

¿Por qué son tan importantes las consultas?

Alrededor de las consultas a las bases de datos se han creado varias especialidades como ETL, o transformación de datos, business intelligence e incluso machine learning.

Estructura básica de un Query

Estructura básica de un Query

SELECT city, count(\*) AS total

FROM people

WHERE active = true

GROUP BY city

ORDER BY total DESC

HAVING total >= 2;

# SELECT

SELECT se encarga de proyectar o mostrar datos

Se puede cambiar el nombre del campo con AS

SELECT título AS encabezado  
FROM posts;

Existe una función de SELECT para poder contar la cantidad de registros

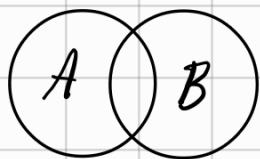
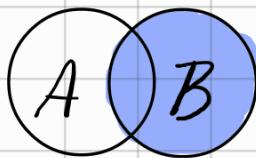
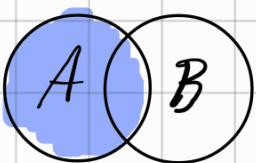
SELECT COUNT(\*)  
FROM posts;

Playground 05.sql  
Select en SQL

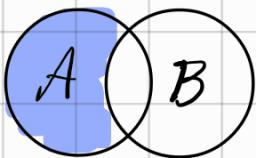
# FROM y SQL JOINS

Join

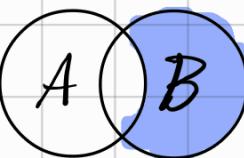
Diferencia



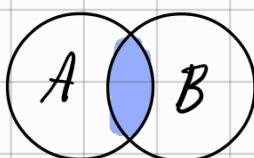
Left Join



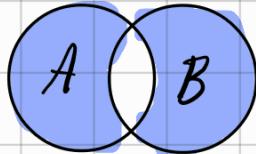
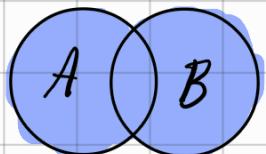
Right Join



Intersección



Inner Join



Union

Diferencia simétrica

Outer Join

Utilizando la sentencia  
FROM

FROM es de donde traer los  
datos

Playground 06. SQL  
FROM, LEFT JOIN en SQL

WHERE

WHERE es la sentencia que nos ayuda a  
filtrar tuplas o registros dependiendo  
de las características que elegimos.

La propiedad **LIKE** nos ayuda a traer registros  
de los cuales conocemos sólo una parte de la  
información. **LIKE '%algo%'**

La propiedad **BETWEEN** nos sirve para  
arrojar registros que estén en el medio de  
dos. Por ejemplo los registros con id  
entre 20 y 30.

Utilizando la sentencia WHERE solo y no nulo

El valor nulo en una tabla generalmente es su valor por defecto cuando nadie le asignó algo diferente. La sintaxis para hacer búsquedas de datos nulos es **IS NULL**. La sintaxis para buscar datos que no son nulos es **IS NOT NULL**.

Playground 07.Sgd  
Filtrando datos con WHERE

GROUP BY

GROUP BY tiene que ver con agrupación. Indica a la base de datos que criterios debe tener en cuenta para agrupar

# ORDER BY y HAVING

La sentencia ORDER BY tiene que ver con el ordenamiento de los datos dependiendo de los criterios que quieras usar.

ASC sirve para ordenar de forma ascendente.

DSC sirve para ordenar de forma descendente.

LIMIT se usa para limitar la cantidad de resultados que arroja el query.

HAVING tiene una similitud con WHERE, sin embargo el uso de ellos depende del orden. Cuando se quiere seleccionar las agrupadas únicamente se puede hacer con HAVING.

Siempre va después del GROUP BY.

Playground 08.sql  
Agrupamiento y Ordenamiento de Datos

# El interminable agujero de conejo (Nested queries) queries anidadas

Los *Nested queries* significan que dentro de un query podemos hacer otro query. Esto sirve para hacer join de tablas, estando una en memoria. También teniendo un query como condicional del otro.

Este proceso puede ser tan profundo como quieras, teniendo infinitas queries anidadas.

Se le conoce como un producto cartesiano ya que se multiplican todos los registros de una tabla con todos los del nuevo query.

Esto provoca que el query sea difícil de procesar por lo pesado que puede resultar.

# ¿Cómo convertir una pregunta en un query SQL?

De pregunta a query

Lo que quieres mostrar

SELECT

De donde lo voy a sacar

FROM

Los filtros de los datos que quieres mostrar

WHERE

Los rubros por los que me interesa agrupar la información

GROUP BY

El orden en que quiero presentar mi información

ORDER BY

Los filtros que quiero que mis datos agrupados tengan

HAVING

Preguntando de la base de datos

GROUP\_CONCAT toma el resultado del query y lo pone como campo separado por comas.

Consultando Platziblog

Puedes usar una abreviación para evitar escribir lo mismo cada vez.

FROM categorias AS c

Playground 09.sql  
Prueba Final con Platziblog