**CUADERNO PL/SQL**

Este cuaderno de PL/SQL tiene como objetivo reforzar la programación PL/SQL con una serie de ejercicios diseñados para ser aplicados en un entorno de desarrollo Oracle. Además de mejorar las habilidades de programación, se busca potenciar la memoria y el desarrollo lógico en la creación y manipulación de datos en bases de datos estructuradas, así como en la programación y el desarrollo dentro del entorno PL/SQL. Este material también proporcionará una oportunidad para consolidar los conocimientos en la gestión de bases de datos en distintos niveles, abarcando desde los conceptos fundamentales hasta niveles más avanzados. El cuaderno debe mantenerse al alcance y completarse a lo largo del curso, el cual se extiende durante 16 semanas según lo establecido en el plan de estudio.

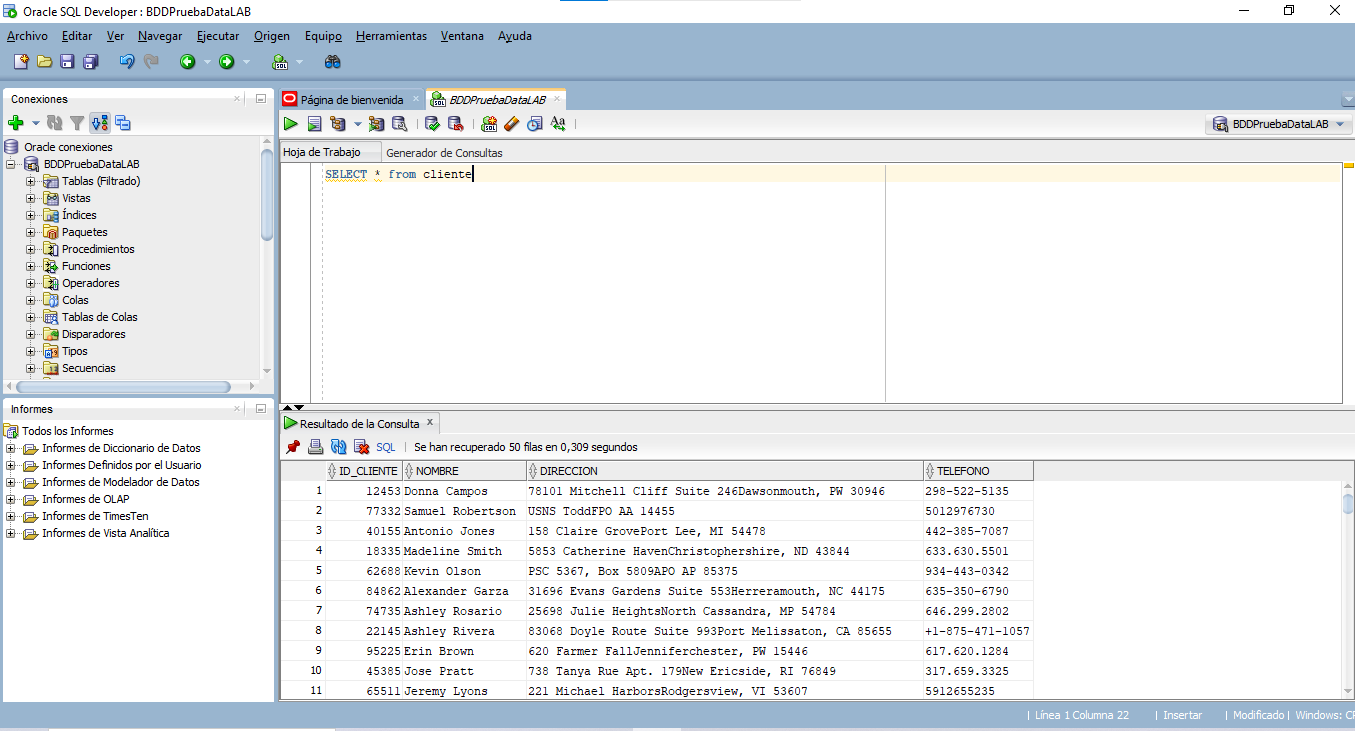
El cuaderno comprende un total de 40 ejercicios diseñados para abordar diversas áreas del aprendizaje. Es esencial que cada estudiante lo lleve de manera individual y que se realice una revisión durante cada clase para verificar el progreso en los ejercicios de la semana. El seguimiento y registro del cuaderno serán elementos fundamentales que contribuirán a la evaluación académica del estudiante. Es importante destacar la relevancia de este ejercicio académico como una herramienta integral para fortalecer las competencias adquiridas en clase y en todo el proceso de aprendizaje relacionado con la temática del curso.

**METODOLOGÍA:**

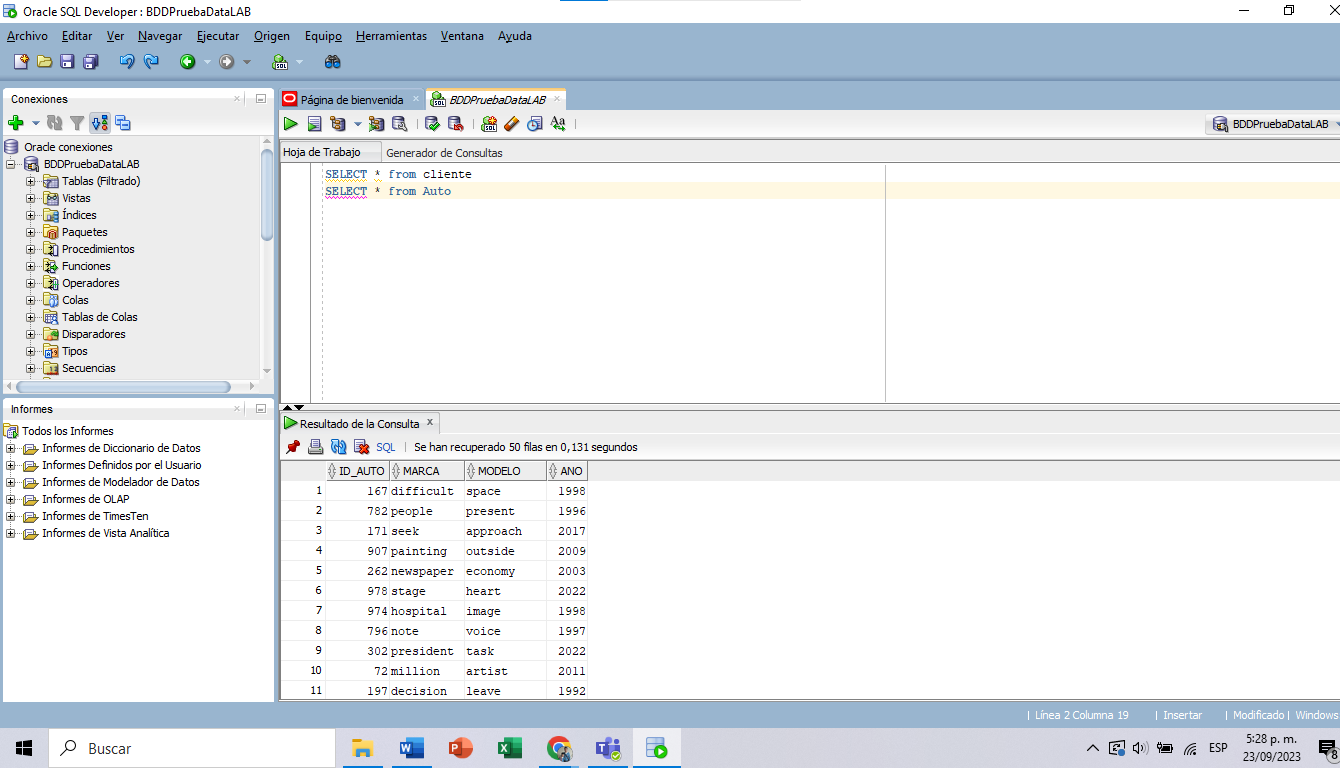
* Cada estudiante debe crear un repositorio de GitHub con el nombre de Bases de Datos.
* Se debe crear el link en el Dashboard en la sección de PL/SQL
* Se debe crear un código SQL por cada Ejercicio, realizando la explicación de cómo funciona el código y que resultados se generaron.

**EJERCICIOS PRIMER CICLO (1-30):**

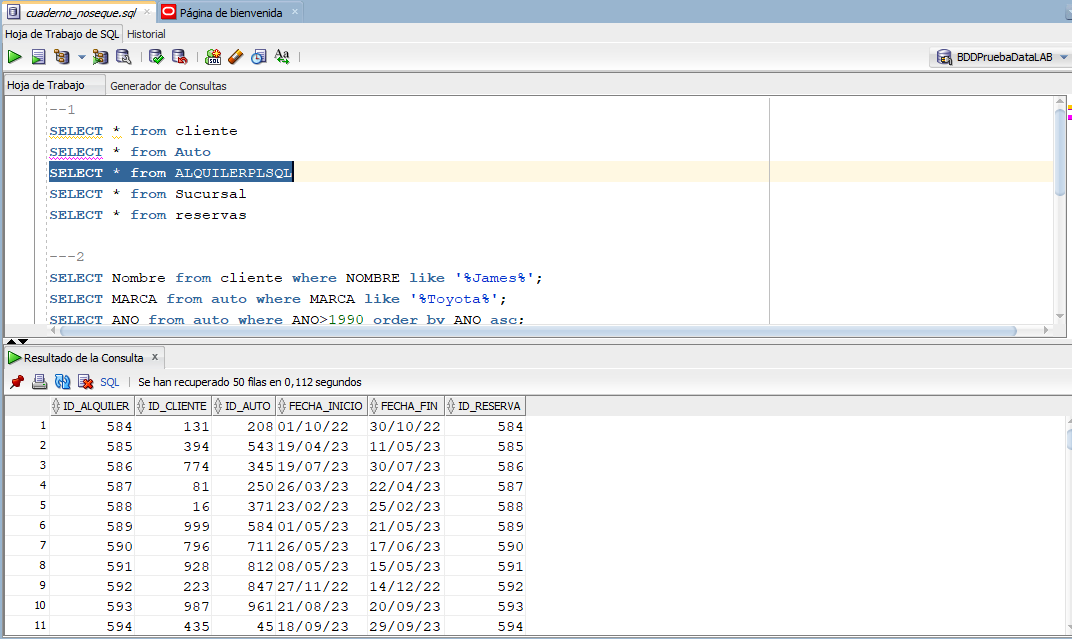
1. **Consultas Básicas:**
   1. Mostrar todos los clientes en la tabla "Cliente".



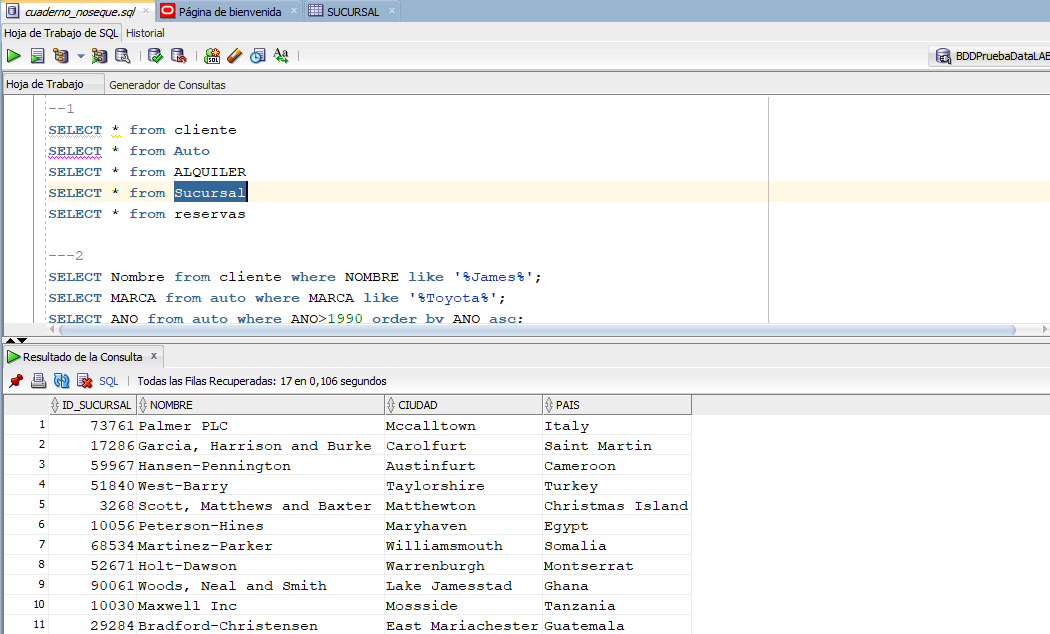
* 1. Mostrar todos los autos en la tabla "Auto".



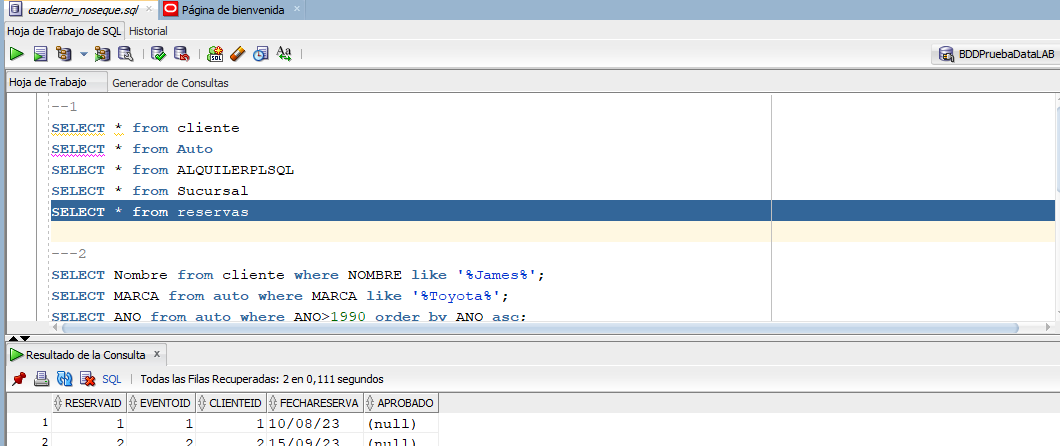
* 1. Mostrar todos los alquileres en la tabla "Alquiler".



