COMPUTER SCIENCE

Database Management System

Query Language

Lecture_4

Vijay Agarwal sir

TOPICS TO BE COVERED

01 SQL Clauses

02 SQL Operators

Relational Algebra.

Selection [σ]

Projection (π]

Cross Product [X)

Union [$\cup$]

Set Difference [$-$]

Rename [ρ]

Intersection [$\cap$)

JOIN ($\bowtie$) & its type

Division [/]

SEQUEL $\rightarrow$ **SQL** : [Structured Query Language]

SE $\downarrow$ English

R.A

SQL
SELECT [DISTINCT] $A_1, A_2, A_3 \ldots A_n$ $\equiv$ Projection [π]
FROM $R_1, R_2, R_3 \ldots R_m$ $\equiv$ CROSS Product [X]
[WHERE Condition] $\equiv$ Selection [σ]

R.A : $$\pi_{A_1, A_2 A_3 \ldots A_n}\left[\sigma_{Condition}\left(R_1 \times R_2 \times R_3 \ldots \times R_m\right)\right]$$

SQL[Structured Query Language]

- DDL(Data Definition Language): Modification allowed at schema (Definition) level
  - CREATE
  - ALTER
  - DROP TABLE
- DML(Data Manipulation Language): Modification allowed at data level
  - INSERT
  - UPDATE
  - DELETE
- DCL(Data Control Language): Control Transactional Operation
  - COMMIT
  - ABORT
- DQL(Data Query Language): Used to Retrieve the Data from DB
  - SELECT
  - FROM
  - WHERE

STUDENT As S

Catalog C, Parts As P

Rename operator

As

SPACE

SQL R.A

SELECT [DISTINCT] $A_1\ A_2\ A_3\ A_n ...$ ≡ Projection (π)

FROM $R_1\ R_2\ R_3\ .....\ R_m$ ≡ CROSS Product (x)

WHERE Condition ≡ Selection [σ]

R.A: $\pi_{A1A2A3..An}[\sigma_{Condition}(R_1 \times R_2 \times R_3 \ldots \times R_m)]$

Select: Not going to eliminate Duplicate Value.

1) SELECT AB
FROM R

$\xrightarrow{\text{Output}}$

A	B
1	2
1	2
2	4

R(A B C)

A	B	C
1	2	3
1	2	4
2	4	5

2) $\boxed{\pi_{AB}(R)}$ $\longrightarrow$

A	B
1	2
2	4

3) SELECT [DISTINCT] AB
FROM R

$\xrightarrow{\text{Output}}$

A	B
1	2
2	4

SQL Clause :

2 Mandatory clause
$$\left\{\begin{array}{l}\text{SELECT}\\ \text{FROM}\end{array}\right.$$

```
SELECT  Sname
FROM    Student
```

SQL Clauses

[ ]

SELECT [DISTINCT] A_1 A_2 A_3 ...A_n

FROM R_1 R_2 R_3...R_m

[WHERE P]

[GROUP By Attribute [[HAVING Condition]]]

[ORDER By Attribute [[DESC]]]

[ ] ← Optional clause

FROM
WHERE
GROUP By
Having
SELECT
DISTINCT
ORDER By

Execution sequence :

$$\begin{tikzcd}
\text{FROM} \arrow[d] \\
\text{WHERE} \arrow[d] \\
\text{GROUP By} \arrow[d] \\
\text{HAVING} \arrow[d] \\
\text{SELECT} \arrow[d] \\
\text{DISTINCT} \arrow[d] \\
\text{ORDER By}
\end{tikzcd}$$

```
SELECT Sname
 FROM  Student
 WHERE  CGPA > 8
```

Query Execution

(1) FROM Clause: It is the first executable Clause. It just simply Relation (or) CROSS Product of Two or more Relation

(2) WHERE Clause: It is the second executable clause. It selects the tuple based on specified condition.

(3) GROUP By Clause: It is the third executable clause if used in the query. It groups the table based on the specified attributes.

STUDENT

Sid	(Branch)	Marks
S_1	CS	90
S_2	IT	70
S_3	CS	70
S_4	EC	56
S_5	CS	NULL

$\xrightarrow{\text{GROUP By (Branch)}}$

Sid	(Branch)	Marks
S_1	CS	90
S_3	CS	70
S_5	CS	NULL
S_2	IT	70
S_4	EC	56

follow order of insertion
if ties.

