USE [AdventureWorks]

--Laboratorio: AdventureWorks - Joins

--1. Mostrar los empleados que también son vendedores.

--Tablas: HumanResources.Employee, Sales.SalesPerson

--Campos: BusinessEntityID

select e.\* from HumanResources.Employee e join Sales.SalesPerson sp

on e.BusinessEntityID = sp.BusinessEntityID

order by e.BusinessEntityID

select BusinessEntityID from HumanResources.Employee

intersect

select BusinessEntityID from Sales.SalesPerson

order by BusinessEntityID

--2. Mostrar los empleados ordenados alfabéticamente por

--apellido y por nombre.

--Tablas: HumanResources.Employee, Person.Person

--Campos: BusinessEntityID, LastName, FirstName

select e.BusinessEntityID, LastName, FirstName

from HumanResources.Employee e join Person.Person p

on e.BusinessEntityID = p.BusinessEntityID

order by LastName, FirstName

--3. Mostrar el código de logueo, código de territorio y sueldo

--básico de los vendedores.

--Tablas: HumanResources.Employee, Sales.SalesPerson

--Campos: LoginID, TerritoryID, Bonus, BusinessEntityID

select LoginID, TerritoryID, Bonus, e.BusinessEntityID

from HumanResources.Employee e join Sales.SalesPerson sp

on e.BusinessEntityID = sp.BusinessEntityID

order by e.BusinessEntityID

--4. Mostrar los productos que sean ruedas.

--Tablas: Production.Product, Production.ProductSubcategory

--Campos: Name, ProductSubcategoryID

select p.Name ProductName, p.ProductSubcategoryID SubCategoryProduct, ps.Name SubCategoryProductName

from Production.Product p join Production.ProductSubcategory ps

on ps.ProductSubcategoryID = p.ProductSubcategoryID

where ps.Name = 'wheels'

--where ps.ProductSubcategoryID = 17

--5. Mostrar los nombres de los productos que no son bicicletas.

--Tablas:Production.Product, Production.ProductSubcategory

--Campos: Name, ProductSubcategoryID

select p.Name ProductName, p.ProductSubcategoryID SubCategoryProduct, ps.Name SubCategoryProductName

from Production.Product p join Production.ProductSubcategory ps

on ps.ProductSubcategoryID = p.ProductSubcategoryID

where ps.Name not like '%bike%'

--6. Mostrar los precios de venta de aquellos productos donde

--el precio de venta sea inferior al precio de lista recomendado

--para ese producto ordenados por nombre de producto.

--Tablas: Sales.SalesOrderDetail, Production.Product

--Campos: ProductID, Name, ListPrice, UnitPrice

select p.ProductID, p.Name, p.ListPrice [Precio de lista], sod.UnitPrice [Precio unidad]

from Sales.SalesOrderDetail sod join Production.Product p

on sod.ProductID = p.ProductID

where sod.UnitPrice < p.ListPrice

-- and sod.UnitPrice < p.ListPrice

order by p.Name

--7. Mostrar todos los productos que tengan igual precio. Se

--deben mostrar de a pares, código y nombre de cada uno de

--los dos productos y el precio de ambos. Ordenar por precio

--en forma descendente.

--Tablas:Production.Product

--Campos: ProductID, ListPrice, Name

select p1.ProductID [ProductID1], p1.ListPrice [ListPrice1], p1.Name [ProductName1],

p2.ProductID [ProductID2], p2.ListPrice [ListPrice2], p2.Name [ProductName2]

from Production.Product p1 join Production.Product p2

on p1.ListPrice = p2.ListPrice

where p1.ListPrice > 0 and p1.ProductID <> p2.ProductID

order by p1.ListPrice desc

--8. Mostrar el nombre de los productos y de los proveedores

--cuya subcategoría es 15 ordenados por nombre de

--proveedor.

--Tablas: Production.Product, Purchasing.ProductVendor,

--Purchasing.Vendor

--Campos: Name ,ProductID, BusinessEntityID,

--ProductSubcategoryID

select p.Name [ProductName], v.Name [BusinessEntityName], p.ProductID, v.BusinessEntityID, p.ProductSubcategoryID

from Production.Product p join Purchasing.ProductVendor pv

on p.ProductID = pv.ProductID join Purchasing.Vendor v

on v.BusinessEntityID = pv.BusinessEntityID

where p.ProductSubcategoryID = 15

order by v.Name

--select \* from Purchasing.ProductVendor

--select \* from Purchasing.Vendor

--select \* from Production.Product

--9. Mostrar todas las personas (nombre y apellido) y en el caso

--que sean empleados mostrar también el login id, sino mostrar

--null.

