**CREAR UNA FUNCIÓN PARA CADA UNA DE LAS SIGUIENTES**:

1. Calcular y mostrar la propina (10%) del total facturado por cada mozo, mostrando: Nombre, Apellido, Total Facturado, Valor Propina

**CREATE Function Ejercicio\_1(@Porcentaje float)**

**Returns Table**

**AS**

**Return**

**(select Mozos.nombre,Mozos.apellido,sum(Facturas.total)as Total,sum(total\*@Porcentaje)as Propina**

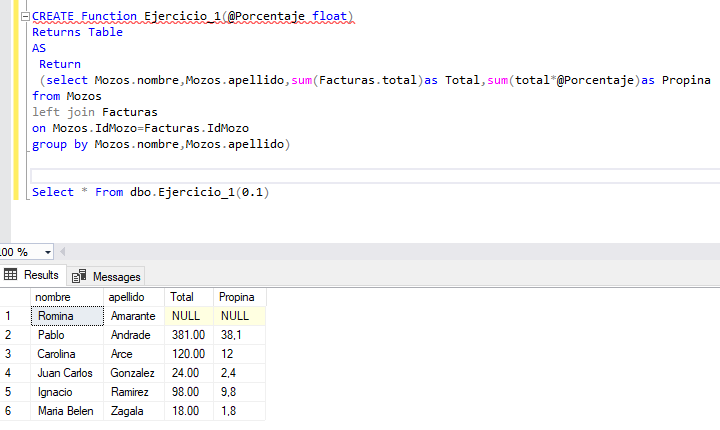
**from Mozos**

**left join Facturas**

**on Mozos.IdMozo=Facturas.IdMozo**

**group by Mozos.nombre,Mozos.apellido)**

**Select \* From dbo.Ejercicio\_1(0.1)**



2. Mostrar ordenado el Top X (ejemplo top 5) de los platos más vendidos por el restaurant: El resultado debe ser la Lista de Platos

