Consultas SQL 1

IT BOARDING

BOOTCAMP

Índice

IT BOARDING
BOOTCAMP

Índice

// Select

IT BOARDING

BOOTCAMP

Select

Las consultas de selección se utilizan para obtener información de la base de datos en forma de registros.

```
SELECT Campos FROM Tabla;
```

lista de campos que deseo obtener, se separan con coma

EJEMPLO + SINTAXIS

```
SELECT id
      ,amount
FROM mov_cuenta_corriente;
```

```
SELECT *
FROM cuenta_corriente;
```

WHERE

Se utiliza para **filtrar** nuestra consulta dependiendo la necesidad. Los campos del SELECT no necesariamente tienen que estar en el filtro, es decir, puedo seleccionar campos y en el filtro colocar otros campos que no estén en la selección.

```
SELECT Campos FROM Tabla
WHERE campo1=valor;
```

Filtro que el campo1 sea igual a 2, entonces me voy a traer todos los registros que cumplan esa condición

EJEMPLO + SINTAXIS

```
SELECT id
      ,amount
FROM mov_cuenta_corriente
WHERE id=1;
```

ORDER BY

Nos permite ordenar los resultados a partir de 1 o más campos.

```
SELECT campo1, campo2, campo3
FROM Tabla
ORDER BY campo2;
```

Voy a ordenar por el campo2 (por default el ordenamiento es ascendente)

Desordenado

Tabla
3
1
2
5
4

Ordenado

Tabla
1
2
3
4
5

Variantes

Tabla
1
2
3
4
5

ASC (menor a mayor)

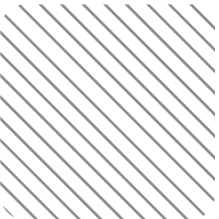
Tabla
5
4
3
2
1

DESC (mayor a menor)

ASCENDENTE o DESCENDENTE

Puedo ordenar en forma ascendente algunos campos y otros campos en forma descendente dependiendo la necesidad que tengamos en cómo mostrar u obtener los datos.

```
SELECT campo1, campo2, campo3
FROM Tabla
ORDER BY campo2 ASC, campo3 DESC;
```



EJEMPLO + SINTAXIS

```
SELECT id
      ,amount
FROM DEBTMOVEMENT
ORDER BY id DESC;
```

nombre del campo

=

```
SELECT id
      ,amount
FROM DEBTMOVEMENT
ORDER BY 1 DESC;
```

posición del campo

AND y OR

- El operador **AND** mostrará los resultados cuando se cumplan **las 2 condiciones**.

  condición1 AND condición2

- El operador **OR** mostrará los resultados cuando se cumpla **alguna de las 2 condiciones**.

  condicion1 OR condicion2

```
SELECT campo1, campo2, campo3 FROM Tabla
WHERE campo1 = 1
AND campo2 = 'TEST';
```

EJEMPLO + SINTAXIS

```
SELECT *
FROM DebtMovement
WHERE debt_id = 33729488
AND type = 'reverted';
```

LIKE y NOT LIKE

- El operador **LIKE** se utiliza en una cláusula **WHERE** para buscar un **patrón específico en una columna**, para **negar** podemos utilizar el **NOT LIKE**.

```
SELECT campo1, campo2, campo3
FROM Tabla
WHERE campo1 = 1
  AND campo2 LIKE 'T%';
```

BETWEEN

Nos permite seleccionar un **rango**, es utilizada par establecer un **criterio** de **selección** dentro de la cláusula **WHERE** y nos permite establecer los valores que deben tener los **extremos (INCLUSIVOS)** de ese **rango deseado**.

```
SELECT campo1, campo2, campo3
FROM Tabla
WHERE campo1 = 1
  AND campo2 BETWEEN '20200101' AND '20200131';
```

```
SELECT campo1, campo2, campo3
FROM Tabla
WHERE campo1 = 1
AND campo1 >= '20200101'
AND campos <= '20200131';
```

Si anteponemos **NOT** delante del **BETWEEN** devolverá aquellas valores **no incluidos en el intervalo**.