Aggregate operator :

$\hookrightarrow$ Always Discard the NULL Value

1. Count
2. SUM
3. AVG
4. MIN
5. MAX

STUDENT

Sid	(Branch)	Marks
S_1	CS	90
S_2	IT	70
S_3	CS	70
S_4	EC	56
S_5	CS	NULL

GROUP By (Branch) →

Sid	(Branch)	Marks
S_1	CS	90
S_3	CS	70
S_5	CS	NULL
S_2	IT	70
S_4	EC	56

follow order of insertion if ties.

Aggregation operator ⇒ Always Discard Null Value

1) COUNT ([DISTINCT] Attribute)
2) SUM ([DISTINCT] Attribute)
3) AVG ([Distinct] Attribute)
4) MIN (Attribute)
5) MAX (Attribute)

$$\left.\begin{array}{l} \text{Count}(*) \\ \text{Count}[\text{Constant}] \end{array}\right\} \Rightarrow \overset{\text{Return}}{\#\text{Tuples}}$$

1) Count(marks) = 4
2) Count (*) = 5 $\longrightarrow$ # Tuples
3) Count ([DISTINCT]marks) = 3
4) SUM(marks) = 286
5) SUM([Distinct]marks) = 216
6) AVG(marks) = $\frac{286}{4}$
7) AVG([Distinct]marks) = $\frac{216}{3}$

Aggregation operator ⇒ Always Discard Null Value

1) COUNT ([DISTINCT] Attribute)

2) SUM ([DISTINCT] Attribute)

3) AVG ([Distinct] Attribute)

4) MIN (Attribute)

5) MAX (Attribute)

1) Count(marks) = $\underline{4}$

2) Count (*) = $\underline{5}$

3) Count ([DISTINCT]marks) = $\underline{3}$

4) SUM(marks) = $\underline{286}$

5) SUM([Distinct]marks) = $\underline{216}$

6) AVG(marks) = $\frac{286}{4}$

7) AVG([Distinct]marks) = $\frac{216}{3}$

$$\boxed{\frac{\text{SUM[DISTINCT]marks}}{\text{COUNT[DISTINCT]marks}}} \Rightarrow \frac{216}{3}$$

Patience.

Having : Condition Applied on each group.

Condition Applied over Aggregate function ⓞ special defind function [Some ⓞ every] (eg)

```
Select A
FROM R
Group By (A)
Having B>25 ✗
```

Having Avg(B) >25 ✓ (Aggregate function) ⓞ HAVING Some(B) >25 ✓

HAVING

SQL Standard

(*)

Having Condition Used with Group By Clause.

Having Condition Applied on every group.

if Group By is Not Present then Having Condition Applied for each Record. (Tuple).

```
SELECT Name
FROM Student
WHERE CGPA>8.
```

$\Downarrow$

Name

STUDENT (RollNo, Name, Gender Branch)

```
Select *
FROM Student
WHERE CGPA > 8
```

$\Downarrow$ O/P.

RollNo	Name	Gender	Branch

HAVING: Fourth executable clause (if used in query).

It is used to select the group which satisfy the condition (condition is for each group).

STUDENT

Sid	Branch	Marks
S_1	CS	60
S_2	IT	70
S_3	CS	90
S_4	IT	60
S_5	EC	55
S_6	EC	NULL

(GROUP By Branch) →

Sid	Branch	Marks
S_1	CS	60
S_3	CS	90
S_2	IT	70
S_4	IT	60
S_5	EC	55
S_6	EC	NULL

$\frac{150}{2}$ ✓

$\frac{130}{2}$ ✓

$\frac{55}{1}$

Select * FROM STUDENT
GROUP By (Branch)
HAVING AVG(Marks) > 61

Sid	Branch	Marks
S_1	CS	60
S_3	CS	90
S_2	IT	70
S_4	IT	60

Simplicity

& Step by step Procedure.

