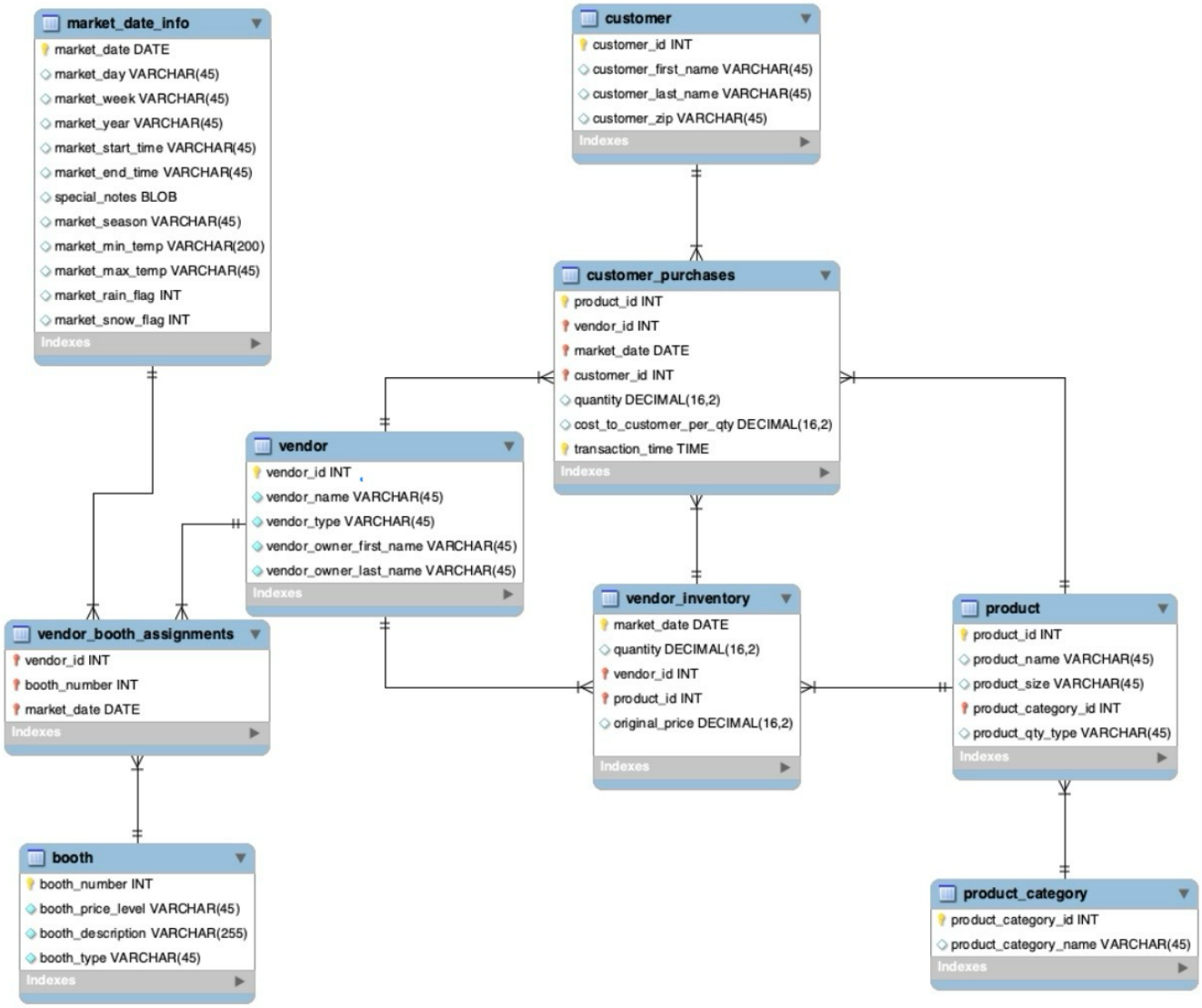
Agenda

1. Need of window function
2. Aggregated window fns
3. Row_Number()
4. Rank(), Dense_Rank()
5. Ntile() Ntile Function
6. Lead(), Lag()
7. Window Frames
8. ROWS vs. RANGE
9. First_value(), Last_value()


