COMPUTER SCIENCE

Database Management System

Query Language - 2

Lecture_06

Vijay Agarwal sir

R.A

1. Selection [σ] → Tuple' (Condition)
2. Projection (π) → Attribute list
3. Union [$\cup$]
4. Intersection [$\cap$]
5. Set Difference [$-$]

Arity

Range

CROSS Product (x)

Equi Join

Conditional Join.

CROSS Product

$R \times S$

R : n_1 Tuple, C_1 Attribute

S : n_2 Tuple, C_2 Attribute

$R \times S$ ⇒ $n_1 \times n_2$ Tuple, $C_1 + C_2$ Attribute

Conditional Join

Equi Join

Natural Join

Step ① Cross Product of R & S.

Step ② : Select the tuple which satisfy equality condition on all Common Attribute. (FROM R x S)

Step ③ Projection of Distinct Attribute.

Natural Join ⋈

R

A	B	C
1	2	4
3	2	6

S

B	C	D
2	4	8
2	7	4

$$\Pi_{ABCD}\left(\sigma_{\substack{R.B=S.B \\ R.C=S.C}} \wedge (R \times S)\right)$$

$R.B = S.B \wedge R.C = S.C$

$R \times S =$

Step1 ⇒

R.A	R.B	R.C	S.B	S.C	S.D
1	2	4	2	4	8
1	2	4	2	7	4
3	2	6	2	4	8
3	2	6	2	7	4

Step2

R.A	R.B	R.C	S.B	S.C	S.D
1	2	4	2	4	8

$$R \bowtie S = \pi_{ABCD} \left\{ \sigma_{\substack{R.B = S.B \\ R.C = S.C}} \wedge (R \times S) \right\}$$

$R \bowtie S =$

A	B	C	D
1	2	4	8

$R \bowtie S =$

A	B	C	D
1	2	4	8

Left outer Join ($R \bowtie S$) ⇒ $R \bowtie S$ && Tuple from Left Side Relation (R) which failed to satisfy Join Condition.

Right outer Join ($R \bowtie S$): $R \bowtie S$ && Tuples from Right Side Relation [S] which failed to Satisfy Join Condition.

Full outer Join ($R \bowtie S$) Left Outer Join ∪ Right outer Join.

Left Outer Join [⟕]

$(R \overset{L}{\bowtie} S)$

R

A	B	C
1	2	4
3	2	6

S

B	C	D
2	4	8
2	7	4

Not satisfy Join

→ failed

$(R \overset{L}{\bowtie} S) =$

A	B	C	D
1	2	4	8

$R \overset{L}{\bowtie} S =$

A	B	C	D
1	2	4	8
3	2	6	Null

$R \overset{F}{\bowtie} S =$

A	B	C	D
1	2	4	8
Null	2	7	4

FULL outer Join

A	B	C	D
1	2	4	8
3	2	6	Null
Null	2	7	4

Right Outer Join [⟖]

$R \rightouterjoin S =$

A	B	C	D
1	2	4	8
Null	2	7	4

Full Outer Join [⟗]

Full outer join = Left outer join Union Right outer join

$R ⟗ S = R ⟕ S \cup R ⟖ S$

A	B	C	D
1	2	4	8
3	2	6	Null

$\cup$

A	B	C	D
1	2	4	8
Null	2	7	4

$R ⟗ S =$

A	B	C	D
1	2	4	8
3	2	6	Null
Null	2	7	4

GATE ?

$(1.00)^{365} = 1$

$(1.01)^{365} = 37.33$

- Engg.
- Job
  - MNC
  - Govt
    - Central
    - PSU's
- Research/PhD
- Mtech

Q.

Let R and S be two relations with the following schema

R(P, Q, R1, R2, R3)

S(P, Q, S1, S2)

[GATE & NIC 2020]

R.P = S.P ∧ R.Q = S.Q

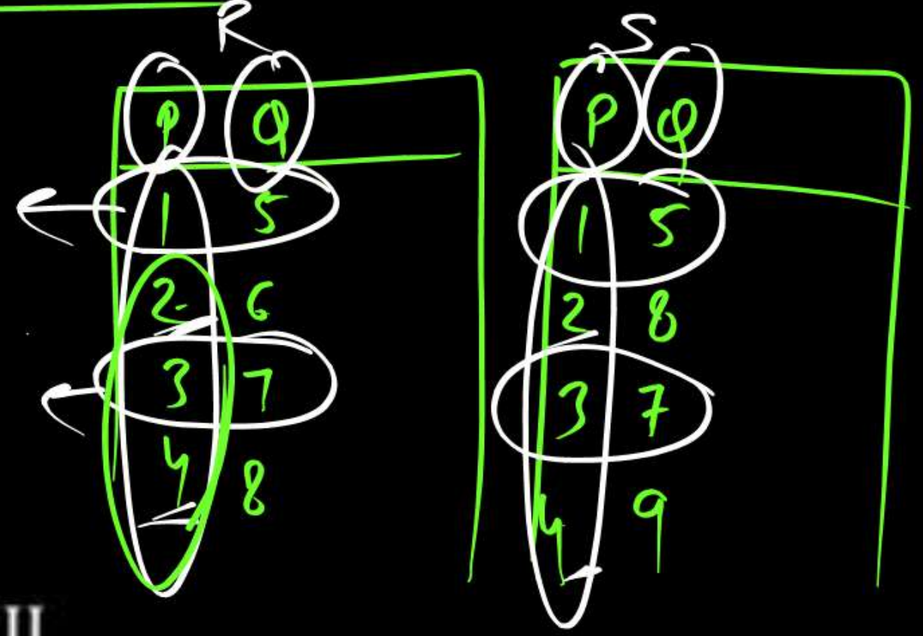
Where {P, Q} is the key for both schemas. Which of the following queries are equivalent?

I. $\pi_P (R \bowtie S)$

II. $\pi_P(R) \bowtie \pi_P(S)$

III. $\pi_P(\pi_{P,Q} (R) \cap \pi_{P,Q}(S))$

IV. $\pi_P(\pi_{P,Q}(R) - (\pi_{P,Q} (R) - \pi_{P,Q} (S)))$



(A) Only I and II

(B) Only I and III

(C) Only I, II and III

(D) Only I, III and IV

$$R \cap S = R - (R - S)$$

Division

Q. Retrieve all student who are Enrolled **Some course** or **Any course** or at least one course?

Solution Π_{Sid} (Enrolled)

Sid → Any/Course

some / Atleast

Enrolled	
Sid	**Cid**
S_1	C_1
S_1	C_2
S_1	C_3
S_2	C_1
S_2	C_3
S_3	C_1

Course
Cid
C_1
C_1
C_3

$R \bowtie S$

Derived operators

1. Intersection
2. JOIN ($\bowtie$)
3. Division

Division

Q. Retrieve all student who are Enrolled every course?

Solution

$\Pi_{Sid,Cid}$ (Enrolled) / Π_{Cid} (Course)

Find

2^{nd} attribute must be same.

