Material Imprimible

Curso Data Analytics

Módulo 2: SQL para Análisis de datos

Contenidos:

- Software SQL
- Sentencias SQL básicas: SELECT y FROM
- Filtrado con Where
- Group By y funciones de agregación
- Order By
- Joins
- Resumen Sintaxis

Software SQL - ¿Qué es SQL? ¿Qué herramienta conoceremos en el curso?

SQL es el lenguaje estructurado para consultas de bases de datos, que nos permite obtener respuestas de forma rápida y eficiente, buscando transformar datos almacenados en información relevante.

Durante el curso aprenderemos todas las principales sintaxis a través del taller SQL de la herramienta ORACLE APEX.

Oracle APEX es una suite para el despliegue de aplicaciones de forma rápida, que incluye un motor de base de datos para el uso y la práctica del lenguaje SQL. Lo más novedoso es que se accede a través de un entorno web, por lo cual, evita que debamos realizar engorrosas instalaciones.

Para comenzar, el alumno deberá crearse un espacio gratuito ingresando en el siguiente link y completando sus datos verdaderos, ya que se requiere una validación vía correo electrónico.

https://apex.oracle.com/es/learn/getting-started/

Importante: Seleccionar "Solicitar espacio de trabajo gratuito"

Con esta herramienta se podrán ejecutar las sentencias SQL que aprenderemos durante este módulo.

Sentencias SQL básicas

SELECT

Es la sentencia más utilizada para la extracción y consulta de datos. Se usa en conjunto con el FROM obligatoriamente.
En el SELECT (que debe escribirse en inglés) se deben mencionar todas las columnas que me interesan obtener. En el caso de que desee acceder a todas las columnas, debo completar con un asterisco (*)

FROM

Esta sentencia indica el origen de donde se obtendrán los datos. Típicamente es una tabla y debe contener las columnas que se le solicita en el SELECT

Algunos ejemplos:

En este caso, seleccionamos algunas columnas de la tabla CLIENTES:

```
SELECT CODIGOCLIENTE, SEGMENTO, EDAD
FROM CLIENTES
```

En este caso, seleccionamos todas las columnas de la tabla CLIENTES:

```
SELECT *
FROM CLIENTES
```

Fuente:
https://www.w3schools.com/sql/sql_select.asp

Filtrado de filas

WHERE

La sentencia WHERE nos permite establecer una o más condiciones para filtrar las filas de datos.

Al inicializar el WHERE debemos introducir los distintos campos a filtrar y su condición, donde típicamente se usan operadores relacionales que permiten establecer un vínculo entre el campo y los datos.

Los operadores relacionales más utilizados son los siguientes:

=	Evalúa si un campo es igual	campo = “dato”
>	Evalúa si un campo es mayor. Puede expresarse mayor igual también	campo > 40 campo >= 40
<	Evalúa si un campo es menor. Puede expresarse menor igual también	campo < 40 campo <= 40
<>	Evalúa si un campo es distinto	campo <> “dato”

A su vez, podemos añadir más condiciones usando operadores lógicos, tales como:

AND	Retorna verdadero sólo si todas las condiciones se cumplen	campo1 = “abc” AND campo2 > 25000
OR	Retorna verdadero si una de las condiciones se cumple	campo1 = “abc” OR campo2 > 25000
IN	Permite evaluar múltiples valores en una única condición generando una lista	campo1 in (“valor1”, “valor2”, “valor3”)
NOT	Permite negar una condición posterior	NOT campo1 = “abc”

Ejemplos:

-- Listar todas las columnas de los clientes del Segmento Pequeña Empresa

```
SELECT *
FROM CLIENTES
WHERE SEGMENTO = 'Pequeña empresa'
```

-- Listar todas las columnas de los clientes del Segmento Pequeña Empresa y que sean mayores a los 40 años de Edad

```
SELECT *
FROM CLIENTES
WHERE SEGMENTO = 'Pequeña empresa' AND EDAD > 40
```

-- Listar todas las columnas de los clientes del Segmento Cliente o Segmento Empresa (dos alternativas posibles)

```
Alternativa 1 – Usando el OR
SELECT *
FROM CLIENTES
WHERE SEGMENTO = 'Cliente' OR SEGMENTO = 'Empresa'

Alternativa 2 - Usando el IN
SELECT *
FROM CLIENTES
WHERE SEGMENTO IN ('Cliente','Empresa')
```

Fuente:
https://www.w3schools.com/sql/sql_where.asp
https://www.w3schools.com/sql/sql_in.asp

GROUP BY y funciones de agregación

El GROUP BY es una sentencia que utilizamos cuando necesitamos resumir un conjunto de filas bajo un determinado criterio (uno o más campos). Se usa típicamente con funciones de agregación, que permiten adicionar a nuestra consulta conteos, sumatorias, promedios, etc.