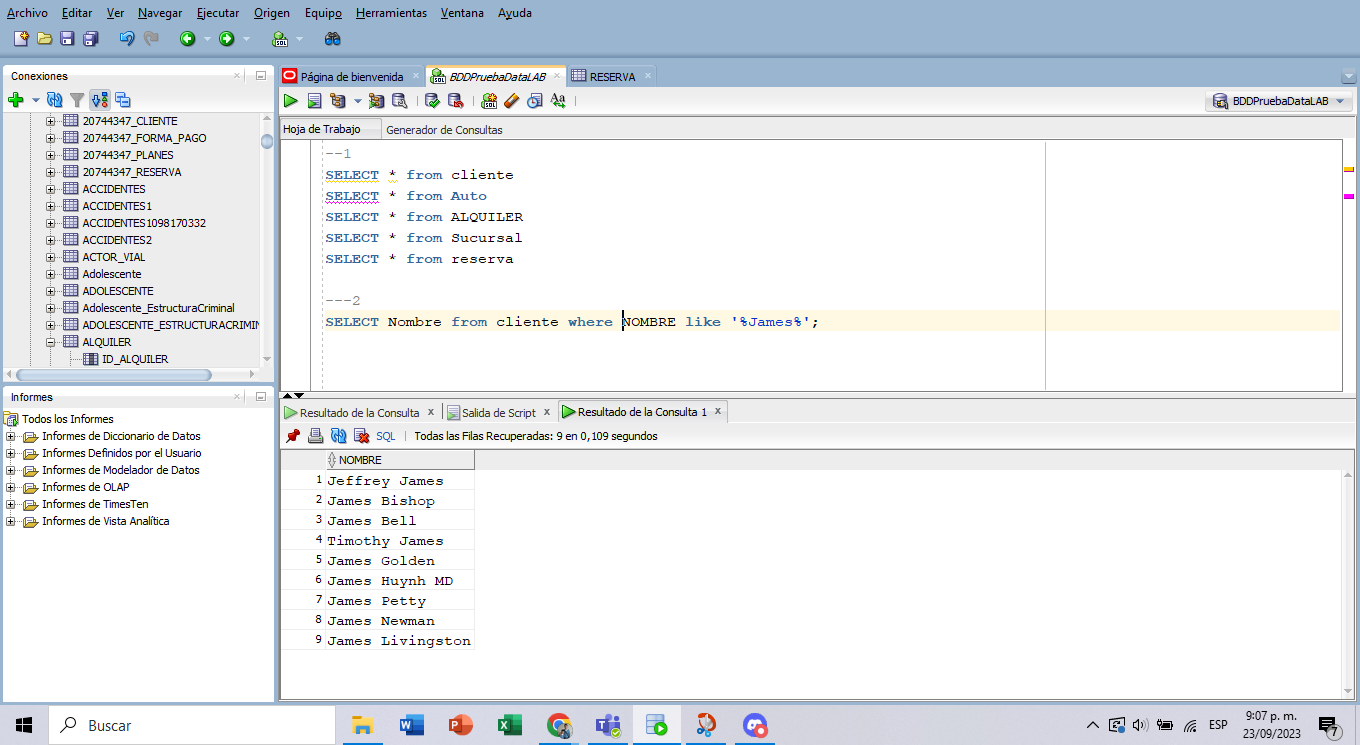
* 1. Mostrar todas las sucursales en la tabla "Sucursal".



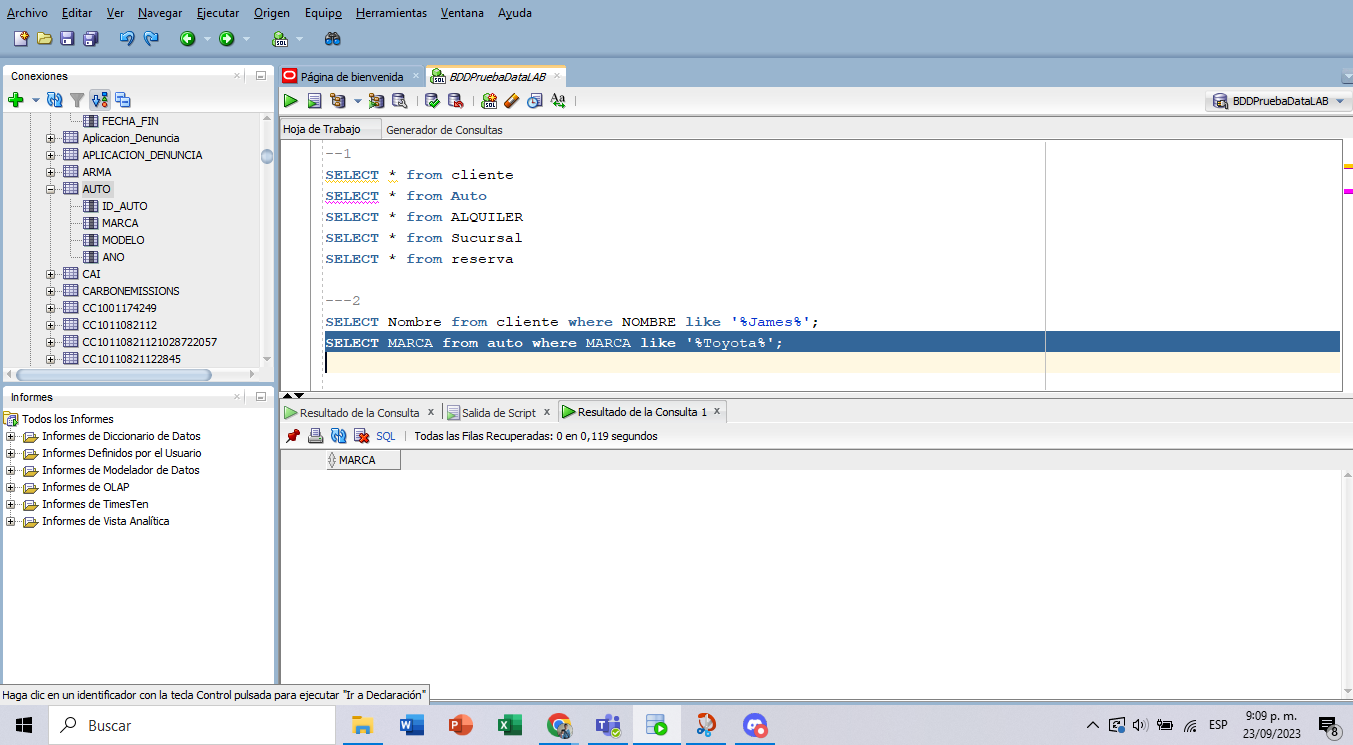
* 1. Mostrar todas las reservas en la tabla "Reserva".



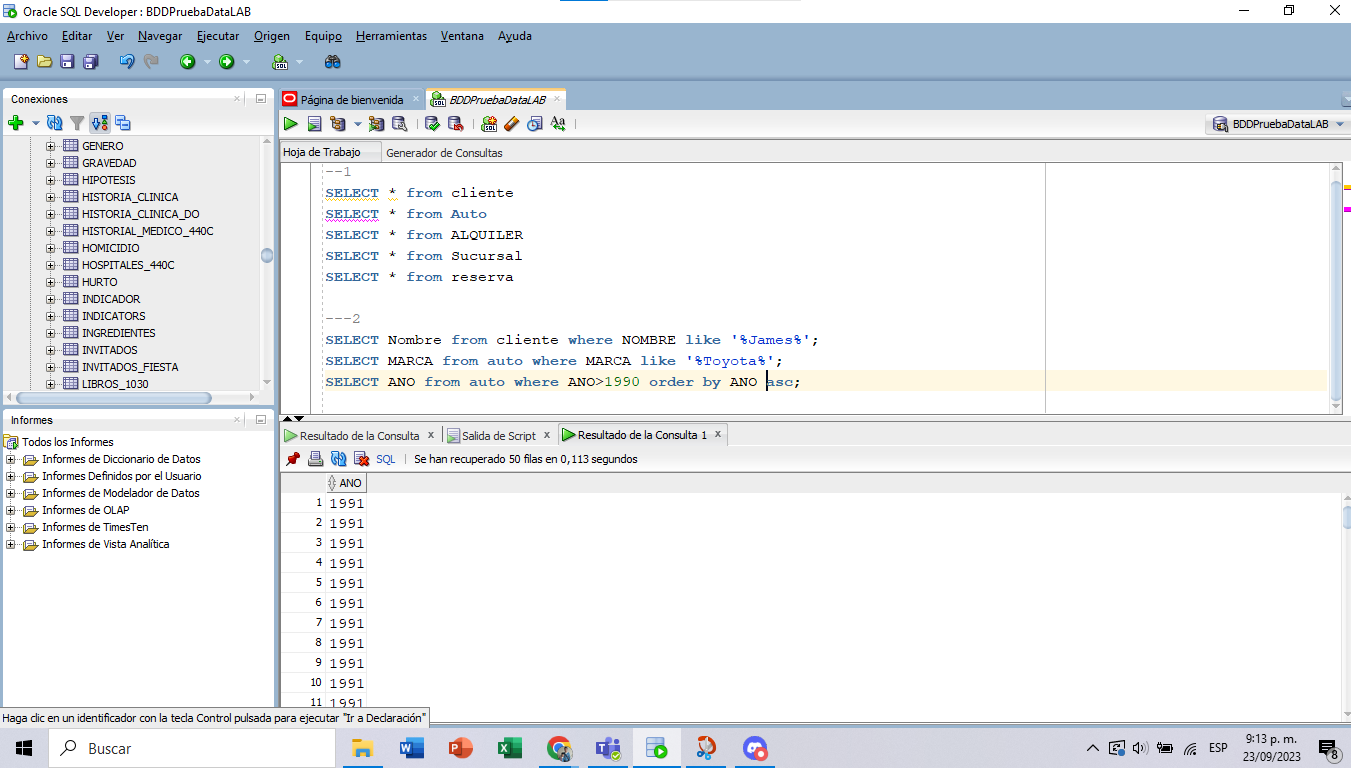
1. **Filtros y Ordenamiento:**
   1. Mostrar a los clientes que se llaman "Juan".



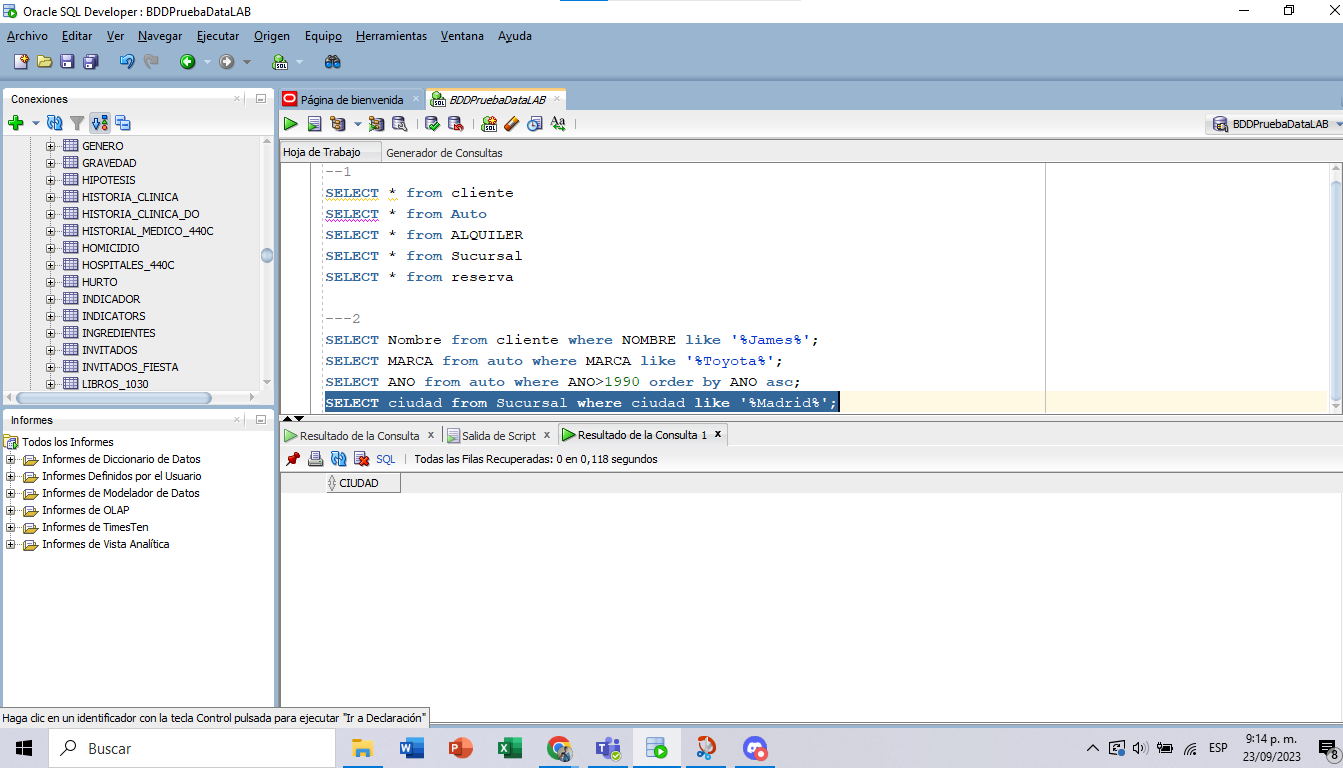
* 1. Mostrar los autos de marca "Toyota".



