**DEFINICIONES:**

**SELECT:**

La sentencia SELECT se utiliza para seleccionar datos de una base de datos.

Los datos devueltos se almacenan en una tabla de resultados, denominada conjunto de resultados.

**WHERE:**

La cláusula WHERE se utiliza para filtrar registros.

Sirve para extraer sólo los registros que cumplen una condición especificada.

Sirve para una variable.

**ORDER BY:**

La palabra clave ORDER BY se utiliza para ordenar el conjunto de resultados en orden ascendente o descendente.

Por defecto, la palabra clave ORDER BY ordena los registros en orden ascendente. Para ordenar los registros en orden descendente, utilice la palabra clave DESC.

**LIKE:**

El operador LIKE se utiliza en una cláusula WHERE para buscar un patrón especificado en una columna.

Hay dos comodines que se utilizan a menudo junto con el operador LIKE:

El signo de porcentaje (%) representa cero, uno o varios caracteres

El guión bajo (\_) representa un único carácter.

El signo de porcentaje y el guión bajo también pueden utilizarse en combinaciones.

**IN:**

El operador IN permite especificar varios valores en una cláusula WHERE.

El operador IN es una abreviatura de las condiciones OR múltiples.

**GROUP BY:**

La sentencia GROUP BY agrupa filas que tienen los mismos valores en filas resumen, como "encontrar el número de clientes en cada país".

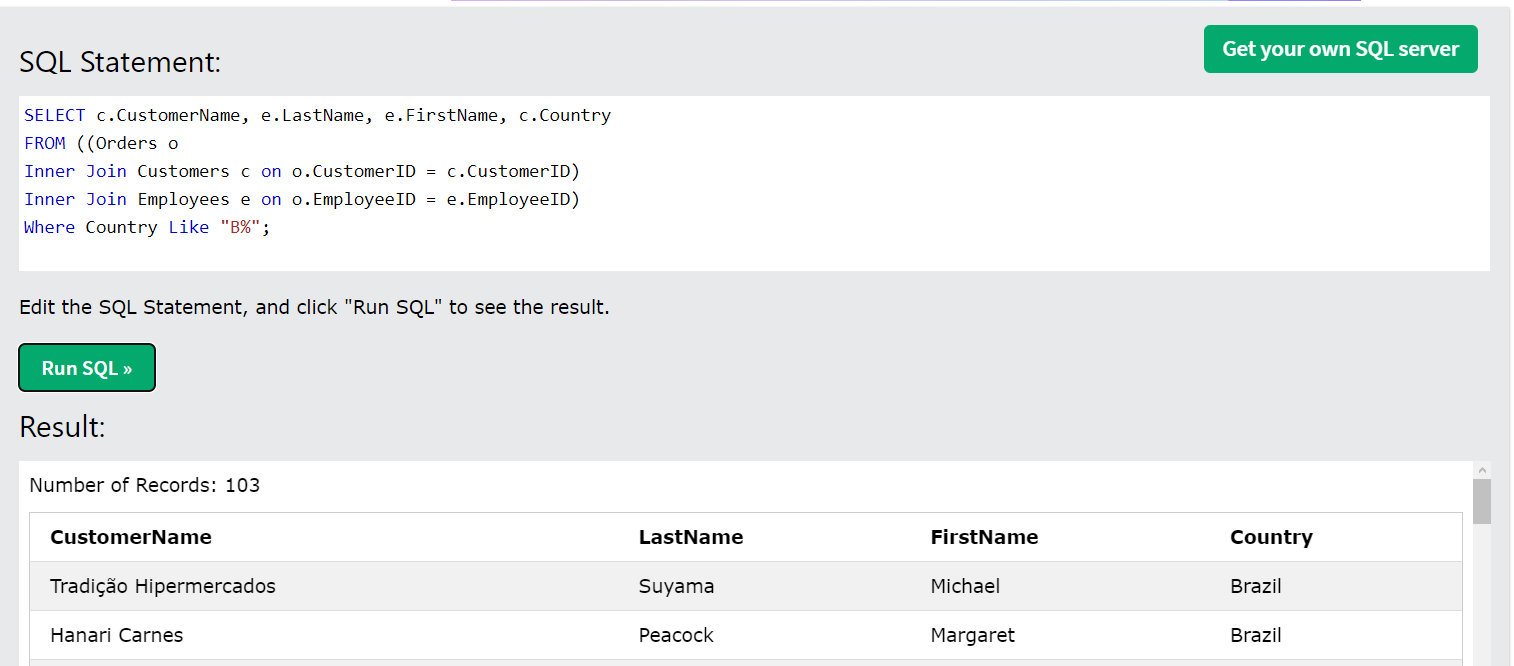
La sentencia GROUP BY suele utilizarse con funciones de agregación (COUNT(), MAX(), MIN(), SUM(), AVG()) para agrupar el conjunto de resultados por una o varias columnas.

**HAVING:**

La cláusula HAVING se añadió a SQL porque la palabra clave WHERE no puede utilizarse con funciones agregadas.

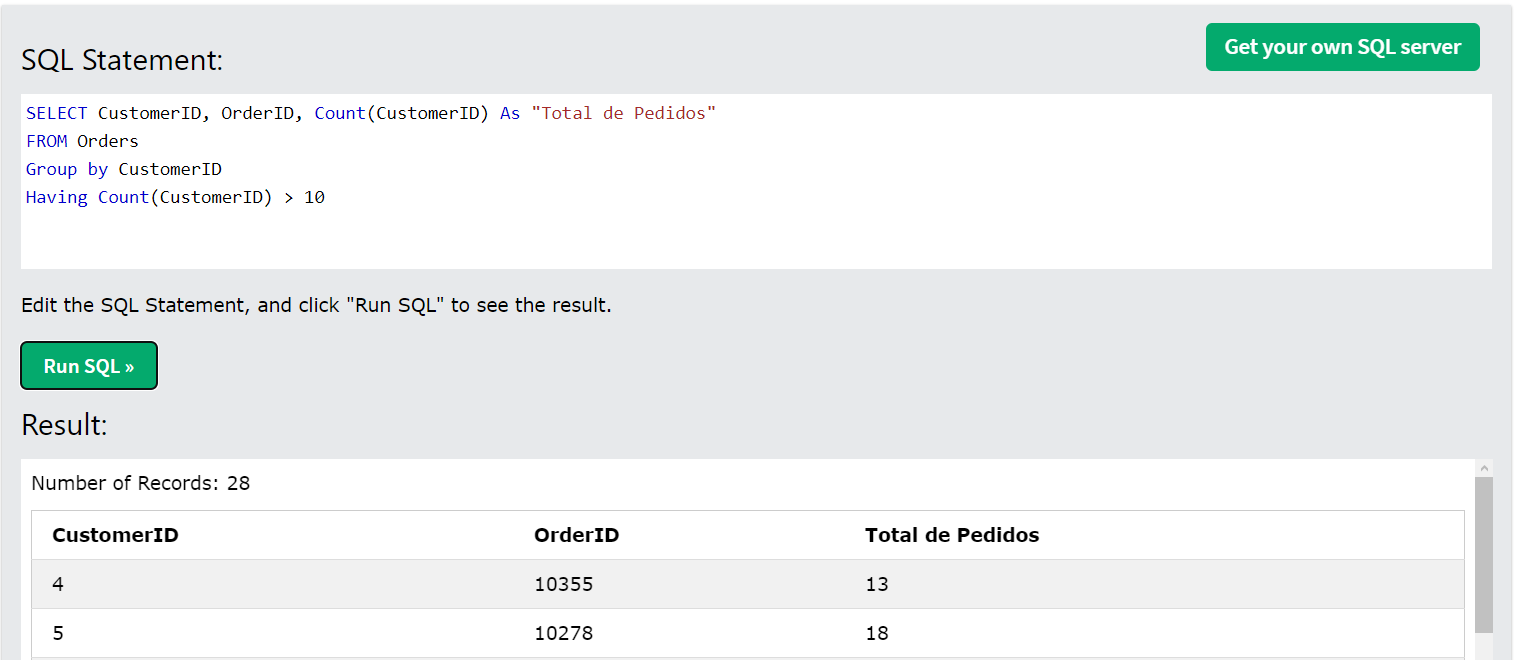
Sirve para mas de una variable a utilizar.

