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".

SELECT * FROM ClientePLSQL;

Esta consulta selecciona todos los registros de la tabla ClientePLSQL. Devolverá todas las columnas y todas las filas de la tabla ClientePLSQL. Los resultados mostrarán la información de todos los clientes, incluyendo su ID, nombre, dirección y teléfono.

- Mostrar todos los autos en la tabla "Auto".
-

SELECT * FROM AutoPLSQL;

Esta consulta selecciona todos los registros de la tabla AutoPLSQL. Devolverá todas las columnas y todas las filas de la tabla AutoPLSQL. Los resultados mostrarán la información de todos los autos, incluyendo su ID, marca, modelo y año.

- Mostrar todos los alquileres en la tabla "Alquiler".
-

SELECT * FROM AlquilerPLSQL;

Esta consulta selecciona todos los registros de la tabla AlquilerPLSQL. Devolverá todas las columnas y todas las filas de la tabla AlquilerPLSQL. Los resultados mostrarán la información de todos los alquileres, incluyendo su ID, fechas de inicio y fin, ID del cliente, ID del auto y ID de la reserva.

- Mostrar todas las sucursales en la tabla "Sucursal".

SELECT * FROM SucursalPLSQL;

Esta consulta selecciona todos los registros de la tabla SucursalPLSQL. Devolverá todas las columnas y todas las filas de la tabla SucursalPLSQL. Los resultados mostrarán la información de todas las sucursales, incluyendo su ID, nombre, ciudad y país.

- Mostrar todas las reservas en la tabla "Reserva".
-

SELECT * FROM ReservaPLSQL;

Esta consulta selecciona todos los registros de la tabla ReservaPLSQL. Devolverá todas las columnas y todas las filas de la tabla ReservaPLSQL. Los resultados mostrarán la información de todas las reservas, incluyendo su ID, fecha de reserva, ID del cliente y ID de la sucursal.

2. Filtros y Ordenamiento:
    - Mostrar los clientes que se llaman "Juan".

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

Esta consulta selecciona todos los registros de la tabla ClientePLSQL donde el nombre del cliente contiene la cadena "Juan". La cláusula LIKE '%Juan%' busca cualquier nombre que contenga la secuencia

"Juan" en cualquier posición. Los resultados mostrarán información sobre los clientes cuyos nombres cumplen con ese criterio.

- Mostrar los autos de marca "Toyota".

```
SELECT * FROM AutoPLSQL WHERE marca = 'Toyota';
```

Esta consulta selecciona todos los registros de la tabla AutoPLSQL donde la marca del auto es "Toyota". Los resultados mostrarán información sobre los autos que son de la marca Toyota.

- Mostrar los alquileres que ocurrieron después de una fecha específica.

```
SELECT * FROM AlquilerPLSQL WHERE fecha_inicio > '2023-11-01';
```

Esta consulta selecciona todos los registros de la tabla AlquilerPLSQL donde la fecha de inicio del alquiler es posterior al 1 de noviembre de 2023. Los resultados mostrarán información sobre los alquileres que comenzaron después de esa fecha.

- Mostrar las sucursales ubicadas en "Madrid".

```
SELECT * FROM SucursalPLSQL WHERE ciudad = 'Madrid';
```

Esta consulta selecciona todos los registros de la tabla SucursalPLSQL donde la ciudad de la sucursal es "Madrid". Los resultados mostrarán información sobre las sucursales ubicadas en la ciudad de Madrid.

- Mostrar las reservas realizadas por un cliente específico.

```
SELECT * FROM ReservaPLSQL WHERE id_cliente = 1026273328;
```

Esta consulta selecciona todos los registros de la tabla ReservaPLSQL donde el ID del cliente es igual a 1026273328. Los resultados mostrarán información sobre las reservas asociadas al cliente con ese ID.