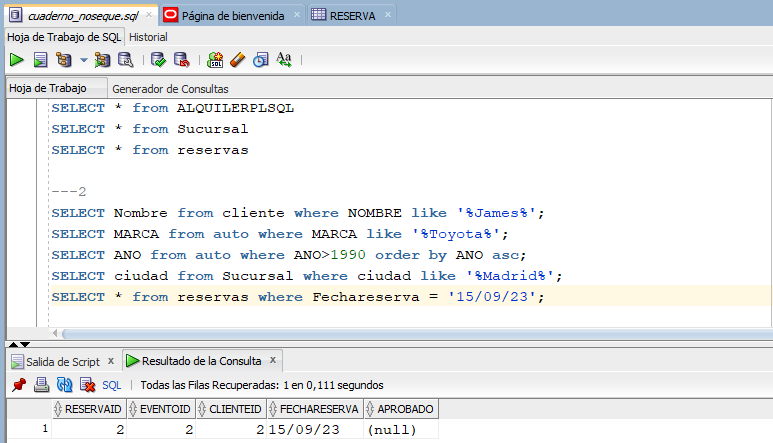
* 1. Mostrar los alquileres que ocurrieron después de una fecha específica.



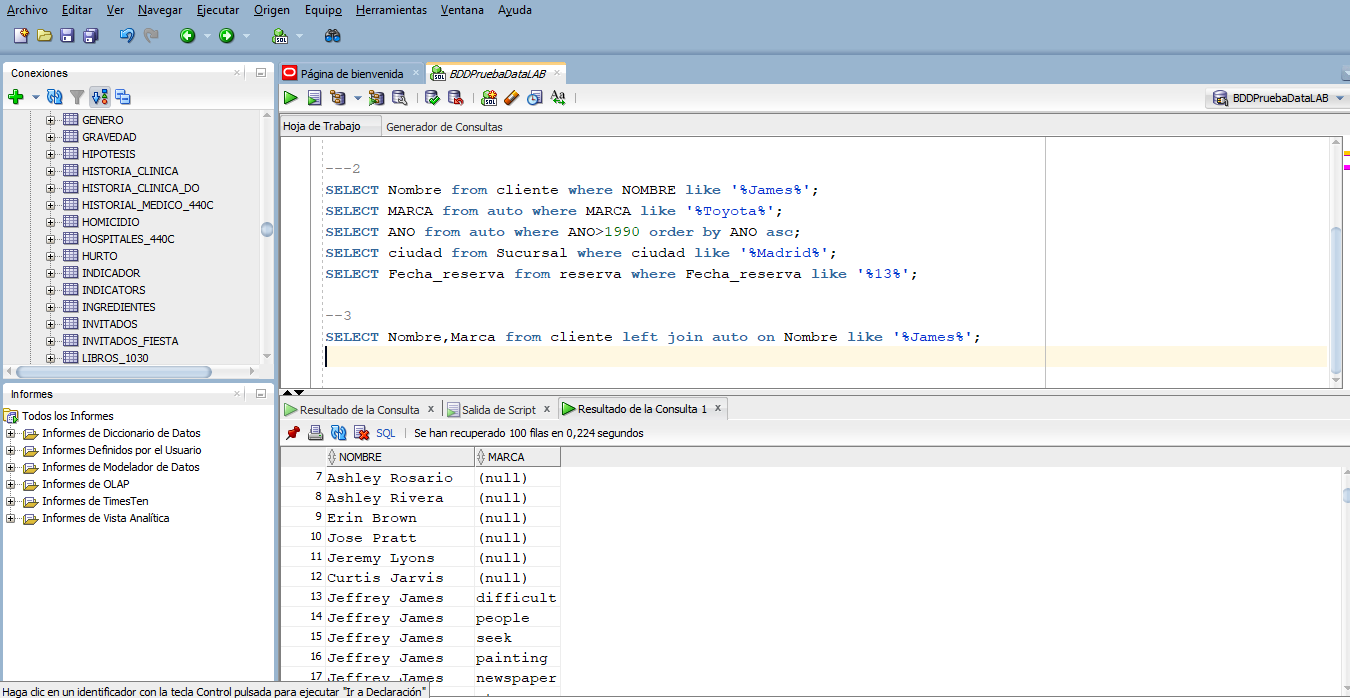
* 1. Mostrar las sucursales ubicadas en "Madrid".



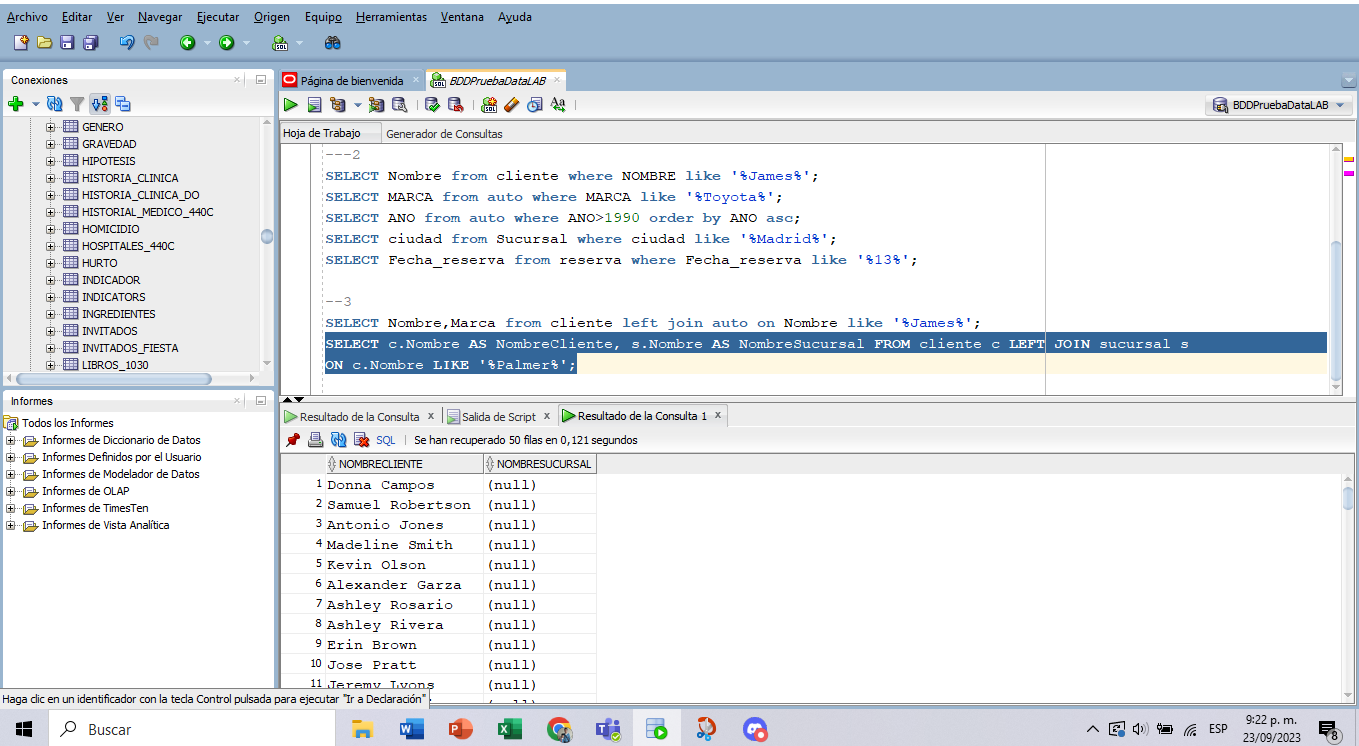
* 1. Mostrar las reservas realizadas por un cliente específico.



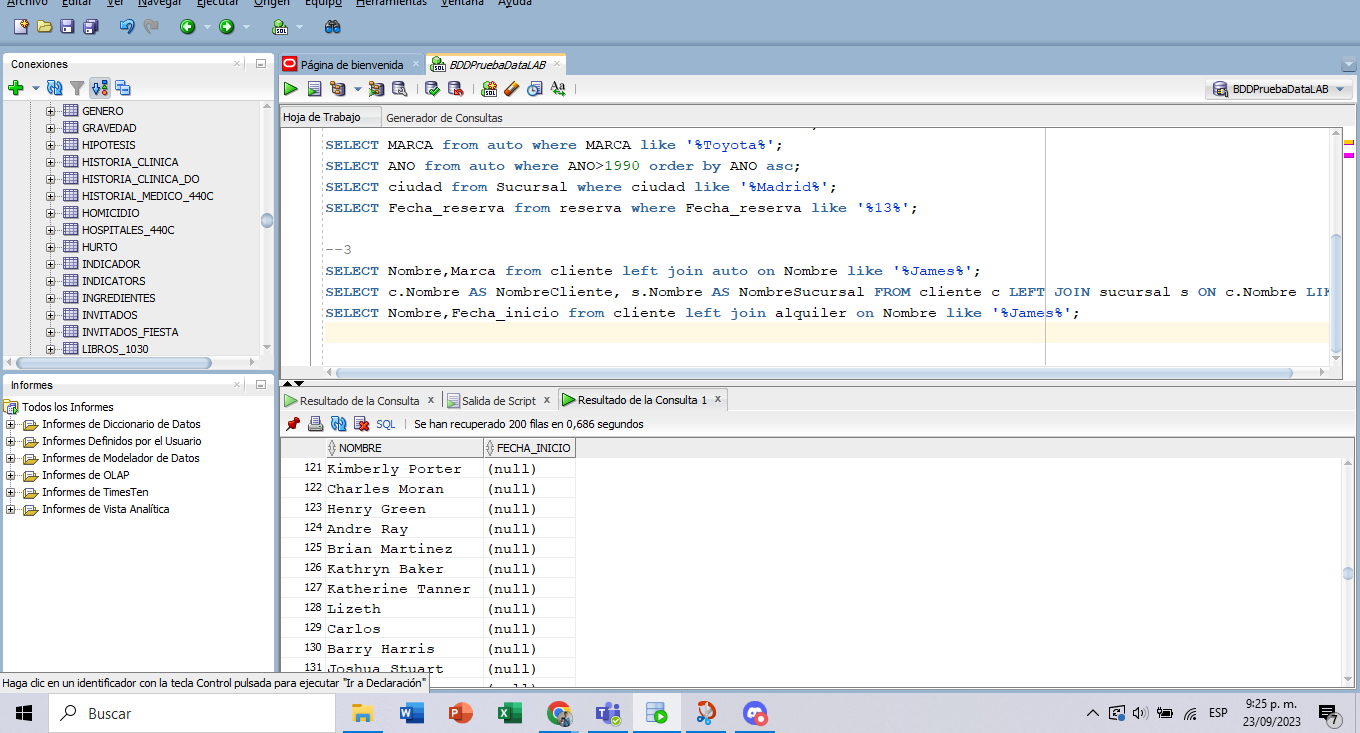
1. **Join y Relaciones:**
   1. Mostrar los alquileres con los nombres de los clientes y las marcas de los autos.



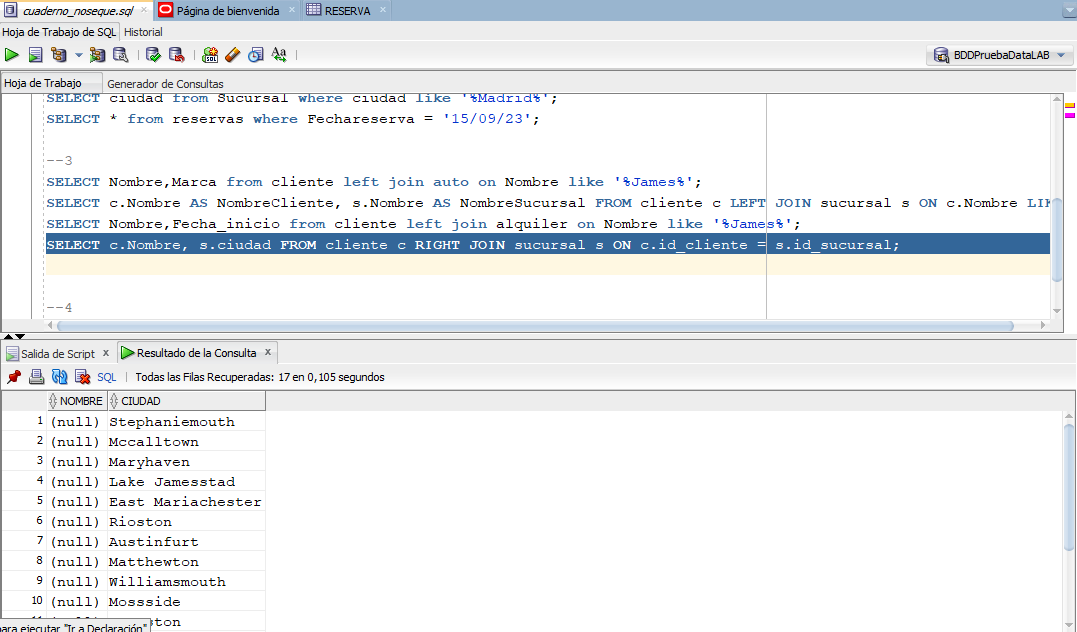
* 1. Mostrar los clientes que han realizado reservas en una sucursal específica.



