Microsoft SQL Server
Sesión 3

Usando Transact-SQL

1. Consulta de los datos
2. Funciones SQL

La instrucción SELECT

Clausula	Descripción
SELECT	Seguida de una lista de columnas o un asterisco. Indicando que quieres obtener todas las columnas
FROM	Seguida del nombre de la tabla o vista, o múltiples tabla con expresiones de JOIN
WHERE	Seguida por criterios de filtraje
ORDER BY	Seguida por una lista de columna de ordenación

Ejemplos

- SELECT Name, StandardCost, Color FROM Production.Product
- SELECT * FROM Production.Product

Consulta Multi-Tabla

PRÁCTICA 3.1: Ejecutar las siguientes consultas y analizar sus diferencias.

1

```
SELECT ProductID
   ,Name
   ,Color
   ,StandardCost
   ,ListPrice
FROM Production.Product
```

2 Agregando información sobre el Modelo

```
SELECT ProductID
   ,Name
   ,Color
   ,StandardCost
   ,ListPrice
   ,ProductModelId
   ,Name
FROM Production.Product
   INNER JOIN Production.ProductModel
      ON Production.Product.ProductModelID =
         Production.ProductModel.ProductModelID
```

3 Falla corregida

```
SELECT ProductID
   , Production.Product.Name
   ,Color
   ,StandardCost
   ,ListPrice
   , Production.ProductModel.ProductModelID
   , Production.ProductModel.Name
FROM Production.Product
   INNER JOIN Production.ProductModel
      ON Production.Product.ProductModelID =
         Production.ProductModel.ProductModelID
```

4 Nombre completo

```
SELECT Production.Product.ProductID
   ,Production.Product.Name AS Product
   ,Production.Product.Color
   ,Production.Product.StandardCost
   ,Production.Product.ListPrice
   ,Production.ProductModel.ProductModelId
   ,Production.ProductModel.Name AS Model
FROM Production.Product
   INNER JOIN Production.ProductModel
      ON Production.Product.ProductModelID =
         Production.ProductModel.ProductModelID
```

Esquemas y Resolución de Nombres

- Cada objeto en SQL Server es identificado por cuatro partes
  - Servidor
  - Base de datos
  - Esquema
  - Objeto
- Algunos de acuerdo al contexto pueden ser omitidos
- Ejemplo

```
SELECT AdventureWorks2019.Production.Product.ProductID
FROM [SERVER].AdventureWorks2019.Production.Product
```

- Para identificar el nombre del servidor

```
SELECT * FROM SYS.servers
```

- El esquema por defecto en dbo.

Alias de Columnas

- Razones para cambiar el nombre a una columna
  - Más fácil de entender
  - Nombre más descriptivo
  - Compatibilidad hacia atrás
  - Nombre repetidos en consultas multi-tablas
- **PRACTICA 3.2**: Ejecutar el ejemplo de tres diferentes sintaxis para implementar un alias en una columna.

Sintaxis	Descripción	Ejemplo
Column AS *Alias*	La técnica más legible.	SELECT ListPrice - StandardCost AS **Margin** FROM Production.Product
Column Alias	La técnica mas común	SELECT ListPrice - StandardCost **Margin** FROM Production.Product
Alias = Column	No es común en T-SQL	SELECT **Margin** = ListPrice - StandardCost FROM Production.Product

Columnas Calculadas y Derivadas

- La columna calculada es resultado de una expresión o calculo.
- **PRÁCTICA 3.3**: Ejecutar los ejemplos mostrados

1. Calculando un cantidad a partir de dos columnas

```
SELECT SalesOrderID, ProductID
    ,UnitPrice * OrderQty As PurchasePrice
FROM Sales.SalesOrderDetail
```

2. Usando una función

```
SELECT NationalIDNumber,BirthDate
    ,DATEDIFF(YY, BirthDate, GETDATE()) As Age
FROM HumanResources.Employee
```

