UTN
Facultad Regional Córdoba

TUP

Tecnicatura Universitaria en Programación

LABORATORIO DE COMPUTACIÓN II

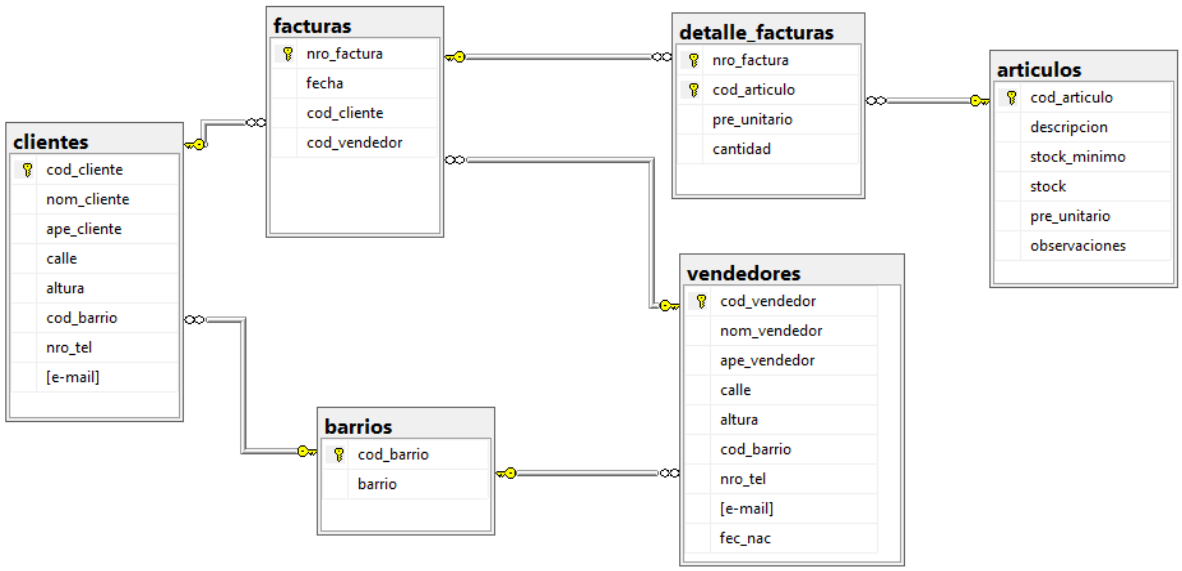
Unidad Temática N°2:
Subconsultas

Guía
1° Año – 2° Cuatrimestre

Índice

Problema 2.1: Subconsultas en el Where

Continuando con el negocio dedicado a la venta mayorista de artículos de librería se vuelve a presentar el diagrama de la base de datos “LIBRERIA”.



Planteo del problema: Los usuarios finales del sistema necesitan obtener la siguiente información para el funcionamiento del negocio y la toma de decisiones:

1. **Se solicita un listado de artículos cuyo precio es inferior al promedio de precios de todos los artículos.**

Para resolver este requerimiento, se podría utilizar una subconsulta que calcule el precio promedio de los artículos a la venta; luego en la consulta principal listar los artículos cuyo precio unitario sea menor al resultado de dicha subconsulta:

```
Select cod_articulo, descripcion, pre_unitario
from articulos
where pre_unitario <(Select AVG(pre_unitario) from articulos)
```

Results | Messages

	cod_articulo	descripcion	pre_unitario
1	1	Lápiz Evolution HB2 * 4 u	15.50
2	2	Papel p/forrar Fantas?a * 10 u	22.30
3	3	Papel p/forrar Ara?a * 10 u	25.50
4	4	Lápices Color cortos * 12 u	20.50
5	5	Fibras cortas * 6	25.30
6	6	Lápices Color largos * 12 u	27.90
7	7	Separadores tamaño rivadavia * 6 u	15.70
8	8	Carpeta fibra negra - Tamaño rivadavia	12.90
9	9	Carpeta Fantasía - Tamaño Rivadavia	15.90
10	10	Adhesivo sintético 30 gr	6.00

La subconsulta especificada en este test debe producir una *única fila* de resultados, es decir un único valor que en este caso sería el promedio de los precios es por ello que en este tipo de test la

subconsulta es una consulta sumaria de una sola columna asegurándose así que dará un único resultado.

Además, debemos tener en cuenta que la columna a la izquierda del operador de comparación es un valor numérico que está siendo comparado con otro valor numérico de la subconsulta, y no solo eso, ambos valores a comparar son precios.