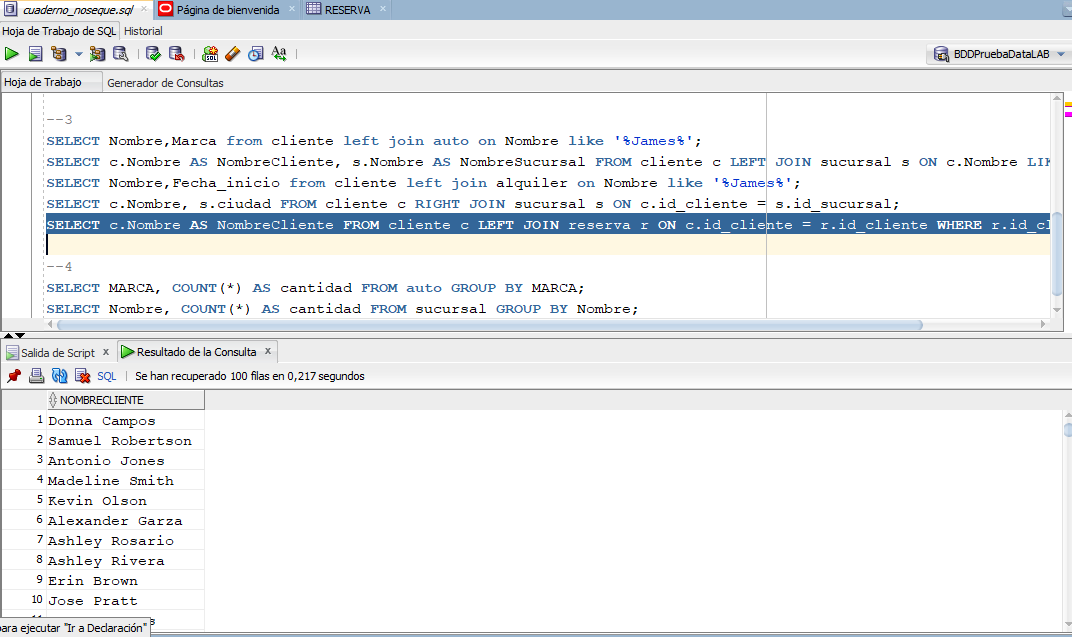
* 1. Mostrar los autos que han sido alquilados junto con los nombres de los clientes.



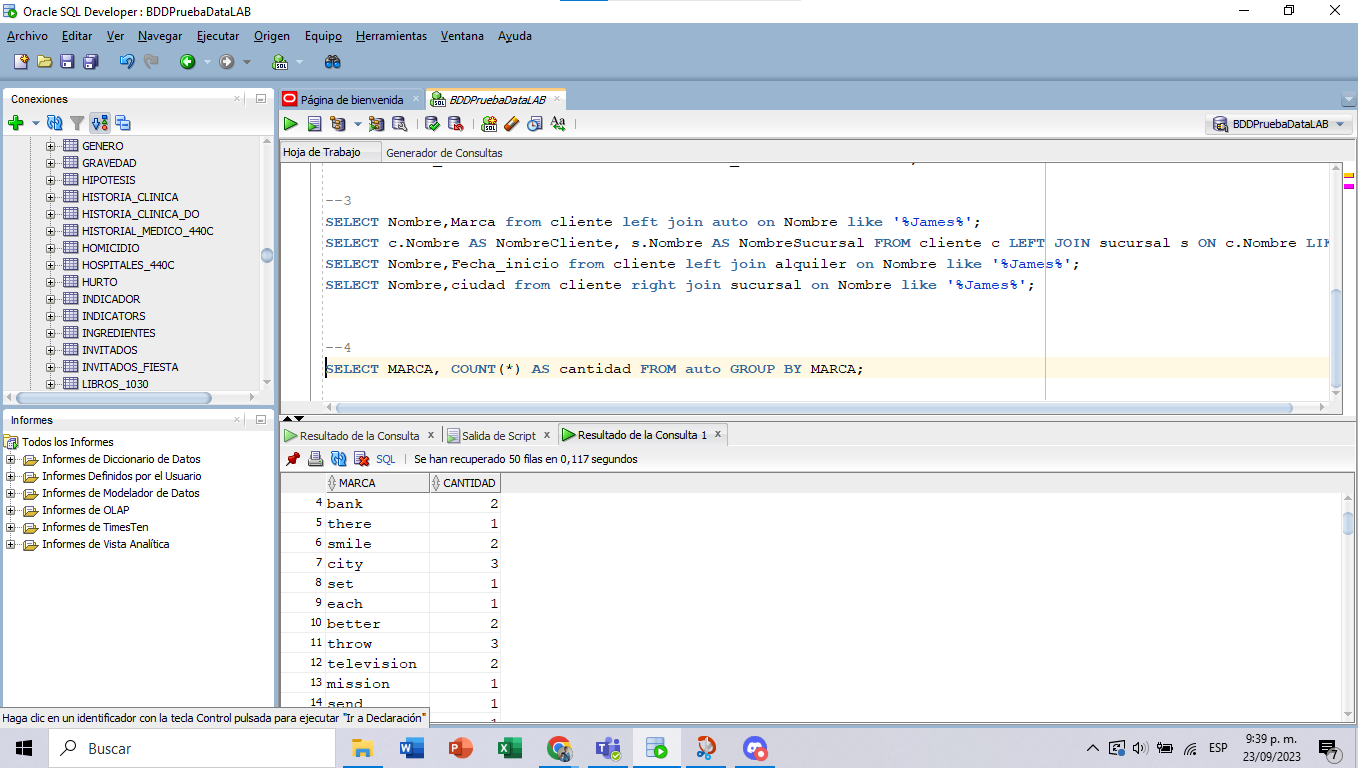
* 1. Mostrar los detalles de las reservas con los nombres de los clientes y las ciudades de las sucursales.



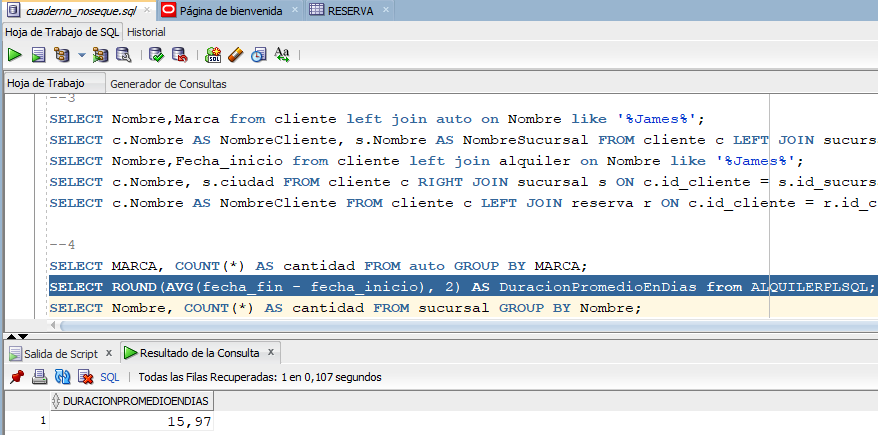
* 1. Mostrar a los clientes que no han realizado ninguna reserva.



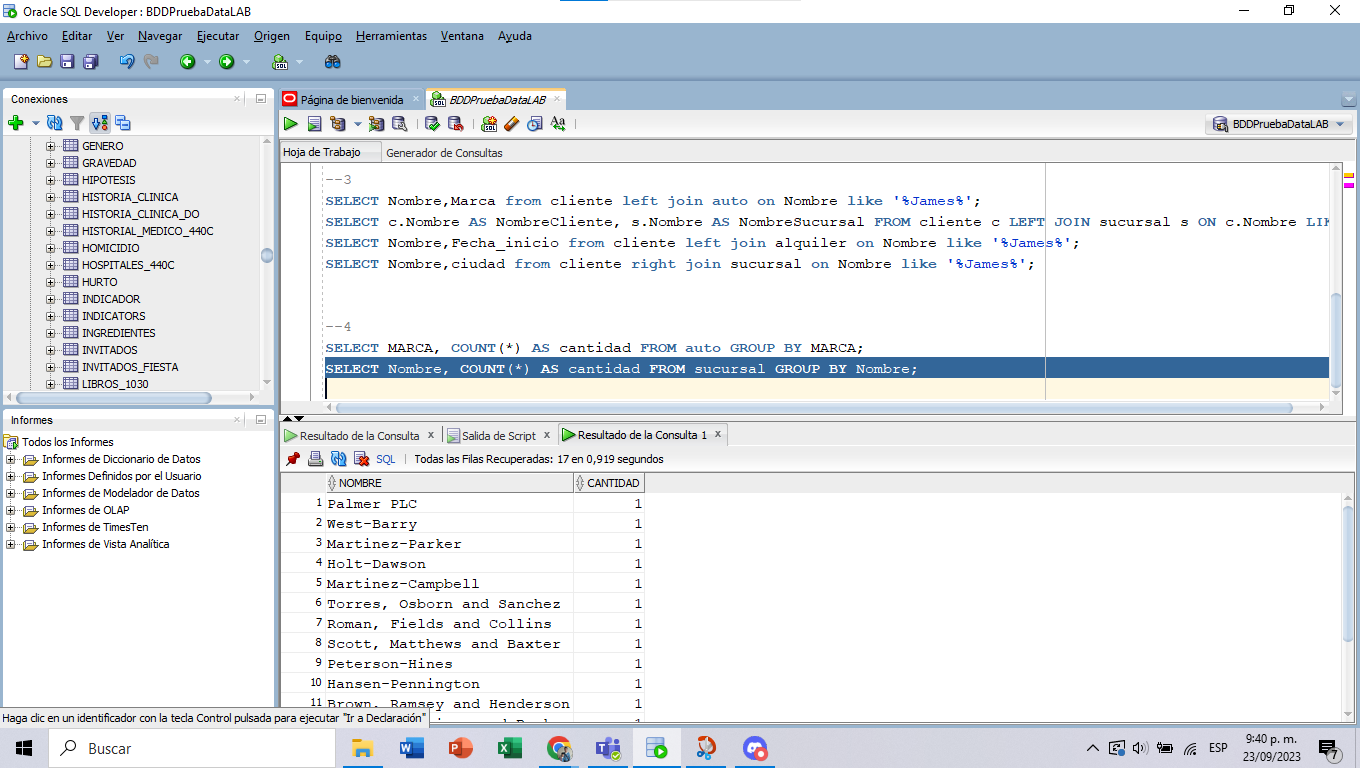
1. **Agregación y Agrupamiento:**
   1. Contar cuántos autos hay de cada marca en la tabla "Auto".



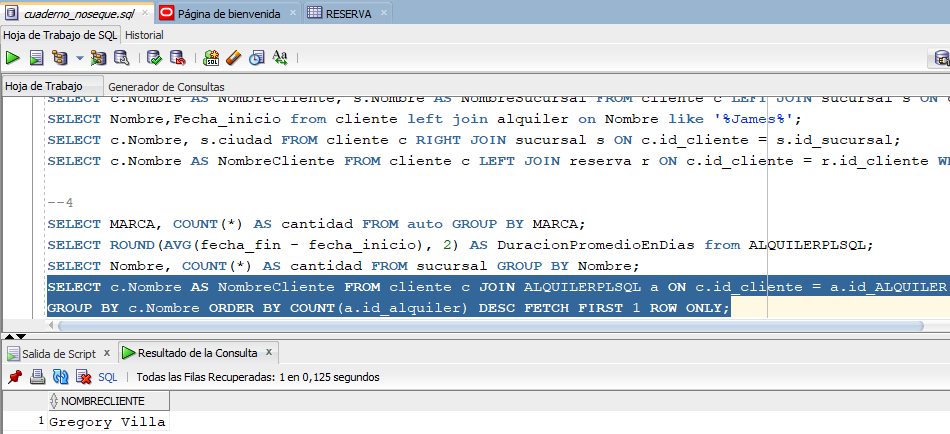
* 1. Calcular la duración promedio de los alquileres.



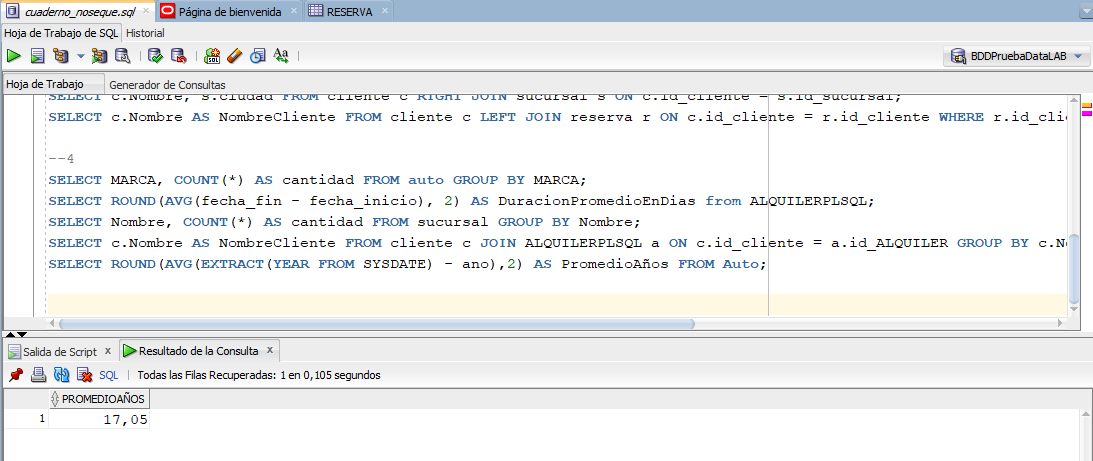
* 1. Mostrar el número total de reservas realizadas en cada sucursal.