Enrolled	
Sid	Cid
S_1	C_1
S_1	C_2
S_1	C_3
S_2	C_1
S_2	C_3
S_3	C_1

Course
Cid
C_1
C_2
C_3

$\Pi_{Sid}(Enrolled) - \Pi_{Sid}\left[\Pi_{Sid}(Enrolled) \times \Pi_{Cid}(Course) - Enrolled\right]$

S_2
S_3

S_1
S_2
S_3 — S_2, S_3

o/p

Sid
S_1

Enrolled every Course.

Sid
S_1
S_2
S_3

×

Cid
C_1
C_2
C_3

⇒

Sid	Cid
S_1	C_1
S_1	C_2
S_1	C_3
S_2	C_1
S_2	C_2
S_2	C_3
S_3	C_1
S_3	C_2
S_3	C_3

$3 \times 3 = 9$ Tuple

$1+1 = 2$ Attribute

All Student Enrolled every Course

—

Sid	Cid
S_1	C_1
S_1	C_2
S_1	C_3
S_2	C_1
S_2	C_3
S_3	C_1

≠

Sid	Cid
S_2	C_2
S_3	C_2
S_3	C_3

These Student Not enrolled these Course.

Π_{Sid}

Sid
S_2
S_3

Not enrolled all Course

Division

$\Pi_{Sid,Cid}(Enrolled) / \Pi_{Cid}(Course)$

$\Rightarrow \Pi_{Sid}(Enrolled) - \Pi_{Sid}[\Pi_{Sid}(Enrolled) \times \Pi_{Cid}(Course) - Enrolled]$

$\Pi_{AB}(R) / \Pi_{B}(S) \Rightarrow \Pi_{A}(R) - \Pi_{A}\left(\Pi_{A}(R) \times \Pi_{B}(S) - R\right)$

$$\pi_{AB}(R) \div \pi_{B}(S) \Rightarrow \pi_{A}(R) - \pi_{A}\left[\pi_{A}(R) \times \pi_{B}(S) - R\right]$$

Course Deposit (Submit)

$$\pi_{AB\,CD}(R) \div \pi_{CD}(S) \Rightarrow \pi_{AB}(R) - \pi_{AB}\left[\pi_{AB}(R) \times \pi_{CD}(S) - R\right]$$

Name

F. Contact

Division

$$\Pi_{AB}(R) \,/\, \Pi_{B}(S) = \Pi_{A}(R) - \Pi_{A}\left[\Pi_{A}(R) \times \Pi_{B}(S) - R\right]$$

Find Connection

$$\Pi_{ABCD}(R) \,/\, \Pi_{CD}(S) \Rightarrow \Pi_{AB}(R) - \Pi_{AB}\left[\Pi_{AB}(R) \times \Pi_{CD}(S) - R\right]$$

Q.

Consider the following three relations in a relational database:

Employee (eId, Name), Brand (bId, bName), Own(eId, bId)

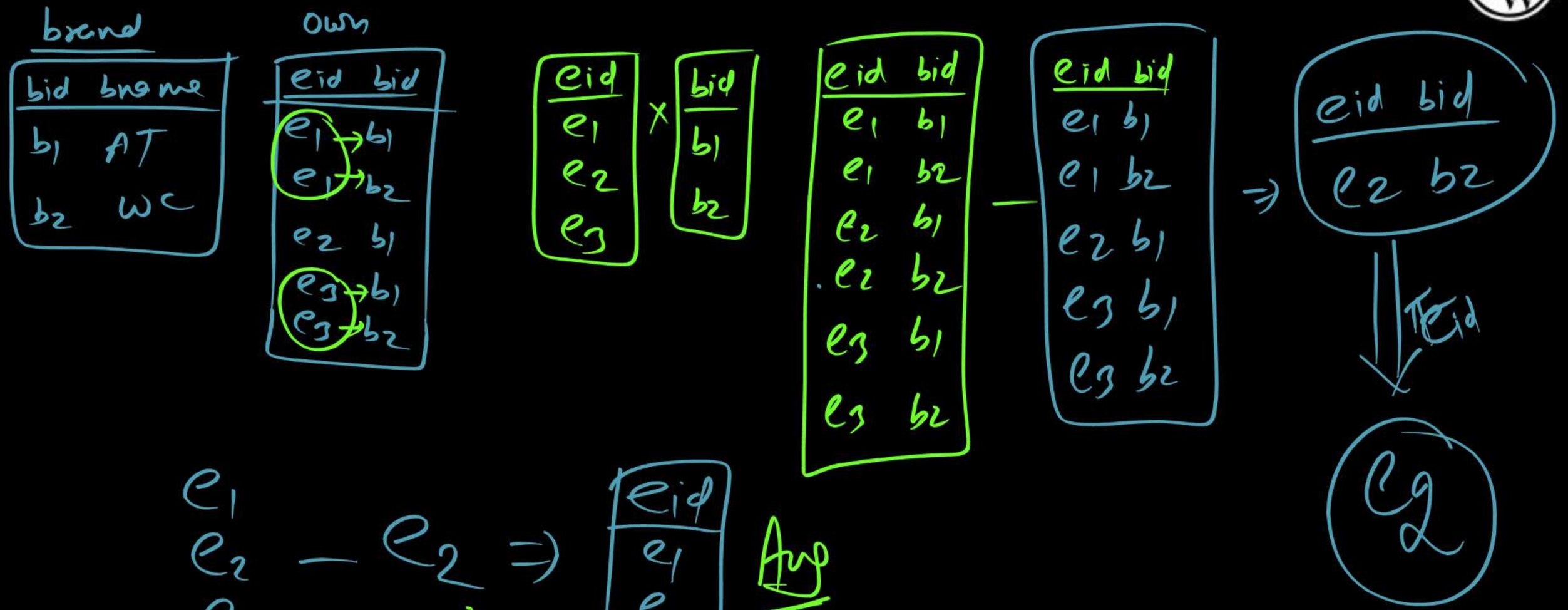
Which of the following relational algebra expressions return the set of eIds who own all the brands? MSQ [GATE: 2022]

