JOBSHEET

PRAKTIKUM BASIS DATA LANJUT

Jurusan Teknologi Informasi

POLITEKNIK NEGERI MALANG

Week 7

SQL SERVER - Window Ranking, Offset, Fungsi Agregat

Information Technology Department, State Polytechnic of Malang **Jobsheet Week -7 : Window Ranking, Offset, Aggregate Function Advanced Database Course (BDL)**
Supervisor: Database Teaching Team

Topics

1. Create a Window with OVER
2. Conducting Function exploration Windows

Objective

1. Students understand how to explain the T-SQL components used to define windows and the relationship between the two. the
2. Students understand how to write queries using the OVER clause with *partitioning* , *ordering* , and *framing* to define window
3. Students understand how to write queries using window functions. aggregate
4. Students understand how to write queries using window functions. ranking
5. Students understand how to write queries using window functions. offset

General Instructions

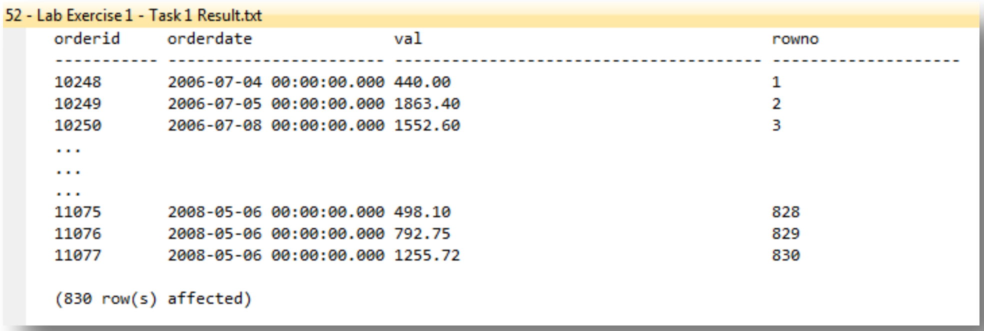
1. Follow the steps in the practical sections in the order given. given.
2. You can use SQL Server 2012 Standard Edition to try the practicum on this jobsheet. Adjust it to your computer's condition. You.
3. Answer all questions marked [Question-X] found in the specific steps in each section. practicum.
4. In each step of the practicum there is an explanation that will help you answer the questions in instruction number 3, so read and do all the practicum parts in the jobsheet. This.
5. Write the answers to the questions in the instructions number 3 in a report that is done using a word processing application (Word, OpenOffice, or other similar). Export as a **PDF file** with the name format as following:
   - **BDL_09_Your_Full_Name_Class** .pdf
   - Example:
     - **BDL_09_TI2U_Mukiyo** .pdf
   - Pay close attention to the format its naming.
   - Collect the PDF files as a practical report to the lecturer. guardian.
   - In addition to the file name, also include your identity on the first page of the report. the.

Lab – Part 1: Writing Queries Using the RANKING Function

Step	Information
1	Scenario : The sales department wants to determine the order based on the value of each customer. To do this, it is necessary to report using the RANK function (including a calculation result column that adds a calculation result column to display the row number with the SELECT clause). To do the experiment in this practicum part 1, first log in to SQL Server Management Studio (SSMS). Make sure the database is connected to “TSQL”.

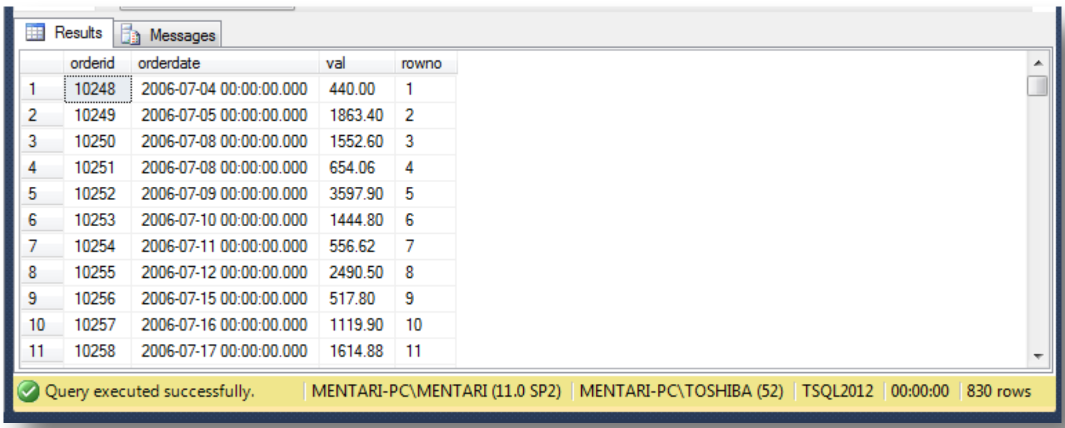
2

[Question-1] Write a SELECT statement to retrieve the orderid, orderdate, and val columns and a calculated column named rowno from the Sales.OrderValues view! Use the ROW_NUMBER function to return the rowno, sort the row numbers by the orderdate column!



52 - Lab Exercise 1 - Task 1 Result.txt

```
orderid    orderdate               val                                     rowno
---------- ----------------------- --------------------------------------- --------------------
10248      2006-07-04 00:00:00.000 440.00                                  1
10249      2006-07-05 00:00:00.000 1863.40                                 2
10250      2006-07-08 00:00:00.000 1552.60                                 3
...
...
...
11075      2008-05-06 00:00:00.000 498.10                                  828
11076      2008-05-06 00:00:00.000 792.75                                  829
11077      2008-05-06 00:00:00.000 1255.72                                 830

(830 row(s) affected)
```