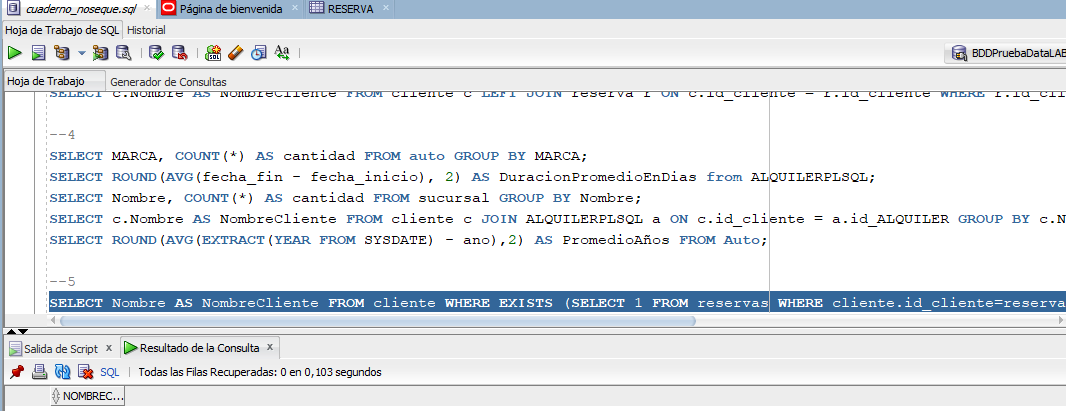
* 1. Encontrar el cliente que ha realizado la mayor cantidad de alquileres.



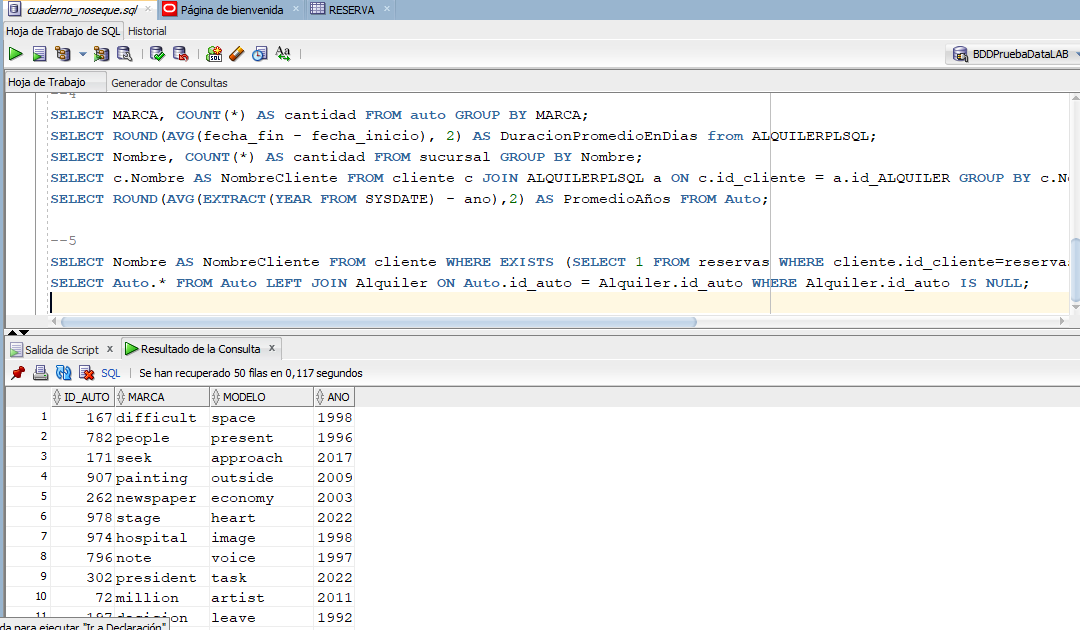
* 1. Calcular el promedio de años de los autos en la tabla "Auto".



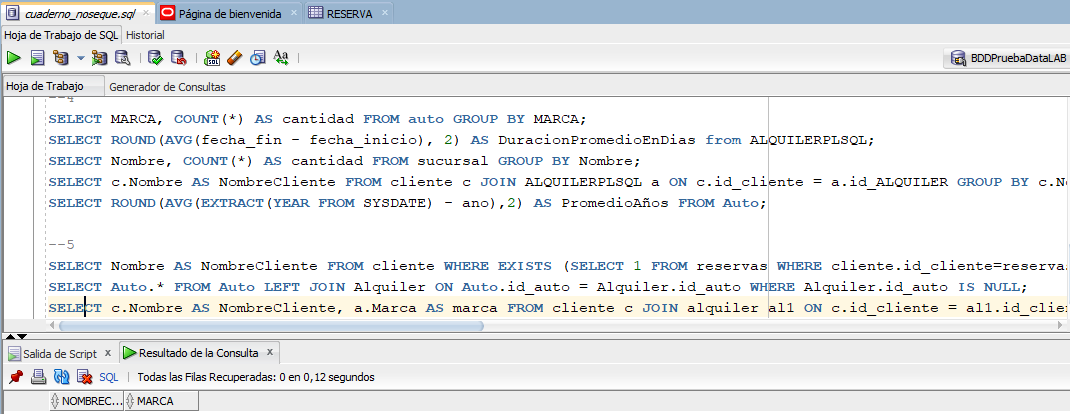
1. **Subconsultas**:
   1. Mostrar los clientes que han realizado al menos una reserva.



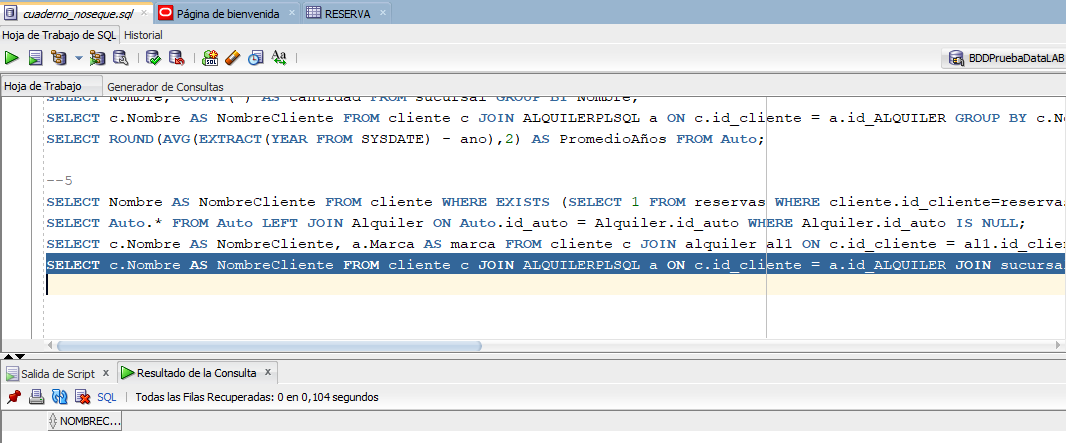
* 1. Mostrar los autos que no han sido alquilados aún.



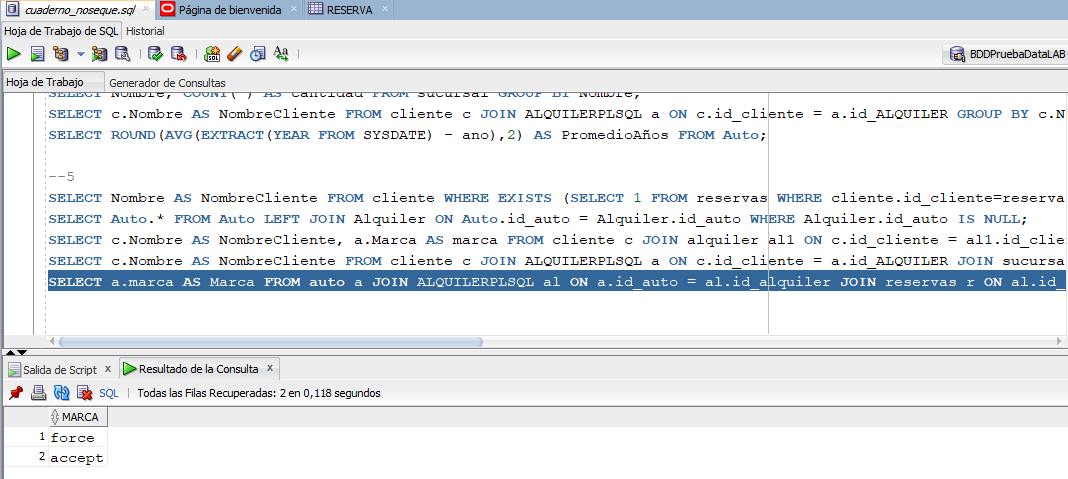
* 1. Encontrar los clientes que han alquilado el mismo auto más de una vez.



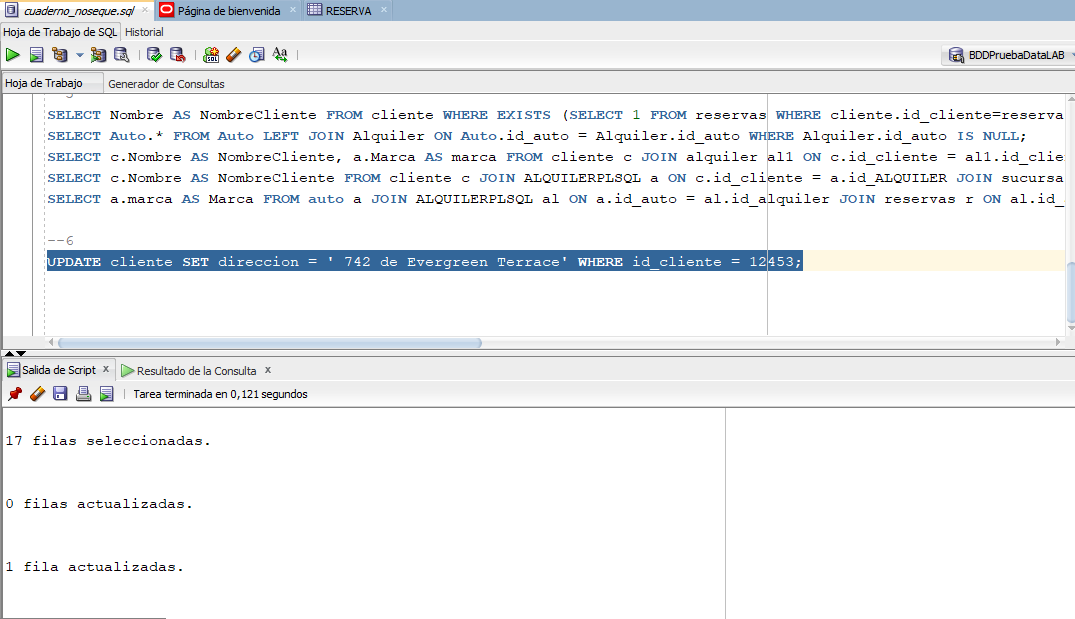
* 1. Mostrar los clientes que han realizado alquileres en la misma ciudad en la que viven.



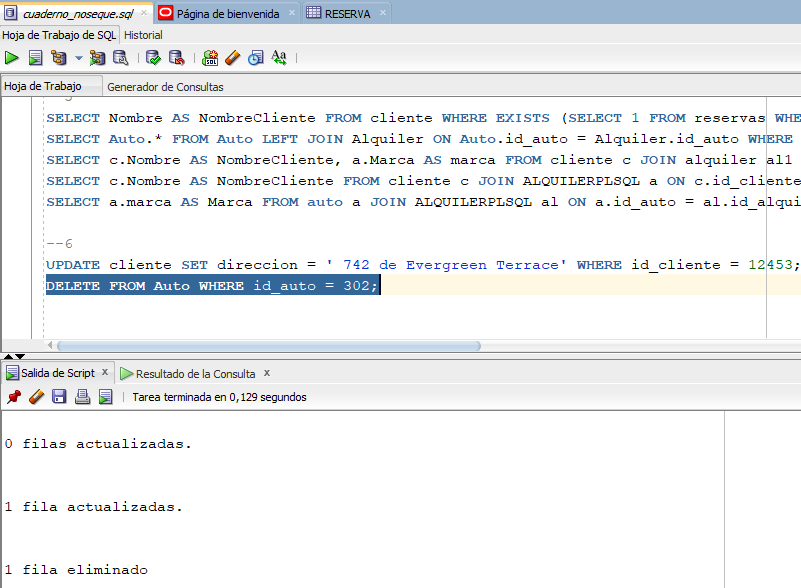
* 1. Encontrar los autos que han sido alquilados en la misma sucursal donde se realizó una reserva.



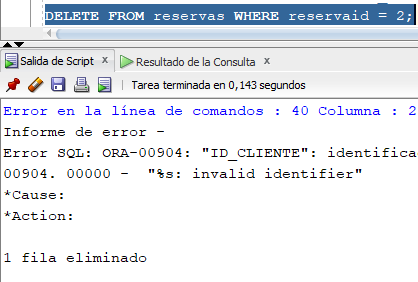
1. **Actualizaciones y Eliminaciones:**
   1. Actualizar la dirección de un cliente específico.



