COMPUTER SCIENCE

Database Management System

Query Language

Lecture_6

Vijay Agarwal sir

TOPICS TO BE COVERED

01 Queries

02 SQL Clauses

SQL : Non Procedural / Declarative

SELECT [DISTINCT] ≡ Projection [π]

FROM ≡ CROSS Product [X]

WHERE ≡ Selection [σ]

SQL Clause	Execution Sequence
FROM	→ i/p → Relation or CROSS Product of Relation.
WHERE	↓
GROUP BY	↓
HAVING	→ Select → Distinct → ORDER BY.

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

Difference b/w SELECT (SQL) & π (in RA) ?

Important Point

① In SQL (key words) is Not Case Sensitive.

But String's Case Sensitive.

sid	Branch
	CS
	IT

Q Branch = 'it' ✗

Branch = 'IT' ✓

② SQL is Non Procedural | Declative Query language But Relation Algebra Formal | Procedural Query language.

Important Point

WHY SQL is Most Widely Used ?

∵ Relational Algebra : 'Set' Theory.

SQL : Multi Set ('Bag') [Duplicate Allowed]

Every Set is a MultiSet But Every Multiset is Not a Set

eg Set : [1, 2, 3, 4]

Multiset : [1, 1, 2, 2, 3, 3, 3, 4, 4]

Select: Not going to eliminate Duplicate Value.

1) SELECT AB FROM R → Output

A	B
1	2
1	2
2	4

Multi set (Bag)

R(A B C)

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

2) $\pi_{AB}(R)$ →

A	B
1	2
2	4

3) SELECT [DISTINCT]AB FROM R → Output

A	B
1	2
2	4

Important Point

WHY SQL is Most Widely Used ?

Commerical Database Works on Multiset (Bag).

WHY ?

① If we want to Perform union of Two Table (Relation), SQL just simply One table & Simply add the Data of another Table (∵ its Fastest)

But in Relation first we eliminate Duplicates then Perform operations. (very time consuming)

Important Point

WHY SQL is Most Widely Used ?

Commerical Database Works on Multiset (Bag).

WHY ?

Sid	Marks
S1	9
S2	10
S3	9
S4	10

In RA eleminates the Duplicate so Can not Find Average.

But in SQL $\frac{9+10+9+10}{4}$

Removing Duplicates in very expensive.

Important Point

- In SQL Aggregate function are Allowed but its Not allowed in Relational Algebra.
- SQL have Lots of features that Not Supported by Relational Algebra.

Important Point

Rename operator / Alias

keyword

AS, SPACE

eg

FROM Supplier AS S, Parts P

SELECT Sname As Name
SupplierId As Id

Supplier (Sid, Sname, Rating)
Parts (Pid, Pname, Color)
Catalog (Sid Pid Cost)

RR (Raghu Rama Krishna)
Sailors (Sid, S———)
Reserve (Sid bid
Boat (bid, bname

Important Point

Q) WAQ to Find Sid of Red Color Parts?

RA:

$$\Pi_{sid}\left[\sigma_{\substack{C.Pid = P.Pid \\ \wedge P.Color = Red}}\left(Parts \times Catalog\right)\right]$$

SQL

```
SELECT sid
FROM Catalog Parts
WHERE Catalog.Pid = Parts.Pid
AND Parts.Color = 'Red'
```

Supplier (Sid, Sname, Rating)

Parts (Pid, Pname, Color)

Catalog (Sid Pid Cost)

RR (Raghu Rama Krishna)

Sailors (Sid, S———)

Reserve (Sid bid

Boat (bid, bname

Important Point

Q) WAQ to Find Sid of Red Color Parts?

