Año de la lucha contra la corrupción y la impunidad”

“Universidad Nacional De La Amazonia Peruana”

**Facultad De Ingeniería De Sistemas E Informática**

* **Docente:** Ing. José Edgar Garcia Diaz
* **Integrantes:**
* Neyer Gutierrez Torres



* Paul Alexander García Pilco
* Pedro Alexander Nashnato Curitima
* Jheyson Estivar Alayo Valencia

* **Asignatura:** Base de datos II
* **Nivel:** 2
* **Ciclo:** 4

**IQUITOS-PERÚ**

**2019**

índice

[**1.** **5 Ejemplos con Procesos almacenados utilizando la base de datos Northwind** 3](#_Toc27620774)

[**2. 5 ejemplos de funciones con la base de datos Northwind** 6](#_Toc27620775)

[**3. 5 ejemplos de Triggers con la base de datos Northwind** 8](#_Toc27620776)

[**4. 5 ejemplos de Cursores con la base de datos Northwind** 11](#_Toc27620777)

1. **5 Ejemplos con Procesos almacenados utilizando la base de datos Northwind**
2. Crear un procedimiento almacenado que muestre los productos que se encuentra en un rango de precios.

create proc rango\_precios

@mini float,

@maxi float as

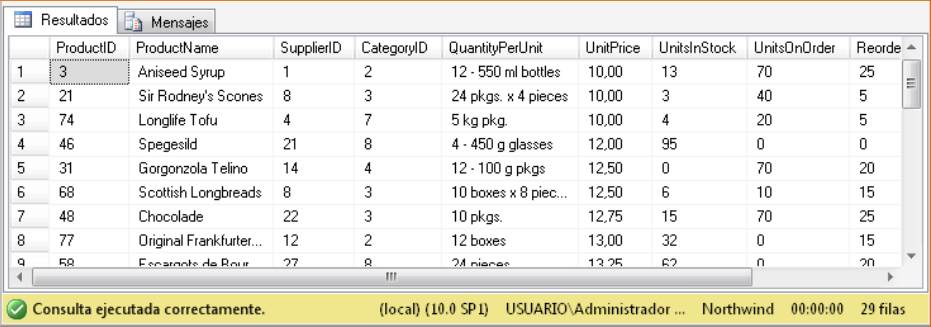
select \*

from Products as p

where p.UnitPrice between @mini and @maxi

order by UnitPrice

Podemos ejecutar estableciendo una consulta en la cual defina un precio mínimo y un máximo.



1. Los productos más vendidos por año según el número de registros que se quiere mostrar y mostrar en monto vendido.

create proc monto\_vendido

@anio int, @num int as select top (@num) p.ProductName, SUM( Quantity\*od.UnitPrice) as monto

from Products as p

inner join [Order Details] as od on od.ProductID=p.ProductID

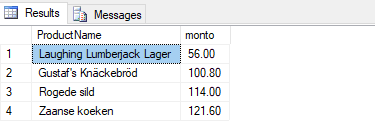
inner join Orders as o on o.OrderID=od.OrderID

where year(OrderDate)=@anio

group by p.ProductName

order by monto

Ejecutamos el procedimiento almacenado en una consulta estableciendo un año y el número de registro que queremos saber.



1. Crear un procedimiento almacenado para actualizar los precios de los productos solo a los productos de la categoría ingresada (ingresar el porcentaje de aumentos de los precios ) solamente se actualizaran los productos continuos.

create proc actu\_p

@c int,

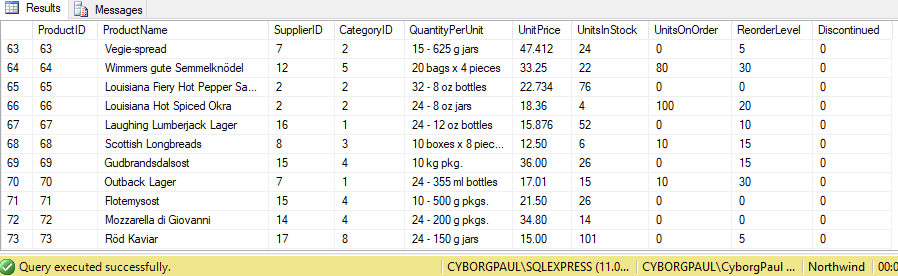
@incr float

as --incio del cuerpo del procedimiento

update Products

set UnitPrice=UnitPrice+UnitPrice\*@incr/100

where Discontinued=0 and CategoryID=@c



1. Procedimiento para el listado de productos de una determinada categoría.

Create procedure spProductosListadoPorCategoria

(

@CategoriaCodigo int

)

