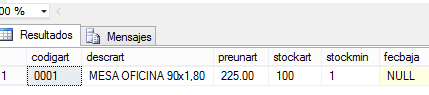
1. **Muestra el primer registro de la tabla artículos en la BD Compras. Usa cursores**



USE compras

GO

DECLARE @codigart AS char(6)

DECLARE @descrart AS char(40)

DECLARE @preunart AS decimal(9, 2)

DECLARE @stockart AS INT

DECLARE @stockmin AS INT

DECLARE @fecbaja AS DATE

DECLARE PrimerRegistroArticulos CURSOR FOR

SELECT codigart, descrart, preunart,stockart, stockmin, fecbaja FROM Articulos

OPEN PrimerRegistroArticulos

FETCH NEXT FROM PrimerRegistroArticulos

WHILE @@FETCH\_STATUS = 0

BEGIN

FETCH NEXT FROM PrimerRegistroArticulos INTO @codigart, @descrart, @preunart,@stockart, @stockmin, @fecbaja

END

GO

CLOSE PrimerRegistroArticulos

DEALLOCATE PrimerRegistroArticulos

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

USE compras

GO

DECLARE PrimerRegistroArticulos CURSOR FOR

SELECT TOP 1 \* FROM Articulos

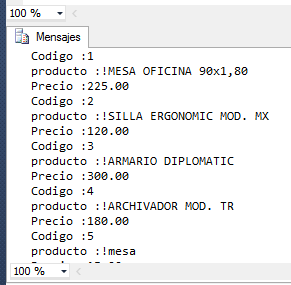
OPEN PrimerRegistroArticulos

FETCH NEXT FROM PrimerRegistroArticulos

CLOSE PrimerRegistroArticulos

DEALLOCATE PrimerRegistroArticulos

1. **Cursor que permita mostrar por pantalla, en la forma que se indica, el código del artículo, descripción y precio para todos los artículos de la BD compras.**



USE Compras

GO

DECLARE @codigart AS char(6)

DECLARE @descrart AS char(40)

DECLARE @preunart AS decimal(9, 2)

DECLARE LISTA CURSOR LOCAL FOR

SELECT codigart, descrart, preunart FROM Articulos

OPEN LISTA

FETCH NEXT FROM LISTA INTO @codigart, @descrart, @preunart

WHILE @@FETCH\_STATUS = 0

BEGIN

PRINT 'Codigo: ' + @codigart

PRINT 'Descripcion: ' + @descrart

PRINT 'Precio: ' + cast(@preunart as char) ;

FETCH LISTA INTO @codigart, @descrart, @preunart

end

--cuando acabe esto habre impreso toda la descripcion del articulo---

close LISTA

--lo cierro lo puedo reusar--

deallocate LISTA

--lo mato ya no lo puedo usar--

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*corrección gemma

USE Compras

GO

DECLARE @codigart AS char(6)

DECLARE @descrart AS char(40)

DECLARE @preunart AS decimal(9, 2)

DECLARE LISTA CURSOR FOR

SELECT codigart, descrart, preunart FROM Articulos

OPEN LISTA

FETCH NEXT FROM LISTA INTO @codigart, @descrart, @preunart

WHILE @@FETCH\_STATUS = 0

BEGIN

PRINT 'Codigo: ' + cast(@codigart as varchar(10))

PRINT 'Descripcion: ' + @descrart

PRINT 'Precio: ' + cast(@preunart as varchar(10)) ;

FETCH next from LISTA INTO @codigart, @descrart, @preunart

end

--cuando acabe esto habre impreso toda la descripcion del articulo---

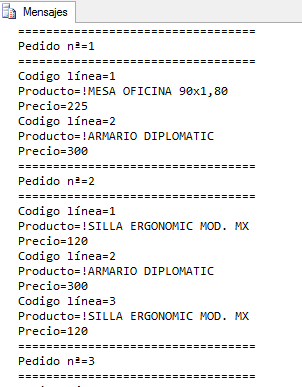
close LISTA

--lo cierro lo puedo reusar--

deallocate LISTA

--lo mato ya no lo puedo usar--

1. **Cursor que permita mostrar las líneas de pedido de cada pedido en la BD Compras de la siguiente manera.**



