|  |  |
| --- | --- |
| TRABAJO INTEGRADOR FINAL  Descripción del Proyecto, D.E.R, Código | Nombre del proyecto  Lab TIF  AUTOR  Aprile, Ignacio  MATERIA  Laboratorio 3  CARRERA  Tecnicatura universitaria en programación  AÑO CURSADO  Primer Cuatrimestre del 2do. Año  UNIVERSIDAD  Universidad Tecnológica Nacional  PROFESORES  Herrera, Ariel  Español, Diego  FECHA  10/03/2023 |

**INDICE**

Introduccion 2

Diagrama Entidad-Relación 3

Código *4*

Creación de la base de datos *4*

Creación de las tablas *4*

Relaciones entre las tablas *8*

Carga de datos *9*

Procedimientos Almacenados *12*

Vistas *14*

Triggers *15*

Consultas *16*

Examen Final *19*

# INTRODUCCIÓN

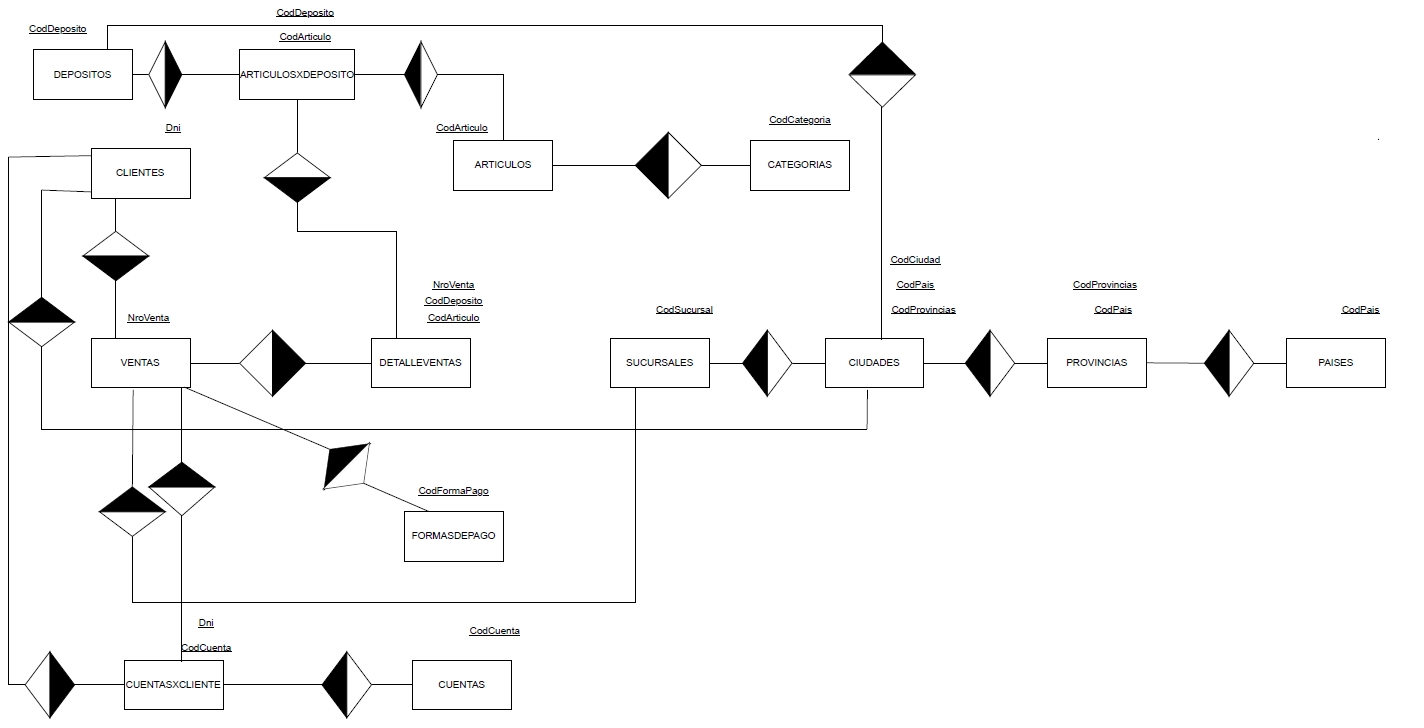
En el mundo empresarial actual, la gestión eficiente de la información y las relaciones con los clientes se ha convertido en un factor crítico para el éxito y la competitividad de las organizaciones. En este contexto, el desarrollo de sistemas de gestión de relaciones con los clientes (CRM, por sus siglas en inglés) se ha vuelto fundamental para optimizar las interacciones con los clientes y maximizar su satisfacción.

