

Guía N° 4

Tema: Consultas de selección

I. Objetivos

1. Mostrar datos almacenados en una base de datos
2. Ejecutar sentencias básicas SELECT
3. Implementar diferentes clausulas con la sentencia SELECT para seleccionar datos tomando ciertas decisiones

II. Introducción Teórica

Lenguaje de manipulación de datos

Lenguaje de cierta complejidad que permite el manejo y procesamiento del contenido de la base de datos.

Sentencias DML:

- **Select**
- **Insert**
- **Update**
- **Delete**

SELECT. Permite recuperar datos de una o varias tablas. Esta sentencia es de la más compleja y potente de las sentencias SQL.

Sintaxis:

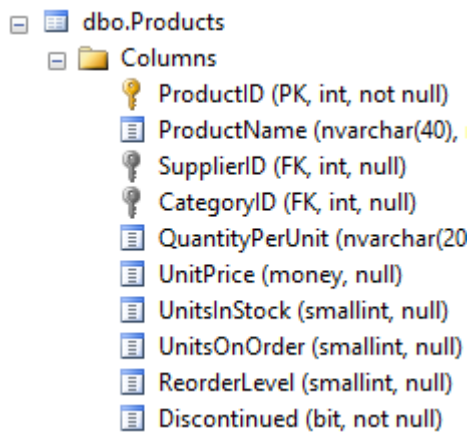
```
SELECT lista_de_campos  
[ FROM nombre_tabla ]  
[ WHERE condicion_individual]  
[ GROUP BY campos_a_agrupar ]  
[ HAVING condicion_grupo ]  
[ ORDER BY campo_a_ordenar [ ASC | DESC ] ]
```

Nota: debe tomar en cuenta el orden de las clausulas o palabras reservadas mostradas en la sintaxis anterior, las instrucciones GROUP BY y HAVING se estudiarán en la Guía 8

FROM

Se utiliza en conjunto con la **cláusula FROM** con la cual se indica en qué tabla o tablas se tiene que buscar la información.

Ejemplo 1:



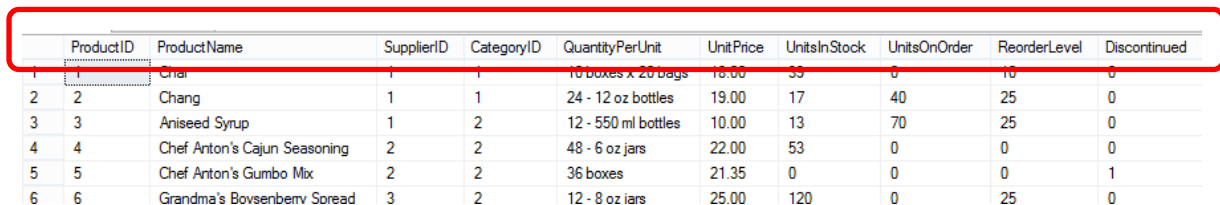
The screenshot shows the 'Columns' folder for the 'dbo.Products' table. The columns listed are: ProductID (PK, int, not null), ProductName (nvarchar(40), not null), SupplierID (FK, int, null), CategoryID (FK, int, null), QuantityPerUnit (nvarchar(20), null), UnitPrice (money, null), UnitsInStock (smallint, null), UnitsOnOrder (smallint, null), ReorderLevel (smallint, null), and Discontinued (bit, not null).

Column	DataType	Constraints
ProductID	int	PK, not null
ProductName	nvarchar(40)	not null
SupplierID	int	FK, null
CategoryID	int	FK, null
QuantityPerUnit	nvarchar(20)	null
UnitPrice	money	null
UnitsInStock	smallint	null
UnitsOnOrder	smallint	null
ReorderLevel	smallint	null
Discontinued	bit	not null

-- El asterisco significa que se quiere seleccionar todos los campos de la
-- tabla Products

```
SELECT * FROM Products
```

Resultado:



The screenshot shows the results of the query 'SELECT * FROM Products'. The table has 10 columns: ProductID, ProductName, SupplierID, CategoryID, QuantityPerUnit, UnitPrice, UnitsInStock, UnitsOnOrder, ReorderLevel, and Discontinued. The first 6 rows are visible.