3. Join y Relaciones:
   - Mostrar los alquileres con los nombres de los clientes y las marcas de los autos.

```
SELECT AlquilerPLSQL.id_alquiler, ClientePLSQL.nombre AS nombre_cliente, AutoPLSQL.marca
FROM AlquilerPLSQL
        JOIN ClientePLSQL ON AlquilerPLSQL.id_cliente = ClientePLSQL.id_cliente
   JOIN AutoPLSQL ON AlquilerPLSQL.id_auto = AutoPLSQL.id_auto;
```

Esta consulta selecciona el ID del alquiler, el nombre del cliente y la marca del auto asociado al alquiler. Utiliza JOIN para combinar las tablas AlquilerPLSQL, ClientePLSQL y AutoPLSQL utilizando las claves correspondientes (id_cliente e id_auto).

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

```
SELECT ClientePLSQL.id_cliente, ClientePLSQL.nombre AS nombre_cliente, ClientePLSQL.direccion,
ClientePLSQL.telefono
FROM ClientePLSQL
```

```
  LEFT JOIN ReservaPLSQL ON ClientePLSQL.id_cliente = ReservaPLSQL.id_cliente
  LEFT JOIN SucursalPLSQL ON ReservaPLSQL.id_sucursal = SucursalPLSQL.id_sucursal
    AND SucursalPLSQL.nombre = 'Centro'
WHERE SucursalPLSQL.id_sucursal IS NOT NULL;
```

Esta consulta muestra clientes que han realizado al menos una reserva en una sucursal específica ('Centro'). Utiliza LEFT JOIN para incluir todos los clientes, incluso aquellos que no han realizado reservas en esa sucursal.

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

```
SELECT AlquilerPLSQL.id_alquiler, ClientePLSQL.nombre AS nombre_cliente, AutoPLSQL.marca, AutoPLSQL.modelo, AutoPLSQL.anho
FROM AlquilerPLSQL
  JOIN ClientePLSQL ON AlquilerPLSQL.id_cliente = ClientePLSQL.id_cliente
  JOIN AutoPLSQL ON AlquilerPLSQL.id_auto = AutoPLSQL.id_auto;
```

Similar a la primera consulta, esta selecciona el ID del alquiler, el nombre del cliente, y detalles del auto asociado al alquiler.

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

```
SELECT ReservaPLSQL.id_reserva, ClientePLSQL.nombre AS nombre_cliente, SucursalPLSQL.ciudad, ReservaPLSQL.fecha_reserva
FROM ReservaPLSQL
  JOIN ClientePLSQL ON ReservaPLSQL.id_cliente = ClientePLSQL.id_cliente
  JOIN SucursalPLSQL ON ReservaPLSQL.id_sucursal = SucursalPLSQL.id_sucursal;
```

Esta consulta muestra detalles de reservas, incluyendo el ID de la reserva, el nombre del cliente, la ciudad de la sucursal y la fecha de reserva.

- Mostrar los clientes que no han realizado ninguna reserva.

```
SELECT ClientePLSQL.id_cliente, ClientePLSQL.nombre AS nombre_cliente
FROM ClientePLSQL
  LEFT JOIN ReservaPLSQL ON ClientePLSQL.id_cliente = ReservaPLSQL.id_cliente
WHERE ReservaPLSQL.id_reserva IS NULL;
```

Esta consulta muestra clientes que no han realizado ninguna reserva. Utiliza LEFT JOIN para incluir todos los clientes, y la condición WHERE ReservaPLSQL.id_reserva IS NULL filtra aquellos que no tienen coincidencias en la tabla de reservas.

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

```
SELECT marca,
  COUNT(*) AS cantidad_autos
FROM AutoPLSQL
```

```
GROUP BY marca;
```

Esta consulta utiliza GROUP BY para agrupar los resultados por la columna marca en la tabla AutoPLSQL. La función de agregación COUNT(*) cuenta la cantidad de autos para cada marca.

- Calcular la duración promedio de los alquileres.

```
SELECT AVG(DATEDIFF(fecha_fin, fecha_inicio)) AS duracion_promedio
FROM AlquilerPLSQL;
```

Esta consulta utiliza la función AVG para calcular el promedio de la diferencia en días entre las fechas de inicio y fin de los alquileres en la tabla AlquilerPLSQL.

- Mostrar el número total de reservas realizadas en cada sucursal.

```
SELECT id_sucursal,
   COUNT(*) AS total_reservas
FROM ReservaPLSQL
GROUP BY id_sucursal;
```

Esta consulta utiliza GROUP BY para agrupar los resultados por la columna id_sucursal en la tabla ReservaPLSQL. La función de agregación COUNT(*) cuenta la cantidad total de reservas para cada sucursal.