**CREATE Function Ejercicio\_2 (@Filtro int)**

**returns Table**

**AS**

**return**

**(select top (@Filtro) Platos.descripcion,count(\*)as cantVentas**

**from Platos**

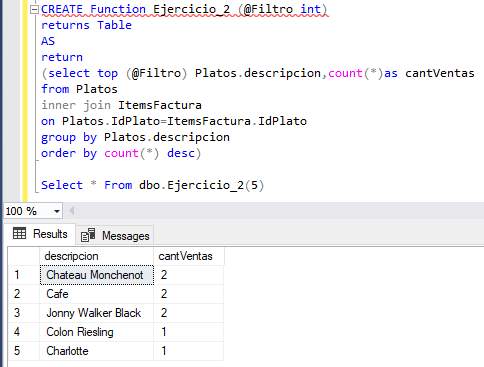
**inner join ItemsFactura**

**on Platos.IdPlato=ItemsFactura.IdPlato**

**group by Platos.descripcion**

**order by count(\*) desc)**

**Select \* From dbo.Ejercicio\_2(5)**



3. Mostrar ordenadas las X (Ejemplo 10) mesas más utilizadas en el restaurant: El resultado debe ser la Lista de Mesas

**create Function Ejercicio\_3 (@Filtro int)**

**returns table**

**as return(**

**select p.descripcion**

**from**

**(select top (@Filtro) Mesas.IdMesa, Mesas.Descripcion,count(\*) as cantUsos**

**from Mesas**

**left join Facturas**

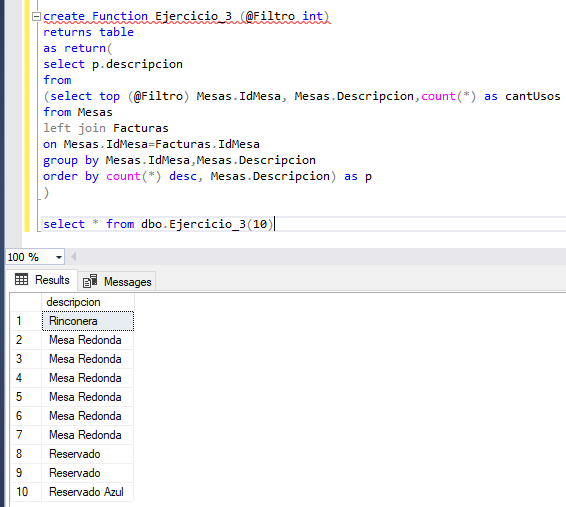
**on Mesas.IdMesa=Facturas.IdMesa**

**group by Mesas.IdMesa,Mesas.Descripcion**

**order by count(\*) desc, Mesas.Descripcion) as p**

**)**

**select \* from dbo.Ejercicio\_3(10)**



4. Mostrar cual es el ingrediente más utilizado en los platos. Mostrar: Su precio, punto de reposición y la cantidad de platos donde se utiliza

**create function Ejercicio\_4()**

**returns table**

**as return(**

**select top 1 Ingredientes.IdIngrediente,Ingredientes.descripcion,Ingredientes.precComp,Ingredientes.puntoRep,count(\*) as CantPlatos**

**from Ingredientes**

**inner join PlaIng**

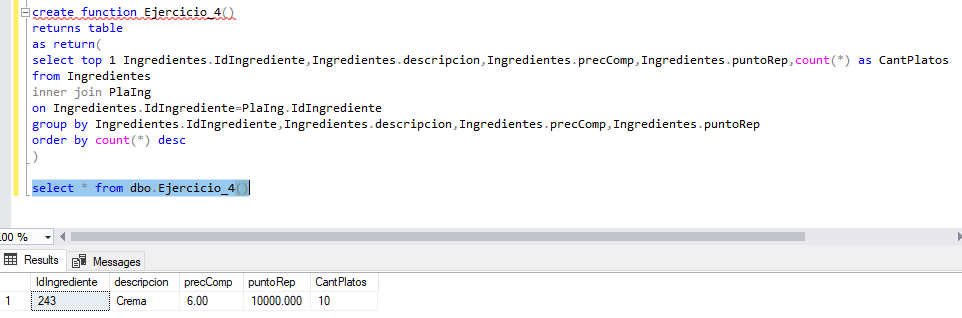
**on Ingredientes.IdIngrediente=PlaIng.IdIngrediente**

**group by Ingredientes.IdIngrediente,Ingredientes.descripcion,Ingredientes.precComp,Ingredientes.puntoRep**

**order by count(\*) desc**

**)**

**select \* from dbo.Ejercicio\_4()**



5. Mostrar por cada ingrediente su proveedor con los siguientes datos: Nombre del Ingrediente, Nombre del Proveedor, Dirección, Teléfono

