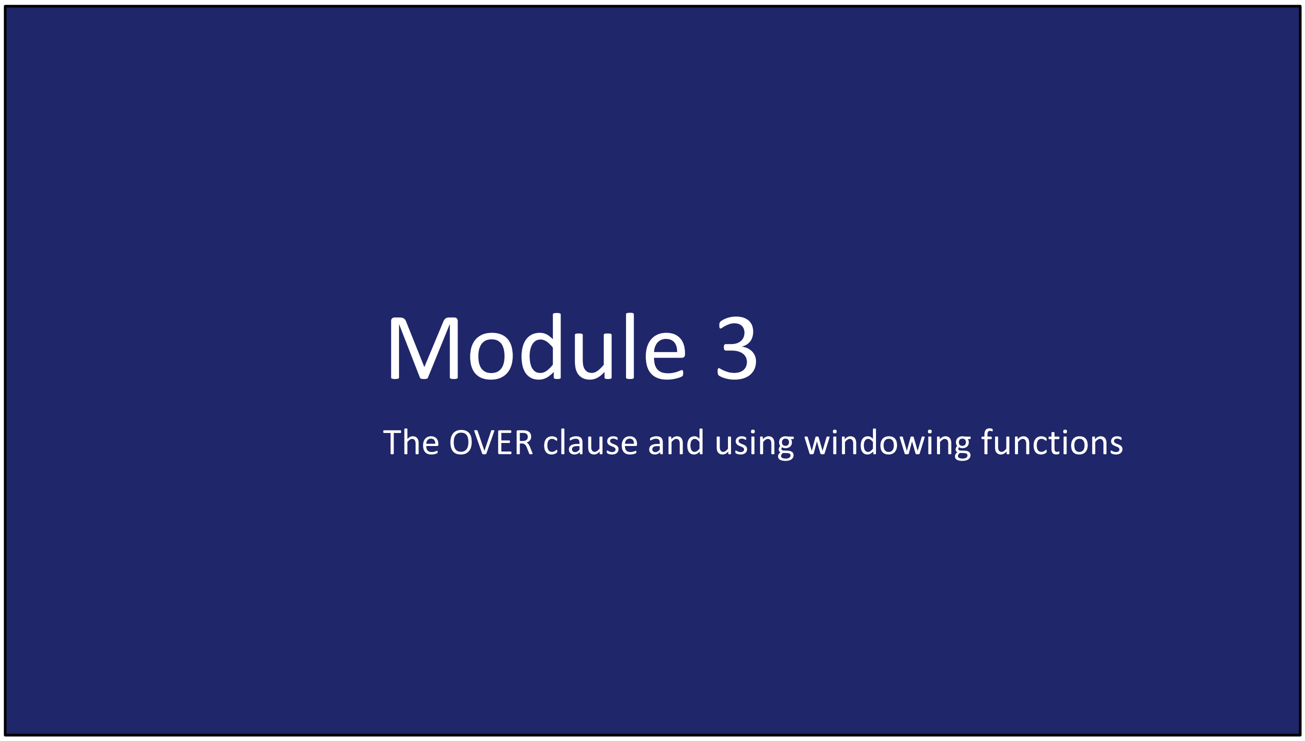
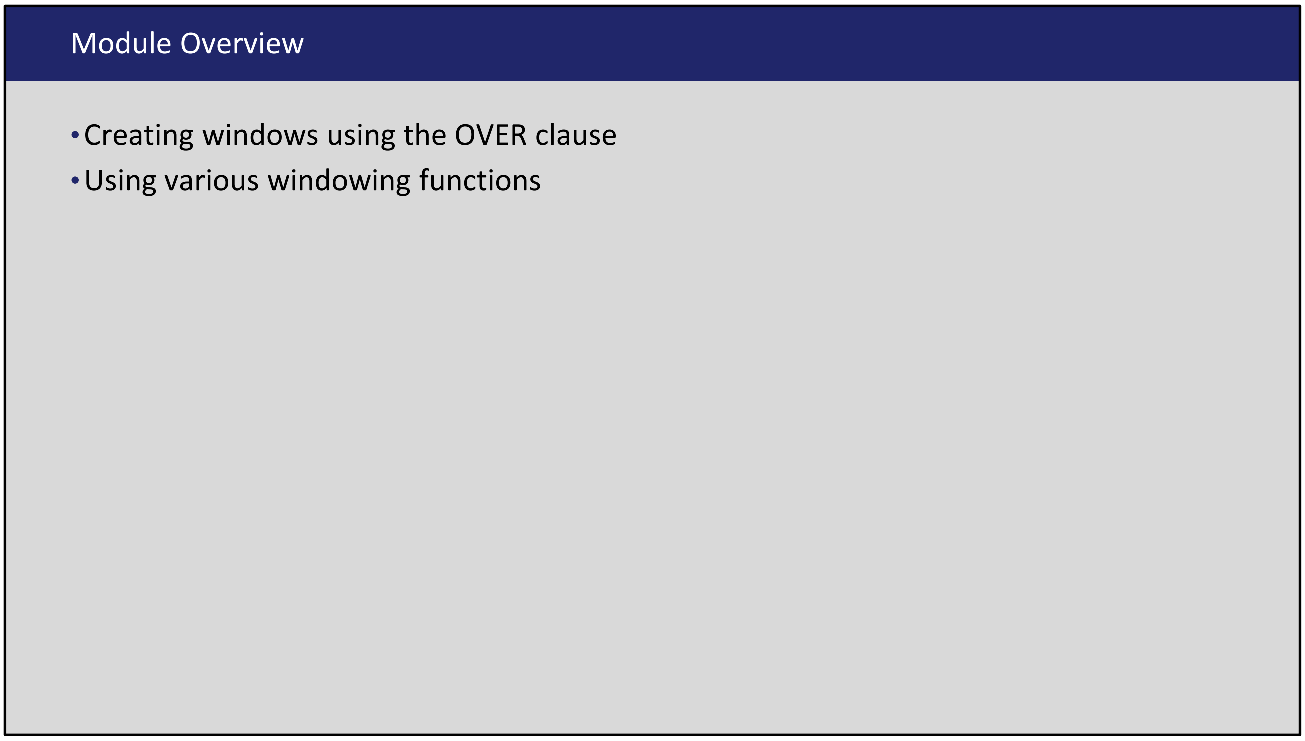


