Módulo 2

Código: 484

Bases de datos

Técnico Superior en Desarrollo de Aplicaciones Multiplataforma

3. Contenidos

CONTENIDOS CONCEPTUALES	CONTENIDOS PROCEDIMENTALES
SELECT … FROM … WHERE DISTINCT ORDER BY GROUP BY HAVING COUNT, SUM, AVG, MIN, MAX ALL Y ANY (INNER, NATURAL, OUTER) JOIN GRANT REVOKE	Realizar consultas, con distinto grado de dificultad, sobre la base de datos utilizando el Lenguaje de Consulta de datos DQL de SQL: • Consultas simples. • Agrupación. • Unión. • Ordenación. • Múltiples files y operadores. Crear vistas, usuarios y asignarles privilegios sobre la base de datos utilizando el Lenguaje de Control de datos DCL de SQL.

Sesiones 1 a 4. Contenidos teóricos

DATA QUERY LANGUAGE

DQL es la abreviatura del Data Query Language (lenguaje de consulta de datos) de SQL.
El único comando que pertenece a este lenguaje es el versátil comando SELECT. Este comando permite:

- Proyección: Obtener datos de ciertas columnas de una tabla.
- Selección: Obtener registros (filas) de una tabla de acuerdo con ciertos criterios.
- Asociación (JOIN): Mezclar datos de tablas diferentes
- Realizar cálculos sobre los datos.
- Agrupar datos.

Sesiones 1 a 4. Contenidos teóricos

DATA QUERY LANGUAGE

SINTAXIS SENCILLA DEL COMANDO SELECT

SELECT * | {[DISTINCT] columna | expresión [[AS] alias], ...} FROM tabla

Donde:
* El asterisco significa que se seleccionan todas las columnas
DISTINCT: Hace que no se muestren los valores duplicados.
Columna: Es el nombre de una columna de la tabla que se desea mostrar.
Expresión. Una expresión válida SQL
Alias: nombre que se le da a la cabecera de la columna en el resultado de esta instrucción.

Sesiones 1 a 4. Contenidos teóricos

DATA QUERY LANGUAGE

Ejemplo: Mostrar toda la información almacenada en la tabla cliente.

```
SELECT * FROM cliente;
```

Ejemplo: Mostrar el código y nombre de todos los registros almacenados en la tabla vendedor.

```
SELECT codigo, nombre FROM vendedor;
```

Sesiones 1 a 4. Contenidos teóricos

DATA QUERY LANGUAGE

Ejemplo: Mostrar el código de ticket, número de línea e importe de cada línea de ticket. El importe se calcula multiplicando la cantidad por el precio.

```
SELECT ticket, nlinea, cant*precio FROM linea_ticket;
```

Sesiones 1 a 4. Contenidos teóricos

CLAUSULA DISTINCT

Mediante la cláusula DISCTINT evitamos que se muestren valores repetidos.

Ejemplo: Mostrar los distintos tipos de iva aplicados en las ticket.

```
SELECT DISTINCT iva FROM ticket;
```

Sesiones 1 a 4. Contenidos teóricos

OPERADORES ARITMÉTICOS

Los operadores + (suma), - (resta), * (multiplicación) y / (división), se pueden utilizar para hacer cálculos en las consultas.

Ejemplo: Mostrar el código de articulo y el doble del precio de cada artículo.

```
SELECT codigo, precio * 2 FROM articulo;
```

Sesiones 1 a 4. Contenidos teóricos

ALIAS DE LAS COLUMNAS

Los ALIAS permiten cambiar el nombre asignado a una columna o expresión en el resultado de una sentencia select. Se define con la palabra AS ("como").

A continuación se muestra una sentencia que utiliza un alias para el importe de las líneas de ticket

```
SELECT ticket, nlinea, cant * precio AS importe FROM linea_ticket;
```

Sesiones 1 a 4. Contenidos teóricos

CLAUSULA WHERE

La clausula WHERE se utiliza para condicionar la información que formará parte de la salida de la consulta. Tras ella se sitúa el requisito que han de cumplir todos los registros de salida. Así, los que no la cumplan no aparecerán en el resultado.

Ejemplo: Mostrar el código y nombre de aquellos clientes cuyo código es menor que 5

```
SELECT codigo, nombre FROM cliente WHERE codigo < 5;
```

Sesiones 1 a 4. Contenidos teóricos

OPERADORES LÓGICOS Y DE COMPARACIÓN

En la cláusula WHERE, como hemos visto en el ejemplo anterior, se pueden utilizar operadores lógicos. Estos operadores sirven tanto para comparar números como para comparar textos y fechas.

