Bases de Datos Estructuradas

2024

Script Base de datos



https://github.com/ebravo930/Inacap_2024_Bases-de-Datos-Estructuradas

Ejercicio 1: Procedimiento Almacenado para Actualizar el Precio de un Producto

Descripción:

Crea un procedimiento almacenado llamado sp_ActualizarPrecioProducto que tome como parámetros el ID del producto y el nuevo precio, y actualice el precio en la tabla Productos. Luego, verifique si el nuevo precio es mayor que el precio promedio de todos los productos. Si es así, el procedimiento debe retornar un mensaje indicando que el precio es superior al promedio.

```
CREATE OR REPLACE PROCEDURE sp_ActualizarPrecioProducto (
    p_ProductoID NUMBER,
    p_NuevoPrecio NUMBER
) IS
    v_PrecioPromedio NUMBER;
BEGIN
    -- Actualizar el precio del producto
    UPDATE Productos
    SET Precio = p_NuevoPrecio
    WHERE ProductoID = p_ProductoID;

    -- Obtener el precio promedio de todos los productos
    SELECT AVG(Precio)
    INTO v_PrecioPromedio
    FROM Productos;

    -- Verificar si el nuevo precio es mayor que el promedio
    IF p_NuevoPrecio > v_PrecioPromedio THEN
        DBMS_OUTPUT.PUT_LINE('El nuevo precio es superior al precio promedio
de todos los productos.');
    ELSE
        DBMS_OUTPUT.PUT_LINE('El nuevo precio no es superior al precio
promedio.');
    END IF;
END;
/
```

Ejercicio 2: Consulta con INNER JOIN para Listar Ventas y Detalles de Clientes

Descripción: Escribe una consulta que use INNER JOIN para mostrar todas las ventas con los detalles de los clientes que realizaron cada venta, incluyendo el nombre del cliente, apellido, fecha de la venta, y el total de la venta.

Ejercicio 2: Consulta con INNER JOIN para Listar Ventas y Detalles de Clientes

Descripción:Escribe una consulta que use INNER JOIN para mostrar todas las ventas con los detalles de los clientes que realizaron cada venta, incluyendo el nombre del cliente, apellido, fecha de la venta, y el total de la venta.

```
SELECT
    c.Nombre,
    c.Apellido,
    v.FechaVenta,
    v.TotalVenta
FROM Ventas v
INNER JOIN Clientes c ON v.ClienteID = c.ClienteID;
```

Ejercicio 3: Consulta con LEFT JOIN para Mostrar Todos los Clientes y sus Ventas (Si las Hay)

Descripción:

Escribe una consulta que use LEFT JOIN para mostrar todos los clientes, incluyendo aquellos que no han realizado ninguna venta. La consulta debe incluir el nombre del cliente, apellido, fecha de la venta (si existe), y el total de la venta (si existe).

Ejercicio 3: Consulta con LEFT JOIN para Mostrar Todos los Clientes y sus Ventas (Si las Hay)

Descripción:

Escribe una consulta que use LEFT JOIN para mostrar todos los clientes, incluyendo aquellos que no han realizado ninguna venta. La consulta debe incluir el nombre del cliente, apellido, fecha de la venta (si existe), y el total de la venta (si existe).

```
SELECT
    c.Nombre,
    c.Apellido,
    v.FechaVenta,
    v.TotalVenta
FROM Clientes c
LEFT JOIN Ventas v ON c.ClienteID = v.ClienteID;
```

Ejercicio 4: Consulta con RIGHT JOIN para Mostrar Todas las Ventas y Clientes (Si Existen)
Descripción:
Escribe una consulta que use RIGHT JOIN para mostrar todas las ventas, incluyendo aquellas que podrían no tener un cliente asociado (en caso de que los datos se eliminen accidentalmente). La consulta debe incluir el nombre del cliente, apellido, fecha de la venta, y el total de la venta.