Module 3

The OVER clause and using windowing functions



Module Overview

- Creating windows using the OVER clause
- Using various windowing functions

Lesson: Creating windows using the OVER clause

- SQL Windowing
- Windowing Components
- Using OVER
- Partitioning Windows
- Ordering and Framing

SQL Windowing

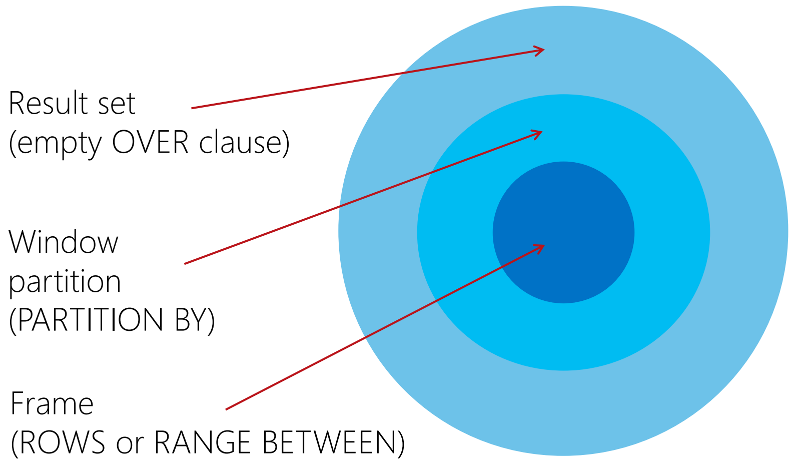
- Windows allow you to specify an order as part of a calculation, without regard to order of input or final output order
- Windows allow partitioning and framing of rows to support functions
- Window functions can simplify queries that need to find running totals, moving averages, or gaps in data

```
SELECT Category, Qty, Orderyear,
        SUM(Qty) OVER (
                PARTITION BY category
                ORDER BY orderyear
                ROWS BETWEEN UNBOUNDED PRECEDING
                AND CURRENT ROW) AS RunningQty
FROM Sales.CategoryQtyYear;
```