	ProductID	ProductName	SupplierID	CategoryID	QuantityPerUnit	UnitPrice	UnitsInStock	UnitsOnOrder	ReorderLevel	Discontinued
1	1	Chai	1	1	10 boxes x 20 bags	18.00	53	0	10	0
2	2	Chang	1	1	24 - 12 oz bottles	19.00	17	40	25	0
3	3	Aniseed Syrup	1	2	12 - 550 ml bottles	10.00	13	70	25	0
4	4	Chef Anton's Cajun Seasoning	2	2	48 - 6 oz jars	22.00	53	0	0	0
5	5	Chef Anton's Gumbo Mix	2	2	36 boxes	21.35	0	0	0	1
6	6	Grandma's Boysenberry Spread	3	2	12 - 8 oz jars	25.00	120	0	25	0

Ejemplo 2:

-- Para seleccionar ciertos campos hay que colocarlos como en un listado
-- en este ejemplo solo se mostraran los registros almacenados en los campos ProductID
-- ProductName y UnitPrice

```
SELECT ProductID, ProductName, UnitPrice FROM Products
```

Resultado:



The screenshot shows the results of the query 'SELECT ProductID, ProductName, UnitPrice FROM Products'. The table has 4 columns: ProductID, ProductName, and UnitPrice. The first 8 rows are visible.

	ProductID	ProductName	UnitPrice
1	1	Chai	18.00
2	2	Chang	19.00
3	3	Aniseed Syrup	10.00
4	4	Chef Anton's Cajun Seasoning	22.00
5	5	Chef Anton's Gumbo Mix	21.35
6	6	Grandma's Boysenberry Spread	25.00
7	7	Uncle Bob's Organic Dried Pears	30.00
8	8	Northwoods Cranberry Sauce	40.00

WHERE

La **cláusula WHERE** permite seleccionar únicamente las filas que cumplan con una condición de selección especificada. Sólo se mostrarán las filas para las cuales la evaluación de la condición sea verdadera (TRUE).

Los campos con valores NULL no se incluirán en las filas de resultado.

La condición de la **cláusula WHERE** puede ser cualquier condición válida o combinación de condiciones utilizando operadores lógicos (NOT, AND, OR) y relacionales (=, <, >, <=, >=).

Ejemplo 1:

```
-- Seleccionar los datos de la tabla Products donde el dato almacenado en el campo
-- UnitPrice sea mayor a 15
SELECT ProductID, ProductName, UnitPrice
FROM Products
WHERE UnitPrice > 15
```

Resultado:

	ProductID	ProductName	UnitPrice
1	1	Chai	18.00
2	2	Chang	19.00
3	4	Chef Anton's Cajun Seasoning	22.00
4	5	Chef Anton's Gumbo Mix	21.35
5	6	Grandma's Boysenberry Spread	25.00
6	7	Uncle Bob's Organic Dried Pears	30.00
7	8	Northwoods Cranberry Sauce	40.00
8	9	Mishi Kobe Niku	97.00

Ejemplo 2:

```
-- Seleccionar los datos de la tabla Products donde el dato almacenado en el campo
-- UnitPrice sea mayor o igual a 15 y menor o igual a 50
SELECT ProductID, ProductName, UnitPrice
FROM Products
WHERE UnitPrice >= 15 AND UnitPrice <= 50
```

Resultado:

	ProductID	ProductName	UnitPrice
1	1	Chai	18.00
2	2	Chang	19.00
3	4	Chef Anton's Cajun Seasoning	22.00
4	5	Chef Anton's Gumbo Mix	21.35
5	6	Grandma's Boysenberry Spread	25.00
6	7	Uncle Bob's Organic Dried Pears	30.00
7	8	Northwoods Cranberry Sauce	40.00
8	10	Ikura	31.00

Ejemplo 3:

```
--Otra forma de crear la consulta anterior es utilizando la instruccion BETWEEN
SELECT ProductID, ProductName, UnitPrice
FROM Products
WHERE UnitPrice BETWEEN 15 AND 50
```

Ejemplo 4:

```
--Haciendo uso del operador NOT obtenemos los registros de la tabla Products donde
-- el dato almacenado en el campo UnitPrice sea menor que 15
SELECT ProductID, ProductName, UnitPrice
FROM Products
WHERE NOT UnitPrice > 15
```