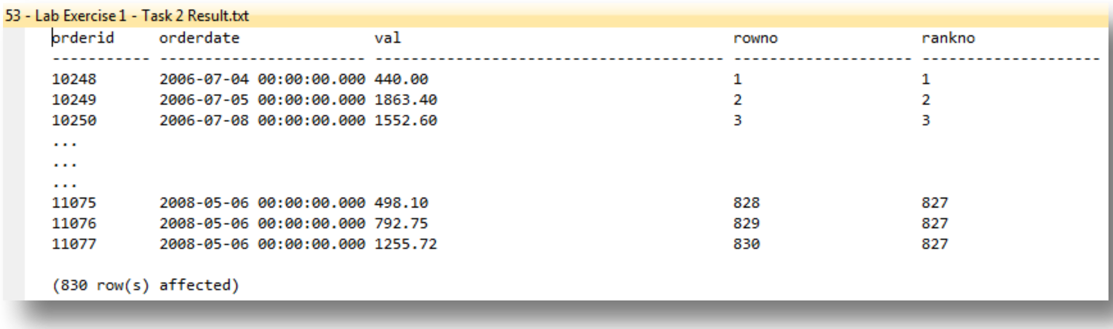


	orderid	orderdate	val	rowno
1	10248	2006-07-04 00:00:00.000	440.00	1
2	10249	2006-07-05 00:00:00.000	1863.40	2
3	10250	2006-07-08 00:00:00.000	1552.60	3
4	10251	2006-07-08 00:00:00.000	654.06	4
5	10252	2006-07-09 00:00:00.000	3597.90	5
6	10253	2006-07-10 00:00:00.000	1444.80	6
7	10254	2006-07-11 00:00:00.000	556.62	7
8	10255	2006-07-12 00:00:00.000	2490.50	8
9	10256	2006-07-15 00:00:00.000	517.80	9
10	10257	2006-07-16 00:00:00.000	1119.90	10
11	10258	2006-07-17 00:00:00.000	1614.88	11

```
Query executed successfully.  | MENTARI-PC\MENTARI (11.0 SP2) | MENTARI-PC\TOSHIBA (52) | TSQL2012 | 00:00:00 | 830 rows
```

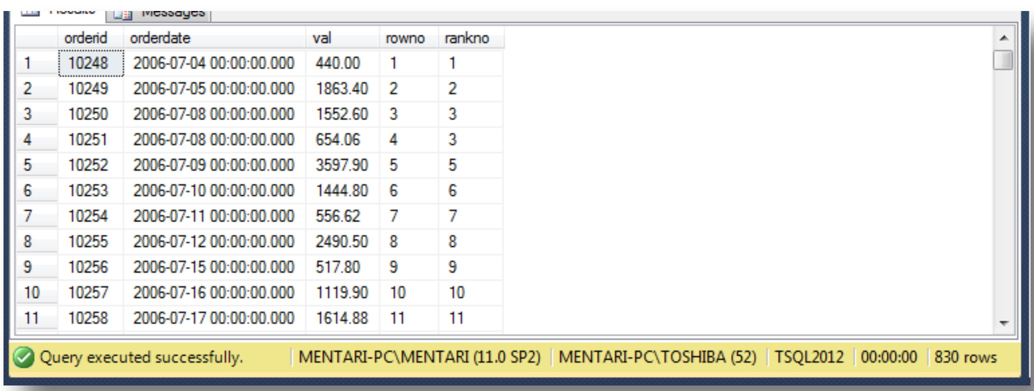
3

[Question-2] Copy the T-SQL in question no. 1. Then modify it by inserting an additional column named rankno. To create rankno, use the RANK function with the ranking order based on the orderdate column!



53 - Lab Exercise 1 - Task 2 Result.txt

```
orderid    orderdate               val                                     rowno                rankno
---------- ----------------------- --------------------------------------- -------------------- --------------------
10248      2006-07-04 00:00:00.000 440.00                                  1                    1
10249      2006-07-05 00:00:00.000 1863.40                                 2                    2
10250      2006-07-08 00:00:00.000 1552.60                                 3                    3
...
...
...
11075      2008-05-06 00:00:00.000 498.10                                  828                  827
11076      2008-05-06 00:00:00.000 792.75                                  829                  827
11077      2008-05-06 00:00:00.000 1255.72                                 830                  827

(830 row(s) affected)
```



	orderid	orderdate	val	rowno	rankno
1	10248	2006-07-04 00:00:00.000	440.00	1	1
2	10249	2006-07-05 00:00:00.000	1863.40	2	2
3	10250	2006-07-08 00:00:00.000	1552.60	3	3
4	10251	2006-07-08 00:00:00.000	654.06	4	3
5	10252	2006-07-09 00:00:00.000	3597.90	5	5
6	10253	2006-07-10 00:00:00.000	1444.80	6	6
7	10254	2006-07-11 00:00:00.000	556.62	7	7
8	10255	2006-07-12 00:00:00.000	2490.50	8	8
9	10256	2006-07-15 00:00:00.000	517.80	9	9
10	10257	2006-07-16 00:00:00.000	1119.90	10	10
11	10258	2006-07-17 00:00:00.000	1614.88	11	11

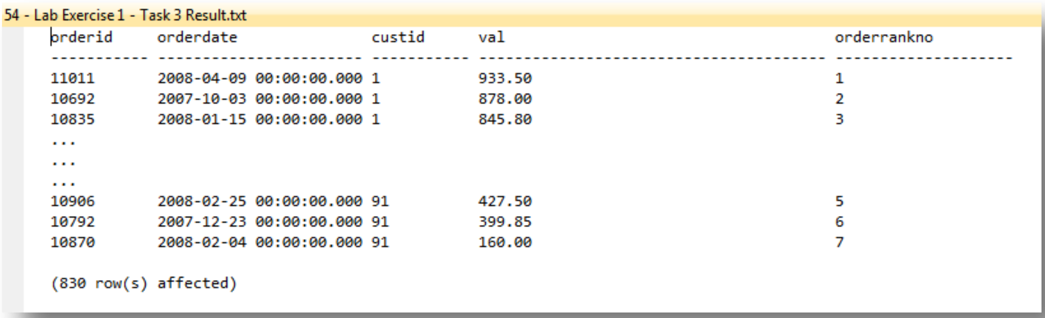
Query executed successfully. | MENTARI-PC\MENTARI (11.0 SP2) | MENTARI-PC\TOSHIBA (52) | TSQL2012 | 00:00:00 | 830 rows

4

[Question-3] What is the difference between the RANK function and the ROW_NUMBER function?