El presente trabajo final tiene como objetivo presentar el diseño y desarrollo de una base de datos para la empresa "Lab TIF". Lab TIF es una compañía dedicada a la investigación y desarrollo de productos electrónicos de alta calidad, comprometida con la innovación y la excelencia en el mercado. Conscientes de la importancia de una gestión efectiva de sus relaciones con los clientes, Lab TIF decidió implementar un sistema CRM personalizado para mejorar la interacción con sus clientes y fortalecer su posicionamiento en la industria.

A lo largo de este trabajo, se describirá en detalle el proceso de diseño de la base de datos CRM de Lab TIF, incluyendo la identificación de los requisitos del sistema, el modelado conceptual, el diseño lógico y físico de la base de datos, así como la implementación y las funcionalidades clave del sistema.

El desarrollo de esta base de datos CRM para Lab TIF no solo constituye un proyecto técnico y desafiante, sino también una oportunidad para impulsar la eficiencia operativa, el conocimiento del cliente y la toma de decisiones informada en la empresa. Sin duda, este trabajo representa un paso importante en la transformación digital de Lab TIF y su compromiso continuo con la excelencia en el servicio al cliente.

# D.E.R (Diagrama Entidad-Relación)

****

**CÓDIGO**

**Creación de la base de datos**

create database LabTif

GO

ON

(Name='LabTif\_DAT', --> nombre logico

FILENAME='F:\BASES DATOS\LabTif.MDF' --> ruta donde se guarde

)

GO

use LabTif

GO

**Creación de las tablas**

create table Paises (

CodPais char (5) not null,

Nombre varchar (50) not null,

constraint PK\_Paises primary key (CodPais)

)

GO

create table Provincias (

CodProvincia char (5) not null,

CodPais char (5) not null,

Nombre varchar (50) not null,

constraint PK\_Provincias primary key (CodProvincia,CodPais),

constraint FK\_Provincias\_Paises foreign key (CodPais)

references Paises (CodPais)

)

GO

create table Ciudades (

CodProvincia char (5) not null,

CodPais char (5) not null,

CodCiudad char (5) not null,

Nombre varchar (50) not null,

constraint PK\_Ciudades primary key (CodProvincia,CodPais,CodCiudad)

)

GO

create table Depositos (

CodDeposito char (5) not null,

CodProvincia char (5) not null,

CodPais char (5) not null,

CodCiudad char (5) not null,

Nombre char (50) not null,

Activo bit default 1,

constraint PK\_Depositos primary key (CodDeposito)

)

GO

create table Categorias (

CodCategoria char (5) not null,

Nombre varchar (50) not null,

constraint PK\_Categorias primary key (CodCategoria)

)

GO

create table Articulos (

CodArticulo char (5) not null,

CodCategoria char (5) not null,

Nombre varchar (50) not null,

Precio decimal (10,2) default 0,

--10 NUMEROS EN TOTAL, Y DE ESOS 10 LOS ULTIMOS DOS DECIMALES

Activo bit default 1,

constraint PK\_Articulos primary key (CodArticulo),

constraint FK\_Articulos\_Categorias foreign key (CodCategoria)

references Categorias (CodCategoria)

)