USE Compras

GO

DECLARE @numeroPedido AS INT

DECLARE @numeroPedido1 AS INT

DECLARE @numlin AS smallint

DECLARE @descrart AS char(40)

DECLARE @preunart AS decimal(9, 2)

DECLARE LISTA CURSOR LOCAL FOR

SELECT Lineas.numped, Lineas.numlin, Articulos.descrart, Articulos.preunart FROM Articulos

JOIN Lineas ON Articulos.codigart = Lineas.codigart

OPEN LISTA

FETCH NEXT FROM LISTA INTO @numeroPedido,@numlin, @descrart, @preunart

WHILE @@FETCH\_STATUS = 0

BEGIN

PRINT '======================='

PRINT 'PEDIDO: ' + cast(@numeroPedido as char)

PRINT '======================='

PRINT 'Codigo: ' + cast(@numlin as char)

PRINT 'Descripcion: ' + @descrart

PRINT 'Precio: ' + cast(@preunart as char)

PRINT '=======================' ;

SET @numeroPedido1 = @numeroPedido

FETCH LISTA INTO @numeroPedido,@numlin, @descrart, @preunart

WHILE @@FETCH\_STATUS = 0 AND @numeroPedido1 = @numeroPedido

BEGIN

PRINT 'Codigo: ' + cast(@numlin as char)

PRINT 'Descripcion: ' + @descrart

PRINT 'Precio: ' + cast(@preunart as char)

PRINT '=======================' ;

SET @numeroPedido1 = @numeroPedido

FETCH LISTA INTO @numeroPedido,@numlin, @descrart, @preunart

END

end

--cuando acabe esto habre impreso toda la descripcion del articulo---

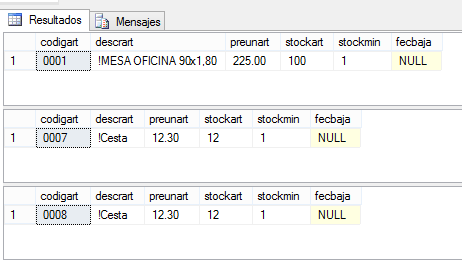
close LISTA

--lo cierro lo puedo reusar--

deallocate LISTA

--lo mato ya no lo puedo usar--

1. **Defina un cursor dinámico que permita visualizar: el primer registro, el registro en la posición 6 y el último registro de la tabla artículos de la BD Compras**



USE compras

GO

DECLARE RegistroArticulos CURSOR SCROLL FOR

SELECT \* FROM Articulos

OPEN RegistroArticulos

-- IMPRIMIR LOS REGISTROS

FETCH FIRST FROM RegistroArticulos

FETCH ABSOLUTE 6 FROM RegistroArticulos

FETCH LAST FROM RegistroArticulos

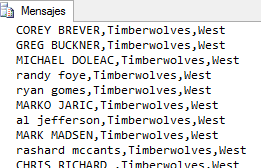
-- CERRAR

CLOSE RegistroArticulos

-- LIBERAR

DEALLOCATE RegistroArticulos

1. ***Trabajando con un cursor, listar nombre, nombre del equipo y Conferencia de los jugadores de la nba registrados en la base de datos.***



USE nba

GO

DECLARE @nombre AS VARCHAR

DECLARE @nombreEquipo AS VARCHAR

DECLARE @conferencia AS VARCHAR

DECLARE LISTA CURSOR FOR

SELECT jugadores.Nombre, jugadores.Nombre\_Equipo, Conferencia FROM jugadores

JOIN equipos ON jugadores.Nombre\_Equipo = equipos.Nombre

OPEN LISTA

FETCH LISTA INTO @nombre,@nombreEquipo, @conferencia

WHILE @@FETCH\_STATUS = 0

BEGIN

PRINT @nombre + ' , ' + @nombreEquipo + ' , ' + @conferencia

FETCH LISTA INTO @nombre,@nombreEquipo, @conferencia

end

close LISTA

--lo cierro lo puedo reusar--

deallocate LISTA

--lo mato ya no lo pued

1. **Sobre la BD Jardinería y trabajando con un cursor, listar la relación de los clientes hayan registrado pedidos o no. En dicho proceso, debemos imprimir el nombre del cliente, la cantidad de pedidos registrados a su nombre. Si no ha realizado pedidos debe aparecer 0. Al finalizar totalizar el proceso (suma total de todos los pedidos realizados por todos los clientes).**



--Sobre la BD Jardinería y trabajando con un cursor, listar la relación de los clientes hayan registrado pedidos o no.

