1. Crea una función IVA que incremente en un 16% el valor de la cantidad que se le pasa.

Utilízala para mostrar por pantalla los productos de la BD compras aplicado ese IVA.

USE compras;

GO

--Creamos la funcion conel nombre IVA

-- Indicamos el parámetro de entrada y tipo: @cantidad money

Create function dbo.IVA (@cantidad decimal(9,2))

Returns decimal(9,2) -- Indicamos el tipo de parámetros que retornará la función.

as

-- Encapsulamos el conjunto de funciones dentro de un Begin y un end.

Begin

Declare @resultado decimal(9,2)

set @resultado = @cantidad \* 1.16

-- Cuando terminamos la función ponemos el Return

Return (@resultado) -- Y la variable que devolverá la función @resultado.

End

select descrart,dbo.IVA(preunart) as 'con IVA', preunart from Articulos

1. Utilizando funciones lista la descripción del artículo, su precio, el precio medio(función a crear) y la diferencia con el precio medio(función a crear), para la tabla artículos en la bd de compras.

USE Compras;

GO

CREATE FUNCTION DBO.PRECIOPROMEDIO() RETURNS DECIMAL AS

BEGIN

RETURN (SELECT AVG(preunart) FROM Articulos)

END

GO

CREATE FUNCTION DBO.PRECIODIFER(@precio decimal) RETURNS DECIMAL AS

BEGIN

RETURN @precio-dbo.PRECIOPROMEDIO()

END

GO

SELECT descrart,preunart, dbo.PRECIOPROMEDIO() as promedio, dbo.PRECIODIFER(preunart) as diferencia FROM Articulos

--WHERE codigart ='0001'

1. Crea una función en la bd de jardinería que devuelva el listado de nombre del cliente , y el nombre\_contacto unido con una coma al apellido\_contacto en la tabla clientes.

USE Jardineria;

Go

CREATE FUNCTION dbo.fnLista() RETURNS TABLE

AS

RETURN(SELECT NombreCliente,NombreContacto + ',' + ApellidoContacto as contacto FROM Clientes)

GO

------------------------------------------------

SELECT \* FROM dbo.fnLista()

Liste sólo aquellos registros cuyos contactos empecen por A.

SELECT \* FROM dbo.fnLista() where contacto LIKE 'A%'

1. Modifica la función anterior tal que se le pueda pasar por qué letra/s debe empezar el nombre

USE Jardineria;

Go

CREATE FUNCTION dbo.fnLista\_PARAM(@empieza varchar(50)) RETURNS TABLE

AS

RETURN(SELECT NombreCliente,NombreContacto + ',' + ApellidoContacto as contacto FROM Clientes

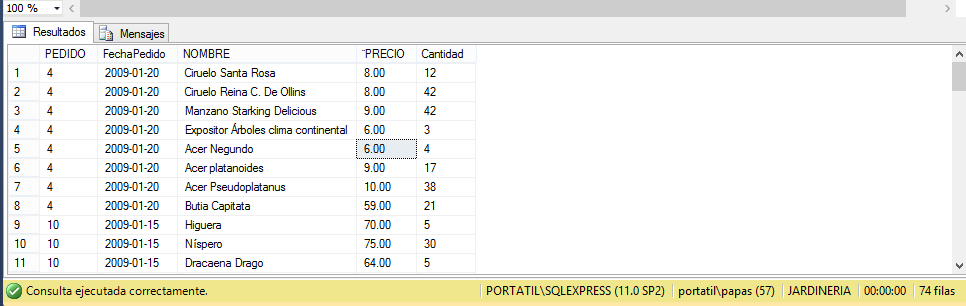
WHERE NombreContacto LIKE @empieza + '%')

GO

------------------------------------------------

SELECT \* FROM dbo.fnLista\_PARAM('A')

1. Definir sobre la BD Jardinería una función que liste los registros de los pedidos para un determinado año, incluyendo código del pedido, el nombre del producto, el precio que fue vendido y la cantidad vendida



CREATE FUNCTION DBO.PEDIDOSAÑO(@Y INT) RETURNS TABLE

AS

RETURN (SELECT PC.CodigoPedido AS 'PEDIDO', PC.FechaPedido, p.NOMBRE,

D.PRECIOUNIDAD AS '¨PRECIO',D.Cantidad

FROM dbo.Pedidos PC JOIN dbo.DetallePedidos D ON PC.CodigoPedido=D.CodigoPedido JOIN dbo.Productos P

ON D.CodigoProducto=P.CodigoProducto

WHERE YEAR(PC.FechaPedido) = @Y)

GO

select \* FROM dbo.pedidosaño(2009)

1. En la bd Jardinería implementar una función que devuelva el código de pedidos, el número de líneas de ese pedido y el montante total del pedido para un determinado cliente y en un determinado año.

USE Jardineria;

go

CREATE FUNCTION dbo.UDF\_PEDIDOSCLIENTE(@ID INT,@Y INT)

RETURNS @tabla TABLE(ID\_PEDIDO INT,

NUM\_lineas INT,

total DECIMAL)

AS

BEGIN

INSERT INTO @tabla

SELECT PEDIDOS.CodigoPedido, COUNT(\*) as num\_de\_lineas, SUM(detallePedidos.Cantidad\*DetallePedidos.PrecioUnidad) as total

FROM pedidos JOIN DetallePedidos ON pedidos.CodigoPedido=DetallePedidos.CodigoPedido

WHERE Pedidos.CodigoCliente =@ID AND YEAR(PEDIDOS.FechaPedido) = @Y

group by PEDIDOS.CodigoPedido

RETURN

END

GO

----------------------------

SELECT \* FROM dbo.UDF\_PEDIDOSCLIENTE(1,2009)

//USO LA TABLA CREADA POR MI PORQUE NO ES UN SOLO SELECT SIN I VARIOS SELECT QUE NO SE PUEDEN RESOLVER EN UNA SOLA CONSULTA.

