SQL

PAGE No.
DATE / /

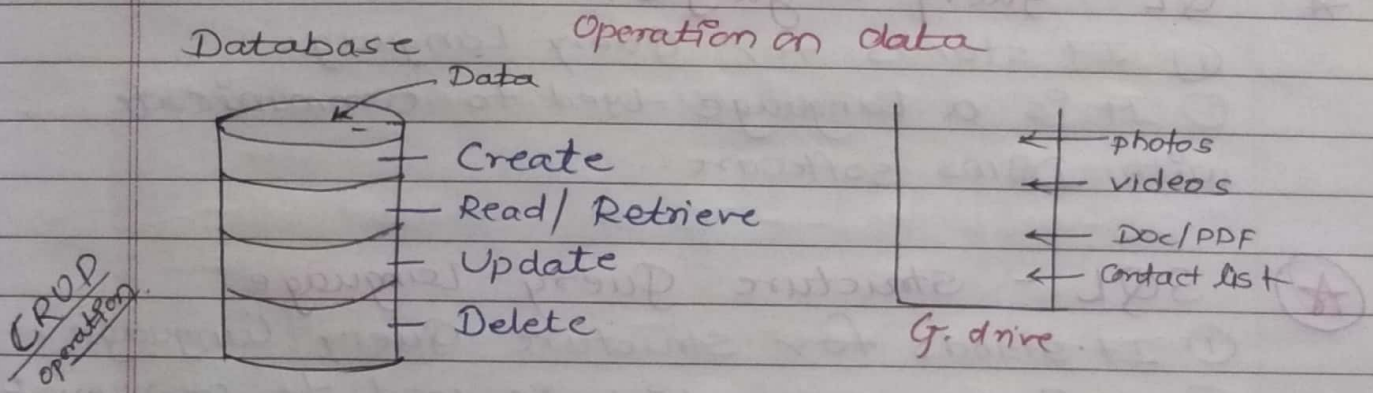
Database

i) **Data**: Data is a raw fact which describes attributes of an entity.

ii) **Attributes**:- It is an properties of an object

iii) **Entity**: It is an object

iv) **Database**: It is place or medium where we store data in systematic and organised manner.



DBMS → Database Management System

① It stands for Database management system

② It is a software which maintain and manages the database.

Important features of DBMS

1) Security → (to protect)
2) Authorisation → (to give access/permission)

TYPES OF DBMS

1. Relational DBMS - (RDBMS) - HUMP ← Highly used most popular
2. Network DBMS
3. Object oriented DBMS
4. Heirarchial DBMS

Ego → Oracle, MySQL

(A) RDBMS :- Relational DataBase Management System.

① It is a type of DBMS software where we store Data in the form of tables or relations.
② In RDBMS we can store unlimited data.

* Other Defination of RDBMS :
① If DBMS software follows Relational model then it is known as RDBMS
② If DBMS software follows E.F. Codd Rules then it is knowns as RDBMS.

* SQL - Query Language
① It stands for Query Language
② It is a Language tyed to communicate with DBMS software.

(A) SQL - Structure Query Language.
① It stands for Structure Query Language.
② It is language which is used to communicate with RDBMS software

(A) Relational Model
① It is designed by Data Scientist E.F. Codd (Edger Frank Codd)
② when data is stored in the form of tables or relation then we called it as relational model.

(A) Imp E.F. Codd Rules
① Data to be entered into cell should be single value data or atomic data
② we can store data in multiple table and

established the connection between those two table. using key Attribute.

③ To validate the data we have to assign datatypes and constraints.

④ Datatypes are mandatory & constraints are optional.

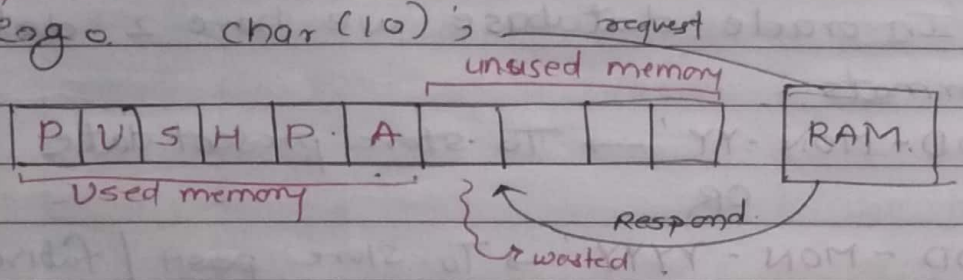
Datatype.

Define:- Rules given to the table is known as Datatype.

① char → It is used to store 'A'-'Z', 'a' to 'z' '0' to '9', special characters like '@', '$', '#'

Syntax → char(size);

eg:- char(10); — request



① It can store upto 2000 memory space.

② Unused memory in char datatype will be wasted.

