SUB CONSULTAS

DEFINICIÓN: Es una consulta dentro de otra consulta SQL. Las subconsultas se pueden incluir dentro de una cláusula SELECT, FROM, JOIN o WHERE.

Ejemplo 01: Se desea un reporte que muestre la cantidad de retardos registrados por cada tipo de excusa presentada. Dicho reporte debe mostrar los siguientes datos de las excusas: Código, descripción y cantidad de retardos registrados. Además, los datos se deben presentar ordenados por el campo código de la excusa en forma **Descendente**.

```
SELECT   pkcodexcu ,
         Dexcusa ,
         (SELECT COUNT(*) FROM tdretardos WHERE  FKcodexcu = PKcodexcu)  AS  Total
FROM     tmexcusas
ORDER BY   pkcodexcu DESC ;
```

Ejemplo 02: Se desea un reporte que muestre la cantidad de retardos registrados por cada tipo de excusa presentada. Dicho reporte debe mostrar los siguientes datos de las excusas: Código, descripción y cantidad de retardos registrados. Además, los datos se deben presentar ordenados por el campo total de retardos en forma **Descendente**.

```
SELECT   PKcodexcu ,
         Dexcusa ,
         (SELECT COUNT(*) FROM tdretardos WHERE  FKcodexcu = PKcodexcu)  AS  Total
FROM     tmexcusas
ORDER BY   Total  DESC ;
```

Ejemplo 03: Se desea un reporte que muestre la cantidad de retardos que tiene cada empleado. Dicho reporte debe mostrar de cada empleado: el número de cédula, el nombre y la cantidad de retardos que tuvo cada empleado. Además, el reporte debe presentar los datos ordenados por el campo Total (**Ascendente**) y número de cédula (**Descendente**).

```
SELECT PKcedemple , Nomemple ,
    (
        SELECT COUNT(FKcodexcu) FROM tdretardos WHERE FKcedemple = PKcedemple
    ) AS Total
FROM tmempleados
ORDER BY Total, PKcedemple::BIGINT DESC ;
```

Ejemplo 04: Se desea un reporte que muestre de todos los empleados los siguientes datos: cédula, nombre, fecha de nacimiento, la edad actual, el código del cargo, la descripción del cargo y el sueldo en base al cargo en formato 999,999,999.99. Además, los datos se deben mostrar ordenador por el campo código del cargo (**Ascendente**) y la cédula (**Ascendente**).

```
SELECT PKCedemple , Nomemple , Fecha ,
    EXTRACT(YEAR FROM AGE(CURRENT_DATE, fecha)) AS "Edad" ,
    Fkcodcar ,
    (SELECT dcargo FROM tmcargos WHERE PKcodcar = FKcodcar) AS "Cargo" ,
    TO_CHAR(
            (SELECT sueldo FROM tmcargos WHERE PKcodcar = FKcodcar) ,
        '999G999G999D99'
    ) AS "Sueldo"
FROM tmempleados
ORDER BY FKcodcar , PKcedemple::BIGINT ;
```

Ejemplo 05: Se desea mostrar el listado de los retardos de todos los empleados. Dicho reporte debe mostrar: el Número del retardo; del empleado: número de cédula y nombre; de la excusa: código y descripción; la fecha y la hora del registro. Además, se desea el reporte ordenado por los campos: código de la excusa (**Descendente**) y fecha (**Ascendente**).

```
SELECT  PKnr ,     Fkcedemple ,
 (
     SELECT nomemple FROM tmempleados  WHERE  A.FKcedemple = PKcedemple
 )   AS Nombre ,
 Fkcodexcu ,
 (
     SELECT dexcusa FROM tmexcusas   WHERE  A.fkcodexcu = PKcodexcu
  )  AS Excusa ,
   Fecha ,     Hora ,      fkcods
FROM  tdretardos   AS   A
ORDER BY  FKcodexcu DESC  ,     fecha  ;
```

Ejemplo 06: Se desea mostrar el listado de los retardos del empleado con cédula '6000' y donde dichos retardos estén entre el 01/Nov/2022(Inclusive) y 30/Nov/2022(Inclusive). Dicho reporte debe mostrar: el Número del retardo; del empleado: número de cédula y nombre; de la excusa: código y descripción; la fecha y la hora del registro. Además, se desea el reporte ordenado por los campos: código de la excusa (**Descendente**) y fecha (**Ascendente**).

```
SELECT  C.PKnr ,
  C.FKcedemple ,
  (
    SELECT nomemple FROM tmempleados AS A WHERE  C.fkcedemple  =  A.PKcedemple
  )  AS Nombre  ,
  C.FKcodexcu  ,
  (
      SELECT dexcusa FROM tmexcusas AS B WHERE  C.FKcodexcu  =  B.PKcodexcu
  )  AS  Excusa ,
  C.fecha ,
  C.hora
FROM  tdretardos  AS  C
WHERE ( C.fecha  BETWEEN '2022-11-01' AND '2022-11-30' )  AND  (C.fkcedemple ='6000')
ORDER BY  C.fkcodexcu DESC,  C.fecha ;
```