1. Obtén una lista de todos los pedidos que incluya el nombre del cliente, el nombre del empleado que lo atendió, y el país de destino, pero solo para los pedidos enviados a ciudades que comiencen con la letra 'B'.



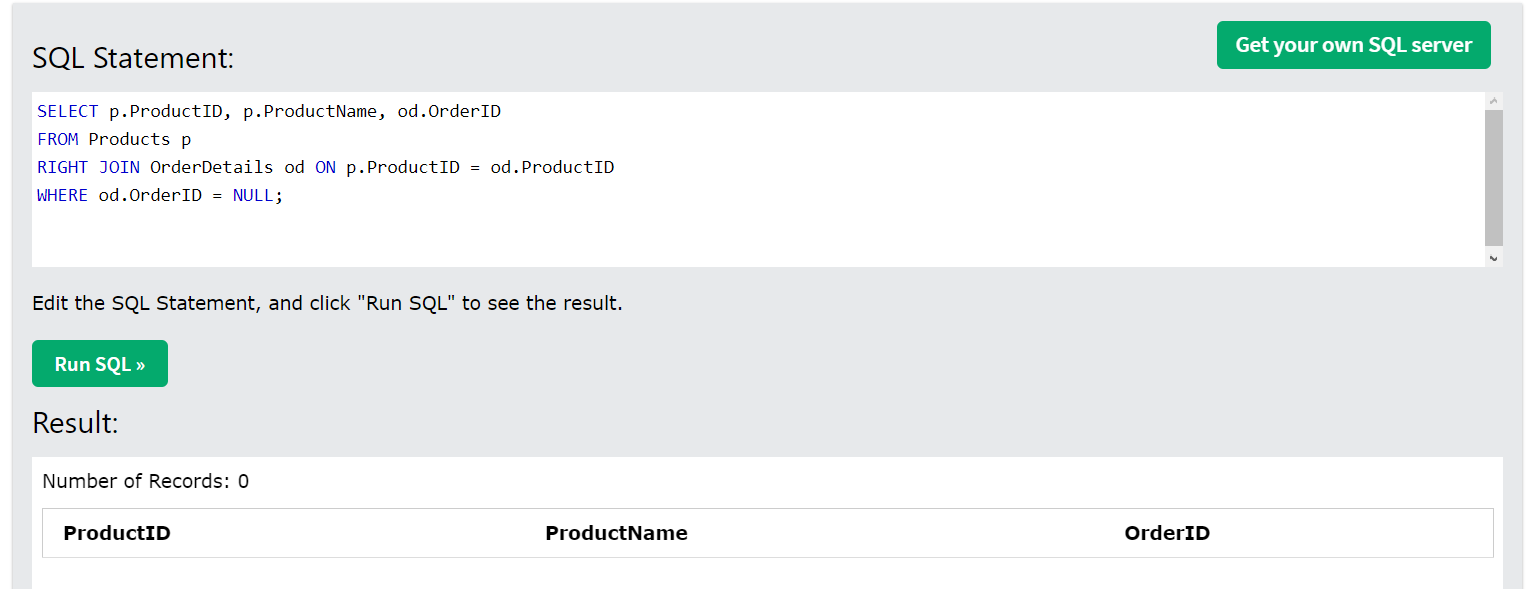
|  |
| --- |
| SELECT c.CustomerName, e.LastName, e.FirstName, c.Country  FROM ((Orders o  Inner Join Customers c on o.CustomerID = c.CustomerID)  Inner Join Employees e on o.EmployeeID = e.EmployeeID)  Where Country Like "B%"; |

2. Encuentra los clientes que han realizado más de 10 pedidos, incluyendo la cantidad total de pedidos realizados, utilizando una subconsulta para calcular la cantidad de pedidos primero.



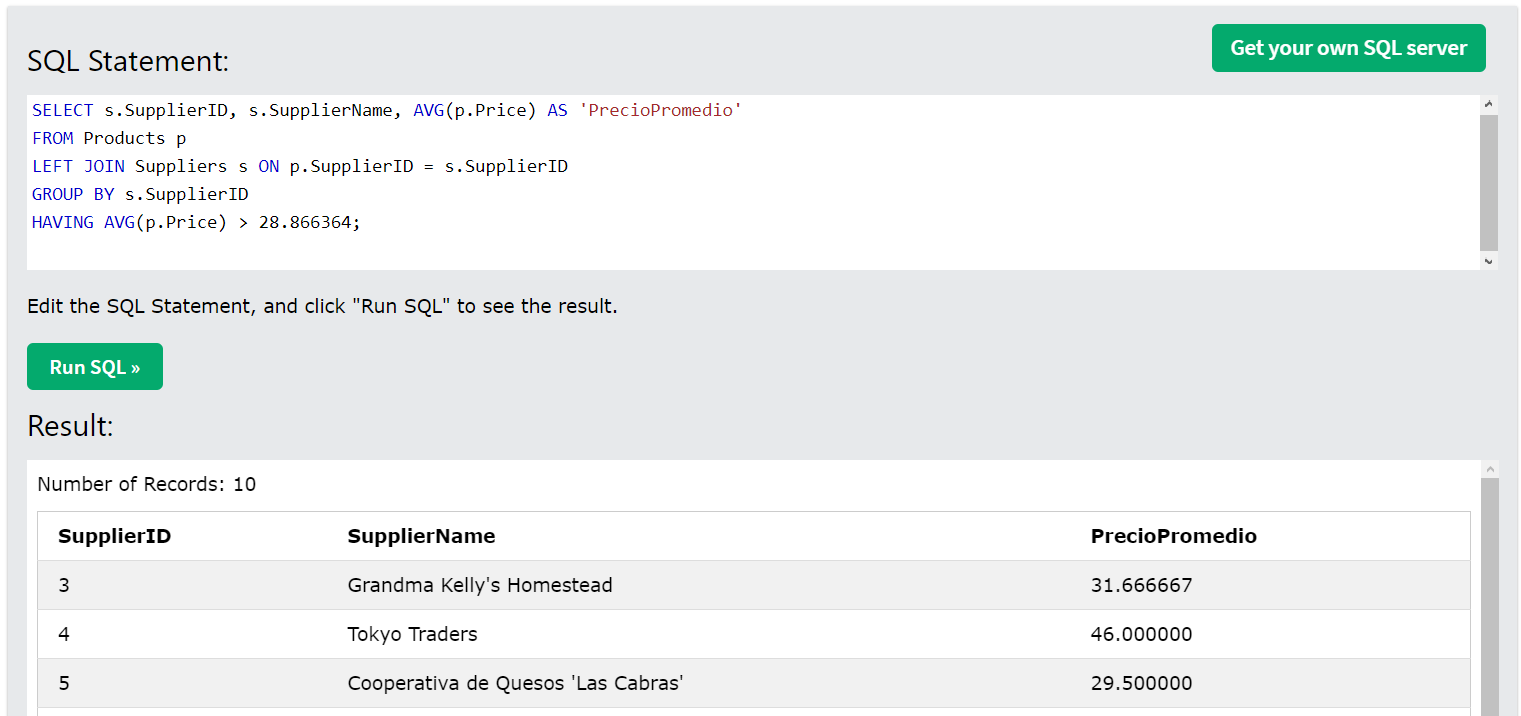
|  |
| --- |
| SELECT CustomerID, OrderID, Count(CustomerID) As "Total de Pedidos"  FROM Orders  Group by CustomerID  Having Count(CustomerID) > 10 |

* Muestra todos los productos que nunca se han vendido, utilizando un RIGHT JOIN entre Order Details y Products y filtrando por aquellos que tienen NULL en la columna de OrderID.



|  |
| --- |
| SELECT p.ProductID, p.ProductName, od.OrderID  FROM Products p  RIGHT JOIN OrderDetails od ON p.ProductID = od.ProductID  WHERE od.OrderID = NULL; |

3. Selecciona los proveedores cuyos productos tienen un precio promedio superior al costo promedio de todos los productos, agrupando por proveedor y filtrando con HAVING.



