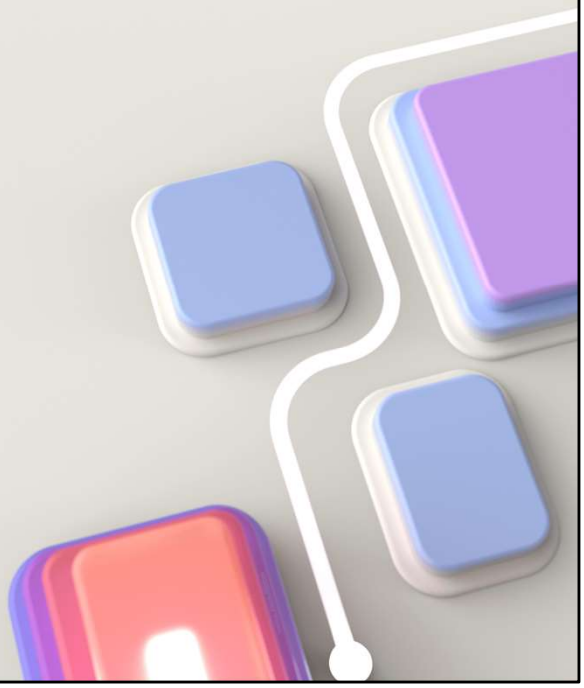
Microsoft

4: Using built-in functions

Agenda

- Getting started with scalar functions
- Grouping aggregated results

1: Getting started with scalar functions



Introduction to built-in functions

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

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

```
SELECT UPPER(ProductName) AS Product,
       ROUND(ListPrice, 0) AS ApproxPrice,
       YEAR(SaleStartDate) AS SoldSince
FROM Production.Product;
```

Scalar function categories

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

Logical functions

Output is determined by comparative logic

IIF

- Evaluate logical expression, return first value if true and second value if false

```
SELECT AddressType,
       IIF(AddressType = 'Main Office', 'Billing', 'Mailing') AS UseFor
FROM Sales.CustomerAddress;
```

CHOOSE

- Return value based ordinal position of expression in 1-based list

```
SELECT SalesOrderID, Status,
       CHOOSE(Status, 'Ordered', 'Shipped', 'Delivered')   AS OrderStatus
FROM Sales.SalesOrderHeader;
```

Ranking functions

Functions applied to a partition, or set of rows

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

ProductID	Name	ListPrice	RankByPrice
8	Gizmo	263.50	1
29	Widget	123.79	2
9	Thingybob	97.00	3

Ranking functions are an advanced technique. They're included here for completeness, but a detailed exploration is beyond the scope of this course.

Rowset functions

Return a rowset that can be used in a FROM clause

- OPENDATASOURCE – Get data from an object on a remote server
- OPENROWSET – Run an ad-hoc query on a remote server or file
- OPENQUERY – Get query results from a linked server
- OPENXML – Read elements and attributes from XML into a rowset
- OPENJSON – Read values from JSON objects into a rowset

```
SELECT a.*
FROM OPENROWSET('SQLNCLI',
     'Server=server1;Trusted_Connection=yes;',
     'SELECT Name, ListPrice
      FROM adventureworks.SalesLT.Product') AS a;
```


Rowset functions are generally only used in specialized scenarios.

Like ranking functions, they're included here for completeness, but a detailed exploration is beyond the scope of this course.

Aggregate functions

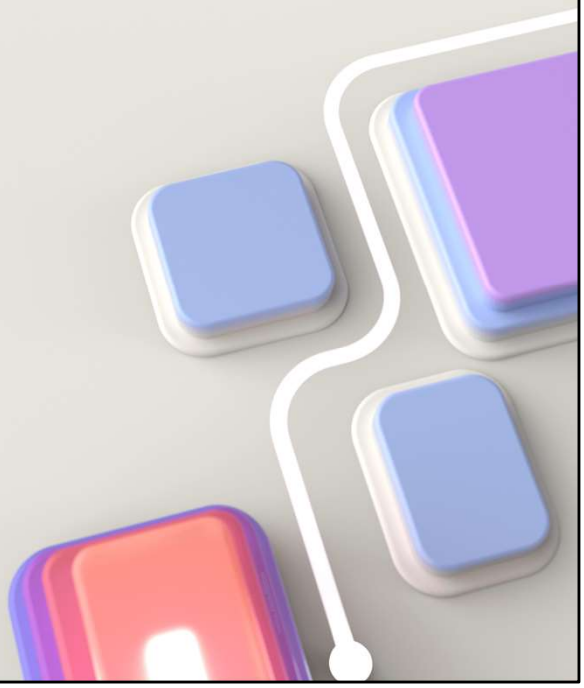
Functions that operate on sets, or rows of data

- 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

2: Grouping aggregated results



Grouping with GROUP BY

- GROUP BY creates groups for output rows, according to 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 not available after GROUP BY clause is processed


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

Filtering groups 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;
```

Lab: Using built-in functions

- Use scalar functions
- Use logical functions
- Use aggregate functions
- Group aggregated results with GROUP BY clause
- Filter groups with the HAVING clause

Review

1 **Which OrderState value does this query return for rows with a Status value of 2:**

```
SELECT OrderNo, CHOOSE(Status, 'Ordered', 'Shipped', 'Delivered') AS OrderState FROM
Sales.Order;
```

- ☑ Shipped
- ☐ Delivered
- ☐ NULL

2 **Which query returns the number of customers in each city?**

- ☐ `SELECT City, COUNT(*) AS CustomerCount FROM Sales.Customer;`
- ☑ `SELECT City, COUNT(*) AS CustomerCount FROM Sales.Customer GROUP BY City;`
- ☐ `SELECT City, COUNT(*) AS CustomerCount FROM Sales.Customer ORDER BY City;`

3 **Which query returns a row for each category with an average price over 10.00?**

- ☐ `SELECT Category, AVG(Price) FROM Store.Product WHERE AVG(Price) > 10.00;`
- ☐ `SELECT Category, AVG(Price) FROM Store.Product GROUP BY Category WHERE AVG(Price) > 10.00;`
- ☑ `SELECT Category, AVG(Price) FROM Store.Product GROUP BY Category HAVING AVG(Price) > 10.00;`

Use the slide animation to reveal the correct answers.

Microsoft