market_date_info
market_date DATE
market_day VARCHAR(45)
market_week VARCHAR(45)
market_year VARCHAR(45)
market_start_time VARCHAR(45)
market_end_time VARCHAR(45)
special_notes BLOB
market_season VARCHAR(45)
market_min_temp VARCHAR(200)
market_max_temp VARCHAR(45)
market_rain_flag INT
market_snow_flag INT
Indexes
customer
customer_id INT
customer_first_name VARCHAR(45)
customer_last_name VARCHAR(45)
customer_zip VARCHAR(45)
Indexes
customer_purchases
product_id INT
vendor_id INT
market_date DATE
customer_id INT
quantity DECIMAL(16,2)
cost_to_customer_per_qty DECIMAL(16,2)
transaction_time TIME
Indexes
vendor
vendor_id INT
vendor_name VARCHAR(45)
vendor_type VARCHAR(45)
vendor_owner_first_name VARCHAR(45)
vendor_owner_last_name VARCHAR(45)
Indexes
vendor_booth_assignments
vendor_id INT
booth_number INT
market_date DATE
Indexes
vendor_inventory
market_date DATE
quantity DECIMAL(16,2)
vendor_id INT
product_id INT
original_price DECIMAL(16,2)
Indexes
product
product_id INT
product_name VARCHAR(45)
product_size VARCHAR(45)
product_category_id INT
product_qty_type VARCHAR(45)
Indexes
booth
booth_number INT
booth_price_level VARCHAR(45)
booth_description VARCHAR(255)
booth_type VARCHAR(45)
Indexes
product_category
product_category_id INT
product_category_name VARCHAR(45)
Indexes

Window fns

↳ It gives the ability to put the values from 1 row of data into context compare to a group of rows.

Sum | Min | Max | Avg | Count

↳ aggregation performed at group level but displayed at row level.

→ Amazon (20-21)