Resultado:

	ProductID	ProductName	UnitPrice
1	3	Aniseed Syrup	10.00
2	13	Konbu	6.00
3	19	Teatime Chocolate Biscuits	9.20
4	21	Sir Rodney's Scones	10.00
5	23	Tunnbröd	9.00
6	24	Guaraná Fantástica	4.50
7	25	NuNuCa Nuß-Nougat-Creme	14.00
8	31	Gorgonzola Telino	12.50
9	33	Geitost	2.50

Ejemplo 5

```
-- Seleccionar los registros de la tabla Products donde el dato almacenado en el campo
--ProductID sea mayor a 50 y el dato almacenado en el campo UnitPrice sea menor a 10
SELECT ProductID, ProductName, UnitPrice
FROM Products
WHERE ProductID > 15 OR UnitPrice < 10
```

Resultado:

	ProductID	ProductName	UnitPrice
1	13	Konbu	6.00
2	16	Pavlova	17.45
3	17	Alice Mutton	39.00
4	18	Camarvon Tigers	62.50
5	19	Teatime Chocolate Biscuits	9.20
6	20	Sir Rodney's Marmalade	81.00
7	21	Sir Rodney's Scones	10.00
8	22	Gustaf's Knäckebröd	21.00
9	23	Tunnbröd	9.00

Del resultado tomamos el registro 1 y vemos que el dato de ProductID es igual 13 y no cumple la condición, pero el dato almacenado en el campo UnitPrice si es menor que 10 y se cumple la condición, y como se está utilizando el operador OR con uno que sea verdadero es suficiente para que el registro se muestre como un resultado de la consulta SELECT

LIKE

Puede utilizarse la cláusula LIKE para formar patrones de comparación con caracteres comodín.

Los comodines más utilizados son:

- `_`: Cualquier carácter.
- `%`: Cualquier cadena de cero o más caracteres
- `[]`: Cualquier carácter individual del intervalo ([a-z])

Puede utilizarse únicamente LIKE o NOT LIKE, indicando igual o no igual respectivamente

Ejemplos:

Ejemplo 1:

```
-- Seleccionar los campos EmployeeID y LastName de la tabla Employees donde el dato almacenado
-- en el campo LastName comience con la letra D
SELECT EmployeeID, LastName FROM Employees
WHERE LastName LIKE 'D%'
```

Resultado:

	EmployeeID	LastName
1	1	Davolio
2	9	Dodsworth

Ejemplo 2:

```
-- Seleccionar los campos EmployeeID y LastName de la tabla Employees donde el dato almacenado
-- en el campo LastName termine con la letra N
SELECT EmployeeID, LastName FROM Employees
WHERE LastName LIKE '%N'
```

Resultado:

	EmployeeID	LastName
1	5	Buchanan
2	8	Callahan

Ejemplo 3:

```
-- Seleccionar los campos EmployeeID, LastName y Title de la tabla Employees donde el dato
-- almacenado en el campo Title se encuentre la palabra SALES, no importando en que posición
SELECT EmployeeID, LastName, Title FROM Employees
WHERE Title LIKE '%SALES%'
```

Resultado:

	EmployeeID	LastName	Title
1	1	Davolio	Sales Representative
2	2	Fuller	Vice President, Sales
3	3	Leverling	Sales Representative
4	4	Peacock	Sales Representative
5	5	Buchanan	Sales Manager
6	6	Suyama	Sales Representative
7	7	King	Sales Representative
8	8	Callahan	Inside Sales Coordinator
9	9	Dodsworth	Sales Representative

Ejemplo 4:

```
--Seleccionar los campos EmployeeID y LastName de empleados EXCEPTO aquellos donde el dato almacenado
--en LastName comience con la letra D
SELECT EmployeeID, LastName FROM Employees
WHERE LastName NOT LIKE 'D%'
```

Resultado:

	EmployeeID	LastName
1	5	Buchanan
2	8	Callahan
3	2	Fuller
4	7	King
5	3	Leverling
6	4	Peacock
7	6	Suyama

Ejemplo 5:

```
--Seleccionar los campos EmployeeID y LastName de empleados EXCEPTO aquellos donde el dato almacenado
--en Title posea en cualquier posición la palabra REPRESENTATIVE
SELECT EmployeeID, LastName, Title FROM Employees
WHERE Title NOT LIKE '%REPRESENTATIVE%'
```