Introduction to T-SQL Window Functions
https://www.red-gate.com/simple-talk/databases/sql-server/t-sql-programming-sql-server/introduction-to-t-sql-window-functions/

Windowing Components

- Conceptual relationship between window elements:



Using OVER

- OVER defines a window, or set, of rows to be used by a window function, including any ordering
- With a specified window partition clause, the OVER clause restricts the set of rows to those with the same values in the partitioning elements
- By itself, OVER() is unrestricted and includes all rows
- Multiple OVER clauses can be used in a single query, each with its own partitioning and ordering, if needed

```
OVER ( [ <PARTITION BY clause> ]
       [ <ORDER BY clause> ]
       [ <ROWS or RANGE clause> ]
     )
```

SELECT - OVER Clause (Transact-SQL)
https://docs.microsoft.com/en-us/sql/t-sql/queries/select-over-clause-transact-sql

Understanding the OVER Clause in SQL Server
https://www.sqlservercentral.com/articles/understanding-the-over-clause

Partitioning Windows

- Partitioning limits a set to rows with the same value in the partitioning column
- Use PARTITION BY in the OVER() clause
- Without a PARTITION BY clause defined, OVER() creates a single partition of all rows

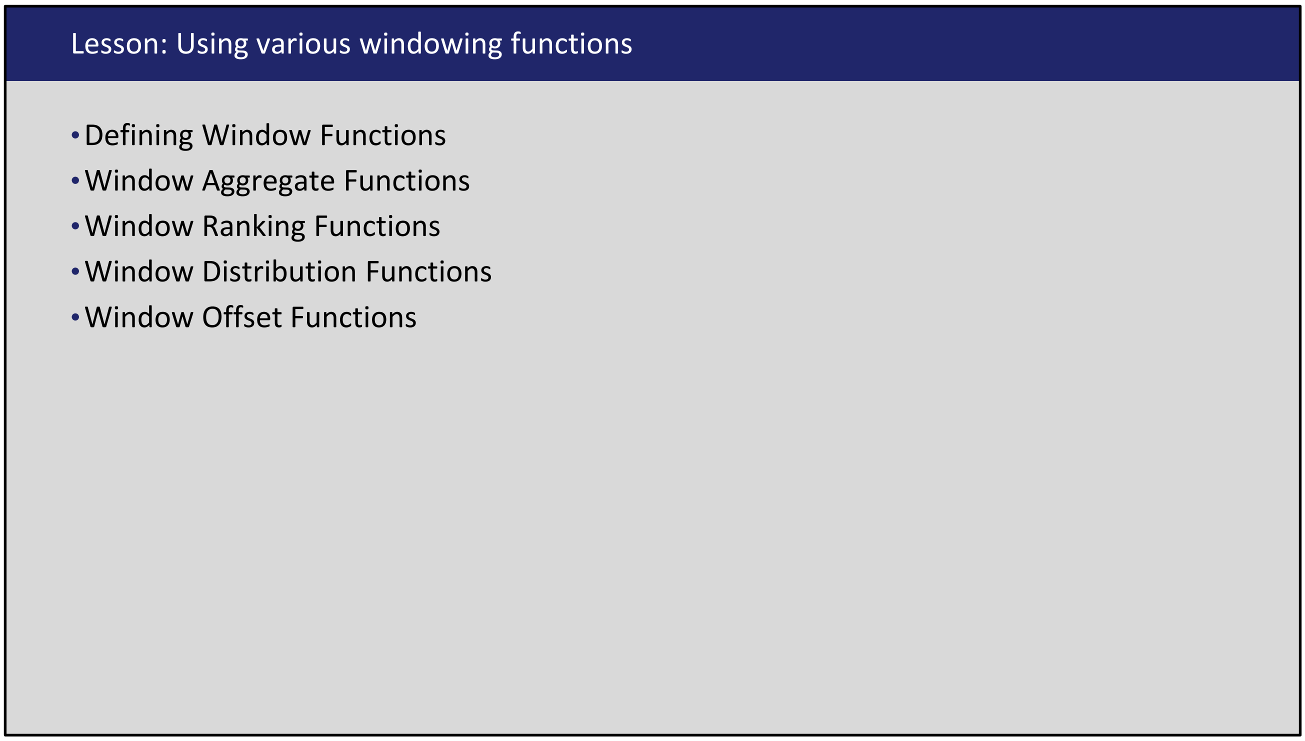
```
SELECT  custid, ordermonth, qty,
        SUM(qty) OVER(PARTITION BY custid)
        AS totalbycust
FROM Sales.CustOrders;
```

```
custid ordermonth              qty totalbycust
------ ----------------------- --- -----------
1      2007-08-01 00:00:00.000 38  174
1      2007-10-01 00:00:00.000 41  174
2      2006-09-01 00:00:00.000 6   63
2      2007-08-01 00:00:00.000 18  63
3      2006-11-01 00:00:00.000 24  359
3      2007-04-01 00:00:00.000 30  359
```