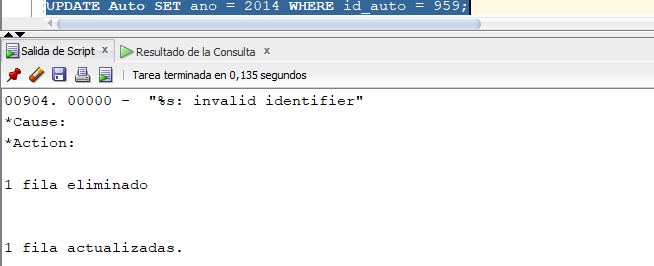
* 1. Eliminar un auto de la tabla "Auto".



* 1. Marcar una reserva como completada actualizando la fecha de fin.
  2. Eliminar todas las reservas realizadas por un cliente específico.

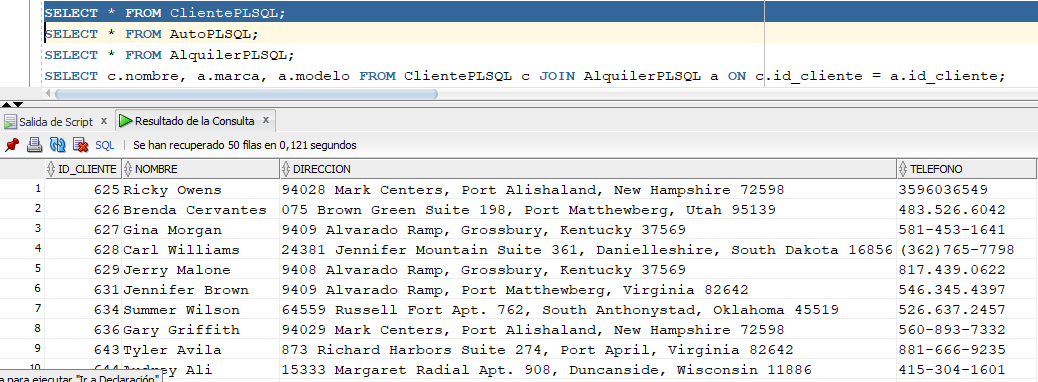


* 1. Actualizar el año de un auto en la tabla "Auto".

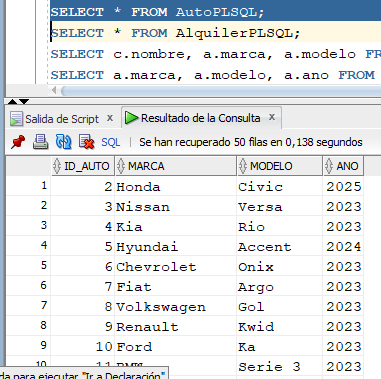


**EJERCICIOS SEGUNDO CICLO (31-80):**

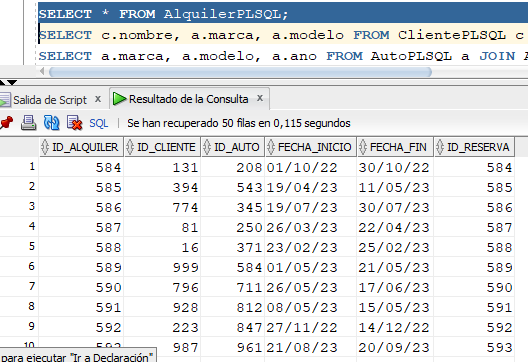
* SELECT \* FROM ClientePLSQL;



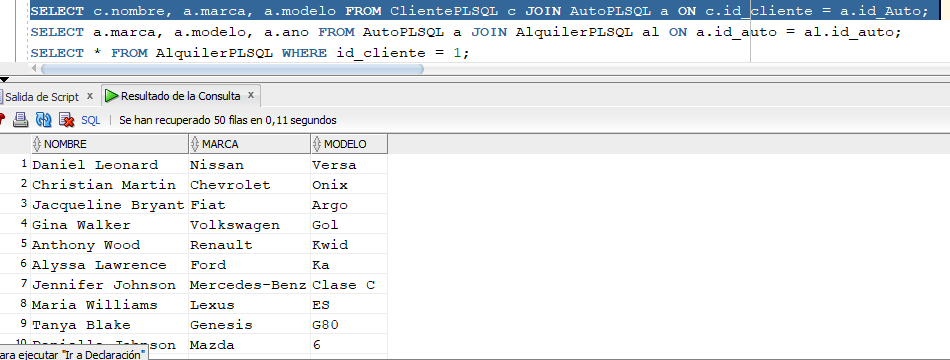
* SELECT \* FROM AutoPLSQL;



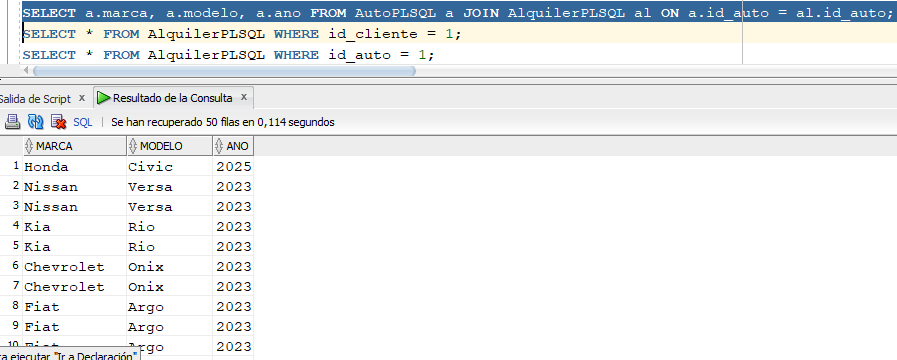
* SELECT \* FROM AlquilerPLSQL;



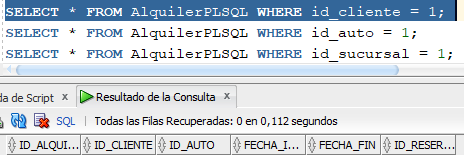
* SELECT c.nombre, a.marca, a.modelo FROM ClientePLSQL c JOIN AlquilerPLSQL a ON c.id\_cliente = a.id\_cliente;



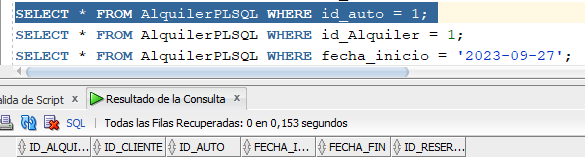
* SELECT a.marca, a.modelo, a.ano FROM AutoPLSQL a JOIN AlquilerPLSQL al ON a.id\_auto = al.id\_auto;



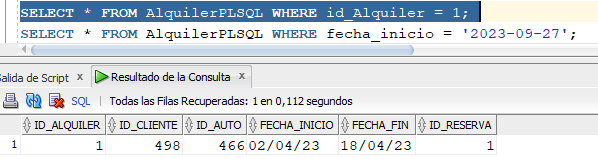
* SELECT \* FROM AlquilerPLSQL WHERE id\_cliente = 1;



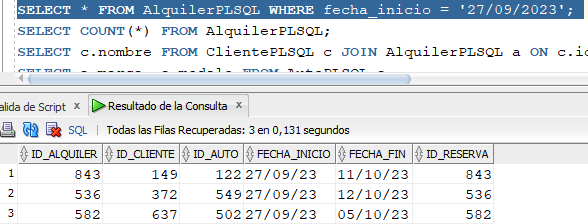
* SELECT \* FROM AlquilerPLSQL WHERE id\_auto = 1;



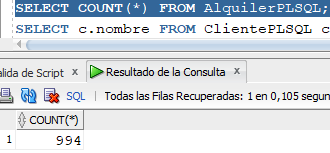
* SELECT \* FROM AlquilerPLSQL WHERE id\_sucursal = 1;



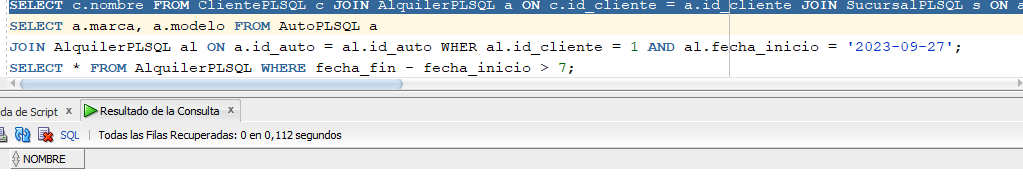
* SELECT \* FROM AlquilerPLSQL WHERE fecha\_inicio = '2023-09-27';



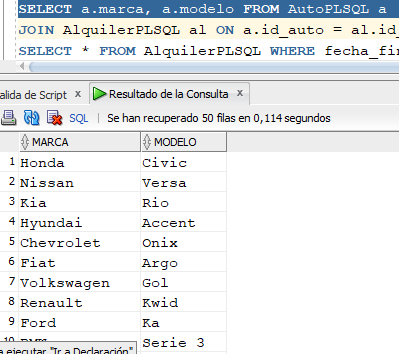
* SELECT COUNT(\*) FROM AlquilerPLSQL;



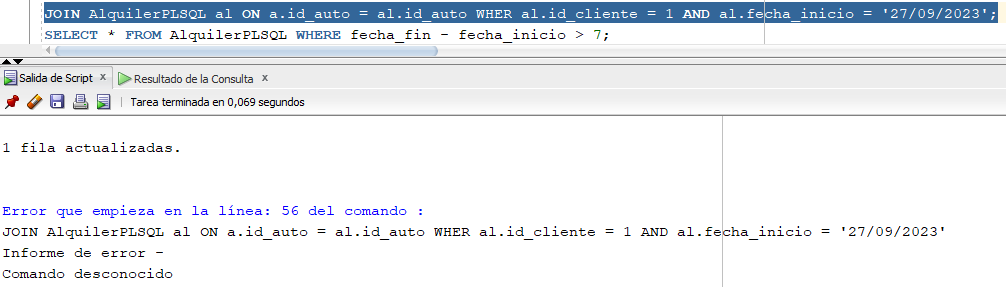
* SELECT c.nombre FROM ClientePLSQL c JOIN AlquilerPLSQL a ON c.id\_cliente = a.id\_cliente JOIN SucursalPLSQL s ON a.id\_sucursal = s.id\_sucursal WHERE s.nombre = 'Sucursal Central';



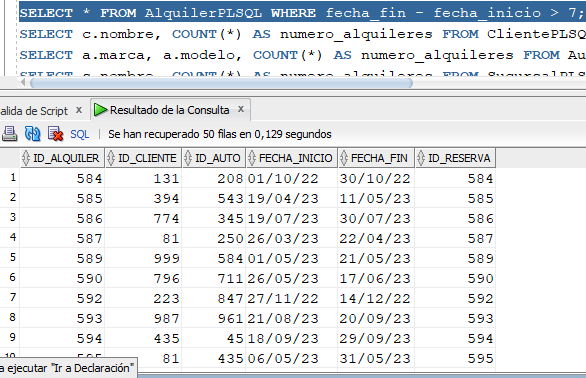
* SELECT a.marca, a.modelo FROM AutoPLSQL a



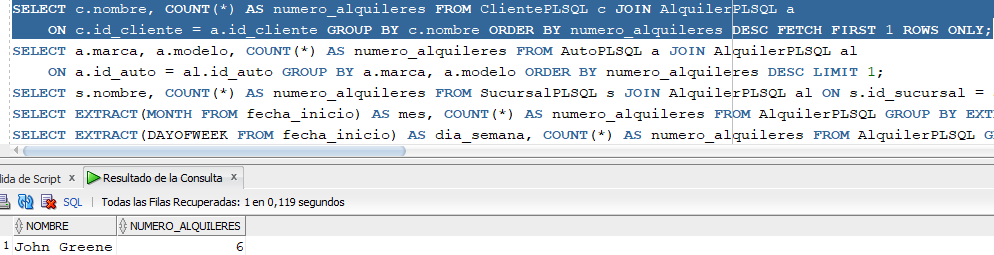
* JOIN AlquilerPLSQL al ON a.id\_auto = al.id\_auto WHER al.id\_cliente = 1 AND al.fecha\_inicio = '2023-09-27';



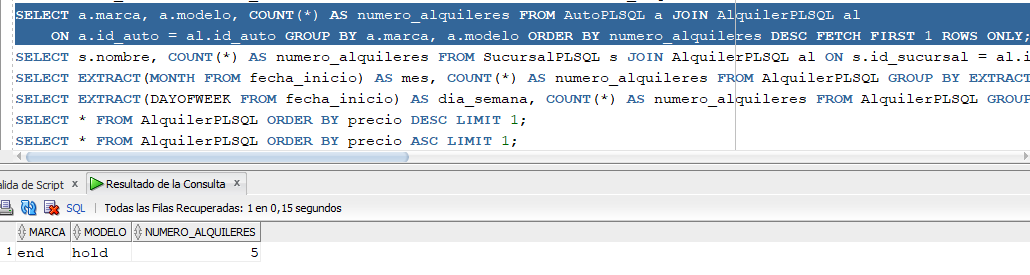
* SELECT \* FROM AlquilerPLSQL WHERE fecha\_fin - fecha\_inicio > 7;



