**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 100 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.
* En caso de que una sentencia no genere ningún resultado, explicar la razón del comportamiento de esa sentencia

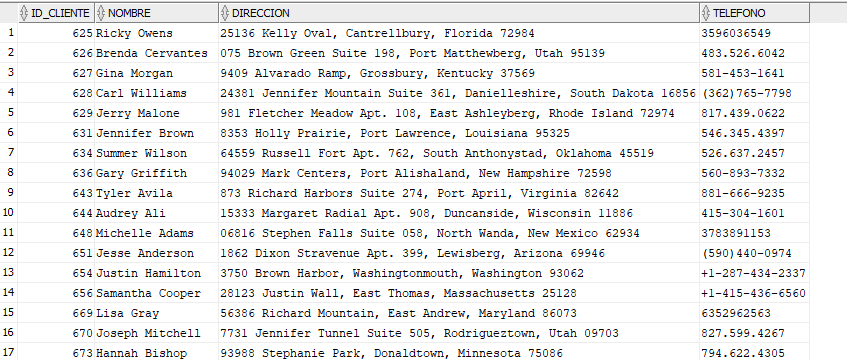
**ESTRUCTURA DE LA BASE DE DATOS:**

|  |
| --- |
| -- Tabla de clientes  CREATE TABLE ClientePLSQL (  id\_cliente NUMBER PRIMARY KEY,  nombre VARCHAR2(50),  direccion VARCHAR2(100),  telefono VARCHAR2(15)  );  -- Tabla de autos  CREATE TABLE AutoPLSQL (  id\_auto NUMBER PRIMARY KEY,  marca VARCHAR2(50),  modelo VARCHAR2(50),  ano NUMBER  );  -- Tabla de alquileres  CREATE TABLE AlquilerPLSQL (  id\_alquiler NUMBER PRIMARY KEY,  id\_cliente NUMBER,  id\_auto NUMBER,  fecha\_inicio DATE,  fecha\_fin DATE,  id\_reserva NUMBER,  FOREIGN KEY (id\_cliente) REFERENCES Cliente(id\_cliente),  FOREIGN KEY (id\_auto) REFERENCES Auto(id\_auto),  FOREIGN KEY (id\_reserva) REFERENCES Reserva(id\_reserva)  );  -- Tabla de sucursales  CREATE TABLE SucursalPLSQL (  id\_sucursal NUMBER PRIMARY KEY,  nombre VARCHAR2(50),  ciudad VARCHAR2(50),  pais VARCHAR2(50)  );  -- Tabla de reservas  CREATE TABLE ReservaPLSQL (  id\_reserva NUMBER PRIMARY KEY,  id\_cliente NUMBER,  id\_sucursal NUMBER,  fecha\_reserva DATE,  FOREIGN KEY (id\_cliente) REFERENCES Cliente(id\_cliente),  FOREIGN KEY (id\_sucursal) REFERENCES Sucursal(id\_sucursal)  ); |
| * Cliente: Almacena información sobre los clientes, como su nombre, dirección y número de teléfono. * Auto: Almacena información sobre los autos, como su marca, modelo y año. * Alquiler: Almacena información sobre los alquileres, como la fecha de inicio, la fecha de finalización y el auto alquilado. * Sucursal: Almacena información sobre las sucursales, como su nombre, ciudad y país. * Reserva: Almacena información sobre las reservas, como la fecha de la reserva y la sucursal en la que se realizó la reserva. |

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

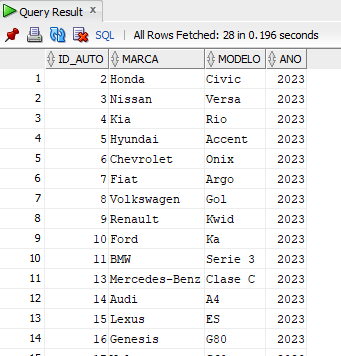
1. **Consultas Básicas:**
   * Mostrar todos los clientes en la tabla "Cliente".





