Overview

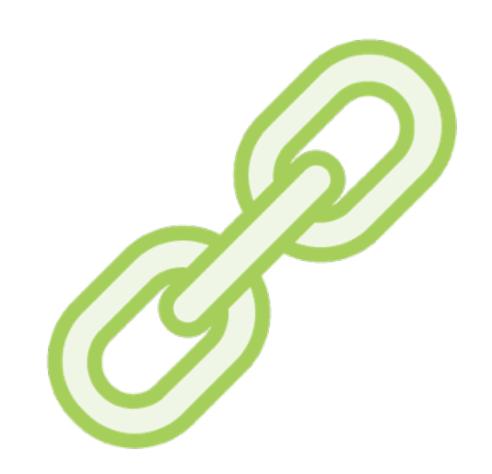
Understand windowing functions as a way to write complex queries

Use windowing functions to calculate

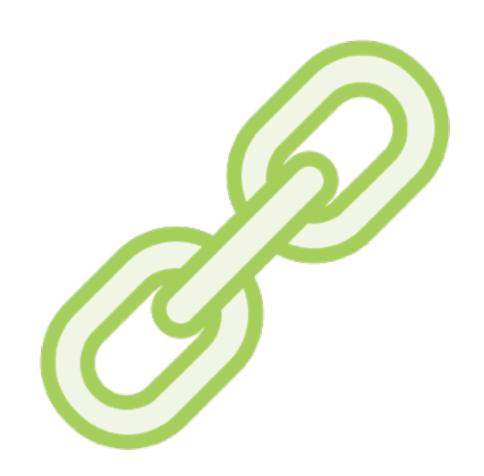
- Moving averages
- Percentage contribution
- Percentile for a value

Implement queries for all of these

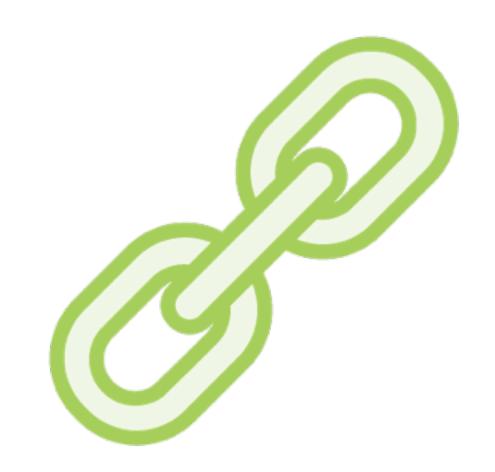
What are Window Functions?



A suite of functions which are syntactic sugar for complex queries



Make complex operations simple without needing many intermediate calculations



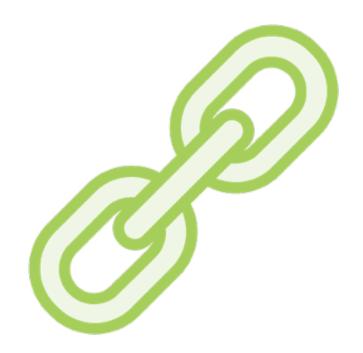
Reduces the need for intermediate tables to store temporary data



What were the top selling N products in last week's sale?

Window = one week

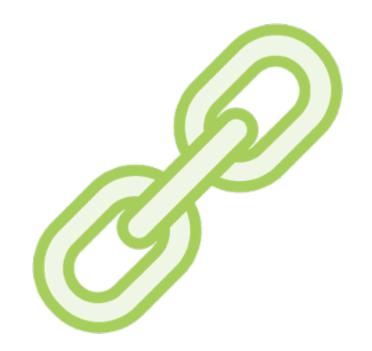
Operation = ranking product sales



What revenue percentile did this supplier fall into for this quarter?

Window = one quarter

Operation = percentiles on revenues



What were the top selling N products in last week's sale?

What revenue percentile did this supplier fall into for this quarter?