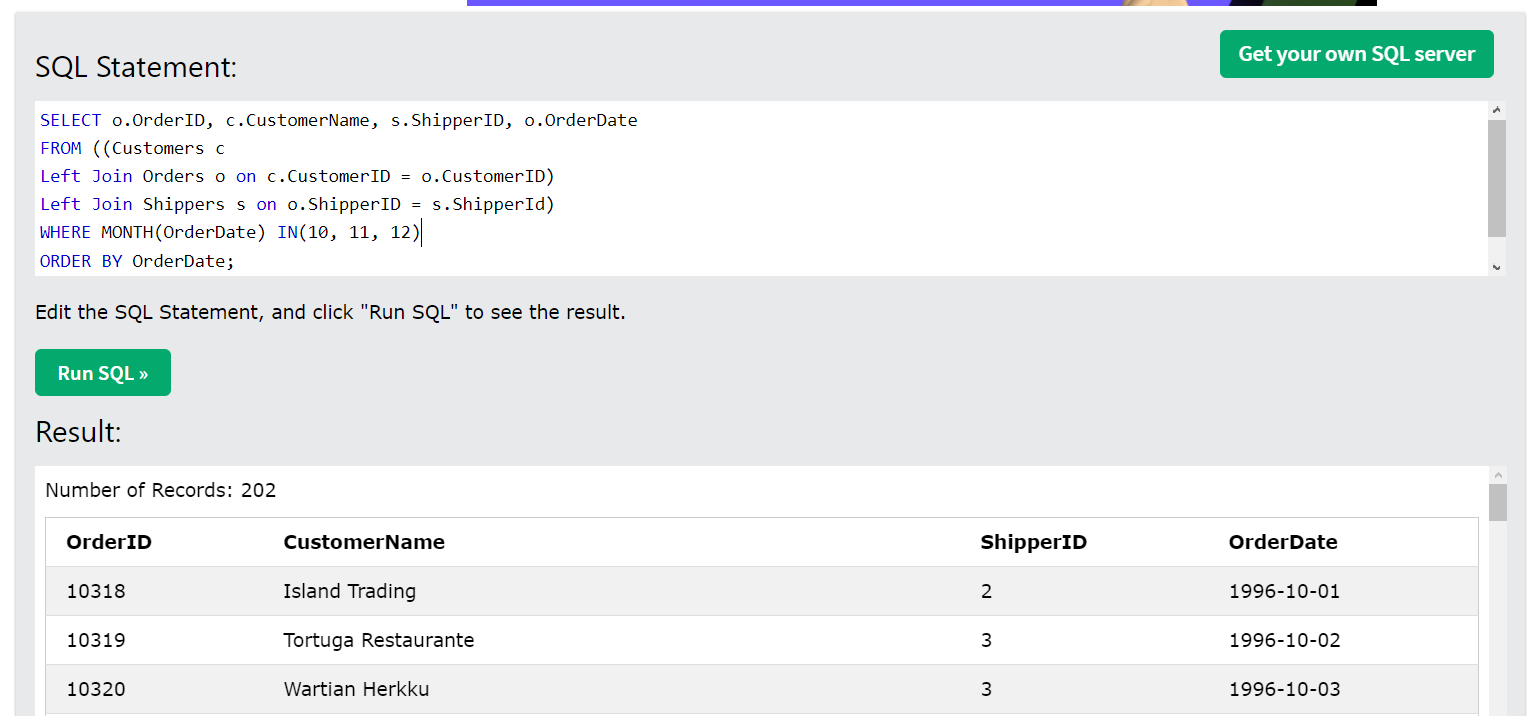
|  |
| --- |
| SELECT s.SupplierID, s.SupplierName, AVG(p.Price) AS “PrecioPromedio”  FROM Products p  LEFT JOIN Suppliers s ON p.SupplierID = s.SupplierID  GROUP BY s.SupplierID  HAVING AVG(p.Price) > 28.866364; |

4. Lista los empleados junto con el nombre de su jefe directo (suponiendo una jerarquía en los empleados) y la cantidad de pedidos que han procesado, ordenados por la cantidad de pedidos descendente.



|  |
| --- |
| ALTER TABLE Employees  ADD Jefe Varchar(250)  SELECT o.EmployeeID, COUNT(o.OrderID) AS “Cantidad de Pedidos”, e.Jefe  FROM Employees e  LEFT JOIN Orders o ON e.EmployeeID = o.EmployeeID  GROUP BY o.EmployeeID |

5. Presenta una lista de todos los envíos que incluya información del pedido y la fecha de envío para los pedidos realizados en el último trimestre, ordenados por fecha de envío.



|  |
| --- |
| SELECT o.OrderID, c.CustomerName, s.ShipperID, o.OrderDate  FROM ((Customers c  Left Join Orders o on c.CustomerID = o.CustomerID)  Left Join Shippers s on o.ShipperID = s.ShipperId)  WHERE MONTH(OrderDate) IN(10, 11, 12)  ORDER BY OrderDate; |

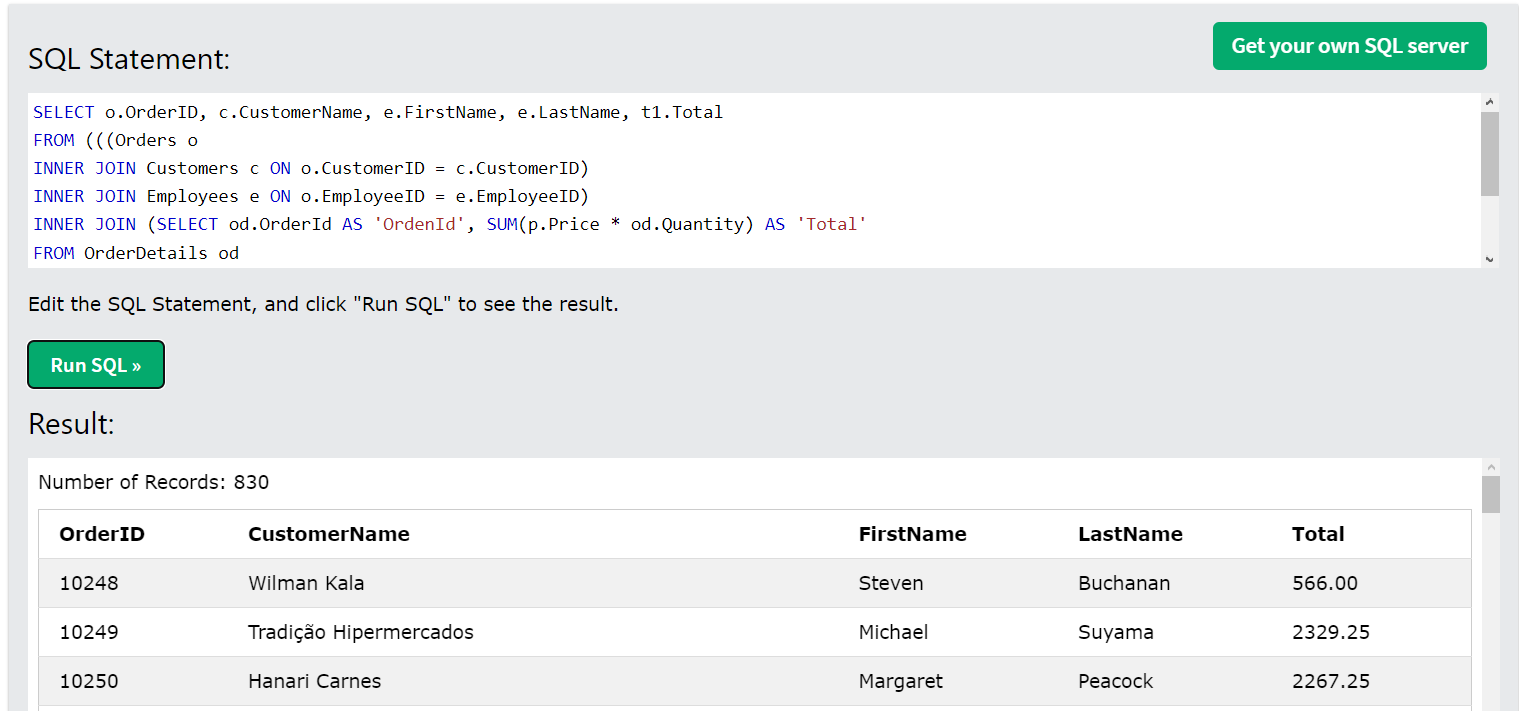
S

Encuentra los productos que tienen ventas (cantidad de orden) superiores al promedio de ventas de su categoría.



