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;
   - Mostrar todos los autos en la tabla "Auto".
     SELECT * FROM AutoPLSQL;
   - Mostrar todos los alquileres en la tabla "Alquiler".
     ***SELECT * FROM AlquilerPLSQL*;**
   - Mostrar todas las sucursales en la tabla "Sucursal".
     SELECT * FROM SucursalPLSQL;
   - Mostrar todas las reservas en la tabla "Reserva".
     SELECT * FROM ReservaPLSQL;
2. **Filtros y Ordenamiento:**
   - Mostrar los clientes que se llaman "Juan".
     SELECT * FROM ClientePLSQL WHERE NOMBRE LIKE '%Juan%';
   - Mostrar los autos de marca "Toyota".
     SELECT * FROM AutoPLSQL WHERE MARCA LIKE '%Toyota%';
   - Mostrar los alquileres que ocurrieron después de una fecha específica.
     SELECT * FROM AlquilerPLSQL WHERE FECHA_INICIO > '19-04-23';
   - Mostrar las sucursales ubicadas en "Madrid".
     SELECT * FROM SucursalPLSQL WHERE CIUDAD LIKE '%Madrid%';
   - Mostrar las reservas realizadas por un cliente específico.
     SELECT * FROM ReservaPLSQL WHERE ID_CLIENTE = 197;
3. **Join y Relaciones:**
   - Mostrar los alquileres con los nombres de los clientes y las marcas de los autos.
     SELECT cli.nombre, au.marca
     FROM AlquilerPLSQL al
     INNER JOIN ClientePLSQL cli ON cli.ID_CLIENTE = al.ID_CLIENTE
     INNER JOIN AutoPLSQL au on au.ID_AUTO = al.ID_AUTO;
   - Mostrar los clientes que han realizado reservas en una sucursal específica.
     SELECT cli.nombre, su.*
     FROM ReservaPLSQL re
     INNER JOIN SucursalPLSQL su ON su.ID_SUCURSAL = re.ID_SUCURSAL

INNER JOIN ClientePLSQL cli ON cli.ID_CLIENTE = re.ID_CLIENTE
WHERE su.ID_SUCURSAL = 610;

- Mostrar los autos que han sido alquilados junto con los nombres de los clientes.
  SELECT au.*, cli.nombre
  FROM AutoPLSQL au
  INNER JOIN AlquilerPLSQL al ON al.ID_AUTO = au.ID_AUTO
  INNER JOIN ClientePLSQL cli ON cli.ID_CLIENTE = al.ID_CLIENTE

- Mostrar los detalles de las reservas con los nombres de los clientes y las ciudades de las sucursales.
  SELECT cli.*, su.ciudad
  FROM ReservaPLSQL re
  INNER JOIN ClientePLSQL cli ON cli.ID_CLIENTE = re.ID_CLIENTE
  INNER JOIN SucursalPLSQL SU ON su.ID_SUCURSAL = re.ID_SUCURSAL
- Mostrar los clientes que no han realizado ninguna reserva.
  SELECT CLI.NOMBRE, CLI.ID_CLIENTE
  FROM CLIENTEPLSQL CLI
  LEFT JOIN RESERVAPLSQL RE ON RE.ID_CLIENTE = CLI.ID_CLIENTE
  WHERE RE.ID_CLIENTE IS NULL;

4. Agregación y Agrupamiento:

- Contar cuántos autos hay de cada marca en la tabla "Auto".
  SELECT MARCA, COUNT (MARCA) AS NU_AUTOS
  FROM AUTOPLSQL GROUP BY MARCA
- Calcular la duración promedio de los alquileres.
  SELECT ROUND (AVG(FECHA_FIN - FECHA_INICIO),0) AS DURACION_PROMEDIO
  FROM ALQUILERPLSQL;
- Mostrar el número total de reservas realizadas en cada sucursal.
  SELECT SU.ID_SUCURSAL, SU.NOMBRE, COUNT(RE.ID_RESERVA) AS TOTAL_RESERVA
  FROM SUCURSALPLSQL SU
  LEFT JOIN RESERVAPLSQL RE ON RE.ID_SUCURSAL = SU.ID_SUCURSAL
  GROUP BY SU.ID_SUCURSAL, SU.NOMBRE;
- Encontrar el cliente que ha realizado la mayor cantidad de alquileres.
  select cli.nombre, count(al.id_cliente) as No_alquiler
  from clienteplsql cli
  inner join alquilerplsql al on al.id_cliente = cli.id_cliente
  group by cli.nombre
  order by count(al.id_alquiler) desc
  fetch first 1 row only;
- Calcular el promedio de años de los autos en la tabla "Auto".
  select round (avg(ano),0) as promedio_años
  from autoplsql;