Ordering and Framing

- Window framing allows you to set start and end boundaries within a window partition
  - UNBOUNDED means go all the way to boundary in direction specified by PRECEDING or FOLLOWING (start or end)
  - CURRENT ROW indicates start or end at current row in partition
  - ROWS BETWEEN allows you to define a range of rows between two points
- Window ordering provides a context to the frame
  - Sorting by an attribute enables meaningful position of a boundary
  - Without ordering, "start at first row" is not useful because a set has no order

Demo OVER



Lesson: Using various windowing functions

- Defining Window Functions
- Window Aggregate Functions
- Window Ranking Functions
- Window Distribution Functions
- Window Offset Functions

Defining Window Functions

- A window function is a function applied to a window, or set, of rows
- Window functions include aggregate, ranking, distribution, and offset functions
- Window functions returns data based on data created by OVER()

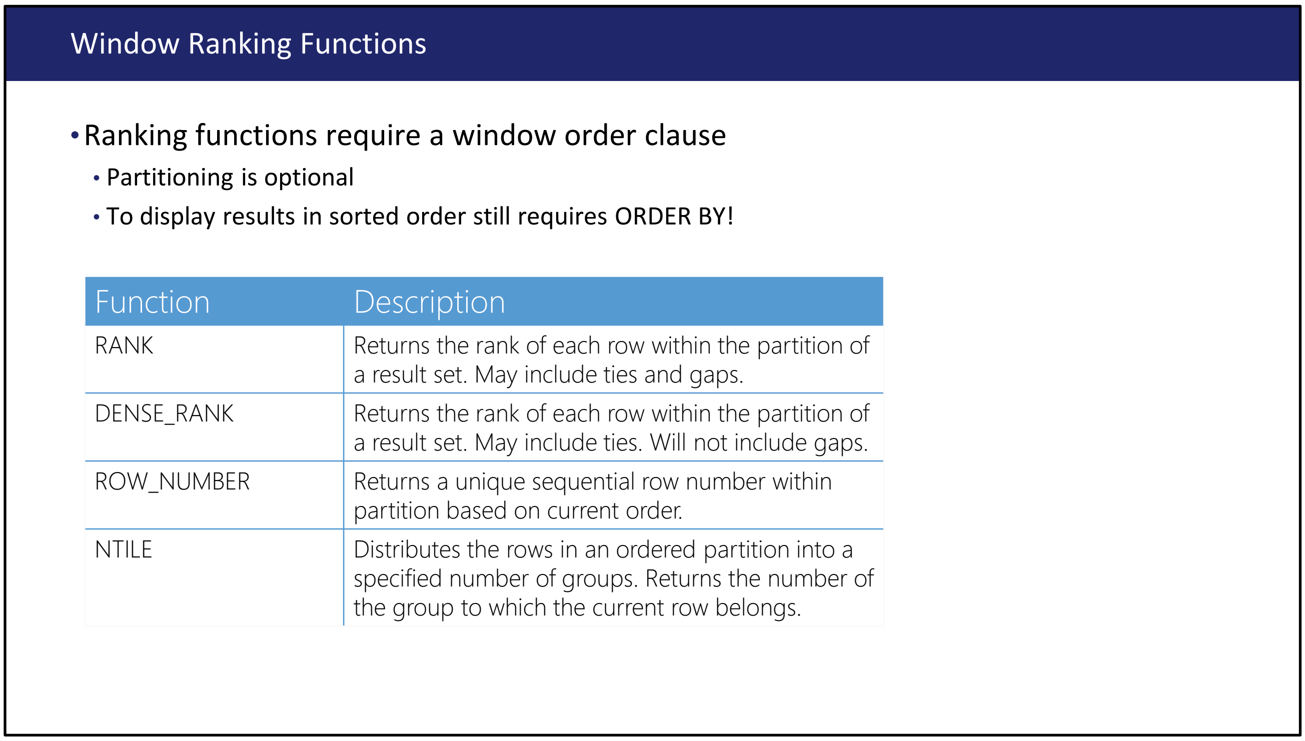
```
SELECT  productid, productname, unitprice,
        RANK() OVER(ORDER BY unitprice DESC)
        AS pricerank
FROM Production.Products
ORDER BY pricerank;
```