**create function Ejercicio\_5()**

**returns table**

**as return**

**(**

**select distinct p.descripcion,p.razsoc,p.Direccion,p.Telefono**

**from (**

**select Ingredientes.descripcion, Proveedores.IdProveedor, Proveedores.razsoc,**

**concat(TelefonoProveedor.codArea,TelefonoProveedor.central,TelefonoProveedor.numero) as Telefono,**

**concat(Proveedores.calle,Proveedores.numero,' ',Proveedores.piso,' ',Proveedores.departamento) as Direccion**

**from Proveedores**

**inner join TelefonoProveedor**

**on Proveedores.IdProveedor=TelefonoProveedor.IdProveedor**

**inner join ProIng**

**on Proveedores.IdProveedor = ProIng.IdProveedor**

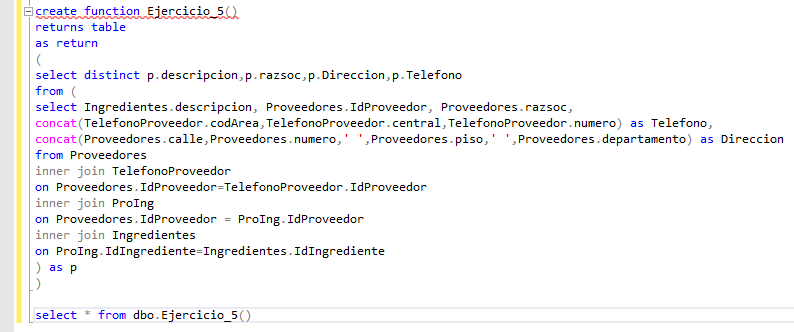
**inner join Ingredientes**

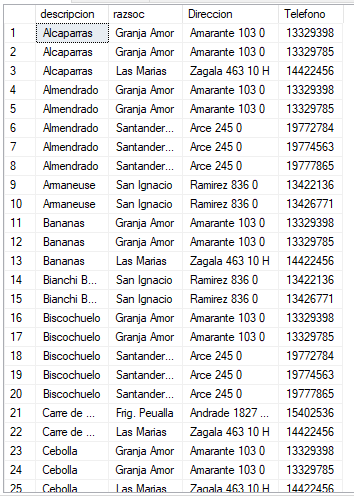
**on ProIng.IdIngrediente=Ingredientes.IdIngrediente**

**) as p**

**)**

**select \* from dbo.Ejercicio\_5()**





6. Mostrar el inventario actual de Ingredientes y muestre un campo de bandera / Flag/ alerta para indicar cuando un ingrediente tiene un stock menor al punto de reposición

**CREATE FUNCTION Flag (**

**@stock decimal(8,3),**

**@puntoRep decimal(8,3)**

**)**

**RETURNS int**

**AS**

**BEGIN**

**declare @flag int**

**if(@stock<=@puntoRep)**

**set @flag = 0**

**else**

**set @flag = 1**

**return @flag**

**END**

**create function Ejercicio\_6()**

**returns table**

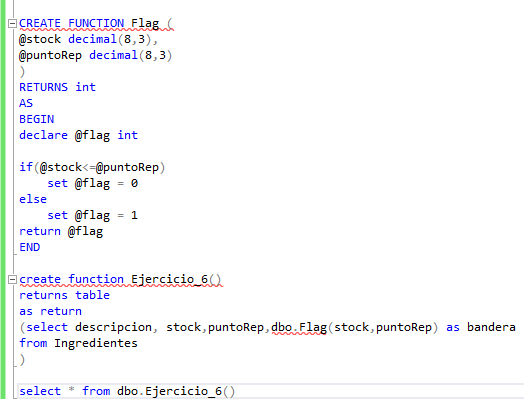
**as return**

**(select descripcion, stock,puntoRep,dbo.Flag(stock,puntoRep) as bandera**

**from Ingredientes**

**)**

**select \* from dbo.Ejercicio\_6()**





7. Mostrar el piso y sector al que pertenece cada factura (Se puede consultar por piso). El restaurant solo posee un piso que es ‘PB’

CREATE Function facturasxpiso (@piso VARCHAR(12))