--En dicho proceso, debemos imprimir el nombre del cliente, la cantidad de pedidos registrados a su nombre.

--Si no ha realizado pedidos debe aparecer 0. Al finalizar totalizar el proceso (suma total de todos los pedidos realizados

--por todos los clientes).

USE jardineria1

GO

DECLARE @CLIENTE AS VARCHAR(50)

DECLARE @CANTIDAD\_PEDIDO AS INT

DECLARE @CANTIDAD\_TOTAL AS INT

DECLARE LISTA CURSOR LOCAL FOR

SELECT Clientes.NombreCliente , COUNT(Pedidos.CodigoPedido), SUM(Pedidos.CodigoPedido)

FROM Clientes ,Pedidos

WHERE Clientes.CodigoCliente IN (SELECT Pedidos.CodigoCliente FROM Pedidos) GROUP BY Clientes.NombreCliente

OPEN LISTA

FETCH NEXT FROM LISTA INTO @CLIENTE, @CANTIDAD\_PEDIDO, @CANTIDAD\_TOTAL

WHILE @@FETCH\_STATUS = 0

BEGIN

PRINT cast(@CLIENTE as char) + ' ' + cast(@CANTIDAD\_PEDIDO as char)

PRINT cast(@CANTIDAD\_TOTAL as char)

FETCH LISTA INTO @CLIENTE, @CANTIDAD\_PEDIDO, @CANTIDAD\_TOTAL

end

close LISTA

--lo cierro lo puedo reusar--

deallocate LISTA

--lo mato ya no lo puedo usar--

USE jardineria1

GO

DECLARE @CLIENTE AS VARCHAR(50)

DECLARE @CANTIDAD\_PEDIDO AS INT

DECLARE @CANTIDAD\_TOTAL AS INT

SET @CANTIDAD\_TOTAL = 0

DECLARE LISTA CURSOR LOCAL FOR

SELECT Clientes.NombreCliente , COUNT(Pedidos.CodigoPedido)

FROM Clientes ,Pedidos

WHERE Clientes.CodigoCliente IN (SELECT Pedidos.CodigoCliente FROM Pedidos) GROUP BY Clientes.NombreCliente

-- ABRIR

OPEN LISTA

-- LEER EL PRIMER REGISTRO

FETCH NEXT FROM LISTA INTO @CLIENTE, @CANTIDAD\_PEDIDO

-- MIENTRAS PUEDA LEER EL REGISTRO

WHILE @@FETCH\_STATUS = 0

BEGIN

-- IMPRIMIR PRIMER REGISTRO

PRINT cast(@CLIENTE as char) + ' ' + cast(@CANTIDAD\_PEDIDO as char)

-- ACUMULAR

SET @CANTIDAD\_TOTAL += @CANTIDAD\_PEDIDO

-- LEER EL REGISTRO SIGUIENTE

FETCH LISTA INTO @CLIENTE, @CANTIDAD\_PEDIDO

end

close LISTA

--lo cierro lo puedo reusar--

deallocate LISTA

--lo mato ya no lo puedo usar--

-- IMPRIMIR

PRINT 'TOTAL DE PEDIDOS:' + CAST(@CANTIDAD\_TOTAL AS VARCHAR)

GO

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*CORRECUION GEMMA

USE jardineria

GO

DECLARE @CLIENTE AS VARCHAR(50)

DECLARE @CANTIDAD\_PEDIDO AS INT

DECLARE @CANTIDAD\_TOTAL AS INT

DECLARE LISTA CURSOR LOCAL FOR

SELECT Clientes.NombreCliente , ISNULL(COUNT(Pedidos.CodigoCliente),0), SUM(Pedidos.CodigoPedido)

FROM Clientes LEFT JOIN Pedidos

ON Clientes.CodigoCliente = Pedidos.CodigoCliente GROUP BY Clientes.NombreCliente