RA:

$$\Pi_{sid}\left[\sigma_{\substack{C.Pid = P.Pid \\ \wedge P.Color = Red}}\left(Parts \times Catalog\right)\right]$$

Rename operator

SQL

```
SELECT C.sid
FROM Catalog ASC, Parts ...P
WHERE C.Pid = P.Pid AND
      P.Color = Red
```

Supplier (Sid, Sname, Rating)

Parts (Pid, Pname, Color)

Catalog (Sid Pid Cost)

RR (Raghu Rama Krishna)

Sailors (Sid, S———)

Reserve (Sid bid

Boat (bid, bname

Important Point

Q) WAQ to Find Sid of Red Color Parts?

RA:

$$\Pi_{sid}\left[\sigma_{\substack{C.Pid = P.Pid \\ \wedge P.Color = Red}}\left(Parts \times Catalog\right)\right]$$

SQL

```
SELECT C.Sid As krishna.
FROM Catalog ASC, Parts ...P
WHERE C.Pid = P.Pid AND
      P.Color = Red
```

krishna

Supplier (Sid, Sname, Rating)

Parts (Pid, Pname, Color)

Catalog (Sid Pid Cost)

RR (Raghu Rama Krishna)

Sailors (Sid, S———)

Reserve (Sid bid

Boat (bid, bname

Important Point

Q2 WAQ to Find Sname of Red Color Parts?

```
SELECT  S.Sname
FROM Supplier AS S, Part P, Catalog AS C
WHERE   P.Pid = C.Pid   AND
        C.Sid = S.Sid   AND
        P.Color = Red
```

Supplier (Sid, Sname, Rating)

Parts (Pid, Pname, Color)

Catalog (Sid Pid Cost)

RR (Raghu Ramakrishna)

Sailors (Sid, S———)

Reserve (Sid bid

Boat (bid, bname

WHY Aliasing/Renaming Play a Role ?

Important Point

Course (Sid, Cid, Contact)

Q) Find the student who Participate At least one Game? / Some / Any Game?

GAME (RollNo, Gid) → Game Id

$\pi_{RollNo}(GAME)$

RollNo
1
2

GAME

RollNo	Gamid
1	G_1
1	G_2
2	G_1

G_1: Cricket

G_2: Football.

Important Point

Course (Sid, Cid, Contact)

Q) Find the student who Participate At least Two Game?

GAME (RollNo, Gid) → Game Id

Renaming/Aliasing

$\rho(T_1, GAME)$, $\rho(T_2, GAME)$

$$\Pi_{T_1.RollNo}\left[\sigma_{T_1.RollNo = T_2.RollNo \,\wedge\, T_1.Gamid \neq T_2.Gamid}(T_1 \times T_2)\right]$$

GAME

RollNo	Gamid
1	G_1
1	G_2
2	G_1

G_1: Cricket

G_2: Football.

Important Point

Course (Sid, Cid, Contact)

Q Find the student who Participate At least Two Game?

GAME(RollNo, Gid) → Game Id

$T_1 \times T_2$:

T_1RN	T_1Gid	T_2RN	T_2Gid
1	G_1	1	G_1
1	G_1	1	G_2
1	G_1	2	G_1
1	G_2	1	G_1
1	G_2	1	G_2
1	G_2	2	G_1
2	G_1	1	G_1
2	G_1	1	G_2
2	G_1	2	G_1

O/P.

RollNo
1

Ans

T_2

RollNo	Gid
1	G_1
1	G_2
2	G_1

GAME(T_2)

RollNo	Gamid
1	G_1
1	G_2
2	G_1

G_1: Cricket

G_2: Football.

Important Point

Course (Sid, Cid, Contact)

Q Find the student who Participate At least Three Game ?

GAME (RollNo, Gid) → Game Id

$\rho(T_1, Game)$ $\rho(T_2, Game)$ $\rho(T_3, GAME)$

Non key

Gid	Gid	Gid
T_1	T_2	T_3
2	1	2

Total → Two Game

$$\pi_{T_1.RollNo}\left(\sigma_{T_1.RNo = T_2.RNo = T_3.RNo \,\wedge\, T_1.Gid \neq T_2.Gid \,\wedge\, T_2.Gid \neq T_3.Gid}\left(T_1 \times T_2 \times T_3\right)\right)$$

Important Point

Course (Sid, Cid, Contact)

Q) find the student who Participate At least Three Game ?