1. Sobre la base de datos banco crea una función que incremente el saldo de la cuenta al multiplicarlo por un valor que se dé.

USE banco;

Go

CREATE FUNCTION dbo.UDF\_IncrementaSaldo

(

@NumCuenta int,

@VALOR\_MULT DECIMAL(10,2)

)

RETURNS DECIMAL(10,2)

AS

BEGIN

DECLARE @Saldo int,

@saldo\_final DECIMAL(10,2)

SELECT @Saldo = saldo

FROM cuenta

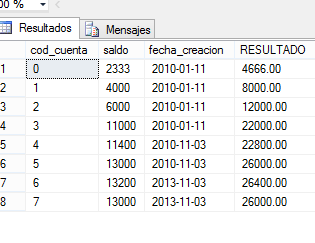
WHERE cod\_cuenta= @NumCuenta

SET @saldo\_final = @Saldo \* @VALOR\_MULT

RETURN @saldo\_final

END

Usa la función creada para listar como quedaría para cada cuenta su saldo si se le aplicase un incremento de 2 ptos. Ejemplo de uso de la función en una sentencia de T-sql



use banco;

GO

SELECT cod\_cuenta,

saldo,

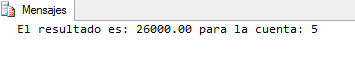
fecha\_creacion,

-- Ejecucion de la funcion:

dbo.UDF\_IncrementaSaldo( cod\_cuenta, 2) AS RESULTADO

FROM cuenta

Usa la función creada en un script tal que declarando variables, para un código de cuenta determinado, imprima por pantalla como quedaría el saldo. El incremento=2.



use banco;

go

DECLARE @NumCuenta int,

@Resultado DECIMAL(10,2)

SET @NumCuenta = 5

SET @Resultado = dbo.UDF\_IncrementaSaldo(@NumCuenta, 2)

PRINT 'El resultado es: ' + cast(@Resultado as varchar) + ' para la cuenta: ' + cast(@NumCuenta as varchar)

1. Sobre la BD banco crear una función que me devuelva todos los movimientos realizados en una determinada cuenta.

use banco;

go

CREATE FUNCTION dbo.udf\_MovimientosCuenta

(@NumCuenta int

)

RETURNS TABLE

AS

RETURN

(SELECT fecha, cantidad, dni, movimiento.cod\_cuenta as cuenta, id\_movimiento FROM dbo.movimiento

INNER JOIN CUENTA ON movimiento.cod\_cuenta = cuenta.cod\_cuenta

and cuenta.cod\_cuenta = @NumCuenta)

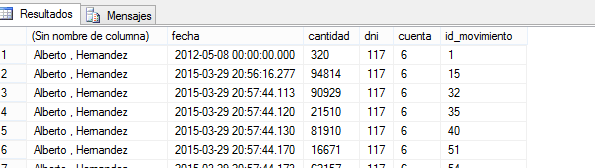
Go

Usa la función anterior para ver los movimientos de la cuenta con código 6

---------------------------------------

SELECT \* from dbo.udf\_MovimientosCuenta(6)

Realiza una consulta tal que, utilizando la función anterior, muestres al lado del movimiento el nombre del propietario de la cuenta con código 6



SELECT cliente.nombre + ' , ' + cliente.apellido1, f.\* from cliente

join tiene on cliente.dni=tiene.dni join dbo.udf\_MovimientosCuenta(6) as f

on tiene.cod\_cuenta=f.cuenta

1. Sobre la bd nba crear una función, tal que cuando se le pase el nombre de una división saque todos los datos de todos los jugadores pertenecientes a equipos que jueguen en la conferencia a la que pertenece esa división.

use nba;

go

create function dbo.udf\_jugadores(@division varchar(9))

returns @t\_jugadores table

(t\_codigo int,

t\_nombre varchar(30),

t\_procedencia varchar(20),

t\_altura varchar(4),

t\_peso int,

t\_posicion varchar(5),

t\_nom\_equipo varchar(20))

as

begin

DECLARE @VConferencia varchar (4)

select @VConferencia = Conferencia from equipos

where Division=@division

INSERT INTO @t\_jugadores select jugadores.\* FROM jugadores join equipos

on jugadores.Nombre\_equipo = equipos.Nombre and equipos.Conferencia=@VConferencia

return

end

---------------------------------------------------------------------

SELECT \* from dbo.udf\_jugadores('Central')

1. Crear sobre la bd prueba una función que devuelva el valor de la hipotenusa de un triángulo a partir de los valores de sus lados.

USE pruebas;

GO

CREATE FUNCTION dbo.hipotenusa (@lado1 int, @lado2 int)

RETURNS FLOAT as

BEGIN

DECLARE @hipot float, @clado1 int,@clado2 int

set @clado1=POWER(@lado1,2)

set @clado2=POWER(@lado2,2)

set @hipot=SQRT(@clado1+@clado2)

return @hipot

end

Go

-----------------------------

select dbo.hipotenusa(2,2)

1. Crear sobre la bd prueba una función que devuelva 1 o 0 si un número es divisible por otro. Utiliza la función y que se vea escrito por pantalla si son divisibles o no.

use pruebas;

go

create function dbo.divisible(@num int, @den int) returns bit

as

begin

declare @esdiv bit, @modulo int

set @modulo=@num%@den

IF (@modulo=0)

set @esdiv=1

else

set @esdiv=0

return @esdiv

end

Go

---------------------

IF (dbo.divisible(6,6) = 1)

PRINT 'Es divisible'

else

PRINT 'No es divisible'