RETURNS TABLE

AS

RETURN

(SELECT 'PB' as 'Piso', NroFactura AS Factura,Sectores.descripcion AS Sector

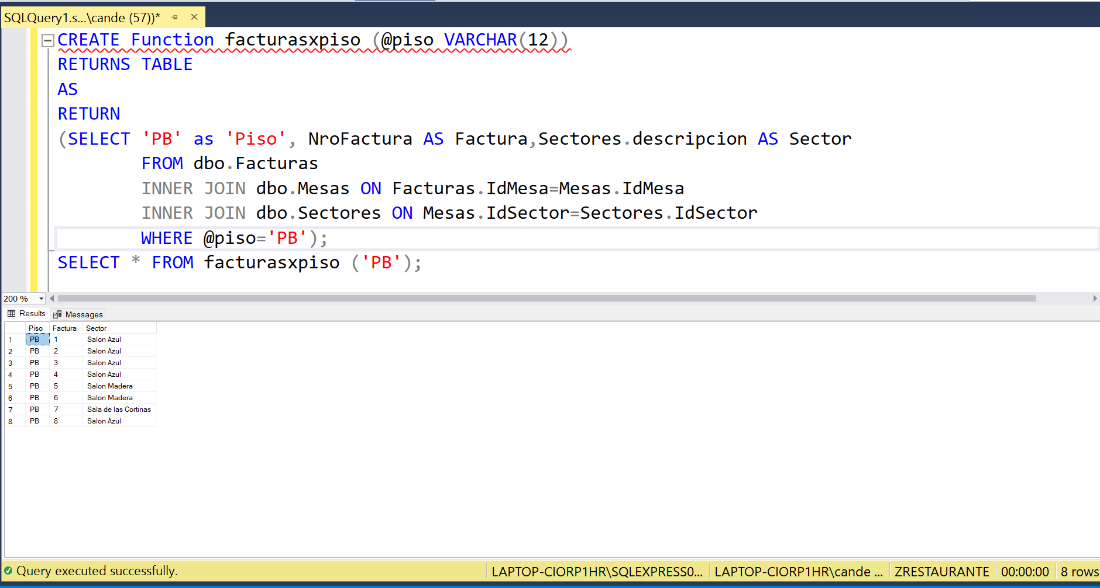
FROM dbo.Facturas

INNER JOIN dbo.Mesas ON Facturas.IdMesa=Mesas.IdMesa

INNER JOIN dbo.Sectores ON Mesas.IdSector=Sectores.IdSector

WHERE @piso='PB');

SELECT \* FROM facturasxpiso ('PB');



8. Muestre el monto facturado por el restaurant hasta este momento (Cierre de caja), considere utilizar solo la tabla Facturas.

CREATE FUNCTION CierreDeCaja ()

RETURNS float

AS

BEGIN

DECLARE @suma float

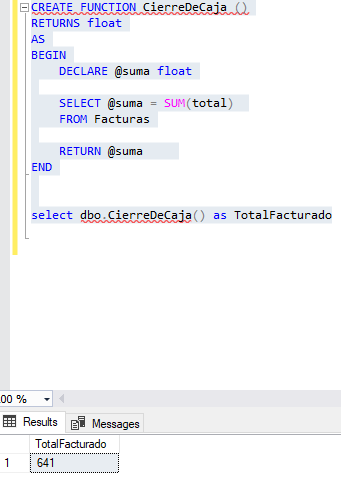
SELECT @suma = SUM(total)

FROM Facturas

RETURN @suma

END

select dbo.CierreDeCaja() as TotalFacturado



9. Muestre la lista de platos y el costo de cada plato según sus ingredientes

CREATE Function costoxplato ()

