select count(\*) as cantidadTitles from titles

select count(\*) as cantidadAutores from authors

select count(\*) as cantidad from titleauthor

select t.title, a.au\_id, a.au\_fname

from titles as t inner join titleauthor as ta

on t.title\_id = ta.title\_id

inner join authors as a

on ta.au\_id = a.au\_id

go

select t.title, count(\*) as cantidadAutores

from titles as t inner join

titleauthor as ta

on t.title\_id = ta.title\_id

group by t.title

having count(\*) >=3

go

select t.title, count(\*) as cantidadAutores

from titles as t inner join

titleauthor as ta

on t.title\_id = ta.title\_id

group by t.title

go

select a.au\_lname

from titles as t inner join

titleauthor as ta

on t.title\_id = ta.title\_id

inner join authors as a

on ta.au\_id = a.au\_id

where t.title ='Sushi, Anyone?'

go

-- ejemplo de funcion escalar

create function fn\_sumar(@num1 int, @num2 int)

returns int

begin

declare @sum int

set @sum = @num1 + @num2

return @sum

end

go

sp\_helptext fn\_sumar

go

create function fn\_TotalOrden(@numeroOrden int)

returns money

with encryption

begin

declare @total money

select @total = sum(od.Quantity \* od.UnitPrice)

from [Order Details] as od –No es necesario los [] solo que Order Details tiene espacio

where od.OrderID = @numeroOrden

return @total

end

go

select o.OrderID, o.OrderDate, dbo.fn\_TotalOrden(o.OrderID) as total, o.Shipcity

from Orders as o

select dbo.fn\_sumar(13,16)

select \*from Orders

select \*from [Order Details]

create function fn\_Empleados(@tipo char(11))

returns @fn\_Empleados TABLE

(

CodigoEmp int primary key not null,

nombreEmp varchar(61) not null

)

BEGIN

if @tipo = 'NombreCorto'

insert @fn\_Empleados select e.EmployeeID ,e.LastName

from Employees as e

else

if @tipo = 'NombreLargo'

insert @fn\_Empleados select e.EmployeeID ,e.LastName + ' ' + e.FirstName

from Employees as e

return

end

go

select \*

from dbo.fn\_Empleados('NombreLargo')

create function fn\_clientexciudad(@ciudad varchar(15))

returns table

as return(

select c.CustomerID, c.CompanyName, c.ContactName

from Customers as c

where c.City = @ciudad

)

go

select \*

from dbo.fn\_clientexciudad('Berlin')

go

deleted // tabla temporal propia de desencadenadores

-- Descencadenador para Ventas

create trigger des\_OrderDetalle\_Insercion

ON [Order Details]

FOR INSERT

AS

UPDATE P

SET P.UnitsInStock = (P.UnitsInStock - I.Quantity)

FROM Products AS P INNER JOIN inserted AS I

ON P.ProductID = I.ProductID

GO

-- Buscando un Producto

select P.ProductID,P.ProductName,P.UnitPrice,P.UnitsInStock

FROM Products AS P

WHERE P.ProductID = 2

-- 2 CHang 19.00 17

-- Insertando una orden de venta

INSERT INTO Orders(EmployeeID,ShipVia,Freight)

VALUES (1,1,3)

GO

-- resultado orden = 11078

-- Registrar en la orden 11078

-- El producto 2 en una cantidad de 10

Insert into [Order Details]

values(11078,2,19.0,10,0)

go

SELECT \* FROM Orders WHERE OrderID = 11078

SELECT \* FROM [Order Details] where OrderID = 11078

-- declaracion de cursor

declare c\_productos cursor

for select p.ProductID, P.ProductName, P.UnitPrice

from Products as P

-- apertura del cusror

open c\_productos

-- declaracion de variable spara extraccion de datos

declare @codigo int

declare @nombre nvarchar(40)

declare @precio money

fetch next from c\_productos into @codigo, @nombre, @precio

while(@@FETCH\_STATUS <> -1)

begin

if(@@FETCH\_STATUS <> -2)

begin

print 'codigo: ' + cast(@codigo as nvarchar(10)) + 'nombre: ' + @nombre

end

--acceder al sigueinte registro del crusor

fetch next from c\_productos into @codigo, @nombre, @precio

end

go

--cerramos el cusor

close c\_productos

--descargar la memoria

deallocate c\_productos

go

create procedure pa\_PedidosAtrasados

as

begin

select \*FROM dbo.Orders

WHERE RequiredDate < GETDATE()

AND ShippedDate IS NUll

end

go

-- EJECUTANDO EL PROCESO ALMACENADO

execute pa\_PedidosAtrasados

go

create proc pa\_EmployeeCustomer

as

select upper(substring(LastName,1,4)+ substring(FirstName, 1,1)), 'Northind traders',

rtrim (firstName)+' ' +LastName, 'Employee', Address, City,Region,PostalCode, country,('(206) 555-1234' +' x'+Extension), NULL

FROM Employees

Where HireDate < GETDATE()

go

exec pa\_EmployeeCustomer

go

alter proc pa\_EmployeeCustomer

as

select upper(substring(LastName,1,3)+ substring(FirstName, 1,2)), 'Northind traders',

rtrim (firstName)+' ' +LastName, 'Employee', Address, City,Region,PostalCode, country,('(206) 555-1234' +' x'+Extension), NULL

FROM Employees

Where HireDate < GETDATE()

go

--Insertando nuevos clientes usando en PROC, almacenado

insert into Customers

exec pa\_EmployeeCustomer

go

select \* from Customers

--Procedimientos almacenados con parametros de entrada

Create procedure pa\_ventasxAños

@FechaIni Date,

@FechaFin Date

as

begin

if @FechaIni is null or @FechaFin is null

begin

raiserror('Los valores no pueden ser nulos', 14,1)

return

end

select O.ShippedDate, O.OrderID, OS.Subtotal,

datename(yy,ShippedDate) as year

from orders O

inner join [Order Subtotals] OS

ON O.OrderID = OS.OrderID

where O.ShippedDate between @FechaIni and @FechaFin

end

go

exec pa\_ventasxAños '1996-07-10', '1996-07-16'

go