LIMIT Y OFFSET

- Estos operadores nos permiten **limitar** la **cantidad** de **registros** de una **consulta**.
- “**LIMIT N**” es la palabra clave y **N** es cualquier número que comienza desde 0, poniendo 0 porque el límite no devuelve ningún registro en la consulta. Poner un número digamos 5 devolverá cinco registros. Si los registros en la tabla especificada son menores que N, entonces todos los registros de la tabla consultada se devuelven en el conjunto de resultados.
- El valor **OFFSET** también se usa con más frecuencia junto con la palabra clave **LIMIT**. El valor OFFSET nos permite especificar **qué fila comenzar a partir de la recuperación de datos.**

EJEMPLO + SINTAXIS

```
SELECT campo1, campo2, campo3
FROM Tabla
LIMIT 10;
```

```
SELECT campo1, campo2, campo3
FROM Tabla
LIMIT 4 OFFSET 5;
-->recuperamos 4 registros a partir del 5to registro
```

DISTINCT

- Utilizamos el **DISTINCT** cuando se desea omitir registros que contienen datos duplicados en los campos seleccionados.

```
SELECT DISTINCT campo2
FROM Tabla;
```

EJEMPLO + SINTAXIS

```
SELECT DISTINCT type
FROM DebtMovement
LIMIT 10;
```

// Funciones de agregación

IT BOARDING

BOOTCAMP

FUNCIONES DE AGREGACIÓN

Las vamos a utilizar para visualizar cierta información agrupada y resumida, para esto utilizaremos las funciones de agregación.

- **COUNT**: devuelve el número total de filas seleccionadas por la consulta.
- **MIN**: devuelve el valor mínimo del campo que especifiquemos.
- **MAX**: devuelve el valor máximo del campo que especifiquemos.
- **SUM**: suma los valores del campo que especifiquemos.
- **AVG**: devuelve el valor promedio del campo que especifiquemos.

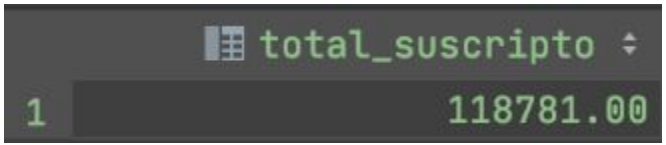
FUNCIONES DE AGREGACIÓN

```sql
SELECT MAX(campo1) FROM Tabla;
```

```sql
SELECT COUNT(*) FROM Tabla;
```

EJEMPLO + SINTAXIS

```
SELECT sum(amount) as total_suscripto
FROM DebtMovement
WHERE debt_id = 33729488
  AND detail = 'total';
```



	total_suscripto
1	118781.00

Ejemplo + Sintaxis

```
SELECT sum(amount) as total_suscripto
FROM DebtMovement
WHERE debt_id = 33729488
  AND detail = 'total';
```

	total_suscripto
1	118781.00

BUENAS PRÁCTICAS

- Por **default** el **order by** utiliza ordenamiento **ASC**.
- Tratar de usar el **nombre** de las columnas en lugar de la posición.
- No se recomienda usar el **order by** para consultas con **grandes volúmenes de datos**.
- **Evitar** el uso del **select * from tabla sin filtros** o **limite** de **registros**.
- Usar los **filtros** adecuados para acotar nuestro set de datos.

// Práctica

IT BOARDING

BOOTCAMP

Resolver

1. Realizar una consulta que devuelva todo los campos de la tabla debtmovement en donde el debt_id sea igual a 33729.
2. Aplicarle a la consulta anterior un ordenamiento en forma descendente por created_at.
3. Aplicarle a la consulta anterior un filtro por type en donde este sea igual a income.
4. Aplicarle a la consulta del punto 1 un filtro en donde el type sea income o paid.
5. Realizar una consulta sobre la tabla debt que devuelva todos los campos y filtrar para que los registros correspondan a Enero 2020.
6. Realizar una consulta sobre la tabla debt limitando la cantidad de registros a 10.
7. Realizar una consulta que devuelva los diferentes type de la tabla debtmovement.
8. Realizar una consulta que devuelva la suma de amount del debtid 33729488 en donde el type se income.
9. Realizar una consulta donde devuelva la cantidad de registros de la tabla deadletter donde el status sea igual a error.

Verdadero o Falso

1. La función de agregación sum sirve para contar registros de una tabla **Verdadero / Falso.**
2. Si aplicamos un Limit en la consulta lo que hacemos es limitar la cantidad de registros **Verdadero / Falso.**
3. Si aplicamos un ORDER BY por algún campo, por default el ordenamiento será en forma descendente **Verdadero / Falso**.
4. El Where se utiliza únicamente para filtrar campos que se encuentran después del Select **Verdadero / Falso**.
5. Si colocamos el NOT delante del between estamos incluyendo los rangos de valores en la cláusula.**Verdadero / Falso**.

Corregir las siguientes consultas

```
1) SELECT sum(amount) as total_suscripto
FROM DebtMovement
  AND debt_id = 33729488
  AND detail = 'total';
```

```
2) SELECT total_suscripto and detail
FROM DebtMovement
  AND debt_id = 33729480
  AND detail = 'total';
```

Gracias.

IT BOARDING

BOOTCAMP