--Tablas: Person.Person, HumanResources.Employee

--Campos: FirstName, LastName, LoginID, BusinessEntityID

select FirstName, LastName, LoginID, p.BusinessEntityID

from Person.Person p left join HumanResources.Employee e

on p.BusinessEntityID = e.BusinessEntityID

--Tablas temporales locales--

--Forma de crear tabla e insertarle datos

--Eliminamos tabla si existe

drop table if exists #empleados

--Creamos tabla con sus respectivos campos

create table #empleados(

legajo int primary key,

nombre varchar(50),

apellido varchar(50)

)

--Selecciamos tabla para comprobar que se haya creado

select \* from #empleados

--Insertamos datos desde otra tabla

insert #empleados

select BusinessEntityID, FirstName, LastName

from Person.Person

--Volvemos a seleccionar para comprobar carga de datos

select \* from #empleados

--

--Otra forma de crear tabla

--Primero elimino la tabla si existe

drop table if exists #empleados

--Creamos la tabla ya con los nombres de las

--columnas predeterminadas de la tabla de origen

--a menos que les genere un alias

select BusinessEntityID, FirstName [Nombre], LastName [Apellido]

into #empleados

from Person.Person

--Le agrego la clave primaria

--(puede agregarse directamente debajo del from)

alter table #empleados add primary key (BusinessEntityID)

--Compruebo la carga de datos

select \* from #empleados

--

--Tablas temporales globales

--A la global puedo acceder desde otra ventana, sesión.

--Su ciclo de vida termina cuando se cierra la sesión que la creó

select BusinessEntityID, FirstName [Nombre], LastName [Apellido]

into ##empleados

from Person.Person

alter table ##empleados add primary key (BusinessEntityID)

select \* from ##empleados

--

--CTE's

--Ciclo de vida acotado a la consulta que la va a utilizar

with cte\_TemporalEmpleados(codigo, nombre, apellido)

as

(

select BusinessEntityID, FirstName, LastName

from Person.Person

)

select \* from cte\_TemporalEmpleados

--Podria crear una tabla temporal en base a esa consulta anterior

with cte\_TemporalEmpleados(codigo, nombre, apellido)

as

(

select BusinessEntityID, FirstName, LastName

from Person.Person

)

select \* into #empleados\_A from cte\_TemporalEmpleados

where LEFT (apellido, 1) = 'A' -- order by codigo

select \* from #empleados\_A

--Laboratorio: AdventureWorks -

--Tablas temporales - CTE

--1. Clonar estructura y datos de los campos nombre,

--color y precio de lista de la tabla Production.Product

--en una tabla llamada #Productos.

--Tablas: Production.Product

--Campos: Name, ListPrice, Color

select Name, Color, ListPrice

into #Productos

from Production.Product

select \* from #Productos

--2. Clonar solo estructura de los campos identificador,

--nombre y apellido de la tabla Person.Person en una

--tabla llamada #Personas

--Tablas: Person.Person

--Campos: BusinessEntityID, FirstName, LastName

select BusinessEntityID, FirstName, LastName

into #Personas

from Person.Person

where 1 = 2 -- Valor falso para solo crear la estrucutura

select \* from #Personas

--3. Eliminar si existe la tabla #Productos

--Tablas: #Productos

drop table if exists #Productos

--4. Eliminar si existe la tabla #Personas

--Tablas: #personas

drop table if exists #Personas

--5. Crear una CTE con las órdenes de venta

--Tablas: Sales.SalesOrderHeader

--Campos: SalesPersonID, SalesOrderID,

--OrderDate

--with cte\_SalesOrder (Vendedor, Orden, Fecha)

--as (

--select SalesPersonID, SalesOrderID, OrderDate

--from Sales.SalesOrderHeader

--)

--select \* from cte\_SalesOrder

--select \* into #SalesOrderTemp from cte\_SalesOrder

--select \* from #SalesOrderTemp

--drop table if exists #SalesOrderTemp

--así se hizo en el video

with cte\_SalesOrder (Vendedor, Orden, Fecha)

as (

select SalesPersonID, SalesOrderID, OrderDate

from Sales.SalesOrderHeader

where SalesPersonID is not null

)

select Vendedor, YEAR(Fecha) [Año], COUNT(\*) [Cantidad de Ventas]

from cte\_SalesOrder

group by Vendedor, YEAR(Fecha)

order by Vendedor, [Año]