Can be expressed in a single query

An Example of a Window Function

ID	Store	Product	Date	Revenue
01	Seattle	Bananas	2017-01-01	7
02	Kent	Apples	2017-01-02	20
03	Bellevue	Flowers	2017-01-02	10
04	Redmond	Meat	2017-01-03	40
05	Seattle	Potatoes	2017-01-04	9
06	Bellevue	Bread	2017-01-04	5
07	Redmond	Bread	2017-01-05	5
08	Issaquah	Onion	2017-01-05	4

Orders in a grocery store

ID	Store	Product	Date	Revenue
01	Seattle	Bananas	2017-01-01	7
02	Kent	Apples	2017-01-02	20
03	Bellevue	Flowers	2017-01-02	10
04	Redmond	Meat	2017-01-03	40
05	Seattle	Potatoes	2017-01-04	9
06	Bellevue	Bread	2017-01-04	5
07	Redmond	Bread	2017-01-05	5
80	Issaquah	Onion	2017-01-05	4

Sorted by order ID

ID	Store	Product	Date	Revenue
01	Seattle	Bananas	2017-01-01	7
02	Kent	Apples	2017-01-02	20
03	Bellevue	Flowers	2017-01-02	10
04	Redmond	Meat	2017-01-03	40
05	Seattle	Potatoes	2017-01-04	9
06	Bellevue	Bread	2017-01-04	5
07	Redmond	Bread	2017-01-05	5
08	Issaquah	Onion	2017-01-05	4

Running Total
7
27
37
77
86
91
96
10

ID	Store	Product	Date	Revenue
01	Seattle	Bananas	2017-01-01	7
02	Kent	Apples	2017-01-02	20
03	Bellevue	Flowers	2017-01-02	10
04	Redmond	Meat	2017-01-03	40
05	Seattle	Potatoes	2017-01-04	9
06	Bellevue	Bread	2017-01-04	5
07	Redmond	Bread	2017-01-05	5
08	Issaquah	Onion	2017-01-05	4

R	ınning To	tal
	7	
	27	
	37	
	77	
	86	
	91	
	96	
	10	

ID	Store	Product	Date	Revenue
01	Seattle	Bananas	2017-01-01	7
02	Kent	Apples	2017-01-02	20
03	Bellevue	Flowers	2017-01-02	10
04	Redmond	Meat	2017-01-03	40
05	Seattle	Potatoes	2017-01-04	9
06	Bellevue	Bread	2017-01-04	5
07	Redmond	Bread	2017-01-05	5
08	Issaquah	Onion	2017-01-05	4

Rı	Running Total				
	7				
	27				
	3/				
	77				
	86				
	91				
	96				
	10				

ID	Store	Product	Date	Revenue
01	Seattle	Bananas	2017-01-01	7
02	Kent	Apples	2017-01-02	20
03	Bellevue	Flowers	2017-01-02	10
04	Redmond	Meat	2017-01-03	40
05	Seattle	Potatoes	2017-01-04	9
06	Bellevue	Bread	2017-01-04	5
07	Redmond	Bread	2017-01-05	5
80	Issaquah	Onion	2017-01-05	4

R	Running Total				
	7				
	27	ı			
	37				
	77				
	86				
	91				
	96				
	10				

ID	Store	Product	Date	Revenue
01	Seattle	Bananas	2017-01-01	7
02	Kent	Apples	2017-01-02	20
03	Bellevue	Flowers	2017-01-02	10
04	Redmond	Meat	2017-01-03	40
05	Seattle	Potatoes	2017-01-04	9
06	Bellevue	Bread	2017-01-04	5
07	Redmond	Bread	2017-01-05	5
08	Issaquah	Onion	2017-01-05	4

Rı	unning To	tal
	7	
	27	
	37	1
	77	
	86	•
	91	
	96	
	10	

ID	Store	Product	Date	Revenue
01	Seattle	Bananas	2017-01-01	7
02	Kent	Apples	2017-01-02	20
03	Bellevue	Flowers	2017-01-02	10
04	Redmond	Meat	2017-01-03	40
05	Seattle	Potatoes	2017-01-04	9
06	Bellevue	Bread	2017-01-04	5
07	Redmond	Bread	2017-01-05	5
08	Issaquah	Onion	2017-01-05	4