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

```
SELECT ClientePLSQL.id_cliente, ClientePLSQL.nombre AS nombre_cliente,
   COUNT(*) AS cantidad_alquileres
FROM ClientePLSQL
   JOIN AlquilerPLSQL ON ClientePLSQL.id_cliente = AlquilerPLSQL.id_cliente
GROUP BY ClientePLSQL.id_cliente, ClientePLSQL.nombre
ORDER BY cantidad_alquileres DESC LIMIT 1;
```

Esta consulta utiliza JOIN para combinar las tablas ClientePLSQL y AlquilerPLSQL utilizando la clave id_cliente. Luego, utiliza GROUP BY para agrupar por el ID del cliente y el nombre del cliente, y COUNT(*) para contar la cantidad de alquileres por cliente. La cláusula ORDER BY cantidad_alquileres DESC ordena los resultados en orden descendente según la cantidad de alquileres, y LIMIT 1 muestra solo el cliente con la mayor cantidad de alquileres.

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

```
SELECT AVG(anho) AS promedio_anhos
FROM AutoPLSQL;
```

Esta consulta utiliza la función AVG para calcular el promedio de los años de los autos en la tabla AutoPLSQL.

5. Subconsultas:
   - Mostrar los clientes que han realizado al menos una reserva.

```
SELECT DISTINCT ClientePLSQL.id_cliente, ClientePLSQL.nombre AS nombre_cliente
FROM ClientePLSQL
   JOIN ReservaPLSQL ON ClientePLSQL.id_cliente = ReservaPLSQL.id_cliente;
```

Esta consulta utiliza JOIN para combinar las tablas ClientePLSQL y ReservaPLSQL utilizando la clave id_cliente. La cláusula DISTINCT asegura que cada cliente aparezca solo una vez en los resultados, incluso si ha realizado múltiples reservas.

- Mostrar los autos que no han sido alquilados aún.

```
SELECT AutoPLSQL.id_auto, AutoPLSQL.marca, AutoPLSQL.modelo, AutoPLSQL.anho
FROM AutoPLSQL
   LEFT JOIN AlquilerPLSQL ON AutoPLSQL.id_auto = AlquilerPLSQL.id_auto
WHERE AlquilerPLSQL.id_alquiler IS NULL OR AlquilerPLSQL.id_auto IS NULL;
```

Esta consulta utiliza LEFT JOIN para incluir todos los autos de la tabla AutoPLSQL, incluso aquellos que no tienen coincidencias en la tabla de alquileres (AlquilerPLSQL). La condición en la cláusula WHERE selecciona solo los autos que no han sido alquilados aún.

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

```
SELECT ClientePLSQL.id_cliente, ClientePLSQL.nombre AS nombre_cliente, AlquilerPLSQL.id_auto,
   COUNT(*) AS cantidad_alquileres
FROM ClientePLSQL
   JOIN AlquilerPLSQL ON ClientePLSQL.id_cliente = AlquilerPLSQL.id_cliente
GROUP BY ClientePLSQL.id_cliente, ClientePLSQL.nombre, AlquilerPLSQL.id_auto
HAVING COUNT(*) > 1;
```

Esta consulta utiliza JOIN para combinar las tablas ClientePLSQL y AlquilerPLSQL utilizando la clave id_cliente. Luego, utiliza GROUP BY para agrupar por el ID del cliente, el nombre del cliente y el ID del auto, y HAVING COUNT(*) > 1 filtra solo aquellos grupos (clientes y autos) que han tenido más de un alquiler.

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

Esta consulta no puede realizarse debido a que se esta solicitando realizar un JOIN por medio del dato Ciudad aunque la tabla cliente no posee el campo, solo Direccion y mediante esto realizar la consulta no seria claro.