GO

create table ArticulosXDeposito (

CodDeposito char (5) not null,

CodArticulo char (5) not null,

Stock int default 0 check (Stock >= 0),

constraint PK\_ArticulosXDeposito primary key (CodDeposito,CodArticulo)

)

GO

create table FormasDePago (

CodFormaDePago char (5) not null,

TipoDePago varchar (50) not null,

constraint PK\_FormasDePago primary key (CodFormaDePago)

)

GO

create table Clientes (

Dni char (8) not null,

CodProvincia char (5) not null,

CodPais char (5) not null,

CodCiudad char (5) not null,

Nombre varchar (50) not null,

Apellido varchar (50) not null,

FechaNacimiento date not null,

Direccion varchar (50) not null,

Telefono varchar (50) not null,

Email varchar (50) not null,

Activo bit default 1,

constraint PK\_Clientes primary key (Dni)

)

GO

create table Cuentas (

NroCuenta char (16) not null,

Alias varchar (40) not null unique,

CodSeguridad char (4) not null,

Saldo decimal (10,2) default 0,

TipoDecuenta varchar (50) not null,

constraint PK\_Cuentas primary key (NroCuenta)

)

GO

create table CuentasXCliente (

Dni char (8) not null,

NroCuenta char (16) not null,

constraint PK\_CuentasXCliente primary key (Dni,NroCuenta)

)

GO

create table Sucursales (

CodSucursal char (5) not null,

CodProvincia char (5) not null,

CodPais char (5) not null,

CodCiudad char (5) not null,

Nombre varchar (50) not null,

Direccion varchar (50) null,

Telefono varchar (20) null,

Correo varchar (50) null,

Activo bit default 1,

constraint PK\_Sucursales primary key (CodSucursal)

--Para lo que es Online el nombre sera "ONLINE"

)

GO

create table Ventas (

NroVenta int identity (1,1) not null,

Dni char (8) not null,

NroCuenta char (16) not null,

CodSucursal char (5) not null,

CodFormaDePago char (5) not null,

FechaDeVenta date not null,

Total decimal (10,2) not null default 0

constraint PK\_Ventas primary key (NroVenta)

)

GO

create table DetalleVentas (

NroVenta int not null,

CodDeposito char (5) not null,

CodArticulo char (5) not null,

PrecioUnitario decimal (10,2) not null default 0,

Cantidad int not null,

constraint PK\_Detalle\_Ventas primary key (NroVenta,CodArticulo,CodDeposito)

)

**Relaciones entre las tablas**

alter table Ciudades add constraint

FK\_Ciudades\_Provincias foreign key (CodProvincia,CodPais)

references Provincias (CodProvincia,CodPais)

GO

alter table Depositos add constraint

FK\_Depositos\_Ciudades foreign key (CodProvincia,CodPais,CodCiudad)

references Ciudades (CodProvincia,CodPais,CodCiudad)

GO

alter table ArticulosXDeposito

add constraint FK\_ArticulosXDeposito\_Depositos foreign key (CodDeposito)

references Depositos (CodDeposito),

constraint FK\_ArticulosXDeposito\_Articulos foreign key (CodArticulo)

references Articulos (CodArticulo)

GO

alter table Clientes add constraint

FK\_Clientes\_Ciudades foreign key (CodProvincia,CodPais,CodCiudad)

references Ciudades (CodProvincia,CodPais,CodCiudad)

GO

alter table CuentasXCliente

add constraint FK\_CuentasXCliente\_Clientes foreign key (Dni)

references Clientes (Dni),

constraint FK\_CuentasXCliente\_Cuentas foreign key (NroCuenta)

references Cuentas (NroCuenta)

GO

alter table Sucursales add constraint

FK\_Sucursales\_Ciudades foreign key (CodProvincia,CodPais,CodCiudad)

references Ciudades (CodProvincia,CodPais,CodCiudad)