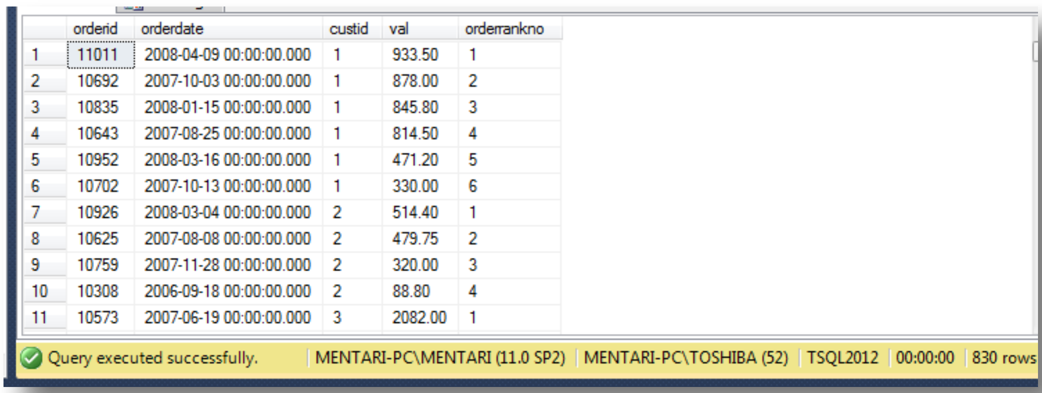
5

[Question-4] Write a SELECT statement to retrieve the orderid, orderdate, custid, and val columns and calculate a column named orderrankno from the Sales.OrderValues view. The orderrankno column should display the ranking per customer independently, based on the ordering of val in descending order!



54 - Lab Exercise 1 - Task 3 Result.txt

```
orderid     orderdate               custid      val                                        orderrankno
----------- ----------------------- ----------- ------------------------------------------ --------------------
11011       2008-04-09 00:00:00.000 1           933.50                                     1
10692       2007-10-03 00:00:00.000 1           878.00                                     2
10835       2008-01-15 00:00:00.000 1           845.80                                     3
...
...
...
10906       2008-02-25 00:00:00.000 91          427.50                                     5
10792       2007-12-23 00:00:00.000 91          399.85                                     6
10870       2008-02-04 00:00:00.000 91          160.00                                     7

(830 row(s) affected)
```



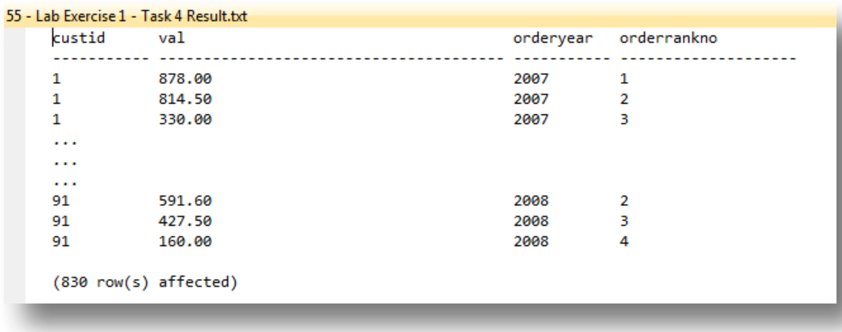
	orderid	orderdate	custid	val	orderrankno
1	11011	2008-04-09 00:00:00.000	1	933.50	1
2	10692	2007-10-03 00:00:00.000	1	878.00	2
3	10835	2008-01-15 00:00:00.000	1	845.80	3
4	10643	2007-08-25 00:00:00.000	1	814.50	4
5	10952	2008-03-16 00:00:00.000	1	471.20	5
6	10702	2007-10-13 00:00:00.000	1	330.00	6
7	10926	2008-03-04 00:00:00.000	2	514.40	1
8	10625	2007-08-08 00:00:00.000	2	479.75	2
9	10759	2007-11-28 00:00:00.000	2	320.00	3
10	10308	2006-09-18 00:00:00.000	2	88.80	4
11	10573	2007-06-19 00:00:00.000	3	2082.00	1

Query executed successfully. | MENTARI-PC\MENTARI (11.0 SP2) | MENTARI-PC\TOSHIBA (52) | TSQL2012 | 00:00:00 | 830 rows

6

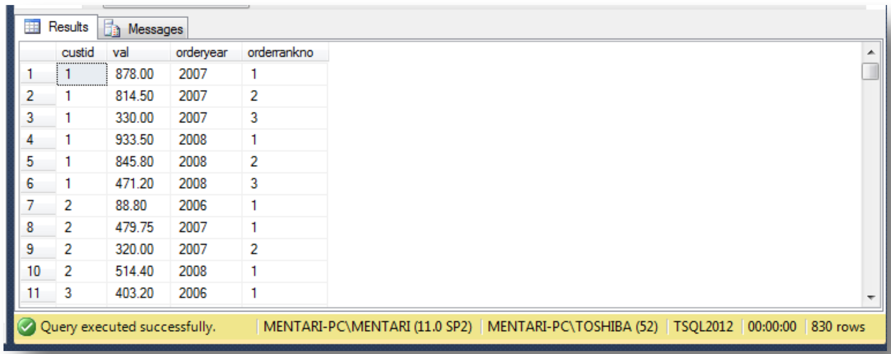
[Question-5] Write a SELECT statement to retrieve the custid and val columns from the Sales.OrderValues view. Add the following two columns:

1) orderyear as the year of the column order date
2) orderrankno as a sequence number, partitioned by customer and order year, and sorted by order value in descending order. decrease!



55 - Lab Exercise 1 - Task 4 Result.txt

```
custid      val                                     orderyear   orderrankno
----------- --------------------------------------- ----------- --------------------
1           878.00                                  2007        1
1           814.50                                  2007        2
1           330.00                                  2007        3
...
...
...
91          591.60                                  2008        2
91          427.50                                  2008        3
91          160.00                                  2008        4

(830 row(s) affected)
```



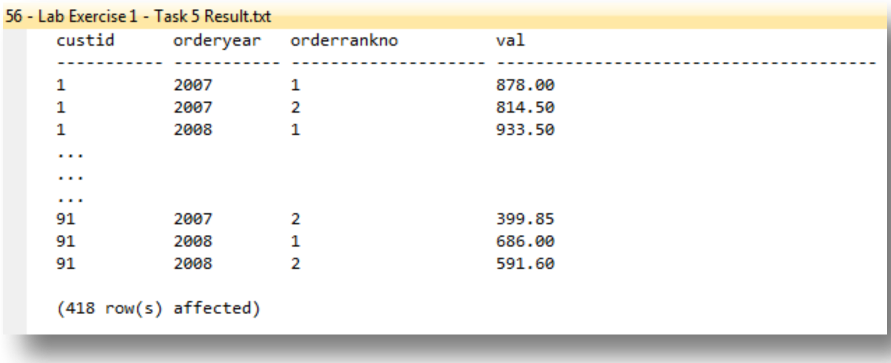
Results | Messages

	custid	val	orderyear	orderrankno
1	1	878.00	2007	1
2	1	814.50	2007	2
3	1	330.00	2007	3
4	1	933.50	2008	1
5	1	845.80	2008	2
6	1	471.20	2008	3
7	2	88.80	2006	1
8	2	479.75	2007	1
9	2	320.00	2007	2
10	2	514.40	2008	1
11	3	403.20	2006	1

