W|E EDUCACIÓN EJECUTIVA

SQL Server

Sesión 3

Sesión 3 | Objetivo

1. Funciones de Agregación
2. Uso de Group BY
3. Tipos de Datos
4. Tipos de Datos Carácter
5. Tipos de Datos Date y Time
6. Insertar Datos en Tablas
7. Actualización de Datos de Tablas
8. Uso de Subqueries

Funciones de Agregación

Retorna un valor escalar.
Ignora los valores NULL excepto cuando se utiliza el COUNT(*)
Puede ser usado en : SELECT, HAVING, ORDER BY
Frecuentemente se utiliza con la clausula GROUP BY

Comunes

- SUM
- MIN
- MAX
- AVG
- COUNT
- COUNT_BIG

Estadísticas

- STDEV
- STDEVP
- VAR
- VARP

Otras

- CHECKSUM_AGG
- GROUPING
- GROUPING_ID

Uso de Group By

Podemos utilizar la clausula DISTINCT para sumarizar solo valores únicos.
Ejemplo:

```
SELECT empid, YEAR(orderdate) AS orderyear,
COUNT(custid) AS all_custs,
COUNT(DISTINCT custid) AS unique_custs
FROM Sales.Orders
GROUP BY empid, YEAR(orderdate);
```

```
empid       orderyear   all_custs   unique_custs
----------- ----------- ----------- ------------
1           2006        26          22
1           2007        55          40
1           2008        42          32
2           2006        16          15
```

Uso de Group By

La clausula GROUP BY me permite crear grupos que me retornen un conjunto de filas.
El detalle de las filas es perdido.
Sintaxis:

```
SELECT <select_list>
FROM <table_source>
WHERE <search_condition>
GROUP BY <group_by_list>;
```

Ejemplo:

```
SELECT empid, COUNT(*) AS cnt
FROM Sales.Orders
GROUP BY empid;
```

Uso de Group By

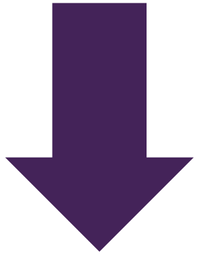
```
SELECT SalesOrderID,
SalesPersonID, CustomerID
FROM Sales.SalesOrderHeader;
```

SalesOrder ID	SalesPerson ID	CustomerID
43659	279	29825
43660	279	29672
43661	282	29734
43662	282	29994
43663	276	29565
…	…	…
75123	NULL	18759

```
WHERE
CustomerID IN
(29777, 30010)
```

SalesOrder ID	SalesPerson ID	Customer ID
51803	290	29777
69427	290	29777
44529	278	30010
46063	278	30010

```
GROUP BY
SalesPersonID
```



SalesPersonID	Count(*)
278	2
290	2

Tipos de datos en SQL Server

- El tipo de dato indica el valor que puede ser almacenado en una columna o variable.
- Se pueden crear tipos de datos personalizados.

Tipos de datos

- Numéricos

Tipo de Dato	Rango	Storage (bytes)
tinyint	0 to 255	1
smallint	-32,768 to 32,768	2
int	2^{31} (-2,147,483,648) to 2^{31}-1 (2,147,483,647)	4
bigint	-2^{63} - 2^{63}-1 (+/- 9 quintillion)	8
bit	1, 0 or NULL	1
decimal/numeric	-10^{38} +1 through 10^{38} – 1 when maximum precision is used	5-17
money	-922,337,203,685,477.5808 to 922,337,203,685,477.5807	8
smallmoney	-214,748.3648 to 214,748.3647	4

Tipos de Datos

- Binarios

Tipo de Dato	Rango	Storage (bytes)
binary(n)	1 to 8000 bytes	n bytes
varbinary(n)	1 to 8000 bytes	n bytes + 2
varbinary(max)	1 to 2.1 billion (approx.) bytes	n bytes + 2

- Nota:El tipo de datos image es también un string binario pero está marcado en los tipos de datos que no serán ya utilizados , por lo que debería empezar a utilizar el **varbinary(max)**.

