A	B	C
a_1	b_1	80
a_2	b_4	90
NULL	b_3	60
a_2	b_4	60
a_1	b_1	90
NULL	b_3	40
a_1	a_2	NULL

group by(A)

A	B	C
a_1	b_1	80 ✓
a_1	b_1	90 ✓
a_1	b_2	NULL
NULL	b_3	60
NULL	b_3	40
a_2	b_4	90 ✓
a_2	b_4	60

```
Select A
From R
Group by A
Having Some(C) > 80
```

a_1
a_2

a_1
a_2

```
Select A
From R
Group by A
Having C > 60
```

↓ this can't be applied for groups.

where clause is tested for each record. (tuple). (row)

Having clause is tested for each group.

Queries: Retrieve eids who get max salaries. (max ⇒ ∀)

emp(eid, Sal)

RA:

$\pi_{eid}(Emp) - \pi_{eid}(emp \bowtie_{Sal<S} \rho_{I,S}(Emp))$

$A = U - \bar{A}$ ✓

Some ✓

ρ_c

all employee

Emp Cannot be max.

eids

```
Select eid
From emp
Except (-)
Select T.eid
From Emp T1, Emp T2
where T1.sal < T2.sal
```

eid

using max:

Retrive (eids) who gets max Sal.

(Distinct) not max sal ✗

```
Select eid
From emp
where sal=(Select max (sal)
           From emp;)
```

↓

Sub query

nested non correlated Sub query.

The sub query is written inside where clause.

we can also do it in the from clause

Subquery in From Clause :-

```
Select Distinct Eid
x
From Emp,  (Select max(Sal) as msal
            from Emp temp)
where Emp.Sal = temp.msal
```

From A, B

A × B

Subquery ⇒ From clause.

Emp

Eid	Sal
e_1	70
e_2	80
e_3	60
e_4	80
e_5	40

Temp

msal
80

Eid	Sal	msal
e_1	70	80
e_2	80	80
e_3	60	80
e_4	80	80
e_5	40	80

e_4

Output

Eid
e_2
e_4

⇒ If you use inner query in where clause, the result will be of Single value / Set of values. If you write inner query in from clause, then result is a table.

Emp (eid, sal)

Retrieve emps who gets second highest salary

$$\rho(Temp, \pi_{eid,Sal}(Emp \underset{Sal \leq S}{\bowtie} \rho(emp)_{I,S}))$$

RA:

Emp

eid	sal
e_1	10
e_2	20
e_3	30

eid	sal	I	S
e_1	10	e_1	10
e_2	20	e_1	10
e_3	30	e_1	10
e_1	10	e_2	20
e_2	20	e_2	20
e_3	30	e_2	20
e_1	10	e_3	30
e_2	20	e_3	30
e_3	30	e_3	30

Temp

eid	sal
e_1	10
e_2	20

max is deleted

$\equiv$ $U - A$ $\Rightarrow$ <Some>

$\bowtie_C$

$\pi_{eid}(temp) - \pi_{eid}(Temp \underset{Sal<S}{\bowtie} \rho(temp)_{I,S})$

$\pi_{eid}(temp)$: All emp except max

$\Rightarrow$ $e_1, e_2 - e_1$

$= e_2$

eid	sal	I	S
e_1	10	e_1	10
e_2	20	e_1	10
e_1	10	e_2	20
e_2	20	e_2	20

First get all the rows except the max.

Then from the resulting table find out the max

Retrieve ~~emp~~ eids who gets second highest salary:

Emp (eid, Sal)

First remove the max.
Then find max in the remaining table

$$\int(Temp, \pi_{eid\ sal}(Emp \underset{Sal<S}{\bowtie} \int(Emp)_{I,S}))$$

$$\pi_{eid}(Temp \underset{Sal<S}{\bowtie} \int(Temp)_{I,S})$$

2nd max

Emp

eid	Sal
e_1	10
e_2	20
e_3	30

eid	sal	I	S
~~e_1~~	~~10~~	~~e_1~~	~~10~~
~~e_2~~	~~20~~	~~e_1~~	~~10~~
~~e_3~~	~~30~~	~~e_1~~	~~10~~
e_1	10	e_2	20
~~e_2~~	~~20~~	~~e_2~~	~~20~~
~~e_3~~	~~30~~	~~e_2~~	~~20~~
e_1	10	e_3	30

$\pi_{temp\ Eid}$ — Total emp.

~~e_2 20 e_3 30~~

~~e_3 30 e_3 30~~

non max emps

Temp

eid	sal
e_1	10
e_2	20

e_1 10	e_1 10
e_2 20	e_1 10
~~e_1 10~~	~~e_2 20~~
e_2 20	e_2 20

A $\geq$ Every

U − A

Etc.

A < Some

using max : SQL Find 2nd max.

Emp

Eid	Sal
e_1	70
e_2	50
e_3	80
e_4	60

(1)

```
Select max(sal)
From emp
```

80 ✓

(2)

```
From emp ✓
Where sal < (Select max(sal) ✓
             From emp)
```

80 ✓

(3)

II max.

```
Select  max (Sal)
From    Emp
where   Sal < (Select max(Sal)
               From Emp)
```

we eleminated I max.