As

select P.ProductID, P.ProductName,

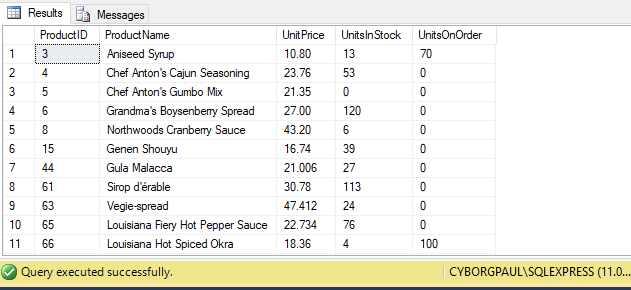
P.UnitPrice , P.UnitsInStock, P.UnitsOnOrder

from Products As P

where CategoryID = @CategoriaCodigo

go

Se establece la ejecución del procedimiento con una consulta ingresando la categoría que se quiere mostrar.



1. Modificar el procedimiento de Listado de Productos por categoría que lo muestre ordenados por precio descendente.

Alter procedure spProductosListadoPorCategoria

(

@CategoriaCodigo int

)

As

select P.ProductID, P.ProductName,

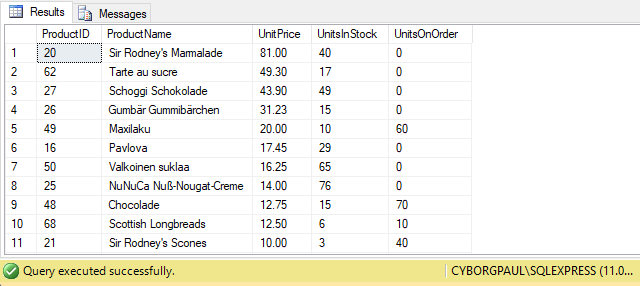
P.UnitPrice , P.UnitsInStock, P.UnitsOnOrder

from Products As P

where CategoryID = @CategoriaCodigo

order by P.UnitPrice desc

go



# **2. 5 ejemplos de funciones con la base de datos Northwind**

1. Crear una función que te muestre el estado del producto (en caso no haya stock escribirá agotado, en caso haya se escribirá vigente)

create function estado\_producto()

returns table

as

return

select ProductName producto, (case when UnitsInStock = 0

then 'Agotado' else 'vigente' end) estado from Products

2. Crear una función que devuelva la cantidad de pedidos que se realizo en un determinado producto en un determinado año.

create function cantidad\_pedidos (@IDproducto as int, @año as int)

returns table

as

return

select pr.ProductName,count(\*) cantidad

from (select p.ProductName , count(o.OrderID) cantidad from Orders o

inner join [Order Details] od on o.OrderID=od.OrderID

inner join Products p on od.ProductID=p.ProductID

group by p.ProductName,o.OrderDate,o.OrderID, p.ProductID

having p.ProductID = @IDproducto and datepart(yyyy,o.OrderDate) = @año) as tabla

inner join Products pr on tabla.ProductName=pr.ProductName

group by pr.ProductName

3. Crear una función que devuelva el monto de pedidos por producto que se realizaron en un determinado mes.

create function producto\_mes(@mes int)

returns table

as

return

select p.ProductName producto,sum(od.UnitPrice\*od.Quantity) monto,

OrderDate fecha from [Order Details] od

inner join Orders o on o.OrderID=od.OrderID

inner join Products p on p.ProductID=od.ProductID

group by p.ProductName, o.OrderDate

having datepart(MM,o.OrderDate) = @mes

4. Crear una función que devuelva las ventas de producto que realizo un determinado empleado y a que cliente se lo proporciono.