GO

alter table Ventas

add constraint FK\_Ventas\_Clientes foreign key (Dni)

references Clientes (Dni),

constraint FK\_Ventas\_Cuentas foreign key (NroCuenta)

references Cuentas (NroCuenta),

constraint FK\_Ventas\_Sucursales foreign key (CodSucursal)

references Sucursales (CodSucursal),

constraint FK\_Ventas\_FormasDePago foreign key (CodFormaDePago)

references FormasDePago (CodFormaDePago)

GO

alter table DetalleVentas

add constraint FK\_DetalleVentas\_Ventas foreign key (NroVenta)

references Ventas (NroVenta),

constraint FK\_DetalleVentas\_ArticulosXDeposito foreign key (CodDeposito,CodArticulo)

references ArticulosXDeposito (CodDeposito,CodArticulo)

GO

**Carga de datos**

insert into Paises (CodPais,Nombre)

select 'Argen','Argentina' union

select 'Brasi','Brasil' union

select 'Chile','Chile' union

select 'Peru1','Peru'

GO

insert into Provincias (CodProvincia,CodPais,Nombre)

select 'Cordo','Argen','Cordoba' union

select 'Riode','Brasi','RioDeJaneiro' union

select 'Santi','Chile','Santiago'union

select 'Lima1','Peru1','Lima'

GO

insert into Ciudades (CodProvincia,CodPais,CodCiudad,Nombre)

select 'Cordo','Argen','Cord1','Carlos Paz' union

select 'Riode','Brasi','Bras1','Rio de Janeiro' union

select 'Santi','Chile','Chil1','Santiago de Chile' union

select 'Lima1','Peru1','Comas','Comas'

GO

insert into Depositos (CodDeposito,CodProvincia,CodPais,CodCiudad,Nombre)

select 'Depo1','Cordo','Argen','Cord1','Deposito de Cordoba'

GO

insert into Categorias (CodCategoria,Nombre)

select 'GRAND','Grandes Electrodomesticos' union

select 'PEQUE','Pequeños Electrodomesticos' union

select 'INFOR','Informatica y Telecomunicaciones' union

select 'CONSU','Aparatos electronicos de consumo' union

select 'HERRA','Herramientas electricas'

GO

insert into Articulos (CodArticulo,CodCategoria,Nombre,Precio)

select 'GRAN1','GRAND','Heladera Coventry',150000 union

select 'GRAN2','GRAND','Lavarropas Samsung',230000 union

select 'GRAN3','GRAND','Aire Acodicionado LG',280000 union

select 'PEQU1','PEQUE','Plancha ATMA',50000 union

select 'PEQU2','PEQUE','Maquina Coser Singer',90000 union

select 'PEQU3','PEQUE','Freidora Liliana',40000 union

select 'INFO1','INFOR','Imprsora HP',70000 union

select 'INFO2','INFOR','Notebook Lenovo',300000 union

select 'INFO3','INFOR','Calculadora Casio',25000 union

select 'CONS1','CONSU','Radio Spica',15000 union

select 'CONS2','CONSU','TV LG',250000 union

select 'CONS3','CONSU','Videocamara Nikon',150000 union

select 'HERR1','HERRA','Soldadora',200000 union

select 'HERR2','HERRA','Taladro',50000 union

select 'HERR3','HERRA','Destornillador',30000

GO

INSERT INTO ArticulosXDeposito (CodDeposito, CodArticulo, Stock)

VALUES ('Depo1', 'GRAN1', 10),

('Depo1', 'PEQU2', 5);

insert into FormasDePago (CodFormaDePago,TipoDePago)

select 'Debit', 'Debito' union

select 'Credi', 'Credito' union

select 'Efect', 'Efectivo'

GO

insert into Clientes (Dni,CodProvincia,CodPais,CodCiudad,Nombre,Apellido,FechaNacimiento,Direccion,Telefono,Email)

select '29890134','Cordo','Argen','Cord1','Facundo','Lopez','1978-3-27','Los Alamos 322','1122334422','faculopez@hotmail.com' union

select '29890211','Riode','Brasi','Bras1','Carlos','Perez','1982-3-27','Cura Brochero 1223','1122335590','carlosperez@hotmail.com' union

select '29890300','Santi','Chile','Chil1','Jesus','Cabello','1989-3-27','Av Rio 322','1122334400','jesuscabello@hotmail.com' union

select '29890333','Lima1','Peru1','Comas','Fernando','Jesus','1970-3-27','Av Tupac Amaru 567','1122334432','fernandojesus@hotmail.com'