2. **Emitir un listado de los artículos que no fueron vendidos este año. En ese listado solo incluir aquellos cuyo precio unitario del artículo oscile entre 50 y 100.**

Para entender la resolución podríamos comenzar con lo siguiente:

```
--Emitir un listado de los artículos que no fueron vendidos este año.
select cod_articulo, descripcion,pre_unitario,observaciones
from articulos
where cod_articulo not in (select cod_articulo
                           from detalle_facturas d
                           join facturas f on d.nro_factura=f.nro_factura
                           where year(fecha)=year(getdate())
                           )
```

La subconsulta devolvería un conjunto de códigos de artículos que SI se vendieron este año. El where de la consulta principal evaluaría los códigos de artículos de la tabla artículo que no estén dentro del conjunto de códigos de artículos devueltos por la consulta principal

Luego añadiremos la 2da parte de la condición de búsqueda la que va a estar fuera de la subconsulta ya que lo que se quiere filtrar con la nueva condición, son los artículos que NO fueron vendidos este año.

```
--Emitir un listado de los artículos que no fueron vendidos este año.
--En ese listado solo incluir aquellos cuyo precio unitario del artículo
--oscile entre 50 y 100.
select cod_articulo, descripcion,pre_unitario,observaciones
from articulos
where cod_articulo not in (select cod_articulo
                           from detalle_facturas d
                           join facturas f on d.nro_factura=f.nro_factura
                           where year(fecha)=year(getdate())
                           )
and pre_unitario between 50 and 100
```

Por otro lado, observemos cómo la columna a la izquierda y la devuelta por la subconsulta son códigos de artículos dada que en caso contrario no tendría lógica la comparación.

La misma consulta resuelto con el test de subconsultas EXISTS sería la siguiente:

```
select cod_articulo, descripcion,pre_unitario,observaciones
from articulos a
where not exists (select cod_articulo
                  from detalle_facturas d
                  join facturas f on d.nro_factura=f.nro_factura
                  where year(fecha)=year(getdate())
                   and d.cod_articulo=a.cod_articulo
                  )
and pre_unitario between 50 and 100
```

La subconsulta planteada devolverá los cod_articulo desde la tabla detalle_facturas de las facturas correspondiente a este año cuyo cód_artículo sean iguales al cód_articulo que se está intentando listar en la consulta principal. Este a.cod_articulo que es tomado desde la consulta principal es lo que se conoce como referencia externa ya que es un campo que no pertenece a la subconsulta, viene de afuera de la misma.

Por otro lado, para este test (es el único que es así) no interesa, qué devuelve la subconsulta, podrían ser varias columnas y hasta podría haber sido un * (asterisco) ya que el test EXISTS no compara nada, solo devuelve verdadero o falso si la subconsulta devuelve filas de resultado o no. Particularmente en este caso el test NOT EXISTS devolverá verdadero si la subconsulta no devuelve resultados, lo que implica que en las facturas de este año, la tabla detalle_facturas no tiene registros de ventas del artículo cuyo cód_articulo es igual al que se está intentando listar en la consulta principal

3. Genere un reporte con los clientes que vinieron más de 2 veces el año pasado.

Para este ejercicio, encontramos varias formas de resolverlo:

```
1. select cod_cliente,ape_cliente,nom_cliente
   from clientes c
   where 2 <(select count(*)
             from facturas f
             where c.cod_cliente=f.cod_cliente
             and year(fecha)=year(getdate())-1
             )
```

Esta primer forma, la subconsulta contará la cantidad de registros que hay en la tabla facturas correspondientes al año pasado donde el cód_cliente es igual al

cliente que se intenta mostrar en la consulta principal (referencia externa). Si esa cantidad es mayor a 2 (con un test de comparación en el where de la consulta principal), el registro del cliente se muestra.

```
2. select cod_cliente,ape_cliente,nom_cliente
   from clientes c
   where cod_cliente in (select cod_cliente
                         from facturas f
                         where year(fecha)=year(getdate())-1
                         group by cod_cliente
                         having count(*) >2
                         )
```

En este otro caso, la subconsulta lista los códigos de clientes desde la tabla facturas filtradas por el año pasado, se agrupa por el cod_cliente, se cuenta la cantidad de registros y si es mayor a 2 (condición del having) el cod_cliente forma parte del conjunto de resultados de la subconsulta.