Query executed successfully. | MENTARI-PC\MENTARI (11.0 SP2) | MENTARI-PC\TOSHIBA (52) | TSQL2012 | 00:00:00 | 830 rows

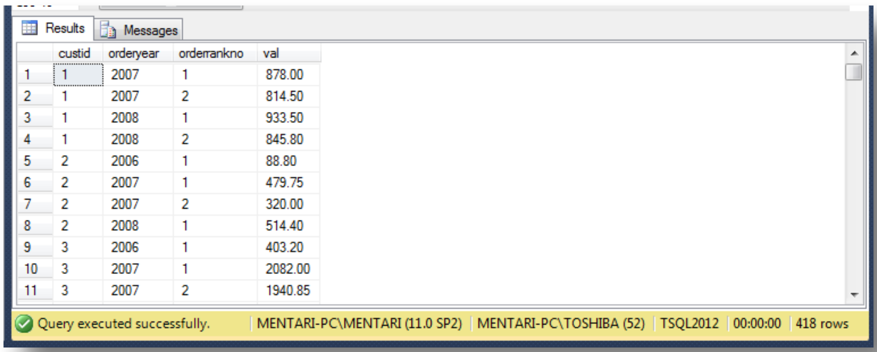
7

[Question-6] Copy the query answer to question number 6 And modification to filter only orders with the first two ranks based on column orderrankno!



56 - Lab Exercise 1 - Task 5 Result.txt

```
custid      orderyear   orderrankno          val
----------- ----------- -------------------- ---------------------------------------
1           2007        1                    878.00
1           2007        2                    814.50
1           2008        1                    933.50
...
...
...
91          2007        2                    399.85
91          2008        1                    686.00
91          2008        2                    591.60

(418 row(s) affected)
```



Results | Messages

	custid	orderyear	orderrankno	val
1	1	2007	1	878.00
2	1	2007	2	814.50
3	1	2008	1	933.50
4	1	2008	2	845.80
5	2	2006	1	88.80
6	2	2007	1	479.75
7	2	2007	2	320.00
8	2	2008	1	514.40
9	3	2006	1	403.20
10	3	2007	1	2082.00
11	3	2007	2	1940.85

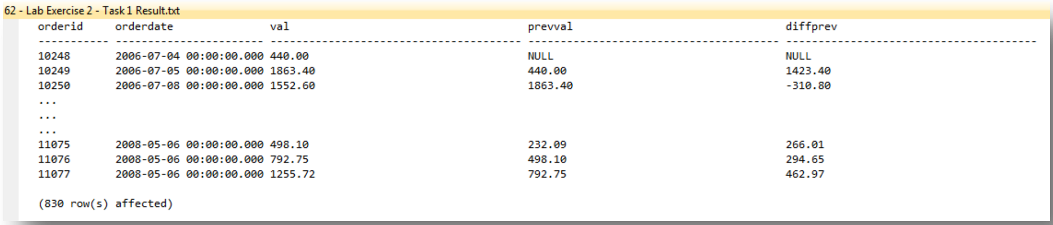
Query executed successfully. | MENTARI-PC\MENTARI (11.0 SP2) | MENTARI-PC\TOSHIBA (52) | TSQL2012 | 00:00:00 | 418 rows

8

Conclusion: After carrying out this section of the practicum, students know how to use the ranking function in T-SQL statements.

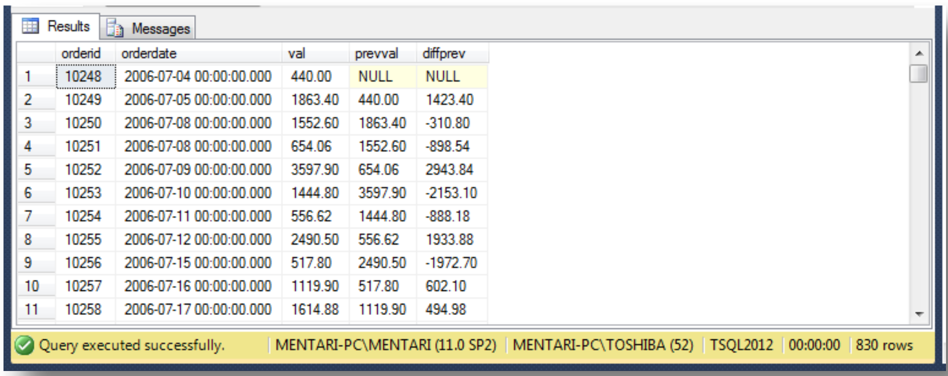
Lab – Part 2: Writing Queries Using the OFFSET Function

Step	Information
1	Scenario : Another report is needed to analyze the difference between two consecutive rows. This will make it easier for *business users* to analyze growth and trends. To carry out the experiment in this practical part 2, make sure the database is connected to "TSQL".
2	[Question-7] Create a (*common table expression*) CTE named OrderRows based on a query that retrieves the orderid, orderdate, and val columns from the Sales.OrderValues view. Add a calculated result column named rowno using the ROW_NUMBER function sorted by the orderdate and orderid columns!
3	[Question-8] Write a SELECT statement against a CTE and use a LEFT JOIN with the same CTE to retrieve the current row *and* previous row *based* on the rowno column. Return the orderid, orderdate, and val columns for the current row and the val column for the previous row as prevval. Add a calculated column named diffprev that shows the difference between the current and previous val!



