TOP(N), Ranking, NTILE and Analytic Functions

Presented by Steve Stedman and Aaron Buma

Welcome

- Welcome to all those joining us remotely.
  - For any questions via Google On Air Broadcasts, we will address these at the end of the training
- Training provided by Emergency Reporting
  - http://EmergencyReporting.com
- Presented by Steve Stedman
- And Aaron Buma

Agenda

- TOP(N)
- RANK and DENSE_RANK
- ROW_NUMBER
- NTILE
- OVER clause enhancements
- LAG and LEAD, FIRST_VALUE and LAST_VALUE
- PERCENT_RANK
- PERCENTILE_DISC and PERCENTILE_CONT
- CUME_DIST - Cumulative Distribution

Top (), Rank, Row Number

Presented by Aaron Buma

Part One – Aaron Buma

- TOP (n)
  - WITH TIES
  - PERCENT
- RANK and DENSE_RANK
- ROW_NUMBER

SELECT TOP (N)

- TOP (n)
  - n as variable
  - WITH TIES
  - PERCENT

Rank, Dense_Rank, Row_number

- RANK and DENSE_RANK
  - Requires “ OVER(ORDER BY columnName ASC/DESC) to declare what to rank and in what order
  - Identical results until duplicates are encountered
- ROW_NUMBER
  - Disregards duplicates

NTILE and Analytic Functions

Presented by Steve Stedman

Part Two – Steve Stedman

- NTILE
- OVER Clause Enhancements
  - ROWS PRECEDING, FOLLOWING, UNBOUNDED
  - RANGE PRECEDING, FOLLOWING, UNBOUNDED
- LAG and LEAD
- FIRST_VALUE and LAST_VALUE
- PERCENT_RANK
- PERCENTILE_DISC and PERCENTILE_CONT
- CUME_DIST - Cumulative Distribution

NTILE

- Median – Split the group into 2 evenly sized groups.
  - Evenly in this case is defined as being as close to even as possible given the number of items in the group.
- Quartile – Split the set into 4 evenly sized groups.
- Decile – Split the set into 10 evenly sized groups.
- Percentile – Split the group into 100 evenly sized groups.
- NTile – Split the group into N evenly sized groups.

The Pre-2012 OVER Clause

- The OVER clause before SQL Server 2012…
- Really handy to perform aggregates over a different range than your standard grouping.

```
select *,
        avg(Revenue) OVER (PARTITION by DepartmentID) as AverageRevenue,
        sum(Revenue) OVER (PARTITION by DepartmentID) as TotalRevenue
from REVENUE
order by departmentID, year;
```

OVER Clause Enhancements

- ROWS PRECEDING, FOLLOWING, UNBOUNDED
  - Rows refers to the current row and those before or after base on preceding or following.

	Year	DepartmentID	Revenue	RowsCumulative	RangeCumulative
1	2003	1	10300	10300	10300
2	2004	1	10000	20300	20300
3	2005	1	20000	40300	50300
4	2005	1	10000	50300	50300
5	2006	1	40000	90300	90300
6	2007	1	70000	160300	160300
7	2008	1	50000	210300	210300

PRECEDING, FOLLOWING, UNBOUNDED

- ROWS or RANGE- specifying rows or range
- PRECEDING – get rows before the current one
- FOLLOWING – get rows after the current one
- UNBOUNDED – get all before or after
- CURRENT ROW

```
sum(Revenue) OVER (PARTITION by DepartmentID
                   ORDER BY [YEAR]
                   ROWS BETWEEN 3 PRECEDING AND CURRENT ROW) as Prev3
```

```
sum(Revenue) OVER (PARTITION by DepartmentID
                   ORDER BY [YEAR]
                   ROWS BETWEEN 1 PRECEDING AND 1 FOLLOWING) as BeforeAndAfter
```

Demo

```
--ROWS PRECEDING
-- http://stevestedman.com/?p=1454
--  look at the sum of revenue over a trailing 3 year period
select Year, DepartmentID, Revenue,
   sum(Revenue) OVER (PARTITION by DepartmentID
  ORDER BY [YEAR]
  ROWS BETWEEN 3 PRECEDING AND CURRENT ROW) as Prev3
from REVENUE  order by departmentID, year;

--ROWS PRECEDING
select Year, DepartmentID, Revenue,
   sum(Revenue) OVER (PARTITION by DepartmentID
  ORDER BY [YEAR]
  ROWS BETWEEN 5 PRECEDING AND 2 PRECEDING) as Prev5to3
from REVENUE order by departmentID, year;

-- ROWS FOLLOWING
-- http://stevestedman.com/?p=1454
select Year, DepartmentID, Revenue,
   sum(Revenue) OVER (PARTITION by DepartmentID
  ORDER BY [YEAR]
  ROWS BETWEEN CURRENT ROW AND 3 FOLLOWING) as Next3
from REVENUE order by departmentID, year;

--ROWS PRECEDING
select Year, DepartmentID, Revenue,
   sum(Revenue) OVER (PARTITION by DepartmentID
  ORDER BY [YEAR]
  ROWS BETWEEN 1 PRECEDING AND 1 FOLLOWING) as BeforeAndAfter
from REVENUE order by departmentID, year;
```