A $\pi_{eId}\,(\pi_{eId,\,bId}\,(Own) / \pi_{bId}\,(Brand))$

B $\pi_{eId}\,(Own) - \pi_{eId}\,((\pi_{eId}\,(Own) \times \pi_{bId}\,(Brand)) - \pi_{eId,\,bId}\,(Own))$

C $\pi_{eId}\,(\pi_{eId,\,bId}\,(Own) / \pi_{bId}\,(Own))$

D $\pi_{eId}\,((\pi_{eId}\,(Own) \times \pi_{bId}\,(Own) / \pi_{bId}\,(Brand))$


brand
bid bname
b1 AT
b2 WC
own
eid bid
e1 b1
e1 b2
e2 b1
e3 b1
e3 b2
eid
e1
e2
e3
X
bid
b1
b2
eid bid
e1 b1
e1 b2
e2 b1
e2 b2
e3 b1
e3 b2
eid bid
e1 b1
e1 b2
e2 b1
e3 b1
e3 b2
⇒
eid bid
e2 b2
Teid
e2
e1
e2 − e2 ⇒
eid
e1
e3
Ans

Consider the Database with relations:

S Supplier (Sid, Sname, Rating)

P Parts (Pid, Pname, Color)

S Catalog (Sid Pid, Cost)

Rename [S]

① $S(T_1, Parts)$

T_1		
Pid	Pname	Color

Q. Find the Sid of Supplier whose Rating greater than 9?

Find → 3 Table

Name of Red

P.Color = Red ∧

P.Pid = C.Pid ∧

C.Sid = S.Sid

$\Pi_{Sid}(\sigma_{Rating>9}(Supplier))$

Q.

Find the Pid of Red Color Parts?

$$\Pi_{Pid}\left(\sigma_{Color=Red}\left(Parts\right)\right)$$

Q.

Retrieve Sid of Supplier whose cost is greater than 20,000?

$$\Pi_{Sid}\left(\sigma_{Cost>20,000}(Catalog)\right)$$

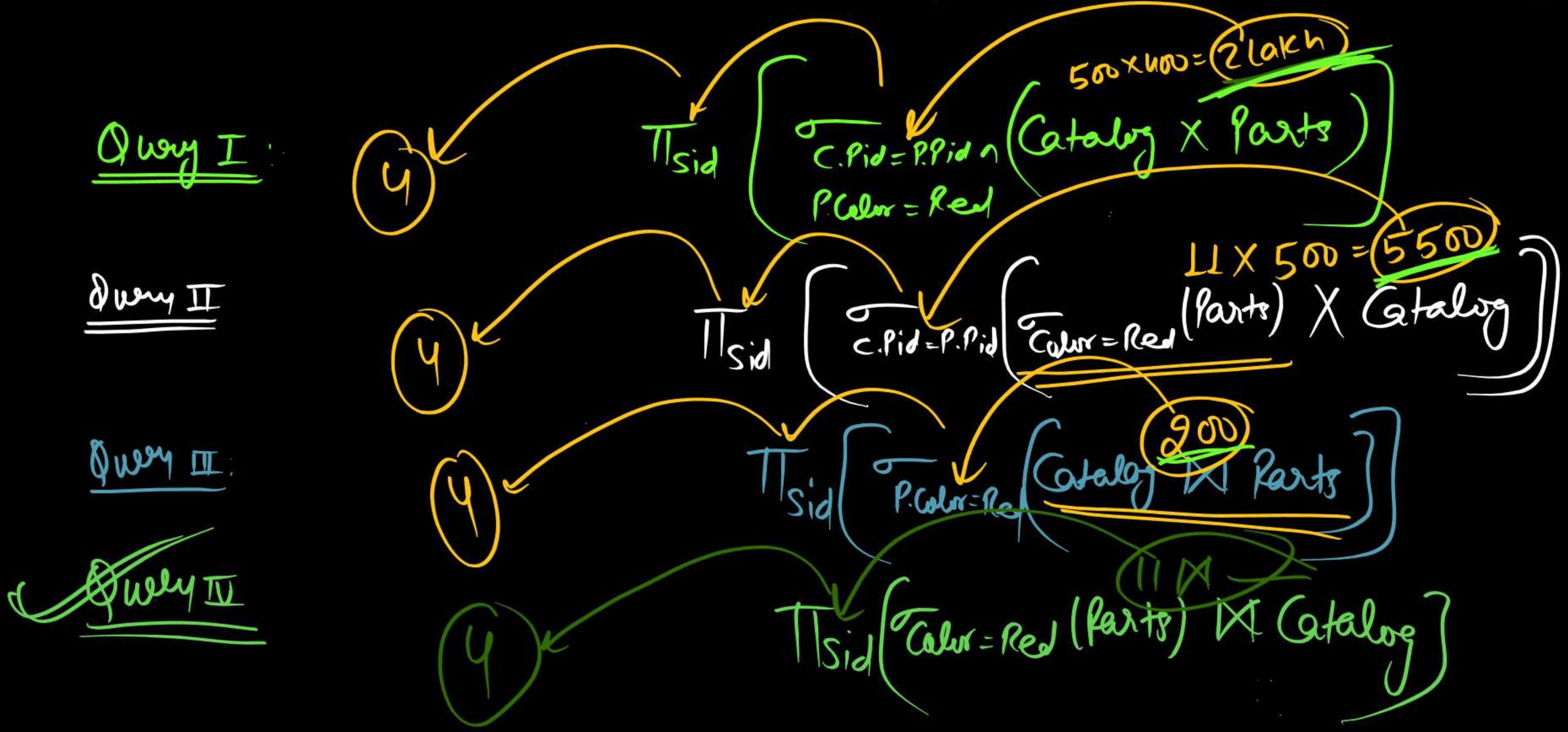
Q. Retrieve Sid of Supplier who supplied some Red color parts?