SQL Server Window Functions
https://www.sqlservertutorial.net/sql-server-window-functions/

Window Aggregate Functions

- Similar to grouped aggregate functions
  - SUM, MIN, MAX, and so on
- Applied to windows defined by OVER clause
- Window aggregate functions support partitioning, ordering, and framing

```
SELECT  custid, ordermonth, qty,
     SUM(qty) OVER(PARTITION BY custid)
     AS totalpercust
FROM Sales.CustOrders;
```



Window Ranking Functions

- Ranking functions require a window order clause
  - Partitioning is optional
  - To display results in sorted order still requires ORDER BY!

Function	Description
RANK	Returns the rank of each row within the partition of a result set. May include ties and gaps.
DENSE_RANK	Returns the rank of each row within the partition of a result set. May include ties. Will not include gaps.
ROW_NUMBER	Returns a unique sequential row number within partition based on current order.
NTILE	Distributes the rows in an ordered partition into a specified number of groups. Returns the number of the group to which the current row belongs.

Ranking Functions (Transact-SQL)
https://docs.microsoft.com/en-us/sql/t-sql/functions/ranking-functions-transact-sql

Overview of SQL RANK functions
https://www.sqlshack.com/overview-of-sql-rank-functions/

Window Distribution Functions

- Window distribution functions perform statistical analysis on data, and require a window order clause
- Rank distribution performed with PERCENT_RANK and CUME_DIST
- Inverse distribution performed with PERCENTILE_CONT and PERCENTILE_DISC

Analytic Functions (Transact-SQL)
https://docs.microsoft.com/en-us/sql/t-sql/functions/analytic-functions-transact-sql

Window Offset Functions

- Window offset functions allow comparisons between rows in a set without the need for a self-join
- Offset functions operate on a position relative to the current row, or to the start or end of the window frame

Function	Description
LAG	Returns an expression from a previous row that is a defined offset from the current row. Returns NULL if no row at specified position.
LEAD	Returns an expression from a later row that is a defined offset from the current row. Returns NULL if no row at specified position.
FIRST_VALUE	Returns the first value in the current window frame. Requires window ordering to be meaningful.
LAST_VALUE	Returns the last value in the current window frame. Requires window ordering to be meaningful.

Analytic Functions (Transact-SQL)
https://docs.microsoft.com/en-us/sql/t-sql/functions/analytic-functions-transact-sql

Example: LEAD Offset Window Function

```
SELECT employee, orderyear ,totalsales AS currsales,
  LEAD (totalsales, 1,0) OVER (PARTITION BY employee
         ORDER BY orderyear) AS nextsales
FROM Sales.OrdersByEmployeeYear
ORDER BY employee, orderyear;
```

```
employee orderyear currsales nextsales
-------- --------- --------- ---------
1        2006      38789.00  97533.58
1        2007      97533.58  65821.13
1        2008      65821.13  0.00
2        2006      22834.70  74958.60
2        2007      74958.60  79955.96
2        2008      79955.96  0.00
3        2006      19231.80  111788.61
3        2007      111788.61 82030.89
```

Demo Window functions

V1 Lab 3: Using Window Ranking, Offset, and Aggregate Functions

- ***Note carefully the folder name in the lab instructions!!!***
- Exercise 1: Writing Queries That Use Ranking Functions
- Exercise 2: Writing Queries That Use Offset Functions
- Exercise 3: Writing Queries That Use Window Aggregate Functions

Estimated Time: 60 minutes

V2 Lab 3: Using Window Ranking, Offset, and Aggregate Functions

- Ex 1. Using ranking functions
- Ex 2. Generate running totals

Estimated Time: 60 minutes