Day =1	150 M	150
2	250 M	400
3	70 M	470

DB Cr

↳ Available

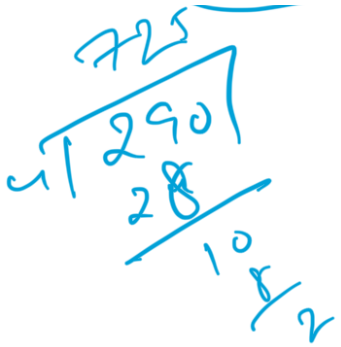
Normalization

2 yrs IT – Narendra

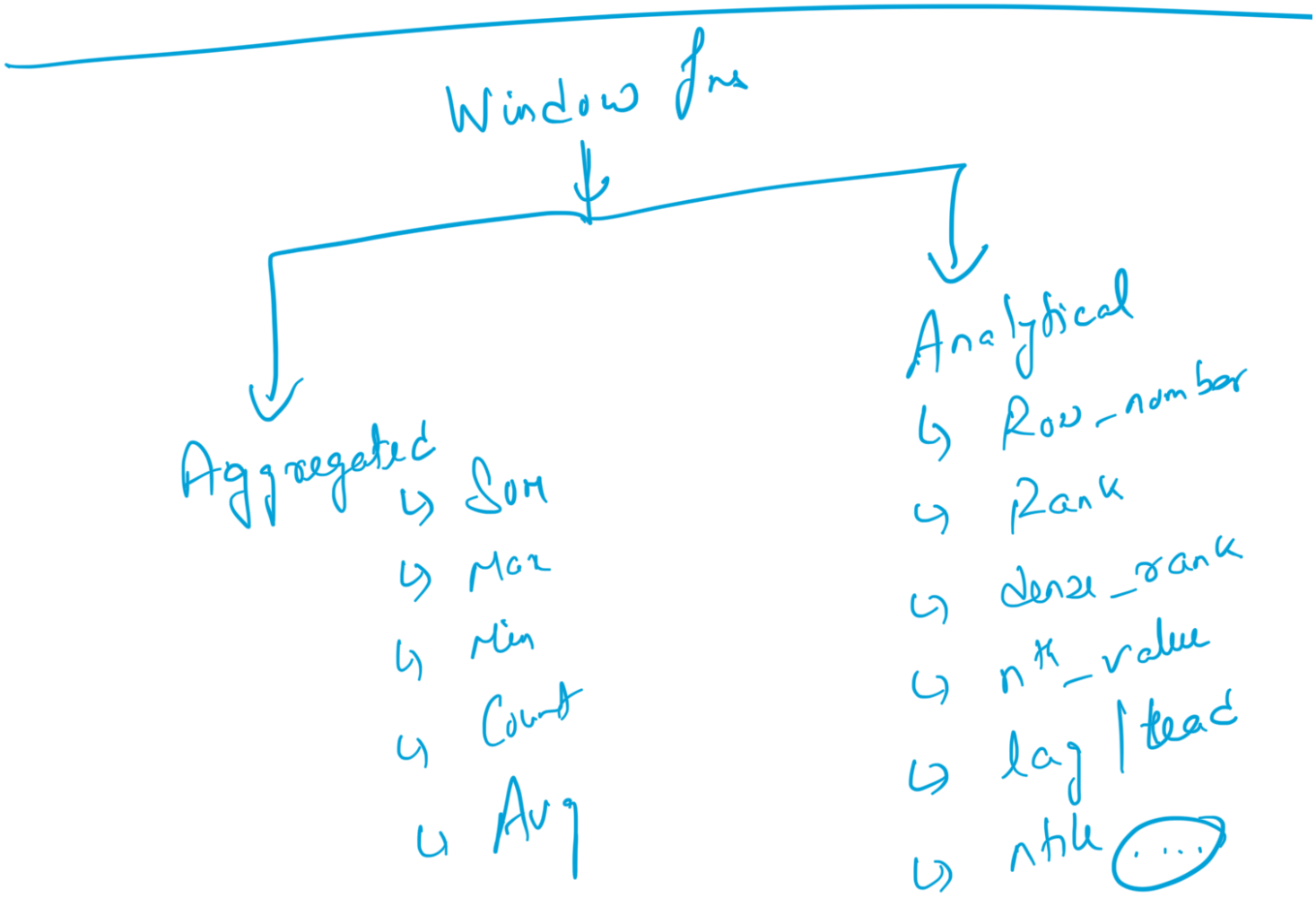
1.8 yrs IT – Rohit

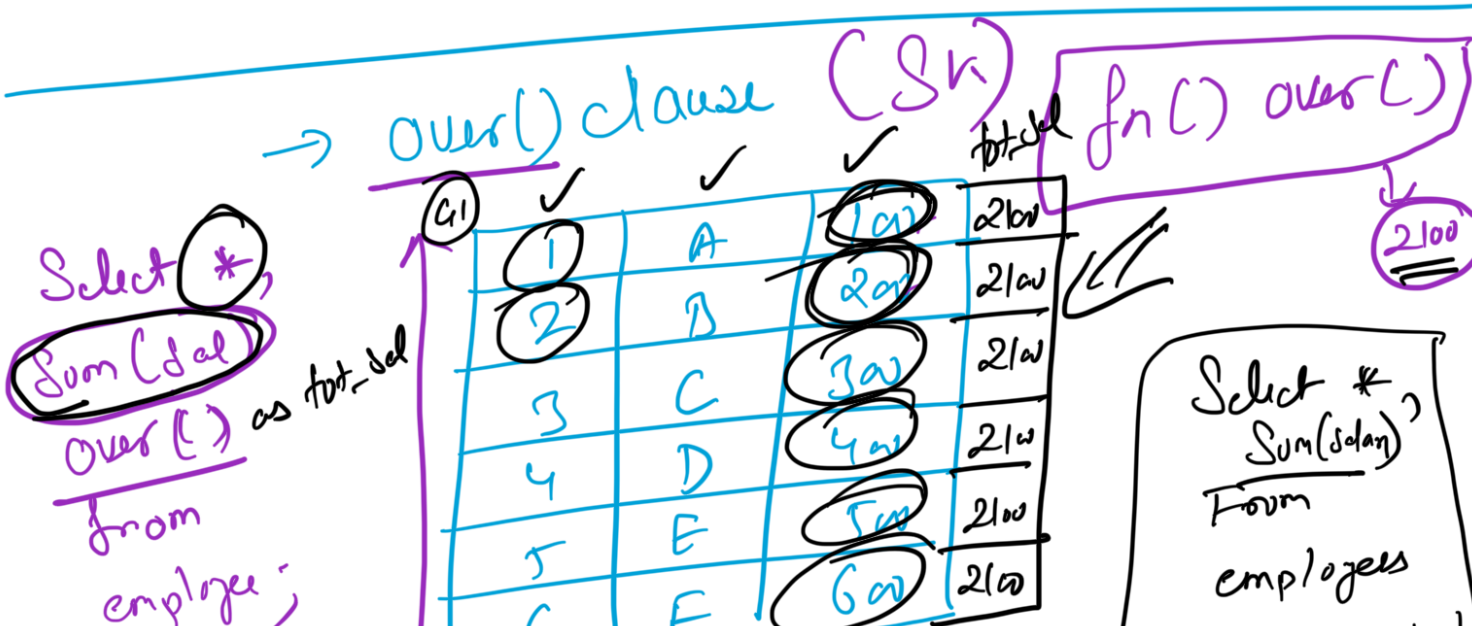
2.2 yrs IT – Raushan

Saurabh


725
290
28
10 & 2

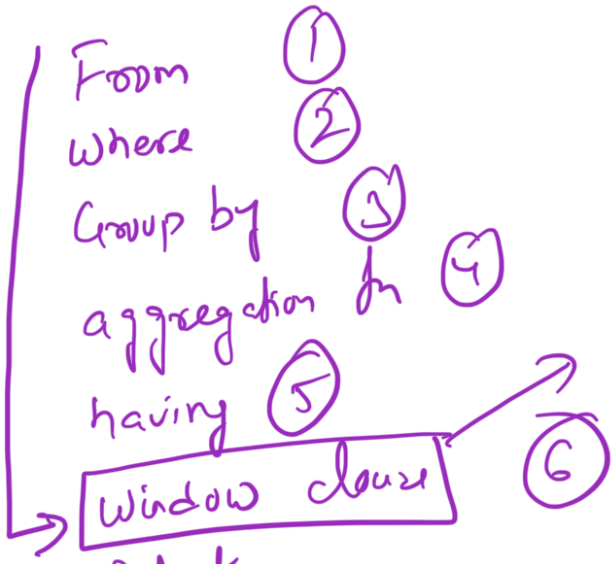
06


Window fns
Aggregated
Sum
Max
Min
Count
Avg
Analytical
Row_number
Rank
Dense_rank
n*_value
lag | lead
ntile ...


→ over() clause (SK)
fn() over()
Select *,
Sum(sal)
over() as tot_sal
from
employee;
tot_sal
2100
Select *
Sum(sal)
From
employees

Group by ic

Execution order



IN
Select clause

over ()