Q.1 Select min(marks)

FROM Student

Sid	Branch	Marks
S_1	CS	60
S_2	IT	70
S_3	CS	90
S_4	IT	60
S_5	EC	55
S_6	EC	NULL

$\boxed{55}$

ANSWER: 55

Q.2 Select min(marks)

FROM Student

WHERE Branch = 'CS'

Sid	Branch	Marks
S_1	CS ✓	60 ✓
S_2	IT	70
S_3	CS ✓	90 ✓
S_4	IT	60
S_5	EC	55
S_6	EC	NULL

$\boxed{60}$

ANSWER: 60

Q.3

```
Select min(marks), Branch
FROM Student
GROUP By (Branch)
```

Brn   marks

Sid	Branch	Marks
S_1	CS	60
S_2	IT	70
S_3	CS	90
S_4	IT	60
S_5	EC	55
S_6	EC	NULL

CS	60
IT	60
EC	55

Q.3 Select min(marks) , Branch

FROM Student

GROUP By (Branch)

```
60
60
55
```

Sid	Branch	Marks
S_1	CS	60
S_2	IT	70
S_3	CS	90
S_4	IT	60
S_5	EC	55
S_6	EC	NULL

Q.4 Select min(marks), Branch

FROM Student

$\times$ 55 CS
IT
CS
IT
EC
EC

Q.4 Select min(marks), Branch

FROM Student

➡ Such Syntax is not allowed in SQL

CS
IT
CS
IT
EC
EC

Note When Aggregate operator & other Attribute used in select clause, is allowed Only if other Attribute Must be in Group By clause.

NOTE:

When aggregate operator & other Attribute used in select clause is Allowed only of other attribute must be in Group of Clause.

```
Select min (marks) Branch
FROM Student
GROUP By (Branch)
```

(According to SQL standard)

Group By clause

① Every attribute of Group By clause Must Present in Select clause.

② Not allowed to Select (Present) other Attributes in Select clause.

③ allowed to Select (Present) Aggregate function In Select Clause.

Group By clause

~~①~~ X

```
SELECT Sid   → sname
FROM Student
GROUP By (sname)
```

~~②~~ X

```
SELECT Sid → Sname missing
FROM Student
GROUP By (Sid Sname)
```

~~③~~

```
SELECT Sname   → Sid missing
FROM Student
GROUP By (Sid Sname)
```

~~④~~

```
SELECT A, min(B)  → B
FROM R
GROUP By (B)
```

ALL are Incorrect.

Aggregate function ↓ ✓

✓

```
SELECT  A, min(B)
FROM    R
GROUP BY (A)
```

CS

IT

EC

✗

```
SELECT  A, B
FROM    R
GROUP BY (A)
```

Set operator

①	UNION	UNION ALL
②	INTERSECT	INTERSECT ALL
③	MINUS	MINUS ALL
	↑ Supported by R.A	⇑ Not Supported by R.A

Q.

```
Select min(A), B
FROM R
Group By (C) ✗
Group by (B) ✓
```

OTHER	Set Operator
Followed by R.A ↓	Not followed By R.A↓

1) UNION/ UNION ALL

2) INTERSECT/ INTERSECT ALL

3) MINUS / MINUS ALL

R	S
1	1
1	1
1	2
2	4
2	5
4	5
6	
6	

① R UNION S

R UNION ALL S

② R INTERSECT S

R INTERSECT ALL S.

③ R MINUS S

R MINUS ALL S.