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

```
SELECT AutoPLSQL.id_auto, AutoPLSQL.marca, AutoPLSQL.modelo, AutoPLSQL.anho
FROM AutoPLSQL
   JOIN AlquilerPLSQL ON AutoPLSQL.id_auto = AlquilerPLSQL.id_auto
   JOIN ReservaPLSQL ON AlquilerPLSQL.id_reserva = ReservaPLSQL.id_reserva
   JOIN SucursalPLSQL ON ReservaPLSQL.id_sucursal = SucursalPLSQL.id_sucursal;
```

Esta consulta utiliza JOIN para combinar las tablas AutoPLSQL, AlquilerPLSQL, ReservaPLSQL y SucursalPLSQL utilizando las claves correspondientes (id_auto, id_reserva e id_sucursal). La consulta muestra autos que han sido alquilados en la misma sucursal donde se realizó una reserva.

6. Actualizaciones y Eliminaciones:
    - Actualizar la dirección de un cliente específico.

```
UPDATE ClientePLSQL
SET direccion = 'Nueva Direccion'
WHERE id_cliente = 123;
```

Esta consulta actualiza la columna direccion en la tabla ClientePLSQL estableciendo el valor 'Nueva Direccion' para el cliente cuyo ID es 123.

- Eliminar un auto de la tabla "Auto".

```
DELETE FROM AutoPLSQL
WHERE id_auto = 456;
```

Esta consulta elimina el registro correspondiente al auto con ID 456 de la tabla AutoPLSQL.

- Marcar una reserva como completada actualizando la fecha de fin.

Esta consulta no es realizable con los datos actuales debido a que la tabla ReservaPLSQL no posee el campo fecha_fin, solo posee el campo fecha_reserva, se puede dar fin a un alquiler en la tabla AlquilerPLSQL ya que esta si posee el campo fecha_fin.

- Eliminar todas las reservas realizadas por un cliente específico.

```
DELETE FROM ReservaPLSQL
WHERE id_cliente = 123;
```

Esta consulta elimina todos los registros de la tabla ReservaPLSQL donde el ID del cliente es 123, eliminando así todas las reservas asociadas a ese cliente.

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

```
UPDATE AutoPLSQL
SET anho = 2025
WHERE id_auto = 456;
```

Esta consulta actualiza la columna anho en la tabla AutoPLSQL estableciendo el valor 2025 para el auto cuyo ID es 456.

EJERCICIOS SEGUNDO CICLO (31-80):

- SELECT * FROM ClientePLSQL;

Esta consulta selecciona todos los registros de la tabla ClientePLSQL. Devolverá todas las columnas y todas las filas de la tabla ClientePLSQL. Los resultados mostrarán la información de todos los clientes, incluyendo su ID, nombre, dirección y teléfono.

- SELECT * FROM AutoPLSQL;

Esta consulta selecciona todos los registros de la tabla AutoPLSQL. Devolverá todas las columnas y todas las filas de la tabla AutoPLSQL. Los resultados mostrarán la información de todos los autos, incluyendo su ID, marca, modelo y año.

- SELECT * FROM AlquilerPLSQL;

Esta consulta selecciona todos los registros de la tabla AlquilerPLSQL. Devolverá todas las columnas y todas las filas de la tabla AlquilerPLSQL. Los resultados mostrarán la información de todos los alquileres, incluyendo su ID, fechas de inicio y fin, ID del cliente, ID del auto y ID de la reserva.

- SELECT c.nombre, a.marca, a.modelo FROM ClientePLSQL c JOIN AlquilerPLSQL a ON c.id_cliente = a.id_cliente;

Este codigo genera error debido a que falta la sentencia AS para abreviar los nombres de las tablas ClientePLSQL as c y AlquilerPLSQL as A.

- SELECT a.marca, a.modelo, a.ano FROM AutoPLSQL a JOIN AlquilerPLSQL al ON a.id_auto = al.id_auto;

Este codigo genera error debido a que falta la sentencia AS para abreviar los nombres de las tablas AutoPLSQL as a y AlquilerPLSQL as al.

- SELECT * FROM AlquilerPLSQL WHERE id_cliente = 1;

Selecciona todos los registros de la tabla AlquilerPLSQL para el cliente identificado con 1 en el campo id_cliente.

- SELECT * FROM AlquilerPLSQL WHERE id_auto = 1;

Selecciona todos los registros de la tabla AlquilerPLSQL para el auto identificado con 1 en el campo id_auto.