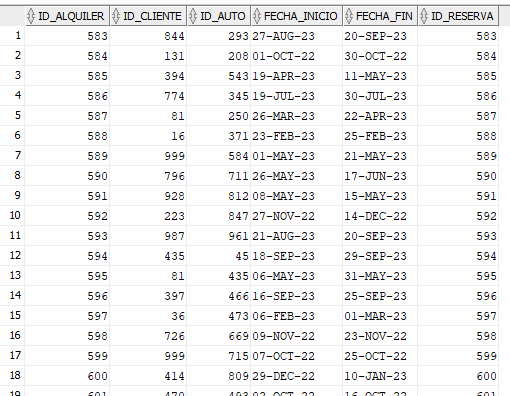
* + Mostrar todos los autos en la tabla "Auto".





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





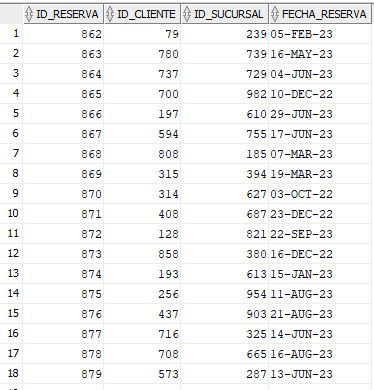
* + Mostrar todas las sucursales en la tabla "Sucursal".





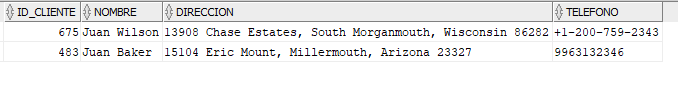
* + Mostrar todas las reservas en la tabla "Reserva".



****

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

****

****

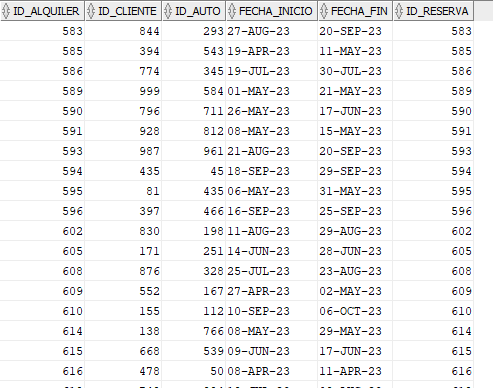
* + Mostrar los autos de marca "Toyota".





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





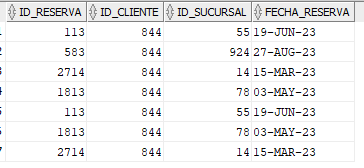
* + Mostrar las sucursales ubicadas en "Madrid".



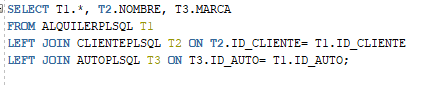


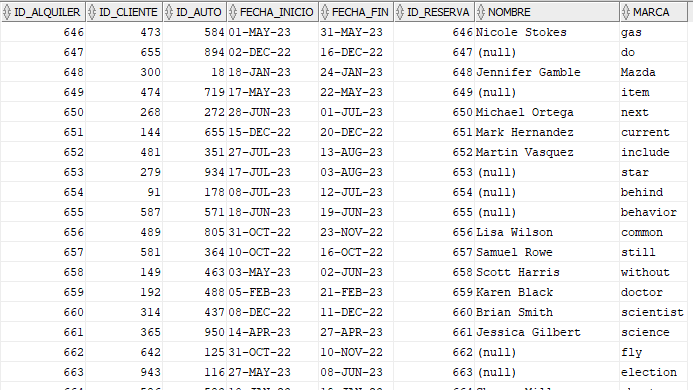
* + Mostrar las reservas realizadas por un cliente específico.