Ejemplo 07 (WHERE): Se desea mostrar los primeros 10 retardos de la empleada con nombre "**NATASHA**", donde dicho reporte debe mostrar los datos debidamente ordenado por el campo fecha (**Ascendente**).

```
SELECT   COUNT(FKcedemple)  AS "Total Retardos"
FROM   tdretardos
WHERE FKcedemple = ( SELECT PKcedemple FROM tmempleados WHERE nomemple ='NATASHA')
;
```

```
SELECT   PKnr , FKcedemple , FKcodexcu , fecha
FROM   tdretardos
WHERE FKcedemple = ( SELECT PKcedemple FROM tmempleados WHERE nomemple ='NATASHA')
ORDER BY   fecha
LIMIT   10 ;
```

```
SELECT   PKnr , FKcedemple , FKcodexcu , fecha
FROM   tdretardos
WHERE FKcedemple = (
                        SELECT   PKcedemple   FROM   tmempleados
                        WHERE    nomemple  LIKE   'NATASHA'
                   )
ORDER BY   fecha
LIMIT   10 ;
```

Ejemplo 08: Visualizar un listado de los cargos cuyo sueldo esa mayor o igual al sueldo promedio .

```
SELECT   to_char( AVG(sueldo), '999G999G999D99' )
FROM   tmcargos  ;

SELECT  PKcodcar  ,   dcargo  ,  to_char(sueldo, '999G999G999D99' )
FROM   tmcargos
WHERE  sueldo >=  (   SELECT AVG(sueldo)  FROM tmcargos   )  ;
```

Ejemplo 09 (Gemini) : Se quiere mostrar el listado de todos los empleados cuyo salario sea superior al promedio de la empresa.

```
SELECT nombre, apellido, salario
FROM empleados
WHERE salario > (SELECT AVG(salario) FROM empleados)  ;
```

Ejemplo usando CTE (Common Table Expressions):

```
WITH promedio_empresa AS (
        SELECT AVG(salario) as valor FROM empleados
)
SELECT nombre, apellido, salario
FROM empleados, promedio_empresa
WHERE salario > promedio_empresa.valor;
```

Ejemplo 10 (Gemini - IN): Se desea un reporte que muestre el nombre de los productos donde en la descripción de la categoría específique la palabra "Electrónica".

```
SELECT nombre_producto
FROM productos
WHERE id_categoria IN (
                        SELECT   id_categoria
                        FROM    categorias
                        WHERE   descripcion   LIKE    '%Electrónica%'

                    ) ;
```

Ejemplo 11 (Gemini): Mostrar de cada cliente, el total de pedidos realizados.

```
SELECT
   nombre_cliente,
   (
        SELECT COUNT(*)
        FROM pedidos
        WHERE pedidos.id_cliente = clientes.id_cliente
    ) AS total_pedidos

FROM   clientes   ;
```

Ejemplo 12 (Gemini – FROM) El ejemplo más común para entender una **subconsulta en el FROM** es cuando necesitas realizar un cálculo (como un promedio o una suma) y luego usar ese resultado como si fuera una tabla normal para compararlo con otros datos.

Imagina que tienes una tabla de **ventas** y quieres saber qué vendedores superaron el promedio de ventas de la empresa.

la tabla ventas:

vendedor	monto
Ana	1000
Luis	500
Marta	1500

La Consulta

Queremos comparar cada venta individual contra el promedio global (1000 en este caso).

```
SELECT    vendedor,   monto,   promedio_global
FROM   ( -- Esta es la subconsulta que actúa como tabla temporal
                  SELECT AVG(monto) AS promedio_global FROM ventas
         ) AS tabla_promedio, ventas
WHERE ventas.monto > tabla_promedio.promedio_global;
```

¿Qué está pasando aquí?

1.- La Subconsulta (El "mini-select"): SQL ejecuta primero lo que está entre paréntesis. Calcula el promedio total y crea una "tabla virtual" de una sola celda con el valor 1000.

2.- El Alias (AS tabla_promedio): Es obligatorio. SQL necesita que le pongas un nombre a esa tabla temporal para poder referenciarla.

3.- El FROM principal: Ahora el motor dice: "Tengo mi tabla real ventas y mi tabla inventada tabla_promedio. Voy a cruzarlas".

4.- El WHERE: Compara el monto de cada fila con el promedio_global que calculamos antes.

Resultado Final:

vendedor	monto	promedio_global
Marta	1500	1000

Ejemplo 13 (Gemini – FROM): Obtener el salario máximo de los departamentos, pero solo de aquellos cuyo promedio salarial es mayor a 5000.

```
SELECT nombre_depto, max_salario
FROM (
        SELECT id_departamento, AVG(salario) as promedio, MAX(salario) as max_salario
        FROM empleados
        GROUP BY id_departamento
    ) AS resumen_deptos
JOIN departamentos   d    ON    d.id = resumen_deptos.id_departamento
WHERE promedio > 5000;
```

Estás ante una consulta SQL que utiliza una **subconsulta** (también llamada tabla derivada) y un **JOIN**. Básicamente, el código busca identificar qué departamentos tienen un sueldo promedio alto y mostrar cuánto gana la persona que más cobra en ellos.

