Introduction to T-SQL Window Functions

Kathi Kellenberger

Redgate Software


11.560
1.286
2.286
0.156
N/A
1.2404
0.234
N/A
2.344
0.1204
0.1902
0.12%
+ 1.51%
+ 0.11%
+ 0.15%
- 1.93%
- 1.92%

What are T-SQL Window Functions?

The details…

- Not OS, actually T-SQL functions based on ANSI-SQL Standards
- The function performs over a SET or “window” of the results
- Window functions can *only* go in the SELECT and ORDER BY
- The FROM, WHERE, GROUP BY and HAVING clauses operate *before* the window functions
- Window functions always* have an OVER clause

The OVER clause

How rows line up with *ORDER BY*

Divide up rows with

PARTITION BY

OrderID	CustID	OrderDate	TotalAmt
1	102	2019-01-09	$1050
2	208	2019-01-09	$750
3	102	2019-02-07	$30
4	102	2019-03-01	$500
5	208	2019-03-02	$672
6	102	2019-03-03	$296
7	103	2019-04-01	$2070
8	202	2019-04-10	$99
9	202	2019-04-11	$70

OVER(ORDER BY TotalAmt)

OrderID	CustID	OrderDate	TotalAmt
3	102	2/7/2019	$30
9	202	4/11/2019	$70
8	202	4/10/2019	$99
6	102	3/3/2019	$296
4	102	3/1/2019	$500
5	208	3/2/2019	$672
2	208	1/9/2019	$750
1	102	1/9/2019	$1,050
7	103	4/1/2019	$2,070

OVER(PARTITION BY CustID ORDER BY TotalAmt)

OrderID	CustID	OrderDate	TotalAmt	
3	102	2/7/2019	$30	← Window
6	102	3/3/2019	$296	
4	102	3/1/2019	$500	
1	102	1/9/2019	$1,050	
7	103	4/1/2019	$2,070	← Window
9	202	4/11/2019	$70	← Window
8	202	4/10/2019	$99	
5	208	3/2/2019	$672	← Window
2	208	1/9/2019	$750	

Ranking Functions

ROW_NUMBER

RANK

DENSE_RANK

NTILE

ROW_NUMBER()
OVER(ORDER BY TotalAmt)

OrderID	CustID	OrderDate	TotalAmt	RowNumber
3	102	2/7/2019	$30	1
9	202	4/11/2019	$70	2
8	202	4/10/2019	$99	3
6	102	3/3/2019	$296	4
4	102	3/1/2019	$500	5
5	208	3/2/2019	$672	6
2	208	1/9/2019	$750	7
1	102	1/9/2019	$1,050	8
7	103	4/1/2019	$2,070	9

ROW_NUMBER()
OVER(PARTITION BY CustID ORDER BY TotalAmt)

OrderID	CustID	OrderDate	TotalAmt	RowNumber
3	102	2/7/2019	$30	1
6	102	3/3/2019	$296	2
4	102	3/1/2019	$500	3
1	102	1/9/2019	$1,050	4
7	103	4/1/2019	$2,070	1
9	202	4/11/2019	$70	1
8	202	4/10/2019	$99	2
5	208	3/2/2019	$672	1
2	208	1/9/2019	$750	2