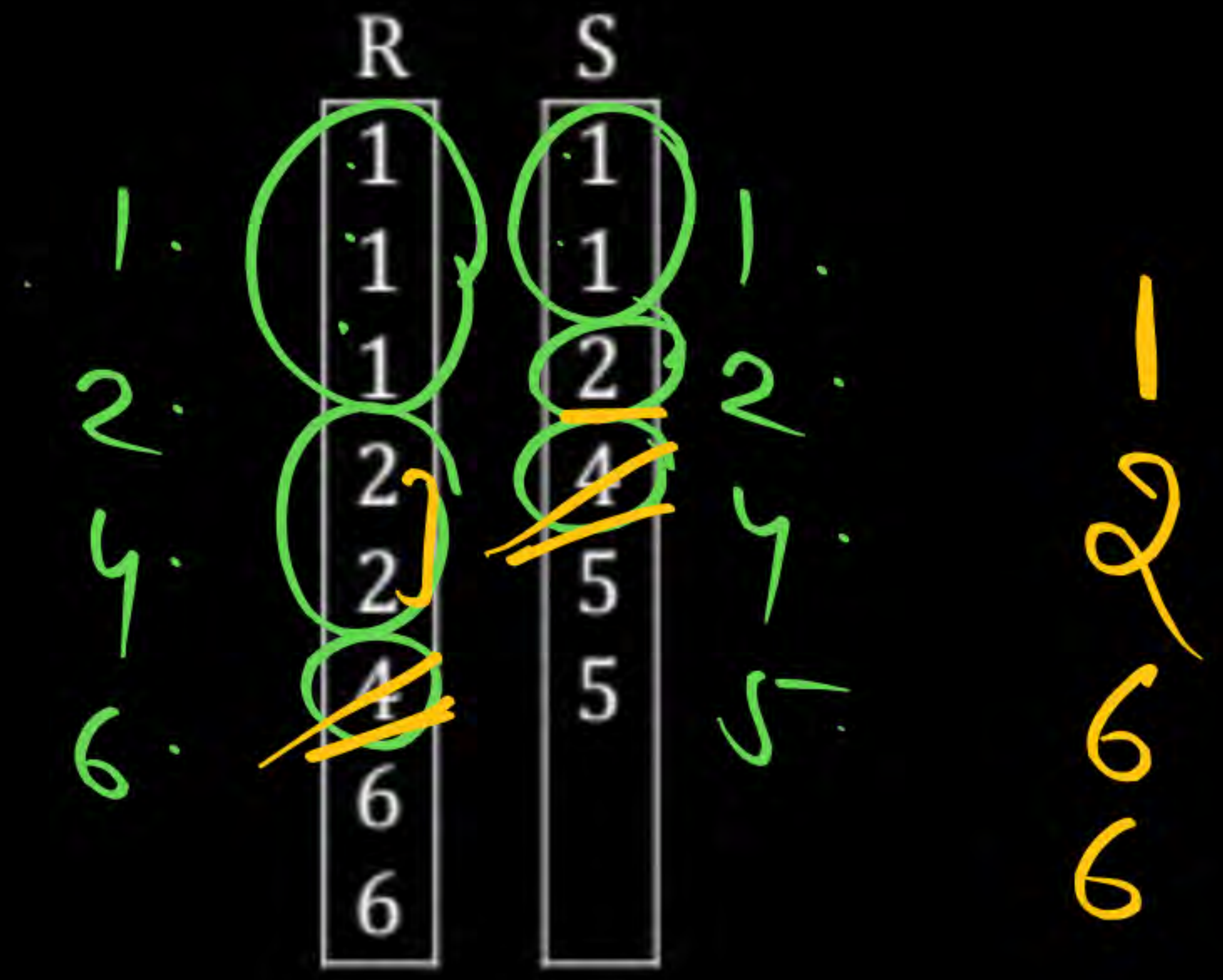
Q. Select min(A), B
FROM R
Group By (C) ✗
Group by (B) ✓

OTHER	Set Operator
Followed by R.A ↓	Not followed By R.A↓

1) UNION/ UNION ALL

2) INTERSECT/ INTERSECT ALL

3) MINUS / MINUS ALL



R	S
1	1
1	1
1	2
2	~~4~~
2	5
~~4~~	5
6	
6	

1
2
6
6

1) R UNION S

$\hookrightarrow$ Result Distinct tuple

1
2
4 ✓
6
5

2) R UNION ALL S

$\hookrightarrow$ Result all values

1
1
1
2
2
4
6
6
1
1
2
4
5
5

3) R INTERSECT S

$\hookrightarrow$ Distinct Common tuples from R & S

1
2
4

✓

4) R INTERSECT ALL S

$\hookrightarrow$ How many maximum number of times Common in both R & S

1 ✓
1 ✓
2 ✓
4 ✓

4) R MINUS S

→ Distinct tuples from R those are not there in S

→ | 6 | ✓

5) R MINUS ALL S

→ # Duplicates in R − # Duplicates in S

→

1 ✓
2 ✓
6 ✓
6 ✓

THANK YOU!