Resultado:

	EmployeeID	LastName	Title
1	2	Fuller	Vice President, Sales
2	5	Buchanan	Sales Manager
3	8	Callahan	Inside Sales Coordinator

Ejemplo 6:

```
-- Seleccionar todas las ordenes de pedido donde el dato almacenado en el campo OrderID termine con los
-- dígitos 0248
SELECT OrderID FROM [Order Details]
WHERE OrderID LIKE '_0248'
```

Resultado:

	OrderID
1	10248
2	10248
3	10248

Ejemplo 7:

```
-- Seleccionar todas las ordenes de pedido donde el dato almacenado en el campo OrderID comience con 10
-- en el tercer dígito contenga un número entre 1 y 5 y termine con los dígitos 48
SELECT OrderID FROM [Order Details]
WHERE OrderID LIKE '10[1-5]48'
```

Resultado:

	OrderID
1	10348
2	10248
3	10348
4	10448
5	10548
6	10448
7	10548
8	10248
9	10248

ORDER BY

Se puede hacer uso de la cláusula ORDER BY para mostrar los datos de forma ordenada.

Si se utiliza en conjunto la cláusula ASC, los registros se mostrarán en orden ascendente (de menor a mayor) según el campo(s) especificado(s) en la cláusula ORDER BY. Si se utiliza la cláusula DESC, los registros serán mostrados en orden descendente (de mayor a menor).

Ejemplo 1:

```
-- Ordenar de forma ascendente los registros almacenados en los campos ProductID, ProductName
-- y UnitPrice de la tabla Products, se ordenaran por medio del campo ProductID
SELECT ProductID, ProductName, UnitPrice
FROM Products
ORDER BY ProductID ASC
--De forma predeterminada los datos se ordenan de forma ascendente, por lo tanto la instrucción
--ASC opcional
```

Resultado:

	ProductID	ProductName	UnitPrice
1	1	Chai	18.00
2	2	Chang	19.00
3	3	Aniseed Syrup	10.00
4	4	Chef Anton's Cajun Seasoning	22.00
5	5	Chef Anton's Gumbo Mix	21.35
6	6	Grandma's Boysenberry Spread	25.00
7	7	Uncle Bob's Organic Dried Pears	30.00

Ejemplo 2:

```
-- Ordenar de forma descendente los registros almacenados en los campos ProductID, ProductName
-- y UnitPrice de la tabla Products, se ordenaran por medio del campo ProductID
SELECT ProductID, ProductName, UnitPrice
FROM Products
ORDER BY ProductID DESC
```

Resultado:

	ProductID	ProductName	UnitPrice
1	77	Original Frankfurter grüne Soße	13.00
2	76	Lakkalikööri	18.00
3	75	Rhönbräu Klosterbier	7.75
4	74	Longlife Tofu	10.00
5	73	Röd Kaviar	15.00
6	72	Mozzarella di Giovanni	34.80
7	71	Flotemysost	21.50

DISTINCT

La cláusula DISTINCT especifica que los registros con ciertos datos duplicados sean ignorados en el resultado.

Ejemplo 1:

```
-- Seleccionar todos los registros no repetidos almacenados en el campo OrderID
-- de la tabla Order Details
SELECT DISTINCT OrderID FROM [Order Details]
```

Resultado:

	OrderID
1	10248
2	10249
3	10250
4	10251
5	10252

TOP N

TOP n, especifica que solo se mostrará el primer conjunto de filas del resultado de la consulta.