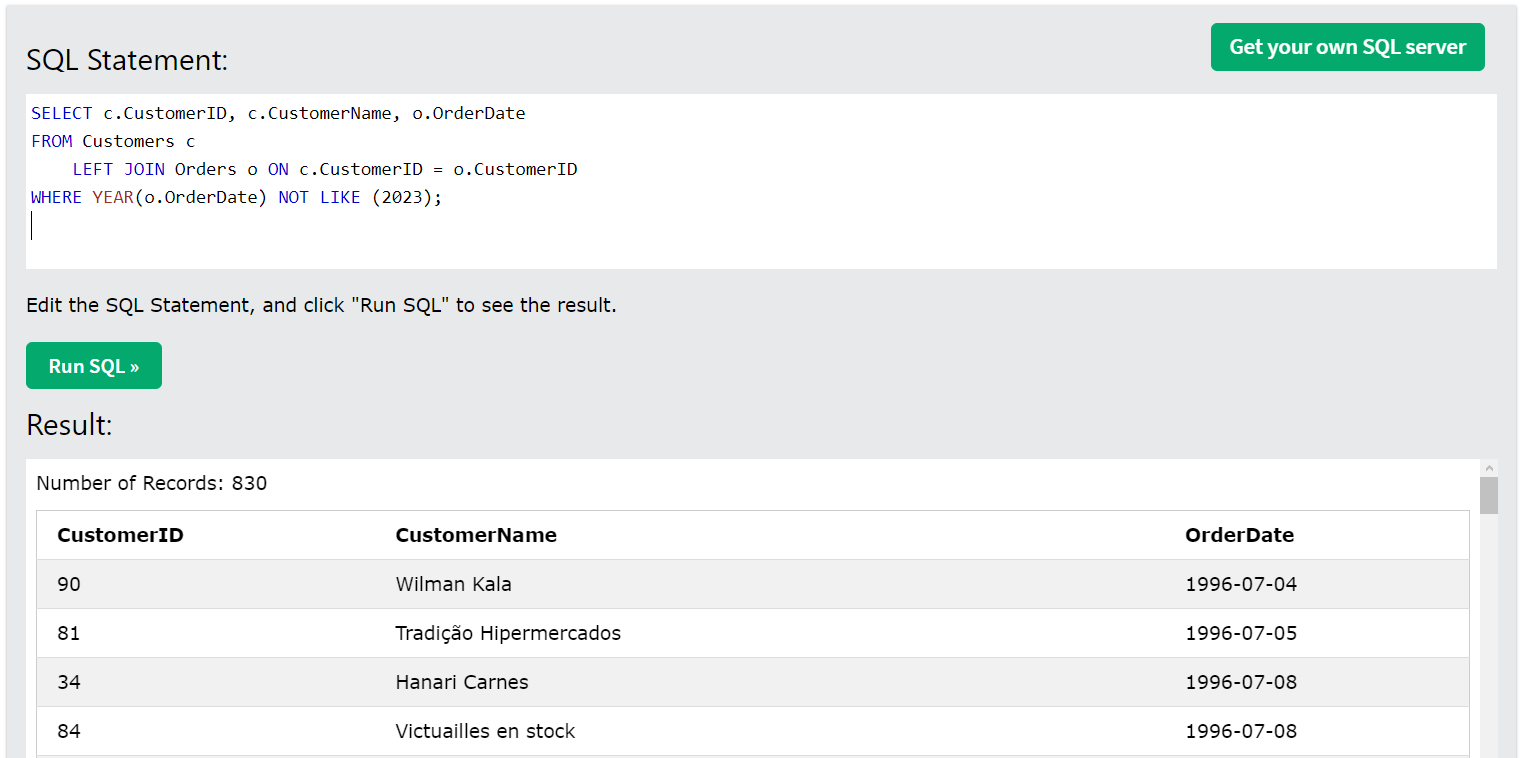
|  |
| --- |
| SELECT  Products.ProductID,  Products.ProductName,  Products.CategoryID,  COUNT(Orders.OrderID) as ‘Cantidad’  FROM Products  INNER JOIN OrderDetails ON Products.ProductID = OrderDetails.ProductID  INNER JOIN Orders ON OrderDetails.OrderID = Orders.OrderID  GROUP BY Products.ProductID, Products.ProductName, Products.CategoryID  HAVING COUNT(Orders.OrderID) >  ( SELECT AVG(SalesCountPerProduct) FROM ( SELECT Products.CategoryID, COUNT(Orders.OrderID) as SalesCountPerProduct FROM Products INNER JOIN OrderDetails ON Products.ProductID = OrderDetails.ProductID INNER JOIN Orders ON OrderDetails.OrderID = Orders.OrderID GROUP BY  Products.CategoryID, Orders.OrderID ) AS CategorySales) |

6. Para cada pedido, muestra el cliente, el empleado que lo tomó, y el total de la orden calculado como una subconsulta en la selección.



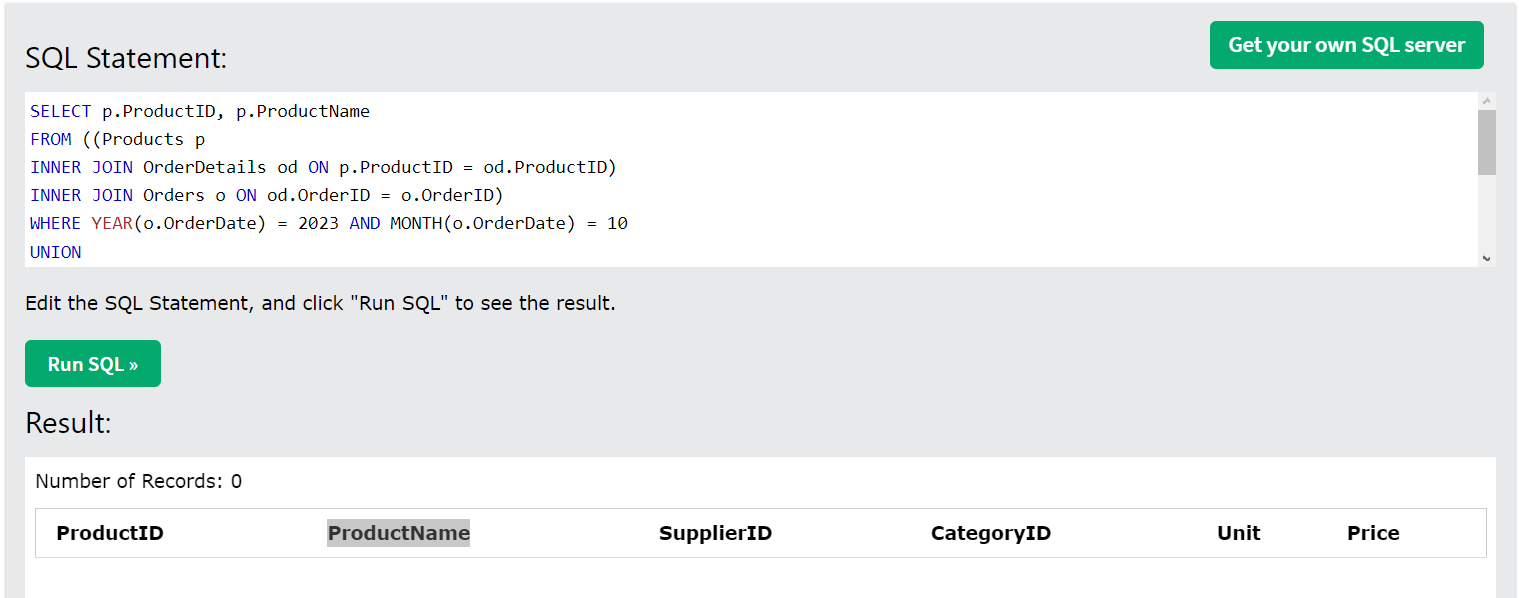
|  |
| --- |
| SELECT  o.OrderID,  c.CustomerName,  e.FirstName,  e.LastName,  t1.Total  FROM (((Orders o  INNER JOIN Customers c ON o.CustomerID = c.CustomerID)  INNER JOIN Employees e ON o.EmployeeID = e.EmployeeID)  INNER JOIN (SELECT od.OrderId AS 'OrdenId', SUM(p.Price \* od.Quantity) AS 'Total'  FROM OrderDetails od  INNER JOIN Products p ON od.ProductID = p.ProductID  GROUP BY od.OrderID) t1 ON o.OrderID = t1.OrdenId)  GROUP BY o.OrderID; |

