

Ex-1 Shopping, Bill payment - online

- Commit (~~If Transaction~~ to commit (done), then save ~~data~~)
- Roll back. (~~If Transaction~~ fails).
- Save point. (~~Ex:~~ we can save after 20-30% of transaction).

Syntax like in Downloading & Buffering,

We have save points. Ex: If downloads fails on 90%, then if it starts from '0' again or from 90%. This is Save point.

Note: To execute these commands, we can use Oracle, MySQL, Server, etc.

* Constraints → We mostly use these constraints in DDL & DML.

- Primary key (for uniqueness).
- Foreign key (" reference).
- Check
- Unique
- Default
- Not Null (mandatory *).

→ Constraints are rules which we made for store the data.

S3

Create Table in SQL with

execution →

→ In DDL Commands, very first command is Create Table.

Now, we use Oracle syntax. (In other like MySQL, SQL Server, there is only little diff.).

Create table <table-name>
(
Column 1 name datatype,
Column 2 name datatype,
Column 3 name datatype
);
desc table-name;

If no space.

Ex: abc-xyz.

If describe.

Ex:

Create table emp
(
id int,
name varchar(20),
Salary number (10)
);
desc emp;

Ex-
(2¹⁶ fixed)

→ fixed type of datatype
→ variable - " "
" (can be increased)

54.

Alter Command (DDL) in SQL :→

Use →

(किसी भी चीज़ में अल्टर डाटा चेंज करना है।)

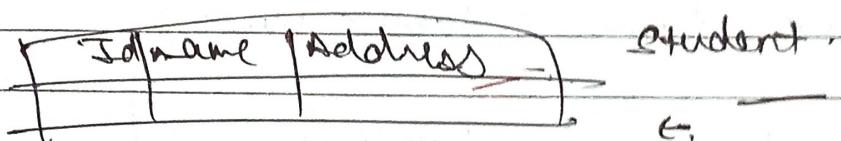
Ex:

Student

int	ID	name	varchar(20)

→ already डिजिट

- Add or Remove columns.
- modify datatype.
- Modify datatype length.
- Add constraints.
- Remove constraints.
- Rename column / table.



Alter table student add address varchar(30);

(,);

+ multiple columns.

→ Modify datatype:-

Ex:- from int to varchar

+ Create table emp

(

id int,

name varchar(10)

);

Alter table emp add address varchar(10);
desc emp;

alter table emp drop column address;

" " " " modify id varchar(30);

" " " " rename column id to reg_no;

" " " " rename to emp1;

alter table emp1 add primary key (reg_no);

(SS)

R/RB Alter & Update : →

- ④ Alter: " only change in structure, (DDL) & not in data.

St: Emp

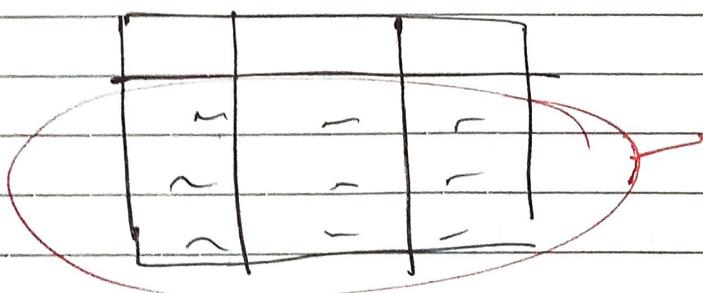
ID	Name	Salary	Email
1	A	10k	1
2	B	20k	1
3	C	30k	1

} → can only change in this struct.

- Int → varchar
- name var char (10)
- ↓
- 30 → Eid.
- Emp → Emp-detail

colⁿ
(Name change)
(table name -)

- * update: " can only change in data, (DML) & not in structure.



→ update Emp
Set Salary = Salary * 2 ; } → for all student

→ update Emp
Set Salary = Salary * 2 ; } → for only student
Where id = 1 ; of id = 1 .

(S6)

D/B

Delete, Drop & Truncate:

(a) Delete: →
(DML).

Syntax! →

Delete from table name

Ex: Delete from Student, → All rows deleted.

↓
Desc Student;

Student:

ID	name	
1	A	X
2	B	X
3	C	X

Note: But here, also flexibility, we can apply condit' for delete some particular rows.

