UNIVERSIDAD NACIONAL AUTÓNOMA DE MÉXICO

Facultad de Ingeniería

Bases de Datos

Grupo: 1

Prof: Ing. Fernando Arreola

Alumna: Soriano Barrera María Elena

TAREA 8

Semestre 2025-2

Subconsulta

La subconsulta, también conocida como consulta anidada, es una instrucción SELECT que se inserta dentro de otra consulta SQL, generalmente en las cláusulas WHERE, FROM o SELECT. Sirve para devolver un conjunto de datos que puede ser utilizado por la consulta principal. Es decir, una subconsulta es una consulta dentro de otra consulta.

Una subconsulta actúa como una fuente auxiliar de datos que permite que la consulta principal:

- Tome decisiones basadas en valores dinámicos (por ejemplo, promedios o máximos actuales),
- Filtre registros basados en listas derivadas,
- Genere columnas calculadas por fila,
- O combine información agrupada o relacionada sin necesidad de JOIN explícito.

Existen dos tipos fundamentales de subconsultas:

- **Subconsulta no correlacionada**
  - No depende de ningún valor de la consulta principal.
  - Se ejecuta una sola vez y su resultado se reutiliza.

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

- **Subconsulta correlacionada**
  - Utiliza una columna de la consulta principal para evaluarse.
  - Se ejecuta una vez por cada fila de la consulta externa.
  - Suelen tener mayor costo computacional.

```sql
SELECT nombre
FROM productos p
WHERE precio > (SELECT AVG(precio)
                FROM productos
                WHERE categoria_id = p.categoria_id);
```

Donde implementarlas

Las subconsultas se pueden utilizar en diversas cláusulas, siempre que cumplan con las reglas de sintaxis y contexto.

- **WHERE** → Comparar un valor con una lista o resultado dinámico
- **FROM** → Tratar el resultado como una tabla derivada (subconsulta en línea)
- **SELECT** → Incluir datos agregados por fila
- **HAVING** → Aplicar condiciones sobre funciones agregadas
- **INSERT**, **UPDATE**, **DELETE** → Para operaciones condicionales según subresultados

Condiciones y reglas para implementarlas

- La subconsulta debe estar entre paréntesis ().
- Puede devolver un único valor (ESCALAR), una lista (IN), o una tabla completa (subconsulta en línea).
- Puede ser **correlacionada** (usa valores de la consulta externa) o **no correlacionada** (independiente).
- Las subconsultas correlacionadas suelen tener un mayor costo computacional.

Ventajas de su implementación

- Mejoran la legibilidad del código.
- Permiten modularizar consultas complejas.
- Pueden usarse donde no se pueden usar JOINs.
- Útiles para filtros dinámicos o basados en agregados.
- Facilitan la reutilización de resultados intermedios.
- Permiten mayor expresividad y precisión en los filtros condicionales.

Características clave de una subconsulta:

- Se encierra entre **paréntesis**.
- Puede devolver un:
    - **Valor escalar** (una sola celda),
    - **Lista** (para operadores como IN o ALL),
    - **Tabla** completa (subconsulta en línea).
- Puede usarse en **cualquier parte** donde se espera una expresión o conjunto de resultados.

Ejemplos

- **WHERE**

  Seleccionar empleados cuyo salario sea mayor al salario promedio.

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

  Esta subconsulta calcula el salario promedio, y la consulta externa filtra a quienes ganan más.

- **FROM**

  Subconsulta que calcula el total vendido por cliente y lo usa como tabla.

```sql
SELECT cliente_id, total
FROM (
    SELECT cliente_id, SUM(monto) AS total
    FROM ventas
    GROUP BY cliente_id
) AS resumen_clientes
WHERE total > 10000;
```

  En esta subconsulta se crea una tabla temporal a partir de la subconsulta, y luego se filtra por condición externa.

- **Correlacionada en cláusula WHERE**

  Obtener productos cuya cantidad vendida sea superior al promedio de su categoría.

```sql
SELECT nombre
FROM productos p
WHERE cantidad > (
    SELECT AVG(cantidad)
    FROM productos
    WHERE categoria_id = p.categoria_id);
```

La subconsulta depende del valor de la fila actual (p.categoria_id).

- **SELECT**

Mostrar clientes junto con el total de compras que han hecho.

```sql
SELECT nombre,
       (SELECT SUM(monto) FROM ventas WHERE ventas.cliente_id = clientes.id) AS total_compras
FROM clientes;
```

Se agrega una columna calculada para cada fila usando una subconsulta.

- **HAVING**

Categorías que vendieron más de 100 productos. Aplicar condiciones sobre funciones agregadas.

```sql
SELECT categoria_id, SUM(cantidad) AS total_vendidos
FROM productos
GROUP BY categoria_id
HAVING SUM(cantidad) > (
    SELECT AVG(total)
    FROM (
        SELECT SUM(cantidad) AS total
        FROM productos
        GROUP BY categoria_id
    ) AS sub);
```

La subconsulta anidada devuelve el promedio de productos vendidos por categoría, y la HAVING lo compara con cada grupo.

- **INSERT**

Insertar en tabla de antiguos clientes los que no han comprado este año. Insertar datos obtenidos de otra tabla.

```sql
INSERT INTO antiguos_clientes (id, nombre)
SELECT id, nombre
FROM clientes
WHERE id NOT IN (
    SELECT cliente_id
    FROM ventas
    WHERE fecha >= '2024-01-01');
```

Inserta clientes que no aparecen en ventas recientes.

- **UPDATE**

  Aumentar sueldo a empleados por debajo del promedio. Actualizar registros condicionalmente según subresultados.

```
UPDATE empleados
SET salario = salario + 500
WHERE salario < (
    SELECT AVG(salario)
    FROM empleados);
```

  Solo se actualizan los empleados con salario menor al promedio.

- **DELETE**

  Eliminar productos que no tienen ventas. Eliminar registros que cumplan una condición basada en otra tabla.

```
UPDATE empleados
SET salario = salario + 500
WHERE salario < (
    SELECT AVG(salario)
    FROM empleados);
```

  Elimina productos que no están asociados a ninguna venta.

Referencias

[1] PostgreSQL Global Development Group, *The PostgreSQL Documentation: Subqueries*, PostgreSQL 15. [En línea]. Disponible en: https://www.postgresql.org/docs/current/functions-subquery.html

[2] J. Murach y J. Joel, *Murach's PostgreSQL Server SQL*. Fresno, CA: Mike Murach & Associates, 2021.

[3] W3Schools, "SQL Subqueries," *W3Schools*, 2024. [En línea]. Disponible en: https://www.w3schools.com/sql/sql_subqueries.asp

[4] C. Date, *An Introduction to Database Systems*, 8th ed. Boston, MA: Addison-Wesley, 2003.