LAG and LEAD

- Allows you to reference a value in a row before or a row after the current row
- Over clause is used to specify the ordering and partitioning

FIRST_VALUE and LAST_VALUE

- FIRST_VALUE returns the first value in an ordered set of values and
- LAST_VALUE returns the last value in an ordered set of values
- Utilize the OVER clause to set ranges and orders

- Interesting catch – LAST_VALUE specifies the last value so far unless you specify otherwise

PERCENT RANK

AnalyticFunctions.s...(xps420\Steve (56))

Editor | Results | Messages

	DepartmentID	Revenue	Year	RankYear	PercentRank
1	1	10000	2004	1	0
2	1	10000	2012	1	0
3	1	10030	1998	3	0.142857142857143
4	1	10300	2003	4	0.214285714285714
5	1	20000	2009	5	0.285714285714286
6	1	20000	1999	5	0.285714285714286
7	1	20000	2005	5	0.285714285714286
8	1	30000	2001	8	0.5
9	1	30000	2010	8	0.5
10	1	40000	2000	10	0.642857142857143
11	1	40000	2006	10	0.642857142857143
12	1	50000	2008	12	0.785714285714286
13	1	70000	2007	13	0.857142857142857
14	1	80000	2011	14	0.928571428571429
15	1	90000	2002	15	1

14.2% of the values are less than or equal to this one.

50% of the values are less than or equal to this one.

PERCENTILE_DISC and PERCENTILE_CONT

- PERCENTILE_CONT interpolates the appropriate value, even if it is not in the set.
- PERCENTILE_DISC picks a percentile that exists in the set.
- Keep in mind that any nulls in the data set are ignored.
- What's the difference in the names
  - CONT stands for continuous
  - DISC stands for discrete distribution which means that the percentile will exist in the set.

CUME_DIST - Cumulative Distribution

- CUME_DIST is the function that maps values to their percentile rank in a distribution.
- The CUME_DIST function calculates the possibility of another occurrence being of that value or lesser than that among a group of values.

More statistics with Cumulative Distribution

- Testing the Rand() function
- Testing probability with dice rolls

Quiz 1:

- Which of the following is **<u>incorrect</u>**:
  - In regards to: TOP (n) PERCENT
    - n can be any value
    - n can be any value from 0 – 100
    - n can be a variable, valued 0 - 100

Quiz 2:

- If a table with four values of ‘1’ and a single ‘2’ in ColumnA, what will the result set be with the following statement:

```
SELECT DENSE_RANK() OVER (ORDER BY ColumnA ASC) AS ColumnA FROM dbo.TestTable;
```

	ColumnA
1	1
2	1
3	1
4	1
5	5

	ColumnA
1	1
2	2
3	3
4	4
5	5

	ColumnA
1	1
2	1
3	1
4	1
5	2

Quiz 3

- If a duplicate value is encountered, ROW_NUMBER does which of the following:
  - Repeat the counter/row number until the first non-duplicate is found, the continue with next number
  - Continue to increase the counter/row number when a duplicate is found
  - Repeat the counter/row number until the first non-duplicate is found, then skip to number which would match the row number as if no duplicates were encountered.

Quiz 4:

- NTILE(100) is also known as?
  - Centile
  - Percentile
  - Tile100

```
SELECT *, NTILE(100) OVER(ORDER BY [Revenue] ASC) AS Percentile
  FROM Revenue
  ORDER BY [Year];
```

Quiz 5:

```
AVG(Revenue) OVER (PARTITION by DepartmentID
                   ORDER BY [YEAR]
                   ROWS BETWEEN 2 PRECEDING AND CURRENT ROW)
```

- The query fragment above uses up to how many rows in the AVG calculation?
  - All rows
  - 2 rows
  - 3 rows

Any Questions?

- TOP(N)
- RANK and DENSE_RANK
- ROW_NUMBER
- NTILE
- OVER clause enhancements
- LAG and LEAD, FIRST_VALUE and LAST_VALUE
- PERCENT_RANK
- PERCENTILE_DISC and PERCENTILE_CONT
- CUME_DIST - Cumulative Distribution

For More Information

- Visit http://EmergencyReporting.com to find out more about Emergency Reporting.
- Aaron on the web
  - http://AaronBuma.com
  - Twitter: @AaronDBuma
- Steve on the web
  - http://SteveStedman.com
  - twitter: @SqlEmt

Tune in next week

- Thursday 2/12 at 9:00am (pacific time).

- Topics
  - Set Operators
    - UNION, UNION ALL, INTERSECT, EXCEPT
    - Set Grouping