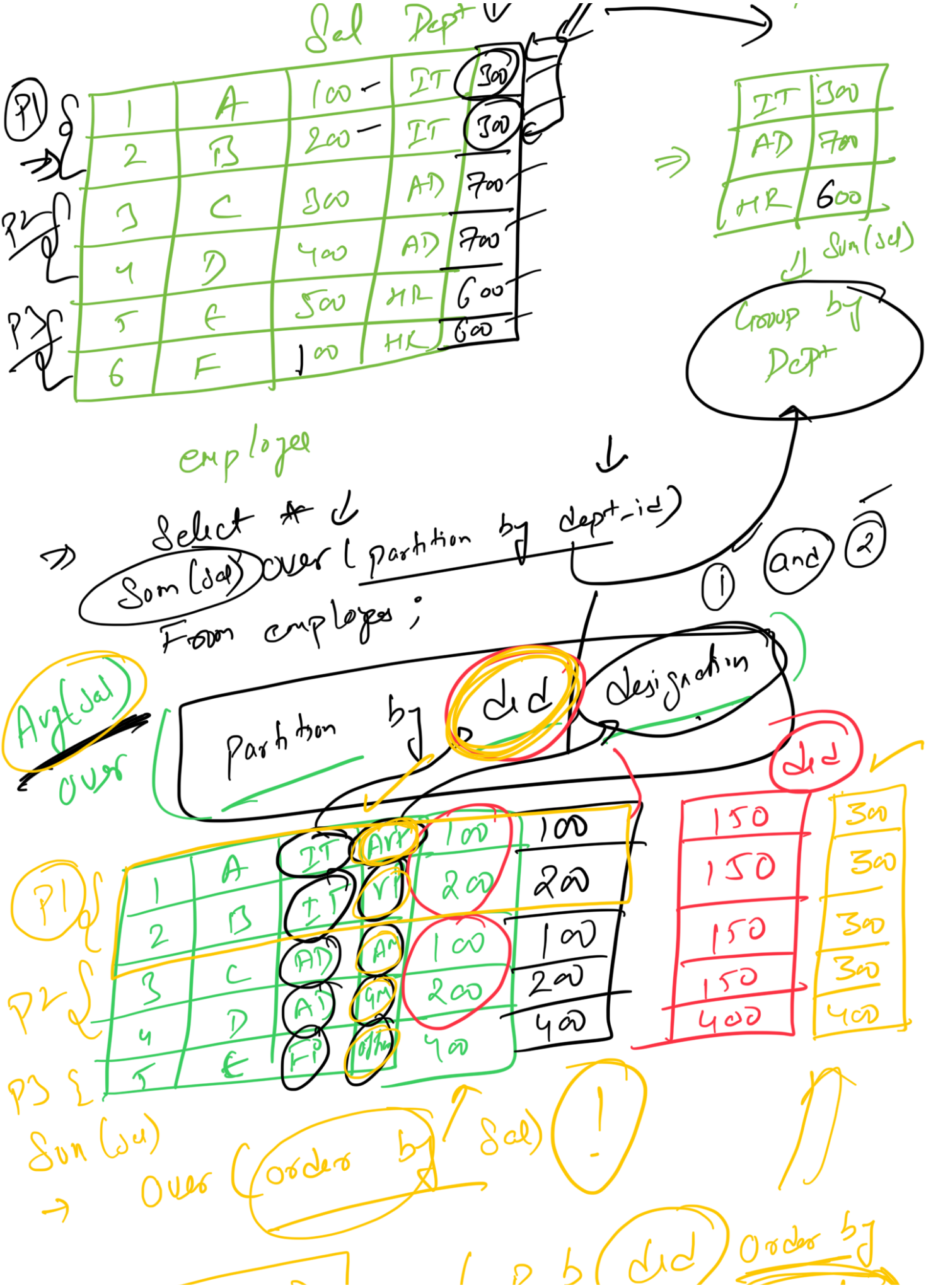
(a) = Partition By (AK)

(b) = order by (SK)

over (a), over (b), over (a,b)

→ over (Partition by col)

Group by → DO Sequence Your table


Sal
Dept
IT
AD
HR
Sum(sal)
Group by
Dept
employee
Select
Sum(sal) over (partition by dept_id)
From employee;
and
Avg(sal)
over
Partition by
did
Designation
Sum(sal)
over (order by sal)

» Sum (sal) over (P [illegible] designation desc)

P1 »

2	200	VP	300
1	100	AVP	300
4	200	GM	300

P2

3	100	AM	300

① Row_Number ()

: assigns a unique increasing no to each row.

② Rank () :

			Rowno	Rank	dense_rank
P	M	100	1	1	1
S	M	100	2	1	1
A	M	99	3	3	2
Aj	M	95	4	4	3
Pooja	M	90	5	5	4