OPEN LISTA

FETCH NEXT FROM LISTA INTO @CLIENTE, @CANTIDAD\_PEDIDO, @CANTIDAD\_TOTAL

WHILE @@FETCH\_STATUS = 0

BEGIN

PRINT cast(@CLIENTE as char) + ' ' + cast(@CANTIDAD\_PEDIDO as char)

PRINT cast(@CANTIDAD\_TOTAL as char)

FETCH LISTA INTO @CLIENTE, @CANTIDAD\_PEDIDO, @CANTIDAD\_TOTAL

end

close LISTA

--lo cierro lo puedo reusar--

deallocate LISTA

--lo mato ya no lo puedo usar--

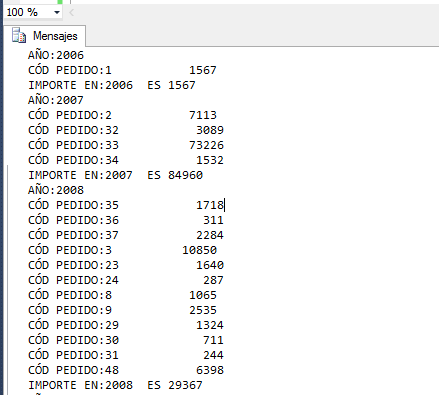
close LISTA

--lo cierro lo puedo reusar--

deallocate LISTA

--lo mato ya no lo puedo usar--

1. Sobre la BD Jardinería y trabajando con un cursor, saca por pantalla un listado de los pedidos realizados por cada año. La forma de presentación es : listar por cada año los pedidos registrados y totalizar dichos pedidos por dicho año.



--Sobre la BD Jardinería y trabajando con un cursor, saca por pantalla un listado de los pedidos realizados por cada año.

--La forma de presentación es : listar por cada año los pedidos registrados y totalizar dichos pedidos por dicho año.

USE jardineria1

GO

DECLARE @AÑOPEDIDO AS INT

DECLARE @CODIGOPEDIDO AS INT

DECLARE @CANTIDAD AS INT

DECLARE @CANTIDAD\_TOTAL\_AÑO AS INT

DECLARE LISTA CURSOR LOCAL FOR

SELECT YEAR(Pedidos.FechaPedido) AS AÑO, Pedidos.CodigoPedido, DetallePedidos.Cantidad ,

SUM(DetallePedidos.Cantidad) AS CANTIDAD\_TOTAL\_AÑO

FROM Pedidos

JOIN DetallePedidos ON Pedidos.CodigoPedido = DetallePedidos.CodigoPedido

GROUP BY Pedidos.CodigoPedido, YEAR(Pedidos.FechaPedido), DetallePedidos.Cantidad

OPEN LISTA

FETCH NEXT FROM LISTA INTO @AÑOPEDIDO,@CODIGOPEDIDO, @CANTIDAD,@CANTIDAD\_TOTAL\_AÑO

WHILE @@FETCH\_STATUS = 0

BEGIN

PRINT 'AÑO PEDIDO: ' + cast(@AÑOPEDIDO as char)

PRINT '======================='

PRINT 'CODIGO PEDIDO: ' + cast(@CODIGOPEDIDO as char) + ' ,CANTIDAD PEDIDO: ' + cast(@CANTIDAD as char)

PRINT 'IMPORTE: ' + cast(@CANTIDAD\_TOTAL\_AÑO as char) + 'AÑO PEDIDO: ' + cast(@AÑOPEDIDO as char)

PRINT '=======================' ;