Ex: Delete from Student
Where id = 1;

⇒ It is a slower process: - bcz, it also creates log.

→ Log! → Every file completely delete
test, Computer will temp file stored
&, to restore data, test log stored.

Ex: On delete, our data comes into
Recycle bin. (It is log file).

'Rollback';

(possible, but it creates log).

* Drop! ~~Delete~~ Delete the complete table at once.
(DDL)

`Drop table student;
Desc student;`

→ no object found.
(all deleted).



'Rollback';

* (not possible).

* Truncate! +
(DDL)

`Truncate Student;
Desc student;`

All rows delete
at once.

ID	name

→ faster process.

'Rollback'; * (but, no log).

(S7.)

Constraints in SQL →

→ Constraints means (conditions) definition.

Ex: In gmail → (2 account IDs can't be same).

Anurag@gmail.com * (not present).

Anurag123@gmail.com ✓ (present). ✓

1) Cond' → that new mail-id must be unique.

2) Cond' → length of password must be 6 digit, Capital letter etc.

Note:- We apply conditions on Attributes (column).

→ the data which fulfill these cond', only comes into the column.

1.) Unique → ex: Mobile no (no duplicate can exist in that m.no. colⁿ).

2.) Not Null! *

(We can't skip that colⁿ),

i.e., * - mandatory. (value can't be null)

3.) Primary key = Unique + Not Null.
Ex: Reg. no.

4.) Check: →

age int
x(-10)

[check (age > 18)].

→ 10 x

→ 20 ✓

i.e., check particular Domain it is in

User std. (std. & 1)

Domain - fixed.

1. Note

5.) Foreign key:

6.) Default:

Salary isn't default 10k.

If we don't fill anything, they default
Salary col^m take value \rightarrow 10k

\Rightarrow We can use multiple constraint in a col^m also.

(58) * Index

SQl Queries & Sub-Queries? \rightarrow
(from Beginning to end).

Emp.

Q1:	E-id	E-name	Dept	Salary
	1	Ram	HA	10k
	2	Amit	MRKT	20k
	3	Rani	HR	30k
	4	Nithi	MRKT	40k
	5	Varun	IT	50k

(i) Write a SQL Query to display maximum Salary from Emp Table.

(ii) Write a SQL Query to display Employee name who is taking max^m Salary.

Note: Aggregate func ! \rightarrow for max, count, average.

(i) Select max (Salary) from Emp;

Output → 5000

(ii) Use here, take help of Nested Query or Sub-Query. (Query inside a Query).

→ Select Ename from Emp where

Salary = (Select max (Salary) from Emp);

Outer Query.

Inner Query

Output → Varun

→ Inner Query execute first 1 time

Then

Outer	Inner
10k	= 50k X
20k	= 50k X
30k	X
40k	Sof = Sof. ✓

(iii)

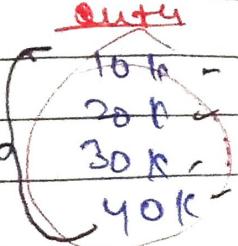
Write a SQL query to display 2nd highest Salary from Emp table?

(iv) Write a SQL query to display Employee name who is taking 2nd highest Salary?

(v)

[Logic] Highest Salary - remove 1
 Rest Salary - find highest.
 It is 2nd highest in table

Max



(Sof).

↳ Inner

Sof. X <

(iii)

Query :-

Select max (Salary) from Emp where
 Salary <> (Select max (Salary) from Emp);

Take 1 st .

Sop.

(iv).

of = , when we have to compare only }
 with 1 value.

of in , when we have to compare with }
 multiple values.

Ex:-

$$2 = 2$$

✓

$$\begin{cases} 2 = 1, 3, 2, 4, 5 \\ 2 \text{ in } (1, 3, 2, 4, 5) \end{cases}$$

(Wrong way)

✓ (True)

Query :- ✓ only add name in (iv).

Select E-name from Emp where

Salary = (Select max (Salary) from Emp where
 Salary <> (Select max (Salary) from Emp));

(Output → Nithi)

X

X

60.

(v) Write a Query to display all the
 dept names along with no. of Emps.
 working in that ?

Output:-

HR	2
M.R.K.T.	?
I.T.	1

Ex:- In a university, find
 how many students are
 there in diff. branches.

