

# Consultas Avanzadas con Algebra Relacional: Clientes, Productos y Pedidos

---

Camilo Esteban Paez - 1110450891

## 1. Identificación de Clientes según Método de Pago

```
select distinct c.Nombre, c.Telefono from Clientes c
inner join Pedidos p
on p.ClienteID = c.ClienteID
inner join MetodoPago m
on m.MetodoPagoID = p.MetodoPagoID
where m.descripcion = 'Efectivo';
```

	Nombre	Telefono
1	María Gómez	3129876543
2	Luisa López	3203456789
3	Ana Torres	3145678901

Ya normalizada la base es necesario realizar 2 inner joins (intersección entre ambas tablas) para poder determinar el método de pago, Cliente y Pedidos luego MetodoPago y Pedidos, si conociéramos el ID de antemano sería más veloz, se usa la keyword **distinct** para que se seleccionen únicamente las entradas que sean únicas y usando **where** discriminamos aquellas entradas que no equivalgan a 'Efectivo'.

## 2. Determinación de Productos No Solicitados

```
select p.Nombre from Productos p
left join Pedidos pe
on p.ProductoID = pe.ProductoID
where pe.ProductoID is null;
```

Query 1/1: No Results. 0 rows affected. See the select box in the bottom left ✓ for more query results.

Es importante entender el uso de `left join` en esta consulta, si hicieramos un `inner join` perderiamos aquellos productos que no han sido solicitados, justamente lo opuesto a lo que es requerido, por lo tanto el `left join` mantiene la informacion de la "izquierda" (Productos) y lo une con la informacion coincidente de la "derecha" (Pedidos) o `null`.

### 3. Renombramiento y Filtrado de Datos en "Pedidos"

```
select * from Pedidos as Compras
where Compras.Fecha = '2024-12-01';
```

	PedidoID ▲	ClienteID ▲	ProductoID ▲	Cantidad ▲	Fecha ▲	MetodoPagoID ▲
1	1	1	1	2	2024-12-01	1
2	2	1	2	1	2024-12-01	1
3	3	1	3	1	2024-12-01	1

La palabra clave `as` (alias) se utiliza para renombrar de forma temporal una columna o tabla en este caso.

### 4. Análisis de Frecuencia de Productos Solicitados

```
select p.Nombre from Productos p
join Pedidos pe
on p.ProductoID = pe.ProductoID
group by Nombre
having sum(pe.Cantidad) > 1;
```

	Nombre ▲
1	Zapatos
2	Camisa
3	Pantalón

Para realizar esta consulta usamos **group by** que agrupa por nombre y nos permite realizar operacion como **SUM()** o **AVG()**, se usa **having** debido a que la implementacion de **where** no funciona con datos agrupados.

## 5. Clientes que Adquirieron Productos de una Categoría Especifica

```
select distinct c.Nombre from Clientes c
inner join Pedidos pe
on pe.ClienteID = c.ClienteID
inner join Productos pr
on pe.ProductoID = pr.ProductoID
inner join Categoria ca
on pr.CategoriaID = ca.CategoriaID
where ca.descripcion = 'Ropa';
```

	Nombre ▲
1	Juan Pérez
2	Carlos Ruiz
3	Pedro Martínez
4	Ana Torres
5	Luisa López
6	Jorge Castillo

¿Por que razon tantos **inner join**, al ya estar normalizada la base de datos **Productos** no contiene la descripcion y es necesario un join mas para obtener los resultados, primero se seleccionan los clientes que han hecho pedidos, luego sobre esos pedidos se obtiene la informacion del producto y por ultimo de esos productos se obtiene la informacion de su categoria, si es igual a 'Ropa' se selecciona el nombre.