5. **Subconsultas**:
   - Mostrar los clientes que han realizado al menos una reserva.
     select distinct cli.*
     from clienteplsql cli
     join reservaplsql re on re.id_cliente = cli.id_cliente;
   - Mostrar los autos que no han sido alquilados aún.
     select au.*
     from autoplsql au
     left join alquilerplsql al on al.id_auto = au.id_auto
     where al.id_auto is null;
   - Encontrar los clientes que han alquilado el mismo auto más de una vez.
     select cli.id_cliente, cli.nombre, al.id_auto, count(al.id_alquiler) as veces_alquilado
     from alquilerplsql al
     inner join clienteplsql cli on cli.id_cliente = al.id_cliente
     group by cli.id_cliente, cli.nombre, al.id_auto
     having count(*) > 1;
   - Mostrar los clientes que han realizado alquileres en la misma ciudad en la que viven.
     SELECT cli.id_cliente, cli.nombre, cu.ciudad AS ciudad_vivienda, al.ciudad AS ciudad_alquiler
     from clienteplsql cli
     join alquilerplsql al on cli.id_cliente = al.id_cliente
     join sucursalplsql su on cli.ciudad = su.ciudad;

   - Encontrar los autos que han sido alquilados en la misma sucursal donde se realizó una reserva.
     select re.id_reserva, au.id_auto, au.marca, au.modelo, su.nombre as sucursal_reserva
     from reservaplsql re
     join alquilerplsql al on re.id_reserva = al.id_reserva
     join autoplsql au on al.id_auto = au.id_auto
     join sucursalplsql su on re.id_sucursal = su.id_sucursal;

6. **Actualizaciones y Eliminaciones:**
   - Actualizar la dirección de un cliente específico.
     update clienteplsql
     set direccion = 'calle falsa 123'
     where id_cliente = 631;
   - Eliminar un auto de la tabla "Auto".

     DELETE FROM AutoPLSQL
     WHERE id_auto = 20;

   - Marcar una reserva como completada actualizando la fecha de fin.
     Update reservaplsql
     Set fecha_fin = to_date('2023-11-21','YYYY-MM-DD')
     Where id_reserva = 870;

- Eliminar todas las reservas realizadas por un cliente específico.
  delete from reservaplsql
  where id_cliente = 890;
- Actualizar el año de un auto en la tabla "Auto".
  update autoplsql
  set ano = 1999
  where id_auto = 30;

EJERCICIOS SEGUNDO CICLO (31-80):

- `SELECT * FROM ClientePLSQL;`

Muestra toda la tabla de clientes.

- `SELECT * FROM AutoPLSQL;`

Muestra toda la tabla de autos.

- `SELECT * FROM AlquilerPLSQL;`

Muestra toda la tabla de alquiler.

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

Muestra el nombre de un cliente, la marca y el modelo de la que alquilo

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

Muestra una tabla con la marca el modelo y al año de los autos que fueron alquilados.

- `SELECT * FROM AlquilerPLSQL WHERE id_cliente = 1;`

Muestra como resultado el detalle del alquiler de un cliente en específico.

- `SELECT * FROM AlquilerPLSQL WHERE id_auto = 1;`

Muestra los datos detallados del es alquiler de una auto espercifico.

- `SELECT * FROM AlquilerPLSQL WHERE id_sucursal = 1;`

Muestra la información detallada de alquiler para una sucursal especifica.

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

Muestra la información de alquiler en una fecha especifica.

- SELECT COUNT(*) FROM AlquilerPLSQL;

Muestra la información total de registros de alquieres en la tabla de 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';

Muestra la información del nombre de los clientes que han alquilado en la 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';

  Muestra la información de la marca y modelo de autos que alquilo el cliente 1 en una fecha específica.

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

Muestra los alquileres que sean mayores a 7 dias.

- 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;

Muestra el nombre del cliente con la cantidad de alquileres que haya hecho que se tenga como limite 1 alquiler

- 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;

Muestra la marca y el modelo de un auto y la cantidad de veces que lo han 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;

Muestra el nombre del cliente que mas veces ha alquilado

- 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;

Muestra él mes en el cual se han hecho mas 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;

Muestra el día de la semana en el cual se realizan más alquileres

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

Muestra el precio más alto por el alquiler de un auto.

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

Muestra el precio más bajo por el alquiler de un auto.

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

Muestra todos los clients que se llamen Juan

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

Muestra la marca el modelo y año de los carros que tengan un precio menor a 10000.