En la consulta principal el cliente se mostrará si su cod_cliente está dentro del conjunto obtenido en la subconsulta

```
3. select cod_cliente,ape_cliente,nom_cliente
   from clientes c
   where exists (select count(*)
                 from facturas f
                 where year(fecha)=year(getdate())-1
                 and c.cod_cliente=f.cod_cliente
                 having count(*) >2
                 )
```

La tercer forma, con una consulta sumaria se cuentan los registros de facturas de este año del cliente que se quiere listar en la consulta principal y se aplica un having sobre el count de forma tal que si el cliente vino más de 2 veces dará un registro de resultado y si no lo hizo no da ningún registro de resultado.

```
4. select cod_cliente,ape_cliente,nom_cliente
   from clientes c
   where exists (select f.cod_cliente
                 from facturas f
                 where year(fecha)=year(getdate())-1
                 and c.cod_cliente=f.cod_cliente
                 group by f.cod_cliente
                 having count(*) >2
                 )
```

Por último, similar al anterior per utilizando una consulta agrupada.

4. **Se quiere saber qué clientes no vinieron entre el 12/12/2015 y el 13/7/2020**

```
select cod_cliente,ape_cliente, nom_cliente,nro_tel
from clientes
where cod_cliente not in (select cod_cliente
                          from facturas
                          where fecha between '12/12/2015' and '13/7/2020'
                          )

select cod_cliente,ape_cliente, nom_cliente,nro_tel
from clientes c
where not exists (select *
                  from facturas f
                  where fecha between '12/12/2015' and '13/7/2020'
                  and f.cod_cliente=c.cod_cliente
                  )
```

5. **Listar los datos de las facturas de los clientes que solo vienen a comprar en febrero es decir que todas las veces que vienen a comprar haya sido en el mes de febrero (y no otro mes).**

```
select *
from facturas f join clientes c on f.cod_cliente=c.cod_cliente
join detalle_facturas d on d.nro_factura=f.nro_factura
where 2 =all(select month(fecha)
             from facturas
             where cod_cliente=c.cod_cliente
             )
```

En este caso, la subconsulta lista todos los meses de las fechas de compra del cliente que se quiere listar en la consulta principal. En la consulta principal, con un test cuantificado ALL se comprueba que todos los meses sean iguales a 2.

Otra alternativa es que la subconsulta liste los cod_cliente de las facturas que no sean del mes 2 y la consulta principal debería listar las facturas de los clientes que no están incluidos en el resultado de la subconsulta, solución mostrada en el siguiente código.

```
select *
from facturas f
where cod_cliente not in (select cod_cliente
                          from facturas
                          where month(fecha)<>2
                          )
```

6. **Mostrar los datos de las facturas para los casos en que por año se hayan hecho menos de 9 facturas.**

```
Select f.nro_factura 'Número de factura',
       format(fecha, 'yyyy-MM-dd') Fecha,
       nom_cliente + ' ' + ape_cliente Cliente,
       nom_vendedor + ' ' + ape_vendedor Vendedor
from facturas f join clientes c on c.cod_cliente=f.cod_cliente
join vendedores v on v.cod_vendedor=f.cod_vendedor
where year(fecha) in (select year(fecha)
                      from facturas
                      group by year(fecha)
                      having count(*)<9
                      )
order by 2
```

En la primera solución, la subconsulta lista los años de las facturas (agrupadas por año) donde la cantidad de registros por año sea menor a 9. Luego la consulta principal deberá comprobar que el año de la factura que se quiere listar esté dentro de ese conjunto de años obtenidos en la subconsulta

```
--otra solución
Select f.nro_factura 'Número de factura',
       format(fecha, 'yyyy-MM-dd') Fecha,
       nom_cliente + ' ' + ape_cliente Cliente,
       nom_vendedor + ' ' + ape_vendedor Vendedor
from facturas f join clientes c on c.cod_cliente=f.cod_cliente
join vendedores v on v.cod_vendedor=f.cod_vendedor
where 9>(select count(*)
         from facturas f1
         where year(f1.fecha)=year(f.fecha)
         )
```

En cambio, esta otra solución, la subconsulta contará los registros de las facturas emitidas en el mismo año del año de la factura que se quiere listar en la consulta principal. En el where de la consulta principal, con un test de comparación verificará que el único valor de resultado de la subconsulta sea menor a 9.