GAME (RollNo, Gid)

Gid → Game Id

$\rho(T_1, Game)$ $\rho(T_2, Game)$ $\rho(T_3, GAME)$

Gid		
T_1	T_2	T_3
1	2	3

$$\Pi_{T_1.RollNo}\left(\sigma_{T_1.RNo = T_2.RNo = T_3.RNo \;\wedge\; T_1.Gid \neq T_2.Gid \;\wedge\; T_2.Gid \neq T_3.Gid \;\wedge\; T_3.Gid \neq T_1.Gid}\left(T_1 \times T_2 \times T_3\right)\right)$$

Important Point

FROM STUDENT

STUDENT

DB
Hard disk

MAIN Memory

'Rename' operator. ← Copies/Aliasing

Important Point

Q.1 select Sname
FROM Supplier.

O/P

Sname

Q.2 Select *
FROM Supplier

Sid	Sname	Rating

Supplier (Sid, Sname, Rating)
Parts (Pid, Pname, Color)
Catalog (Sid Pid Cost)

RR (Raghu Rama Krishna)
Sailors (Sid, S———)
Reserve (Sid bid
Boat (bid, bname

Important Point

(Project)

* → gives all attribute of that Particular Schema in the output

eg

```
select *
FROM Parts
WHERE Color = 'Red'
```

Assume (Total 100 Tuple (2 Tuple Color = Red))

Pid	Pname	Color
P_1	A	Red
P_3	B	Red

Ans

Assume → Parts Table 100 Tuples & 2 Tuples having color Red.

Supplier (Sid, Sname, Rating)

Parts (Pid, Pname, Color)

Catalog (Sid Pid Cost)

RR (Raghu Rama Krishna)

Sailors (Sid, S———)

Reserve (Sid bid

Boat (bid, bname

Important Point

SQL

Rename operator / Alias

AS, ↑ SPACE

eg

Supplier AS S,

Supplier (Sid, Sname, Rating)

Parts (Pid, Pname, Color)

Catalog (Sid Pid Cost)

RR (Raghu Rama Krishna)

Sailors (Sid, S——)

Reserve (Sid bid

Boat (bid, bname

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

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.

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

Aggregate function

∵ Aggregate function are the function that takes Collection of Value (set or Multiset) as a Input & Produce Single output.

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

Important Point ABOUT Aggregate function :

Note Arithmatic operation with NULL Gives Result NULL.

NULL ⊕ 100 = NULL

NULL − 100 = NULL

NULL * 100 = NULL

NULL ÷ 100 = NULL

$\boxed{\text{NULL} \circledast \text{Any Value} = \text{NULL}}$

O/P.

Arithmatic operation

Important Point ABOUT Aggregate function :

Comparsion operation with NULL Gives Result 'Unknown'.

Comparsion operators: =, <, >, >=, <=, <>

Not equal.

0 < NULL

Comparsion Unknown

≠ X

!= Prog. lang.

5 = 'NULL'

Comparision

: Unknown

Important Point ABOUT Aggregate function :

- Aggregate operator Always Discard/Ignored the NULL value.
- While Performing Aggregate operation then first Discard/Ignored/Remove the NULL then Performed Computation/calculation.

Important Point ABOUT Aggregate function :

NULL :
- Not existed
- Unknown
- Two NULL values are Not equal.
- Non Zero.
- Random ASCII values Assigned by DBMS S/w.

Aditya SHARMA

Name

FN MN LN

MN ⇓ Not Applicable

Pass Port

Applicable but some Student Not having Passport [unknown]

Important Point ABOUT Aggregate function :

COUNT([DISTINCT] marks) = '3' Previous Table.

COUNT (*) = 5 (# of Tuples)

COUNT ([DISTINCT] *) →

COUNT (* [DISTINCT]) →

No

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

Aggregation operator ⇒ Always Discard Null Value

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

[ ]

optional

1) Count(marks) = 4
2) Count (*) = '5' → # of 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) = $\frac{216}{3}$

④ SUM = 90 + 70 + 70 + 56 = 286.