El conjunto de filas puede ser un número o un porcentaje de las filas (TOP n PERCENT)

TOP n WITH TIES: Esta cláusula permite incluir en la selección, todos los registros que tengan el mismo valor del campo por el que se ordena

Ejemplo 1:

```
--Mostrar los primeros cinco registros de la tabla Order Details
SELECT TOP 5 OrderID, ProductID, Quantity
FROM [Order Details]
```

Resultado:

	OrderID	ProductID	Quantity
1	10248	11	12
2	10248	42	10
3	10248	72	5
4	10249	14	9
5	10249	51	40

Ejemplo 2:

```
--Mostrar los primeros dos registros de la tabla Order Details, pero si existen uno o mas registros
--con el mismo dato almacenado en OrderID se mostraran tambien en el resultado de la consulta
SELECT TOP 2 WITH TIES OrderID, ProductID, Quantity
FROM [Order Details]
ORDER BY OrderID
```

Resultado:

	OrderID	ProductID	Quantity
1	10248	11	12
2	10248	42	10
3	10248	72	5

Ejemplo 3:

```
--En el ejemplo siguiente se mostraran el 10% de todos los pedidos almacenados en la tabla Order Details
SELECT TOP 10 PERCENT OrderID, ProductID, Quantity
FROM [Order Details]
```

Resultado:

	OrderID	ProductID	Quantity
1	10248	11	12
2	10248	42	10
3	10248	72	5
4	10249	14	9
5	10249	51	40
6	10250	41	10
7	10250	51	35
8	10250	65	15
9	10251	22	6

Renombrar columnas de una consulta.

En ocasiones en las consultas es necesario colocarles un sobrenombre a las columnas, ya que en ocasiones en algunos resultados de la consulta algunas columnas se obtienen a partir de operaciones o las columnas tienen nombres en inglés y así se puede poner un nombre más entendible.

Para colocar el sobrenombre a la columna se utiliza la instrucción AS.

Ejemplo 1:

```
--Seleccionar los datos almacenados en el campo CategoryName de la tabla Categories y renombrar  
-- a la columna con el nombre Nombre de Categoria  
SELECT CategoryName AS [Nombre de Categoria]  
FROM Categories
```

Resultado:

	Nombre de Categoria
1	Beverages
2	Condiments
3	Confections
4	Dairy Products
5	Grains/Cereals
6	Meat/Poultry
7	Produce
8	Seafood

Ejemplo 2:

```
--Se quiere conocer cual seria la fecha de envio (ShippedDate) con un retraso de 5 dias  
--Mostrar los campos OrderID, OrderDate y ShippedDate de la tabla Orders  
SELECT OrderId, OrderDate, ShippedDate, ShippedDate + 5 AS RetrasoEnvio  
FROM Orders
```

Resultado:

	OrderId	OrderDate	ShippedDate	RetrasoEnvio
1	10248	1996-07-04 00:00:00.000	1996-07-16 00:00:00.000	1996-07-21 00:00:00.000
2	10249	1996-07-05 00:00:00.000	1996-07-10 00:00:00.000	1996-07-15 00:00:00.000
3	10250	1996-07-08 00:00:00.000	1996-07-12 00:00:00.000	1996-07-17 00:00:00.000
4	10251	1996-07-08 00:00:00.000	1996-07-15 00:00:00.000	1996-07-20 00:00:00.000
5	10252	1996-07-09 00:00:00.000	1996-07-11 00:00:00.000	1996-07-16 00:00:00.000
6	10253	1996-07-10 00:00:00.000	1996-07-16 00:00:00.000	1996-07-21 00:00:00.000
7	10254	1996-07-11 00:00:00.000	1996-07-23 00:00:00.000	1996-07-28 00:00:00.000
8	10255	1996-07-12 00:00:00.000	1996-07-15 00:00:00.000	1996-07-20 00:00:00.000
9	10256	1996-07-15 00:00:00.000	1996-07-17 00:00:00.000	1996-07-22 00:00:00.000

III. Requerimientos

- Máquina con SQL Server 2012 o 2016
- Guía Número 4 de Modelamiento y diseño de base de datos

IV. Procedimiento

Parte 1: Iniciando sesión desde SQL Server Managment Studio

1. Abrir el **SQL Server Management Studio**

Para conectarse con el servidor de base de datos elija los siguientes parámetros de autenticación:

- **Tipo de servidor:** Database Engine
- **Nombre del servidor:** Colocar el nombre del servidor local, por ejemplo PCNumMaquina-SALAx
Nota: NumMaquina es el número de la maquina local
- **Autenticación:** SQL Server Authentication
- **Login:** sa
- **Password:** 123456

Parte 2. Haciendo uso de sentencias de manipulación de datos

1. Para los ejercicios de esta parte, se utilizará la base de datos Northwind
2. Digitar la siguiente instrucción SQL en el editor de consultas:

```
USE Northwind
```