Funciones de agregación más utilizadas:

COUNT	Cuenta la cantidad de filas	COUNT(Nombre de columna)

		COUNT(*)
SUM	Sumarizan todos los valores existentes de un campo numérico	SUM(Nombre de columna)
AVG	Extraen el promedio de un campo numérico	AVG(Nombre de columna)
MIN / MAX	Extraen el valor mínimo o máximo de un campo	MIN(Nombre de columna) MAX(Nombre de columna)

Ejemplos:

-- ¿Cuál es la Cantidad de productos por cada Categoría?

```
SELECT CATEGORIA, COUNT(IDPRODUCTO) AS CANTIDAD_PRODUCTO
FROM PRODUCTOS
GROUP BY CATEGORIA
```

-- ¿Cuál es el monto total vendido en Argentina?

```
SELECT PAISTIENDA, SUM(TOTAL) AS CANTIDAD_TOTAL_EN_PESOS
FROM COMPRAS
WHERE PAISTIENDA = 'Argentina'
GROUP BY PAISTIENDA
```

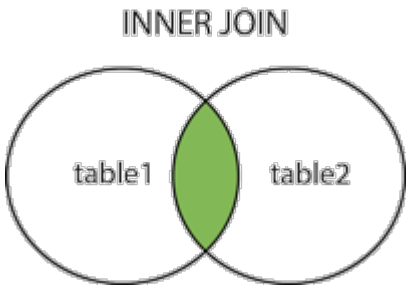
Fuente:
https://www.w3schools.com/sql/sql_groupby.asp

JOINS

Permiten unir distintas tablas a través de claves que complementan los datos existentes entre sí. Los JOINS se realizan a través de campos que existen en ambas tablas y la union se hace a nivel fila.

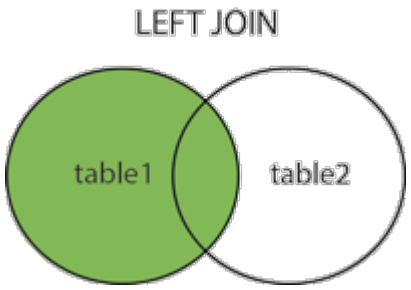
Existen distintos tipos de Joins:

- **Inner Join:** une dos tablas y sólo retorna los registros que estén presentes en ambas tablas. Por ejemplo, si la tabla 1 es CLIENTES y la tabla 2 es COMPRAS, en la respuesta solo tendremos clientes que hayan realizado alguna compra, y por el contrario, algún cliente registrado que aún no compró, no aparecerá.

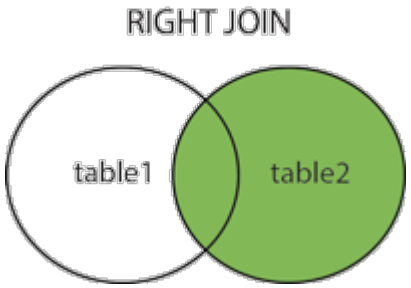


- **Left Join**: une dos tablas, definiendo la primera como “fuerte” y de ella permanecen todos los registros. Si hay contenido de los mismos en la tabla 2 (o débil), se agregan los datos correspondientes. Por ejemplo, se

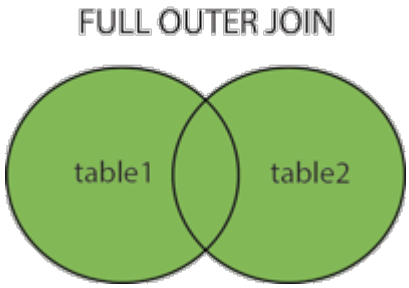
retornarán todos los CLIENTES en la respuesta de la consulta, y tendrían los datos de las COMPRAS si las hubiesen realizado.



- **Right Join:** une dos tablas, definiendo la primera como "débil" y la segunda como "fuerte", de la cuál se retornan todos los registros. Invierte la lógica del Left Join, priorizando la segunda tabla por sobre la primera.



- **Full Outer Join**: devuelve todos los registros estén presentes en la tabla 1 o 2.



Ejemplos:

-- Listar las compras con su categoría de producto

```
SELECT IDPEDIDO, CODIGOCLIENTE, CATEGORIA, TOTAL
FROM COMPRAS T1
LEFT JOIN PRODUCTOS T2 ON T1.IDPRODUCTO = T2.IDPRODUCTO
```

-- ¿Cuánto gastó cada cliente?

```
SELECT T1.CODIGOCLIENTE, NOMBRE, SUM(TOTAL) AS TOTAL_CLIENTE
FROM CLIENTES T1
INNER JOIN COMPRAS T2 ON T1.CODIGOCLIENTE = T2.CODIGOCLIENTE
GROUP BY T1.CODIGOCLIENTE, NOMBRE
```

Fuente:

https://www.w3schools.com/sql/sql_join.asp

ORDER BY

Esta sentencia se coloca la final de nuestra consulta y nos permite ordenar el resultado por una o más columnas, de manera ascendente (ASC) o descendente (DESC)

Ejemplos:

-- Ordenar mi lista de clientes por Edad, de mayor a menor

```
SELECT *
FROM CLIENTES
ORDER BY EDAD DESC
```

Fuente:
https://www.w3schools.com/sql/sql_orderby.asp

Resumen Sintaxis integral SQL

Orden de implementación de sintaxis

SELECT

FROM

JOIN

WHERE

GROUP BY

ORDER BY

- Debe respetarse el orden de escritura de sintaxis.
- Las sentencias Select y el From deben usarse obligatoriamente.
- De las restantes, no todas son obligatorias, solo deben usarse cuando sea necesaria su funcionalidad.