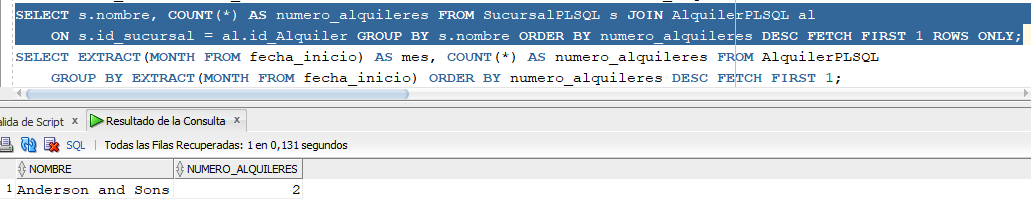
* SELECT c.nombre, COUNT(\*) AS numero\_alquileres FROM ClientePLSQL c JOIN AlquilerPLSQL a ON c.id\_cliente = a.id\_cliente GROUP BY c.nombre ORDER BY numero\_alquileres DESC LIMIT 1;



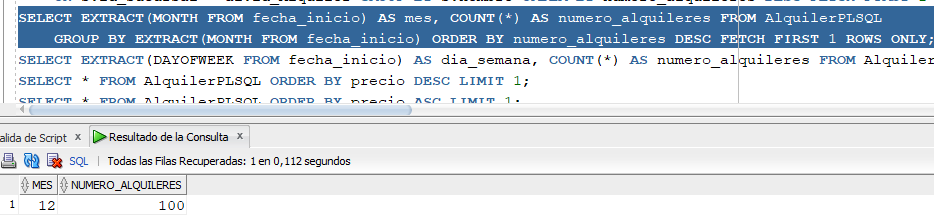
* SELECT a.marca, a.modelo, COUNT(\*) AS numero\_alquileres FROM AutoPLSQL a JOIN AlquilerPLSQL al ON a.id\_auto = al.id\_auto GROUP BY a.marca, a.modelo ORDER BY numero\_alquileres DESC LIMIT 1;



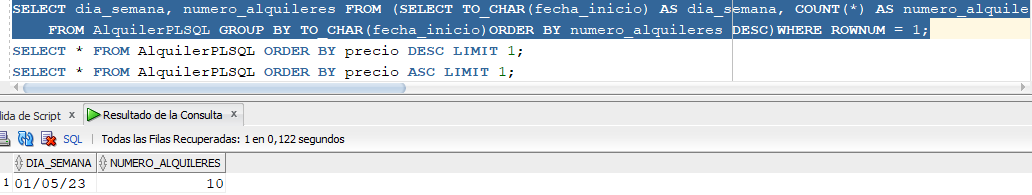
* SELECT s.nombre, COUNT(\*) AS numero\_alquileres FROM SucursalPLSQL s JOIN AlquilerPLSQL al ON s.id\_sucursal = al.id\_sucursal GROUP BY s.nombre ORDER BY numero\_alquileres DESC LIMIT 1;



* SELECT EXTRACT(MONTH FROM fecha\_inicio) AS mes, COUNT(\*) AS numero\_alquileres FROM AlquilerPLSQL GROUP BY EXTRACT(MONTH FROM fecha\_inicio) ORDER BY numero\_alquileres DESC LIMIT 1;



* SELECT EXTRACT(DAYOFWEEK FROM fecha\_inicio) AS dia\_semana, COUNT(\*) AS numero\_alquileres FROM AlquilerPLSQL GROUP BY EXTRACT(DAYOFWEEK FROM fecha\_inicio) ORDER BY numero\_alquileres DESC LIMIT 1;



* SELECT \* FROM AlquilerPLSQL ORDER BY precio DESC LIMIT 1;

No existe un precio

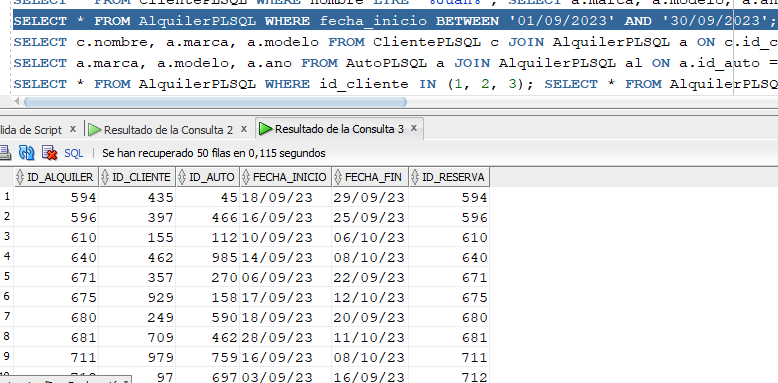
* SELECT \* FROM AlquilerPLSQL ORDER BY precio ASC LIMIT 1;

No existe un precio

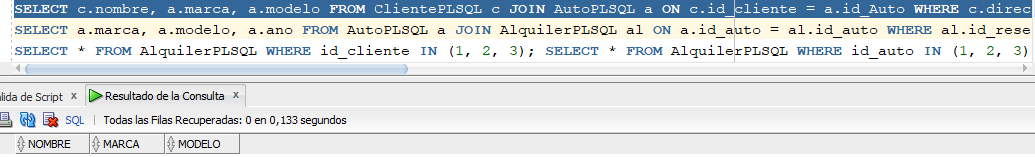
* SELECT \* FROM ClientePLSQL WHERE nombre LIKE '%Juan%'; SELECT a.marca, a.modelo, a.ano FROM AutoPLSQL a WHERE precio < 10000;

No existe un precio

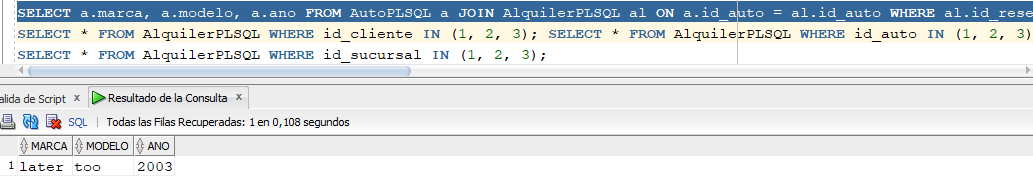
* SELECT \* FROM AlquilerPLSQL WHERE fecha\_inicio BETWEEN '2023-09-01' AND '2023-09-30';



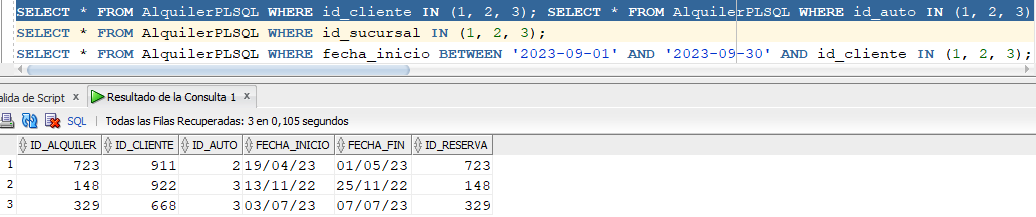
* SELECT c.nombre, a.marca, a.modelo FROM ClientePLSQL c JOIN AlquilerPLSQL a ON c.id\_cliente = a.id\_cliente WHERE c.direccion LIKE '%Bogotá%';



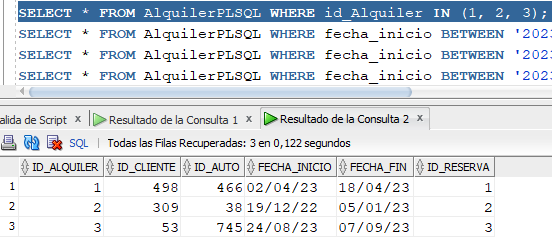
* SELECT a.marca, a.modelo, a.ano FROM AutoPLSQL a JOIN AlquilerPLSQL al ON a.id\_auto = al.id\_auto WHERE al.id\_reserva = 1;



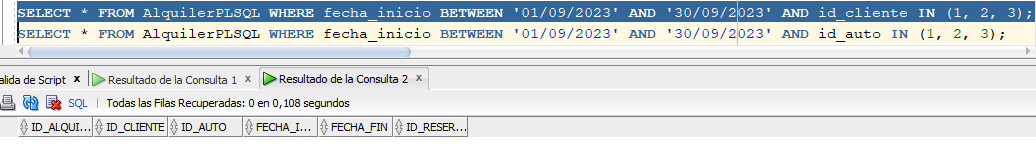
* SELECT \* FROM AlquilerPLSQL WHERE id\_cliente IN (1, 2, 3); SELECT \* FROM AlquilerPLSQL WHERE id\_auto IN (1, 2, 3);



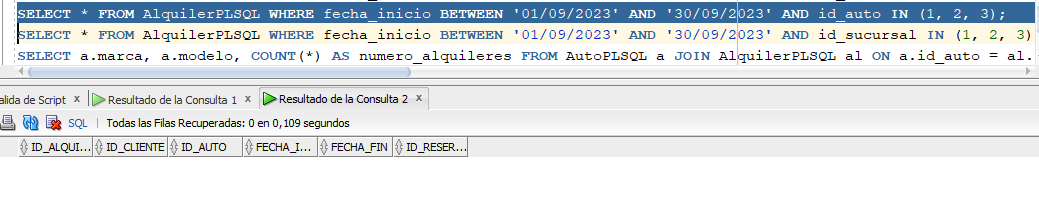
* SELECT \* FROM AlquilerPLSQL WHERE id\_sucursal IN (1, 2, 3);



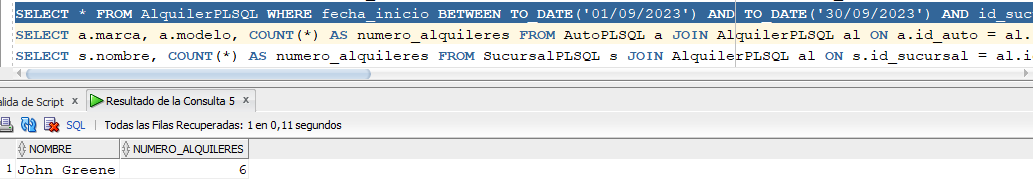
* SELECT \* FROM AlquilerPLSQL WHERE fecha\_inicio BETWEEN '2023-09-01' AND '2023-09-30' AND id\_cliente IN (1, 2, 3);



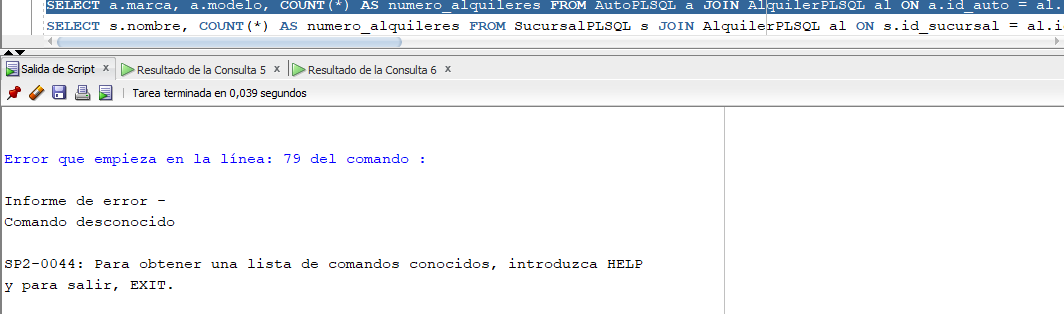
* SELECT \* FROM AlquilerPLSQL WHERE fecha\_inicio BETWEEN '2023-09-01' AND '2023-09-30' AND id\_auto IN (1, 2, 3);



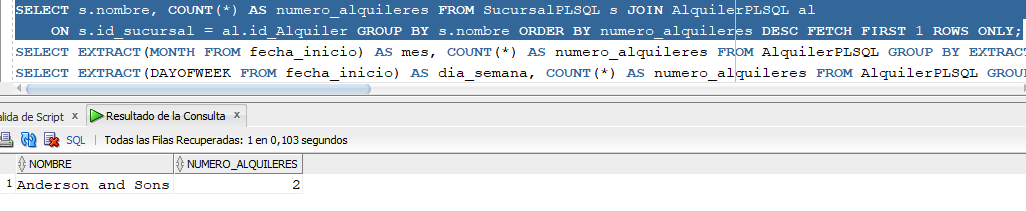
* SELECT \* FROM AlquilerPLSQL WHERE fecha\_inicio BETWEEN '2023-09-01' AND '2023-09-30' AND id\_sucursal IN (1, 2, 3); SELECT c.nombre, COUNT(\*) AS numero\_alquileres FROM ClientePLSQL c JOIN AlquilerPLSQL a ON c.id\_cliente = a.id\_cliente GROUP BY c.nombre ORDER BY numero\_alquileres DESC LIMIT 1;



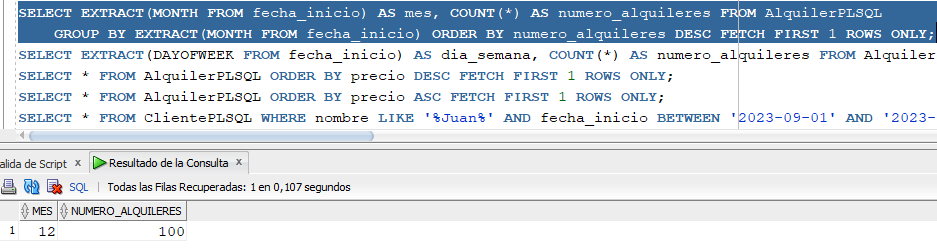
* SELECT a.marca, a.modelo, COUNT(\*) AS numero\_alquileres FROM AutoPLSQL a JOIN AlquilerPLSQL al ON a.id\_auto = al.id\_auto GROUP BY a.marca, a.modelo ORDER BY numero\_alquileres DESC LIMIT 1;