Ejercicio 4: Consulta con RIGHT JOIN para Mostrar Todas las Ventas y Clientes (Si Existen)
Descripción:
Escribe una consulta que use RIGHT JOIN para mostrar todas las ventas, incluyendo aquellas que podrían no tener un cliente asociado (en caso de que los datos se eliminen accidentalmente). La consulta debe incluir el nombre del cliente, apellido, fecha de la venta, y el total de la venta.

```
SELECT
    c.Nombre,
    c.Apellido,
    v.FechaVenta,
    v.TotalVenta
FROM Ventas v
RIGHT JOIN Clientes c ON c.ClienteID = v.ClienteID;
```

Ejercicio 5: Trigger para Prevenir Eliminaciones de Clientes con Ventas Asociadas
Descripción:

Crea un trigger llamado trg_PrevenirEliminacionCliente que se active antes de que un cliente sea eliminado de la tabla Clientes. El trigger debe verificar si el cliente tiene ventas asociadas en la tabla Ventas. Si el cliente tiene ventas, el trigger debe prevenir la eliminación y mostrar un mensaje.

Ejercicio 5: Trigger para Prevenir Eliminaciones de Clientes con Ventas Asociadas
Descripción:

Crea un trigger llamado trg_PrevenirEliminacionCliente que se active antes de que un cliente sea eliminado de la tabla Clientes. El trigger debe verificar si el cliente tiene ventas asociadas en la tabla Ventas. Si el cliente tiene ventas, el trigger debe prevenir la eliminación y mostrar un mensaje.

```
CREATE OR REPLACE TRIGGER trg_PrevenirEliminacionCliente
BEFORE DELETE ON Clientes
FOR EACH ROW
DECLARE
   v_VentasCount NUMBER;
BEGIN
   -- Contar las ventas asociadas al cliente
   SELECT COUNT(*) INTO v_VentasCount
   FROM Ventas
   WHERE ClienteID = :OLD.ClienteID;

   -- Verificar si el cliente tiene ventas asociadas
   IF v_VentasCount > 0 THEN
      RAISE_APPLICATION_ERROR(-20001, 'No se puede eliminar el cliente
porque tiene ventas asociadas.');
   END IF;
END;
/
```

Ejercicio 6: Subconsulta para Encontrar Productos con Precio Superior al Promedio
Descripción:

Escribe una consulta que use una subconsulta para encontrar todos los productos cuyo precio es superior al precio promedio de todos los productos en la tabla Productos.

Ejercicio 6: Subconsulta para Encontrar Productos con Precio Superior al Promedio
Descripción:

Escribe una consulta que use una subconsulta para encontrar todos los productos cuyo precio es superior al precio promedio de todos los productos en la tabla Productos.

```
SELECT NombreProducto, Precio
FROM Productos
WHERE Precio > (SELECT AVG(Precio) FROM Productos);
```

Ejercicio 7: Subconsulta de Varias Filas para Encontrar Clientes con Ventas Superiores a las Ventas de un Cliente Específico

Descripción:

Escribe una consulta que use una subconsulta de varias filas para encontrar todos los clientes que han realizado ventas superiores al total de ventas realizadas por un cliente específico (por ejemplo, el cliente con ClienteID = 2).

Ejercicio 7: Subconsulta de Varias Filas para Encontrar Clientes con Ventas Superiores a las Ventas de un Cliente Específico

Descripción:

Escribe una consulta que use una subconsulta de varias filas para encontrar todos los clientes que han realizado ventas superiores al total de ventas realizadas por un cliente específico (por ejemplo, el cliente con ClienteID = 2).

```
SELECT Nombre, Apellido
FROM Clientes c
WHERE (SELECT SUM(TotalVenta)
      FROM Ventas v
      WHERE v.ClienteID = c.ClienteID) >
     (SELECT SUM(TotalVenta)
      FROM Ventas
      WHERE ClienteID = 2);
```

MUCHAS GRACIAS!

inacap.cl