Operador	Significado	Operador	Significado
>	Mayor que	AND	Devuelve verdadero si las expresiones a su izquierda y derecha son ambas verdaderas.
<	Menor que		
>=	Mayor o igual que	OR	Devuelve verdadero si cualquiera de las dos expresiones a la izquierda y derecha es verdadera.
<=	Menor o igual que		
=	Igual	NOT	Invierte la lógica de la expresión que está a su derecha. Si era verdadera pasa a ser falso y viceversa.
<> ó !=	Distinto		

Sesiones 1 a 4. Contenidos teóricos

DATA QUERY LANGUAGE

Ejemplo: Mostrar los distintos ivas aplicados en los tickets de aquellos vendedores cuyo código es mayor o igual a 5.

```
SELECT DISTINCT iva FROM ticket WHERE vendedor >= 5;
```

Ejemplo: Mostrar el código y descripción de aquellos artículos cuyo stock iguala o supera las 10 unidades

```
SELECT codigo, descripcion FROM articulo WHERE stock >=10;
```

Ejemplo: Mostrar el código y fecha de los tickets del vendedor con código 1 a las que se le ha aplicado un 10% de iva.

```
SELECT codigo, fecha FROM ticket WHERE iva=10 and vendedor=1;
```

Sesiones 1 a 4. Contenidos teóricos

DATA QUERY LANGUAGE

Ejemplo: Mostrar el código y fecha de los tickets del vendedor con código 1 a las que se le ha aplicado un 10% de iva o con descuento del 10%

```
SELECT codigo, fecha FROM ticket WHERE vendedor=1 AND (iva=10 OR dto=10);
```

Ejemplo: Mostrar el número de línea, el ticket al que pertenecen y el precio total (sin considerar descuentos ni impuestos) de aquellas líneas de ticket cuyo importe supera los 15 euros.

```
SELECT nlinea, ticket, cant*precio AS total FROM linea_ticket WHERE cant*precio > 15.0;
```

Sesiones 1 a 4. Contenidos teóricos

OPERADOR BETWEEN

Permite obtener datos que se encuentren en un rango.

Ejemplo: Consultar el código y nombre de aquellos vendedores cuyos códigos se encuentran comprendidos entre 3 y 7.

```
SELECT codigo, nombre FROM vendedor WHERE codigo BETWEEN 3 and 7;
```

Ejemplo: Obtener toda la información correspondiente a los artículos cuyo stock no se encuentre en el rango (stock mínimo, stock mínimo +10).

```
SELECT * FROM articulos WHERE stock NOT BETWEEN stock_min AND stock_min + 10 ;
```

Sesiones 1 a 4. Contenidos teóricos

OPERADOR IN

El operador IN posibilita obtener registros de los cuales uno de los datos está en una lista de valores.

Ejemplo: Escribir una expresión que devuelva el código y nombre de los municipios pertenecientes a las provincias con códigos 'VA' y 'AL'.

```
SELECT codigo, nombre FROM municipio WHERE provincia IN ( 'VA' , 'AL' ) ;
```

Sesiones 1 a 4. Contenidos teóricos

OPERADOR LIKE

Permite obtener registros cuyo valor en un campo, cumpla una condición textual. LIKE utiliza una cadena que puede contener estos símbolos.

SÍMBOLO	SIGNIFICADO
%	Una serie cualquiera de caracteres
_	Un carácter cualquiera

Ejemplo: Mostrar el nombre de todos los municipios cuyo nombre comience por la letra "V".

```
SELECT nombre FROM municipio WHERE nombre LIKE 'V%';
```

Sesiones 1 a 4. Contenidos teóricos

OPERADOR IS NULL

El operador IS NULL devuelve verdadero si el valor que examina es nulo. También se puede usar la expresión IS NOT NULL que devuelve verdadero en el caso contrario, cuando la expresión no es nula.

Ejemplo: Escribir una sentencia que muestre los datos de aquellos tickets sin código de cliente o sin código de vendedor.

```
SELECT codigo, fecha FROM ticket WHERE (cliente IS NULL) OR (vendedor IS NULL);
```

Ejemplo: Escribir una sentencia que muestre la información correspondiente a aquellos vendedores que no tienen jefe.

```
SELECT * FROM vendedor WHERE jefe IS NULL;
```

Sesiones 1 a 4. Contenidos teóricos

PRECEDENCIA DE OPERADORES	
A veces las expresiones que se producen en los SELECT son muy extensas y es difícil saber quéparte de la expresión se evalúa primero. A continuación se una tabla con la precedencia. A pesar de conocerla, es aconsejable utilizar paréntesis para ordenar las operaciones y evitar errores.	
Orden de precedencia	Operador
1	* (multiplicar) / (dividir)
2	+ (suma) – (resta)
3	|| (concatenación)
4	Comparaciones (>, <, !=, …)
5	IS [NOT] NULL, [NOT] LIKE, IN
6	NOT
7	AND
8	OR

Sesiones 1 a 4. Contenidos teóricos

ORDENACIÓN

Los registros obtenidos al ejecutar una sentencia SELECT se pueden ordenar mediante ORDER BY y las palabras ASC O DESC (por defecto se toma ASC)

Ejemplo: Mostrar los datos de todas las provincias ordenadas por su nombre de forma ascendente.

```
SELECT * FROM provincia ORDER BY nombre ASC;
```

Ejemplo: Mostrar los datos de todos los municipios ordenados de forma ascendente por su código de provincia y descendente por su nombre de municipio.

```
SELECT * FROM municipio ORDER BY provincia ASC, nombre DESC;
```

Sesiones 1 a 4. Contenidos teóricos

EJECUCIÓN DE SENTENCIAS

Cuando el SGBD ejecuta una sentencia sigue siempre el mismo método:

- Examina la cláusula FROM para localizar las tablas con las que se va a trabajar
- Si existe la cláusula WHERE, lleva a cabo la extracción de aquellas filas que cumplan la condición o condiciones de esta cláusula.
- A partir del contenido de la cláusula SELECT resuelve qué información se desea mostrar de las filas previamente seleccionadas
- Si existe el modificador DISTINCT, elimina las filas sobrantes.