GO

--Tipos de cuenta podrian ser Caja Ahorro, Cuenta Corriente, Cuenta Sueldo

insert into Cuentas (NroCuenta,Alias,CodSeguridad,Saldo,TipoDecuenta)

select '4564232123456789','PASTO.VELOZ.DEBIL','0948',100000,'Caja de ahorro' union

select '5678093423214532','CASA.CERRO.ALTO','3456',200000,'CuentaCorriente' union

select '5645768567894356','CASA.N&Y.ALTO','2346',760000,'CuentaCorriente'

GO

insert into CuentasXCliente (Dni,NroCuenta)

select '29890134','4564232123456789' union

select '29890211','5678093423214532' union

select '29890134','5645768567894356'

GO

insert into Sucursales (CodSucursal,CodProvincia,CodPais,CodCiudad,Nombre)

select 'AAAA1','Cordo','Argen','Cord1','Cordoba S.A.' union

select 'BBBB1','Riode','Brasi','Bras1','Rio de Janeiro Hnos.' union

select 'PPPP1','Santi','Chile','Chil1','Santiago de Chile Company'

GO

**Procedimientos Almacenados**

--SP ActualizarStockArticulo

CREATE PROCEDURE ActualizarStockArticulo

@CodArticulo char(5),

@CodDep char (5),

@NuevoStock int

AS

BEGIN

-- Actualizar el stock del articulo en la tabla ArticulosXDeposito

UPDATE ArticulosXDeposito

SET Stock = @NuevoStock

WHERE CodArticulo = @CodArticulo and CodDeposito = @CodDep;

-- Mensaje de exito

SELECT 'Stock actualizado correctamente.' AS Result;

END

GO

--exec ActualizarStockArticulo 'GRAN1','Depo1',20

-- SP ActualizarTotalVenta

CREATE PROCEDURE ActualizarTotalVenta

@NroVenta int,

@NuevoTotal decimal(10, 2)

AS

BEGIN

-- Actualizar el total de la venta en la tabla Ventas

UPDATE Ventas

SET Total = @NuevoTotal

WHERE NroVenta = @NroVenta;

-- Mensaje de exito

SELECT 'Total de venta actualizado correctamente.' AS Result;

END

GO

--Proc almacenado le paso como parametro su dni y me muestra su mayor compra fecha total y sucursal

--Con este proc, puedo decidir que estrategia usar frente a un cliente para darle el valor para la empresa

create procedure SP\_mayorCompraHistoricaDni

@DniCliente char (8)

as

select concat (c.Nombre, ' ' , c.Apellido) as [Nombre Completo], v.CodSucursal, v.FechaDeVenta, max(v.Total) as [Mayor Compra]

from Ventas as v

inner join Clientes as c

on v.Dni = c.Dni

where v.Dni = @DniCliente

group by c.Nombre, c.Apellido, v.CodSucursal, v.FechaDeVenta

GO

--exec SP\_mayorCompraHistoricaDni '29890134'

--Proc almacenado para conocer la cantidad de cada artículo vendida históricamente.

create procedure SP\_articuloMasVendido

as

select distinct a.Nombre, SUM(dv.Cantidad) as [Mas Vendido], c.Nombre

from DetalleVentas as dv

inner join Articulos as a

on dv.CodArticulo = a.CodArticulo

inner join Categorias as c

on a.CodCategoria = c.CodCategoria

group by a.Nombre,c.Nombre

GO

exec SP\_articuloMasVendido

--Reporte indicándome las sucursales en las que compro un cliente (SP), con su ciudad y país.