62 - Lab Exercise 2 - Task 1 Result.txt

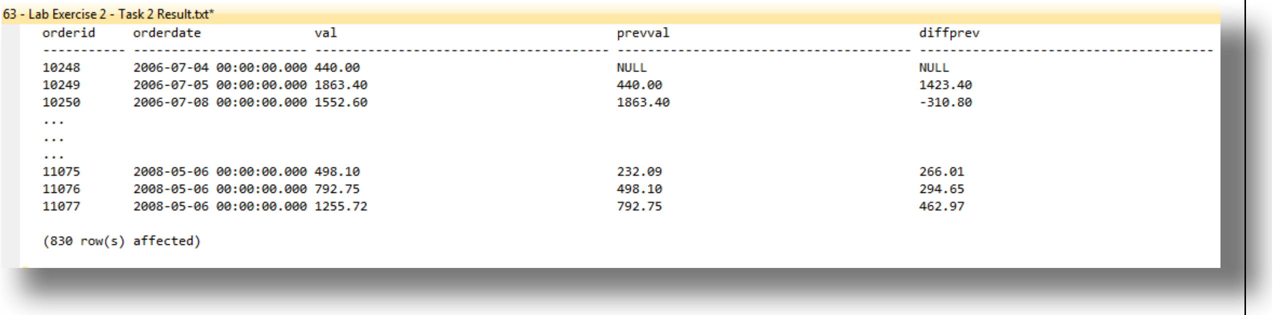
```
orderid     orderdate               val                                     prevval                                 diffprev
----------- ----------------------- --------------------------------------- --------------------------------------- ---------------------------------------
10248       2006-07-04 00:00:00.000 440.00                                  NULL                                    NULL
10249       2006-07-05 00:00:00.000 1863.40                                 440.00                                  1423.40
10250       2006-07-08 00:00:00.000 1552.60                                 1863.40                                 -310.80
...
...
...
11075       2008-05-06 00:00:00.000 498.10                                  232.09                                  266.01
11076       2008-05-06 00:00:00.000 792.75                                  498.10                                  294.65
11077       2008-05-06 00:00:00.000 1255.72                                 792.75                                  462.97

(830 row(s) affected)
```



	orderid	orderdate	val	prevval	diffprev
1	10248	2006-07-04 00:00:00.000	440.00	NULL	NULL
2	10249	2006-07-05 00:00:00.000	1863.40	440.00	1423.40
3	10250	2006-07-08 00:00:00.000	1552.60	1863.40	-310.80
4	10251	2006-07-08 00:00:00.000	654.06	1552.60	-898.54
5	10252	2006-07-09 00:00:00.000	3597.90	654.06	2943.84
6	10253	2006-07-10 00:00:00.000	1444.80	3597.90	-2153.10
7	10254	2006-07-11 00:00:00.000	556.62	1444.80	-888.18
8	10255	2006-07-12 00:00:00.000	2490.50	556.62	1933.88
9	10256	2006-07-15 00:00:00.000	517.80	2490.50	-1972.70
10	10257	2006-07-16 00:00:00.000	1119.90	517.80	602.10
11	10258	2006-07-17 00:00:00.000	1614.88	1119.90	494.98

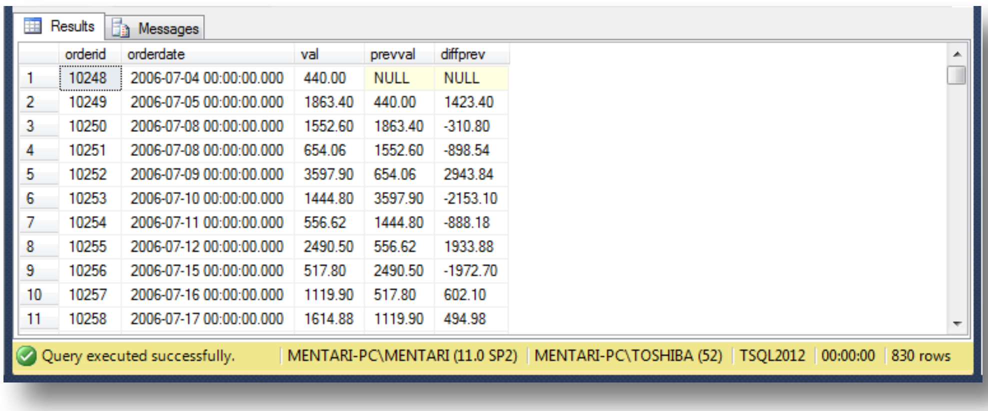
Query executed successfully. | MENTARI-PC\MENTARI (11.0 SP2) | MENTARI-PC\TOSHIBA (52) | TSQL2012 | 00:00:00 | 830 rows

4	[Question-9] Write a SELECT statement using the LAG function to get the same results as the query in question no.2! The query created in this problem does not use CTE.



63 - Lab Exercise 2 - Task 2 Result.txt*

```
orderid     orderdate               val                                       prevval                                   diffprev
----------- ----------------------- ----------------------------------------- ----------------------------------------- -----------------------------------------
10248       2006-07-04 00:00:00.000 440.00                                    NULL                                      NULL
10249       2006-07-05 00:00:00.000 1863.40                                   440.00                                    1423.40
10250       2006-07-08 00:00:00.000 1552.60                                   1863.40                                   -310.80
...
...
...
11075       2008-05-06 00:00:00.000 498.10                                    232.09                                    266.01
11076       2008-05-06 00:00:00.000 792.75                                    498.10                                    294.65
11077       2008-05-06 00:00:00.000 1255.72                                   792.75                                    462.97

(830 row(s) affected)
```



	orderid	orderdate	val	prevval	diffprev
1	10248	2006-07-04 00:00:00.000	440.00	NULL	NULL
2	10249	2006-07-05 00:00:00.000	1863.40	440.00	1423.40
3	10250	2006-07-08 00:00:00.000	1552.60	1863.40	-310.80
4	10251	2006-07-08 00:00:00.000	654.06	1552.60	-898.54
5	10252	2006-07-09 00:00:00.000	3597.90	654.06	2943.84
6	10253	2006-07-10 00:00:00.000	1444.80	3597.90	-2153.10
7	10254	2006-07-11 00:00:00.000	556.62	1444.80	-888.18
8	10255	2006-07-12 00:00:00.000	2490.50	556.62	1933.88
9	10256	2006-07-15 00:00:00.000	517.80	2490.50	-1972.70
10	10257	2006-07-16 00:00:00.000	1119.90	517.80	602.10
11	10258	2006-07-17 00:00:00.000	1614.88	1119.90	494.98