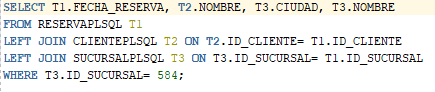


1. **Join y Relaciones:**
   * Mostrar los alquileres con los nombres de los clientes y las marcas de los autos.





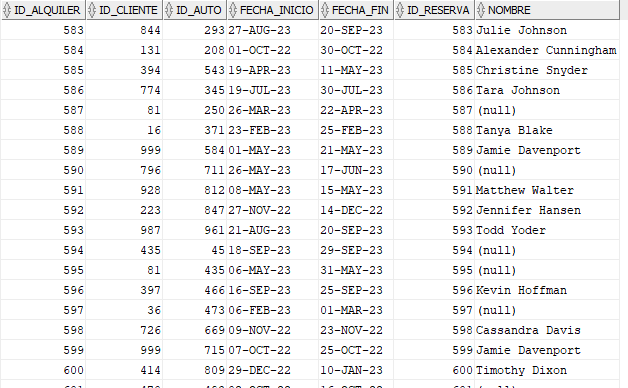
* + Mostrar los clientes que han realizado reservas en una sucursal específica.



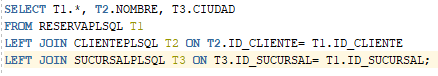


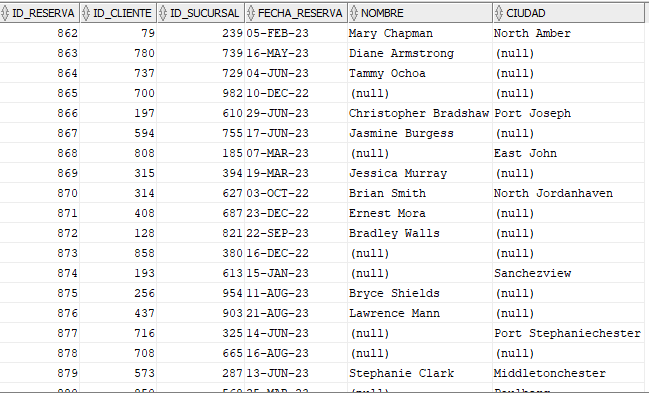
* + Mostrar los autos que han sido alquilados junto con los nombres de los clientes.





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





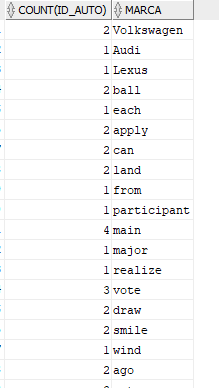
* + Mostrar los clientes que no han realizado ninguna reserva.





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



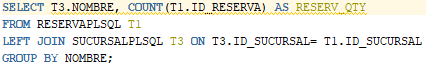


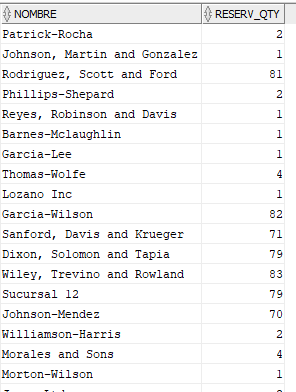
* + Calcular la duración promedio de los alquileres.



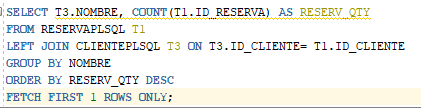


* + Mostrar el número total de reservas realizadas en cada sucursal.

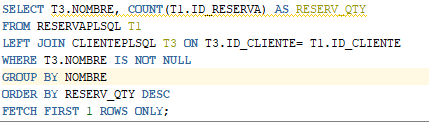




* + Encontrar el cliente que ha realizado la mayor cantidad de alquileres.