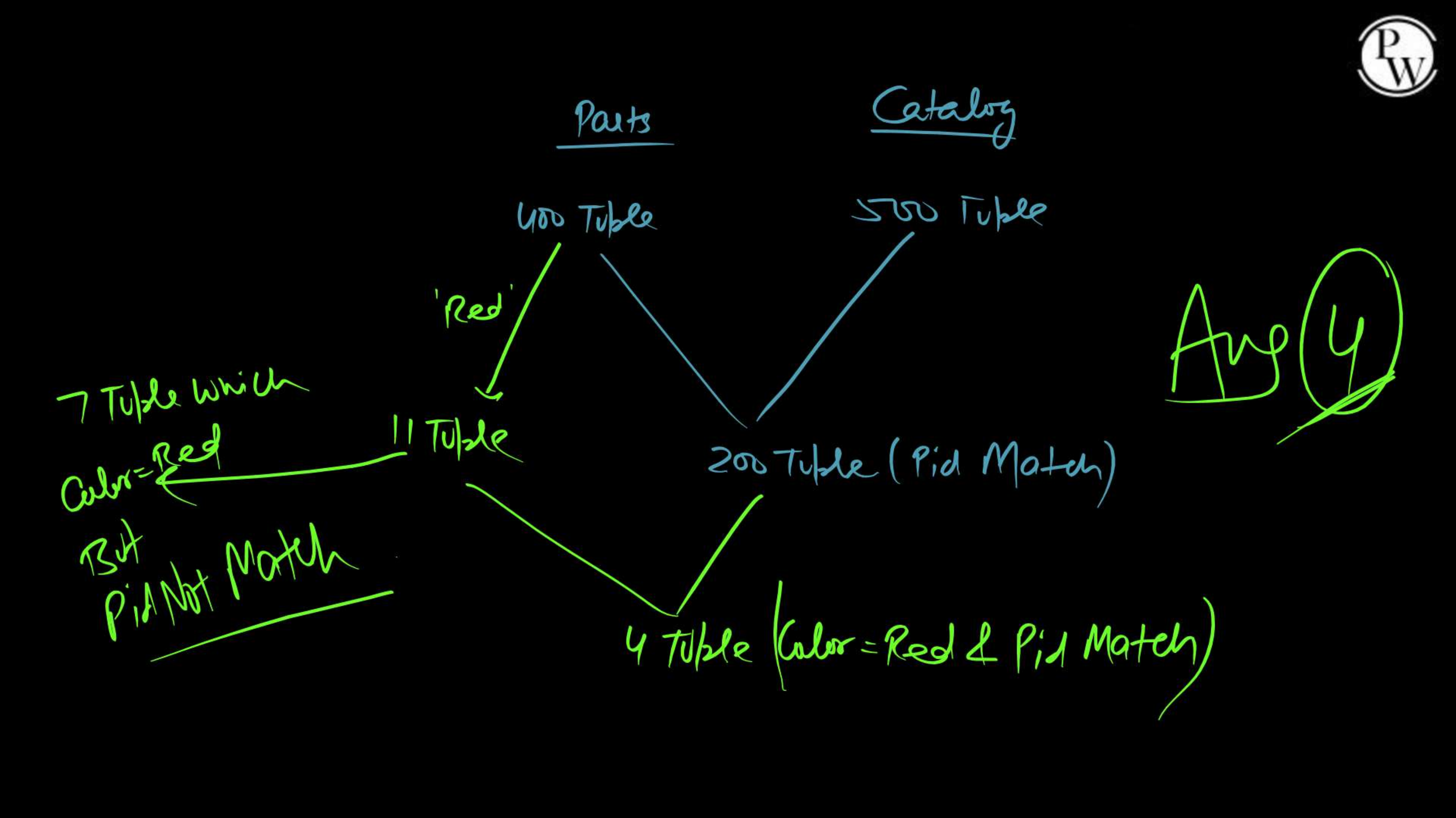
Solution:

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

AND

Parts (Pid, Pname, Color) Catalog (Sid, Pid, Cost)




Parts
Catalog
400 Tuple
≤700 Tuple
'Red'
7 Tuple which Color=Red But Pid Not Match
11 Tuple
200 Tuple (Pid Match)
4 Tuple (Color=Red & Pid Match)
Ans 4

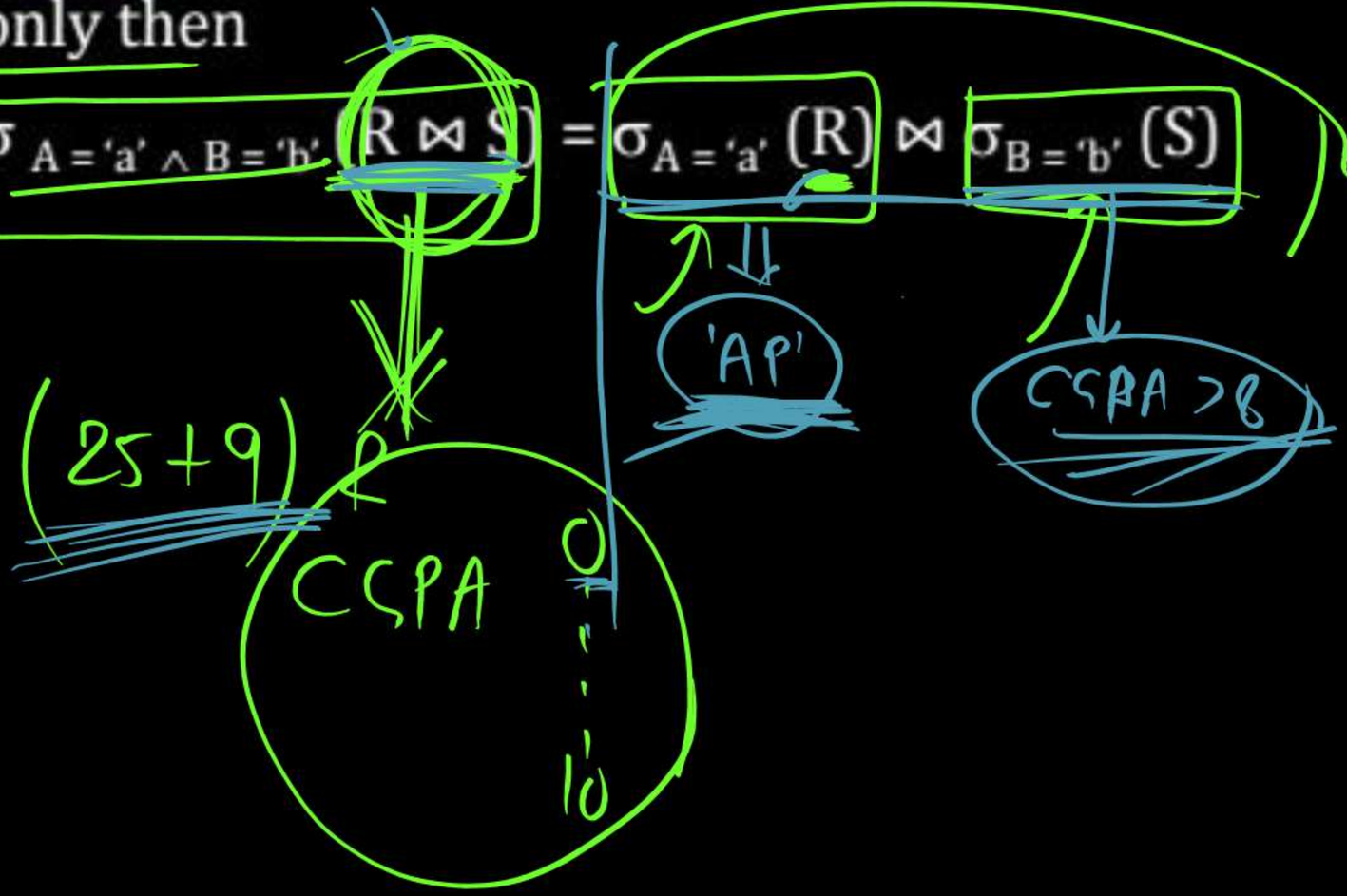
Note: Let an Attribute A belongs to R only then