3. Con un valor constante

```
SELECT Name, ListPrice,
    'Mountain Bike' AS SubCategoryName
FROM Production.Product WHERE ProductSubCategoryID = 1
```

4. Concatenando campos y con alias en columnas y tablas

```
SELECT
    PP.FirstName + ' ' + PP.LastName AS Name
    , PP.Title AS Titulo
FROM Person.Person AS PP
ORDER BY Titulo
```

Filtrando Registros

- Existen dos maneras para limitar la cantidad de registros resultantes de una consulta
  - WHERE revisa cada registro contra un criterio de filtraje
  - TOP limita los registros de acuerdo a una cantidad de registros
- Ejemplos

```
SELECT Name, StandardCost, Color
FROM Production.Product
WHERE Color = 'Black'
```

```
SELECT Name, ListPrice
FROM Production.Product
WHERE ListPrice < 5.00
```

Operadores de Comparación

Operadores

Operador	Descripción
=	Igual que
<> Ó !=	No igual que
<	Menor que
>	Mayor que
!<	No menor que
!>	No mayor que
<=	Menor o igual que
>=	Mayor o igual que
LIKE	Para comparar valores de caracteres con comodines

Operador LIKE

Comodín	Descripción
%	Cualquier cadena de cero o más caracteres
_	Cualquier único carácter
[]	Cualquier único carácter dentro del rango especificado
[^]	Cualquier único carácter fuera del rango especificado

PRÁCTICA 3.4a: Creando Tabla de pruebas

Tabla de Pruebas

Lastname	Firstname	Position
Flintstone	Fred	Bronto Driver
Rubble	Barney	Accountant
Turley	Paul	Developer
Wood	Dan	DBA
Rockhead	Don	System Administrator
Rockstone	Pauline	Manager

Ejecutar las siguientes Instrucciones

```
USE AdventureWorks2019
GO

CREATE TABLE dbo.SlateGravel
(     LastName varchar(25) NULL
      ,FirstName varchar(25) NULL
      ,Position varchar(25) NULL);

INSERT SlateGravel
VALUES
      ('Flintstone', 'Fred', 'Bronto Driver')
      ,('Rubble', 'Barney', 'Accountant')
      ,('Turley', 'Paul', 'Developer')
      ,('Wood', 'Dan', 'DBA')
      ,('Rockhead', 'Don', 'System Administrator')
      ,('Rockstone', 'Pauline', 'Manager')
```

PRÁCTICA 3.4b: Analizando el uso del operador LIKE

1. Ejecutar las siguientes instrucciones y analizar el resultado.

No	Instrucción	Resultado Esperado
1	SELECT * FROM SlateGravel WHERE LastName LIKE 'Flint%'	Flintstone
2	SELECT * FROM SlateGravel WHERE LastName LIKE '%stone'	Flintstone y Rockstone
3	SELECT * FROM SlateGravel WHERE LastName LIKE '%sto%'	Flintstone y Rockstone
4	SELECT * FROM SlateGravel WHERE LastName LIKE '_urley'	Turley

PRÁCTICA 3.4c: Analizando el uso del operador LIKE

1. Ejecutar las siguientes instrucciones y analizar el resultado.

No	Instrucción	Resultado Esperado
5	SELECT * FROM SlateGravel WHERE FirstName LIKE 'D[ao]n'	Dan y Don
6	SELECT * FROM SlateGravel WHERE FirstName LIKE 'D[a-o]n'	Dan y Don
7	SELECT * FROM SlateGravel WHERE FirstName LIKE 'D[^o]n'	Dan
8	SELECT * FROM SlateGravel WHERE FirstName NOT LIKE 'Dan'	Dan

Comparaciones Lógicas

- **PRACTICA 3.5**: Ejecutar los ejemplos con diferentes operadores lógicos.

