SQL-05 Group By.

Recap:

- Inequi
  - >=, <>, between
- Sub-query.

Agenda:

- Group By
- Having.
- Order of execution.

Usually top management want aggregated data

- Sum of monthly sales
- Total new customer
- Avg. sales / day
- Customer with high purchase.

Cus. id.	Date	Amt
A	1	10
A	1	20
B	4	60
A	2	15
B	3	5

⇒

Cus. id	sum(Amt)
A	45
B	65.

↳ This is called aggregation

To do this we use a clause called "Group by"

select , where, from, limit, offset, order by.

select
from
where
Group by
Order by
Limit
Offset

Find sales of each customer

Cus. id.	Date	Amt
A	1	10
A	1	20
B	4	60
A	2	15
B	3	5

⟹

Cus.id	sum(Amt)
A	45
B	65.

← 2.

Syntax:

```
Select
    cus_id,
    sum(Amt)
from DB.tbl
Group by
    cus_id
```

⇒ Give me the sales of each customer.

Find all sales of each customer, each

Find the overall ... day.

Cus. id.	Date	Amt
A	1	10
A	1	20
B	4	60
A	2	15
B	1	5

⇒

Cus id	Date	Sum(Amt)
A	1	30
A	2	15
B	4	60
B	1	5.

← 4

how the query changes :

```
select
    cus_id, date, sum(Amt) as overall
from DB.tbl
group by
    cus_id, Date
```

- Why all columns that are not grouped should not be in the select :

Cus.id	P.id	Amt
A	P1	10
A	P2	15
A	P4	20
B	P10	5

⇒

```
Select cus_id
    sum(Sale)
from
group by
    cus_id
```

⇒

A	45
B	5

↳

```
select cus_id, Prod_id
    sum(Amt)
from T
group by cus_id
```

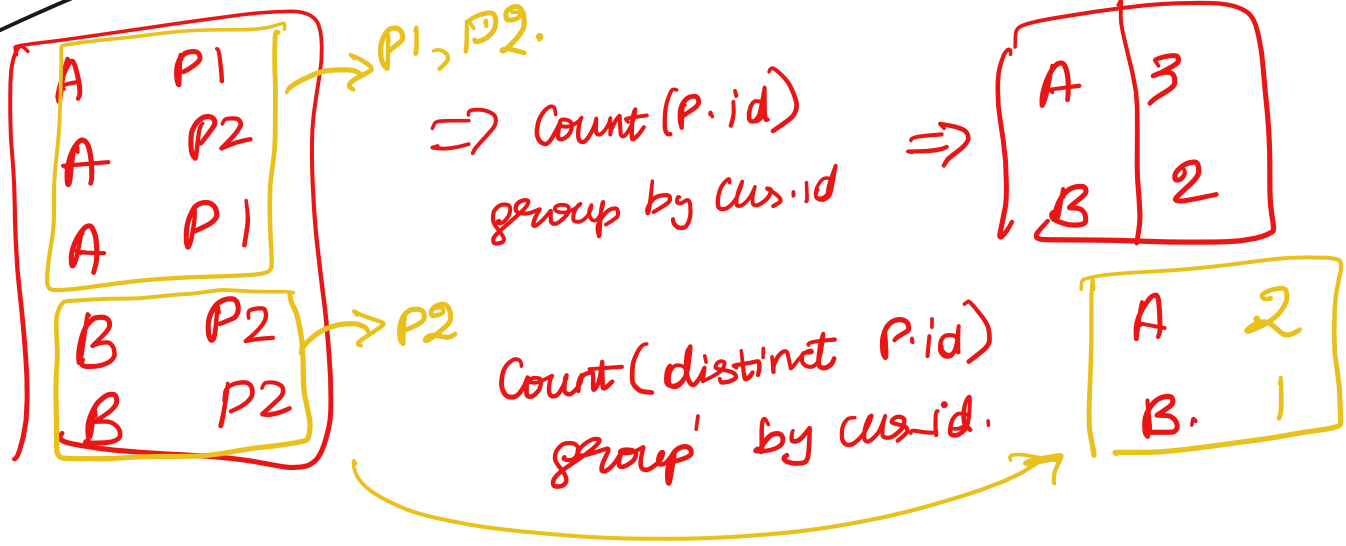
Cus_id		Sale)
A	P1	45
B	P2	5
	P3	
	P10	

Count:

Cus.	Pr.id
A	P1
A	P3
B	P1

A → 2
B → 1

count distinct



Count (*) → count all the rows → including NULL

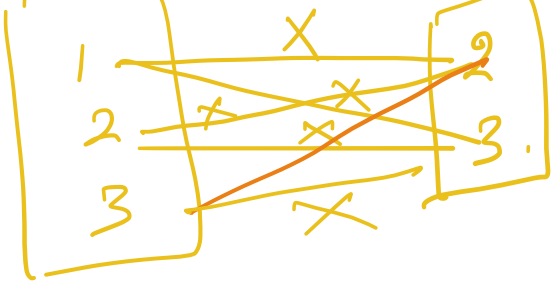
Count (a column_name) → count all rows in the column → excluding NULL

count (1) → creates a column with all 1 and then counts

Count (distinct col) → count only uniq values in the column → excluding Null

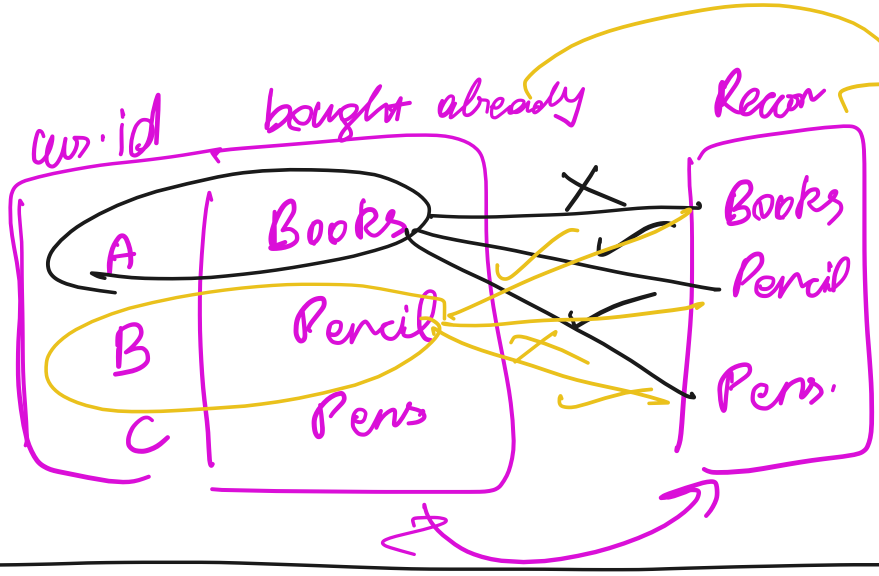
---

Quiz question:



on
T1 > T2.

1 row.



T1 <> T2.

bough already <> Reco

Min and Max.

Vendor	Product	Amt
V1	DrA	10
V2	A	14
V2	B	8
V4	B	24.

→

cheapest	costliest
8	24.

```
select
  min(Amt) as cheapest,
  max(Amt) as costliest
from DB.tbl.
```

Order of execution:

- FROM including Join. → table alias is available
- where
- Group by
- Having
- Window fn
- Select → column alias is created here.
- Distinct
- Union
- Order by
- Offset
- Limit

It is because of the order of execution than we cannot use the column alias in group by, where clause.

i.e: select (qty*price) as cost from tbl where price>10 X

instead use subqueries

```
select * from
(select (qty*price) as cost from tble) x
where cost >10.
```

It is the same reason we cannot use the aggregate function

use the aggregate
in the where clause as group by
comes after where clause.

i.e. where count(prod_id>10) ✗

instead

Having count(prod_id>10) ✓