

Procedimientos almacenados

Insert

```
create or replace PROCEDURE SP_INSERTAR_CLIENTE (  
    P_CEDULA_CLIENTE NUMBER,  
    P_NOMBRE VARCHAR2,  
    P_APELLIDO_1 VARCHAR2,  
    P_APELLIDO_2 VARCHAR2,  
    P_NACIMIENTO DATE,  
    P_CORREO VARCHAR2,  
    P_TELEFONO VARCHAR2,  
    P_DIRECCION VARCHAR2  
)  
AS  
BEGIN  
    INSERT INTO CLIENTE (CEDULA_CLIENTE, NOMBRE, APELLIDO_1, APELLIDO_2, NACIMIENTO, CORREO, TELEFONO, DIRECCION)  
    VALUES (P_CEDULA_CLIENTE, P_NOMBRE, P_APELLIDO_1, P_APELLIDO_2, P_NACIMIENTO, P_CORREO, P_TELEFONO, P_DIRECCION);  
    COMMIT;  
END SP_INSERTAR_CLIENTE;
```

Recibe los parámetros necesarios para insertar un registro en la tabla.

Insert Factura

```
create or replace PROCEDURE SP_INSERTAR_FACTURA (  
    P_FAC_CED_CLIENTE NUMBER,  
    P_FAC_COD_PRODUCTO NUMBER,  
    P_FAC_ID_COLABORADOR NUMBER,  
    P_FECHA DATE,  
    P_TOTAL_PAGADO NUMBER,  
    P_PRECIO_UNITARIO NUMBER,  
    P_CANTIDAD NUMBER  
)  
AS  
    V_STOCK_ACTUAL NUMBER;  
BEGIN  
    SELECT STOCK INTO V_STOCK_ACTUAL FROM PRODUCTOS WHERE COD_PRODUCTO = P_FAC_COD_PRODUCTO;  
  
    UPDATE PRODUCTOS SET STOCK = V_STOCK_ACTUAL - P_CANTIDAD WHERE COD_PRODUCTO = P_FAC_COD_PRODUCTO;  
  
    IF (V_STOCK_ACTUAL - P_CANTIDAD <= 1) THEN  
        UPDATE PRODUCTOS SET ESTADO = 'Agotado' WHERE COD_PRODUCTO = P_FAC_COD_PRODUCTO;  
    END IF;  
  
    INSERT INTO FACTURAS (FAC_CED_CLIENTE, FAC_COD_PRODUCTO, FAC_ID_COLABORADOR, FECHA, TOTAL_PAGADO, PRECIO_UNITARIO, CANTIDAD)  
    VALUES (P_FAC_CED_CLIENTE, P_FAC_COD_PRODUCTO, P_FAC_ID_COLABORADOR, P_FECHA, P_TOTAL_PAGADO, P_PRECIO_UNITARIO, P_CANTIDAD);  
  
    COMMIT;  
END SP_INSERTAR_FACTURA;
```

Este procedimiento, es lo mismo que el normal, pero se hacen dos update en la tabla de productos, busca el producto por el cod_producto y se hace una operación en el stock, se le resta al stock la cantidad del producto de la factura y si el stock es igual o menor a 1 se cambia el estado a “Agotado”.

Insert Devoluciones

```

create or replace PROCEDURE SP_INSERTAR_DEVOLUCION (
    P_DEV_COD_FACTURA NUMBER,
    P_MONTO_DEVOLUCION NUMBER
)
AS
    V_FECHA DATE := SYSDATE;
    V_DEV_CED_CLIENTE NUMBER;
    V_DEV_COD_PRODUCTO NUMBER;
BEGIN
    SELECT FAC_CED_CLIENTE, FAC_COD_PRODUCTO INTO V_DEV_CED_CLIENTE, V_DEV_COD_PRODUCTO
    FROM FACTURAS
    WHERE COD_FACTURA = P_DEV_COD_FACTURA;

    INSERT INTO DEVOLUCIONES (FECHA, DEV_CED_CLIENTE, DEV_COD_PRODUCTO, DEV_COD_FACTURA, MONTO_DEVOLUCION)
    VALUES (V_FECHA, V_DEV_CED_CLIENTE, V_DEV_COD_PRODUCTO, P_DEV_COD_FACTURA, P_MONTO_DEVOLUCION);

    UPDATE FACTURAS SET ESTADO = 0 WHERE COD_FACTURA = P_DEV_COD_FACTURA;

    COMMIT;
END SP_INSERTAR_DEVOLUCION;

```

Este procedimiento lo que hace es buscar la factura y obtener la cedula del cliente y el código del producto y agregarlos al insert. Y tiene un update que cambia el estado de la factura a 0. 0 significa que la factura tiene devolución y 1 que la factura sigue activa.

Delete

```

create or replace PROCEDURE SP_ELIMINAR_CLIENTE (
    P_CEDULA_CLIENTE NUMBER
)
AS
BEGIN
    DELETE FROM CLIENTE
    WHERE CEDULA_CLIENTE = P_CEDULA_CLIENTE;
    COMMIT;
END SP_ELIMINAR_CLIENTE;

```

Este procedimiento recibe de parámetro el id, código o cedula para eliminar el registro.