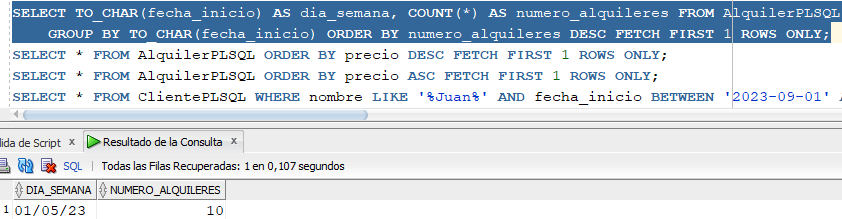
* SELECT s.nombre, COUNT(\*) AS numero\_alquileres FROM SucursalPLSQL s JOIN AlquilerPLSQL al ON s.id\_sucursal = al.id\_sucursal GROUP BY s.nombre ORDER BY numero\_alquileres DESC LIMIT 1;



* SELECT EXTRACT(MONTH FROM fecha\_inicio) AS mes, COUNT(\*) AS numero\_alquileres FROM AlquilerPLSQL GROUP BY EXTRACT(MONTH FROM fecha\_inicio) ORDER BY numero\_alquileres DESC LIMIT 1;



* SELECT EXTRACT(DAYOFWEEK FROM fecha\_inicio) AS dia\_semana, COUNT(\*) AS numero\_alquileres FROM AlquilerPLSQL GROUP BY EXTRACT(DAYOFWEEK FROM fecha\_inicio) ORDER BY numero\_alquileres DESC LIMIT 1;



* SELECT \* FROM AlquilerPLSQL ORDER BY precio DESC LIMIT 1;

No existe un precio

* SELECT \* FROM AlquilerPLSQL ORDER BY precio ASC LIMIT 1;

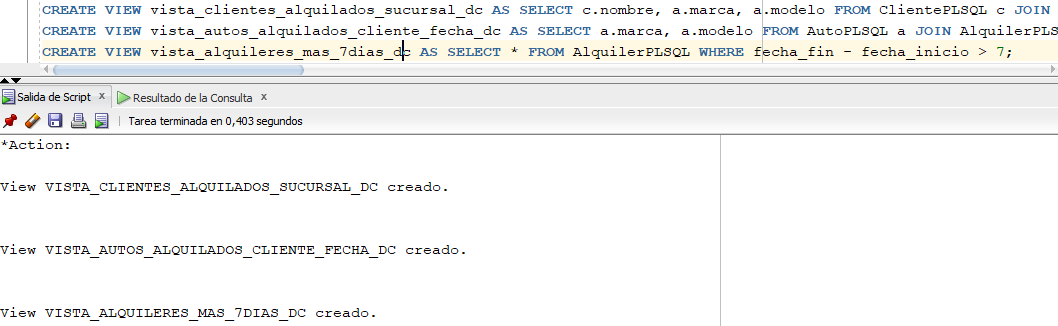
No existe un precio

* SELECT \* FROM ClientePLSQL WHERE nombre LIKE '%Juan%' AND fecha\_inicio BETWEEN '2023-09-01' AND '2023-09-30'; SELECT a.marca, a.modelo, a.ano FROM AutoPLSQL a WHERE precio < 10000 AND fecha\_inicio BETWEEN '2023-09-01' AND '2023-09-30';

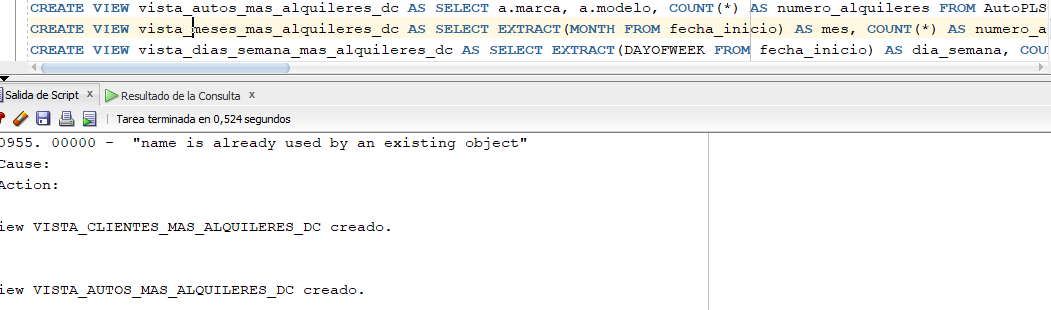
No existe un precio

**EJERCICIOS TERCER CICLO (81-90):**

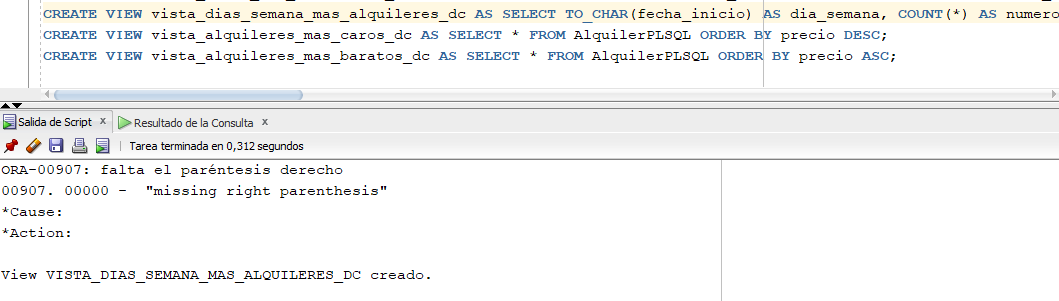
* CREATE VIEW vista\_clientes\_alquilados\_sucursal AS SELECT c.nombre, a.marca, a.modelo FROM ClientePLSQL c JOIN AlquilerPLSQL a ON c.id\_cliente = a.id\_cliente JOIN SucursalPLSQL s ON a.id\_sucursal = s.id\_sucursal WHERE s.nombre = 'Sucursal Central';
* CREATE VIEW vista\_autos\_alquilados\_cliente\_fecha AS SELECT a.marca, a.modelo FROM AutoPLSQL a JOIN AlquilerPLSQL al ON a.id\_auto = al.id\_auto WHERE al.id\_cliente = 1 AND al.fecha\_inicio = '2023-09-27';
* CREATE VIEW vista\_alquileres\_mas\_7dias AS SELECT \* FROM AlquilerPLSQL WHERE fecha\_fin - fecha\_inicio > 7; CREATE VIEW vista\_clientes\_mas\_alquileres AS SELECT c.nombre, COUNT(\*) AS numero\_alquileres FROM ClientePLSQL c JOIN AlquilerPLSQL a ON c.id\_cliente = a.id\_cliente GROUP BY c.nombre ORDER BY numero\_alquileres DESC;



* CREATE VIEW vista\_autos\_mas\_alquileres AS SELECT a.marca, a.modelo, COUNT(\*) AS numero\_alquileres FROM AutoPLSQL a JOIN AlquilerPLSQL al ON a.id\_auto = al.id\_auto GROUP BY a.marca, a.modelo ORDER BY numero\_alquileres DESC; CREATE VIEW vista\_sucursales\_mas\_alquileres AS SELECT s.nombre, COUNT(\*) AS numero\_alquileres FROM SucursalPLSQL s JOIN AlquilerPLSQL al ON s.id\_sucursal = al.id\_sucursal GROUP BY s.nombre ORDER BY numero\_alquileres DESC;
* CREATE VIEW vista\_meses\_mas\_alquileres AS SELECT EXTRACT(MONTH FROM fecha\_inicio) AS mes, COUNT(\*) AS numero\_alquileres FROM AlquilerPLSQL GROUP BY EXTRACT(MONTH FROM fecha\_inicio) ORDER BY numero\_alquileres DESC;



* CREATE VIEW vista\_dias\_semana\_mas\_alquileres AS SELECT EXTRACT(DAYOFWEEK FROM fecha\_inicio) AS dia\_semana, COUNT(\*) AS numero\_alquileres FROM AlquilerPLSQL GROUP BY EXTRACT(DAYOFWEEK FROM fecha\_inicio) ORDER BY numero\_alquileres DESC;

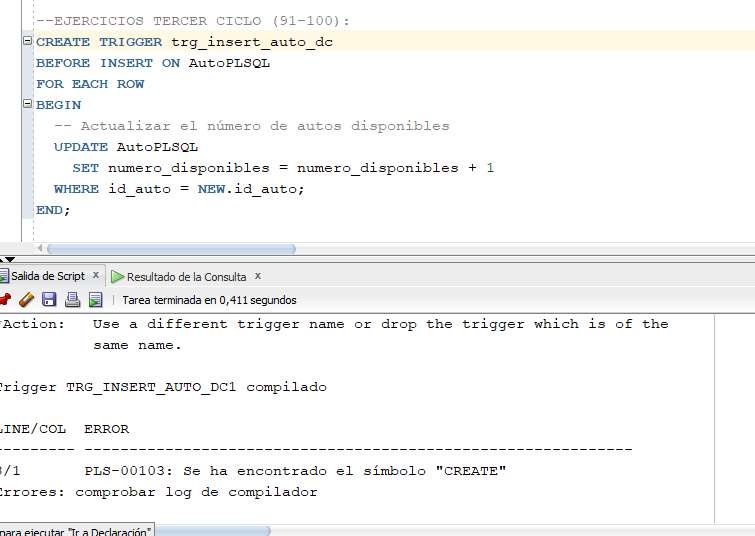


* CREATE VIEW vista\_alquileres\_mas\_caros AS SELECT \* FROM AlquilerPLSQL ORDER BY precio DESC;
* CREATE VIEW vista\_alquileres\_mas\_baratos AS SELECT \* FROM AlquilerPLSQL ORDER BY precio ASC;

No existe un precio

**EJERCICIOS TERCER CICLO (91-100):**

| CREATE TRIGGER trg\_insert\_auto  BEFORE INSERT ON AutoPLSQL  FOR EACH ROW  BEGIN  -- Actualizar el número de autos disponibles  UPDATE AutoPLSQL  SET numero\_disponibles = numero\_disponibles + 1  WHERE id\_auto = NEW.id\_auto;  END;  CREATE TRIGGER trg\_delete\_auto  BEFORE DELETE ON AutoPLSQL  FOR EACH ROW  BEGIN  -- Actualizar el número de autos disponibles  UPDATE AutoPLSQL  SET numero\_disponibles = numero\_disponibles - 1  WHERE id\_auto = OLD.id\_auto;  END;  CREATE TRIGGER trg\_update\_auto  BEFORE UPDATE ON AutoPLSQL  FOR EACH ROW  BEGIN  -- Actualizar el número de autos disponibles  IF NEW.numero\_disponibles != OLD.numero\_disponibles THEN  UPDATE AutoPLSQL  SET numero\_disponibles = NEW.numero\_disponibles  WHERE id\_auto = NEW.id\_auto;  END IF;  END;  CREATE TRIGGER trg\_insert\_cliente  BEFORE INSERT ON ClientePLSQL  FOR EACH ROW  BEGIN  -- Actualizar el número de clientes  UPDATE ClientePLSQL  SET numero\_clientes = numero\_clientes + 1;  END;  CREATE TRIGGER trg\_delete\_cliente  BEFORE DELETE ON ClientePLSQL  FOR EACH ROW  BEGIN  -- Actualizar el número de clientes  UPDATE ClientePLSQL  SET numero\_clientes = numero\_clientes - 1;  END;  CREATE TRIGGER trg\_update\_cliente  BEFORE UPDATE ON ClientePLSQL  FOR EACH ROW  BEGIN  -- Actualizar el número de clientes  IF NEW.numero\_alquileres != OLD.numero\_alquileres THEN  UPDATE ClientePLSQL  SET numero\_alquileres = NEW.numero\_alquileres  WHERE id\_cliente = NEW.id\_cliente;  END IF;  END; |
| --- |
| CREATE PROCEDURE proc\_calcular\_precio\_alquiler  (  IN id\_alquiler INT,  IN id\_auto INT,  IN fecha\_inicio DATE,  IN fecha\_fin DATE  )  AS  BEGIN  -- Calcular el precio del alquiler  DECLARE  precio\_base NUMERIC(10, 2);  dias\_alquiler INT;  BEGIN  precio\_base := (SELECT precio FROM AutoPLSQL WHERE id\_auto = id\_auto);  dias\_alquiler := (fecha\_fin - fecha\_inicio) + 1;  SET NEW.precio = precio\_base \* dias\_alquiler;  END;  END;  CREATE PROCEDURE proc\_listar\_alquileres\_cliente  (  IN id\_cliente INT  )  AS  BEGIN  -- Listar los alquileres del cliente  SELECT \*  FROM AlquilerPLSQL  WHERE id\_cliente = id\_cliente;  END;  CREATE PROCEDURE proc\_listar\_autos\_sucursal  (  IN id\_sucursal INT  )  AS  BEGIN  -- Listar los autos de la sucursal  SELECT \*  FROM AutoPLSQL  WHERE id\_sucursal = id\_sucursal;  END;  CREATE PROCEDURE proc\_agregar\_auto  (  IN marca VARCHAR(255),  IN modelo VARCHAR(255),  IN ano INT,  IN numero\_disponibles INT  )  AS  BEGIN  -- Insertar un nuevo auto  INSERT INTO AutoPLSQL (marca, modelo, ano, numero\_disponibles)  VALUES (marca, modelo, ano, numero\_disponibles);  END;  CREATE PROCEDURE proc\_eliminar\_auto  (  IN id\_auto INT  )  AS  BEGIN  -- Eliminar un auto  DELETE FROM AutoPLSQL  WHERE id\_auto = id\_auto;  END; |



Daniel Andrés Cervantes Flórez