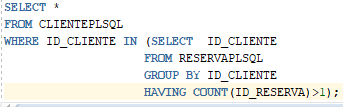


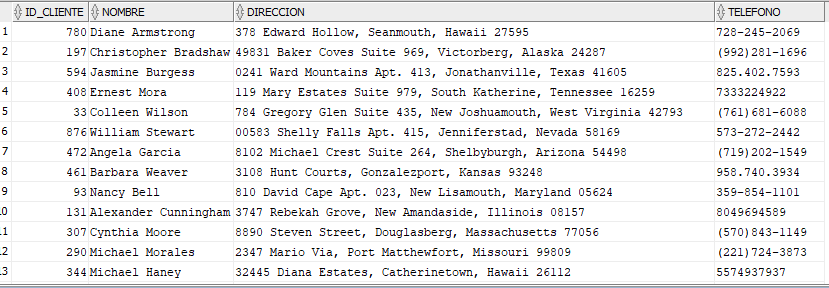
* + Calcular el promedio de años de los autos en la tabla "Auto".



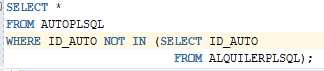


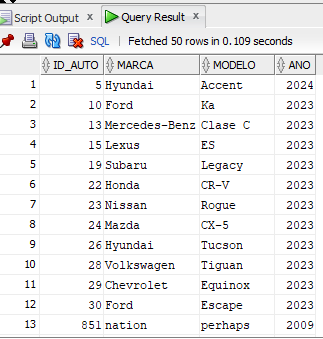
1. **Subconsultas**:
   * Mostrar los clientes que han realizado al menos una reserva.



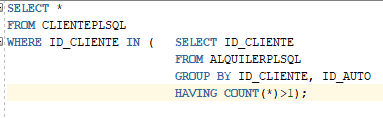


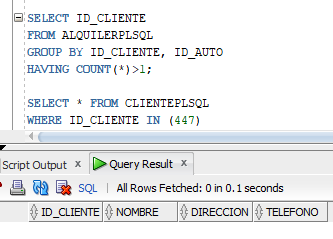
* + Mostrar los autos que no han sido alquilados aún.





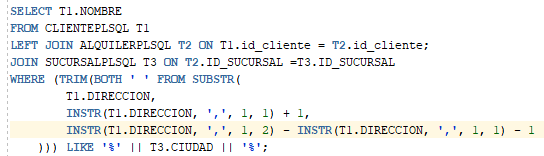
* + Encontrar los clientes que han alquilado el mismo auto más de una vez.



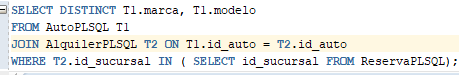


El cliente fue eliminado de la base de datos de clientes.

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

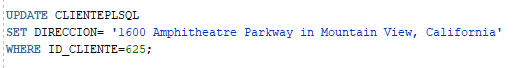


* + Encontrar los autos que han sido alquilados en la misma sucursal donde se realizó una reserva.



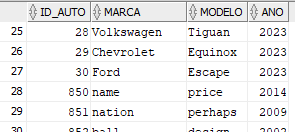
1. **Actualizaciones y Eliminaciones:**
   * Actualizar la dirección de un cliente específico.






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











* + Marcar una reserva como completada actualizando la fecha de fin.





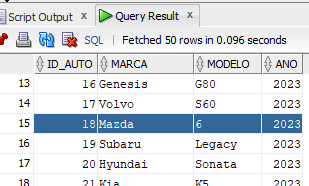


* + Eliminar todas las reservas realizadas por un cliente específico.