Sintaxis	Descripción	Ejemplo
AND	Todos los criterios deben ser verdaderos	SELECT ProductID, Name, ListPrice FROM Production.Product WHERE ProductSubCategoryID = 1 AND ListPrice < 1000
OR	Al menos uno de los criterios debe ser verdadero	SELECT ProductID, Name, ListPrice FROM Production.Product WHERE ProductSubCategoryID = 1 OR ListPrice < 1000
NOT	El criterio debe ser falso	SELECT ProductID, Name, ListPrice FROM production.Product WHERE NOT ProductSubCategoryID = 2
NULL	La columna tiene un valor nulo	SELECT ProductID, Name, Color FROM Production.Product WHERE Color IS NULL
NOT NULL	La columna no tiene un valor nulo	SELECT ProductID, Name, Color FROM Production.Product WHERE Color IS NOT NULL

El operador BETWEEN y la función IN

PRACTICA 3.6a
El operador BETWEEN

1. **Ejecutar las siguientes instrucciones**

```
SELECT NationalIDNumber, LoginID
FROM HumanResources.Employee
WHERE BirthDate > = '1962-1-1'
AND BirthDate < = '1985-12-31'
```

2. **Ejecutar utilizando su equivalente con el operador BETWEEN**

```
SELECT NationalIDNumber, LoginID
FROM HumanResources.Employee
WHERE BirthDate BETWEEN '1962-1-1' AND
    '1985-12-31'
```

PRACTICA 3.6b
La función IN

1. **Ejecutar las siguientes instrucciones**

```
SELECT ProductID
    ,Name AS Product
FROM Production.Product
WHERE ProductSubCategoryID = 1
OR ProductSubCategoryID = 2
OR ProductSubCategoryID = 3
```

2. **Ejecutar utilizando su equivalente con la función IN**

```
SELECT ProductID
    ,Name AS Product
FROM Production.Product
WHERE ProductSubCategoryID IN (1,2,3)
```

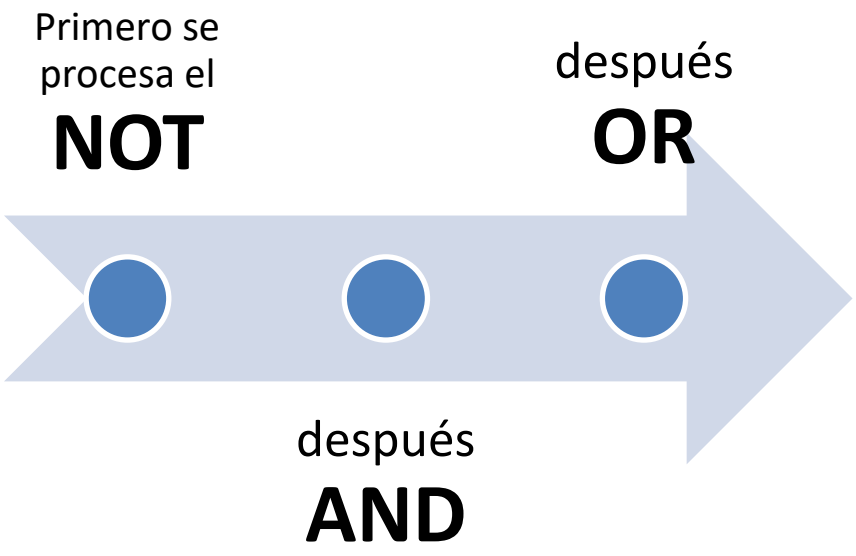
La función IN con sub-consultas

PRACTICA 3.6c: Ejecutar la siguiente consulta que recupera todos los productos cuyos ProductCategoryID son 1 ó 2. Como la tabla de productos solo contiene sub-categorias se debe hacer una sub consulta a la tabla de categoría.

```
SELECT ProductID
    ,Name AS Product
FROM Production.Product
WHERE ProductSubCategoryID IN (
                                SELECT ProductSubCategoryID
                                FROM Production.ProductSubCategory
                                WHERE ProductCategoryID IN (1,2))
```