⑤ SUM [DISTINCT] = 90 + 70 + 56 = 216.

⑥ $\text{AVG (Marks)} = \dfrac{\text{SUM (Marks)}}{\text{COUNT (Marks)}} = \dfrac{286}{4} = 71.25$

⑦ $\text{AVG}\left(\text{[DISTINCT] Marks}\right) = \dfrac{\text{SUM [DISTINCT] Marks}}{\text{COUNT}\left(\text{[DISTINCT] Marks}\right)} = \dfrac{216}{3}$

Important Point ABOUT Aggregate function :

marks.

90
70
NULL
70
56

SUM

As we know ✓ $\boxed{\text{NULL} + 100 = \text{NULL}.}$

So SUM of Marks is NULL ?

Soln

No., if we say Answer is NULL that means all Student Marks is NULL.

Important Point ABOUT Aggregate function :

Company

Salary
NULL
NULL
70
80
90
60

Board of Director of Reliance Group.

1lakh Employee

Sal'n

As we know ✓ $\boxed{\text{NULL} + 100 = \text{NULL}.}$

So SUM of Salary is NULL ?

NO., if we say Answer is NULL that means all All employees Salary is NULL.

NO

WHY?

P.T.O

Important Point ABOUT Aggregate function :

marks.

90
70
NULL
70
56

SUM

$\boxed{\text{NULL} + 100 = \text{NULL}.}$

So SUM of Marks is NULL ?

Soln

No.

Bcz Aggegate function firstly Discard/Remove/Ignored NULL then Apply operation.

Important Point ABOUT Aggregate function :

- Aggregate operator Always Discard / Ignored the NULL value.
- While Performing Aggregate operation then first Discard / Ignored / Remove the NULL then Performed Computation / calculation.

Important Point ABOUT Aggregate function :

marks.

90
70
NULL
70
56

NULL + 100 = NULL.

then WHER is the USE of NULL + arithmatic ?

P.T.O

2 Marks GATE

R(STUDENT

Roll	Marks
1	15
2	20
3	25
4	NULL

UPDATE R

Marks = Marks + 5

then

Calculate Avg(Marks)?

Marks = Marks + 5

15 + 5 = 20

20 + 5 = 25

25 + 5 = 30

NULL + 5 = NULL

Roll	Marks
1	20
2	25
3	30
4	NULL

then Avg.

$$Avg = \frac{20+25+30}{3}$$

$$= 75/3 \quad \underline{Avg}$$

R

A
7
5
-2
NULL
0
NULL
7
NULL

$0+7+7+5-2$

$\frac{17}{5}$

SELECT COUNT(A), Count(*), SUM(A)

AVG(A) MIN(A) MAX(A)

FROM R

Firstly Discard the NULL then Perform operation.

① COUNT(A) = 5

COUNT(*) = 8 → # Rows (Tuples)

SUM(A) = 17

AVG = $\frac{17}{5}$

MIN(A) = -2

MAX(A) = 7

R

A
NULL
NULL
NULL
NULL
NULL
NULL

Count(*) : 6

COUNT(A) : 0

SUM(A) : NULL

AVG(A) : NULL

MIN(A) : NULL

MAX(A) : NULL.

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

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

HAVING

Use Having without Group By clause.

Latest DBMS S/W.

GATE

SQL Standard

1992

HAVING

Always Preceed by Group By clause.

According to Standards (SQL-1992)

HAVING :
- → USE After GROUP By clause.
- → Having Condition Applied on each Group.

---

latest (Not Standard): If Having is Used Without Group By clause then Having Condition is Applied on each Tuple/Record.