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





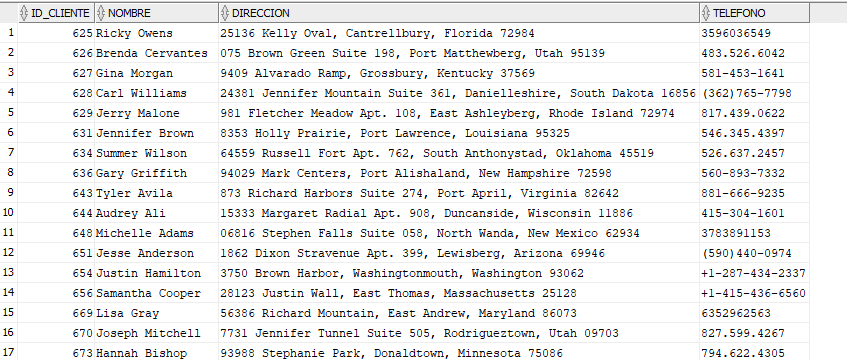




**EJERCICIOS SEGUNDO CICLO (31-80):**

* SELECT \* FROM ClientePLSQL;

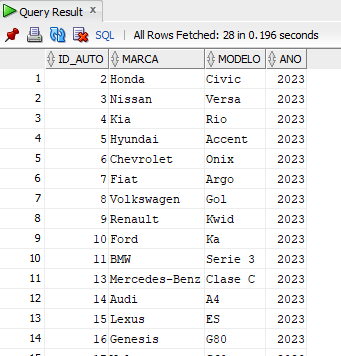




Muestra todos los elementos de la tabla clientes

* SELECT \* FROM AutoPLSQL;

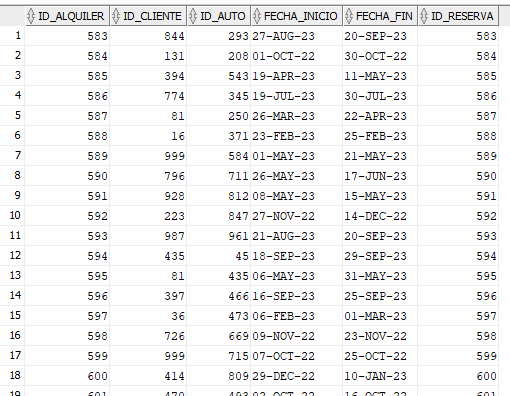




Muestra todos los elementos de la tabla autos

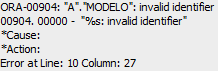
* SELECT \* FROM AlquilerPLSQL;





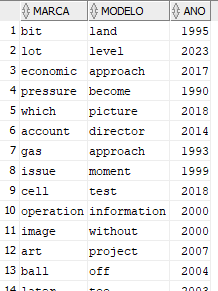
Muestra todos los elementos de la tabla alquiler

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



La columna modelo no existe en la tabla alquiler

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



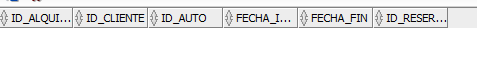
Los autos que han sido alquilados

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



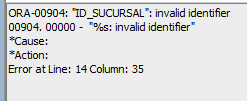
No hay alquileres para el cliente cuyo ID es 1

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



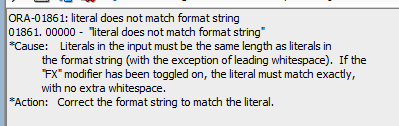
No hay alquileres para el auto cuyo ID es 1

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



No existe la columna ID sucursal en la tabla alquiler

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



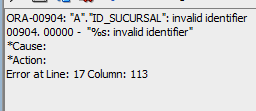
El formato de fecha usado para filtrar no es compatible con el formato de fecha de los datos

* SELECT COUNT(\*) FROM AlquilerPLSQL;



El total de alquileres realizados

* 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';



La columna sucursal ID no existe en la tabla alquiler

SELECT c.nombre

FROM ClientePLSQL c

JOIN AlquilerPLSQL a ON c.id\_cliente = a.id\_cliente

JOIN RESERVAPLSQL R ON a.id\_reserva = R.id\_reserva

JOIN SucursalPLSQL s ON s.id\_sucursal = s.id\_sucursal

WHERE s.nombre = 'Sucursal Central';