A.I - 101
C.S.E - 100

Note:- Here, we use a special func'.

[group by] clause:

(forms grp of dept.).

2	→	2
-	HR HR	-
-	MRKT MRKT	-
-	IT.	1

Note:- Cond' of group by → ~~or att. other than~~

~~if group by on \exists att. use \exists on \exists ,
then att. Select _____ on \exists Third stand.~~

If,

we have to write other att. in Select _____
other than the att. of group by func,
then we have to use aggregate func'.

• Query:-

Select dept from emp group by dept;

[Output] →

HR	}
HR	
MRKT	}
MRKT	
IT	

→ Same.

* Note:-

Count(dept) or Count(*)

• Query:-

aggregate func.

Select dept, Count(*) from emp group by dept;

Output :-	HR - 2
	MRKT - 2
	IT - 1

Q61. (iii) Write a Query to display all the dept names where no. of Emps are less than 2.

HR 2

MKT 2

IT 1

sh.

Note:- 'Where' clause works on complete Table, but now we group by this table. Hence, group by & where are independent. not help each others.
Hence,
we use 'having'.

Query:-

Select dept from Emp group by dept
having Count(*) < 2;

Output → IT.

Q:- If they ask of Employee name in this query, then → (return 7 output)

Query →

Select E-name from Emp where dept In
(Select dept from Emp group by dept
having Count(*) < 2);

↳ Warun.

* HR IT
* MKT
* IT sh.

(62)

(Qn.) Write a query to display highest salary department wise and name of emp who is taking that salary.

Note: A SQL query always starts from 'from'

Select — from table name.

Note: Select max(y) from nn group by X;

Here, It is Right, bcz we use aggregate func ('max') with 'y'

If max(y) & if we don't use max, then we can only use X, bcz group by X. (Lam).

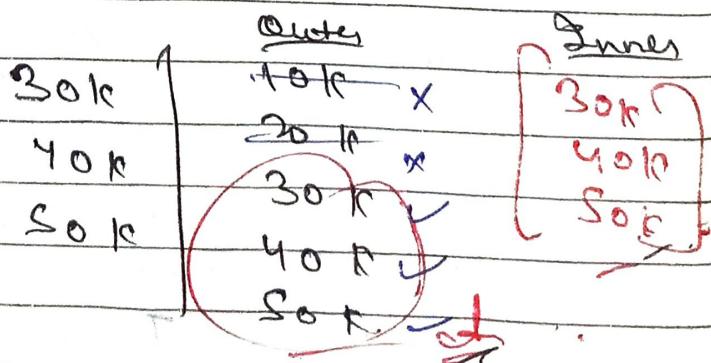
Query!

Select Ename from Emp where Salary In (Select max(Salary) from Emp group by dept);

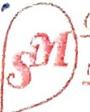
Output → Rani, Nitin, Maran... dr.

Steps →

HR	HR - 30,000
HR	
MRKT	MRKT - 40,000
MKT	IT - 50,000
IT	



* means All attributes.



Date: _____
Page: _____

101

63. Use of IN and Not IN

Simple
in Query.

1 = (1, 2, 3, 4, 5)

X (Wrong way).

1 = 2

false. (but, Right way)

Emp

E_id	E_name	Address
1	Ravi	Chd
2	Varun	Delhi
3	Nitin	Pune
4	Robin	Bangalore
5	Ammy	Chd.

Project

E_id	P_id	Pname	Loca^n
1	P ₁	IOT	Bangalore
2	P ₂	Big Data	Delhi
3	P ₃	Retail	Mumbai
4	P ₄	Android	Hyderabad

Q:- Detail of Emp whose address is either Delhi or Chd or Pune;

Query:-

Select * from Emp where Address = 'Delhi';

Output:-

2	Varun	Delhi
---	-------	-------

⇒ But, we have 3 cities. So,

Query:-

Select * from Emp where Address In ('Delhi', 'Pune', 'Chd');

Output:-

1	Ravi	Chd
2	Varun	Delhi
3	Nitin	Pune
4	Ammy	Chd.

Chd ✓

Delhi ✓

Pune ✓

Bangalore ✗

Chd. ✓

Nested Query or Query (Query)

Now, NOT IN : (means not included).