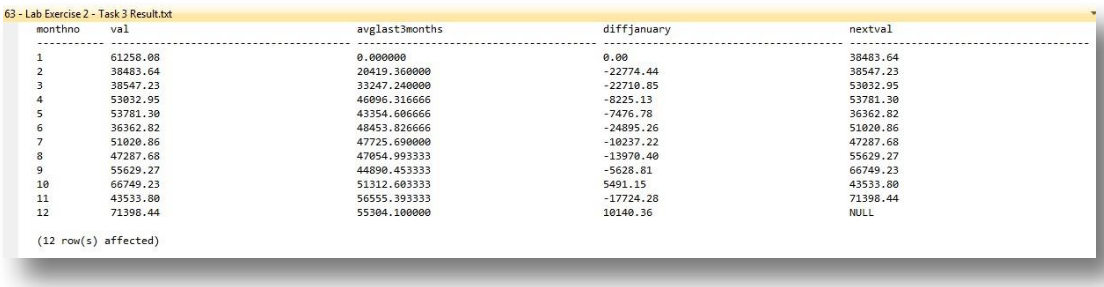
Query executed successfully. | MENTARI-PC\MENTARI (11.0 SP2) | MENTARI-PC\TOSHIBA (52) | TSQL2012 | 00:00:00 | 830 rows

5 [Question-10] Create a CTE named SalesMonth2007 that creates two columns, namely, monthno (the number of months from the orderdate column) and val (the aggregate of the val column)! Then filter the results only for the order year 2007 and group by monthno!

6 [Question-11] Write a SELECT statement that will take the monthno and val columns from the CTE and add 3 columns to display, namely:

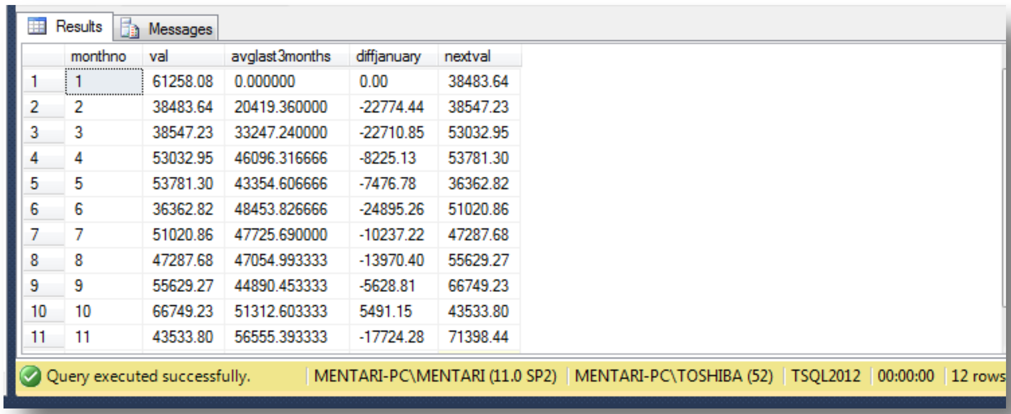
1) avglast3months (average sales amount of three months) final)
2) diffjanuary (difference between current val and val in january, use FIRST_VALUE function)
3) nextval (value of val column in month furthermore)

Information: The average amount for the last three months is not calculated correctly because the total amount of the first 2 months is divided by 3.



63 - Lab Exercise 2 - Task 3 Result.txt

```
monthno     val                                       avglast3months                            diffjanuary                               nextval
----------- ----------------------------------------- ----------------------------------------- ----------------------------------------- -----------------------------------------
1           61258.08                                  0.000000                                  0.00                                      38483.64
2           38483.64                                  20419.360000                              -22774.44                                 38547.23
3           38547.23                                  33247.240000                              -22710.85                                 53032.95
4           53032.95                                  46096.316666                              -8225.13                                  53781.30
5           53781.30                                  43354.606666                              -7476.78                                  36362.82
6           36362.82                                  48453.826666                              -24895.26                                 51020.86
7           51020.86                                  47725.690000                              -10237.22                                 47287.68
8           47287.68                                  47054.993333                              -13970.40                                 55629.27
9           55629.27                                  44890.453333                              -5628.81                                  66749.23
10          66749.23                                  51312.603333                              5491.15                                   43533.80
11          43533.80                                  56555.393333                              -17724.28                                 71398.44
12          71398.44                                  55304.100000                              10140.36                                  NULL

(12 row(s) affected)
```

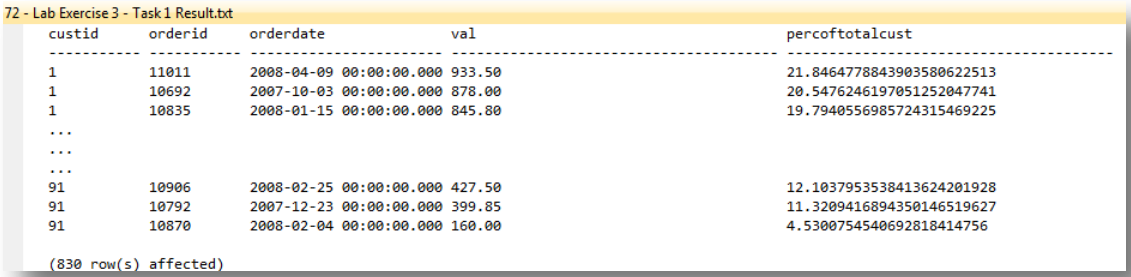
Results | Messages

	monthno	val	avglast3months	diffjanuary	nextval
1	1	61258.08	0.000000	0.00	38483.64
2	2	38483.64	20419.360000	-22774.44	38547.23
3	3	38547.23	33247.240000	-22710.85	53032.95
4	4	53032.95	46096.316666	-8225.13	53781.30
5	5	53781.30	43354.606666	-7476.78	36362.82
6	6	36362.82	48453.826666	-24895.26	51020.86
7	7	51020.86	47725.690000	-10237.22	47287.68
8	8	47287.68	47054.993333	-13970.40	55629.27
9	9	55629.27	44890.453333	-5628.81	66749.23
10	10	66749.23	51312.603333	5491.15	43533.80
11	11	43533.80	56555.393333	-17724.28	71398.44

Query executed successfully. | MENTARI-PC\MENTARI (11.0 SP2) | MENTARI-PC\TOSHIBA (52) | TSQL2012 | 00:00:00 | 12 rows

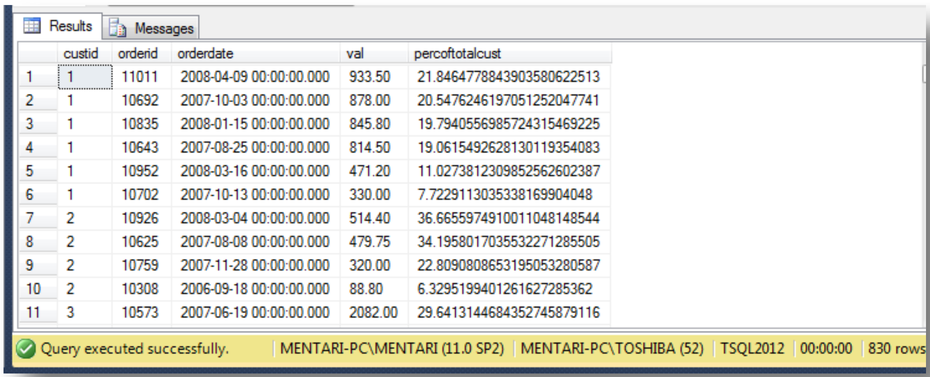
7	**Conclusion** : After carrying out this section of the practicum, students can use the OFFSET function in T-SQL statements.