Tipos de Datos

- Otros tipos de datos

Tipo de Dato	Rango	Storage (bytes)	Descripción
xml	0-2 GB	0-2 GB	Stores XML in native hierarchical structure
uniqueidentifier	Auto-generated	16	Globally unique identifier (GUID)
hierarchyid	n/a	Depends on content	Represents position in a hierarchy
rowversion	Auto-generated	8	Previously called timestamp
geometry	0-2 GB	0-2 GB	Shape definitions in Euclidian geometry
geography	0-2 GB	0-2 GB	Shape definitions in round-earth geometry
sql_variant	0-8000 bytes	Depends on content	Can store data of various other data types in the same column
cursor	n/a	n/a	Not a storage datatype—used for cursor operations
table	n/a	n/a	Not a storage data type—used for query operations

Tipos de Datos

- Tipos de Conversión de Datos
  - Implícitos
    - Cuando se compara un tipo de dato con otro
    - Transparente al usuario final

```
WHERE <column of smallint type> = <value of int type>
```

  - Explícitos
    - Cuando usas CAST o CONVERT

```
CAST(unitprice AS INT)
```

  - **Nota:** El tipo de dato con menor precedencia es convertido al tipo de dato con mayor precedencia.

Tipos de Datos Carácter

- Para concatenar cadenas se tienen dos opciones
  - Usando la función CONCAT

```
SELECTcustid, city, region, country,
  CONCAT(city, ', ' + region, ', ' + country) AS location
FROM Sales.Customers;
```

  - Usando el operador +

```
SELECT empid, lastname, firstname,
firstname + N' ' + lastname AS fullname
FROM HR.Employees;
```

Tipos de Datos Carácter

- Funciones para Strings

Function	Syntax	Remarks
SUBSTRING	SUBSTRING (expression , start , length)	Returns part of an expression.
LEFT, RIGHT	LEFT (expression , integer_value) RIGHT (expression , integer_value)	LEFT returns left part of string up to integer_value. RIGHT returns right part of string up to integer value.
LEN, DATALENGTH	LEN (string_expression) DATALENGTH (expression)	LEN returns the number of characters in string_expression, excluding trailing spaces. DATALENGTH returns the number of bytes used.
CHARINDEX	CHARINDEX (expressionToFind, expressionToSearch)	Searches expressionToSearch for expressionToFind and returns its start position if found.
REPLACE	REPLACE (string_expression , string_pattern , string_replacement)	Replaces all occurrences of string_pattern in string_expression with string_replacement.
UPPER, LOWER	UPPER (character_expression) LOWER (character_expression)	UPPER converts all characters in a string to uppercase. LOWER converts all characters in a string to lowercase.

Tipos de Datos Carácter

- El predicado LIKE me permite buscar patrones entre strings
- Los caracteres utilizados junto al LIKE son:
  - %, _ , ^ , [ ]

```
SELECT categoryid, categoryname, description
FROM Production.Categories
WHERE description LIKE 'Sweet%';
```

Tipos de Datos Date y Time

- Las versiones antiguas solo soportan datetime y smalldatetime
- SQL Server 2008 introdujo date,time,datetime2 y datetimeoffset
- SQL Server 2012 agrego nuevas funcionalidades para trabajar con estos tipos de datos

Tipo Dato	Storage (bytes)	Rango Dato(Gregorian Calendar)	Exactitud	Formato Recomendado Entrada
datetime	8	January 1, 1753 to December 31, 9999	Rounded to increments of .000, .003, or .007 seconds	YYYYMMDD hh:mm:ss[.mmm]
smalldatetime	4	January 1, 1900 to June 6, 2079	1 minute	YYYYMMDD hh:mm:ss[.mmm]
datetime2	6 to 8	January 1, 0001 to December 31, 9999	100 nanoseconds	YYYYMMDD hh:mm:ss[.nnnnnnn]
date	3	January 1, 0001 to December 31, 9999	1 day	YYYY-MM-DD
time	3 to 5	n/a – time only	100 nanoseconds	hh:mm:ss[.nnnnnnn]
datetimeoffset	8 to 10	January 1, 0001 to December 31, 9999	100 nanoseconds	YYYY-MM-DDThh:mm:ss[.nnnnnnn][{+|-}hh:mm]