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



Selecciona las columnas Marca y modelo de la tabla Auto

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



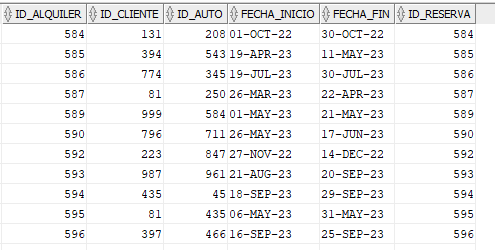


SELECT a.marca, a.modelo FROM AutoPLSQL a

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

No hay autos con fecha de inicio de alquiler del 27 de septiembre de 2023.

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



Seleccionar todos los alquileres con una duración mayor a 7 días

* 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;

Cliente que tiene la mayor cantidad de alquileres, no obstante, limit 1 no puede ser usado.





* 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;

Identifica el auto más popular en términos de alquileres, mostrando así la marca y modelo del auto más frecuentemente alquilado y el número de veces que fue alquilado. Limit 1 no puede ser usado.





* 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;

Sucursal con el mayor número de alquileres. Limit 1 no puede ser usado.



* 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;

Mes con el mayor número de alquileres. Limit 1 no puede ser usado.





* 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;

Día de la semana en el que se realizan más alquileres. Limit 1 no puede ser usado.

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

Alquiler más costoso. Limit 1 no puede ser usado.

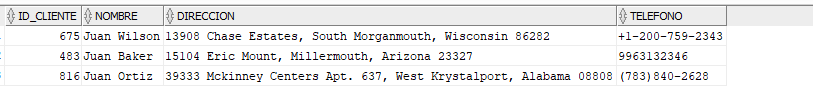


* SELECT \* FROM AlquilerPLSQL ORDER BY precio ASC LIMIT 1;

Alquiler más económico. Limit 1 no puede ser usado.



* SELECT \* FROM ClientePLSQL WHERE nombre LIKE '%Juan%';

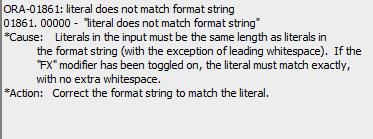


Seleccionar todos los datos de clientes cuyo nombre sea o contenga Juan

* SELECT a.marca, a.modelo, a.ano FROM AutoPLSQL a WHERE precio < 10000;

Precio no se encuentra entre las columnas de la tabla auto.

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

SE

* 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%';

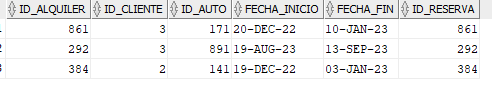
Marca y modelo pertenecen a la tabla auto, no a la tabla alquiler.

* 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;



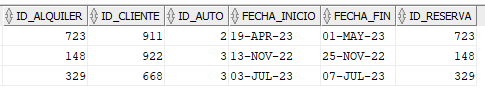
Año, marca y modelo del auto alquilado para la reserva cuyo ID es igual a 1

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



Alquileres realizados por los clientes cuyos ID son 1 2 o 3

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



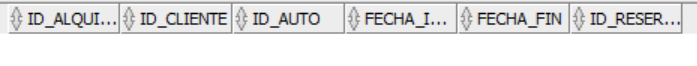
Alquileres realizados de los autos cuyos ID son 1 2 o 3

* 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);

El formato de fecha no coincide.

Alquileres realizados por los clientes cuyos ID son 1 2 o 3 y realizaron alquileres entre el 1 de septiembre y el 30 de septiembre





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

El formato de fecha no coincide.