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

$\frac{150}{2} = 75$

$\frac{130}{2} = 65$

$\frac{55}{1} = 55$

4 1

Select * FROM STUDENT

2 GROUP By (Branch)

3 HAVING AVG(Marks) > 61

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

Ans

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

Soln 1 55 Ans

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

Sol^n 2

Ans (60)

ANSWER: 60

Q.3 Select min(marks)

FROM Student

GROUP By (Branch)

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

Sol^n 3

60
60
55

Ans

Q.3

Select min(marks)

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

Syntax Error.

55 EC

min(marks)	Branch
55	CS
	IT
	CS
	IT
	EC
	EC

GATEMaster

55	CS
55	IT
55	CS
55	IT
55	EC
55	EC

Q.4 Select min(marks), Branch

FROM Student

55 → CS, IT, CS, IT, EC, EC

One to Many

➡ Such Syntax is not allowed in SQL

Note: If Aggregate operator & other Attribute [Normal (or) Unaggregate] Used in Select clause is allowed Only iff Other Attribute [Non-Aggregate 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)
```

Branch is other Attribute (Non Aggegrate Attribute)

O/P

min(marks)	Branch
60	CS
60	IT
55	EC

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

↓ Distinct

⇓ Multiset (Bag)

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

(I) R UNION S
R UNION ALL S

(II) R INTERSECT S
R INTERSECT ALL S

(III) R MINUS S (EXCEPT)
R MINUS ALL S (EXCEPT ALL)

SQL SET OPERATORS

1. UNION & UNION ALL
2. INTERSECT & INTERSECT ALL
3. MINUS [EXCEPT] & MINUS ALL (EXCEPT ALL)

Multi Set (Bag)
⇓
Duplicates allowed.

'Distinct'

⇓
Followed by Relation Algebra

⇓
Not followed by Relational Algebra.

1) R UNION S

Result Distinct tuple

'set'

'Distinct'

1
2
4
6
5

2) R UNION ALL S

Result all values

Multiset (Bag)

'Duplicates'.

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

R

S

3) R INTERSECT S

Distinct Common tuples from R & S

1
2
4

Ans

4) R INTERSECT ALL S

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

1
1
2
4

Ans

EXCEPT

4) R MINUS S

Distinct tuples from R those are not there in S

6

Ans

(EXCEPT ALL)

5) R MINUS ALL S

Duplicates in R − # Duplicates in S

1
2
6
6

Ans

Q. Retrieve Sid & Marks of the Student who secured Highest Marks?

Note: Aggregate friction can not be in lower clause

Query I: Select Sid max(Marks)
From Student

Query II: Select Sid, max(marks)
From Student
Group By (Sid).

Query III: Select Sid marks
From Student
Where marks = $\left(\begin{array}{l}\text{Select max(Marks)}\\ \text{From Student}\end{array}\right)$

Sid	Marks
S_3	90

OTHER SET OPERATOR

1. IN/NOT IN
2. ANY
3. ALL
4. EXISTS/NOT EXISTS

COMPARISION OPERATOR

<, >, < =, < >

Not equal

ANY: Compare a value with each value in a Set & Return true if any value is compared according to given condition.

Example:

$x > \text{ANY}\ (10, 20, 30)$

$(x > 10)$ or $(x > 20)$ or $(x > 30)$ $\xrightarrow{\text{output}}$ 11, 12, 13, ...

Example:

$x < \text{ANY}\ (10, 20, 30)$

$(x < 10)$ or $(x < 20)$ or $(x < 30)$ $\xrightarrow{\text{output}}$ 29, 28, 27, ...

ALL: Compare a value with each value in a Set & Return true if given condition satisfied for every/each value in the Set.

Example:

$x > \text{ALL } (10, 20, 30)$

$(x > 10) \text{ AND } (x > 20) \text{ AND } (x > 30) \xrightarrow{\text{output}} 31, 32, 33, 34 \ldots\ldots$

Example:

$x < \text{ALL } (10, 20, 30)$

$(x < 10) \text{ AND } (x < 20) \text{ AND } (x < 30) \xrightarrow{\text{output}} 9, 8, 7, 6 \ldots\ldots$

THANK YOU!