Lab – Part 3: Writing Queries Using Window Aggregation Functions

Step	**Information**
1	Scenario : To better understand the cumulative sales value of customers over time and to provide sales analysts with year-long analysis a different SELECT statement using the window aggregate function is required. To carry out the experiment in this practical part 3, make sure the database is connected to "TSQL".
2	[Question-12] Write a SELECT statement to retrieve the custid, orderid, orderdate, and val columns from the Sales.OrderValues view. Add a column named percoftotalcust that contains the percentage of each sales order amount compared to the total sales for that customer! 

72 - Lab Exercise 3 - Task 1 Result.txt

```
custid      orderid     orderdate               val                                     percoftotalcust
----------- ----------- ----------------------- --------------------------------------- ---------------------------------------
1           11011       2008-04-09 00:00:00.000 933.50                                  21.8464778843903580622513
1           10692       2007-10-03 00:00:00.000 878.00                                  20.5476246197051252047741
1           10835       2008-01-15 00:00:00.000 845.80                                  19.7940556985724315469225
...
...
...
91          10906       2008-02-25 00:00:00.000 427.50                                  12.1037953538413624201928
91          10792       2007-12-23 00:00:00.000 399.85                                  11.3209416894350146519627
91          10870       2008-02-04 00:00:00.000 160.00                                  4.5300754540692818414756

(830 row(s) affected)
```



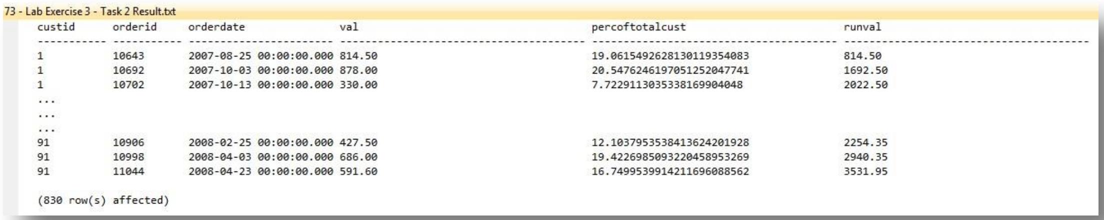
Results | Messages

	custid	orderid	orderdate	val	percoftotalcust
1	1	11011	2008-04-09 00:00:00.000	933.50	21.8464778843903580622513
2	1	10692	2007-10-03 00:00:00.000	878.00	20.5476246197051252047741
3	1	10835	2008-01-15 00:00:00.000	845.80	19.7940556985724315469225
4	1	10643	2007-08-25 00:00:00.000	814.50	19.0615492628130119354083
5	1	10952	2008-03-16 00:00:00.000	471.20	11.0273812309852562602387
6	1	10702	2007-10-13 00:00:00.000	330.00	7.7229113035338169904048
7	2	10926	2008-03-04 00:00:00.000	514.40	36.6655974910011048148544
8	2	10625	2007-08-08 00:00:00.000	479.75	34.1958017035532271285505
9	2	10759	2007-11-28 00:00:00.000	320.00	22.8090808653195053280587
10	2	10308	2006-09-18 00:00:00.000	88.80	6.3295199401261627285362
11	3	10573	2007-06-19 00:00:00.000	2082.00	29.6413144684352745879116

Query executed successfully. | MENTARI-PC\MENTARI (11.0 SP2) | MENTARI-PC\TOSHIBA (52) | TSQL2012 | 00:00:00 | 830 rows

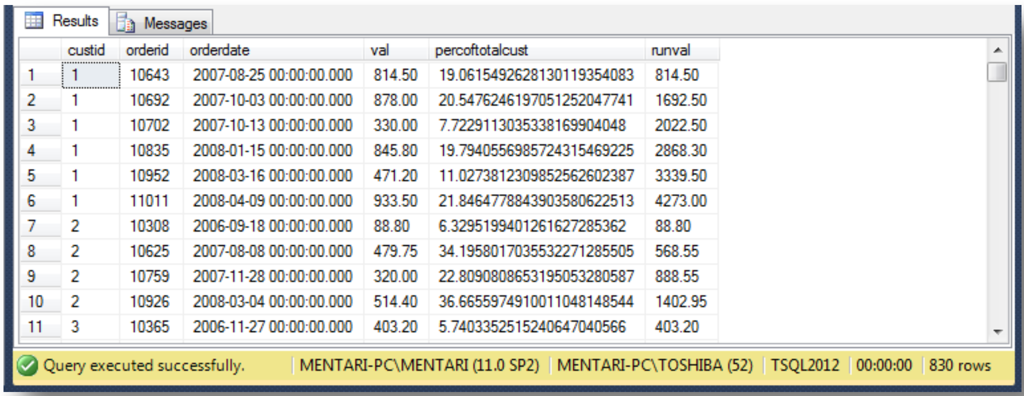
3

<span style="color:red">[Question-13]</span> Copy the previous SELECT statement and modify it by adding a new calculated column named runval! This column should contain the total sales that have occurred for each customer based on the order date, using orderid as the tiebreaker.