7. Lista los clientes que han realizado pedidos pero no en el año actual, utilizando un LEFT JOIN entre Customers y Orders y filtrando por año en la cláusula WHERE.



|  |
| --- |
| SELECT c.CustomerID, c.CustomerName, o.OrderDate  FROM Customers c  LEFT JOIN Orders o ON c.CustomerID = o.CustomerID  WHERE YEAR(o.OrderDate) NOT LIKE (2023); |

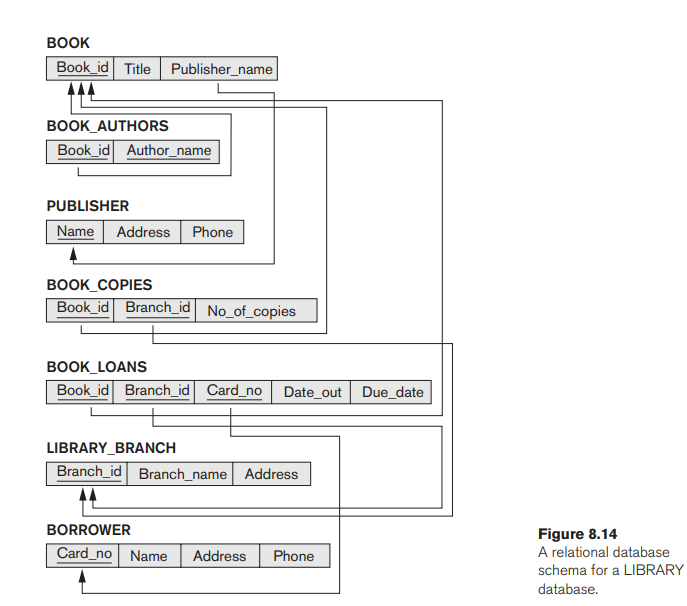
8. Realiza una consulta que combine resultados de dos SELECT diferentes usando UNION o UNION ALL, donde cada SELECT tiene múltiples JOINs, para encontrar productos que han sido ordenados este mes y el mes pasado.



|  |
| --- |
| SELECT p.ProductID, p.ProductName  FROM ((Products p  INNER JOIN OrderDetails od ON p.ProductID = od.ProductID)  INNER JOIN Orders o ON od.OrderID = o.OrderID)  WHERE YEAR(o.OrderDate) = 2023 AND MONTH(o.OrderDate) = 10  UNION  SELECT p.ProductID, p.ProductName  FROM ((Products p  INNER JOIN OrderDetails od ON p.ProductID = od.ProductID)  INNER JOIN Orders o ON od.OrderID = o.OrderID)  WHERE YEAR(o.OrderDate) = 2023 AND MONTH(o.OrderDate) = 11  ORDER BY ProductID; |

**PARTE 2:**

**Ejercicio 1:**



**8.18. Considere el esquema de base de datos relacional BIBLIOTECA que se muestra en la Figura 8.14, que se utiliza para llevar un registro de libros, prestatarios y préstamos de libros. Las restricciones de integridad referencial se muestran como arcos dirigidos en la Figura 8.14, igual que en la notación de la Figura 5.7. Escriba expresiones relacionales para las siguientes consultas:**

* **¿Cuántos ejemplares del libro titulado La tribu perdida posee la sucursal de la biblioteca cuyo nombre es "Sharpstown"?**

π No\_of\_copies (σ Branch\_name = “Sharpstown” AND Title = “The Lost Tribe” (BOOK\_COPIES⋈BOOK))

* **¿Cuántos ejemplares del libro La tribu perdida tiene cada sucursal de la biblioteca?**

π Branch\_name, No\_of\_copies (σ Title = ’The Lost Tribe’ (BOOK\_COPIES⋈BOOK))

* **Obtén los nombres de todos los prestatarios que no tienen ningún libro prestado.**

π Name (σ Card\_no = NULL (BORROWER))

* **Para cada libro prestado por la sucursal de Sharpstown cuya fecha de vencimiento es hoy, obtenga el título del libro, el nombre del prestatario y su dirección.**

π Title, Borrower\_name, Borrower\_address​(σ Branch\_name=’Sharpstown’ AND Due\_date= “22/11/2023” ​(BOOK ⋈ BORROW ⋈ BOOK\_LOANS))

* **Para cada sucursal de la biblioteca, obtenga el nombre de la sucursal y el número total de libros prestados desde esa sucursal.**

*π* Branch\_name, Total\_Books\_Loaned ​(*ρ (*Total\_Books\_Loaned) ← SUM (No\_of\_copies)​(BORROWER ⋈ Book\_id = Book\_id​ BOOK\_LOANS ⋈ BOOK\_COPIES))

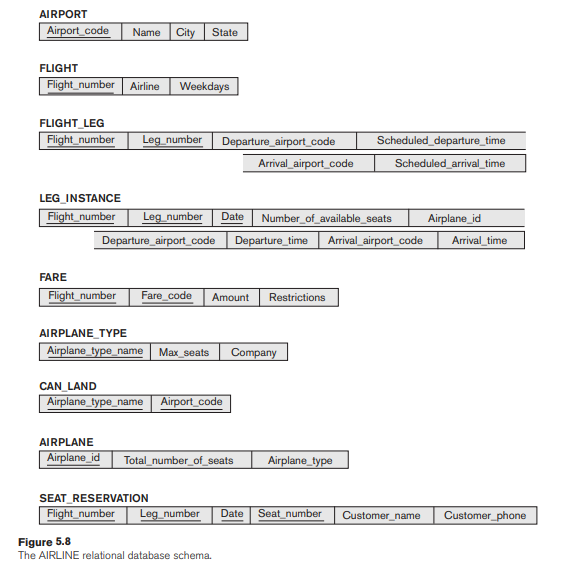
* **Obtenga los nombres, direcciones y número de libros prestados de todos los prestatarios que tengan más de cinco libros prestados.**

*π* Borrower\_name, Address, Books\_Checked\_Out ​(*σ (*Books\_Checked\_Out>5)​*ρ (*Books\_Checked\_Out) ← COUNT(Book\_id)​) (BORROWER ⋈ Book\_id = Book\_id ​BOOK\_LOANS))

* **Para cada libro cuyo autor (o coautor) es Stephen King, obtenga el título y el número de ejemplares que posee la sucursal de la biblioteca cuyo nombre es Central.**

*π* Title, Copies\_Owned ​(*ρ (*Copies\_Owned←COUNT(Book\_id) (*σ (*Author\_name=’Stephen King’)​(BOOK ⋈ BOOK\_COPIES) ⋈ Branch\_name = ’Central’​LIBRARY\_BRANCH))

**Ejercicio 2:**

****

**8.17. Considere el esquema de base de datos relacional AIRLINE mostrado en la Figura 5.8, que se describió en el Ejercicio 5.12. Especifique las siguientes consultas en álgebra relacional:**

* **Para cada vuelo, indique el número de vuelo, el aeropuerto de salida para el primer tramo del vuelo y el aeropuerto de llegada para el último tramo del vuelo.**

*π* flight\_number, Departure\_airport\_code, Arrival\_airport\_code​ (FLIGHT ⋈ flight\_number = flight\_number *(σ (*leg\_number = 1)​(LEG\_INSTANCE) ⋈ flight\_number =  flight\_number))

* **Enumere los números de vuelo y los días de la semana de todos los vuelos o tramos de vuelo que salen del Aeropuerto Intercontinental de Houston (código de aeropuerto "iah") y llegan al Aeropuerto Internacional de Los Ángeles (código de aeropuerto "lax").**

*π* flight\_number, Date ​(*σ (*departure\_airport\_code  =  ’iah’) and (arrival\_airport\_code  =  ’lax’) ​(FLIGHT ⋈ flight\_no  =  flight\_no​ (LEG\_INSTANCE))

* **Indique el número de vuelo, el código del aeropuerto de salida, la hora prevista de salida, el código del aeropuerto de llegada, la hora prevista de llegada y los días de la semana de todos los vuelos o tramos de vuelo que salgan de algún aeropuerto de la ciudad de Houston y lleguen a algún aeropuerto de la ciudad de Los Ángeles.**