7. **Emitir un reporte con las facturas cuyo importe total haya sido superior a 1.500 (incluir en el reporte los datos de los artículos vendidos y los importes).**

```
Select  f.nro_factura, FORMAT(fecha, 'dd/MM/yyyy')'Fecha',
        cantidad * pre_unitario 'Importe'
from facturas f
join detalle_facturas df on df.nro_factura=f.nro_factura
```

```
where 1500 <(select sum(cantidad * pre_unitario)
             from detalle_facturas d1
             where d1.nro_factura=f.nro_factura
             )
```

En la primera resolución, la subconsulta calcula el importe total de la factura que se quiere listar en la consulta principal. Se mostrará si el importe total de esa factura es mayor a 1500.

Las siguientes son otras propuestas de solución

```
select f.nro_factura,a.cod_articulo,descripcion,
cantidad*d.pre_unitario Importe
from detalle_facturas d join facturas f on d.nro_factura=f.nro_factura
join articulos a on d.cod_articulo=a.cod_articulo
where 1500<all(select sum(cantidad*pre_unitario)
               from detalle_facturas
               where nro_factura=f.nro_factura
               group by nro_factura
               )

select f.nro_factura,a.cod_articulo,descripcion,
cantidad*d.pre_unitario Importe
from detalle_facturas d join facturas f on d.nro_factura=f.nro_factura
join articulos a on d.cod_articulo=a.cod_articulo
where f.nro_factura in (select nro_factura
                        from detalle_facturas
                        group by nro_factura
                        having sum(cantidad*pre_unitario)>1500
                        )
```

8. Se quiere saber qué vendedores nunca atendieron a estos clientes: 1 y 6. Muestre solamente el nombre del vendedor.

```
Select ape_vendedor+' '+nom_vendedor Vendedor,
       calle+' '+cast(altura as varchar) Direccion,
       nro_tel Telefono, [e-mail] Email
from vendedores v
where cod_vendedor not in (select cod_vendedor
                           from facturas
                           where cod_cliente in (1,6)
                           )
```

En la primera solución, la subconsulta lista los cod_vendedor de la tabla facturas cuyos cod_cliente son 1 y 6 y la consulta principal va a mostrar el vendedor que no esté dentro del conjunto de cod_vendedor que devuelve la subconsulta

Seguidamente se muestra la resolución utilizando el test NOT EXISTS

```
Select ape_vendedor+' '+nom_vendedor Vendedor,
       calle+' '+cast(altura as varchar) Direccion,
       nro_tel Telefono, [e-mail] Email
from vendedores v
where not exists(select *
                 from facturas
                 where cod_vendedor = v.cod_vendedor
                 and cod_cliente in (1,6)
                 )
```

9. **Listar los datos de los artículos que superaron el promedio del Importe de ventas de $ 1.000.**

```
select a.cod_articulo, descripcion
from articulos a
where 1000 <(select avg(cantidad*pre_unitario)
             from detalle_facturas d
             where a.cod_articulo=d.cod_articulo)
```

10. **¿Qué artículos nunca se vendieron? Tenga además en cuenta que su nombre comience con letras que van de la "d" a la "p". Muestre solamente la descripción del artículo.**

```
select a.descripcion
from articulos a
where a.cod_articulo NOT IN (select distinct cod_articulo
                             from detalle_facturas)
```

11. **Listar número de factura, fecha y cliente para los casos en que ese cliente haya sido atendido alguna vez por el vendedor de código 3.**

```
Select nro_factura, fecha,
       c.nom_cliente+' '+c.ape_cliente 'Nombre Completo'
from facturas f join clientes c on c.cod_cliente=f.cod_cliente
where 3=any(select cod_vendedor
            from facturas f1
            where f1.cod_cliente=f.cod_cliente)
```

12. **Listar número de factura, fecha, artículo, cantidad e importe para los casos en que todas las cantidades (de unidades vendidas de cada artículo) de esa factura sean superiores a 40.**

```
select d.nro_factura,fecha,cod_cliente,cod_articulo,cantidad
from facturas f join detalle_facturas d on d.nro_factura=f.nro_factura
where 40<all(select cantidad
             from detalle_facturas d1
             where d1.nro_factura=d.nro_factura)
```

13. **Emitir un listado que muestre número de factura, fecha, artículo, cantidad e importe; para los casos en que la cantidad total de unidades vendidas sean superior a 80.**

```
Select d.nro_factura,fecha,cod_articulo,cantidad,
       cantidad*pre_unitario importe
from facturas f join detalle_facturas d on d.nro_factura=f.nro_factura
where 80<(select sum(cantidad)
          from detalle_facturas d1
          where d1.nro_factura=d.nro_factura
          )
order by 1
```