Ejercicio 1. Uso de la cláusula FROM

1. Seleccionar todas las columnas de la tabla Categories

```
SELECT * FROM Categories
```

2. Seleccionar los campos CategoryName y CategoryID de la tabla Categories

```
SELECT CategoryName, CategoryID FROM Categories
```

Ejercicio 2. Uso de la cláusula WHERE

1. Seleccionar los campos OrderID, ProductID y UnitPrice de la tabla Order Details donde el OrderID sea igual a 10251

```
SELECT OrderID, ProductID, UnitPrice  
FROM [Order Details]  
WHERE OrderID=10251
```

2. Seleccionar el campo CompanyName de la tabla Customers donde este sea igual a Alfreds Futterkiste

```
SELECT CompanyName FROM Customers  
WHERE CompanyName='Alfreds Futterkiste'
```

3. Seleccionar los campos OrderID, ProductID y UnitPrice de la tabla Order Details donde OrderID sea igual a 10251 y ProductID sea igual a 57

```
SELECT OrderID, ProductID, UnitPrice  
FROM [Order Details]  
WHERE OrderID=10251 AND ProductID=57
```

4. Seleccionar los campos OrderID, ProductID y UnitPrice de la tabla Order Details donde OrderID sea igual a 10251 o donde productid sea igual a 57

```
SELECT OrderID, ProductID, UnitPrice  
FROM [Order Details]  
WHERE OrderID=10251 OR ProductID=57
```

5. Seleccionar los campos OrderID, ProductID y UnitPrice de la tabla Order Details donde el dato almacenado en OrderID sea mayor o igual a 11000

```
SELECT OrderID, ProductID, UnitPrice  
FROM [Order Details]  
WHERE OrderID>=11000
```

6. Seleccionar los campos OrderID, ProductID y UnitPrice de la tabla Order Details donde el dato almacenado en OrderID sea mayor o igual a 11000 y OrderID sea menor o igual a 11003

```
SELECT OrderID, ProductID, UnitPrice  
FROM [Order Details]  
WHERE OrderID>=11000 AND OrderID<=11003
```

7. El mismo resultado de la consulta anterior, utilizando la instrucción BETWEEN

```
SELECT OrderID, ProductID, UnitPrice
FROM [Order Details]
WHERE OrderID BETWEEN 11000 AND 11003
```

8. Seleccionar el campo CompanyName de la tabla Customers donde el dato almacenado en ese campo comience con la letra A

```
SELECT CompanyName FROM Customers
WHERE CompanyName LIKE 'A%'
```

9. Seleccionar el campo CompanyName de la tabla Customers donde el dato almacenado en ese campo termine con los caracteres MA

```
SELECT CompanyName FROM Customers
WHERE CompanyName LIKE '%MA'
```

10. Seleccionar el campo RegionDescription de la tabla Region donde el dato almacenado en el campo contiene la palabra TERN

```
SELECT RegionDescription FROM Region
WHERE RegionDescription LIKE '%TERN%'
```

11. Seleccionar el campo OrderID de la tabla Order Details donde el dato almacenado en el campo comience con un numero cualquiera pero el cual debe terminar con los dígitos 0285

```
SELECT OrderID FROM [Order Details]
WHERE OrderID LIKE '_0285'
```

12. Seleccionar el campo CompanyName de la tabla Customers donde el dato almacenado en ese campo comience con cualquiera de las letras a,b o c

```
SELECT CompanyName FROM Customers
WHERE CompanyName LIKE '[a-c]%'
```

13. Seleccionar todos los campos de la tabla Order Details donde el campo OrderID tenga alguno de los siguientes datos: 10248, 10255 y 10270

```
SELECT * FROM [Order Details]
WHERE OrderID IN (10248, 10255, 10270)
```

14. El mismo resultado de la consulta anterior pero utilizando el operador lógico OR

```
SELECT * FROM [Order Details]
WHERE OrderID=10248 OR OrderID=10255 OR OrderID=10270
```

Ejercicio 3. Uso de la cláusula ORDER BY

1. Ordenar de manera ascendente los registros almacenados en los campos CategoryName y CategoryID de la tabla Categories

```
SELECT CategoryName, CategoryID FROM Categories
ORDER BY CategoryName ASC
```