- SELECT * FROM AlquilerPLSQL WHERE id_sucursal = 1;

Selecciona todos los registros de la tabla AlquilerPLSQL para el auto identificado con 1 en el campo id_sucursal, aunque para este caso no es posible debido a que la tabla AlquilerPLSQL no tiene la información de sucursal.

- SELECT * FROM AlquilerPLSQL WHERE fecha_inicio = '2023-09-27';

Selecciona todos los registros de la tabla AlquilerPLSQL con fecha de inicio '2023-09-27'.

- SELECT COUNT(*) FROM AlquilerPLSQL;

Utilizando la función COUNT cuenta la cantidad de registros para la tabla 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';

De este codigo no se obtienen resultados debido a que falta la sentencia AS para abreviar las tablas ClientePLSQL AS c, AlquilerPLSQL AS a y SucursalPLSQL AS s adicionalmente la tabla AlquilerPLSQL no tiene la información id.sucursal.

- 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';

De este codigo no se obtienen resultados debido a que falta la sentencia AS para abreviar la tabla: AutoPLSQL AS a, corrigiendo esto se obtienen los campos marca y modelo para todos los autos que estén registrados en la tabla AlquilerPLSQL donde la fecha de inicio sea '2023-09-27'.

- SELECT * FROM AlquilerPLSQL WHERE fecha_fin - fecha_inicio > 7;

Para esta ocasión el resultado son los alquileres que registres mas de 7 dias desde su fecha_inicio hasta su fecha_fin.

- 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;

ClientePLSQL se abrevia como c, AlquilerPLSQL se abrevia como a. c.nombre se refiere al nombre del cliente. COUNT(*) AS numero_alquileres cuenta el número de alquileres por cliente, GROUP BY c.nombre agrupa por el nombre del cliente, ORDER BY numero_alquileres DESC ordena en orden descendente por el número de alquileres, LIMIT 1 limita los resultados a solo uno, mostrando así al cliente con el 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;

AutoPLSQL se abrevia como a, AlquilerPLSQL se abrevia como al, a.marca y a.modelo se refieren a la marca y modelo del auto, COUNT(*) AS numero_alquileres cuenta el número de alquileres por marca y modelo. GROUP BY a.marca, a.modelo agrupa por la marca y modelo del auto, ORDER BY numero_alquileres DESC ordena en orden descendente por el número de alquileres, LIMIT 1 limita los resultados a solo uno, mostrando así el auto con el mayor número de alquileres.

- 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;

Este codigo no se obtienen resultados debido a que se usa el campo id_sucursal de la tabla AlquilerPLSQL y este campo no existe en la tabla.

- 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;

Esta consulta SQL extrae el mes de la columna fecha_inicio en la tabla AlquilerPLSQL, cuenta el número de alquileres para cada mes, agrupa los resultados por mes, los ordena en orden descendente según el número de alquileres y finalmente devuelve el mes con el mayor número de alquileres, junto con ese número.

- 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;

Para esta linea existe rebundancia en la utilización de EXTRACT junto a DAYOFWEEK, por lo cual se podría usar la función DAYOFWEEK sin EXTRACT a partir de ajustar el código esta consulta se usa para obtener el día de la semana con el mayor número de alquileres en la tabla AlquilerPLSQL.

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

Esta consulta hace referencia a mostrar todos los alquileres de la tabla AlquilerPLSQL y que sean organizados en orden descendente por el campo precio, aunque debido a que la tabla no contiene este campo no es posible realizarla.

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

Esta consulta hace referencia a mostrar todos los alquileres de la tabla AlquilerPLSQL y que sean organizados en orden ascendente por el campo precio, aunque debido a que la tabla no contiene este campo no es posible realizarla.

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

Esta consulta selecciona todos los registros de la tabla ClientePLSQL donde el nombre del cliente contiene la cadena "Juan".

- SELECT a.marca, a.modelo, a.anho FROM AutoPLSQL a WHERE precio < 10000;

En esta ocasion se seleccionan los campos a.marca, a.modelo, a.anho de la tabla AutoPLSQL donde el precio sea inferior a 10000 aunque debido a la inexistencia del campo precio no se obtienen resultados.