create function empleado(@IDempleado int)

returns table

as

return

select e.LastName empleado, c.CompanyName cliente,

p.ProductName producto from Employees e

inner join Orders o on o.EmployeeID=e.EmployeeID

inner join Customers c on o.CustomerID=c.CustomerID

inner join [Order Details] od on od.OrderID=o.OrderID

inner join Products p on od.ProductID=p.ProductID

group by e.EmployeeID,e.LastName,CompanyName,ProductName

having e.EmployeeID=@IDempleado

5. Crear una función que devuelva las ventas realizadas en el trimestre que indique el usuario, mostrar producto, compañía, empleado, fecha.

create function ventas\_trimestre(@trimestre int)

returns table

as

return

select p.ProductName producto, e.LastName empleado,

c.CompanyName compañia, o.OrderDate from Orders o

inner join [Order Details] od on o.OrderID=od.OrderID

inner join Products p on p.ProductID=od.ProductID

inner join Employees e on e.EmployeeID=o.EmployeeID

inner join Customers c on c.CustomerID=o.CustomerID

where DATEPART(QUARTER,OrderDate) = @trimestre

# **3. 5 ejemplos de Triggers con la base de datos Northwind**

1. crear un trigger que solo permita agregar un producto si su precio es mayor a 100

create trigger TX\_ELIMINAR\_CATEGORIA

on products

for insert

as

if(select UnitPrice from inserted)<100

begin

ROLLBACK TRANSACTION

print 'el producto debe tener un precio mayor a 100'

end

else

print 'el producto se inserto correctamente '

1. crear un trigger que solo permita ingresar empleados mayores a 50 años

create trigger tx\_insertar\_empleado

on employees

for insert

as

if(select (year(GETDATE())-year(BirthDate)) from inserted)<50

begin

print 'la edad del empleado debe ser mayor a 50'

ROLLBACK TRANSACTION

end

else

print 'el empleado se inertó correctamente'

1. crear un trigger que solo permita ingresar una orden en el año 1997

create trigger tx\_insertar\_orden

on orders

for insert

as

if(select year(OrderDate) from inserted)=1997

begin

ROLLBACK TRANSACTION

print 'orden agregada'

end

else

print 'solo se puede agregar una orden del año 1997'

insert into Orders(OrderDate) values('19-08-1997')

1. Solo para proveedores de usa

create trigger \_tx\_ingresar\_proveedor3

on suppliers

for insert

as

if(select Country from inserted)!='USA'

begin

rollback transaction

print 'proveedor solo puede ser de usa'

end

1. Para no actualizar el pais del cliente

create trigger tx\_eliminar\_cliente

on customers

for update

as

if update (country)

begin

print 'no se puede actualizar el pais del cliente'

ROLLBACK TRANSACTION

End

# **4. 5 ejemplos de Cursores con la base de datos Northwind**

Cursores

Ejercicios:

1. Dejar las contraseñas iguales al nombre de usuario. La tabla Cliente tiene estos tres campos: CliCod, CliUser, CliPass

-- declaramos las variables  
declare @cod as int  
declare @user as varchar(50)  
declare @pass as varchar(50)

-- declaramos un cursor llamado "CURSORITO".  
El select debe contener sólo los campos a utilizar.

declare CURSORITO cursor for  
select CliCod, CliUser, CliPass from Cliente  
open CURSORITO

-- Avanzamos un registro y cargamos en las variables los valores encontrados en el primer registro

fetch next from CURSORITO  
into @cod, @user, @pass  
    while @@fetch\_status = 0  
        begin  
        update Cliente set CliPass= @user where CliCod=@cod

 -- Avanzamos otro registro  
        fetch next from CURSORITO  
        into @cod, @user, @pass  
        end

-- cerramos el cursor  
close CURSORITO  
deallocate CURSORITO

1. Reportar la lista de productos, si las unidades en orden son mayores al stock, mostrar COMPRAR URGENTE, de lo contrario mostrar STOCK ADECUADO

Declare cursorProductoComprasUrgente cursor for  
SELECT P.ProductID, P.ProductName,   P.UnitsInStock,  P.UnitsOnOrder    from Products As P  
Open cursorProductoComprasUrgente  
Declare @Codigo int, @Descripcion nvarchar(40), @Stock Numeric(9,2), @PorAtender Numeric(9,2)  
Fetch cursorProductoComprasUrgente into @Codigo, @Descripcion, @Stock, @PorAtender  
Print ‘================ LISTADO ======================’  
While (@@FETCH\_STATUS = 0)

