**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,

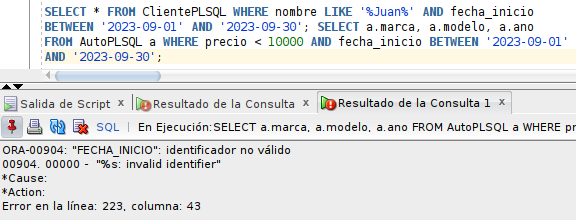
fecha\_reserva DATE,

FOREIGN KEY (id\_cliente) REFERENCES Cliente(id\_cliente),

FOREIGN KEY (id\_sucursal) REFERENCES Sucursal(id\_sucursal));

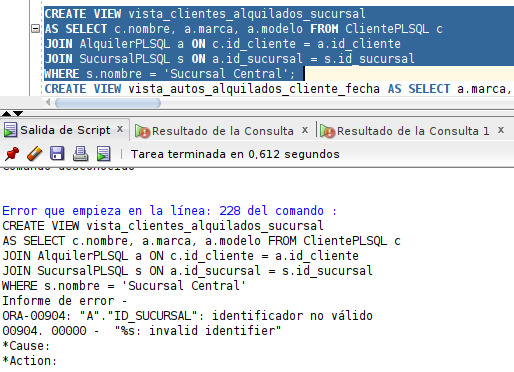
**Ejercicios del 51 al 100:**

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

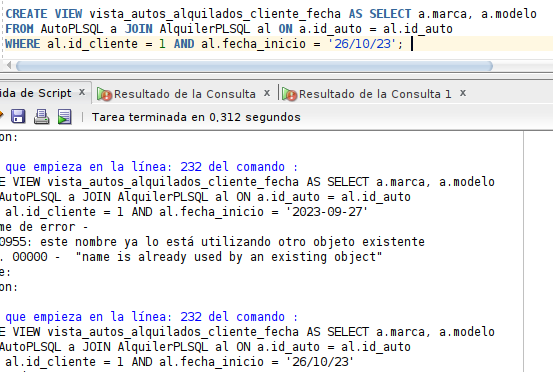


**EJERCICIOS TERCER CICLO (81-90):**

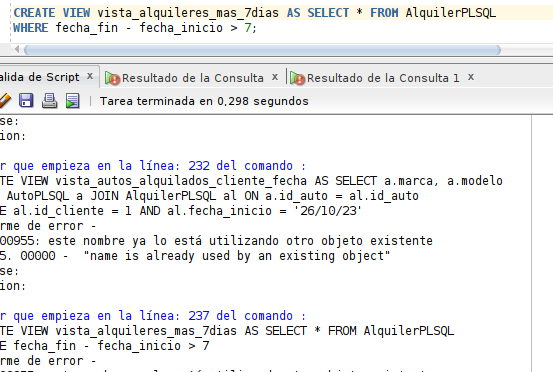
* CREATE VIEW vista\_clientes\_alquilados\_sucursal AS SELECT c.nombre, a.marca, a.modelo FROM ClientePLSQL c JOIN AlquilerPLSQL a ON c.id\_cliente = a.id\_cliente JOIN SucursalPLSQL s ON a.id\_sucursal = s.id\_sucursal WHERE s.nombre = 'Sucursal Central';



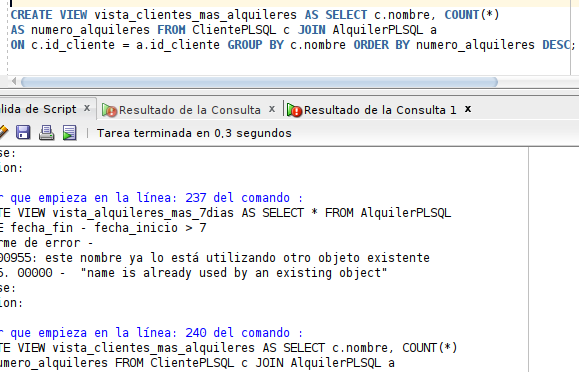
* CREATE VIEW vista\_autos\_alquilados\_cliente\_fecha AS SELECT a.marca, a.modelo FROM AutoPLSQL a JOIN AlquilerPLSQL al ON a.id\_auto = al.id\_auto WHERE al.id\_cliente = 1 AND al.fecha\_inicio = '2023-09-27';



