TEMA 4. LENGUAJE DE CONSULTA ESTRUCTURADO (SQL).

TEMA 5. SGBD: SQL Server, MySQL, Oracle

1. QUÉ ES SQL.

- SQL (Structured Query Language) es un lenguaje que permite expresar operaciones diversas sobre datos almacenados en bases de datos relacionales.
- SQL es un lenguaje de cuarta generación.
- Fue desarrollado por IBM y la versión original se denominaba SEQUEL.
- ANSI-SQL es el nombre que damos a una estandarización de las diversas implementaciones que de la evolución del lenguaje original inventado por IBM se han ido desarrollando posteriormente.

2. CÓMO SE USA.

Las peticiones sobre los datos se expresan en SQL mediante *sentencias*, que deben escribirse de acuerdo con las reglas sintácticas y semánticas de este lenguaje.

- **SQL INTERACTIVO**: Las sentencias pueden escribirse directamente en la pantalla de un terminal interactivo, en el cual también se recibe el resultado.
- **SQL INCORPORADO Y DINÁMICO:** Esto quiere decir que se pueden incorporar instrucciones de SQL en lenguajes de programación como: C++, C, Java, Cobol, Pascal y Fortran.

3. PARA QUÉ SIRVE

Permite:

1. Principalmente a los Programadores y a los Usuarios:
   - Insertar, modificar, borrar y consultar los datos contenidos en las tablas de la Base de Datos. (Lenguaje de Manipulación de Datos: LMD)
2. Realizar tareas propias del Administrador de la Base de Datos, como son:
   - Definición, modificación y destrucción de objetos de la Base de Datos, gestión de las autorizaciones de acceso, etc. (Lenguaje de Definición de Datos: LDD y Lenguaje de Control de Datos: LCD)

4. TIPOS DE SENTENCIAS.

4.1 LENGUAJE DE MANIPULACIÓN DE DATOS (LMD)

Las sentencias de esta parte de SQL permiten realizar consultas y mantenimiento de datos. Son las siguientes:

SELECT, INSERT, UPDATE y DELETE

Veamos la sintaxis de estas sentencias:

(Ver "Convenciones de sintaxis Transact-SQL" en enlace Referencia de Transact-SQL)

SENTENCIA SELECT

Permite realizar consultas sobre una o varias tablas de la base de datos. La sintaxis completa de la instrucción SELECT es compleja, aunque las cláusulas principales se pueden resumir del modo siguiente:

SELECT [DISTINCT] **expresión** [,expresiones,…]
[FROM tabla1 [, tabla2,...] [JOIN…]]
[WHERE **predicados**]
[GROUP BY columna/s]
[HAVING condición]
[ORDER BY columna/s] [DESC];

Si tenemos una sentencia SELECT formada por las cláusulas anteriores, el resultado de ejecutarla se puede obtener mediante los pasos siguientes:

1. Ejecutar la cláusula FROM. Lo obtenido de momento es la tabla resultante de la sentencia.
2. Ejecutar la cláusula WHERE. Se eliminan todas las filas que no satisfagan el predicado.
3. Ejecutar la cláusula GROUP BY. Formar grupos con las filas de la tabla resultante en el paso anterior que tengan iguales valores en las columnas de agrupamiento.
4. Ejecutar la cláusula HAVING. Descartar los grupos que no satisfagan la condición especificada.
5. Ejecutar la cláusula SELECT. Esto implica evaluar sus expresiones para cada grupo, produciendo por cada uno de ellos una fila de la tabla

resultante final, con tantos valores como expresiones. Si se ha especificado DISTINCT se eliminan las filas repetidas.

6. Ejecutar ORDER BY. Es decir presentar la tabla resultante clasificada por las columnas especificadas.

Consideraciones:

1. Una **expresión** es una combinación de operandos, operadores aritméticos y paréntesis (no todos ellos obligatorios) que cuando el SGBD la ejecuta produce un único resultado. Los operandos pueden ser nombres de columnas, constantes, *funciones* u otras expresiones. Como expresión también podemos usa *.

2. Las funciones pueden ser colectivas o escalares.

- Son ***funciones colectivas o de agregado***: SUM, MAX, MIN, AVG, COUNT. El resultado se obtiene a partir de una colección de valores. (Representan un único valor que se obtiene aplicando una *operación a un conjunto de valores).* (Un resultado por tabla o por grupo)
- *Son* ***funciones*** **escalares**: LEN, SUBSTRING, UPPER, LOWER. El resultado se obtiene a partir de un único valor. (Un resultado por fila). Al menos debéis saber localizar las de cadenas, matemáticas y las de fecha-hora.

  (Ver funciones en SQL Tutorial 1 y en enlace Referencia de Transact-SQL)

  (Ver "Operadores" en enlace Referencia de Transact-SQL)

  (Ver "Ejemplos de tipos de expresiones.SQL". El archivo pertenece a EJEMPLOS1)

3. Cuando en la **cláusula FROM** se especifica una sola tabla se buscan los valores en ella. Cuando se pone más de una tabla separadas por coma se realiza el producto cartesiano de estas y sobre ese resultado se buscan los datos. También puede aparecer la **cláusula JOIN**.

