En esta guía, revisaremos los conceptos básicos de la cláusula SELECT en SQL, para que puedas refrescar los conceptos antes de iniciar tu nueva práctica.

Cláusula IN

La **cláusula IN** se utiliza para filtrar resultados en base a un conjunto de valores. Es especialmente útil cuando se quiere realizar una búsqueda en base a varios valores posibles en una columna.

```
SELECT * FROM nombre_tabla WHERE columna3 IN (valor1, valor2, valor3);
```

Cláusula LIKE

La **cláusula LIKE** se utiliza para buscar patrones en una columna de texto. Permite seleccionar registros cuyos valores en una columna coinciden con un patrón específico. Hay dos comodines principales que se utilizan con la cláusula LIKE:

- %: Representa cero o más caracteres.
- _: Representa un solo carácter.

```
SELECT * FROM nombre_tabla WHERE columna1 LIKE 'patrón%';
```

Cláusula ORDER BY

La cláusula ORDER BY se utiliza para ordenar los resultados de una consulta según el valor de una o más columnas.

```
SELECT * FROM nombre_tabla ORDER BY columna1 DESC;
```

Función MAX

La **función MAX()** se utiliza para obtener el valor máximo en una columna específica.

```
SELECT MAX(nombre_columna) FROM nombre_tabla;
```

Función MIN

La **función MIN()** se utiliza para obtener el valor mínimo en una columna específica.

```
SELECT MIN(nombre_columna) FROM nombre_tabla;
```

Función COUNT

La **función COUNT()** se utiliza para contar el número de filas como resultado de una consulta.

```
SELECT COUNT(*) FROM nombre_tabla GROUP BY nombre_columna;
```

Función SUM

La **función** SUM() se utiliza para obtener la suma de los valores en una columna específica.

```
SELECT SUM(nombre_columna) FROM nombre_tabla GROUP BY nombre_columna;
```

Función AVG

La **función** AVG() se utiliza para calcular el promedio de los valores en una columna específica.

```
SELECT AVG(nombre_columna) FROM nombre_tabla GROUP BY nombre_columna;
```

Cláusula GROUP BY

La **cláusula** GROUP BY se utiliza para agrupar filas en base a los valores de una columna y aplicar funciones de agregación, como COUNT, a cada grupo.

```
SELECT COUNT(*) FROM nombre_tabla GROUP BY nombre_columna;
```

Cláusula HAVING

La **cláusula HAVING** se utiliza en SQL para filtrar resultados de una consulta que involucra una operación de agregación (como SUM, COUNT, AVG, etc.) en grupos de filas, similar a la cláusula WHERE pero aplicada a grupos.

```
SELECT nombre_columna, COUNT(nombre_columna) FROM nombre_tabla GROUP BY
nombre_columna HAVING nombre_columna > 3;
```

Cláusula AS

La **cláusula AS** se utiliza para asignar un alias o nombre temporal a una columna o una tabla en una consulta SQL. Esto facilita la referencia a la columna o tabla con un nombre más legible o abreviado.

💡*Recuerda que no hace falta usar la cláusula AS, si dejas un espacio automáticamente se infiere que estas asignando un alias.*

```
SELECT nombre_columna AS alias_columna FROM nombre_tabla AS alias_tabla;
```

Cláusula LIMIT

La **cláusula LIMIT** se utiliza para limitar el número de filas que se devuelven en el resultado de una consulta. Esto es útil cuando se desea obtener sólo un número específico de registros de una tabla.

```
SELECT * FROM nombre_tabla LIMIT cantidad_filas;
```

Cláusula CASE

La **cláusula CASE** se utiliza en SQL para realizar evaluaciones condicionales y devolver un valor basado en una condición especificada. La estructura básica de una cláusula CASE es la siguiente:

```
CASE
    WHEN condición_1 THEN resultado_1
    WHEN condición_2 THEN resultado_2
    ...
    ELSE resultado_predeterminado
END
```

- condición_1, condición_2, etc.: Son las condiciones que se evalúan. Si alguna de estas condiciones es verdadera, se devuelve el resultado correspondiente.
- resultado_1, resultado_2, etc.: Son los valores que se devolverán si las condiciones correspondientes son verdaderas.
- ELSE resultado_predeterminado: Es un valor opcional que se devuelve si ninguna de las condiciones anteriores es verdadera.

Aquí tienes un ejemplo simple para ilustrar cómo se utiliza la cláusula CASE:

Tenemos la tabla empleados con las columnas salario y queremos categorizar a los empleados en función de su salario:

```
SELECT nombre, salario,
    CASE
        WHEN salario >= 3000 THEN 'Alto'
        WHEN salario >= 2000 THEN 'Medio'
        ELSE 'Bajo'
    END AS categoria_salario
FROM empleados;
```

En este ejemplo, evaluamos el salario de cada empleado y los categorizamos como "Alto", "Medio" o "Bajo" según su valor. El resultado incluirá una columna llamada categoria_salario que muestra la categoría correspondiente.