Today's Agenda :-

1) Subqueries ✓

2) Subqueries in IN ✓

3) Subqueries in FROM ✓

4) ALL & ANY ✓

5) Correlated Subqueries

6) EXISTS

7) Subqueries in WHERE

8) Views

Q) Find all students whose psp > psp of student with id = 18

id	Name	psp	b_id
1	A	15	
2	B	41	
3	C	31	
⋮			
18	X	25	3
19	E	24	
⋮			

1) Find the psp of student with id 18

↳
```
Select psp
from Student
where id = 18
```
→ x

2) Find all students whose psp > x

```
Select *
from Students
where psp > x
```

In reality,

1) Break problem into parts

2) Solve smaller problems & we use their result to solve the bigger problem.

↳ **Subqueries**

↳ intuitive way of writing queries.

```
Select *
from Students
where psp > (Select psp
             from Student
             where id = 18)
```

→ readable ?

→ intuitive ?

Case I

```
Select *
from Students
where psp > 25
```

```
Students = [ ]
ans = [ ]
for each st in Students
    ans.add(st)
for each row in ans
    if (row[psp] > 25)
        print( ... )
```

TC : O(N)

Case II :-

```
Select *
from Students
where psp > (Select psp
             from Student
             where id = 18)
```

Nested query

```
Students = [ ]
ans = [ ]
for each st in Students
    ans.add(st)

for each row in ans
    x = -
    for each st in Students
        if (row[id] == 18) {
            x = row[psp]
        } break
    }
    if (row[psp] > x)
        print( ... )
}
```

