Subqueries And. Views. → OS. → Transactions.

Saturday → 3 – 3:30 hour class.

→ full text queries.
→ Clustered and Non-clustered index.
→ Insertion in mapping.

Main Benefit of subqueries.

→ Easy to read and understand complex queries.

Students

id	Name	batch_id	psp	university Name
1.	Alok	2	91	XYZ
7.	Sumit	2	84	XYZ
18.	Manju	3	87	XYZ
21.	Neha	3	92	XYZ
30.	Rahul	2	70	ABC

Quest: find all the students who have a psp greater than psp of student with id = 18

Using self join

```
select a.*
from students a
join students b
on a.psp > b.psp
and b.id = 18
```

Students a

id	Name	batch_id	psp	university Name
1.	Alok	2	91	XYZ
7.	Sumit	2	84	XYZ
18.	Manju	3	87	XYZ
21.	Neha	3	92	XYZ
30.	Rahul	2	70	ABC

Students b

id	Name	batch_id	psp	university Name
1.	Alok	2	91	XYZ
7.	Sumit	2	84	XYZ
18.	Manju	3	87	XYZ
21.	Neha	3	92	XYZ
30.	Rahul	2	70	ABC

Subqueries are mostly used to do things in a more understandable/intuitive way.

(Most of the things that we will do with subqueries can also be done using joins).

Idea

1) Get psp of student with id=18 (87)
2) Get students with psp > 87.

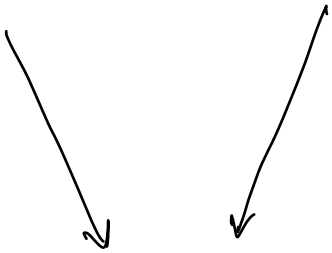
Query 1:-

```
Select psp
from students
where id=18
```

⊕

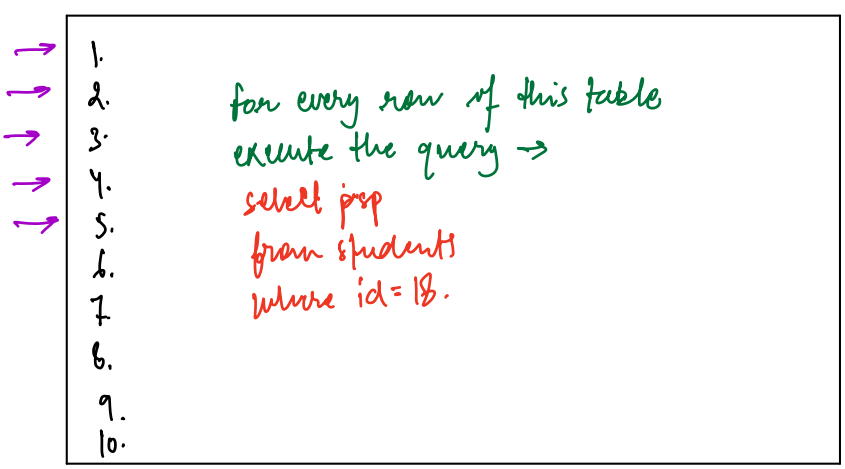
Query 2:-

```
select id
from students
where psp >87
```



```
Select id
from students
where psp > (
    Select psp
    from students
    where id=18
);
```

```
select *
from students
where condition
```



Caveats of subquery.

→ Slower Impact on performance.

Students

id	Name	batch_id	psp
1.	Alok	1	80
2.	Sumit	3	60
3.	Manju	3	4
4.	Neha	2	90
5.	Rahul	1	80

TA

id	Name	isStudent	stu_id

Print psp of students who are also a TA.

Idea1 :- Using Joins.

```
select s.psp
from ta t
join students s
on t.stu_id = s.id.
```

Idea 2 :- Using subquery

```
select psp
from students
where id IN (
    select st_id
    from ta
)
```

id IN (1, 3, 5, 7);

NOT IN

optimise this.

→ EXISTS

```
select s.psp
from students s
where EXISTS (
    select *
    from ta
    where ta.stu_id = s.id
)
```

→ Returns true even if a single row is present.

→ most of the times EXISTS is faster than IN.

Q. Select students who have psp greater than every student of batch_id=3

Students

id	Name	batch_id	psp
1.	Alok	2	91
2.	Sumit	2	84
3.	Manju	3	87
4.	Neha	3	92
5.	Rahul	2	97

Idea 1 :- Joins.

Try it out.

Idea 2:- Subquery.

```
select name
from students
where psp > (
    select max(psp)
    from students
    group by batch_id
    having batch_id=3
);
```

→ where batch_id=3

Idea3 :- ALL

```
select name
from students
where psp > ALL (
    select psp
    from students
    where batch_id=3;
);
```

> ALL (1, 2, 7, 5)
<

① Get multiple values from subquery.
② Matches condition with all values from subquery.
③ Only if all satisfy, select that particular row.

→ ANY

→ Outputs the row, if any of the subquery row matches.

psp > ANY (1, 2, 3) ✓

psp > IN (— .) ✗

→ Correlated Subqueries.

Q Select all students with psp greater than average psp of their batch.

Assume batch-id is given.

```
select name
from students (s) → outer query has a variable.
where psp > (
    select avg(psp)
    from students
    where batch_id = (s).batch_id.
)
```

→ Correlated subquery.

used by an inner query

```
select name
from students s
where psp > (
    select avg(psp)
    from students
    group by batch_id.
    having batch_id = s.batch_id.
)
```

psp > 40

psp >

batchid	avg
1	4
2	6
3	8

✗

for (———)

8:45 → Break

Non-correlated subqueries → 2N time

- N time to run the outer query
- N time to get result for inner query → this is stored and used.

Correlated subqueries → N^2 time.

- Depends on outer query.

→ Correlated subqueries are costly.

→ Try to use joins preferably.

Subqueries in SELECT

Students.

id, name, b_id, psp

o/p table.

id, name, b_id, psp, avg psp.

```
select id, name, psp, b_id
from students;
```

```
select id, name, psp, b_id, ( select avg(psp) As avg_psp
                              from students
                              group by batch_id
                              having batch_id = s.batch_id )
from students s;
```

→ returning a single cell for every student. ✓

```
select id, name, psp, b_id, ( select avg(psp) As avg_psp
                              from students
                              where batch_id = s.batch_id )
from students s;
```

✓

→ for every student

① execute the query.

② the value of the cell is put along with the row.