④ ✓

Eid of II max.

Select Eid
From emp
where Sal = (Select max(Sal)
From emp
where Sal < (Select max(Sal)
From Emp))

II max.

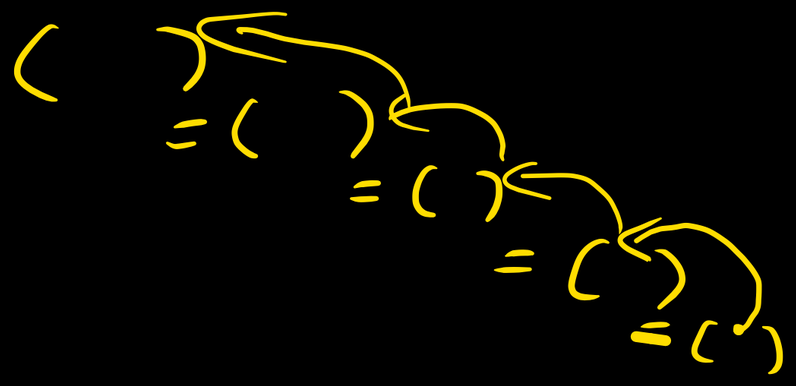
I max.

Elems I max.

nested queries:

- nested queries with out correlation
- nested query with correlation

Nested query without Corelation:



Ex:

```
Select eid
From emp
where sal = (Select max (sid)
             From Emp);
```

outer

inner

Nothing from outer query is present in inner query.

Inner query is independent of outer query. Inner query executes only once.

Execution order is from bottom to top.

nested query without correlation

Select
✓ From
✓ where
groupby
✓ Having
order by

If inner query is used in where, then o/p will be a constant

If inner query is used in having, o/p will be a constant

If inner query is used in From, its output is one table

Nested query with Corelation

R(A .....) S(B.....)

```
Select * from R
where (Select count *
       From S
       where R.A > S.B) < 2
```

30 is selected.

Output

30 ....
10 ....

```
for (i=1 to n)
    for (j=1 to n
    {
    }
```

Nested query with Correlation:

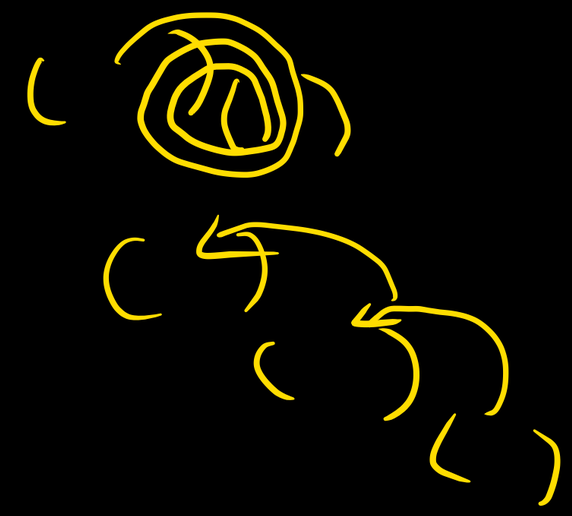
→

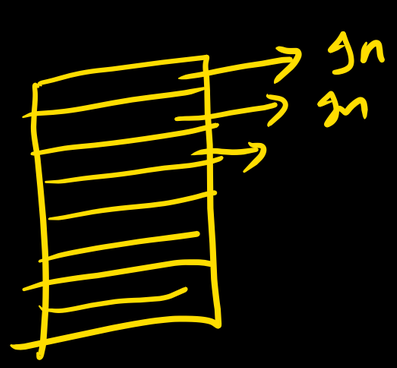
→ Inner query uses attributes of outer query

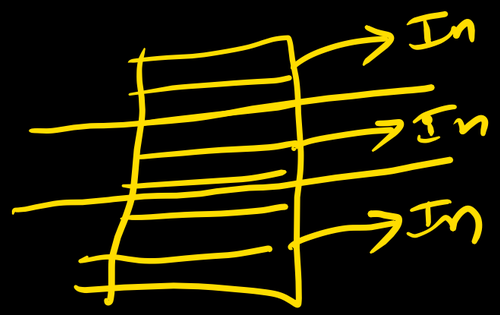
→ Execution flow is Top-bottom-top

→ If Correlation is in where clause, inner query recomputes for each record of outer query

→ If Correlation is in having clause, then inner query recomputes for each group of outer query.







→ ~~Good~~ Correlated queries runs better for complex queries

→ For simpler queries go for non Correlated queries

functions used in nested queries :-

IN / NOT IN

ANY

ALL

EXISTS / NOT EXISTS

These are best suitable for nested queries without Corelation

Best suited for queries with Corelation.

~~Go Sleep~~

```
Select *
From emp
where sal > ANY/ALL ( Select sal
                      From emp
                      where dno=5 )
```

sal → table

X

value

= IN

≠ NOT IN

8:15

Change

Instead of using set of values to compare, we have to used the above fuctions.