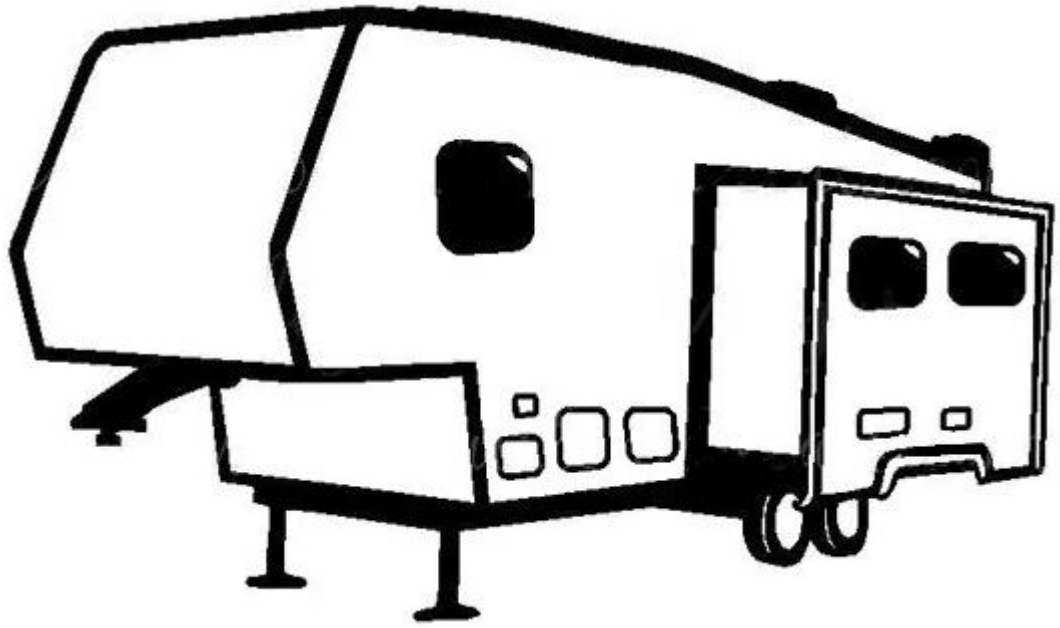
Aaron N. Cutshall, MSCIS, MSHI

SQL RV http://www.sqlrv.com



The Power of Window Functions

Just who is this guy?

SQL RV

healthfirst
Health Insurance for New Yorkers

Enterprise Data Architect

PASS SQLSATURDAY

Speaker – various events

B.S. Computer Science

M.S. Computer Information Systems

M.S. Health Informatics

Doctorate in Healthcare Administration (Aug 2023)

35+ Years

Something to consider…

credit

“If a window of opportunity appears, don't pull down the shade.”

-- Tom Peters, author of *In Search of Excellence*

Agenda

- What are Window Functions?
- Window Function Anatomy
- How Window Functions Work
- The Types of Window Functions
  - Value
  - Ranking
  - Aggregation
- Examples in Action
- Summary

What are Window Functions?

- Has absolutely nothing to do with Microsoft Windows
- Provides a view, or “window,” within the result set
  - Treats data in groups or across all data
  - Results are returned without losing detail
  - Aggregates are most known, but not only feature
  - Allows selection of data from other rows
  - Each function can have a different window

What are Window Functions?

- SQL Order of Operations:
  - FROM (including any JOIN clauses and criteria)
  - WHERE (filter data after FROM/JOIN clauses)
  - GROUP BY (performs aggregates in the specified group)
  - HAVING (can reference aggregates)
  - WINDOW (all window functions [can include aggregates])
  - SELECT (presents all data specified)
  - DISTINCT (yes, this happens after data is SELECTed!)
  - UNION, INTERSECT, EXCEPT (all set operations)
  - ORDER BY (only one stated is needed for all queries)
  - LIMIT, FETCH, TOP (depending upon DB implementation)

What are Window Functions?