RETURNS TABLE

AS

RETURN

(SELECT Platos.descripcion AS Plato, SUM(Ingredientes.precComp) AS Costo

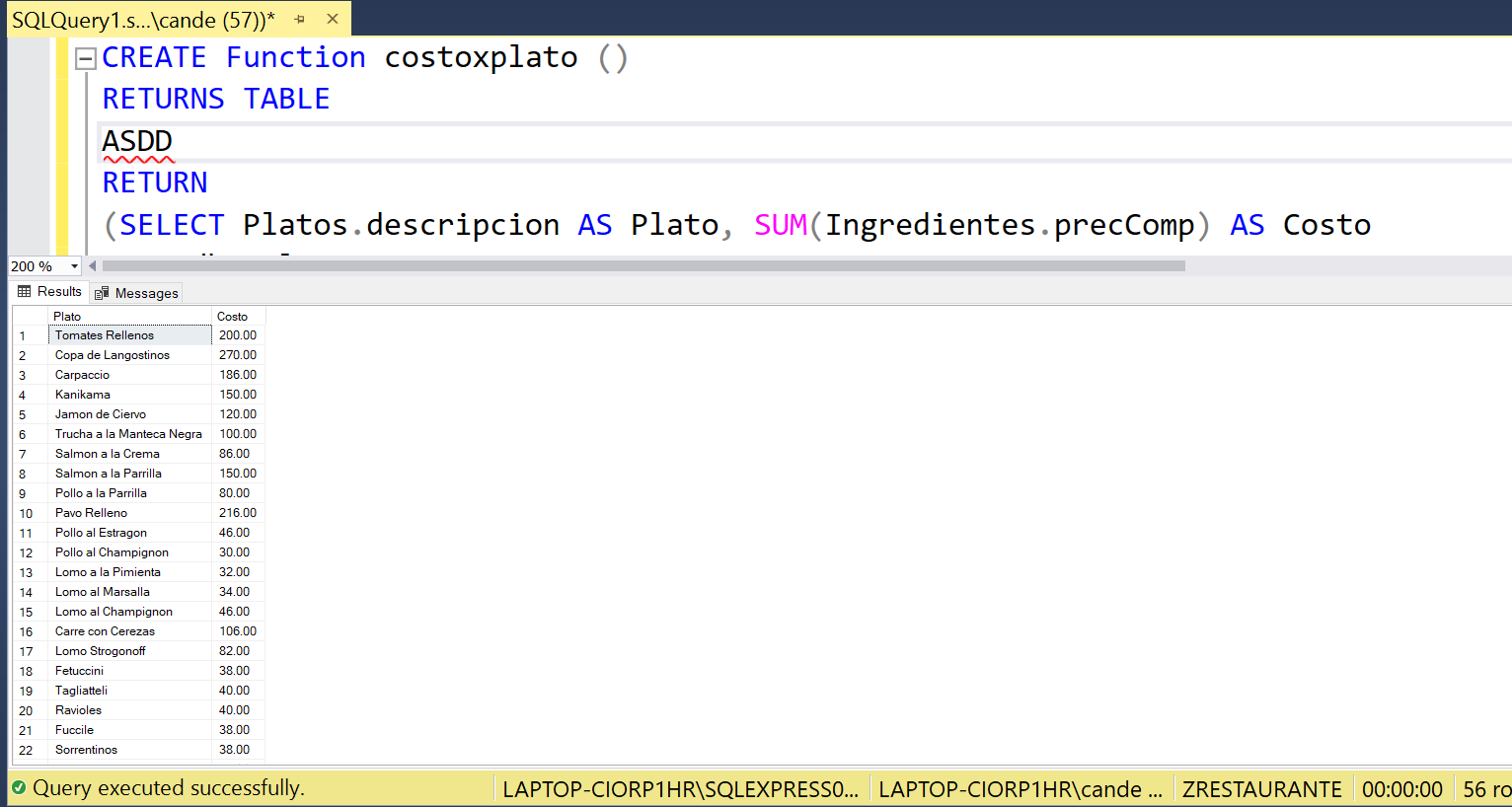
FROM dbo.Platos

INNER JOIN dbo.PlaIng ON Platos.IdPlato=PlaIng.IdPlato

INNER JOIN dbo.Ingredientes ON PlaIng.IdIngrediente=Ingredientes.IdIngrediente

GROUP BY PlaIng.IdPlato,Platos.descripcion);

SELECT \* FROM costoxplato ();



10. Muestre la cantidad de proveedores actuales

CREATE Function ProveedoresActuales()

RETURNS INT

AS

BEGIN

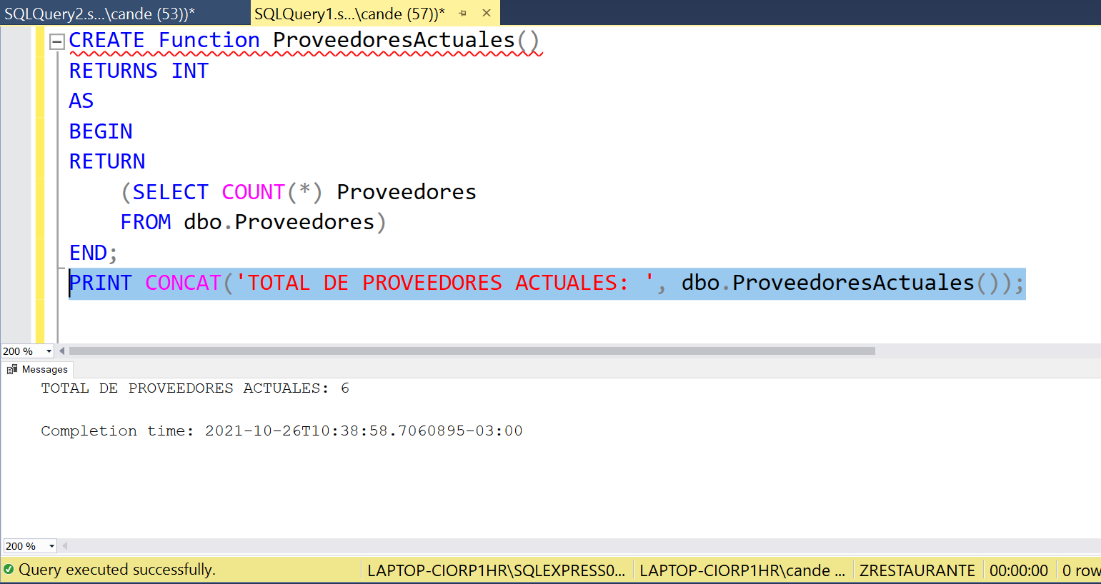
RETURN

(SELECT COUNT(\*) Proveedores

FROM dbo.Proveedores)