Funciones

Listar registros

```

create or replace FUNCTION F_LISTAR_CLIENTES
RETURN SYS_REFCURSOR
AS
    V_CURSOR SYS_REFCURSOR;
BEGIN
    OPEN V_CURSOR FOR
        SELECT * FROM CLIENTE;

    RETURN V_CURSOR;
END F_LISTAR_CLIENTES;

```

Esta función utiliza un cursor para recorrer los registros y retorna los registros.

Update

```
3 create or replace NONEDITIONABLE FUNCTION F_ACTUALIZAR_CATEGORIA(  
  P_ID_CATEGORIA NUMBER,  
  P_NOMBRE VARCHAR2,  
  P_DESCRIPCION VARCHAR2  
) RETURN BOOLEAN  
IS  
  V_ACTUALIZADO BOOLEAN := FALSE;  
BEGIN  
3  UPDATE CATEGORIA SET  
    NOMBRE = P_NOMBRE,  
    DESCRIPCION = P_DESCRIPCION  
  WHERE ID_CATEGORIA = P_ID_CATEGORIA;  
  
  IF SQL%ROWCOUNT = 1 THEN  
    V_ACTUALIZADO := TRUE;  
  END IF;  
  
  RETURN V_ACTUALIZADO;  
EXCEPTION  
  WHEN OTHERS THEN  
    RETURN FALSE;  
END F_ACTUALIZAR_CATEGORIA;
```

Esta función se utiliza para actualizar la información de un registro de una tabla. Y devuelve un valor booleano que indica si la actualización fue exitosa o no.

Triggers

```
CREATE OR REPLACE TRIGGER TRG_CATEGORIA_INSERT  
AFTER INSERT ON CATEGORIA  
FOR EACH ROW  
BEGIN  
  INSERT INTO AUDITORIA_CATEGORIA (ID_CATEGORIA, NOMBRE, DESCRIPCION, OPERACION, FECHA_OPERACION)  
  VALUES (:NEW.ID_CATEGORIA, :NEW.NOMBRE, :NEW.DESCRIPCION, 'INSERT', SYSDATE);  
END;  
/  
  
CREATE OR REPLACE TRIGGER TRG_CATEGORIA_UPDATE  
AFTER UPDATE ON CATEGORIA  
FOR EACH ROW  
BEGIN  
  INSERT INTO AUDITORIA_CATEGORIA (ID_CATEGORIA, NOMBRE, DESCRIPCION, OPERACION, FECHA_OPERACION)  
  VALUES (:OLD.ID_CATEGORIA, :OLD.NOMBRE, :OLD.DESCRIPCION, 'UPDATE', SYSDATE);  
END;  
/  
  
CREATE OR REPLACE TRIGGER TRG_CATEGORIA_DELETE  
AFTER DELETE ON CATEGORIA  
FOR EACH ROW  
BEGIN  
  INSERT INTO AUDITORIA_CATEGORIA (ID_CATEGORIA, NOMBRE, DESCRIPCION, OPERACION, FECHA_OPERACION)  
  VALUES (:OLD.ID_CATEGORIA, :OLD.NOMBRE, :OLD.DESCRIPCION, 'DELETE', SYSDATE);  
END;  
/
```

Se crean tablas de auditoria para cada tabla y con los triggers se registra cada insert, update y delete que se realicen en las tablas, se almacenan los datos del registro, más la fecha y el tipo de operación.

DJANGO

Index

```
def indexClientes(request):
    cursor = connection.cursor()
    clientes_cursor = cursor.callfunc('F_LISTAR_CLIENTES', cx_Oracle.CURSOR)
    clientes = []
    for cliente in clientes_cursor:
        clientes.append({
            'cedula_cliente': cliente[0],
            'nombre': cliente[1],
            'apellido_1': cliente[2],
            'apellido_2': cliente[3],
            'nacimiento': cliente[4],
            'correo': cliente[5],
            'telefono': cliente[6],
            'direccion': cliente[7]
        })
    cursor.close()
    connection.close()

    return render(request, 'Clientes/indexClientes.html', {'clientes': clientes})
```

En la función index de las vistas de Django se declara un cursor que se conecte a la base de datos, y este cursor llama a la función de listar y crea un array para almacenar los datos, se cierra el cursor y se muestran los datos del array en el html.

Registro

Se crearon varios scripts para filtrar datos y seleccionar el cliente por cedula, colaborador por id y el producto por nombre, y cuando se selecciona el producto se carga el precio unitario y pone el máximo en el input de cantidad, el máximo es el stock disponible del producto seleccionado. Y cuando se seleccione la cantidad, se multiplica el precio unitario por la cantidad, más el 13% del IVA.

Vista del formulario de gestión de devoluciones

Gestion de Devoluciones Inicio / Devoluciones

Devoluciones		Factura	
Factura		Codigo factura	Cedula Cliente
Buscar factura...	Facturas ▼	<input type="text"/>	<input type="text"/>
Monto de devolución		Codigo producto	Cantidad
<input type="text" value="Monto de devolución"/>		<input type="text"/>	<input type="text"/>
<input type="button" value="Guardar"/>		Precio unitario	Total Pagado
		<input type="text"/>	<input type="text"/>

En esta vista se filtran las facturas que tengan un estado igual a 1, y también usa un script para seleccionar la factura por el código y se cargan los datos de la factura en el card que esta al lado para comprobar los datos, el monto de devolución se usa un script para restarle el 10% del total pagado.