73 - Lab Exercise 3 - Task 2 Result.txt

```
custid      orderid     orderdate               val                                       percoftotalcust                          runval
----------- ----------- ----------------------- ----------------------------------------- ---------------------------------------- ----------------------------------------
1           10643       2007-08-25 00:00:00.000 814.50                                    19.0615492628130119354083                814.50
1           10692       2007-10-03 00:00:00.000 878.00                                    20.5476246197051252047741                1692.50
1           10702       2007-10-13 00:00:00.000 330.00                                    7.7229113035338169904048                 2022.50
...
...
...
91          10906       2008-02-25 00:00:00.000 427.50                                    12.1037953538413624201928                2254.35
91          10998       2008-04-03 00:00:00.000 686.00                                    19.4226985093220458953269                2940.35
91          11044       2008-04-23 00:00:00.000 591.60                                    16.7499539914211696088562                3531.95

(830 row(s) affected)
```



Results | Messages

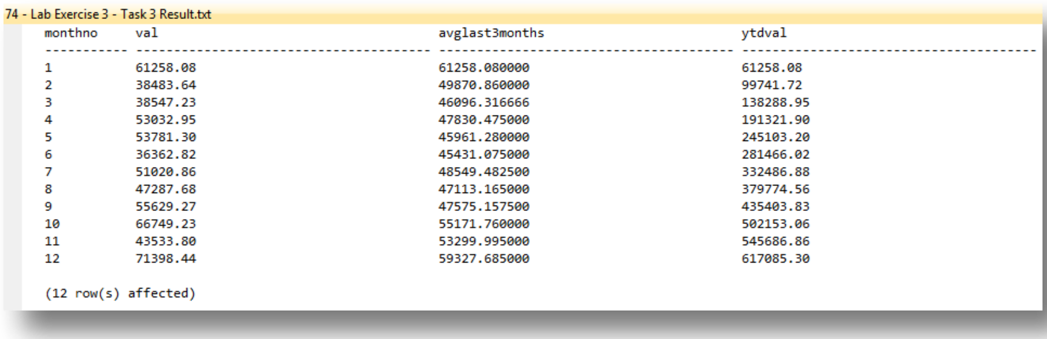
	custid	orderid	orderdate	val	percoftotalcust	runval
1	1	10643	2007-08-25 00:00:00.000	814.50	19.0615492628130119354083	814.50
2	1	10692	2007-10-03 00:00:00.000	878.00	20.5476246197051252047741	1692.50
3	1	10702	2007-10-13 00:00:00.000	330.00	7.7229113035338169904048	2022.50
4	1	10835	2008-01-15 00:00:00.000	845.80	19.7940556985724315469225	2868.30
5	1	10952	2008-03-16 00:00:00.000	471.20	11.0273812309852562602387	3339.50
6	1	11011	2008-04-09 00:00:00.000	933.50	21.8464778843903580622513	4273.00
7	2	10308	2006-09-18 00:00:00.000	88.80	6.3295199401261627285362	88.80
8	2	10625	2007-08-08 00:00:00.000	479.75	34.1958017035532271285505	568.55
9	2	10759	2007-11-28 00:00:00.000	320.00	22.8090808653195053280587	888.55
10	2	10926	2008-03-04 00:00:00.000	514.40	36.6655974910011048148544	1402.95
11	3	10365	2006-11-27 00:00:00.000	403.20	5.7403352515240647040566	403.20

Query executed successfully. | MENTARI-PC\MENTARI (11.0 SP2) | MENTARI-PC\TOSHIBA (52) | TSQL2012 | 00:00:00 | 830 rows

4

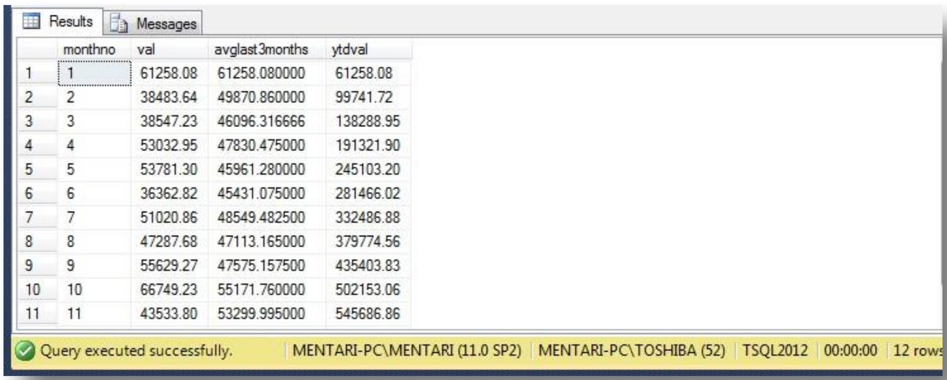
<span style="color:red">[Question-14]</span> Copy the SalesMonth2007 CTE in experiment 2. Write a SELECT statement to retrieve the monthno and val columns. Add two computed columns:

1) avglast3months. This column should contain the average sales amount for the last three months before the current month using the aggregate window function. Assume that there are no *missing months* .
2) ytdval This column must contain the cumulative sales value up to the current month. This.



74 - Lab Exercise 3 - Task 3 Result.txt

```
monthno     val                                     avglast3months                          ytdval
----------- --------------------------------------- --------------------------------------- ---------------------------------------
1           61258.08                                61258.080000                            61258.08
2           38483.64                                49870.860000                            99741.72
3           38547.23                                46096.316666                            138288.95
4           53032.95                                47830.475000                            191321.90
5           53781.30                                45961.280000                            245103.20
6           36362.82                                45431.075000                            281466.02
7           51020.86                                48549.482500                            332486.88
8           47287.68                                47113.165000                            379774.56
9           55629.27                                47575.157500                            435403.83
10          66749.23                                55171.760000                            502153.06
11          43533.80                                53299.995000                            545686.86
12          71398.44                                59327.685000                            617085.30

(12 row(s) affected)
```



	monthno	val	avglast3months	ytdval
1	1	61258.08	61258.080000	61258.08
2	2	38483.64	49870.860000	99741.72
3	3	38547.23	46096.316666	138288.95
4	4	53032.95	47830.475000	191321.90
5	5	53781.30	45961.280000	245103.20
6	6	36362.82	45431.075000	281466.02
7	7	51020.86	48549.482500	332486.88
8	8	47287.68	47113.165000	379774.56
9	9	55629.27	47575.157500	435403.83
10	10	66749.23	55171.760000	502153.06
11	11	43533.80	53299.995000	545686.86

Query executed successfully. | MENTARI-PC\MENTARI (11.0 SP2) | MENTARI-PC\TOSHIBA (52) | TSQL2012 | 00:00:00 | 12 rows

5	**Conclusion** : After doing this practical section, you will gain a basic understanding of how to use the window aggregation function in T-SQL statements.

--- Have a great time doing it ----