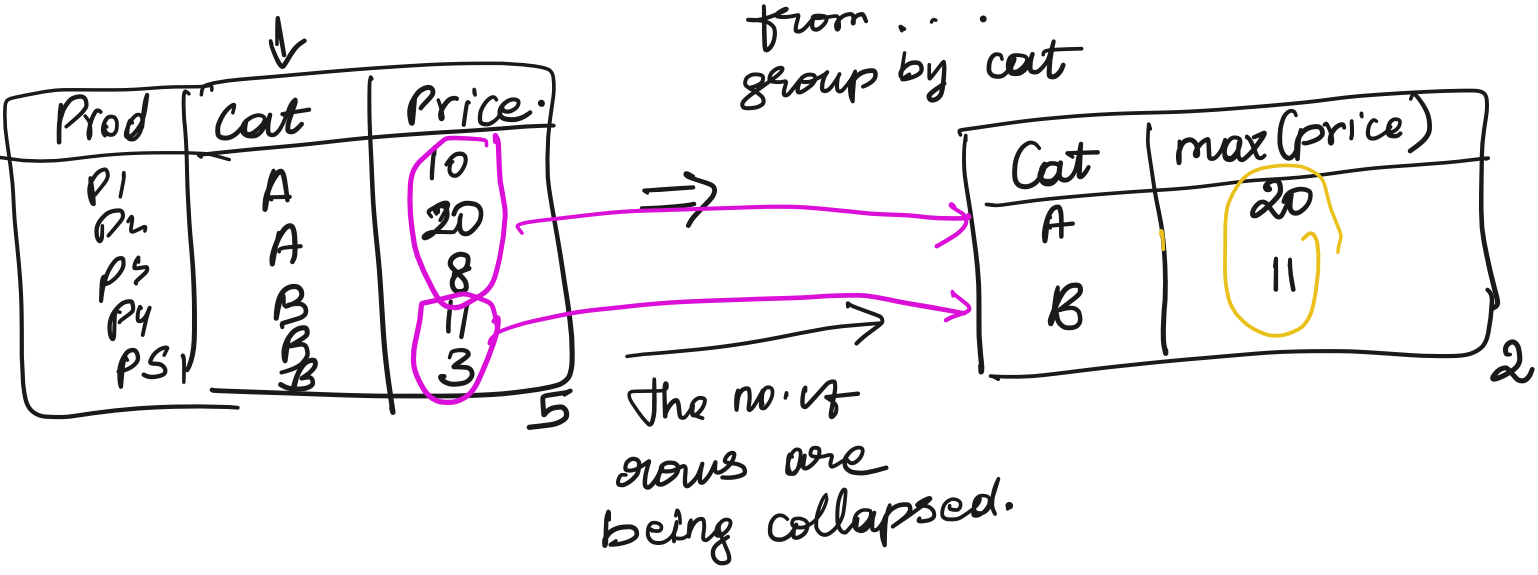
SQL-06 - Window functions.

Recap:

- Group by, Aggregation
- Having.
- Execution order
- Diff count variations.
- Subqueries.

Agenda:

- Window function.



What if I want the agg (max) to be shown in each

row. i.e. The rows should not be ...

Prod	Cat	Price	max(price)
P1	A	10	20
P2	A	20	20
P3	B	8	11
P4	B	11	11
P5	B	3	11

The row-level info is retained, also we got the agg info.

Month	Sales	Sum(sales)	sale / sum(sales)
Jan	10	100	10/100 = 10%
Feb	40	100	40/100 = 40%
Mar	50	100	50/100 = 50%

$$\frac{40}{(10+40+50)}$$

Types of Window function.

- Aggregate — Sum, Count, Min, Max, Avg.
- Ranking — row_number, Rank, DenseRank, NTile
- Value —

What is aggregation → given a series of values return 1 value

[10, 20, 30] ⇒ Sum 60, Count 3, Max 30, Min 10, Avg. 20.

Syntax:

Select *,

Sum( ) over (partition by col1 order by col2)

within which group

order by which

from DB.tbl.

What kind of window fn

sum(Price) over (Partition by vendorid)

Vendor id	Price.
A	100
A	200
A	10
B	20
B	40

Partitioned.