Window Aggregates

Your favorite aggregate functions with no GROUP BY

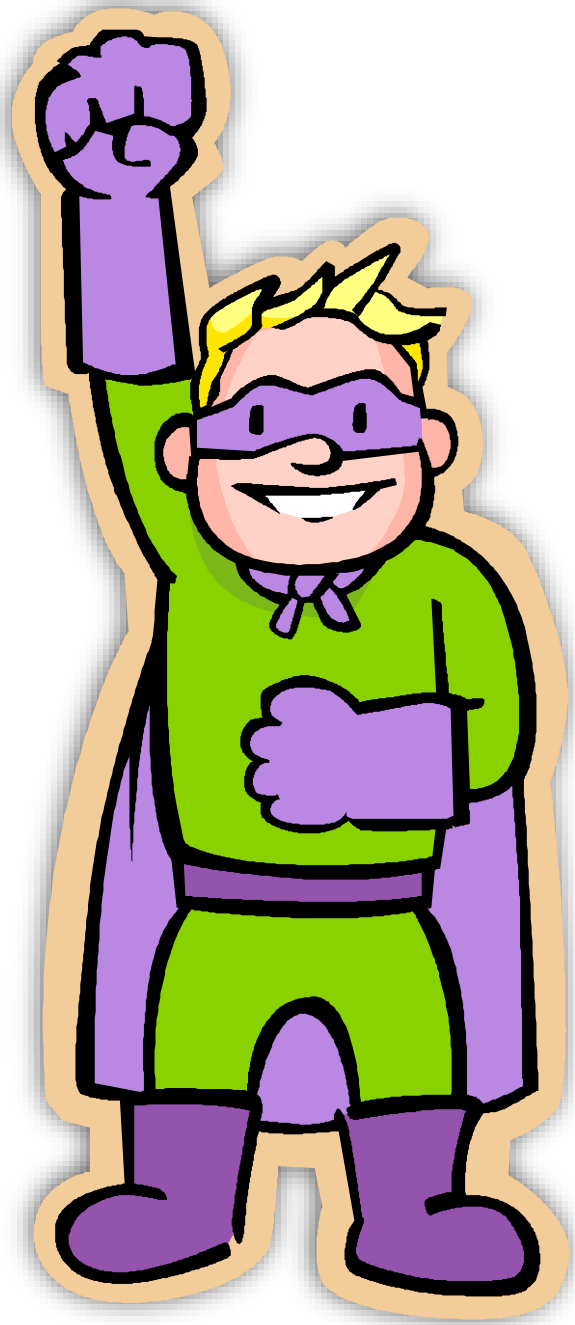
Calculate over the partition

No ORDER BY!

SUM(TotalAmt) OVER()

SUM(TotalAmt) OVER(PARTITION BY CustID)

OrderID	CustID	OrderDate	TotalAmt	GrandTotal	SubTotal
1	102	2019-01-09	$1050	$5,537	$1,876
2	208	2019-01-09	$750	$5,537	$1,422
3	102	2019-02-07	$30	$5,537	$1,876
4	102	2019-03-01	$500	$5,537	$1,876
5	208	2019-03-02	$672	$5,537	$1,422
6	102	2019-03-03	$296	$5,537	$1,876
7	103	2019-04-01	$2070	$5,537	$2,070
8	202	2019-04-10	$99	$5,537	$796
9	202	2019-04-11	$70	$5,537	$796



2012 Enhancements

Accumulating Aggregates

Add *ORDER BY* to OVER clause

SUM(TotalAmt) OVER(ORDER BY OrderID)

SUM(TotalAmt)
OVER(PARTITION BY CustID ORDER BY OrderID)

OrderID	CustID	OrderDate	TotalAmt	RunningTotal	CustRunningTotal
1	102	2019-01-09	$1050	$1,050	$1,050
2	208	2019-01-09	$750	$1,800	$750
3	102	2019-02-07	$30	$1,830	$1,080
4	102	2019-03-01	$500	$2,330	$1,580
5	208	2019-03-02	$672	$3,002	$1,422
6	102	2019-03-03	$296	$3,298	$1,876
7	103	2019-04-01	$2070	$5,368	$2,070
8	202	2019-04-10	$99	$5,467	$99
9	202	2019-04-11	$70	$5,537	$167

Framing

Further defines the frame

Each row can have its own window

Only certain function types

Term	Meaning
ROWS	Positional operator used to define the frame
RANGE	Logical operator used to define the frame The DEFAULT operator
UNBOUNDED PRECEDING	The first row of the partition
UNBOUNDED FOLLOWING	The last row of the partition
CURRENT ROW	The row where the calculation is being performed

