**SINTAXIS:**

SELECT es la sentencia que tiene mayores variaciones y opciones sintácticas en todos los SGBD del mercado y también la más compleja y potente de todas las instrucciones SQL.

Utilizaremos esta sentencia para consultar los datos de una base de datos. En la sentencia SELECT el usuario especifica lo que quiere obtener, no cómo, ni dónde.

Como hemos dicho presenta muchas opciones. Las más comunes son:

**SELECT**

**[ALL | DISTINCT | DISTINCTROW ]**

**{Expresión | Columna1, ... | \* }**

**FROM TablasReferenciadas**

**[WHERE Condiciones]**

**[GROUP BY {NombreColumna | Expresión | Posición} [ASC | DESC], …]**

**[HAVING Condiciones]**

**[ORDER BY { NombreColumna | Expresión | Posición } [ASC | DESC] , ...]**

**[LIMIT { [desplazamiento,] contador | contador OFFSET desplazamiento} ]**

Antes de empezar a analizar la función de cada una de las cláusulas de esta sentencia vernos algunas consideraciones generales:

Para realizar una consulta a una o varias tablas con esta sentencia es imprescindible utilizar al menos las palabras:

**SELECT** seguido del nombre de la columna o columnas que se quieran mostrar.

**FROM** seguido del nombre de la tabla o tablas de las que se obtienen los datos.

El resto de las cláusulas son opcionales, pero en el caso de que se incluyan debe respetarse el orden descrito.

La sentencia **SELECT** no se limita a nombres de columnas de tablas o a un \*, pueden ser otras expresiones, funciones, etc.; incluso aunque no correspondan a ninguna tabla.

**Ejemplo**: **SELECT 7\*4, SIN(3.1414/2), CURRENT\_DATE();**

También es válido, y además necesario cuando se seleccionan varias columnas con el mismo nombre en diferentes tablas, especificar el nombre de la columna separado con un punto del nombre de la tabla, de la siguiente forma:

NombreTabla.NombreColumna

**Ejemplo: SELECT CLIENTES.Nombre, CLIENTES.Apellidos FROM CLIENTES;**

Se pueden consultar datos de tablas de una base de datos que no está actualmente en uso estableciendo el nombre de la base de datos a la que pertenece separado con un punto del nombre de la tabla:

NombreBaseDeDatos.NombreTabla

**Ejemplo: SELECT \* FROM TalleresFaber.CLIENTES;**

**----------------------------------------------------------------------------------------------------------------------------**

**EJERCICIO:**

Queremos consultar la referencia, descripción e importe de las 10 actuaciones más baratas que se pueden llevar a cabo en nuestro taller.

**SELECT Referencia, Descripcion, Importe FROM ACTUACIONES ORDER BY Importe LIMIT 10;**

**------------------------------------------------------------------------------------------------------------------------------**

**Operadores:**

Los operadores se utilizan para construir expresiones que se usan en cláusulas WHERE, ORDER BY y HAVING y además se pueden emplear directamente en las sentencias.

Los operadores sirven para obtener nuevos datos a partir de los ya existentes.

**Ejemplo: SELECT Apellidos, Nombre, Dirección FROM EMPLEADOS WHERE CP LIKE ‘39%’;**

Hemos consultado los datos apellidos, nombre y dirección de los empleados de Talleres Faber que viven en Cantabria. Se establece como condición que los dos primeros caracteres del código postal sean 39.

**PRIORIDAD DE OPERADORES:**

**\*buscar en la web.**

**Operador de asignación y Operadores aritméticos:**

**Operador de asignación:**

**Con SET.**

Ejemplo: SET @hoy= CURRENT\_DATE();

SELECT @hoy;

(Veremos la fecha actual)

Utilizando el operador de asignación :=

Ejemplo: SELECT @X:=10;

SELECT @X;

(Veremos el valor almacenado 10)

Una variable sin asignar será de tipo cadena y su valor será NULL.

**Operadores aritméticos:**

|  |  |
| --- | --- |
| **Operador** | **Significado** |
| + | Se utiliza para sumar dos números y, como operador unario, para simbolizar signo positivo de un número. |
| - | Se utiliza para hallar la diferencia entre dos números y, como operador unario, para simbolizar signo negativo de un número. |
| \* | Se utiliza para multiplicar dos números. |
| / | Se utiliza para dividir dos números. |
| ^ | Se utiliza para elevar un número a la potencia del exponente (número ^ exponente). |
| \ | Se utiliza para dividir dos números y el resultado cociente en forma de entero (división entera) entero. |
| % o **mod** | Dividen dos números y devuelven el resto entero de la división. |
| - | Se obtiene un valor de signo contrario. |

**Los operadores de mayor prevalencia son los aritméticos, y entre ellos, el orden de prevalencia es:**

1. Exponenciación.
2. Operadores de signo.
3. Multiplicación y división.
4. Suma y resta.

**------------------------------------------------------------------------------------------------------------------------------**

**Operadores de comparación.**

|  |  |
| --- | --- |
| **Operador** | **Significado** |
| < = | Menor o igual. |
| < | Menor. |
| > | Mayor. |
| > = | Mayor o igual |
| = | Compara dos expresiones y devuelve 1, sin son iguales y 0, si son diferentes. |
| < => | Funciona como =, salvo que si una de las dos expresiones o las dos es nula, el resultado no es NULL. Si se comparan dos expresiones nulas, el resultado es verdadero (1). |
| < > **ó** != | Si las expresiones comparadas son diferentes, el resultado es verdadero, y si son iguales, el resultado es falso. |

Estos operadores tienen todos ellos la misma prioridad.

**Comparación de cadenas de caracteres y de fechas:**

Los operadores relacionales pueden usarse para comparar cadenas.

* Cuando se **comparan cadenas**, se considera menor la cadena que aparezca antes por orden alfabético no distinguiendo entre mayúsculas y minúsculas.
* Las **cadenas constantes** se han de expresar encerradas entre comillas dobles o simples.
* Si son fechas, se considera menor cuánto más antigua sea.

**Operadores lógicos.**

|  |  |
| --- | --- |
| **Operador** | **Función** |
| **AND** ó**&&** | Devuelve el valor TRUE cuando las dos condiciones son verdaderas y FALSE si alguna de ellas es falsa. |
| **OR** ó || | Devuelve el valor TRUE si cualquiera de las dos condiciones es verdadera y FALSE cuando las dos condiciones son falsas |
| **NOT** ó **!** | Devuelve lo opuesto a la condición que sigue a NOT. Si el operador es NULL devuelve NULL. |
| **XOR** | Devuelve NULL si cualquiera de los operadores es NULL. Cuando uno de los operadores es verdadero, devuelve TRUE y si ambos son verdaderos o falsos devuelve FALSE. |

--------------------------------------------------------------------------------------------------------------------

Ejercicio resuelto:

Consultar las matrículas de los vehículos que hayan entrado a reparar en 2011 con menos de 100000 Km y que hayan salido del taller sin reparar:

**SELECT IdReparacion, Matricula FROM REPARACIONES WHERE (FechaEntrada>=’2011-01-01’ AND km<100000) AND Reparado=False;**