END;

PRINT CONCAT('TOTAL DE PROVEEDORES ACTUALES: ', dbo.ProveedoresActuales());



**CREAR UN PROCEDIMIENTO (SP) PARA CADA UNA DE LAS SIGUIENTES:**

11. Cree un procedimiento para registrar un nuevo mozo

CREATE PROCEDURE alta\_mozo

@idMozo int,

@nombre varchar(20),

@apellido varchar(20),

@calle varchar(40) = '',

@numero int = null,

@piso int = null,

@departamento varchar(4) ='',

@idCodPos int,

@localidad varchar(40),

@IdTelefono int,

@codArea int = null,

@central int = null,

@numeroTel int = null,

@celular int = null

AS

BEGIN

IF NOT EXISTS(SELECT \* FROM Mozos where IdMozo = @IdMozo)

BEGIN

IF @idCodPos IS NOT NULL AND NOT EXISTS (SELECT \* FROM Localidades WHERE IdCodpos=@idCodPos)

BEGIN

INSERT INTO dbo.Localidades (IdCodpos,localida)

VALUES (@idCodPos,@localidad);

PRINT 'Se agregó una nueva Localidad correctamente'

END;

IF @idTelefono IS NOT NULL AND NOT EXISTS (SELECT \* FROM TelefonoMozo WHERE IdTelefono=@idTelefono)

BEGIN

INSERT INTO dbo.TelefonoMozo (IdTelefono,IdMozo,codArea,central,numero,celular)

VALUES (@idTelefono,@idMozo,@codArea,@central,@numeroTel,@celular);

PRINT 'Se agregó un nuevo Telefono correctamente'

END;

INSERT INTO Mozos (IdMozo, nombre, apellido, calle, numero, piso, departamento, IdCodPos)

VALUES(@IdMozo, @nombre, @apellido, @calle, @numero, @piso, @departamento, @IdCodPos)

PRINT 'El mozo se agregó correctamente'

RETURN

END;

ELSE

BEGIN

PRINT 'Ya existe el mozo'

RETURN

END

END

EXEC alta\_mozo 19,'Juan','Fernandez','Lebensohn',null,null,'',7,'Bernal',13,011,1234,54782471,41572147;



12. Cree un procedimiento para crear una factura con la cantidad de personas que desee y fechas del día de ayer

create procedure altaFactura

@NroFactura int,

@IdMesa int,

@IdMozo int,

@cantidadPersonas int,

@total int,

@IdSector int,

@Descripcion varchar,

@fechaApertura datetime,

@horaApertuta datetime,

@fechaCierre datetime,

@horaCierre datetime,

@nombre varchar,

@apellido varchar,

@calle varchar,

@numero int,

@piso int,

@departamento varchar,

@IdCodPos int,

as

begin

if not exists(select \* from Facturas where NroFactura = @NroFactura)

begin

if not exists(select \* from Mesas where IdMesa = @IdMesa)

begin

if not exists(select \* from Sectores where IdSector = @IdSector)

begin

insert into Sectores(IdSector, Descripcion)

values(@IdSector, @Descripcion)

end

insert into Mesas(IdMesa, Descripcion, IdSector)

values(@IdMEsa, @Descripcion, @IdSector)

end

if not exists(select \* from Mozos where IdMozo = @IdMozo)

begin

insert into Mozos(IdMozo, nombre, apellido, calle, numero, piso, departamento, IdCodPos)

values(@Idmozo, @nombre, @apellido, @calle, @numero, @piso, @departamento, @IdCodPos)

end

insert into Facturas(NroFactura, IdMesa, IdMozo, cantidadPersonas, fechaApertura, horaApertuta, fechaCierre, horaCierre)

values(@NroFactura, @IdMesa, @IdMozo, @cantidadPersonas, @total, @fechaApertura, @horaApertuta, @fechaCierre, @horaCierre)

print 'se agregó la factura'

return

else

begin

print 'ese nro de factura ya fue emitida y existe'

return

end

end