14. **Realizar un listado de número de factura, fecha, cliente, artículo e importe para los casos en que al menos uno de los importes de esa factura sea menor a 3.000.**

```
Select f.nro_factura,fecha,cod_cliente,cod_articulo,descripcion
       cantidad*d.pre_unitario Importe
from detalle_facturas d join facturas f on d.nro_factura=f.nro_factura
join artículos a on a.cod_articulo=d.cod_articulo
where 3000>any(select cantidad*d1.pre_unitario
               from detalle_facturas d1
               where d1.nro_factura=f.nro_factura
               )
```

Problema 2.2: Subconsultas en el Having

Los usuarios finales del sistema en esta oportunidad necesitan obtener los siguientes reportes de información para el funcionamiento del negocio y la toma de decisiones:

1. **Se quiere saber ¿cuándo realizó su primer venta cada vendedor? y ¿cuánto fue el importe total de las ventas que ha realizado? Mostrar estos**

datos en un listado solo para los casos en que su importe promedio de vendido sea superior al importe promedio general (importe promedio de todas las facturas).

2. **Liste los montos totales mensuales facturados por cliente y además del promedio de ese monto y el promedio de precio de artículos Todos esto datos correspondientes a período que va desde el 1° de febrero al 30 de agosto del 2014. Sólo muestre los datos si esos montos totales son superiores o iguales al promedio global.**

3. **Por cada artículo que se tiene a la venta, se quiere saber el importe promedio vendido, la cantidad total vendida por artículo, para los casos en que los números de factura no sean uno de los siguientes: 2, 10, 7, 13, 22 y que ese importe promedio sea inferior al importe promedio de ese artículo.**

4. **Listar la cantidad total vendida, el importe y promedio vendido por fecha, siempre que esa cantidad sea superior al promedio de la cantidad global. Rotule y ordene.**

5. **Se quiere saber el promedio del importe vendido y la fecha de la primer venta por fecha y artículo para los casos en que las cantidades vendidas oscilen entre 5 y 20 y que ese importe sea superior al importe promedio de ese artículo.**

6. **Emita un listado con los montos diarios facturados que sean inferior al importe promedio general.**

7. **Se quiere saber la fecha de la primera y última venta, el importe total facturado por cliente para los años que oscilen entre el 2010 y 2015 y que el importe promedio facturado sea menor que el importe promedio total para ese cliente.**

8. **Realice un informe que muestre cuánto fue el total anual facturado por cada vendedor, para los casos en que el nombre de vendedor no comience con ‘B’ ni con ‘M’, que los números de facturas oscilen entre 5 y 25 y que el promedio del monto facturado sea inferior al promedio de ese año.**

Problema 2.3: Otras Subconsultas

1. **Se quiere listar el precio de los artículos y la diferencia de éste con el precio del artículo más caro:**
2. **Listar el precio actual de los artículos y el precio histórico vendido más barato**
3. **Se quiere emitir un listado de las facturas del año en curso detallando número de factura, cliente, fecha y total de la misma.**
4. **Emitir un listado con la código y descripción de los artículos su precio actual, el precio promedio al cuál se vendió el año pasado (ver diferencia entre el promedio ponderado y el promedio simple)**
5. **Generar un reporte un listado con la código y descripción de los artículos su precio actual, el precio más barato y el más caro al que se vendió hace 5 años.**
6. **Descontar un 3,5% los precios de los artículos que se vendieron menos de 5 unidades los últimos 3 meses.**
7. **Se quiere eliminar los clientes que no vinieron nunca.**
8. **Eliminar los clientes que hace más de 10 años que no vienen**

BIBLIOGRAFÍA

Gorman K., Hirt A., Noderer D., Rowland-Jones J., Sirpal A., Ryan D. & Woody B (2019) Introducing Microsoft SQL Server 2019. Reliability, scalability, and security both on premises and in the cloud. Packt Publishing Ltd. Birmingham UK

Microsoft (2021) SQL Server technical documentation. Disponible en: https://docs.microsoft.com/en-us/sql/sql-server/?view=sql-server-ver15

Opel, A. & Sheldon, R. (2010). Fundamentos de SQL. Madrid. Editorial Mc Graw Hill

Varga S., Cherry D., D'Antoni J. (2016). Introducing Microsoft SQL Server 2016 Mission-Critical Applications, Deeper Insights, Hyperscale Cloud. Washington. Microsoft Press