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

Muestra los registros de alquiler que se encuentren dentro de una fechas especificas.

- 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%';

Muestra el nombre de un cliente con la marca y el modelo del alquiler que sean de 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;

Muestra la información de marca modelo y años del primer registro del auto que se alquiló.

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

Muestra información de alquiler de clientes específicos

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

Muestra información de alquiler de autos especificos

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

Muestra información de alquiler en sucursales especificas

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

Muestra información de alquiler de clientes específicos dentro de unas fechas establecidas.

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

Muestra información de alquiler de autos específicos dentro de unas fechas establecidas.

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

Muestra información de alquiler en sucursales específicas dentro de unas fechas establecidas.

- 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;

Muestra el nombre del cliente que hizo mas 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;

Muestra la marca y el modelo del auto que estuvo mas veces 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;

Muestra el nombre de la sucursar que tenido mayor cantidad de alquiler

- 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;

Muestra el mes que tuvo mayor cantidad de alquiler

- 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;

Muestra el dia que tuvo mayor cantidad de alquiler

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

Muestra el alquiler que tenga el precio más alto de los registros

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

Muestra el alquiler que tenga el precio mas bajo de los registros

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

Muestra los clients con nombre Juan que tenga alquileres en un periodo de tiempo específico.

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

Muestra información de marca modelo y años de los autos que tengan un precio menor a 10000 y que se hayan alquilado en un periodo de tiempo específico.

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 en donde va a traer el nombre del cliente, la marca y el modelo del auto que se alquilo en la sucursal centra.

- 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 donde va a mostrar la marca y el modelo del auto que alquilo un cliente especifico en una fecha específica.

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

Crea una vista en donde va a mostrar los alquileres que sea mayores 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 donde va a mostrar los clientes y cuenta la cantidad de alquiler que tiene cada uno y los organiza 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 donde va a mostrar la marca y el modelo de que auto y va a mostrar la cantidad de veces que han sido alquilado y los organiza de mayor a menos.

- 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 en donde va mostrar el nombre de las surcursales y va contar la cantidad veces que fueron lauiladas.

- 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 donde va a mostrar los meses en donde se hicieron alquileres y va contar cuantos se hicieron en cada uno, para organizarlo 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;

Crea una vista donde va mostrar los dias en donde se hicieron alquileres y va contar cuantos se hicieron en cada uno, para organizarlo de mayor a menor.

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

Crea una vista va a mostrar los alquileres con precios mas altos y los organiza de mayor a menor.

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

Crea una vista va a mostrar los alquileres con precios mas bajos y los organiza 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;
```

Esta función quiere realizar una actualización en la tabla de autos, en la cual va a generar un nuevo espacio para poder agregar un nuevo registro y de esta forma tener disponibilidad siempre que se quiera agregar y estar ajustado.

```
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;
```

Esta función quiere realizar una actualización en la tabla de autos, en la cual va a eliminar un auto y que al realizar esta acción la tabla se ajuste con cada cambio.

```
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 código tiene como función mantener actualizado el número de autos disponibles en caso de generar una modificación en la tabla.

```
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 código tiene función realizar un incremento automático en la tabla de clientes, cada vez que se quiere ingresar un nuevo registro, esto se hace para mantener el contador de clientes actualizado.

```
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 código tiene función realizar un decremento automático en la tabla de clientes, cada vez que se quiere eliminar un nuevo registro, esto se hace para mantener el contador de clientes actualizado.

```
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 codigo quiere actualizar la cantidad de alquiler por cliente esto implica que si el nuevo numero de alquiler es diferente del viejo numero de alquiler, este ejecutara una actualización.

```
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;
```

Este procedimiento almacenado va a calcular el precio del alquiler del auto basándose en el precio base del auto y el tiempo que dure el alquiler. Luego con el resultado actualizar el valor del precio de 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;
```

Este procedimiento almacenado tiene como objetivo listar todos los alquileres asociados a un cliente específico.

```
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;
```

Este procedimiento almacenado tiene como objetivo listar todos los alquileres asociados a un auto específico.

```
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;
```

Este procedimiento almacenado tiene como objetivo agregar un nuevo auto a la tabla, agregando todos los campo de marca modelo año y numero disponible, con el fin de agregar un nuevo registro a la tabla.

```
CREATE PROCEDURE proc_eliminar_auto
(
  IN id_auto INT
)
AS
BEGIN
  -- Eliminar un auto
  DELETE FROM AutoPLSQL
  WHERE id_auto = id_auto;
END;
```

Este procedimiento almacenado tiene como objetivo eliminar un auto de la tabla