



t1

id	
1	←
1	←
2	
2	
NULL	

t2

id	
1	←
2	←
3	←
NULL	

LEFT →

t1.id	t2.id
1	1
1	1
2	2
2	2
→ NULL	→ NULL

Right

t1.id	t2.id
1	1
1	1
2	2
2	2
→ NULL	← 3
NULL	NULL

Inner

t1.id	t2.id
1	1
1	1
2	2
2	2

Outer

t1.id	t2.id
1	1
1	1
2	2
2	2
NULL	3
NULL	→ NULL
← NULL	NULL

Agenda

- ① Self Joins & Cross Join ←
- ② Correlated Subquery
- ③ Group By & Aggregation fns.

Q. For each employee, find out the name of their manager from the employee.

employee	
emp_id	←
dept_id	
fname	
lname	
mgr_id	←

employee		
emp_id	mgr_id	name
1	3	Tom
2	3	Dick
3	5	Harry
4	2	Harish
5	NULL	Pritesh

Output

emp_id	manager - name
1	→ Harry
2	Harry
3	Pritesh

Li.	Pick
5	NHLL

↑

Self Join

Syntax:

SELECT

→ emp. emp-id ←

→ mgr. name ←

FROM employees AS emp

→ JOIN employees AS mgr

→ ON emp. mgr-id = mgr. emp-id

SELECT

FROM employees AS emp, employees AS mgr

WHERE

CROSS JOIN

4

Car Model
ilo
Suzuki

3

Color
Red
Blue

= 12

SURUS  
Santro

Black

Cross Join Output ↴

Car_model	Color
ilo	Red
ilo	Blue
ilo	Black
Suzuki	h
...	Blu
...	Blu
...	...

SELECT \*

FROM car\_model, color ↴

Q.

Get all the employees who are earning more than the average salary of their department.

① How to calculate their dept. avg. salary. → employee's

Correlated Subquery

SELECT ... id ? ↴

$\rightarrow$   $e1.salary$   
 FROM employees AS  $(e1)$   
 WHERE  $e1.salary > 7$   
 $\rightarrow$  SELECT AVG(Salary)  
 $\rightarrow$  FROM employees AS  $e2$   
 WHERE  $(e1.dept.id = e2.dept.id)$

## GROUP BY

Purchases

date	customer-id	Amt.	Totalamt-perday
21	1	5	14
21	1	2	
21	1	7	
21	2	3	7
21	2	4	
22	1	10	10
22	2	8	8

[ 21 - 1 ]  
 [ 21 - 2 ]  
 [ 22 - 1 ]

1st group  $\rightarrow$  3  
 Aggregation (contig)

Syntax: SELECT

SELECT  
 → SUM(CANT) As total-cost  
 FROM  
 → WHERE  
 → GROUP BY <sup>1, 2, 3</sup> market-date, cust-id  
 HAVING SUM(CANT) > 10  
 ORDER BY  
 LIMIT

Q. Get a list of customers who made purchases on each date.

SELECT DISTINCT ✓ Expensive Operation  
 c-id  
 date

FROM

SELECT  
 cust-id ✓  
 market-date }

FROM purchases

GROUP BY (cust-id, market-date)

LOD

number of purchases

d. — Count ~~the~~ <sup>0</sup> ~~made~~ on each date.  
each customer

SELECT year,  
c-id,

date  
→ quantity  
~~product-id~~  
→ count(\*) AS num-purchases  
→ SUM(Ant) AS total

FROM

GROUP BY c-id, date, ~~product-id~~

↓ 21-1 → 2018

year	date	c-id	p-id	Ant	totalAnt	qty
→ 2018	21	1	1	5		1
→ 2018	21	1	2	7		4
→ 2018	21	1	3	10		3

21	1	3 records	? error
----	---	-----------	---------

21 - 1 - 1  
21 - 1 - 2  
21 - 1 - 3

↓  
Number of different



Products bought

Handwritten diagram illustrating a table structure and data flow:

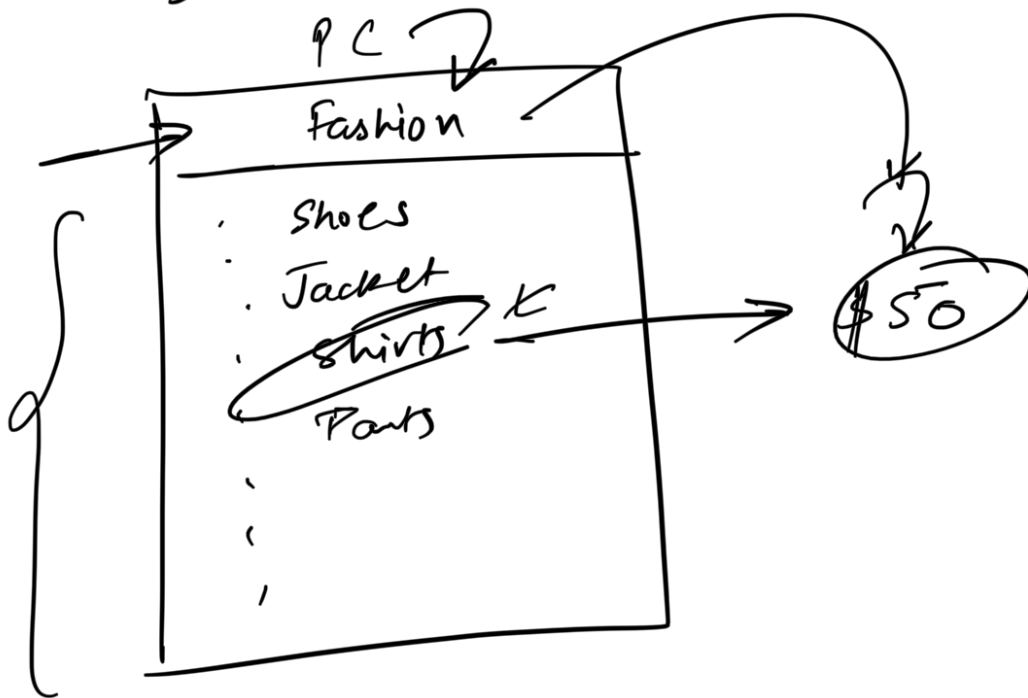
- A table with two columns: `epid` and `count (abc)`.
- The `count (abc)` column contains vertical ellipsis (`...`) indicating multiple rows.
- Arrows point from the left towards the `count (abc)` column, with a bracket and the text `HIO` below them.
- A checkmark and the text `COUNT (1)` with a downward arrow are positioned above the table.

Q Calculate the total amount paid by each customer to each vendor till date.

Q. Get the least & most expensive product per product category.

COUNT  
AVG  
SUM  
MIN  
MAX

MIN  
MAX



Q. filter out vendors who brought at least market between