*π* flight\_ number, departure\_airport\_code, scheduled \_departure\_time, arrival\_airport\_code, scheduled \_arrival \_time, Date ​ (*σ (*city = ’Houston’) and (city = ’Los Angeles’) ​(FLIGHT ⋈ flight\_ number =  flight\_ number ​LEG\_INSTANCE ⋈ leg\_ number = 1 ​))

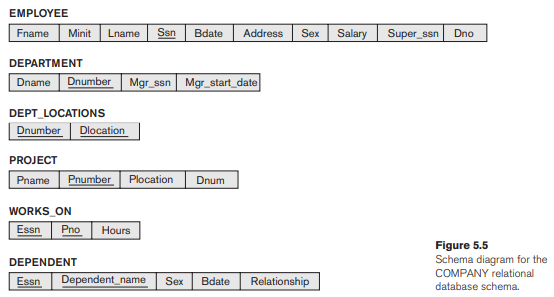
* **Listar toda la información sobre tarifas para el vuelo "co197".**

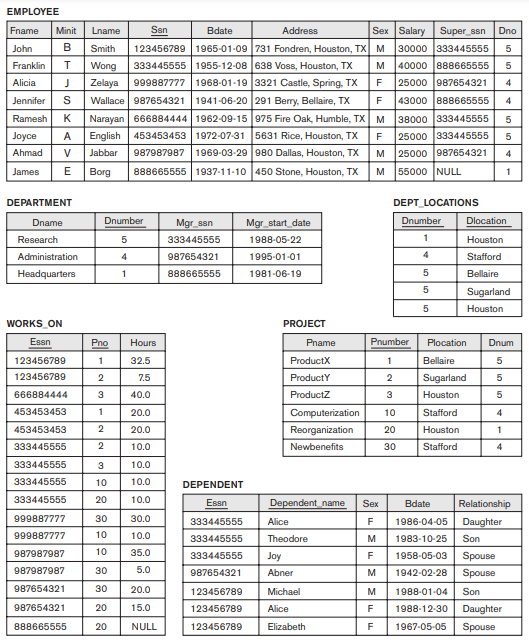
*σ (*flight\_number = ’co197’) ​(FARE)

* **Obtener el número de plazas disponibles para el vuelo "co197" el día '2009-10-09'.**

*π* Number\_of\_available\_seats(*σ (*flight\_number = ’co197’) and Date = ’2009-10-09’​)(LEG\_INSTANCE))

**Ejercicio 3:**

****

****

**8.16. Especifique las siguientes consultas en el esquema de base de datos relacional EMPRESA mostrado en la Figura 5.5 utilizando los operadores relacionales discutidos en este capítulo. Muestre también el resultado de cada consulta tal y como se aplicaría al estado de la base de datos de la Figura 5.6.**

* **Obtenga los nombres de todos los empleados del departamento 5 que trabajan más de 10 horas semanales en el proyecto ProductX.**

*π (Fname)( σ (Dno = 5) AND (Hours > 10) AND (Pname = ‘ProductX’) (EMPLOYEE* ⋈ *WORS\_ON* ⋈ PROJECT*))*

* **Enumerar los nombres de todos los empleados que tienen un dependiente con el mismo nombre que ellos.**

*π (Fname)( σ (Dependent\_name = Fname)(EMPLOYEE* ⋈ DEPENDENT *))*

* **Encuentre los nombres de todos los empleados que son supervisados directamente por "Franklin Wong".**

*π (Fname)( σ (Fname = ‘Franklin’ AND Lname = ‘Wong’)(EMPLOYEE))*

* **Para cada proyecto, enumere el nombre del proyecto y el total de horas semanales (de todos los empleados) dedicadas a ese proyecto.**

*π (Pname, Total) ( ρ (*Total) ← SUM(Hours)(PROJECT ⋈ WORKS\_ON)*)*

* **Obtenga los nombres de todos los empleados que trabajan en cada proyecto.**

*π (Fnamel, P\_name) (σ (Essn = Ssn)(EMPLOYEE* ⋈ *WORKS\_ON))*

* **Obtenga los nombres de todos los empleados que no trabajan en ningún proyecto.**

*π (Fnamel) (π (Fnamel)(EMPLOYEE) - π (Fnamel) (EMPLOYEE⋈WORKS\_ON))*

* **Para cada departamento, obtenga el nombre del departamento y el salario medio de todos los empleados que trabajan en ese departamento.**

*π (Dname, Prom) ( ρ (*Prom) ← AVG(Salary) *(EMPLOYEE* ⋈ *DEPARTMENT))*

* **Obtenga el salario medio de todas las empleadas.**

*π (Fname, AVG(Salary)) (σ (Sex = ‘F’) (EMPLOYEE))*

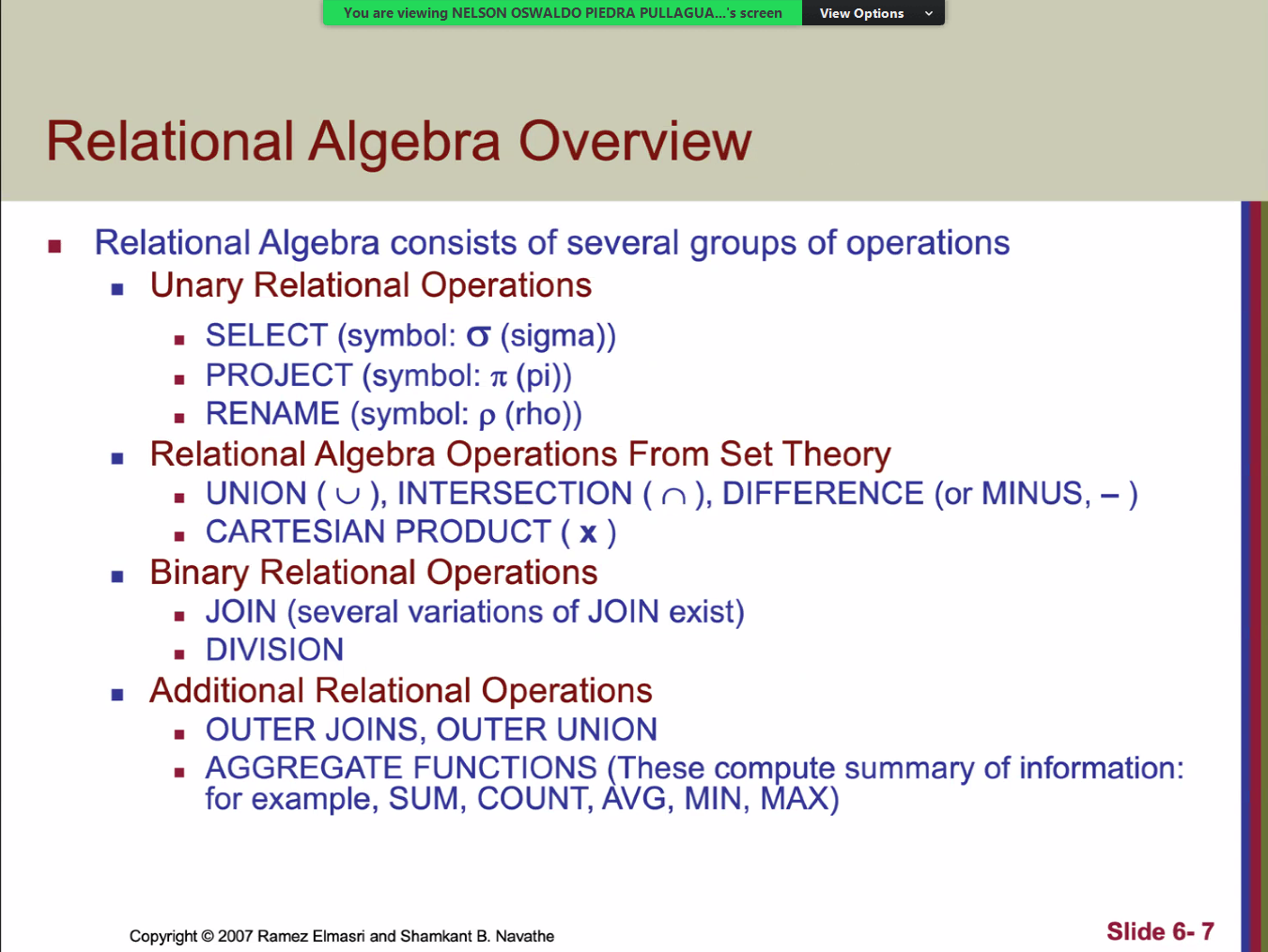
* **Encuentre los nombres y direcciones de todos los empleados que trabajan en al menos un proyecto ubicado en Houston pero cuyo departamento no tiene ubicación en Houston.**