FETCH LISTA INTO @AÑOPEDIDO,@CODIGOPEDIDO, @CANTIDAD,@CANTIDAD\_TOTAL\_AÑO

end

close LISTA

--lo cierro lo puedo reusar--

deallocate LISTA

--lo mato ya no lo puedo usar--

USE jardineria1

GO

DECLARE @AÑO AS INT

DECLARE @AÑOPEDIDO AS INT

DECLARE @CODIGOPEDIDO AS INT

DECLARE @CANTIDAD AS INT

DECLARE @CANTIDAD\_TOTAL\_AÑO AS INT

SET @CANTIDAD\_TOTAL\_AÑO=0

DECLARE LISTA CURSOR LOCAL FOR

SELECT YEAR(Pedidos.FechaPedido) AS AÑO, Pedidos.CodigoPedido,

SUM(DetallePedidos.Cantidad\*PrecioUnidad) AS CANTIDAD\_TOTAL\_AÑO

FROM Pedidos

JOIN DetallePedidos ON Pedidos.CodigoPedido = DetallePedidos.CodigoPedido

GROUP BY YEAR(Pedidos.FechaPedido),Pedidos.CodigoPedido

ORDER BY 1

OPEN LISTA

FETCH LISTA INTO @AÑOPEDIDO,@CODIGOPEDIDO, @CANTIDAD

-- ASIGNAR A LA VARIABLE @Y1 EL VALOR INICIAL DE @Y

SET @AÑOPEDIDO = @AÑO

-- IMPRIMIR EL PRIMER AÑO

PRINT 'AÑO:' + CAST(@AÑOPEDIDO AS VARCHAR)

WHILE @@FETCH\_STATUS=0

BEGIN

-- SI COINCIDEN LOS VALORES ACUMULAR EL TOTAL

IF(@AÑO = @AÑOPEDIDO)

SET @CANTIDAD += @CANTIDAD\_TOTAL\_AÑO

ELSE

BEGIN

PRINT 'IMPORTE EN:' +CAST(@AÑOPEDIDO AS VARCHAR) +

SPACE(2)+ 'ES ' +CAST(@CANTIDAD\_TOTAL\_AÑO AS VARCHAR)

PRINT 'AÑO:' + CAST(@AÑO AS VARCHAR)

SET @AÑOPEDIDO=@AÑO

SET @CANTIDAD\_TOTAL\_AÑO=@CANTIDAD

END

-- IMPRIMIR EL REGISTRO

PRINT 'PEDIDO ' + CAST(@CODIGOPEDIDO AS VARCHAR) + SPACE(5)+STR(@CANTIDAD)

-- LEER EL REGISTRO SIGUIENTE

FETCH LISTA INTO @AÑOPEDIDO,@CODIGOPEDIDO, @CANTIDAD

END

-- CERRAR

CLOSE LISTA

-- LIBERAR

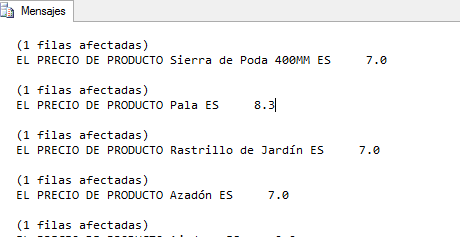
DEALLOCATE LISTA;

-- IMPRIMIR LOS TOTALES FINALES

PRINT 'IMPORTE EN:'+CAST(@AÑOPEDIDO AS VARCHAR)+SPACE(2)+'ES

'+STR(@CANTIDAD\_TOTAL\_AÑO)

1. **En la bd Jardinería, actualizar el precio de los productos: si su stock es mayor o igual a 100, se descuenta al precio del Proveedor el 50%, sino se descuenta al precio del Proveedor el 20%.**



--En la bd Jardinería, actualizar el precio de los productos: si su stock es mayor o igual a 100, se descuenta

--al precio del Proveedor el 50%, sino se descuenta al precio del Proveedor el 20%.

USE jardineria;

go

DECLARE @Nombre AS VARCHAR(400)

DECLARE @PrecioProveedor AS NUMERIC(15, 2)

--CURSORES

DECLARE ACTUALIZARPRECIOS CURSOR DYNAMIC FORWARD\_ONLY FOR

SELECT Nombre, PrecioProveedor FROM Productos FOR UPDATE OF PrecioProveedor

--ABRIR

OPEN ACTUALIZARPRECIOS

--IMPRIMIR EL PRIMER REGISTRO

FETCH NEXT FROM ACTUALIZARPRECIOS INTO @Nombre, @PrecioProveedor

WHILE @@fetch\_status = 0

BEGIN

UPDATE Productos

SET

@PrecioProveedor =

(

CASE

WHEN CantidadEnStock >= 100 THEN PrecioProveedor - ((PrecioProveedor/100)\*50)

ELSE PrecioProveedor - ((PrecioProveedor/100)\*20)