Precedencia de los Operadores

Orden de Precedencia



PRÁCTICA 3.7: Encontrar error

1. Verificar que la siguiente consulta genera una lista de bicicletas montañeras y bicicleta de carretera con precios mayores de $500 y menores de $1000

```
SELECT Name
   ,ProductNumber
   ,ListPrice
   ,ProductSubCategoryID
FROM Production.Product
WHERE ProductSubCategoryID = 1
   OR ProductSubCategoryID = 2
   AND ListPrice > 500
   AND ListPrice < 1000
```

2. Utilizar paréntesis para lograr que la consulta funcione correctamente

Orden de Registros

- Se utiliza la clausula ORDER BY después de WHERE
- Puede contener una o mas columnas delimitadas por comas
- Se puede especificar la dirección
  - ASC (ascendente)
  - DESC (descendente)
- Si no se especifica es ascendente
- También se pueden escribir campos calculados en lugar de columnas

PRÁCTICA 3.8: Ejecutar las siguientes consultas y analizar los resultados

1.

```
SELECT Name AS Product
    ,ListPrice
    ,StandardCost
FROM Production.Product
WHERE ListPrice > 0
ORDER BY ListPrice DESC, StandardCost
```

2.

```
SELECT SalesOrderID, ProductID
   ,UnitPrice * OrderQty As
   PurchasePrice
FROM Sales.SalesOrderDetail
order by UnitPrice * OrderQty
```

3.

```
SELECT SalesOrderID, ProductID
   ,UnitPrice * OrderQty As
   PurchasePrice
FROM Sales.SalesOrderDetail
order by PurchasePrice
```

Operadores TOP, TIE y PERCENT

- TOP n
  - Se escribe después de SELECT y “n” indica que se mostrarán las primeras “n” líneas según el orden establecido en la consulta
- WITH TIES
  - Se escribe a continuación de TOP “n” y mostrará las líneas que coincidan con la línea n a continuación de las primeras “n” líneas
- PERCENT
  - Muestra un porcentaje de la cantidad de registros en la tabla en lugar de una cantidad de líneas

PRÁCTICA 3.9: Ejecutar las siguientes consultas y analizar los resultados

1.
```
SELECT TOP 10 Name,
   ListPrice
FROM Production.Product
ORDER BY ListPrice DESC
```

2.
```
SELECT TOP 10 WITH TIES Name,
   ListPrice
FROM Production.Product
ORDER BY ListPrice DESC
```

3.
```
SELECT TOP 10 PERCENT Name,
   ListPrice
FROM Production.Product
ORDER BY ListPrice DESC
```

La instrucción CASE

- Es una expresión escalar que regresa un valor basado en una lógica condicional
- Se puede usar en SELECT, WHERE, HAVING, ORDER BY y otros
- Si no se escribe ELSE entonces por defecto se tiene ELSE NULL
- Dos formatos
  - Una función CASE simple compara una expresión con un juego simple de expresiones para determinar el resultado
  - Una función CASE de búsqueda evalúa una expresión booleana para determinar el resultado

PRÁCTICAS con la instrucción CASE

PRÁCTICA 3.10a: Ejecutar la siguiente consulta y analizar los resultados

```
SELECT ProductNumber
  ,Category =
    CASE ProductLine
      WHEN 'R' THEN 'Road'
      WHEN 'M' THEN 'Mountain'
      WHEN 'T' THEN 'Touring'
      WHEN 'S' THEN 'Other sale items'
      ELSE 'Not for sale'
    END
  ,Name
FROM Production.Product
ORDER BY ProductNumber
```

PRÁCTICA 3.10b: Ejecutar la siguiente consulta y analizar los resultados

```
SELECT ProductNumber
  ,Name
  ,'Price Range' =
    CASE
    WHEN ListPrice = 0
      THEN 'Mfg item - not for resale'
    WHEN ListPrice < 50
      THEN 'Under $50'
    WHEN ListPrice >= 50 and ListPrice < 250
      THEN 'Under $250'
    WHEN ListPrice>=250 and ListPrice < 1000
      THEN 'Under $1000'
    ELSE 'Over $1000'
  END
FROM Production.Product
ORDER BY ProductNumber
```