Begin  
               Declare @Mensaje nvarchar(20)  
             If (@PorAtender > @Stock)  
            Begin  
                    Set @Mensaje = ‘COMPRAR URGENTE’  
              End  
          Else  
            Begin  
                   Set @Mensaje = ‘STOCK ADECUADO’  
           End

-- reportar el registro y luego leer el siguiente

Print ‘Código: ‘ + STR(@Codigo) Print ‘Descripción: ‘ + @Descripcion + ‘ Stock: +  Ltrim(STR(@Stock)) + ‘ Por Atender:+ Ltrim(Str(@PorAtender))

Print ‘Mensaje: ‘ + @Mensaje  
Print » Print ‘————————————————‘  
Fetch cursorProductoComprasUrgente into @Codigo,@Descripcion, @Stock, @PorAtender

End  
Close cursorProductoComprasUrgente  
Deallocate cursorProductoComprasUrgente  
go

1. Listado de las categorías y sus productos

Declare cursorCategoriasListadoProductos cursor for  
select C.CategoryID, C.CategoryName from Categories As C

Open cursorCategoriasListadoProductos

Declare @CodigoCategoria int, @NombreCategoria nvarchar(15)  
Fetch cursorCategoriasListadoProductos into @CodigoCategoria, @NombreCategoria

Print ‘============================================================’  
Print ‘============ Listado de Productos por categoria ==========’  
Print ‘============================================================’  
Print »

While (@@FETCH\_STATUS = 0) -- Recorre el cursor de Categorías

Begin

Print ‘Código Categoria: ‘ + Ltrim(Str(@CodigoCategoria)) + Space(2) +  
‘Categoría: ‘ + @NombreCategoria  
— Definir el Cursor para los productos – CURSOR ANIDADO  
Declare cursorProductosCategoria cursor for  
select P.ProductID, P.ProductName, P.UnitPrice from Products As P  
where P.CategoryID = @CodigoCategoria  
Open cursorProductosCategoria  
Declare @CodigoProducto int, @NombreProducto nvarchar(40),  
@Precio Numeric(9,2)  
Fetch cursorProductosCategoria into @CodigoProducto, @NombreProducto, @Precio  
Print »  
Print ‘====================== PRODUCTOS ==========================’  
Print ‘ CODIGO DESCRIPCION PRECIO ‘

While (@@FETCH\_STATUS = 0) -- Estructura para Productos

Begin

Print Space(5) + Ltrim(str(@CodigoProducto)) + Space(10) + @NombreProducto + space(20) + Ltrim(str(@Precio))  
Fetch cursorProductosCategoria into @CodigoProducto, @NombreProducto, @Precio

End --Final While de Productos

Print »  
Close cursorProductosCategoria  
Deallocate cursorProductosCategoria

— Leer la siguiente categoria  
Fetch cursorCategoriasListadoProductos into @CodigoCategoria, @NombreCategoria

End --Final While de Categorias

Close cursorCategoriasListadoProductos  
Deallocate cursorCategoriasListadoProductos  
go

1. Recorrer la tabla Production.ProductDescription de la base de datos AdventureWorks y mostrar por pantalla el valor del registro Description luego que el mismo ha sido copiado a la variable @Description.

DECLARE @Description AS nvarchar(400)

DECLARE ProdInfo CURSOR FOR SELECT [Description] FROM Production.ProductDescription

OPEN ProdInfo

FETCH NEXT FROM ProdInfo INTO @Description

WHILE @@fetch\_status = 0

BEGIN

PRINT @Description

FETCH NEXT FROM ProdInfo INTO @Description

END

CLOSE ProdInfo

DEALLOCATE ProdInfo

1. Actualizar las unidades de existencia de los productos asignándoles el valor de 1000 de aquellos con stock cero.

DECLARE @ID INT, @NOMBRE VARCHAR(50)

WHILE EXISTS(SELECT \* FROM PRODUCTS WHERE UNITSINSTOCK=0)

BEGIN

SELECT TOP 1 @ID=PRODUCTID, @NOMBRE=PRODUCTNAME FROM PRODUCTS WHERE UNITSINSTOCK=0

UPDATE PRODUCTS

SET UNITSINSTOCK=1000

WHERE PRODUCTID=@ID

PRINT 'PRODUCTO:'+@NOMBRE + ' SE ACTUALIZO EL

STOCK'

CONTINUE

END