END

)

WHERE CURRENT OF ACTUALIZARPRECIOS

FETCH NEXT FROM ACTUALIZARPRECIOS INTO @Nombre, @PrecioProveedor

PRINT 'EL PRECIO DEL PRODUCTO ' + @Nombre + ' Y EL PRECIO DESCONTADO AHORA ES ' + CAST(@PrecioProveedor AS CHAR)

END

CLOSE ACTUALIZARPRECIOS

DEALLOCATE ACTUALIZARPRECIOS

1. **Sobre la bd Banco, implementar, usando cursores, un procedimiento que muestre los datos del cliente, la cuenta y el saldo de los clientes con saldo negativo en alguna de sus cuentas.**

--Sobre la bd Banco, implementar, usando cursores, un procedimiento que muestre los datos del cliente, la cuenta y el saldo de

-- los clientes con saldo negativo en alguna de sus cuentas.

USE banco;

GO

CREATE PROCEDURE procedimientoBANCO3

AS

BEGIN

SET NOCOUNT ON;

DECLARE @CODIGO\_CLIENTE AS INT,

@DNI AS INT,

@NOMBRE AS VARCHAR(20) ,

@APELLIDO1 AS VARCHAR(20) ,

@APELLIDO2 AS VARCHAR(20) ,

@DIRECCION AS VARCHAR(20)

DECLARE procedimientoBANCO2 CURSOR FOR

--SELECT codigo\_cliente, dni, nombre, apellido1, apellido2, direccion FROM Cliente

--WHERE Cliente.dni IN (SELECT tiene.dni FROM tiene WHERE tiene.cod\_cuenta IN (SELECT cuenta.cod\_cuenta FROM cuenta

--WHERE saldo < 0))

--LO HE HECHO ASI PORQUE NO HAY CLIENTES CON SALDO NEGATIVO Y HE INCLUIDO UN MOVIMIENTO NEGATIVO INSERT

SELECT codigo\_cliente, dni, nombre, apellido1, apellido2, direccion FROM Cliente

WHERE Cliente.dni IN (SELECT movimiento.dni FROM movimiento

WHERE cantidad < 0)

OPEN procedimientoBANCO2

FETCH NEXT FROM procedimientoBANCO2

WHILE @@FETCH\_STATUS = 0

BEGIN

FETCH NEXT FROM procedimientoBANCO2 INTO @CODIGO\_CLIENTE ,@DNI,@NOMBRE,@APELLIDO1,@APELLIDO2,@DIRECCION

END

END

GO

EXEC procedimientoBANCO3

GO

CLOSE procedimientoBANCO2

DEALLOCATE procedimientoBANCO2

1. **Sobre la bd Banco, calcular con un procedimiento la suma de todos los ingresos y cargos (por separado) juntando todas las cuentas.**

CREATE PROCEDURE procedimientoSUMACARGOSTODOS

AS

BEGIN

SET NOCOUNT ON

DECLARE @SUMA AS VARCHAR(200) ---CUENTA SALDO

DECLARE @CARGOS AS VARCHAR(200) --MOVIMIENTO CANTIDAD

DECLARE procedimientoSUMACARGOS2 CURSOR FOR

SELECT SUM(cuenta.saldo) AS SUMA\_TOTAL\_SALDOS, SUM(movimiento.cantidad) AS SUMA\_TOTAL\_CARGOS FROM cuenta, movimiento

OPEN procedimientoSUMACARGOS2

FETCH NEXT FROM procedimientoSUMACARGOS2

WHILE @@FETCH\_STATUS = 0

BEGIN

FETCH NEXT FROM procedimientoSUMACARGOS2 INTO @SUMA, @CARGOS

PRINT 'LA SUMA DEL SALDO DE LAS CUENTAS ES ' + @SUMA + ' Y LA SUMA DE LOS CARGOS DE LAS CUENTAS ESES' + @CARGOS

END

END

CLOSE procedimientoSUMACARGOS2

DEALLOCATE procedimientoSUMACARGOS2

GO

EXEC procedimientoSUMACARGOSTODOS

GO

1. **Modifica el procedimiento anterior para que sobre la bd Banco, calcule la suma de todos los ingresos y cargos (por separado) para una cuenta que se le pasa como parámetro.**