--VER DE AGREGAR LO QUE GASTO EN TOTAL POR SUCURSAL.

create procedure SP\_sucursalesPorDni

@DniCliente char (8)

as

select distinct s.Nombre as Sucursal, c.Nombre as Ciudad, p.Nombre as Pais from sucursales as s

inner join Ventas as v on v.CodSucursal = s.CodSucursal

inner join Ciudades as c on s.CodCiudad = c.CodCiudad

inner join paises as p on c.CodPais = p.CodPais

where v.Dni = @DniCliente order by s.Nombre asc

GO

exec SP\_sucursalesPorDni '29890134'

--El procedimiento recibe un entero y devuelve los artículos con el stock por debajo de la cantidad

create procedure SP\_stockMenorA

@stock int

as

select art.CodArticulo as [Codigo Articulo], art.Nombre , art.Precio, axd.Stock, [nivel de necesidad] =

case when axd.stock < @stock then 'STOCK INSUFICIENTE'

else 'igual o mayor a ' + cast(@stock as varchar(6))

end

from Articulos art

inner join ArticulosXDeposito axd on art.CodArticulo = axd.CodArticulo

order by Stock asc

go

exec SP\_stockMenorA 20

--Obtener el detalle de una venta específica según

--el detalle de una venta, incluyendo los artículos comprados, la cantidad y el precio unitario.

CREATE PROCEDURE ObtenerDetalleVenta

@NroVenta INT

AS

BEGIN

SELECT dv.CodArticulo, a.Nombre AS Articulo, dv.Cantidad, dv.PrecioUnitario

FROM DetalleVentas dv

INNER JOIN Articulos a ON dv.CodArticulo = a.CodArticulo

WHERE dv.NroVenta = @NroVenta;

END

EXEC ObtenerDetalleVenta 3

GO

**Vistas**

--Vista que muestra la información de los clientes junto con la ciudad y el país en el que residen:

CREATE VIEW VistaClientes AS

SELECT c.Dni, c.Nombre, c.Apellido, c.FechaNacimiento, c.Direccion, c.Telefono, c.Email,

ci.Nombre AS Ciudad, p.Nombre AS Pais

FROM Clientes c

INNER JOIN Ciudades ci ON ci.CodProvincia = c.CodProvincia

AND ci.CodPais = c.CodPais

AND ci.CodCiudad = c.CodCiudad

INNER JOIN Paises p ON p.CodPais = c.CodPais

GO

SELECT \* FROM VistaClientes

GO

--Vista que muestra el stock de artículos por depósito y ciudad:

CREATE VIEW VistaStockArticulosxCiudad AS

SELECT c.Nombre as [Ciudad], dep.Nombre as [Deposito] ,art.CodArticulo as [Codigo Articulo], art.Nombre as [Nombre Articulo], axd.Stock

from Ciudades c

INNER JOIN Depositos dep on c.CodPais = dep.CodPais

AND c.CodProvincia = dep.CodProvincia

AND c.CodCiudad = dep.CodCiudad

INNER JOIN ArticulosXDeposito axd on dep.CodDeposito = axd.CodDeposito

INNER JOIN Articulos art on axd.CodArticulo = art.CodArticulo

GO

SELECT \* FROM VistaStockArticulosxCiudad

GO

**Triggers**

CREATE TRIGGER TR\_DetalleVentas ON DetalleVentas

INSTEAD OF INSERT

AS

BEGIN

-- Obtener el código del articulo ingresado

DECLARE @CodArticulo char(5);

SELECT @CodArticulo = CodArticulo FROM inserted;

-- Buscar el stock disponible para ese CodArt en la tabla ArticulosXDeposito para el Deposito ingresado

DECLARE @CodDep char (5);

SELECT @CodDep = CodDeposito FROM inserted;

DECLARE @StockDisponible int;

SELECT @StockDisponible = Stock FROM ArticulosXDeposito WHERE CodArticulo = @CodArticulo

and CodDeposito = @CodDep;