Rı	unning Total
	7
	27
	37
	77
	86
	91
	96
	10

For each row: Calculate sum over all rows till the current row

ID	Store	Product	Date	Revenue
01	Seattle	Bananas	2017-01-01	7
02	Kent	Apples	2017-01-02	20
03	Bellevue	Flowers	2017-01-02	10
04	Redmond	Meat	2017-01-03	40
05	Seattle	Potatoes	2017-01-04	9
06	Bellevue	Bread	2017-01-04	5
07	Redmond	Bread	2017-01-05	5
08	Issaquah	Onion	2017-01-05	4

R	unning Total
	7
	27
	37
	77
	86
	91
	96
	10

Operation = sum

ID	Store	Product	Date	Revenue
01	Seattle	Bananas	2017-01-01	7
02	Kent	Apples	2017-01-02	20
03	Bellevue	Flowers	2017-01-02	10
04	Redmond	Meat	2017-01-03	40
05	Seattle	Potatoes	2017-01-04	9
06	Bellevue	Bread	2017-01-04	5
07	Redmond	Bread	2017-01-05	5
08	Issaquah	Onion	2017-01-05	4

Running Total
7
27
37
77
86
91
96
10

Window is all rows from the top to the current row

Calculate aggregations over a window of records using window functions

- running total
- running average

Window Function with a Reset

Per Day Revenue Totals

ID	Store	Product	Date	Revenue
01	Seattle	Bananas	2017-01-11	7
02	Kent	Apples	2017-01-11	20
03	Bellevue	Flowers	2017-01-11	10
04	Redmond	Meat	2017-01-12	40
		.,,,,		. •
05	Seattle	Potatoes	2017-01-12	9
06	Bellevue	Bread	2017-01-12	5
07	Redmond	Bread	2017-01-13	5
08	Issaquah	Onion	2017-01-13	4

A revenue running total for each day

Per Day Revenue Totals

ID	Store	Product	Date	Revenue
01	Seattle	Bananas	2017-01-11	7
02	Kent	Apples	2017-01-11	20
03	Bellevue	Flowers	2017-01-11	10
04	Redmond	Meat	2017-01-12	40
05	Seattle	Potatoes	2017-01-12	9
06	Bellevue	Bread	2017-01-12	5
07	Redmond	Bread	2017-01-13	5
08	Issaquah	Onion	2017-01-13	4

Reset to 0 when a new day begins

Per Day Revenue Totals

ID	Store	Product	Date	Revenue
01	Seattle	Bananas	2017-01-11	7
02	Kent	Apples	2017-01-11	20
03	Bellevue	Flowers	2017-01-11	10
04	Redmond	Meat	2017-01-12	40
05	Seattle	Potatoes	2017-01-12	9
06	Bellevue	Bread	2017-01-12	5
07	Redmond	Bread	2017-01-13	5
08	Issaquah	Onion	2017-01-13	4

Operate on blocks of one day

Calculate aggregations over a window on blocks of records

- running total
- running count

Moving Averages using a Window Function

ID	Store	Product	Date	Revenue
01	Seattle	Bananas	2017-01-11	7
02	Kent	Apples	2017-01-11	20
03	Bellevue	Flowers	2017-01-11	10
04	Redmond	Meat	2017-01-12	40
05	Seattle	Potatoes	2017-01-12	9
06	Bellevue	Bread	2017-01-12	5
07	Redmond	Bread	2017-01-13	5
80	Issaquah	Onion	2017-01-13	4