$\sigma_{A='a'}(R \bowtie S) = \sigma_{A='a'}(R) \bowtie S$ → More efficiency query

AICTE
25 + 9
State U.T

Note: Let an Attribute A belongs to R only and Attribute B belongs to S only then

$\sigma_{A='a' \wedge B='b'}(R \bowtie S) = \sigma_{A='a'}(R) \bowtie \sigma_{B='b'}(S)$



Q.

Consider the following relation schemas:

b-Schema = (b-name, b-city, assets)

a-Schema = (a-num, b-name, bal)

d-Schema = (c-name, a-number)

Let branch, account and depositor be respectively instances of the above schemas. Assume that account and depositor relations are much bigger than the branch relation.

Consider the following query:

$$\Pi_{\text{c-name}}(\sigma_{\text{b-city} = \text{"agra"} \wedge \text{bal} < 0}(\text{branch} \bowtie \text{account} \bowtie \text{depositor}))$$

Q. Which one of the following queries is the most efficient version of the above query? [GATE-2007: 2 Marks]

A $\Pi_{\text{c-name}}(\sigma_{\text{bal}<0}(\sigma_{\text{b-city}=\text{"Agra"}}\ \text{branch} \bowtie \text{account}) \bowtie \text{depositor})$

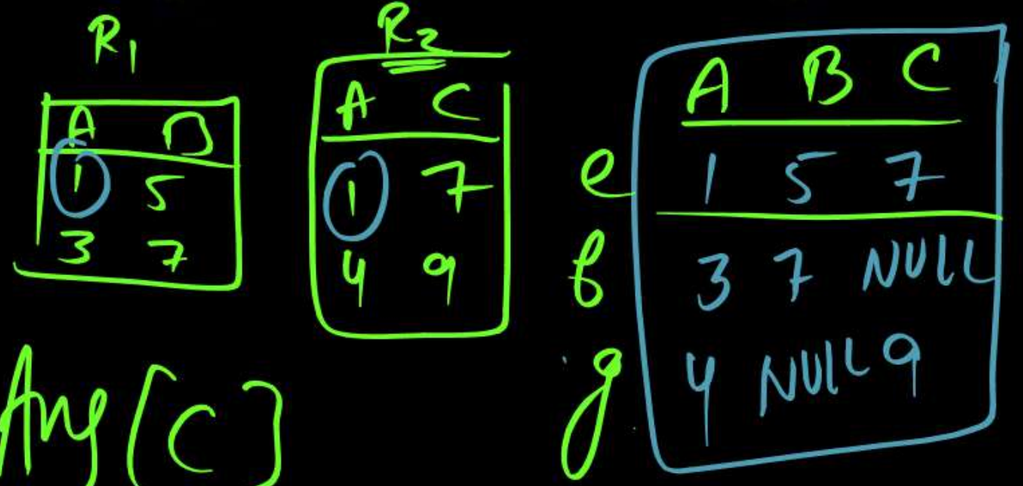
B $\Pi_{\text{c-name}}((\sigma_{\text{b-city}=\text{"Agra"}}\ \text{branch}) \bowtie (\sigma_{\text{bal}<0}\ \text{account}) \bowtie \text{depositor})$

C $\Pi_{\text{c-name}}(\sigma_{\text{b-city}=\text{"Agra"}}\ \text{branch} \bowtie \sigma_{\text{b-city}=\text{"Agra}\wedge\text{bal}<0}\ \text{account} \bowtie \text{depositor})$

D $\Pi_{\text{c-name}}(\sigma_{\text{b-city}=\text{"Agra"}}\ \text{branch} \bowtie (\sigma_{\text{b-city}=\text{"Agra}\wedge\text{bal}<0}\ \text{account} \bowtie \text{depositor}))$

Q. Consider two relations $R_1(A, B)$ with the tuples (1, 5), (3, 7)and R_2 (A, C) = (1, 7)(4, 9)

Assume that R(A, B, C) is the full natural outer join of R_1 and R_2. Consider the following tuples of the form (A, B, C); a = (1, 5, null), b = (1, null, 7), c = (3, null, 9), d = (4, 7, null), e = (1, 5, 7), f = (3, 7, null), g = (4, null, 9). Which one of the following statements is correct? [GATE-2015: 1 Mark]



A. R contains a, b, e, f, g, but not c, d

B. R contains all of a, b, c, d, e, f, g

C. R contains e, f, g, but not a, b

D. R contains e but not f, g

Ans (C)

Q.

Consider the following relations given below:

R

A	B
6	6
7	6
8	8

S

C	D
6	7
8	9
8	10

R.A	R.B	S.C	S.D
6	6	6	7
6	6	8	9
6	6	8	10
7	6	6	7
7	6	8	9
7	6	8	10
8	8	6	7
8	8	8	9
8	8	8	10

A	D
6	7
6	9
6	10
7	7
7	9
7	10
8	7
8	9
8	10

A	D
6	7
8	9
8	10

A	D
6	9
6	10
7	7
7	9
7	10
8	7

$$\Pi_{AD}(R \times S) - \rho_{A \leftarrow B}(\Pi_{BD}(R \bowtie_{B=C} S))$$

Number of tuples return by the above query when it is executed on the above instance of relation R and S is 6

Summary

OPERATION	PURPOSE	NOTATION
SELECT	Selects all tuples that satisfy the selection condition from a relation R.	$\sigma_{\text{<selection condition>}}(R)$
PROJECT	Produces a new relation with only some of the attributes of R, and removes duplicate tuples.	$\pi_{\text{<attribute list>}}(R)$
THETA JOIN	Produces all combinations of tuples from R_1 and R_2 that satisfy the join condition.	$R_1 \bowtie_{\text{<join condition>}} R_2$
EQUIJOIN	Produces all the combinations of tuples from R_1 and R_2 that satisfy a join condition with only equality comparisons.	$R_1 \bowtie_{\text{<join condition>}} R_2$, OR $R_1 \bowtie_{\text{(<join condition 1>), (<join condition 1>)}} R_2$
NATURAL JOIN	Same as EQUIJOIN except that the join attributes of R_2 are not included in the resulting relation; if the join attributes have the same names, they do not have to be specified at all.	$R_1 *_{\text{<join condition>}} R_2$, OR $R_1 *_{\text{(<join attributes 1>), (<join attributes 2>)}} R_2$ OR $R_1 * R_2$