USE banco;

GO

CREATE PROCEDURE procedimientoSUMACARGOS4 (@numerocliente INT = 117)

AS

SET NOCOUNT ON;

DECLARE @SUMA AS VARCHAR(200) ---CUENTA SALDO

DECLARE @CARGOS AS VARCHAR(200) --MOVIMIENTO CANTIDAD

DECLARE procedimientoSUMACARGOS2 CURSOR FOR

SELECT movimiento.dni , SUM(cuenta.saldo), COUNT(movimiento.id\_movimiento) FROM cuenta, movimiento

WHERE cuenta.cod\_cuenta = movimiento.cod\_cuenta and movimiento.dni = @numerocliente

GROUP BY movimiento.dni;

OPEN procedimientoSUMACARGOS2

FETCH NEXT FROM procedimientoSUMACARGOS2 INTO @numerocliente, @SUMA, @CARGOS

WHILE @@FETCH\_STATUS = 0

BEGIN

FETCH NEXT FROM procedimientoSUMACARGOS2 INTO @numerocliente, @SUMA, @CARGOS

PRINT 'EL CODIGO DEL CLIENTE ES ' + CAST(@numerocliente AS VARCHAR) + ' ,LA SUMA DEL SALDO ES ' + CAST(@SUMA AS VARCHAR) + ' Y LA SUMA DE LOS CARGOS ES' + CAST(@CARGOS AS VARCHAR)

END

CLOSE procedimientoSUMACARGOS2

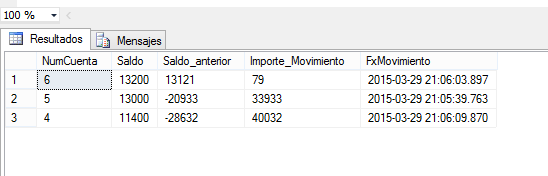
DEALLOCATE procedimientoSUMACARGOS2

GO

EXEC procedimientoSUMACARGOS4

GO

1. Implementa sobre la bd Banco una función que busca la tres cuentas con mayor saldo y obtiene para cada una de ellas: el saldo actual, saldo anterior, la fecha del último movimiento y el importe del movimiento. Usa cursores



--Implementa sobre la bd Banco una función que busca la tres cuentas con mayor saldo y obtiene para cada una de ellas:

--el saldo actual, saldo anterior, la fecha del último movimiento y el importe del movimiento. Usa cursores

USE banco;

GO

CREATE FUNCTION F\_TRES\_CUENTAS\_MAYORES(@TIPO INT) RETURNS TABLE

AS

BEGIN

DECLARE @CUENTA AS INT

DECLARE @SALDO\_ACTUAL INT

DECLARE @SALDO\_ANTERIOR AS INT

DECLARE @ULTIMA\_FECHA AS DATETIME

DECLARE C\_TRES\_CUENTAS\_MAYORES CURSOR FOR

SELECT cuenta.cod\_cuenta AS CUENTA , cuenta.saldo AS SALDO\_ACTUAL,(cuenta.saldo - movimiento.cantidad) AS SALDO\_ANTERIOR,

MAX(movimiento.fecha) AS ULTIMA\_FECHA FROM cuenta, movimiento

WHERE cuenta.cod\_cuenta = movimiento.cod\_cuenta AND cuenta.tipo = @TIPO

GROUP BY cuenta.cod\_cuenta

OPEN C\_TRES\_CUENTAS\_MAYORES

FETCH NEXT FROM C\_TRES\_CUENTAS\_MAYORES

WHILE @@FETCH\_STATUS = 0

BEGIN

WHILE @CUENTA <3

FETCH NEXT FROM procedimientoBANCO2 INTO @CUENTA,@SALDO\_ACTUAL,@SALDO\_ANTERIOR, @ULTIMA\_FECHA

END

CLOSE procedimientoSUMACARGOS2

DEALLOCATE procedimientoSUMACARGOS2

END

GO

SELECT \* FROM F\_TRES\_CUENTAS\_MAYORES(1)

GO

USE banco;

