05 | Using Functions and Aggregating Data

Graeme Malcolm | Senior Content Developer, Microsoft
Geoff Allix | Principal Technologist, Content Master

Module Overview

- Introduction to Built-In Functions
- Scalar Functions
- Aggregate Functions
- Logical Functions
- Window Functions
- Grouping with GROUP BY
- Filtering with HAVING

Introduction to Built-In Functions

Function Category	Description
Scalar	Operate on a single row, return a single value
Logical	Scalar functions that compare multiple values to determine a single output
Aggregate	Take one or more input values, return a single summarizing value
Window	Operate on a window (set) of rows
Rowset	Return a virtual table that can be used subsequently in a Transact-SQL statement

Scalar Functions

- Operate on elements from a single row as inputs, return a single value as output
- Return a single (scalar) value
- Can be used like an expression in queries
- May be deterministic or non-deterministic

Scalar Function Categories

- Configuration
- Conversion
- Cursor
- Date and Time
- Mathematical
- Metadata
- Security
- String
- System
- System Statistical
- Text and Image

DEMO

Using Scalar Functions

Logical Functions

Output is determined by comparative logic

- ISNUMERIC

```
SELECT ISNUMERIC('101.99') AS Is_a_Number;
```

- IIF

```
SELECT productid, listprice, IIF(listprice > 50, 'high','low') AS PricePoint
FROM Production.Product;
```

- CHOOSE

```
SELECT ProductName, Color, Size,
        CHOOSE (ProductCategoryID, 'Bikes','Components','Clothing','Accessories') AS Category
FROM Production.Product;
```

DEMO

Using Logical Functions

Window Functions

- Functions applied to a window, or set of rows
- Include ranking, offset, aggregate and distribution functions

```
SELECT TOP(3) ProductID, Name, ListPrice,
      RANK() OVER(ORDER BY ListPrice DESC) AS RankByPrice
FROM Production.Product
ORDER BY RankByPrice;
```

ProductID	ProductName	UnitPrice	RankByPrice
8	Gizmo	263.50	1
29	Widget	123.79	2
9	Thingybob	97.00	3

DEMO

Using Window Functions

Aggregate Functions

- Functions that operate on sets, or rows of data
- Summarize input rows
- Without GROUP BY clause, all rows are arranged as one group

```
SELECT COUNT(*) AS OrderLines,
       SUM(OrderQty*UnitPrice) AS TotalSales
FROM   Sales.OrderDetail;
```

OrderLines	TotalSales
542	714002.9136

DEMO

Using Aggregate Functions

Grouping with GROUP BY

- GROUP BY creates groups for output rows, according to a unique combination of values specified in the GROUP BY clause
- GROUP BY calculates a summary value for aggregate functions in subsequent phases
- Detail rows are "lost" after GROUP BY clause is processed

```
SELECT CustomerID, COUNT(*) AS Orders
FROM Sales.SalesOrderHeader
GROUP BY CustomerID;
```

DEMO

Grouping with GROUP BY

Filtering with HAVING

- HAVING clause provides a search condition that each group must satisfy
- WHERE clause is processed before GROUP BY, HAVING clause is processed after GROUP BY

```
SELECT CustomerID, COUNT(*) AS Orders
FROM Sales.SalesOrderHeader
GROUP BY CustomerID
HAVING COUNT(*) > 10;
```

DEMO

Filtering with HAVING

Using Functions and Aggregating Data

- Introduction to Built-In Functions
- Scalar Functions
- Aggregate Functions
- Logical Functions
- Window Functions
- Grouping with GROUP BY
- Filtering with HAVING

- Lab: Using Functions and Aggregating Data

Microsoft