OPERATION	PURPOSE	NOTATION
UNION	Produces a relation that includes all the tuples in R_1 or R_2 or both R_1 and R_2; R_1 and R_2 must be union compatible.	$R_1 \cup R_2$
INTERSECTION	Produces a relation that includes all the tuples in both R_1 and R_2; R_1 and R_2 must be union compatible.	$R_1 \cap R_2$
DIFFERENCE	Produces a relation that includes all the tuples in R_1 and that are not in R_2; R_1 and R_2 must be union compatible.	$R_1 - R_2$
CARTESIAN PRODUCT	Produces a relation that has the attributes of R_1 and R_2 and includes as tuples all possible combinations of tuples from R_1 and R_2.	$R_1 \times R_2$
DIVISION	Produces a relation R(X) that includes all tuples t[X] in $R_1(Z)$ that appear in R_1 in combination with every tuple from $R_2(Y)$, where $Z = X \cup Y$	$R_1(Z) \div R_2(Y)$

SQL:

DDL: Schema

DML → Data

DCL → Transaction

DQL →

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

SQL

RA

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]

RA: $\Pi_{A_1 A_2 A_3 \ldots A_n}\left(\sigma_{\text{Condition}}\left(R_1 \times R_2 \times R_3 \ldots \times R_m\right)\right)$

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

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) $\pi_{AB}(R)$

A	B
1	2
2	4

3) SELECT DISTINCT AB
FROM R

Output

A	B
1	2
2	4

Select Name

FROM Student

[WHERE]

Condition

[ ] ← optional

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.

GROUP BY (Gender)

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

Count (Marks) = 5

Count (Marks) = 4

Count ([DISTINCT] Marks) = 3

SUM (Marks) = 286

SUM ([DISTINCT] Marks) = 216

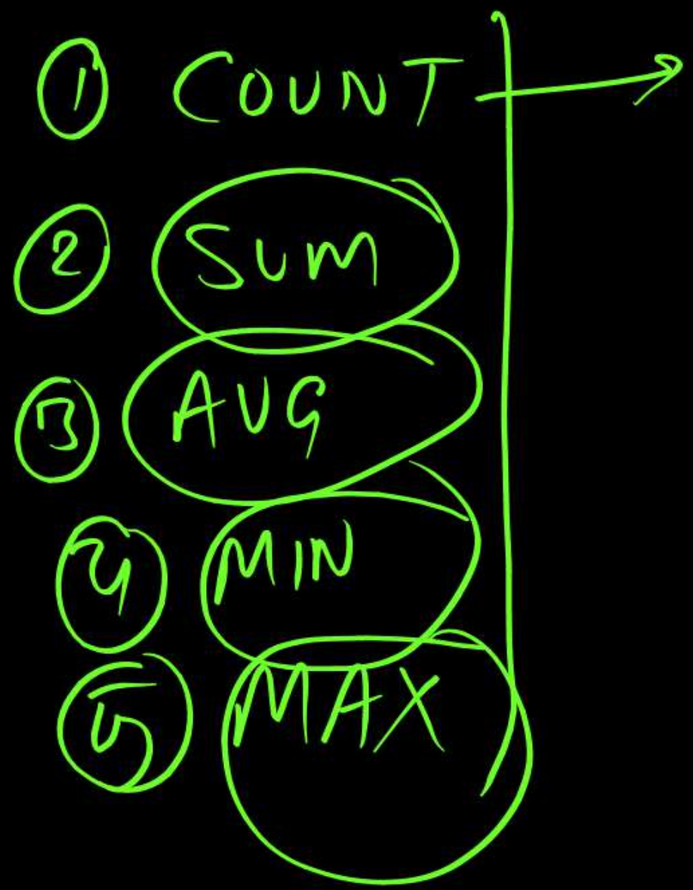
Avg marks = $\frac{286}{4}$

Avg = $\frac{216}{3}$

MIN (marks) = 56

MAX (marks) = 90

Aggregate Operator ⟶ Discard the '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) AVG(marks) = $\frac{286}{4}$
7) AVG([Distinct]marks) = $\frac{216}{3}$

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

HAVING :