- Works only on the final query result set
  - Performed after WHERE clause, before SELECT clause
  - Can be referenced in SELECT and ORDER BY clauses only
- Example:

```
-- Doesn't work, cannot put window functions in GROUP BY
SELECT ntile(4) OVER (ORDER BY age) AS bucket, MIN(age), MAX(age)
FROM customer
GROUP BY ntile(4) OVER (ORDER BY age);
```

```
-- Works:
SELECT quartile, MIN(age), MAX(age)
FROM (
  SELECT age, ntile(4) OVER (ORDER BY age) AS quartile
  FROM customer
  ) c
GROUP BY quartile;
```

Window Function Anatomy

```
window_function_name ( <expression> )
     [filter (where <condition>)]
     over (<over_clause>)
```

- `window_function_name` – Window function name
- `<expression>` – Target expression or column (parameters)
- `<condition>` – Simple aggregate filter criteria (ex. rownbr = 1)
- `<over_clause>`:
  - `<partition_clause>` – The window partition – the groups of rows
  - `<order_clause>` – Specifies the order of rows within a partition
  - `<frame_clause>` – The parameters of the frame size

Window Function Anatomy

- `<partition_clause>`

  `PARTITION BY` $expr_1$, $expr_2$, ... $expr_n$
- Identifies what columns define the grouping
  - Allows one or more columns to specify grouping
  - All column values collective identify the partition
  - If clause is omitted, entire result set will be the partition
- Partition determines the scope of the function
  - Function results calculated only for current partition
  - Multiple functions with different partitions can be used

Window Function Anatomy

- <order_clause>

```
ORDER BY
    expression1 [ASC | DESC] [NULLS {FIRST| LAST}],
    expression2 [ASC | DESC] [NULLS {FIRST| LAST}],
    ...
    expressionn [ASC | DESC] [NULLS {FIRST| LAST}]
```

- Identifies the columns to order the results within partition
  - Specify sort order either ascending (default) or descending
  - Specify if NULLS display in the result set FIRST or LAST (default)
  - Multiple expressions allowable separated by comma
  - *NULLS statement not supported in all RDBMS implementations

Window Function Anatomy

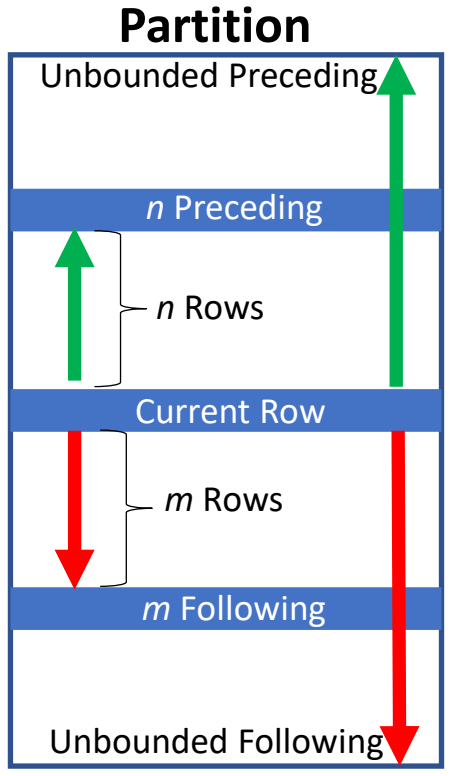
- `<frame_clause>`
  `{ ROWS | RANGE | GROUPS } lower_bound`
  `{ ROWS | RANGE | GROUPS } BETWEEN`
  `lower_bound AND upper_bound`
- Optionally specify the frame structure
  - ROWS are based upon result set rows
  - RANGE is based on an offset value – not count
  - GROUPS are based on peer group values
- Frame references:

`lower_bound`

- *n* PRECEDING
- UNBOUNDED PRECEDING
- CURRENT ROW

`upper_bound`

- *m* FOLLOWING
- UNBOUNDED FOLLOWING
- CURRENT ROW



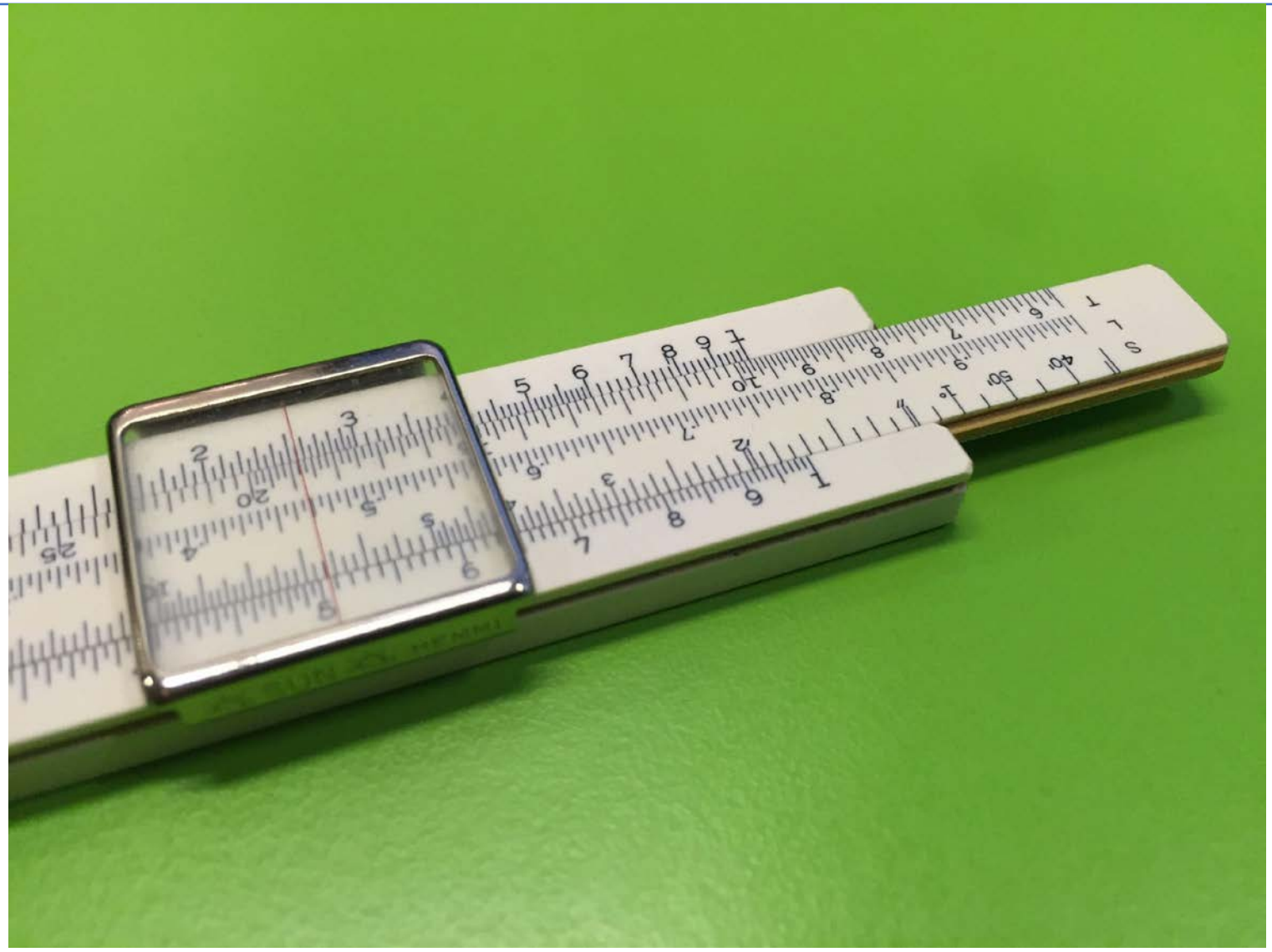
Window Function Anatomy

- `window_clause`

  `WINDOW` *`window_name`* `AS (<over_clause>)`
- Available in PostgreSQL, MySQL, and SQL Server 2022
- Placed after WHERE clause but before the ORDER BY clause – multiple allowed separated by comma
- Example:

```
SELECT quartile, MIN(age), MAX(age)
FROM (
  SELECT age, ntile(4) OVER my_window AS quartile
  FROM customer
  ) c
WINDOW my_window AS (ORDER BY age)
GROUP BY quartile;
```

How Window Functions Work



How Window Functions Work

Partition by:
ordernumber

Order by:
orderlinenumber

ordernumber	orderlinenumber	productcode	category	priceeach	quantityordered	total
10111	1	S18_3136	Vintage Cars	$ 100.00	43	$4,300.00
10111	2	S18_2957	Vintage Cars	$ 64.33	28	$1,801.24
10111	3	S24_4258	Vintage Cars	$ 86.68	26	$2,253.68
10111	4	S18_3320	Vintage Cars	$ 100.00	39	$3,900.00
10111	5	S18_1367	Vintage Cars	$ 49.06	48	$2,354.88
10111	6	S18_1342	Vintage Cars	$ 99.66	33	$3,288.78
10112	1	S10_1949	Classic Cars	$ 100.00	29	$2,900.00
10112	2	S18_2949	Vintage Cars	$ 100.00	23	$2,300.00
10113	1	S32_3522	Trucks and Buses	$ 68.52	23	$1,575.96
10113	2	S12_1666	Trucks and Buses	$ 100.00	21	$2,100.00
10113	3	S18_4668	Vintage Cars	$ 49.81	50	$2,490.50
10113	4	S18_1097	Trucks and Buses	$ 100.00	49	$4,900.00
10114	1	S24_2840	Classic Cars	$ 30.06	24	$ 721.44
10114	2	S32_2509	Trucks and Buses	$ 55.73	28	$1,560.44
10114	3	S18_2319	Trucks and Buses	$ 100.00	39	$3,900.00
10114	4	S18_3232	Classic Cars	$ 100.00	48	$4,800.00
10114	5	S24_2300	Trucks and Buses	$ 100.00	21	$2,100.00
10114	6	S18_2432	Trucks and Buses	$ 68.67	45	$3,090.15
10114	7	S32_1268	Trucks and Buses	$ 100.00	32	$3,200.00
10114	8	S10_4962	Classic Cars	$ 100.00	31	$3,100.00
10114	9	S18_4600	Trucks and Buses	$ 100.00	41	$4,100.00
10114	10	S700_2824	Classic Cars	$ 100.00	42	$4,200.00
10115	1	S24_4048	Classic Cars	$ 100.00	44	$4,400.00
10115	2	S24_1444	Classic Cars	$ 69.36	47	$3,259.92