ROWS BETWEEN UNBOUNDED PRECEDING AND CURRENT ROW

The partition consists of row 1

ROWS BETWEEN UNBOUNDED PRECEDING AND CURRENT ROW

The window consists of rows 1 and 2

ROWS BETWEEN UNBOUNDED PRECEDING AND CURRENT ROW

The window consists of rows 1 to 3

ROWS BETWEEN UNBOUNDED PRECEDING AND CURRENT ROW

The window consists of rows 1 to 4

ROWS BETWEEN UNBOUNDED FOLLOWING AND CURRENT ROW

The window consists of rows 1 to 15

ROWS BETWEEN UNBOUNDED FOLLOWING AND CURRENT ROW

The window consists of rows 2 to 15

ROWS BETWEEN UNBOUNDED FOLLOWING AND CURRENT ROW

The window consists of rows 3 to 15

ROWS BETWEEN UNBOUNDED FOLLOWING AND CURRENT ROW

The window consists of rows 4 to 15

ROWS BETWEEN 2 PRECEDING AND CURRENT ROW

The window consists of row 1

ROWS BETWEEN 2 PRECEDING AND CURRENT ROW

The window consists of rows 1 and 2

ROWS BETWEEN 2 PRECEDING AND CURRENT ROW

The window consists of rows 1 to 3

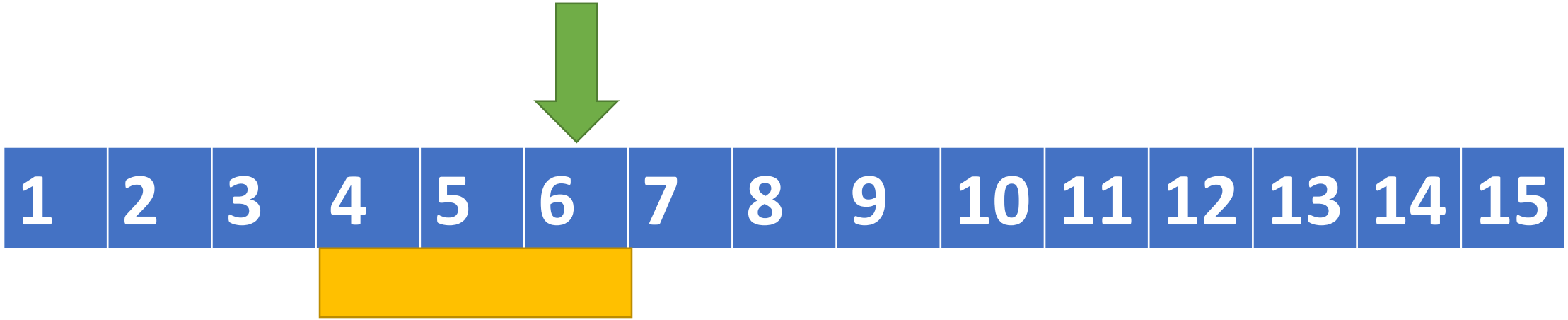
ROWS BETWEEN 2 PRECEDING AND CURRENT ROW

The window consists of rows 2 to 4

ROWS BETWEEN 2 PRECEDING AND CURRENT ROW

The partition consists of rows 3 to 5

ROWS BETWEEN 2 PRECEDING AND CURRENT ROW



The partition consists of rows 4 to 6

Offset Functions

LAG
LEAD
FIRST_VALUE
LAST_VALUE

ORDER BY required

LAG(TotalAmt) OVER(ORDER BY OrderID)
LEAD(TotalAmt) OVER(ORDER BY OrderID)

OrderID	CustID	OrderDate	TotalAmt	LAGAmt	LEADAmt
1	102	2019-01-09	$1050	NULL	$750
2	208	2019-01-09	$750	$1050	$30
3	102	2019-02-07	$30	$750	$500
4	102	2019-03-01	$500	$30	$672
5	208	2019-03-02	$672	$500	$296
6	102	2019-03-03	$296	$672	$2070
7	103	2019-04-01	$2070	$296	$99
8	202	2019-04-10	$99	$2070	$70
9	202	2019-04-11	$70	$99	NULL

LAG(TotalAmt) OVER(PARTITION BY CustID ORDER BY OrderID)

OrderID	CustID	OrderDate	TotalAmt	LAGAmt
1	102	1/9/2019	$1,050	NULL
3	102	2/7/2019	$30	$1,050
4	102	3/1/2019	$500	$30
6	102	3/3/2019	$296	$500
7	103	4/1/2019	$2,070	NULL
8	202	4/10/2019	$99	NULL
9	202	4/11/2019	$70	$99
2	208	1/9/2019	$750	NULL
5	208	3/2/2019	$672	$750