Alquileres realizados para los autos cuyos ID son 1 2 o 3 y realizaron alquileres entre el 1 de septiembre y el 30 de septiembre

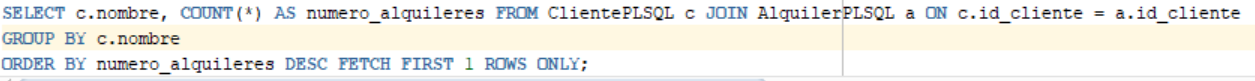


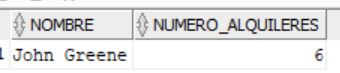


* SELECT c.nombre, COUNT(\*) AS numero\_alquileres FROM ClientePLSQL c JOIN AlquilerPLSQL a ON c.id\_cliente = a.id\_cliente GROUP BY c.nombre ORDER BY numero\_alquileres DESC LIMIT 1;

Limit 1 no puede ser usado.

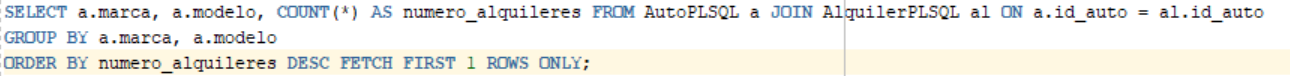
Seleccionar el cliente con mayor número de alquileres.

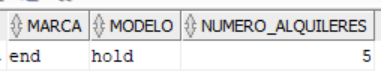




* SELECT a.marca, a.modelo, COUNT(\*) AS numero\_alquileres FROM AutoPLSQL a JOIN AlquilerPLSQL al ON a.id\_auto = al.id\_auto GROUP BY a.marca, a.modelo ORDER BY numero\_alquileres DESC LIMIT 1;

Limit 1 no puede ser usado.





Marca y modelo del auto más alquilado.

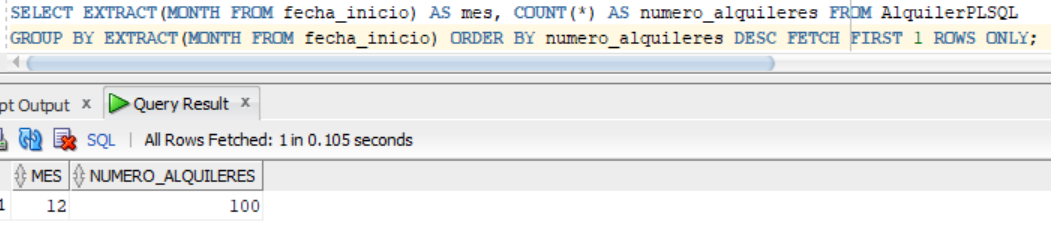
* SELECT s.nombre, COUNT(\*) AS numero\_alquileres FROM SucursalPLSQL s JOIN AlquilerPLSQL al ON s.id\_sucursal = al.id\_sucursal GROUP BY s.nombre ORDER BY numero\_alquileres DESC LIMIT 1;
* Limit 1 no puede ser usado.



Sucursal con mayor número de alquileres.

* 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;

Limit 1 no puede ser usado.

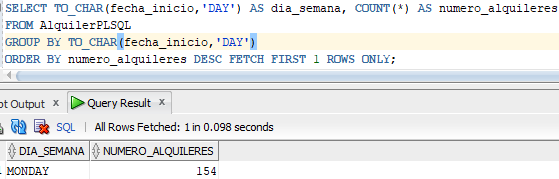


Mes con mayor número de alquileres

* 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;

Limit 1 no puede ser usado.

DayOfweek no puede ser usado.



Día de la semana con mayor número de alquileres.

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

Limit 1 no puede ser usado, la tabla alquiler no tiene una columna precio

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

Limit 1 no puede ser usado, la tabla alquiler no tiene una columna precio

* SELECT \* FROM ClientePLSQL WHERE nombre LIKE '%Juan%' AND fecha\_inicio BETWEEN '2023-09-01' AND '2023-09-30';

Falta el join entre las tablas de cliente y alquiler.