-- Comparar el stock disponible con la cantidad que se desea vender

IF @StockDisponible >= (SELECT Cantidad FROM inserted)

BEGIN

insert into DetalleVentas (NroVenta,CodDeposito,CodArticulo,PrecioUnitario,Cantidad)

select NroVenta,CodDeposito,CodArticulo,PrecioUnitario,Cantidad from inserted

-- Calcular el nuevo stock del articulo luego de la venta

DECLARE @NuevoStock int;

SELECT @NuevoStock = @StockDisponible - (SELECT Cantidad FROM inserted);

-- Llamar al stored procedure que descuenta el stock de artículos

EXEC ActualizarStockArticulo @CodArticulo,@CodDep, @NuevoStock;

-- Obtener el numero de venta desde inserted

DECLARE @NroVenta int;

SELECT @NroVenta = NroVenta FROM inserted;

-- Obtener el total hasta el momento de la tabla Ventas

DECLARE @TotalHastaElMomento decimal(10, 2);

SELECT @TotalHastaElMomento = Total FROM Ventas WHERE NroVenta = @NroVenta;

-- Calcular el total en este detalle

DECLARE @PrecioUnitario decimal(10, 2);

SELECT @PrecioUnitario = PrecioUnitario FROM inserted;

DECLARE @Cantidad int;

SELECT @Cantidad = Cantidad FROM inserted;

DECLARE @TotalDetalle decimal(10, 2);

SET @TotalDetalle = @PrecioUnitario \* @Cantidad;

-- Calcular el nuevo total en Ventas

DECLARE @NuevoTotal decimal(10, 2);

SELECT @NuevoTotal = @TotalHastaElMomento + @TotalDetalle;

-- Actualizar el total en la tabla Ventas para el numero de venta correspondiente

EXEC ActualizarTotalVenta @NroVenta, @NuevoTotal;

-- Lanzar mensaje por "consola"

SELECT 'Venta realizada correctamente' AS Result;

END

ELSE

BEGIN

-- Si el stock disponible es insuficiente, lanzar un mensaje de error

RAISERROR('No hay suficiente stock disponible', 16, 1);

rollback;

END

END

GO

**Consultas**

--PRUEBO CARGANDO PRIMERO UNA VENTA

insert into Ventas (Dni, NroCuenta, CodSucursal, CodFormaDePago, FechaDeVenta)

VALUES ('29890134', '5678093423214532', 'PPPP1', 'Efect', GETDATE())

--GO

-- Mostrar la venta insertada

--SELECT \* FROM Ventas WHERE NroVenta = SCOPE\_IDENTITY()

--GO

--PRUEBO EL TRIGGER procesoDetalleVentas AL INSERTAR UN DETALLE

--AL EVALUAR LA FORMA DE OBTENER DIRECTO EL PRECIO UNITARIO, SIN TIPEARLO

INSERT INTO DetalleVentas (NroVenta, CodDeposito, CodArticulo, PrecioUnitario, Cantidad)

VALUES (5, 'Depo1', 'Gran1', 150000, 1);

--GO

--select\*from DetalleVentas

--GO

--PRUEBO VENDER MAS DE LO QUE TENGO EN STOCK

INSERT INTO DetalleVentas (NroVenta, CodDeposito, CodArticulo, PrecioUnitario, Cantidad)

VALUES (1, 'Depo1', 'PEQU2', 150000, 20);

--GO

--Consulta para definir el Valor de un cliente para una empresa en base a la suma de sus gastos historicos (Reporte)

--case

SELECT Dni, Nombre, Apellido,

CASE WHEN TotalGastos >= 1000000 THEN 'Cliente VIP'

WHEN TotalGastos >= 500000 THEN 'Cliente Preferencial'

WHEN TotalGastos >= 100000 THEN 'Cliente Regular'

ELSE 'Cliente Estándar'