Select AB
FROM R
WHERE → Condition

Select *
FROM R
WHERE Condition

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

Condition Applied on each Group

```
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

Q.1 Select min(marks)

FROM Student

Ans 1 55

ANSWER: 55

Q.2

Select min(marks)

FROM Student

WHERE Branch = 'CS'

Ans 60

ANSWER: 60

Q.3

Select min(marks)

FROM Student

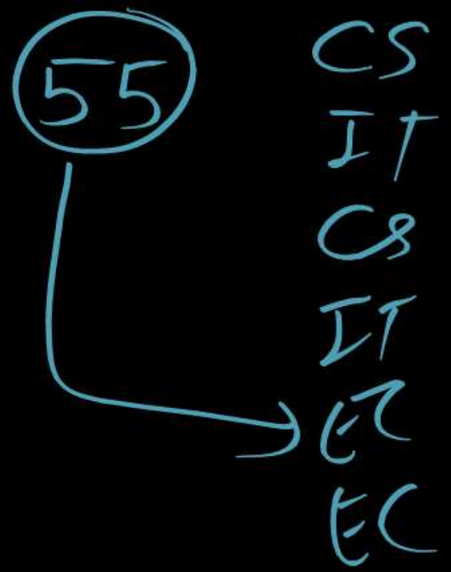
GROUP By (Branch)

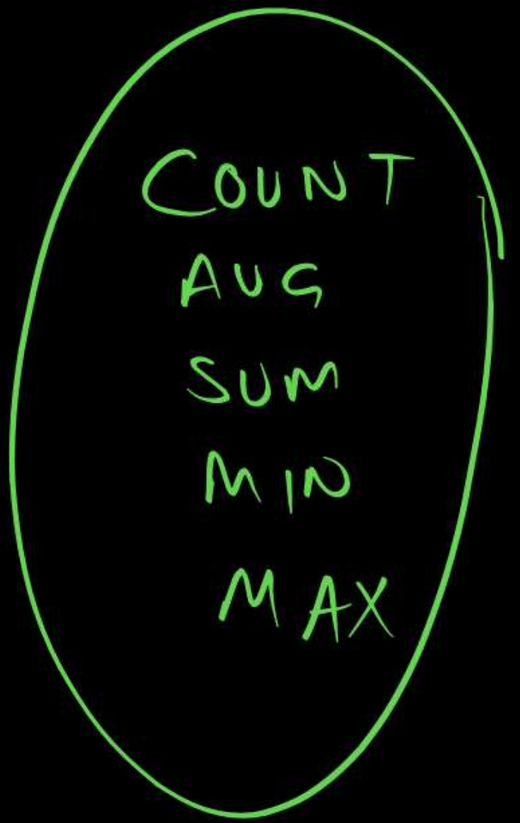
60
60
55

Select min(marks), Branch

FROM Student.

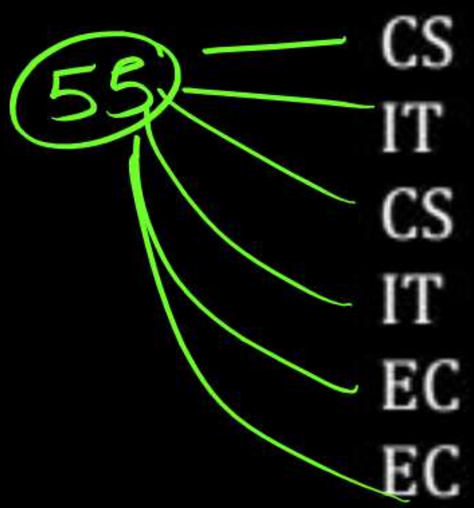
Such syntax Not allowed.





Q.4 Select min(marks), Branch

FROM Student



➡ Such Syntax is not allowed in SQL

Q.4 Select min(marks), Branch

FROM Student

CS
IT
CS
IT
EC
EC

➡ Such Syntax is not allowed in SQL

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)
```