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

Falta el join entre las tablas de auto y alquiler, la tabla alquiler no tiene una columna 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';

Crea una vista con el nombre del cliente la marca y modelo alquilado en la 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';





Crea una vista de la marca y modelo del auto rentado por el cliente cuyo ID es 1 y la fecha de inicio del alquiler es el 27 de septiembre de 2023.

* CREATE VIEW vista\_alquileres\_mas\_7dias AS SELECT \* FROM AlquilerPLSQL WHERE fecha\_fin - fecha\_inicio > 7;





Crea una vista de todos los alquileres cuya duración fue superior a 7 días.

* 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;





Crea una vista de los clientes y el número de alquileres ordenando de mayor a menor.

* 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;





Crea una vista de los autos por modelo y marca y la cantidad de veces que han sido alquilados y los ordena de mayor a menor.

* 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;

Crea una vista de las sucursales y la cantidad de alquileres por sucursal y ordena de mayor a menor.

* 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;





Crea una vista de la cantidad de alquileres realizados por mes y los ordena de mayor a menor

* 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;

DayOfweek no puede ser usado.





Crea una vista de la cantidad de alquileres por día de la semana y los ordena de mayor a menor.

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

Crea una vista de los precios de los alquileres los ordena de mayor a menor.

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

Crea una vista de los precios de los alquileres los ordena de menor a mayor.

**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;  Este trigger funciona durante el proceso de inserción, actualiza el contador de autos disponible en la tabla "AutoPLSQL". Sin embargo, debido a que la columna "número\_disponibles" no existe en la estructura de la tabla "AutoPLSQL", el desencadenante presenta un error relacionado con la actualización de la columna.  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;  Este trigger funciona durante el proceso de eliminación, actualiza el contador de autos disponible en la tabla "AutoPLSQL". Sin embargo, debido a que la columna "número\_disponibles" no existe en la estructura de la tabla "AutoPLSQL", el desencadenante presenta un error relacionado con la actualización de la columna.  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;  Este trigger funciona durante el proceso de actualización, actualiza el contador de autos disponible en la tabla "AutoPLSQL" cada vez que se ingresa un nuevo auto. Sin embargo, debido a que la columna "número\_disponibles" no existe en la estructura de la tabla "AutoPLSQL", el desencadenante presenta un error relacionado con la actualización de la columna.  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;  Este trigger funciona durante el proceso de inserción, actualiza el contador de clientes en la tabla " ClientePLSQL ". Sin embargo, debido a que la columna " numero\_clientes " no existe en la estructura de la tabla " ClientePLSQL", el desencadenante presenta un error relacionado con la actualización de la columna.  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;  Este trigger funciona durante el proceso de eliminación, actualiza el contador de clientes en la tabla " ClientePLSQL ". Sin embargo, debido a que la columna " numero\_clientes " no existe en la estructura de la tabla " ClientePLSQL", el desencadenante presenta un error relacionado con la actualización de la columna.  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;  Este trigger funciona durante el proceso de actualización, actualiza el contador de clientes en la tabla " ClientePLSQL ". Sin embargo, debido a que la columna " numero\_clientes " no existe en la estructura de la tabla " ClientePLSQL", el desencadenante presenta un error relacionado con la actualización de la columna. |
| 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;  La sintaxis no es correcta, se están mezclando Oracle PL/SQL y MySQL y adicionalmente no existe una columna precio en la tabla auto que permita calcular el costo total del alquiler.  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;  La sintaxis no es correcta, se están mezclando Oracle PL/SQL y otro DBMS, se puede corregir de la siguiente forma      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;  Error de sintaxis nuevamente, la columna id\_sucursal no existe en la tabla de autos.  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;  Error de sintaxis nuevamente, numero\_disponibles no existe en la tabla auto  CREATE PROCEDURE proc\_eliminar\_auto  (  IN id\_auto INT  )  AS  BEGIN  -- Eliminar un auto  DELETE FROM AutoPLSQL  WHERE id\_auto = id\_auto;  END;  Error de sintaxis nuevamente, se corrige así: |