2. El ordenamiento ascendente es el que viene por defecto, por lo tanto obtenemos el mismo resultado de la consulta anterior, con esta:

```
SELECT CategoryName, CategoryID FROM Categories
ORDER BY CategoryName
```

3. Ordenar de manera descendente los registros almacenados en los campos CategoryName y CategoryID de la tabla Categories

```
SELECT CategoryName, CategoryID FROM Categories
ORDER BY CategoryName DESC
```

Ejercicio 4. Uso de la instrucción DISTINCT

1. Digitar la siguiente consulta

```
SELECT SupplierID, CategoryID FROM Products
```

2. Se obtienen los siguientes resultados:

	SupplierID	CategoryID
1	1	1
2	1	1
3	1	2
4	2	2
5	2	2
6	3	2

Registros duplicados

3. Con la siguiente consulta, se obtendrán los mismos resultados pero mostrando un único registro de todos los que están repetidos que se mostraron en la consulta anterior

Seleccionar un único registro de los campos SupplierID y CategoryID de la tabla Products

```
SELECT DISTINCT SupplierID, CategoryID FROM Products
```

- Se obtienen los siguientes resultados:

	SupplierID	CategoryID
1	1	1
2	1	2
3	2	2
4	3	2
5	3	7
6	4	6

Ejercicio 5. Renombrar columnas de una consulta

- Con la instrucción AS se está renombrando una columna del resultado de una consulta en este ejercicio se está colocando un sobrenombre a las columnas CategoryID y CategoryName de la tabla Categories

```
SELECT CategoryID AS [Codigo de la Categoria],  
       CategoryName AS [Nombre de la Categoria]  
FROM Categories
```

- En este ejemplo se ha colocado a la columna el sobrenombre Aumento a la operación de calcular un aumento del 10% al precio original del producto

```
SELECT ProductID, ProductName, UnitPrice, UnitPrice*1.10 AS Aumento  
FROM Products
```

Ejercicio 6. Uso de la instrucción TOP N

Para estos ejercicios se ha calculado la venta (unidades vendidas por el precio unitario) de cada orden y se ha renombrado a la columna con el nombre Venta Total

- Seleccionar las primeras 3 mejores ventas de la tabla Order Details

```
SELECT TOP 3 OrderID, (UnitPrice*Quantity) AS [Venta Total]  
FROM [Order Details]  
ORDER BY [Venta Total] DESC
```

- Seleccionar las tres mejores ventas, utilizando la instrucción WITH TIES se permitirá incluir en los resultados los registros que tengan el mismo valor en el cálculo de la Venta Total

```
SELECT TOP 3 WITH TIES OrderID, (UnitPrice*Quantity) AS [Venta Total]  
FROM [Order Details]  
ORDER BY [Venta Total] DESC
```

3. Seleccionar el 25% del total de registros de las ventas las cuales están almacenadas en la tabla Order Details

```
SELECT TOP 25 PERCENT OrderID, (UnitPrice*Quantity) AS [Venta Total]
FROM [Order Details]
ORDER BY [Venta Total] DESC
```

Guardar el script con el nombre: **ConsultasSelect_Guia4.sql**

V. Ejercicio complementario

- Tomando la base de datos **AdventureWorks2012** crear las siguientes consultas de selección:
 - a. Seleccionar todos los datos de la tabla **Sales.SalesPerson**
 - b. Seleccionar todos los registros de los primeros 4 campos de la tabla **Production.Product**
 - c. Mostrar los 10 productos con el costo (StandardCost) más alto almacenados en la tabla **Production.ProductCostHistory**
 - d. Seleccionar los campos Name, ProductNumber, ListPrice de la tabla **Production.Product**, debe renombrar cada campo en el siguiente orden: Nombre Producto, Numero de Producto y Precio, los registros se deben ordenar de forma ascendente
 - e. Seleccionar los primeros tres campos de la tabla **Purchasing.Vendor** donde el dato almacenado en el campo AccountNumber comience con cualquier letra del rango de la G a la T
 - f. Mostrar los datos de la tabla **Person.CountryRegion** donde el campo CountryRegionCode contenga cualquiera de los siguientes datos: AR, BO, CO, ES, SV y VN
 - g. Seleccionar el campo CountryRegionCode de la tabla **Person.StateProvince**, pero en el resultado los datos no tienen que repetirse
 - h. Seleccionar los campos SalesOrderID, OrderQty de la tabla **Sales.SalesOrderDetail** en donde los datos del campo UnitPrice se encuentre entre los valores de 200 y 1000
 - i. Mostrar los campos ProductID, ListPrice de la tabla **Production.Product** y una columna más que muestre un aumento del 15% del dato almacenado en ListPrice renombrar esta nueva columna con el nombre Aumento de Precio
 - j. Seleccionar el 20% de los registros almacenados en la tabla **Sales.SalesOrderDetail**
- Colocar como nombre al script: **ConsultasSelect_EjercicioComplementarioGuia4.sql**