GROUP By (Gender) ✗

min(marks)	Branch
60	CS
60	IT
55	EC

SET operator

1. UNION | UNION ALL
2. Intersect | Intersect ALL
3. MINUS | MINUS ALL

MINUS ⇓ followed by RA

MINUS ALL ⇓ Not followed by RA

(I) UNION

UNION ALL

(II) Intersect

Intersect ALL

(III) MINUS (R − S) EXCEPT

MINUS ALL

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	

2 Marks

EXCEPT

1 1 2 2 4 6 6 1 2 4 6

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

1 2 6 6

1) R UNION S

Result Distinct tuple

1
2
4
6
5

2) R UNION ALL S

Result all values

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

3) R INTERSECT S

Distinct Common tuples from R & S

1
2
4

4) R INTERSECT ALL S

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 - # Duplicates
in R in S

1
2
6
6

Query : One to Many Comparsion Directly Not Possible

IN / NOT IN

ANY

ALL

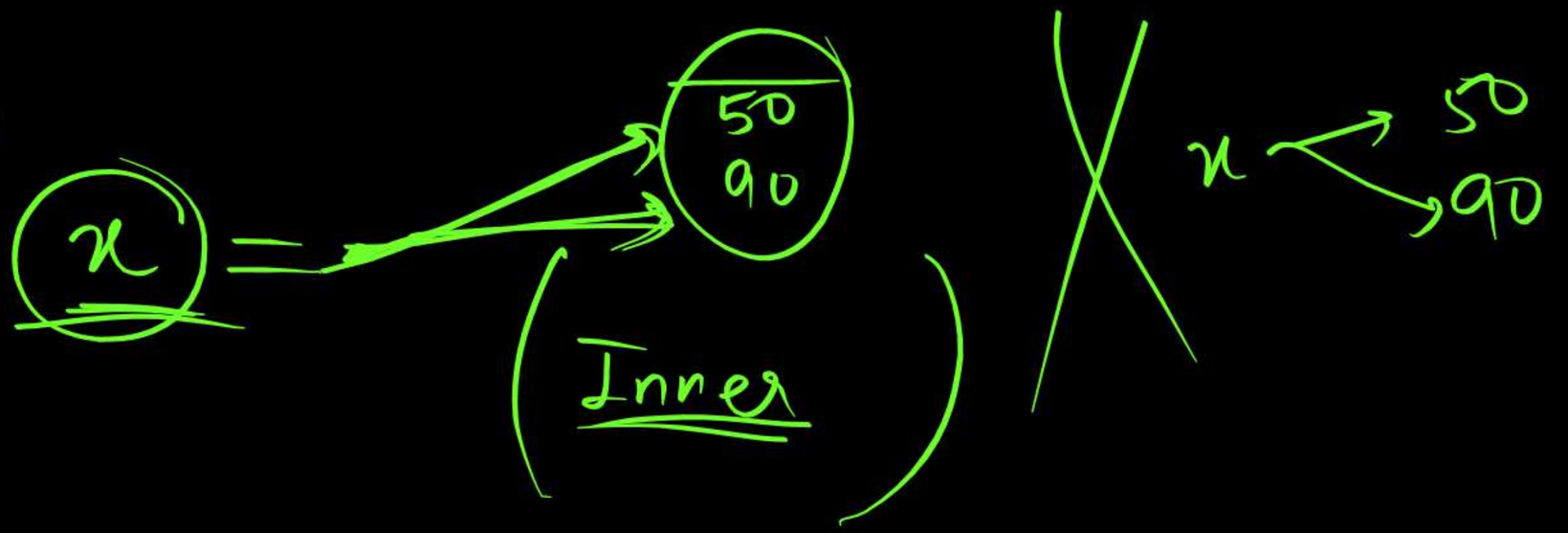
EXISTS

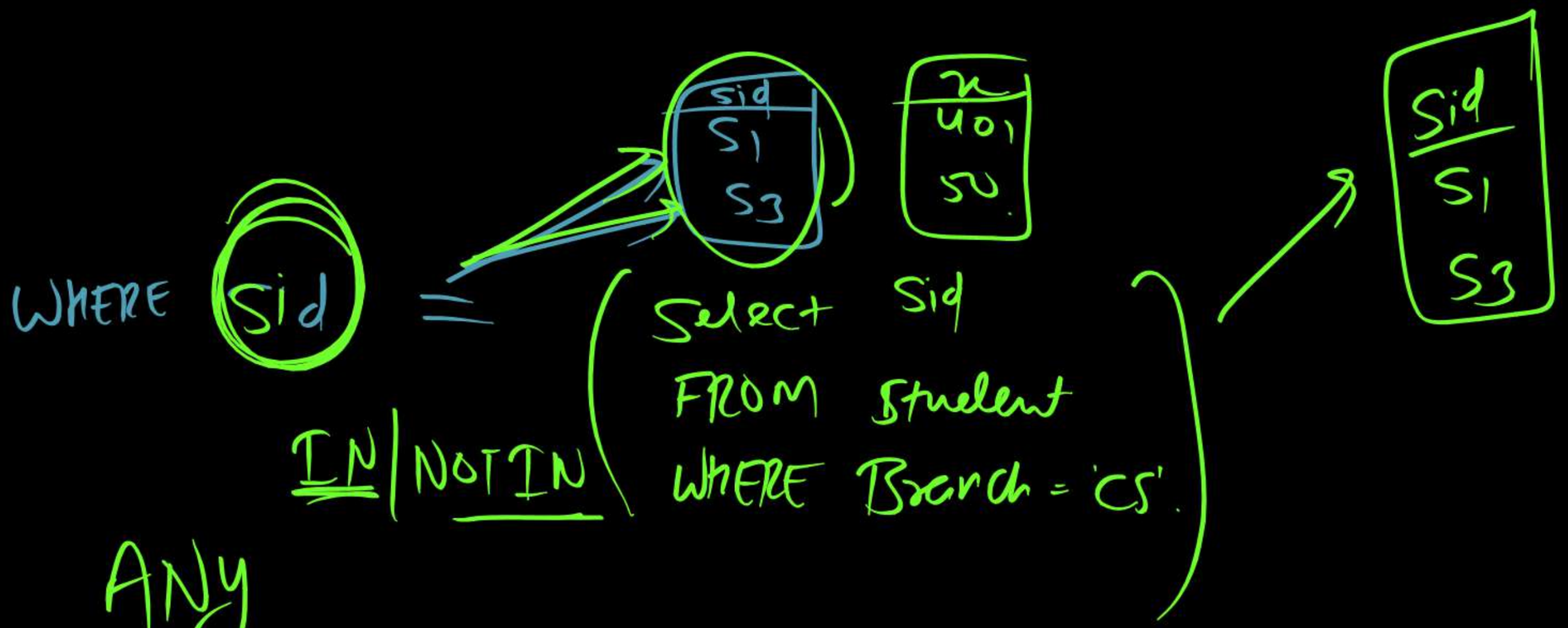
NOT EXSITS.

Select

FROM

WHERE




WHERE
Sid
=
sid
S1
S3
IN | NOT IN
Select Sid
FROM Student
WHERE Branch = 'CS'
Sid
S1
S3

ANY

ALL

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

Query I: Select Sid max(Marks)
From Student

Note: Aggregate ~~friction~~ function can not be in ~~lower~~ WHERE clause

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

Syntax Correct but Wrong o/p

S_1	60
S_2	70
S_3	90
S_4	-
S_5	-
S_9	-

Query III: Select Sid marks
From Student
Where marks =
(Select max(Marks)
From Student)

90

o/p

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.

ANY ⇒ 'OR'

(I) $x > ANY\ (20, 40, 60)$

$(x > 20)$ OR $(x > 40)$ OR $(x > 60)$

ANY ⇒ 'OR'

O/P

(21, 22, 23, 24, 25 ...)

(II) $x < ANY\ (20, 40, 60)$

$(x < 20)$ OR $(x < 40)$ OR $(x < 60)$

O/P

(59, 56, 57, 56 ...)

$x > ALL\ (20, 40, 60)$

O/P

① $(x > 20)$ AND $(x > 40)$ AND $(x > 60) \Rightarrow$ [61, 62, 63, 64, 65, 66, 67 . . .

$x < ALL\ (20, 40, 60)$

$(x < 20)$ AND $(x < 40)$ AND $(x < 60)$ $\xrightarrow{O/P}$ (19, 18, 17, 16 . . .)

Nested Queries

Normal Nested Query

Execution: Inner → Outer

Bottom → Top

(Inner Query)

Co-related Nested Query

Execution: Outer → Inner → Outer

Top → Bottom → Top

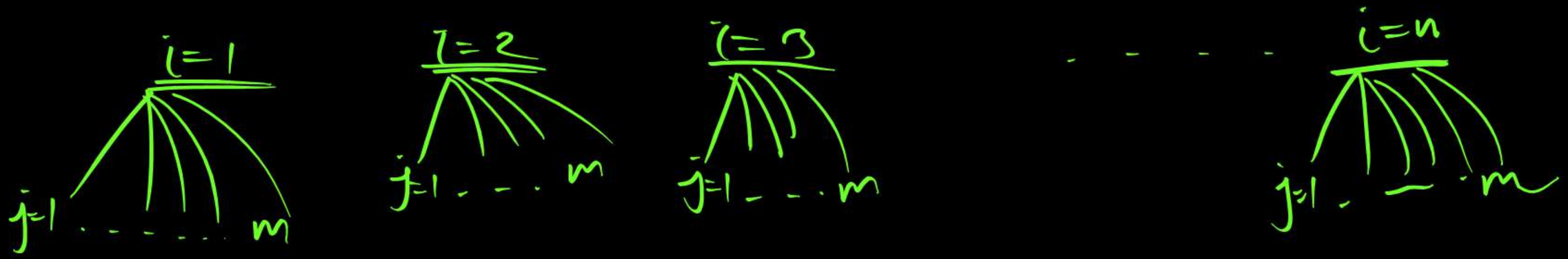
```
Select *
FROM Parts P
WHERE Pid IN/NOTIN (Select Pid
                    FROM Catalog C
                    WHERE C.Cost > 20,000)
```

Corelated Nested

```
Select *
FROM Parts P
WHERE EXIST (Select ~~*~~ 1
             FROM Catalog C
             WHERE P.Pid = C.Pid
             AND Cost > 2000)
```

Inner Query using the Attribute defined in the outer Query

```
For (i=1 ; i<=n ; i++)
    for (j=1 ; j<=m ; j++)
```



EXIST : Return True if Inner Query Result Non Empty

NOT EXIST : Return True if Inner Query is Empty

NULL ⇒ 'IS' | 'IS NOT'

'LIKE'

Q) Start from S & end with M & atleast 5 character.

Sol^n)

```
Select *
From Student
WHERE Sname LIKE S____%M
```

_ : Exactly One character

% : Zero (or) More character.

Company (Name, invoice no)

Q. Display all the company in alphabetical order of their Name?

```
Query:  Select   *
        FROM   Company
        Order by Name asc;
```

Q. If Reverse alphabetical order then
Order by Name Desc

THANK YOU!