En primer lugar se ejecuta el código:

```
SELECT id_departamento,
        AVG(salario) as promedio,
        MAX(salario) as max_salario
FROM empleados
GROUP BY id_departamento
```

¿Qué hace? Agrupa a los empleados por departamento.
¿Qué calcula? El salario promedio y el salario más alto de cada grupo.
Resultado temporal: Una tabla virtual llamada resumen_deptos.

En segundo lugar ejecuta el JOIN y la condición

```
JOIN departamentos   d   ON    d.id = resumen_deptos.id_departamento
WHERE promedio > 5000;
```

El JOIN: Une la subconsulta con la tabla de departamentos para poder obtener el nombre real del departamento (como "Ventas" o "IT"), ya que en la tabla de empleados probablemente solo tienes el ID.
El Filtro (WHERE): Aquí está el truco. Solo se queda con los departamentos donde el sueldo promedio es **mayor a 5000**.

Ejemplo Práctico
Imagina que tenemos estas dos tablas:
Tabla: empleados

nombre	id_departamento	salario
Ana	1 (IT)	6000
Luis	1 (IT)	8000
Eva	2 (RRHH)	3000
Pepe	2 (RRHH)	4000

Tabla: departamentos

id	nombre_depto
1	Tecnologías
2	Recursos Humanos

¿Qué pasaría al ejecutar el código?
1.- Cálculo interno:
* IT tiene un promedio de **7000** y un máximo de **8000**.
* RRHH tiene un promedio de **3500** y un máximo de **4000**.
2.- Filtro: El WHERE promedio > 5000 descarta a Recursos Humanos.
3.- Resultado final:

nombre_depto	max_salario
Tecnologías	8000

Un pequeño detalle técnico
Aunque el código anterior funciona, en SQL estándar es más común filtrar grupos usando la cláusula HAVING. Tu código es totalmente válido, además se considera más "limpio" sin tantas subconsultas, podrías escribirlo así:

```
SELECT      d.nombre_depto,   MAX(e.salario) as max_salario
FROM        empleados   e
JOIN        departamentos d ON e.id_departamento = d.id
GROUP BY    d.nombre_depto
HAVING      AVG(e.salario) > 5000;
```

Ejemplo 14: Se desea un reporte que muestre de las cuentas por cobrar los siguientes datos:
* El número del proyecto
* La cédula del cliente,
* El monto total del proyecto (En formato 99.999.999,99) y
* El monto ACTUALIZADO DE LOS ABONADOS al proyecto (En formato 99.999.999,99).

Además, los datos se deben presentar ordenados por el campo número del proyecto (**Ascendente**).

```
SELECT
   tdctasxcobrar.fknproyec AS "Numero Proyecto",
   tdctasxcobrar.fkced_clie AS "Cedula Clientes",
   TO_CHAR(tdproyectos.costo_p, '99G999G999D99') AS "Monto Total Proyecto",
   TO_CHAR(
                (
                    SELECT SUM(mabono) FROM tdabonos
                    WHERE tdabonos.fknproyec = tdctasxcobrar.fknproyec

                ) ,
                '99G999G999D99'
            )  AS "Monto Abonos Realizados"
FROM        tdctasxcobrar
JOIN        tdproyectos ON tdctasxcobrar.fknproyec = tdproyectos.pknproyec
ORDER BY    tdctasxcobrar.fknproyec    ;
```

Fuente

01.- https://josejuansanchez.org/bd/unidad-09-teoria/index.html
02.-https://learnsql.es/blog/5-ejemplos-de-subconsultas-sql/
03.- https://www.srcodigofuente.es/subconsultas-en-sql
04.- https://jorgesanchez.net/manuales/sql/select-subconsultas-sql2016.html
05.- https://learn.microsoft.com/es-es/sql/relational-databases/performance/subqueries?view=sql-server-ver16
06.- https://desarrolloweb.com/articulos/2337.php

07.- https://docs.aws.amazon.com/es_es/redshift/latest/dg/r_Subquery_examples.html
08.- https://learnsql.es/blog/como-practicar-subconsultas-sql/
09.- https://www.aulaclic.es/sql/t_5_1.htm
10.- SUB CONSULTAS HAVING
https://josejuansanchez.org/bd/unidad-09-teoria/index.html
11.- SUB CONSULTA HAVING - SUBSTRING()
https://www.ibm.com/docs/es/i/7.5?topic=subqueries-example-correlated-subquery-in-having-clause
12.- https://learn.microsoft.com/es-es/sql/relational-databases/performance/subqueries?view=sql-server-ver17
13.- Gemini