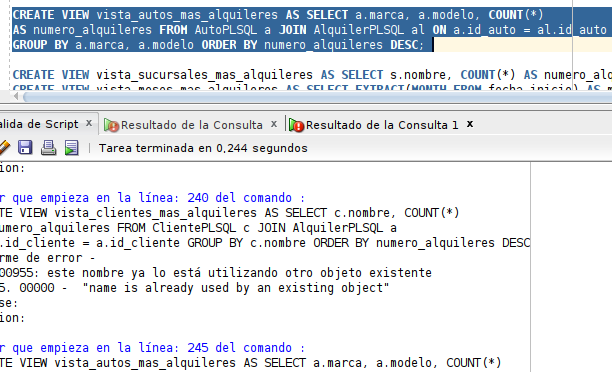
* CREATE VIEW vista\_alquileres\_mas\_7dias AS SELECT \* FROM AlquilerPLSQL WHERE fecha\_fin - fecha\_inicio > 7;



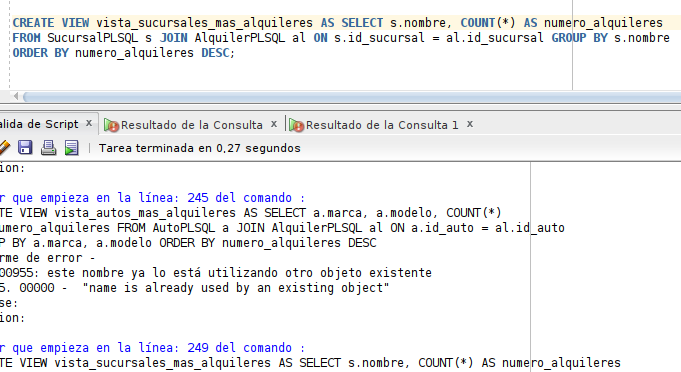
* CREATE VIEW vista\_clientes\_mas\_alquileres AS SELECT c.nombre, COUNT(\*) AS numero\_alquileres FROM ClientePLSQL c JOIN AlquilerPLSQL a ON c.id\_cliente = a.id\_cliente GROUP BY c.nombre ORDER BY numero\_alquileres DESC;



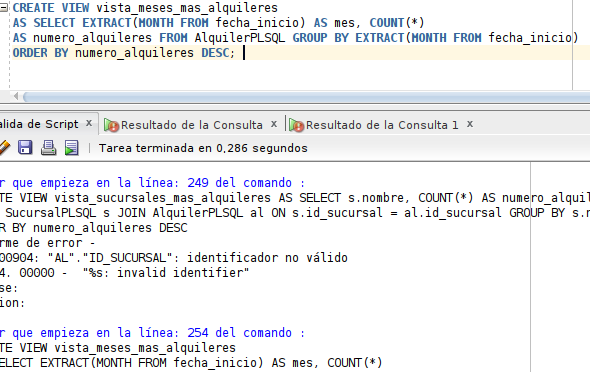
* CREATE VIEW vista\_autos\_mas\_alquileres AS SELECT a.marca, a.modelo, COUNT(\*) AS numero\_alquileres FROM AutoPLSQL a JOIN AlquilerPLSQL al ON a.id\_auto = al.id\_auto GROUP BY a.marca, a.modelo ORDER BY numero\_alquileres DESC;