TC : $O(N^2)$

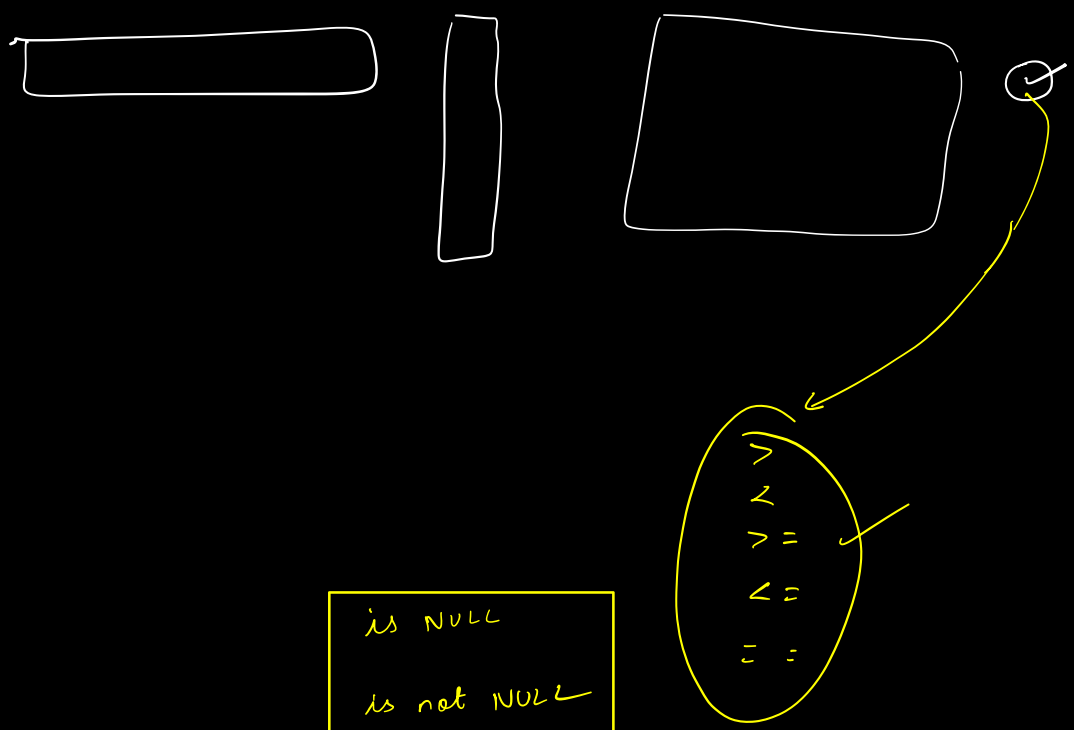
Subquery that is executed every row

↓

leads to bad performance

A Subquery can give any :-

R	C	
1	1	→ Single value
1	M	→ Single row
N	1	→ Single column
N	M	→ Table



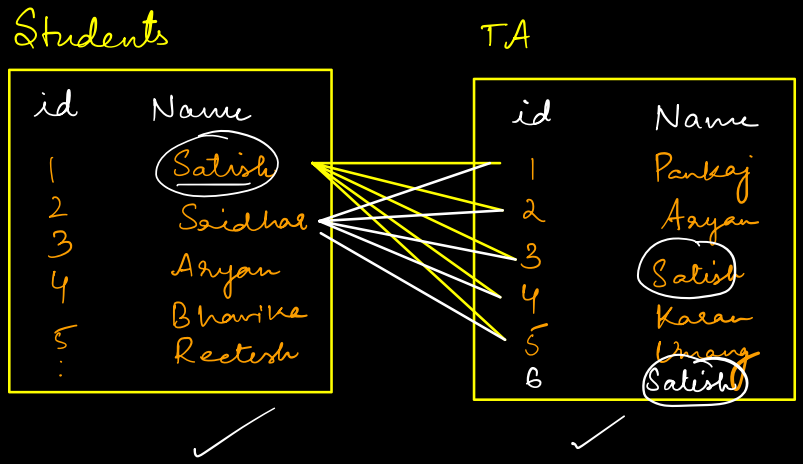
Subqueries & IN Clause :-

Users

id	Name	is_Student	is_ta
1	A	1	0
2	A	0	1
3	B	0	1
4	D	0	0

A = A ✓

Find the names of Students that are also names of TA.



Satish
Aryan

```
Select DISTINCT(name)
from Students s
join TA ta
On s.name = ta.name
```

Satish
Satish

Users s ✓

id	Name	is_Student	is_ta
1	A	1	0
2	A	0	1
3	B	1	0
4	B	1	0

Users ta ✓

id	Name	is_Student	is_ta
1	A	1	0
2	A	0	1
3	B	1	0
4	B	1	0

```
Select DISTINCT (s.name)
from Users s
join Users ta
on s.name = ta.name &&
   s.is_Student = 1 && ta.isTa = 1
```

Users s ✓

id	Name	is_Student	is_ta
1	Nikhil	1	0
2	Nikhil	0	1
3	Raj	0	0
4	Riya	1	0

Users s ✓

id	Name	is_Student	is_ta
1	Nikhil	1	0
2	Nikhil	0	1
3	Raj	0	0
4	Riya	1	0

Nikhil

Find the names of Students that are also names of TA.

① Get names of all TAs

② Get Students whose names is in

① Select distinct (name)
from Users u
where u.isTA = 1

→ [. . . . .]

② Select distinct (name)
from users u
where u.isStudent = 1 && u.name is in (———);

```
Select distinct (name)
from users u
where u.isStudent = 1 && 
      u.name in (Select distinct (name)
                 from Users u
                 where u.isTA = 1)
```

Subquery is in [— — — —]

* Subqueris should be written in parenthesis.

Q) Get me the students whose psp is greater than min psp of every batch.

Batch	Min psp
1	30
2	32
3	24

Student 39 > ✓

31 ✗

St's psp > (_ , _ , _ , _ _ .) ....

∀ batches, min psp.

① Min psp of every batch

```
Select min (psp)
from student
group by (batch_id)
```

→

min (psp)

34
29
31
42
63

32

② Get all students whose psp > all of values returned from above queries

```
Select *
from Students
where psp > ALL ( Select min (psp)
                  from student
                  group by (batch_id ))
```

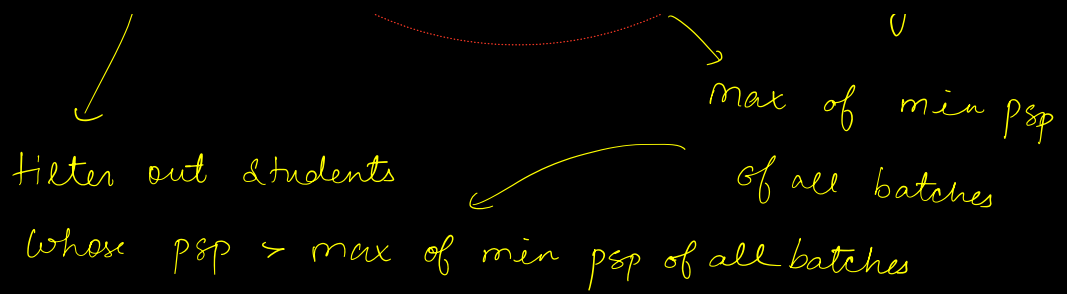
psp > ALL (10, 78, 49, 61)

only when psp > all of values, it'll be considered.