Tipos de Datos Date y Time

- Tip:
  - Cuando se realizan consultas con fechas se podría omitir el tiempo.
  - Las consultas con el operador '=' trabajan con el time de 00:00:00

```
SELECT orderid, custid, empid, orderdate
FROM Sales.Orders
WHERE orderdate= '20070825';
```

  - Si la data almacena el time entonces no se recomienda utilizar el '=' , sino un rango:

```
SELECT orderid, custid, empid, orderdate
FROM Sales.Orders
WHERE orderdate >= '20070825'
AND orderdate <  '20070826';
```

Tipos de Datos Date y Time

- Funciones de Fecha y Tiempo:
  - Fecha del sistema:
    - GETDATE,GETUTCDATE,SYSDATETIME
  - Parte de Fecha y tiempo
    - DATENAME,DATEPART
  - Diferencia entre fechas
    - DATEDIFF,DATEDIFF_BIG
  - Modificar fechas y tiempo
    - DATEADD,EOMONTH
  - Validar fecha y tiempo
    - ISDATE
  - Construir un valor del tipo fecha y tiempo
    - DATETIME2FROMPARTS , DATEFROMPARTS

Insertar Datos a Tablas

- Insertar un registro de valores a la tabla

```
INSERT INTO Sales.OrderDetails
       (orderid, productid, unitprice, qty, discount)
VALUES (10255,39,18,2,0.05);
```

- Inserta múltiples valores a la tabla

```
INSERT INTO Sales.OrderDetails
(orderid, productid, unitprice, qty, discount)

VALUES
(10256,39,18,2,0.05),
(10258,39,18,5,0.10);
```

Insertar Datos a Tablas

- Insertar registro de valores por medio de un SELECT a la tabla

```
INSERT Sales.OrderDetails
(orderid, productid, unitprice, qty, discount)

SELECT * FROM NewOrderDetails
```

- Inserta valores llamando a un procedimiento almacenado

```
INSERT INTO Production.Products
(productID, productname, supplierid, categoryid, unitprice)
EXEC Production.AddNewProducts;
```

Actualizar Datos en Tablas

- La clausula UPDATE actualiza los campos en una tabla o vista.
- Las valores de las columnas son cambiados con la clausula SET.

```
UPDATE Production.Products
   SET   unitprice = (unitprice * 1.04)
WHERE   categoryid =  1 AND discontinued = 0
;
```

```
UPDATE Production.Products
   SET     unitprice *= 1.04
                     -- Using compound
                     -- assignment operators
WHERE   categoryid =  1 AND discontinued = 0;
```

Actualizar Datos en Tablas

- Puedo realizar JOINS entre tablas y actualizar una de ellas

```
UPDATE Reason  -- Notice use of Alias to make reading better
       SET Name += ' ?'

FROM Production.ScrapReason AS Reason
INNER JOIN Production.WorkOrder AS WorkOrder

ON     Reason.ScrapReasonID = WorkOrder.ScrapReasonID
AND    WorkOrder.ScrappedQty > 300;
```

Uso de Subqueries

- Las subqueries son queries dentro de otras.
- Puede darse que una subquerie me devuelva un escalar.
- En caso se retorne más de un escalar la saltará un error.

```
SELECT orderid, productid, unitprice, qty
FROM Sales.OrderDetails
WHERE orderid =
   (SELECT MAX(orderid) AS lastorder
    FROM Sales.Orders);
```

TIPS Y RECOMENDACIONES

- Practicar los ejercicios prácticos realizados en clase.
- Tener un orden al realizar las operaciones en SQL.

Gracias…