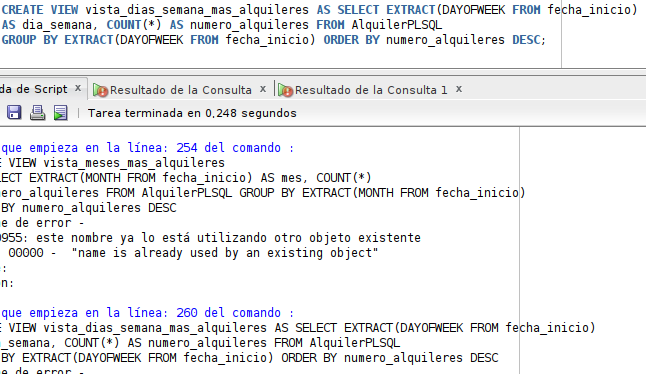
* CREATE VIEW vista\_sucursales\_mas\_alquileres AS SELECT s.nombre, COUNT(\*) AS numero\_alquileres FROM SucursalPLSQL s JOIN AlquilerPLSQL al ON s.id\_sucursal = al.id\_sucursal GROUP BY s.nombre ORDER BY numero\_alquileres DESC;



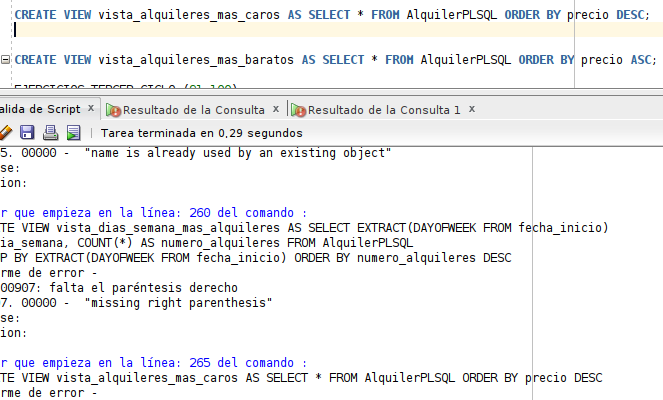
* CREATE VIEW vista\_meses\_mas\_alquileres AS SELECT EXTRACT(MONTH FROM fecha\_inicio) AS mes, COUNT(\*) AS numero\_alquileres FROM AlquilerPLSQL GROUP BY EXTRACT(MONTH FROM fecha\_inicio) ORDER BY numero\_alquileres DESC;



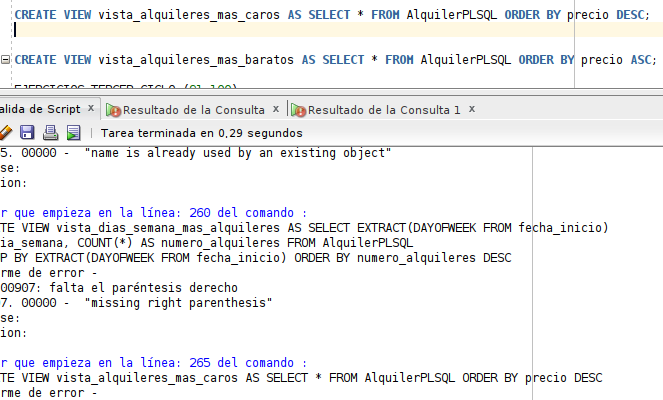
* CREATE VIEW vista\_dias\_semana\_mas\_alquileres AS SELECT EXTRACT(DAYOFWEEK FROM fecha\_inicio) AS dia\_semana, COUNT(\*) AS numero\_alquileres FROM AlquilerPLSQL GROUP BY EXTRACT(DAYOFWEEK FROM fecha\_inicio) ORDER BY numero\_alquileres DESC;



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



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



**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;    Todos estos ya estaban, es por esto que todos daban error que exiten, la manera para que no aparezca el error, es actualizandolos, sin embargo no valia la pena realizarlo porque en si no hay nada que actualizar |