A	100
A	200
A	10

B	20
B	40

Sum(price)

A	100	310
A	200	310
A	10	310
B	20	60
B	40	60

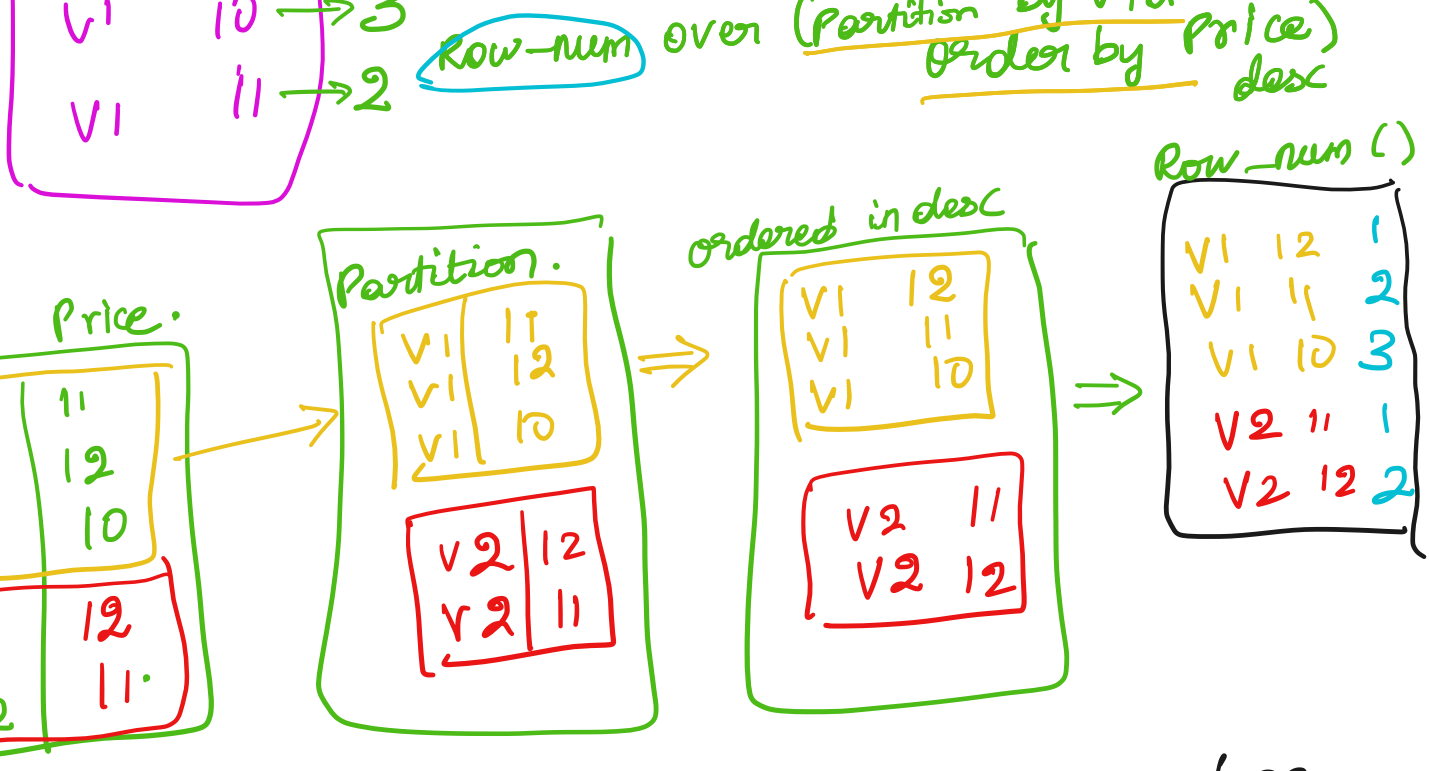
A	100	310
A	200	310
A	10	310
B	20	60
B	40	60

Sum(Price) over ( ) → If no partition is mentioned then entire table is one partition.

Vendor id	Price.
A	100
A	200
A	10
B	20
B	40

A	100	370
A	200	370
A	10	370
B	20	370
B	40	370

Row-num()



Row_number: Create a sequence number

Rank: Creates a Rank based on a column and duplicate values will be

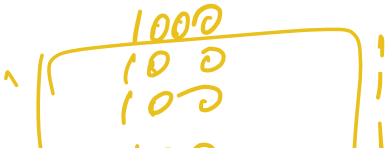
Dense_rank: Rank based on a column, all duplicate value will be ranked same, the next highest is ranked in sequence.

E.salary	Row_number	Rank	D.Rank
100	1	1	1
100	2	1	1
1000	3	3	2
1000	4	3	2

get the 2nd highest salary.

↓

Dense_rank is the right one to use here.



3rd high sal, use D rank

Prod > avg. price within each market date

MD	Vid	Price	Avg. Price	
M1	V1	100	42	100 > 42
M1	V1	50	42	50 > 42
M1	V2	10	42	
M1	V2	20	42	
M1	V2	30	42	
M2	V1	80	40	
M2	V2	20	40	
M2	V2	20	40	

avg(Price) over (partition by MD)

Creating bins.

V1	1
V2	1
V3	2

Ntile

bin = 4

total rows → 9. = 2.25
4

V4 2
V5 3
V6 3
V6 4
V6 4
9 – V6 4

total bins

4 bins of size 2 each

bin → 3.

$\frac{10}{3} = 3$ (3?)

It can be used to create bucket based on price.

price order