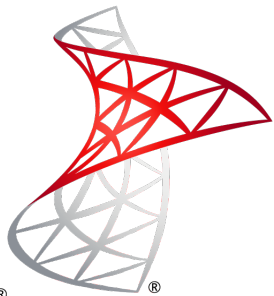


## 8 – Agrupar y Agregar Datos



Microsoft®  
**SQL Server®** 2012

Programa de Asesorías, Actualización y  
Capacitación Computacional



# Agenda

1. Funciones de Agregación
2. Cláusula GROUP BY
3. Cláusula HAVING



## **8.1 FUNCIONES DE AGREGACIÓN**



# Funciones de agregación

- Permiten realizar cálculos sobre conjuntos de datos
- Devuelven valores escalares en una columna sin nombre
- Con excepción de COUNT(\*), las demás funciones ignoran valores **NULL** (no los toman en cuenta)
- Frecuentemente usadas con la cláusula **GROUP BY**
- Pueden ser usadas también en cláusulas SELECT, HAVING y ORDER BY



# Funciones de agregación

Comunes	Estadísticas	Otras
SUM	STDEV	CHECKSUM_AGG
MIN	STDEVP	GROUPING
MAX	VAR	GROUPING_ID
AVG	VARP	
COUNT		
COUNT_BIG		



# Ejemplo

```
SELECT COUNT (DISTINCT SalesOrderID) AS UniqueOrders,
       AVG(UnitPrice) AS Avg_UnitPrice,
       MIN(OrderQty)AS Min_OrderQty,
       MAX(LineTotal) AS Max_LineTotal
FROM Sales.SalesOrderDetail;
```

UniqueOrders	Avg_UnitPrice	Min_OrderQty	Max_LineTotal
31465	465.0934	1	27893.619000

# DISTINCT con funciones de agregación

- Cuando se usa DISTINCT en combinación con funciones de agregación, **se resumen solo los valores únicos**
- Ejemplo 1:

```
SELECT COUNT(DISTINCT CustomerID)  
FROM Sales.SalesOrderHeader;
```



# DISTINCT con funciones de agregación (2)

- Ejemplo 2:

```
SELECT SalesPersonID, YEAR(OrderDate) AS OrderYear,
COUNT(CustomerID) AS All_Custs,
COUNT(DISTINCT CustomerID) AS Unique_Custs
FROM Sales.SalesOrderHeader
GROUP BY SalesPersonID, YEAR(OrderDate);
```

SalesPersonID	OrderYear	All_Custs	Unique_custs
289	2006	84	48
281	2008	52	27
285	2007	9	8
277	2006	140	57



# Ejercicios 8.1

Escriba las siguientes consultas:

- 1 Cuál es el precio del producto más caro y del más barato
- 2 Cuál es el promedio y la desviación estándar de los precios de los productos
- 3 Cuenta cuántas subcategorías distintas de productos hay para los productos de la tabla de Production.Product
- 4 Cuente cuántos Estados distintos hay en la tabla de direcciones.



## 8.2 CLÁUSULA GROUP BY



# GROUP BY

- **GROUP BY**: Agrupa las filas con base en la lista de agrupamiento (atributos) especificada.
- ¿Cómo se hace el agrupamiento?
  - Se agrupan las filas que tienen la **misma combinación de valores** en los atributos de la lista de agrupamiento
- Si una consulta usa GROUP BY, todas las fases posteriores (HAVING, SELECT y ORDER BY) operan sobre los grupos y no sobre las filas originales
- El detalle que contienen las filas originales se “pierde” después de que la cláusula GROUP BY es procesada.

# GROUP BY (2)

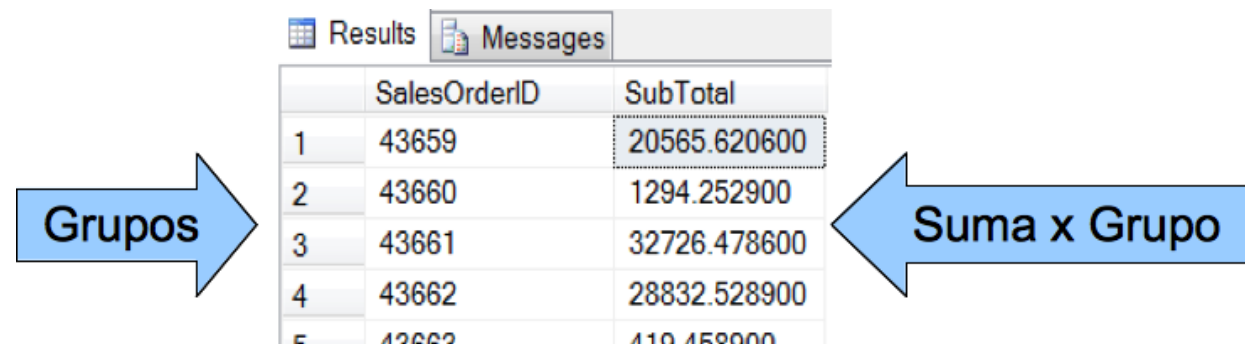
Estructura:

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

# GROUP BY: Ejemplo

- Recuperar la sumatoria de las ventas por cada SalesOrderID

```
SELECT SalesOrderID, SUM(LineTotal) AS SubTotal  
FROM Sales.SalesOrderDetail AS vents  
GROUP BY SalesOrderID
```



	SalesOrderID	SubTotal
1	43659	20565.620600
2	43660	1294.252900
3	43661	32726.478600
4	43662	28832.528900
5	43663	110.150000

# GROUP BY: Ejemplo (2)

- Agrupe por SalesPersonID y cuente cuántos hay
- La función agregada es calculada para cada grupo

```
SELECT SalesPersonID, COUNT(*) AS Cnt  
FROM Sales.SalesOrderHeader  
GROUP BY SalesPersonID;
```

# Consideraciones importantes

- Cuando se usa GROUP BY, todas las columnas que aparezcan en las cláusulas SELECT, HAVING y ORDER BY tienen que:
  - Aparecer en la cláusula GROUP BY también
  - O bien
  - Ser entradas en expresiones de agregación

## Ejercicios 8.2

Apoyándose del diagrama de AdventureWorksDB, realice las siguientes consultas:

- 1 Agrupe todas las direcciones por ciudad
- 2 Cuente cuántas direcciones hay por cada ciudad
- 3 Muestre los estados y ciudades

	name	city
1	Ontario	Ottawa
2	British Columbia	Burnaby
3	Nord	Dunkerque



## Ejercicios 8.2

- 4 Ordene la consulta 3 por nombre de estado
- 5 Agrupe la consulta 4 por nombre del estado ¿Qué sucede? ¿Por qué?
- 6 Agrupe la consulta 4 por nombre del estado y nombre de la ciudad ¿Cuál es el resultado?

## Ejercicios 8.2

7 Explique por qué falla la siguiente consulta:

```
SELECT custid, orderid  
FROM Sales.Orders  
GROUP BY custid;
```

8 Arregle la consulta anterior para que devuelva el customerID y el máximo orderID para ese cliente.

## 8.3 CLÁUSULA HAVING



# HAVING

- La cláusula **HAVING** permite filtrar los resultados de los grupos generados por **GROUP BY**
- Proporciona una **condición** de búsqueda que **cada grupo debe satisfacer**
- Si la columna agrupada contiene valores NULL, **todas las filas con valor NULL** en esa columna se consideran parte de un **mismo grupo**.
- A diferencia de la cláusula WHERE, que filtra filas **antes** de que los grupos sean creados, la cláusula HAVING filtra grupos **después** de que estos han sido creados.

# HAVING: Ejemplo

- La siguiente consulta muestra las ciudades para las cuales hay más de 5 direcciones registradas

```
USE AdventureWorks2012
SELECT a.city
FROM person.Address a
GROUP BY a.City
HAVING count(*) > 5
```

## Ejercicios 8.3

- 1 Muestre las ciudades que contengan entre 5 y 10 direcciones
- 2 Muestre la ciudad con la mayor cantidad de direcciones \*\*
- 3 Para cada subcategoría de productos, muestre la cantidad de productos que hay para esa subcategoría, así como el máximo y mínimo precio de los productos en esa categoría
- 4 A partir de la consulta 3, muestre solamente los grupos donde la cantidad sea mayor que 30
- 5 A partir de la consulta 3, ordene el resultado por la cantidad

\*\* Pista: Use TOP 1 y ordenamiento descendente

## Ejercicios 8.3

- 6 Para cada tamaño y unidad de medida de los productos, muestre la cantidad de productos que hay para esa combinación de tamaño y unidad de medida, así como el máximo y mínimo peso de los productos
- 7 A partir de la consulta 6, muestre solamente los grupos que tengan donde la cantidad sea mayor que 5 y el peso máximo sea mayor que 10
- 8 Ordene la consulta 7 por peso máximo del producto

## Ejercicios 8.3

9 Explique por qué la siguiente consulta no da el resultado esperado:

```
SELECT shipperid, SUM(freight) AS totalfreight
FROM Sales.Orders
WHERE freight > 20000.00
GROUP BY shipperid;
```

10 Arregle la consulta anterior para que devuelva solo aquellas órdenes para las cuales el *total freight* es mayor a 20,000.



# Información Adicional

- GROUP BY

<http://msdn.microsoft.com/en-us/library/ms177673.aspx>

- HAVING

<http://msdn.microsoft.com/en-us/library/ms177673.aspx>



# Referencias

- Ben-Gan, I. Microsoft SQL Server 2012: T-SQL Fundamentals. Microsoft Press, O'Reilly Media, 2012.
- Ben-Gan, I., Sarka D. y Talmage, R. Querying Microsoft SQL Server 2012: Training Kit. Microsoft Press, O'Reilly Media, 2012.
- Elmasri R. y Navathe S. Fundamentos de Sistemas de Bases de Datos, 5ta ed. Pearson-Addison Wesley, 2007.
- Microsoft Virtual Academy. <http://www.microsoftvirtualacademy.com>