado de la cláusula GROUP BY (se desarrollará este concepto más adelante).

- Las expresiones y operadores que podemos emplear para establecer las condiciones de selección, son las empleadas en otros lenguajes de programación (>, <, =, <>, AND, OR, NOT, BETWEEN, IN, LIKE, SOME/ANY)

SELECT

- Las cláusulas FROM, WHERE, GROUP BY, HAVING se evalúan en el orden indicado en la sintaxis. El resultado de esta evaluación es una tabla virtual que se utiliza en la evaluación posterior.

- Después de evaluar la cláusula SELECT se evalúa ORDER BY.

- Si se hace referencia a dos campos con el mismo nombre en distintas tablas, se precede el nombre del campo por el nombre de la tabla y un punto.

Ejemplo: SELECT Cliente.Nombre, Empleado.Nombre
FROM Cliente, Departamento ...

- El predicado NULL se utiliza con la palabra clave IS
SELECT Nombre FROM Clientes WHERE salario IS NULL

SELECT

- Es muy importante tener en cuenta que cuando utilizamos la cláusula GROUP BY, los únicos campos que podemos incluir en el SELECT sin que estén dentro de una función de agregación, son los que vayan especificados en el GROUP BY

SELECT

Operador	Operación	Ejemplo
=	Igualdad	SELECT * FROM EMPLEADO WHERE NDEPTO = 100;
!= , <> , ^=	Desigualdad	SELECT * FROM EMPLEADO WHERE NDEPTO != 100;
<	Menor que	SELECT * FROM EMPLEADO WHERE NDEPTO < 200;
>	Mayor Que	SELECT * FROM EMPLEADO WHERE NDEPTO > 200;
<=	Menor o igual que	SELECT * FROM EMPLEADO WHERE NDEPTO <= 100;
>=	Mayor o igual que	SELECT * FROM EMPLEADO WHERE NDEPTO >= 100;
IN	Igual a cualquiera de los miembros entre paréntesis	SELECT * FROM EMPLEADO WHERE NDEPTO in(100,300);
NOT IN	Distinto a cualquiera de los miembros entre paréntesis	SELECT * FROM EMPLEADO WHERE NDEPTO NOT IN(200);
BETWEEN	Contenido en el Rango	SELECT * FROM EMPLEADO WHERE NDEPTO BETWEEN 100 AND 300;
NOT BETWEEN	Fuera del Rango	SELECT * FROM EMPLEADO WHERE NDEPTO NOT BETWEEN 100 AND 300;
LIKE '_abc%'	Contiene la cadena 'abc' a parir del segundo carácter y luego cualquier cadena de caracteres	SELECT * FROM EMPLEADO WHERE nombre LIKE 'Ma%'
IS NULL	Compara el valor de un campo con el valor "NULL"	SELECT * FROM EMPLEADO WHERE sueldo IS NULL;
IS NOT NULL	Compara el valor de un campo sea diferente de "NULL"	SELECT * FROM EMPLEADO WHERE fnac IS NOT NULL;

SELECT

Operador	Operación	Ejemplo
+	Suma	SELECT nombre,sueldo + comision FROM empleado WHERE oficio = 'VENDEDOR';
-	Resta	SELECT nombre, sueldo - descuento FROM empleado WHERE oficio = 'VENDEDOR';
*	Producto	SELECT nombre, sueldo *12 FROM empleado;
/	División	SELECT nombre, sueldo/31 FROM empleado;

Para ayudarnos a construir la consulta nos debemos hacer 3 preguntas básicas:

¿Qué datos nos piden?

¿Dónde están los datos?

¿Qué requisitos deben cumplir los registros?

FUNCIONES AGREGADAS

El **lenguaje SQL** tiene **funciones incorporadas para hacer cálculos sobre los datos**. Las funciones se pueden dividir en dos grupos (existen muchas más, que dependen del **sistema de bases de datos** que se utilice):

FUNCIONES AGREGADAS: Devuelven un sólo valor, calculado con los valores de una columna.

AVG() - La **media de los valores**

COUNT() - El **número de filas**

MAX() - El **valor más grande**

MIN() - El **valor más pequeño**

SUM() - La **suma de los valores**

FUNCIONES ESCALARES

FUNCIONES ESCALARES : Devuelven un sólo valor basándose en el valor de entrada.

UCASE() - Convierte un campo a **mayúsculas**

LCASE() - Convierte un campo a **minúsculas**

MID() - **Extrae** caracteres de un campo de texto

LEN() - Devuelve la **longitud** de un campo de texto

NOW() - Devuelve la **hora y fecha** actuales del sistema

FORMAT() - Da formato a un **formato** para mostrarlo

Ejemplos sencillos: https://diego.com.es/principales-funciones-en-sql