GO

CREATE FUNCTION F\_TRES\_CUENTAS\_MAYORES() RETURNS @TABLA TABLE

( CUENTA INT,

SALDO\_ACTUAL INT ,

SALDO\_ANTERIOR INT,

IMPORTE\_MOVIMIENTO INT,

ULTIMA\_FECHA DATETIME

)

AS

BEGIN

DECLARE @idcuenta INT

DECLARE @numcuenta VARCHAR(20)

DECLARE @saldo DECIMAL(10,2)

DECLARE C\_TRES\_CUENTAS\_MAYORES CURSOR FOR

SELECT TOP 3 cuenta.cod\_cuenta AS CUENTA , cuenta.saldo AS SALDO\_ACTUAL

FROM cuenta

ORDER BY cuenta.saldo DESC

OPEN C\_TRES\_CUENTAS\_MAYORES

FETCH C\_TRES\_CUENTAS\_MAYORES INTO @idcuenta, @numcuenta, @saldo

WHILE @@FETCH\_STATUS = 0

BEGIN

--INSERTAMOS LA CUENTA EN LA VARIABLE DE SALIDA

INSERT INTO @TABLA (CUENTA,SALDO\_ACTUAL)

VALUES (@numcuenta ,@saldo )

--INSERTAMOS LOS 3 ÚLTIMOS MOVIMIENTOS DE LA CUENTA

INSERT INTO @TABLA (CUENTA, SALDO\_ACTUAL, SALDO\_ANTERIOR, IMPORTE\_MOVIMIENTO, ULTIMA\_FECHA)

SELECT TOP 3 SALDO\_ACTUAL, SALDO\_ANTERIOR, IMPORTE\_MOVIMIENTO, ULTIMA\_FECHA

FROM movimiento

WHERE IDCUENTA = @idcuenta

ORDER BY ULTIMA\_FECHA DESC

--VAMOS A LA SIGUIENTE CUENTA

FETCH C\_TRES\_CUENTAS\_MAYORES INTO @idcuenta, @numcuenta, @saldo

END

CLOSE C\_TRES\_CUENTAS\_MAYORES

DEALLOCATE C\_TRES\_CUENTAS\_MAYORES

RETURN

END

GO

SELECT \* FROM F\_TRES\_CUENTAS\_MAYORES()

GO

1. **Crea un script para realizar un backup de todas las bases de datos del sql server (excepto de la master ,model, msdb, tempdb).**

**NOTA: en Master.dbo.sysdatabases se encuentra información de todas las bases de datos de sql server. Se podrían declarar las siguientes variables:**

**DECLARE @name VARCHAR(50) -- nombre de la bd**

**DECLARE @path VARCHAR(256) -- Ruta para los ficheros de backup**

**DECLARE @fileName VARCHAR(256) -- nombre del fichero para el backup**

**DECLARE @fileDate VARCHAR(20) – fecha para el nombre del fichero**

-- El nombre del fichero tendrá este Formato DB\_YYYYDDMM.BAK

DECLARE @name VARCHAR(50) -- Nombre de la Base de Datos

DECLARE @path VARCHAR(256) -- Ruta para las copias de seguridad

DECLARE @fileName VARCHAR(256) -- Nombre del Fichero

DECLARE @fileDate VARCHAR(20) -- Usado para el nombre del fichero

-- Ruta para las copias de seguridad

SET @path = 'C:\BaseDatos\BackupSQL\'

-- Formato del nombre del fichero

SELECT @fileDate = CONVERT(VARCHAR(20),GETDATE(),112)

DECLARE COPIA\_BD\_BACKUP CURSOR FOR

SELECT name

FROM master.dbo.sysdatabases

WHERE name NOT IN ('master','model','msdb','tempdb') -- excluir estas bases datos

OPEN COPIA\_BD\_BACKUP

FETCH NEXT FROM COPIA\_BD\_BACKUP INTO @name

WHILE @@FETCH\_STATUS = 0

BEGIN

SET @fileName = @path + @name + '\_' + @fileDate + '.BAK'

BACKUP DATABASE @name TO DISK = @fileName

FETCH NEXT FROM COPIA\_BD\_BACKUP INTO @name

END

CLOSE COPIA\_BD\_BACKUP

DEALLOCATE COPIA\_BD\_BACKUP