Query :-

Select * from Emp Where Address Not In ('Delhi', 'Pune', 'Chd');

Output :-

#	Robin	Bangalore
---	-------	-----------

Chd x

Delhi x

Pune x

Bangalore ✓

Chd x

Q4 - Use of IN & Not IN in Subquery :-

(Same 2 tables of before).

Query :- find the name of Emps who are working on a project of .

Suggestion :- First make Dummy Tables.

Here, we need both 2 Tables.

Both tables have common - EID
So, we compare by taking EID.

Nested → Bottom up → जटि के 342

Means first write query & then start writing

Inner Query in Nested Query - Runs 1 time.

51

Date:

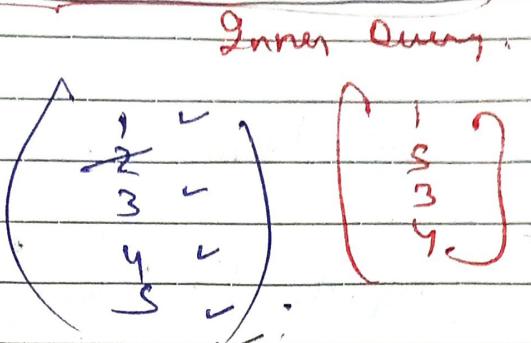
Page:

122

* Query :-

Select Ename from Emp where Eid
In (Select Distinct (Eid) from Project);

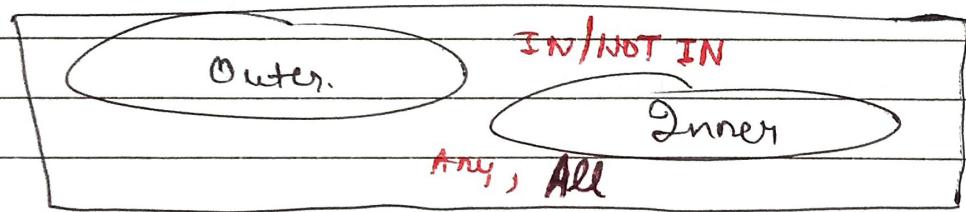
Output:-
Rani
Nitin
Robin
Anny



65

Exist & NOT Exist Subqueries →

→ we use these in Correlated Nested Query.



→ In nested query. - we use IN / NOT IN.

In correlated nested query. we use EXIST / NOT EXIST.

Query : find the detail of Emp who is working on at least one Project ?
(Same 2 Tables)

Note:- In nested Query → Inner Query executes first,
then compare its output with Outer Query.
In Correlated Nested Query : → the one row of Outer
Query is compared with all rows of inner Query.
ie, Top to Down Approach.

Null → means value is not available
Empty

SM Page

Q) Query:

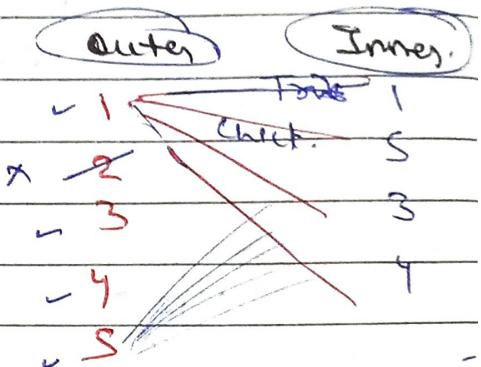
Select * from Emp where S_id

Exists (Select E_id from Project where Emp.S_id = project.E_id);

Note: Exist / not Exist gives True or False.

Output:

1	Ram	Chd.
3	Nitin	Pune
4	Robin	Bang.
5	Ammy	Chd.



66.

Aggregate funcs in SQL!

Max, Min, Count, Avg, Sum.

(Total no. of values in a Table).

Emp

E_id	E_name	Dept	Salary
1	Ram	HR	10k
2	Amrit	MR&T	20k
3	Ram	HR	30k
4	Nitin	MR&T	30k
5	Varun	IT	50k
6	Sandy	Testing	Null.

Q) Query for Max. Salary:

→ Select max(Salary) from Emp;

Output → 50k

ok

→ If SQL repeats 2 times, then it also gives output 2 times.

4 Same for Min .

→ [Select Min (Salary) from Emp;]
Output: 10K.

* COUNT:-

Count (*) Means no' of rows in the Table.