FIRST_VALUE(TotalAmt) OVER(ORDER BY OrderID)
LAST_VALUE(TotalAmt)
OVER(ORDER BY OrderID ROWS BETWEEN CURRENT ROW AND UNBOUNDED FOLLOWING)

OrderID	CustID	OrderDate	TotalAmt	FIRST	LAST
1	102	2019-01-09	$1050	$1050	$70
2	208	2019-01-09	$750	$1050	$70
3	102	2019-02-07	$30	$1050	$70
4	102	2019-03-01	$500	$1050	$70
5	208	2019-03-02	$672	$1050	$70
6	102	2019-03-03	$296	$1050	$70
7	103	2019-04-01	$2070	$1050	$70
8	202	2019-04-10	$99	$1050	$70
9	202	2019-04-11	$70	$1050	$70

FIRST_VALUE(TotalAmt)
OVER(PARTITION BY CustID ORDER BY OrderID)
LAST_VALUE(TotalAmt)
OVER(PARTITION BY CustID ORDER BY OrderID
ROWS BETWEEN CURRENT ROW AND UNBOUNDED PRECEDING)

OrderID	CustID	OrderDate	TotalAmt	FIRST	LAST
1	102	2019-01-09	$1050	$1050	$296
2	208	2019-01-09	$750	$750	$672
3	102	2019-02-07	$30	$1050	$296
4	102	2019-03-01	$500	$1050	$296
5	208	2019-03-02	$672	$750	$672
6	102	2019-03-03	$296	$1050	$296
7	103	2019-04-01	$2070	$2070	$2070
8	202	2019-04-10	$99	$99	$70
9	202	2019-04-11	$70	$99	$70

Exception to the OVER clause

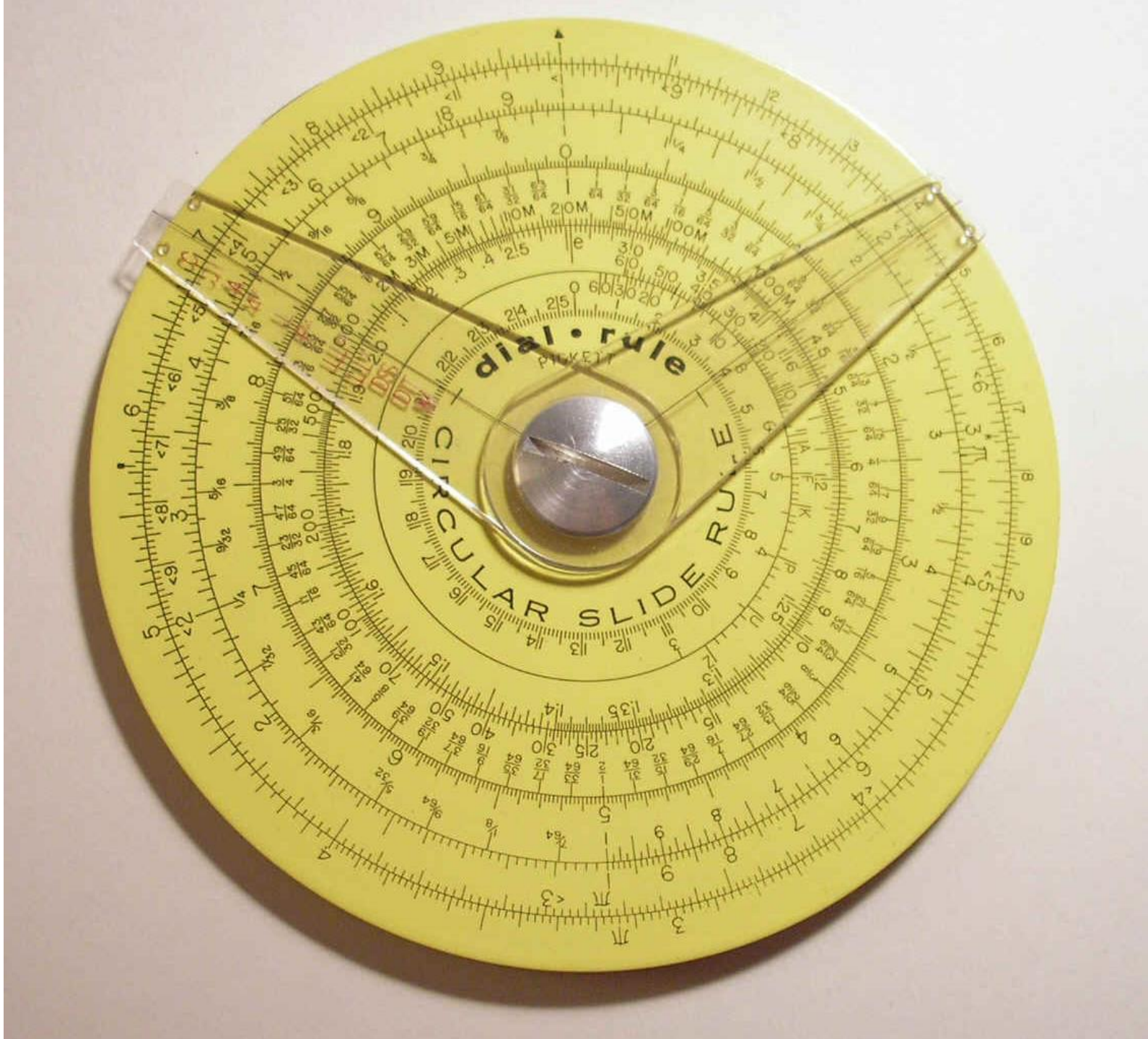
```
SELECT
   P1.MemberID,
   P1.MemberName,
   STRING_AGG(P2.MemberName,
     '->') WITHIN GROUP (GRAPH PATH) AS [MemberName],
   LAST_VALUE(P2.MemberName) WITHIN GROUP (GRAPH PATH)
     AS FinalMemberName,
   COUNT(P2.MemberId) WITHIN GROUP (GRAPH PATH) AS Levels
 FROM
   ForumMembers P1,
   ForumMembers FOR PATH as P2,
   Likes FOR PATH as IPO
 WHERE MATCH(SHORTEST_PATH(P1(-(IPO)->P2)+));
```