Calculate the average for the last 3 items sold

	ID	Store	Product	Date	Revenue
(01	Seattle	Bananas	2017-01-11	7
(02	Kent	Apples	2017-01-11	20
C	03	Bellevue	Flowers	2017-01-11	10
(04	Redmond	Meat	2017-01-12	40
(05	Seattle	Potatoes	2017-01-12	9
(06	Bellevue	Bread	2017-01-12	5
(07	Redmond	Bread	2017-01-13	5
(80	Issaquah	Onion	2017-01-13	4

3 previous rows from the current row

ID	Store	Product	Date	Revenue
01	Seattle	Bananas	2017-01-11	7
02	Kent	Apples	2017-01-11	20
03	Bellevue	Flowers	2017-01-11	10
o 4	Redmond	Meat	2017-01-12	40
05	Seattle	Potatoes	2017-01-12	9
06	Bellevue	Bread	2017-01-12	5
07	Redmond	Bread	2017-01-13	5
08	Issaquah	Onion	2017-01-13	4

3 previous rows from the current row

ID	Store	Product	Date	Revenue
01	Seattle	Bananas	2017-01-11	7
o2	Kent	Apples	2017-01-11	20
03	Bellevue	Flowers	2017-01-11	10
04	Redmond	Meat	2017-01-12	40
05	Seattle	Potatoes	2017-01-12	9
06	Bellevue	Bread	2017-01-12	5
07	Redmond	Bread	2017-01-13	5
08	Issaquah	Onion	2017-01-13	4

3 previous rows from the current row

Calculate aggregations over a window of a specific size

Each Order as a Percentage of the Total

ID	Store	Product	Date	Revenue
01	Seattle	Bananas	2017-01-11	7
02	Kent	Apples	2017-01-11	20
03	Bellevue	Flowers	2017-01-11	10
04	Redmond	Meat	2017-01-12	40
05	Seattle	Potatoes	2017-01-12	9
06	Bellevue	Bread	2017-01-12	5
07	Redmond	Bread	2017-01-13	5
08	Issaquah	Onion	2017-01-13	4

Orders for one day, total revenue = 37

ID	Store	Product	Date	Revenue
01	Seattle	Bananas	2017-01-11	7
02	Kent	Apples	2017-01-11	20
о3	Bellevue	Flowers	2017-01-11	10
04	Redmond	Meat	2017-01-12	40
05	Seattle	Potatoes	2017-01-12	9
06	Bellevue	Bread	2017-01-12	5
07	Redmond	Bread	2017-01-13	5
08	Issaquah	Onion	2017-01-13	4

% contribution of o1 = 7/37 * 100 = 18.9%

ID	Store	Product	Date	Revenue
o 1	Seattle	Bananas	2017-01-11	7
o 2	Kent	Apples	2017-01-11	20
о3	Bellevue	Flowers	2017-01-11	10
04	Redmond	Meat	2017-01-12	40
05	Seattle	Potatoes	2017-01-12	9
06	Bellevue	Bread	2017-01-12	5
07	Redmond	Bread	2017-01-13	5
08	Issaquah	Onion	2017-01-13	4

% contribution of o2 = 20 / 37 * 100 = 54%

ID	Store	Product	Date	Revenue
01	Seattle	Bananas	2017-01-11	7
02	Kent	Apples	2017-01-11	20
о3	Bellevue	Flowers	2017-01-11	10
04	Redmond	Meat	2017-01-12	40
05	Seattle	Potatoes	2017-01-12	9
06	Bellevue	Bread	2017-01-12	5
07	Redmond	Bread	2017-01-13	5
08	Issaquah	Onion	2017-01-13	4

% contribution of o3 = 10 / 37 * 100 = 27%

% contribution =

Revenue x 100

Total Revenue

% contribution =

Revenue x 100

sum(revenue) over (partition by day)

Calculate the percentage contribution of each order to a day's revenue

Use the row_number() and rank() window function and see the difference between the two

Use the ntile() window function to see which percentile each record falls into

Summary

Understood windowing functions as a way to write complex queries

Got hands on practice with windowing functions to calculate

- Moving averages
- Percentage contribution
- Percentile for a value