→ [Select Count (*) from Emp;]
Output → 6. 6 rows.

→ [Select Count (Salary) from Emp;]
Output → 5 (bcz one value is Null)

* Distinct:-

→ [Select Distinct (Count (Salary)) from Emp;]
Output → 4. (bcz 30K is 2 times).

* SUM:-

→ [Select Sum (Salary) from Emp;]
Output → 140 K. (sum of all salary).

→ [Select Distinct (Sum (Salary)) from Emp;]
Output → 110 K. (30K repeats).

AVG :-

$$\text{Avg (Salary)} = \frac{\text{Sum (Salary)}}{\text{Count (Salary)}}$$

$$= \frac{140k}{5}$$

$$= 28k$$

→ Select Avg (Salary) from Emp;
 Output: → 28k.

→ Select Distinct Avg (Salary) from Emp;
 Output: → 27,500.

$$\text{Distinct (Avg (Salary))} = \frac{\text{Distinct (Sum (Salary))}}{\text{Distinct (Count (Salary))}}$$

$$= \frac{110k}{4}$$

$$= 27,500$$

67 Correlated Subquery in SQL: →
 (Synchronized Query).

- It is a subquery that uses values from Outer Query.
- Top to Down Approach: (Outer to Inner)
- for One row of outer Table it compare with every row of inner Table.

→ returns True / False.

Emp

Eid	Name	Address
1	A	Delhi
2	B	Pune
3	A	Chennai
4	B	Delhi
5	C	Pune
6	D	Mumbai
7	E	Hyd.

Dept

D-id	D-name	E-id
D ₁	HR	1
D ₂	IT	2
D ₃	MRKT	3
D ₄	Testing	4

Query:- Find all Employee detail who work in a department.

→ True Eid के दौरान Employee Detail का किया।

→ Query:-

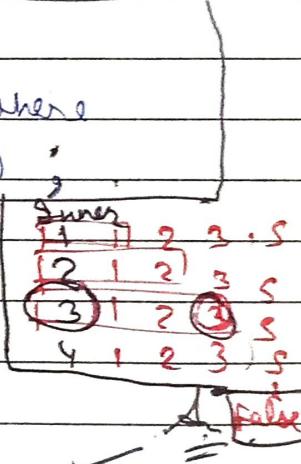
Select * from Emp where

exists (Select * from Dept where

Dept.Eid = Emp.Eid) ;

Output:-

Eid	Name	Address
1	A	Delhi
2	B	Pune
3	C	Chennai
4	B	Delhi



68

DB Joins, Nested Subquery & Correlated Subquery :-

Subquery :-

Nested → Bottom up. (ऊपर से नीचे)

Correlated → Top down approach (नीचे से ऊपर)

Joins → Cross product + Cond'

~~Emp~~

E-id	name
1	A
2	B
3	C
4	D
5	E

Dept.

PK

Dept no	name	E-id
D ₁	IT	1
D ₂	HR	2
D ₃	MBKT	3

Q1: Detail of all Emp who works in any dept.

(Here, we need both 2 tables).

→ Nested Subquery →

Select * from Emp where E-id in
(Select e-id from dept);

Output →

1	A
2	B
3	C

No

→ Correlated Subquery →

Select * from emp where exists
(Select id from dept where
emp.e-id = dept.e-id);

e-id is
common
in both
Tables

Output? :

1	A
2	B
3	C

Index

1 = 1

2 = 1

Index

1 = S

2 = S

3 = S

False

→ If 1st Table $\rightarrow m$ rows

2nd Table $\rightarrow n$ rows

then,

Total Comparison $\rightarrow m \times n$

→ JOINS : →

Cross product + Condⁿ

It creates a new Table with $(m \times n)$ rows.

→ then,

check Condⁿ \rightarrow emp.eid = dept.eid

Notes

Joins is faster than Correlated Subquery
bcz it made a Table of rows $(m \times n)$
at Once. While in Correlated, we again
& again compare one by one row of
Outer Table with all Rows of Inner Table.
So, It takes time. But,
Joins take more space. (bcz big Table of
 $(m \times n)$ Rows).

Q69

Find Nth Highest Salary using SQL : →

Emp.

ID	Salary
1	10k
2	20k
3	20k
4	30k
5	40k
6	50k