*π (Fname, Address) (σ (D\_location = ‘Houston’) AND (D\_number ¡= 1)(DEPARTMENT* ⋈ *DEP\_LOCATIONS))*

* **Enumere los apellidos de todos los directores de departamento que no tienen personas a su cargo.**

*π (Fname, Address) (π (Mgr\_ssn)(DEPARTMENT) - π (Essn)(DEPENDET) (DEPARTMENT* ⋈ *DEPENDENT* ⋈ *EMPLOYEE))*

**PARTE 3:**



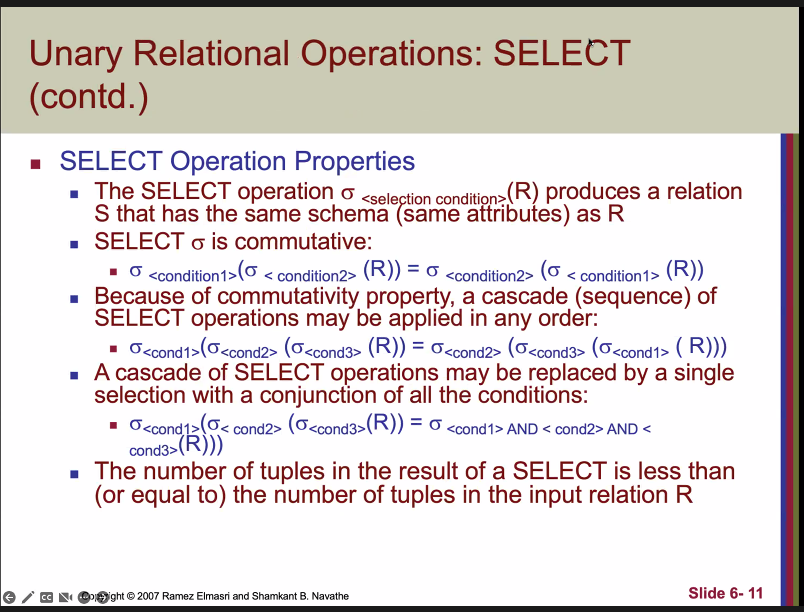
SIGMA: Filas (Select) ()

Pi: Columnas (Project) ()

Rho: Renombrar(Rename) ()

A screenshot of a computer

Description automatically generated



A screenshot of a computer screen

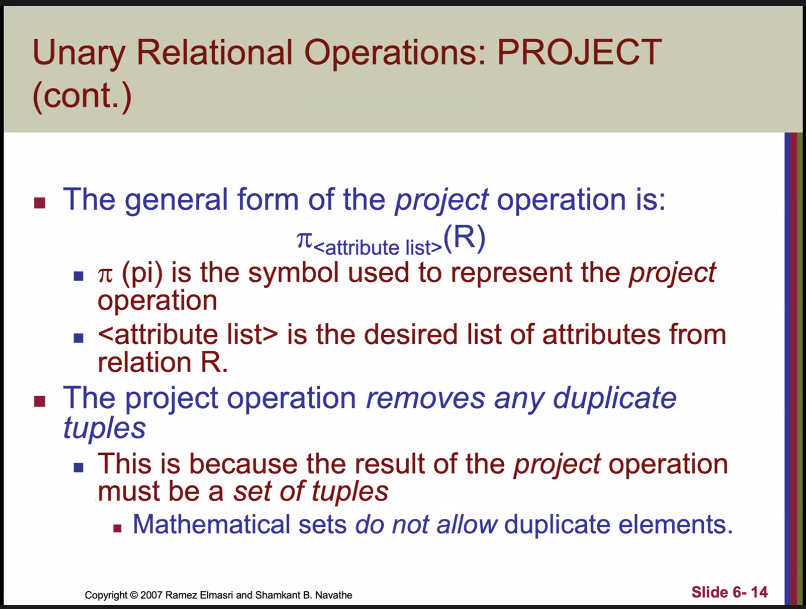
Description automatically generated

Seleccionar los empleados de sexo femenino q trabajan en el departamento 4:

σ Department = 4 (σ Sex = "F" (EMPLOYEE))