END AS ValorCliente

FROM

(

SELECT V.Dni, C.Nombre, C.Apellido, SUM(V.Total) AS TotalGastos

FROM Ventas V INNER JOIN Clientes C ON V.Dni = C.Dni GROUP BY V.Dni, C.Nombre, C.Apellido

)

AS GastosCliente;

GO

--Obtener la lista de productos disponibles en una categoría específica

SELECT \* FROM Articulos

WHERE CodCategoria = 'GRAND';

GO

--Verificar el stock disponible de un articulo en un deposito específico

SELECT axd.Stock, axd.CodArticulo, axd.CodDeposito, d.Activo, d.CodPais, d.CodProvincia, d.CodCiudad FROM ArticulosXDeposito axd

INNER JOIN Depositos d ON axd.CodDeposito = d.CodDeposito

WHERE axd.CodArticulo = 'GRAN1' AND d.Nombre = 'Deposito de Cordoba';

GO

--Consulta cuentas de un dni especifico

select cli.Dni, concat (cli.Nombre, ' ' , cli.Apellido) as [Nombre Completo],c.NroCuenta

from CuentasXCliente cxc

inner join Cuentas c on cxc.NroCuenta =c.NroCuenta

inner join Clientes cli on cxc.Dni = cli.Dni

where cli.Dni = '29890211'

go

--Consulta para definir el Valor de un cliente para una empresa en base a la suma de sus gastos históricos (Reporte)

--case

SELECT Dni, Nombre, Apellido,

CASE WHEN TotalGastos >= 1000000 THEN 'Cliente VIP'

WHEN TotalGastos >= 300000 and TotalGastos < 1000000 THEN 'Cliente Preferencial'

WHEN TotalGastos >= 100000 and TotalGastos < 300000 THEN 'Cliente Regular'

ELSE 'Cliente Estandar'

END AS ValorCliente

FROM

(

SELECT V.Dni, C.Nombre, C.Apellido, SUM(V.Total) AS TotalGastos

FROM Ventas V INNER JOIN Clientes C ON V.Dni = C.Dni GROUP BY V.Dni, C.Nombre, C.Apellido

)

AS GastosCliente;

GO

--Obtener la lista de productos disponibles en una categoría específica

SELECT \* FROM Articulos

WHERE CodCategoria = 'GRAND';

GO

--Verificar el stock disponible de un articulo en un deposito específico

SELECT axd.Stock, axd.CodArticulo, axd.CodDeposito, d.Activo, d.CodPais, d.CodProvincia, d.CodCiudad FROM ArticulosXDeposito axd

INNER JOIN Depositos d ON axd.CodDeposito = d.CodDeposito

WHERE axd.CodArticulo = 'GRAN1' AND d.Nombre = 'Deposito de Cordoba';

GO

**UTN-FRGP/TUP FINAL LABORATORIO DE COMPUTACIÓN III**

|  |  |  |
| --- | --- | --- |
| Nombre y Apellido: | | |
| DNI: | | Legajo: |
| Fecha: | | EXAMEN FINAL Tomo: Folio: |
| CALIFICACIÓN |  | |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| COMPETENCIAS ESPECIFICAS  EVALUADAS | **0**  **No lo**  **hace** | **2**  **Lo hace con errores sustanciales** | **4**  **Lo hace con errores que deben mejorarse** | **6**  **Lo hace correctamente**  **(Aceptable)** | **8**  **Lo hace muy bien** | **10**  **Lo hace excelente mente** |
| Aplicar vocabulario y terminología propia de la materia. |  |  |  |  |  |  |
| Aplicar los resultados del autoaprendizaje, actualización e investigación en la resolución de problemas. |  |  |  |  |  |  |
| Resolver situaciones problemáticas planteadas como proyectos interdisciplinares integradores. |  |  |  |  |  |  |
| Desempeñar las actividades dentro del equipo con responsabilidad individual, grupal, ética y social, en tiempo y forma |  |  |  |  |  |  |
| Participar activamente en el aula (virtual o presencial), realizando aportes a la comunidad de aprendizaje. |  |  |  |  |  |  |