- SELECT * FROM AlquilerPLSQL WHERE fecha_inicio BETWEEN '2023-09-01' AND '2023-09-30';

Esta consulta SQL selecciona todos los registros de la tabla AlquilerPLSQL donde la fecha de inicio (fecha_inicio) se encuentra en el rango entre el 1 de septiembre de 2023 y el 30 de septiembre de 2023.

- 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%';

Se debe tener en cuenta que ya se había abreviado la tabla AutoPLSQL as A por lo cual para abreviar ahora la tabla AlquilerPLSQL a la misma letra se debe hacer con la sentencia AS, adicional a esto la table AlquilerPLSQL no posee el campo marca por lo cual esta consulta no generara resultados.

- 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;

Esta consulta SQL selecciona las columnas marca, modelo, y anho de la tabla AutoPLSQL para aquellos autos que están asociados a un registro en la tabla AlquilerPLSQL donde el id_reserva es igual a 1.

- SELECT * FROM AlquilerPLSQL WHERE id_cliente IN (1, 2, 3);

Esta consulta selecciona todos los registros de la tabla AlquilerPLSQL donde el valor de id_cliente está contenido en el conjunto (1, 2, 3).

- SELECT * FROM AlquilerPLSQL WHERE id_auto IN (1, 2, 3);

Esta consulta selecciona todos los registros de la tabla AlquilerPLSQL donde el valor de id_auto está contenido en el conjunto (1, 2, 3).

- SELECT * FROM AlquilerPLSQL WHERE id_sucursal IN (1, 2, 3);

Para esta consulta no se pueden obtener resultados debido a que la tabla AlquilerPLSQL no posee el campo sucursal.

- SELECT * FROM AlquilerPLSQL WHERE fecha_inicio BETWEEN '2023-09-01' AND '2023-09-30' AND id_cliente IN (1, 2, 3);

Esta consulta SQL selecciona todos los registros de la tabla AlquilerPLSQL que cumplen con dos condiciones simultáneas:
Filtra los registros para incluir solo aquellos donde la fecha de inicio (fecha_inicio) está en el rango del 1 de septiembre de 2023 al 30 de septiembre de 2023, ambos inclusive, y filtra los registros para incluir solo aquellos donde el valor de id_cliente está contenido en el conjunto (1, 2, 3).

- SELECT * FROM AlquilerPLSQL WHERE fecha_inicio BETWEEN '2023-09-01' AND '2023-09-30' AND id_auto IN (1, 2, 3);

Esta consulta SQL selecciona todos los registros de la tabla AlquilerPLSQL que cumplen con dos condiciones simultáneas:
Filtra los registros para incluir solo aquellos donde la fecha de inicio (fecha_inicio) está en el rango del 1 de septiembre de 2023 al 30 de septiembre de 2023, ambos inclusive, y filtra los registros para incluir solo aquellos donde el valor de id_auto está contenido en el conjunto (1, 2, 3).

- SELECT * FROM AlquilerPLSQL WHERE fecha_inicio BETWEEN '2023-09-01' AND '2023-09-30' AND id_sucursal IN (1, 2, 3);

Para esta consulta no se pueden obtener resultados debido a que la tabla AlquilerPLSQL no posee el campo sucursal.

- 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;

Esta consulta realiza una operación de conteo agrupado (COUNT) para contar el número de alquileres por cliente (c.nombre), utiliza una operación de unión (JOIN) entre las tablas ClientePLSQL y AlquilerPLSQL basada en el ID del cliente, agrupa los resultados por el nombre del cliente (c.nombre), ordena los resultados en orden

descendente por el número de alquileres, limita los resultados a solo uno para obtener el cliente con el 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;

Esta consulta SQL realiza una operación de conteo agrupado (COUNT) para contar el número de alquileres por cada combinación única de marca y modelo en la tabla AutoPLSQL.

- 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;

Para esta consulta no se pueden obtener resultados debido a que la tabla AlquilerPLSQL no posee el campo sucursal, aun así realiza lo siguiente:
Esta consulta SQL realiza una operación de conteo agrupado (COUNT) para contar el número de alquileres por cada sucursal en la tabla SucursalPLSQL.