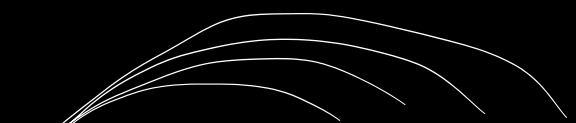
```
Select *
from Students
where psp > ( Select max(psp) from
              ( Select min (psp)
                from student
                group by (batch_id )))
```

Subquery inside from

min psp of every batch



ALL



psp > ALL (_ _ _ _ _)

Similar to AND

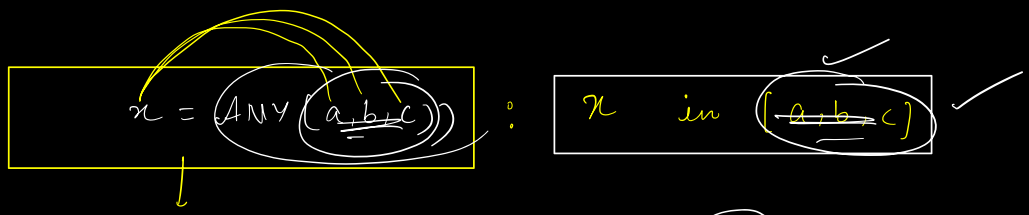
↳ Compares left hand side with every value of rhs & all of them should return true

ANY

psp > ANY (_ _ _ _)

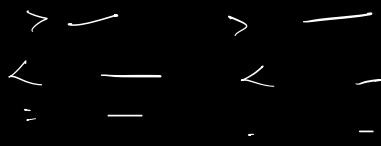
Similar to OR

↳ if any of condn return true



= ANY

ALL



Break of 7 Min

Q) Get all students whose psp > (average psp of his/her batch psp)

Batch 1 → Avg psp
" 2 →

```
Select *
from Students
where psp > (avg of his own batch)
```

x

Step 1: Get avg psp of student's batch → Correlated Subqueries

Step 2: Get students whose psp > (x)

```
Select *
from Students
whose psp > (x)
```

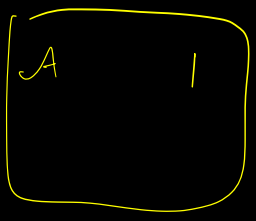
→ Avg psp of student's batch

batch_id = y

```
Select avg(psp)
from Student
where batch_id = (y)
```

→ Student's batch_id

```
Select *
from Students s1
whose psp > (Select avg(psp)
             from Student s2
             where s2.batch_id = s1.batch_id)
```




```
for each r1 in Students
    for each r2 in Student
        if (r2[b_id] = r1[b_id]
            cal avg
    if (r1[psp] > calavg)
        print (  —)
```

Q)

Students

id	Name	PSP
1	Hariprasad	
2	Bhavika	
3	Satish	
4	Manikat	

TA

id	Name	St_id
1	Satish	3
2	Bhavika	2
3	Naman	NULL
4	Mohit	NULL

Get all St_id who are also TAs.

```
Select st_id
from TA
where St_id is not NULL
```

Get names, st_id, PSP of those Student who are also tas.

```
Select *
from Students
where id in (Select st_id
             from TA
             where St_id is not NULL)
```

LHS in [- - -]

EXISTS

better in performance wrt IN.

```
Select *
from Students
where id EXISTS ( Select st_id
                  from TA
                  where st_id is not NULL)
```

uses indexing

better in performance.

```
Select t2.st_id
from ta
where Student_id = t2.st_id
```

for each row in ta

if (row[st_id]

Views :-

```
Select *
from Students
where id in (Select st_id
             from TA
             where st_id is not NULL)
```

ST;

```
CREATE VIEW [ST_TAs] AS
(Select st_id
 from TA
 where st_id is not NULL)
```