Lower Bound

Current Row

Upper Bound

How Window Functions Work

- Groups data within the result set
  - PARTITION clause identifies the grouping for each partition
  - ORDER clause is not tied to the final query ORDER BY statement
  - FRAME clause narrows the partition relative to the current row
- Selects values or calculates ranking or aggregate results
  - They are determined after all other data is selected
  - Can only be visible to SELECT and ORDER BY query clauses
  - They go across all resultant rows according to the OVER clause
- Places result in each row as a column within the data set

The Types of Window Functions

- Value Functions
  - first_value(*column*)/last_value(*value*)
    - Return the first/last value for the column within the OVER clause
    - Defined by the PARTITION BY and ORDER BY sub-clauses
  - lead (*column*[, *offset* [, *default*]]) /lag(*column*[, *offset* [, *default*]])
    - Return the next/last value for the column within the OVER clause
    - Defined by the PARTITION BY and ORDER BY sub-clauses
  - nth_value(*column*, *n*)
    - Return the value for the column of the nth row in the partition
    - A positive value is for a given row
    - Sort values in descending order for a reverse effect

The Types of Window Functions

- Rank Functions
  - row_number()
    - Return the current row sequence within its partition, counting from 1
    - Sequences rows in the partition without duplicates regardless of value
    - Relies upon the ordering specified in the ORDER BY clause
  - rank()
    - Return the rank of the current row with possible gaps or duplicates
    - A sequence based on the value of columns in the ORDER BY clause
  - dense_rank()
    - Return the rank of the current row with possible duplicates but no gaps
    - A sequence based on the value of columns in the ORDER BY clause

The Types of Window Functions

- Rank Functions (cont.)
  - percent_rank()
    - Return a percentile ranking of numeric values from 0 to 1 (w/ gaps)
    - Assigns ranking from lowest value to highest (may have duplicates)
  - cume_dist()
    - Return the cumulative distribution (%) of a value within a set of values
    - # of rows with values <= to current row value ÷ total number of rows
  - ntile(*n*)
    - Divides rows into *n* buckets as evenly as possible
    - Relies upon the ordering specified in the ORDER BY clause

The Types of Window Functions

- Aggregate Functions
  - Defined by the PARTITION BY and ORDER BY sub-clause
  - count(column)
    - Return the number of occurrences of each value of the column
  - min(column)/max(column)
    - Return the min/max value for the column
  - avg(column)
    - Return the average value for the column
  - sum(column)
    - Return the sum value for the column

Window Functions Demo

credit

Closing Thoughts

- Provides a view, or “window,” in data
  - Treats data in groups or across all data
  - Results are returned without losing detail
  - Aggregates are most known, but not only feature
- Allows selection of data from other rows
  - Permits each function to have its own window
  - Used in SELECT or ORDER BY only
  - Cannot be used in WHERE clause or join criteria

Helpful Resources

- Books:
  - T-SQL Window Functions: For data analysis and beyond by Itzik Ben-Gan
  - Expert T-SQL Window Functions in SQL Server by Kathi Kellenberger & Clayton Groom
- Articles:
  - An Overview of PostgreSQL Window Functions by Rohin Daswani
  - Window Functions in PostgreSQL (Course)
- Other Resources:
  - Modern SQL (blog by Markus Winand)
  - SQL Server Central (Postgres too!)

Questions & Comments

BONUS:
A **TON** of free eBooks from Microsoft, RedGate, and SentryOne!

PRESENTATION FEEDBACK:
- Your thoughts needed
- Improve presentations
- Make this event even more valuable!!!

Aaron N. Cutshall

@sqlrv
www.linkedin.com/in/sqlrv
aaron@sqlrv.com
www.sqlrv.com
www.youtube.com/@sqlrv