(Ver "Ejemplos FROM1.SQL" El archivo pertenece a EJEMPLOS1)

4. Un **predicado** expresa una condición entre valores, y según estos pueden resultar Verdadero o Falso. Los predicados pueden ser simples o compuestos.

PREDICADOS SIMPLES

Predicados Básicos

Expresan condiciones de comparación entre dos valores. Tienen esta forma:

"ELEMENTO DE COMPARACIÓN_ 1" Operador_Relacional "ELEMENTO DE COMPARCIÓN_2"

Operador_ Relacional puede ser: =, <>, >, <, >=, <=

Los "ELEMENTOS DE COMPACIÓN" pueden ser expresiones. El "ELEMENTO DE COMPARCIÓN_2" puede ser a su vez el resultado de una sentencia SELECT, que debe ir entre paréntesis y producir un único valor. A las sentencias SELECT incluidas dentro de otra sentencia SELECT se le llaman ***sentencias subordinadas o subselects***.

(Ver "Ejemplos predicados básicos.SQL". El archivo pertenece a EJEMPLOS1)

Predicado NULL

Su formato es: nombre_columna IS [NOT] NULL

(Ver "Ejemplos predicado NULL.SQL". El archivo pertenece a EJEMPLOS1)

Predicados Cuantificados

Cuando se usa una sentencia subordinada en un predicado de comparación, su resultado debe ser un valor único, sin embargo se admite que el resultado tenga varios valores si la sentencia subordinada va precedida de algunas de las palabras cuantificadoras: ANY o ALL

Si se usa ALL . El predicado cuantificado es verdadero si la comparación es verdadera para todos y cada uno de los valores resultantes en la sentencia subordinada. **(Ver "predicado ALL.SQL". El archivo pertenece a EJEMPLOS2)**

Si se usa ANY . El predicado cuantificado es verdadero si la comparación es verdadera para alguno de los valores resultantes en la sentencia subordinada. **(Ver "predicado ANY.SQL". El archivo pertenece a EJEMPLOS2)**

Predicado BETWEEN

Su formato es: Expresion1 [NOT] BETWEEN Expresion2 AND Expresion3

Sirve para determinar si un valor está comprendido entre otros dos. **(Ver "predicado BETWEEN.SQL". El archivo pertenece a EJEMPLOS2)**

Predicado LIKE

Su formato es: Nombre_columna [NOT] LIKE constante alfanumérica

Sirve para buscar combinaciones de caracteres, se pueden usar comodines % o _ . **(Ver "predicado LIKE.SQL". El archivo pertenece a EJEMPLOS2)**

Predicado IN

Su formato es: Expresión [NOT] IN (constante1, constante2, ...)

Sirve para preguntar si el resultado de una expresión esta incluido en esa lista. En vez de una lista podemos poner una sentencia SELECT subordinada. Tendrá esta forma:

Expresion IN (Sub select) equivale a Expresion = ANY (Subselect)

(Ver "predicado IN.SQL". El archivo pertenece a EJEMPLOS2)

Predicado EXISTS

Su formato es: EXISTS (Subselect)

Es verdadero si el resultado de la sentencia subordinada contiene una o más filas.

(Ver "predicado EXISTS.SQL". El archivo pertenece a EJEMPLOS2)

PREDICADOS COMPUESTOS

Son combinaciones de otros predicados simples o compuestos, con los operadores lógicos AND, OR o NOT.

5. La Cláusula GROUP BY indica que se han de agrupar filas de la tabla, de modo que todas las que tengan iguales valores en las columnas de agrupamiento formen un grupo.

6. La Cláusula HAVING sirve para descartar grupos de filas.

(Ver "Ejemplos GROUP BY y HAVING.SQL". El archivo pertenece a EJEMPLOS3)
(Ver "Ejemplos FROM2.SQL". El archivo pertenece a EJEMPLOS3)

7. La cláusula ORDER BY presenta la tabla resultante final clasificada por las columnas indicadas.

SENTENCIA INSERT

Permite añadir una o más filas completas a una tabla.

Formato 1:

```
INSERT [INTO] tabla [(col1,col2, ...)]
VALUES (valor1, valor2,...);
```

Formato 2:

```
INSERT [INTO] tabla [(col1,col2, ...)]
Subselect;
```

Ejemplo:

```
INSERT INTO Tabla1 SELECT * FROM Tabla2
```

Tabla1 y Tabla2 tienen la misma estructura.

SENTENCIA DELETE

Permite borrar una o más filas completas de una tabla.

Formato:

```
DELETE [FROM] tabla
        [WHERE predicado];
```

SENTENCIA UPDATE

Permite modificar o actualizar varias filas de la tabla. Permite actualizar o modificar algunas columnas de las filas de una tabla.

Formato:

UPDATE tabla

SET col1 = expresion1 [,col2 = expresion2]

[WHERE predicado];

Recuerda que predicado puede contener una subconsulta.

(Ver "Ejemplos INSERT-DELETE-UPDATE.SQL". El archivo pertenece a EJEMPLOS4)

(Ver "Ejemplos INSERT-DELETE-UPDATE con subconsultas.SQL". El archivo pertenece a EJEMPLOS4)

(Ver "Ejemplos SELECT INTO-INSERT INTO tabla SELECT y TRUNCATE.SQL". El archivo pertenece a EJEMPLOS4)