A screenshot of a computer

Description automatically generated

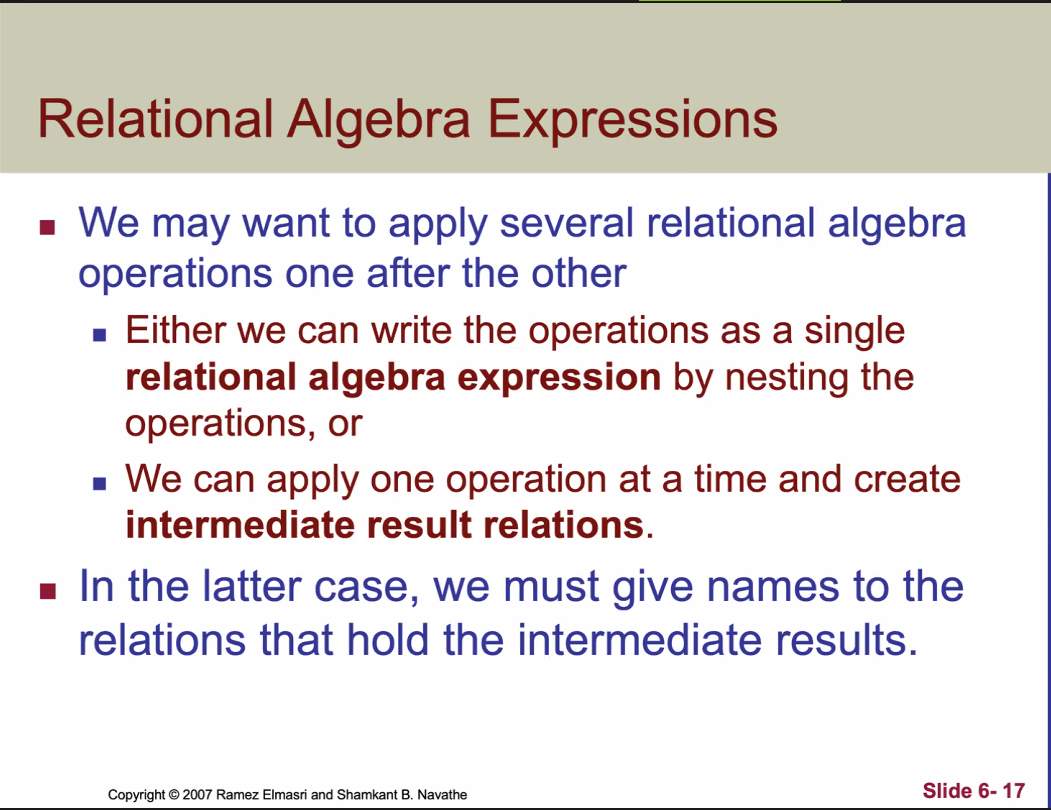


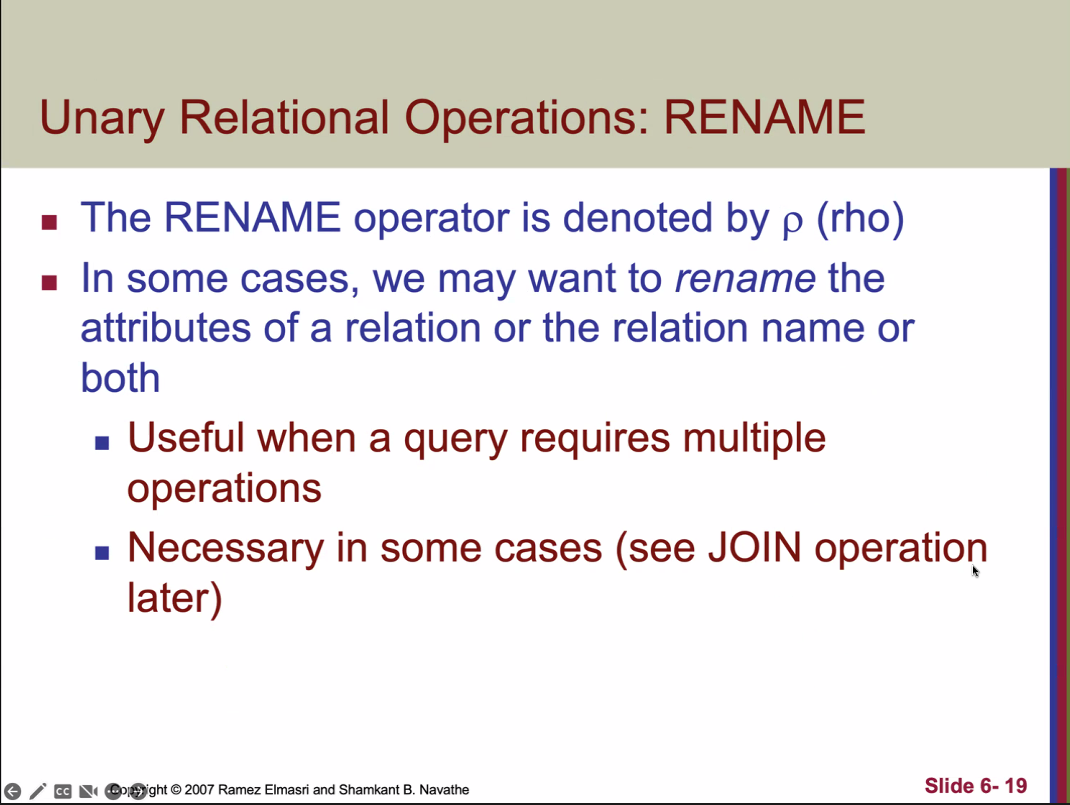
Que hace la siguiente consulta:

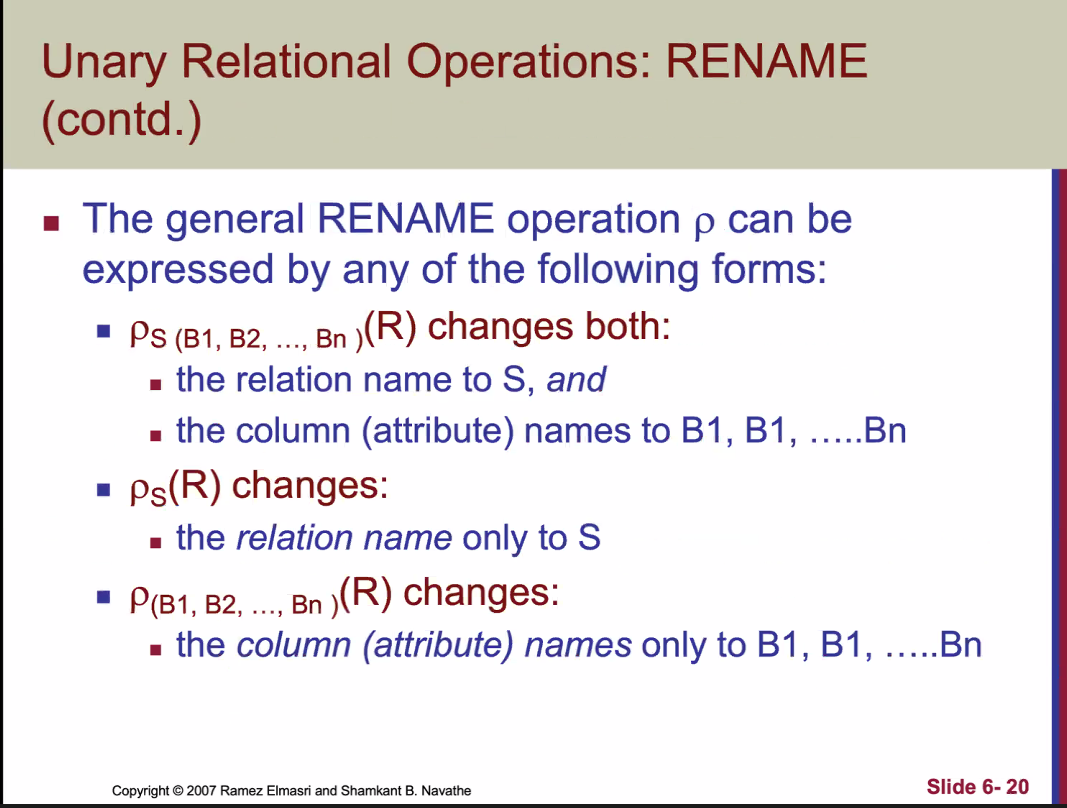
A screenshot of a document

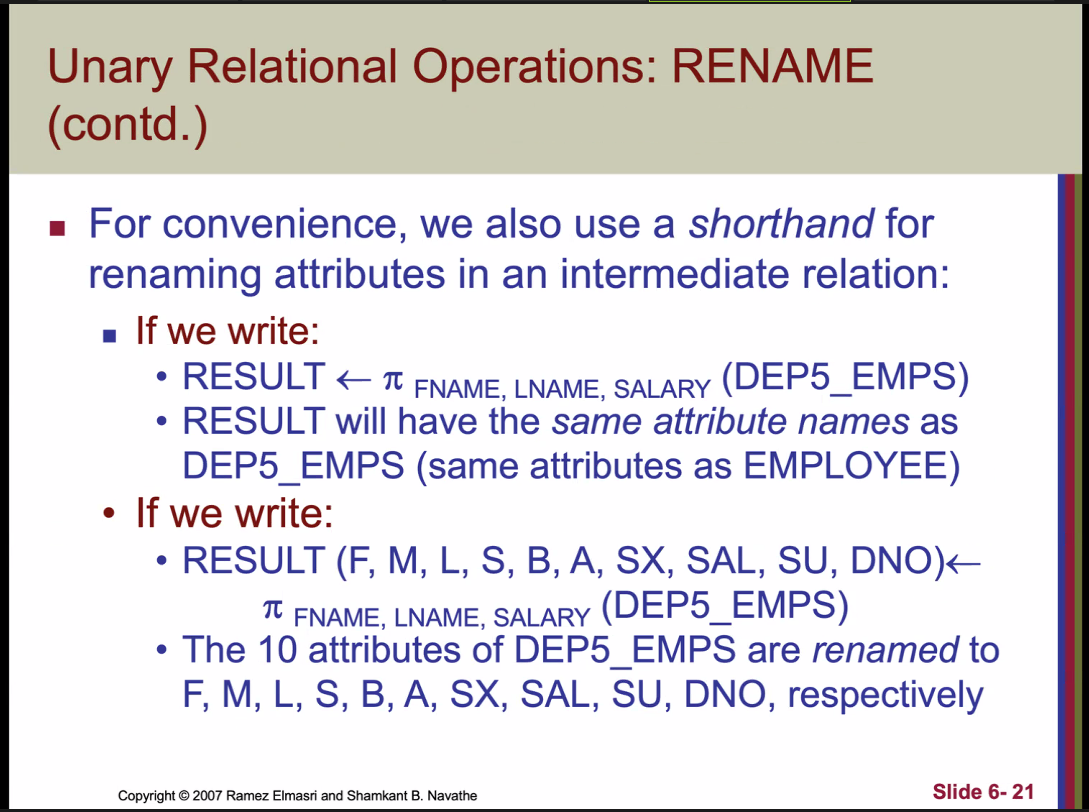
Description automatically generated

Selecciona los (Números de departamento iguales a 4 **y** que tengan un salario mayor a 25 000) **o** (Números de departamento iguales a 5 **y** que tengan un salario mayor a 30 000), su relación es EMPLOYEE, también, selecciona las columnas Lname, Fname, Salary, donde su relación es EMPLOYEE, además, selecciona las columnas Sex, Salary, donde su relacion es EMPLOYEE









Asignamos a RESULT lo siguiente: Seleccionamos las columnas FNAME, LNAME, SALARY, en donde la relación es DEP5\_EMPS

Obtenemos una relación que se llame RESULT donde Seleccionamos las columnas FNAME, LNAME, SALARY, en donde la relación es DEP5\_EMPS