- 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;

Esta consulta SQL tiene el propósito de identificar el mes con el mayor número de alquileres en la tabla AlquilerPLSQL.

- 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;

Para esta linea existe rebundancia en la utilización de EXTRACT junto a DAYOFWEEK, por lo cual se podría usar la función DAYOFWEEK sin EXTRACT a partir de ajustar el código esta consulta tiene el propósito de identificar el día de la semana con el mayor número de alquileres en la tabla AlquilerPLSQL, junto con la cantidad total de alquileres para ese día específico.

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

Esta consulta hace referencia a mostrar todos los alquileres de la tabla AlquilerPLSQL y que sean organizados en orden descendente por el campo precio, aunque debido a que la tabla no contiene este campo no es posible realizarla.

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

Esta consulta hace referencia a mostrar todos los alquileres de la tabla AlquilerPLSQL y que sean organizados en orden ascendente por el campo precio, aunque debido a que la tabla no contiene este campo no es posible realizarla.

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

Esta consulta selecciona todos los registros de la tabla ClientePLSQL donde el nombre del cliente contiene la cadena "Juan" y tengan fecha de inicio entre '2023-09-01 y '2023-09-30.

- SELECT a.marca, a.modelo, a.anho FROM AutoPLSQL a WHERE precio < 10000 AND fecha_inicio BETWEEN '2023-09-01' AND '2023-09-30';

Esta consulta SQL selecciona las columnas marca, modelo, y ano de la tabla AutoPLSQL para aquellos autos que cumplen con dos condiciones: precio < 10000, y filtra los registros para incluir solo aquellos donde el valor de la columna precio es menor a 10000.

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';

Este codigo es para realizar una vista donde se identifiquen los clientes que hayan realizado alquileres en el sucursal: ‘Sucursal Central’, aunque la consulta presenta un error, dado que la tabla a AlquilerPLSQL no contiene el campo marca.

- 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 llamada vista_autos_alquilados_cliente_fecha. Esta vista selecciona las columnas marca y modelo de la tabla AutoPLSQL para aquellos autos que han sido alquilados por el cliente con el ID 1 y tienen una fecha de inicio de alquiler el 27 de septiembre de 2023.

- CREATE VIEW vista_alquileres_mas_7dias AS SELECT * FROM AlquilerPLSQL WHERE fecha_fin - fecha_inicio > 7;

Esta vista se llama vista_alquileres_mas_7dias. Selecciona todos los campos de la tabla AlquilerPLSQL para aquellos registros donde la diferencia entre las fechas de inicio y fin (fecha_fin - fecha_inicio) es mayor que 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;

Esta vista se llama vista_clientes_mas_alquileres. Selecciona el nombre del cliente (c.nombre) y cuenta el número de alquileres realizados por cada cliente.

- 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 llamada vista_autos_mas_alquileres. La vista selecciona la marca y el modelo de los autos de la tabla AutoPLSQL, y cuenta el número total de alquileres para cada combinación única de marca y modelo.

- 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 llamada vista_sucursales_mas_alquileres. La vista selecciona el nombre de la sucursal y cuenta el número total de alquileres para cada sucursal.

- 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 llamada vista_meses_mas_alquileres. La vista selecciona el mes de la fecha de inicio de los alquileres y cuenta el número total de alquileres para cada mes.

- 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;

Al ejecutarla en MYSQL, presenta una rebundancia en el uso de EXTRACT y DAYOFWEEK, luego de ajustar esto el codigo genera lo siguiente:
Crea una vista llamada vista_dias_semana_mas_alquileres. La vista selecciona el día de la semana de la fecha de inicio de los alquileres y cuenta el número total de alquileres para cada día de la semana.

- CREATE VIEW vista_alquileres_mas_caros AS SELECT * FROM AlquilerPLSQL ORDER BY precio DESC;

No se obtienen resultados debido a que la tabla AlquilerPLSQL no tiene el campo precio y de esta manera no se puede ejecutar.

- CREATE VIEW vista_alquileres_mas_baratos AS SELECT * FROM AlquilerPLSQL ORDER BY precio ASC;

No se obtienen resultados debido a que la tabla AlquilerPLSQL no tiene el campo precio y de esta manera no se puede ejecutar.

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;
```