https://www.red-gate.com/simple-talk/sql/sql-development/sql-server-2019-graph-database-and-shortest_path/

Statistical Functions

PERCENT_RANK
CUME_DIST
Ranks the partition

PERCENTILE_DISC
PERCENTILE_CONT
Given a rank, find the value



PERCENT_RANK() OVER(ORDER BY TotalAmt)

CUME_DIST() OVER(ORDER BY TotalAmt)

OrderID	CustID	OrderDate	TotalAmt	PercntRank	CumeDist
3	102	2/7/2019	$30	0	0.1111111
9	202	4/11/2019	$70	0.125	0.2222222
8	202	4/10/2019	$99	0.25	0.3333333
6	102	3/3/2019	$296	0.375	0.4444444
4	102	3/1/2019	$500	0.5	0.5555555
5	208	3/2/2019	$672	0.625	0.6666666
2	208	1/9/2019	$750	0.75	0.7777777
1	102	1/9/2019	$1,050	0.875	0.8888888
7	103	4/1/2019	$2,070	1	1

PERCENTILE_CONT(0.5) WITHIN GROUP(ORDER BY TotalAmt) OVER()

PERCENTILE_DISC(0.5) WITHIN GROUP(ORDER BY TotalAmt) OVER()

OrderID	CustID	OrderDate	TotalAmt	TheMedian	NotTheMedian
3	102	2/7/2019	$30	500	500
9	202	4/11/2019	$70	500	500
8	202	4/10/2019	$99	500	500
6	102	3/3/2019	$296	500	500
4	102	3/1/2019	$500	500	500
5	208	3/2/2019	$672	500	500
2	208	1/9/2019	$750	500	500
1	102	1/9/2019	$1,050	500	500
7	103	4/1/2019	$2,070	500	500

Resources

- Expert T-SQL Window Functions
- High-Performance T-SQL Using Window Functions
  By Itzik Ben-Gan
- Pluralsight Course
- Auntkathisql.com
- Simple Talk articles