VI. Análisis de resultados

Crear las siguientes consultas de selección:

1. Base de datos: **AdventureWorks2012**
 - a. Seleccionar todos los datos de la tabla **HumanResources.Department**
 - b. Seleccionar los campos BusinessEntityID, NationalIDNumber y JobTitle de la tabla **HumanResources.Employee** en donde en el campo JobTitle se encuentre la palabra Production

- c. Seleccionar los datos de la tabla **Sales.Customer** donde en el campo CustomerID se encuentren los siguientes datos: 2,4, 7 y 10
- d. Seleccionar los campos DepartmenID, Name de la tabla **HumanResources.Department** en donde los datos del DepartmenID se encuentre entre los valores 5 y 12
- e. Seleccionar los campos AddressID, City y StateProvinceID de la tabla **Person.Address** donde en el campo City el dato comienza con la letra B
- f. Seleccionar los datos de la tabla **Production.Culture**, donde el dato almacenado en el campo Name se encuentre entre los valores: English o Spanish
- g. Seleccionar el 50% de los datos de la tabla **Sales.CreditCard**
- h. Mostrar las 10 mejores ventas (LineTotal) de la tabla **Sales.SalesOrderDetail**
- i. Seleccionar el campo JobTitle de la tabla **HumanResources.Employee**, pero no deben mostrarse datos duplicados, ordenar los datos de forma descendente
- j. Mostrar los campos Name, ProductNumber y ListPrice y renombrar este campo como Price de la tabla **Production.Product** donde la línea de productos (ProductLine) sea igual a R y el valor correspondiente a los días para fabricar (DaysToMaufacture) sea inferior a 4

2. Base de datos: **Library**

- a. Seleccionar los datos de la tabla **adult**, donde en el campo expr_date los datos se encuentren en el rango 01/12/2006 y 30/06/2007
- b. Seleccionar los datos de la tabla **reservation**
- c. Mostrar los autores (author) de la tabla **title** que comiencen con la letra M
- d. Mostrar el nombre del autor (author) y el titulo (title) de la tabla title y ordenarlos de forma ascendente por el nombre del autor
- e. Seleccionar los datos de la tabla **loan** donde el campo isbn tenga los siguientes datos: 509, 519, 529 y 539

3. Base de datos: **Pubs**

- a. Seleccionar de la tabla **employee** los datos en donde el campo lname contenga la letra K
- b. Seleccionar de la tabla **employee** los datos del campo emp_id que comience con cualquiera de las letras que se encuentren en el rango de la F a la M
- c. Mostrar de la tabla **sales** los datos donde la fecha de perdido (ord_date) sean mayores o iguales a 01/01/1994
- d. Seleccionar de la tabla **stores** los datos donde el campo state sea igual a CA
- e. Seleccionar de la tabla **title** los 5 libros más caros (price) que se encuentran almacenados

VI. Fuente de consulta

1. La Biblia de SQL Server 2005

Madrid, España: Anaya, 2006

Autor: Mike Gundelerloy y Joseph L. Jorden

Biblioteca UDB – Clasificación: 005.361 G975 2006

2. Microsoft SQL Server 2008: Guía del Administrador

Madrid, España: ANAYA, 2009

Autor: William Stanek

Biblioteca UDB – Clasificación: 005.361 S784 2009

3. [https://msdn.microsoft.com/es-sv/library/ms187731\(v=sql.110\).aspx](https://msdn.microsoft.com/es-sv/library/ms187731(v=sql.110).aspx)