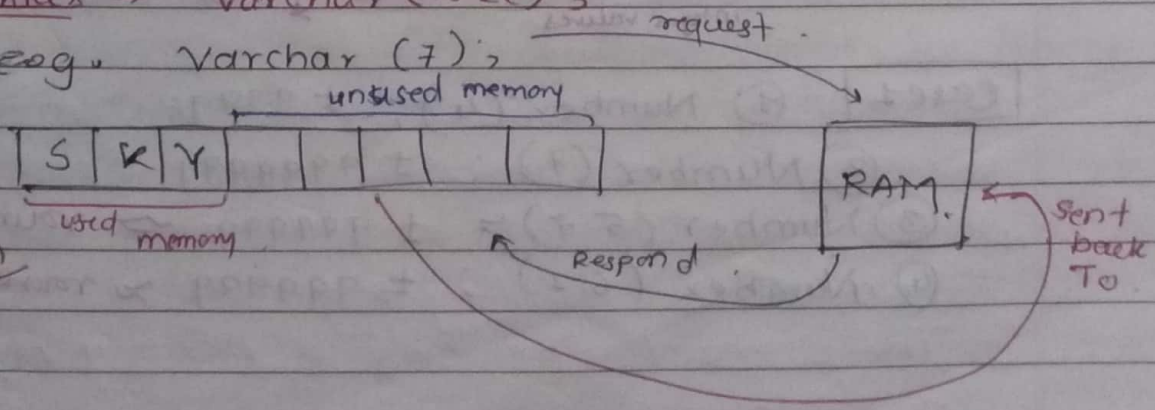
③ It also called as fixed length memory allocation.

② Varchar:- It is used to store 'A' to 'Z', 'a' to 'z' '0' to '9', special character like '@', '$', '#'

Syntax → Varchar(size);

eg:- Varchar(7);

Variable length memory allocation



(14)

PAGE No.
DATE / /

Previous page

① It can be store upto 2000 memory space
② Unused memory in Varchar will not go wasted. It will be sent back to ~~RAM~~ RAM usage.
③ Another name for Varchar is Variable length memory Allocation.

i) Varchar 2 ;- ① It is an upgradation of Varchar
② It can store upto 4000 memory space.

③ DATE

In oracle database we have 2 date formats.

'DD-MON-YY' → To store present date

OR

'DD-MON-YYYY' → To store past / future

Syntax: DATE;

④ NUMBER →

It is used to store numeric values

Number (Precision, [Scale]); → end of stmt

Precision: max. no. of integer values
[Scale]: optional; max. no. of decimal values

Case 1 ① Number (4); ± 9999
② Number (7); ± 9999999
③ Number (5,7); ± 999999 ≈ round of 6
④ Number (6,2); ± 999999 ≈ roure of 6

ASCII - American Standard Code for Information Interchange.

Case 2

① Number (4,2); ± 99.99
② Number (6,3); ± 999.999
③ Number (8,2); ± 999999.99
④ Number (6.5, 2.3); ±99999.99

Case 3

① Number (3,7); ± 0.0000999
② Number (2,6); ±0.000099 9
③ Number (0,4); 0
Precision cannot be ZERO.

⑤ Large object (LOB):
It is used to store data in large amount
In that we have two types.

① char large object
② Binary large Object.

① Char large object It is used to store text files, .xml files in large amount
syntax:- CLOB;
Memory size is 4GB.

② Binary Large Object It is used to store multimedia file audio, video in binary format
Syntax: BLOB;
Memory size is 4 GB.

(3) (A) Constraint : Condition given to the table is known as Constraint

(1) Unique → It is used to avoid duplicate while entering to the table.

(2) Not null → It is a constraint which is used to display that some value is present is cell.

(3) Check → Additional condition given to table is known as Check. : check (Sal > 0)

Unique Number	NN Varchar	NN Number	NN Varchar	
ID	Ename	Sal	Loc	
1	Thor	700	Goa	Check
2	Captain	800	Ladakh	(Age > 29)
3	Stark	900	Goa	
4	loki	1000	Alibaug	
5	Groot	2000	Goa.	

(4) (A) Primary key :-

(1) To uniquely identify records we use primary key.

(2) To represent the complete table we need primary key.

Characteristics :- (1) It should be unique

(2) It should ~~not be null~~ be not null.

(3) Combination of unique and not null is known as primary key.

(4) There should be only 1 primary key present in table.

⑤ primary key is not mandatory but design wise it is preferable.

(A) Foreign Key

To establish the connection between two tables we need foreign key.

Characteristics: ① It can have duplicate values.
② It can have null values.
③ Combination of unique and not null is not required.
④ Foreign key is not mandatory.
⑤ If any column wants become foreign key it should be primary key of its own table.
⑥ It belongs to parent table but it stays in a child table.
⑦ It is also called as referencial integrity ~~concept~~ constraint.

Foreign key ← column name

primary key

Number Unique NN — primary key

varchar — Numbr — Numbr

ID	Ename	Sal	deptno
1	Virat	7000	10
2	Rohit	9000	20
3	Rahul	1000	10
4	Surya	5000	30
5	Bumrah	9000	20
6	Vrushabh	6000	30
7	Ronaldo	1000	

primary key

NN Unique number — NN varchar — N.N varchar

deptno	dname	loc
10	Accounting	Goa
20	Database	ladakh
30	Developer	Alibaug

(★) SQL Languages.

1. DDL - Data Definaction Language
2. DML - Data Manupulation Language
3. TCL - Transaction Control Language
4. DCL - Data Control Language
5. DQL - Data Query Language

Creation of table

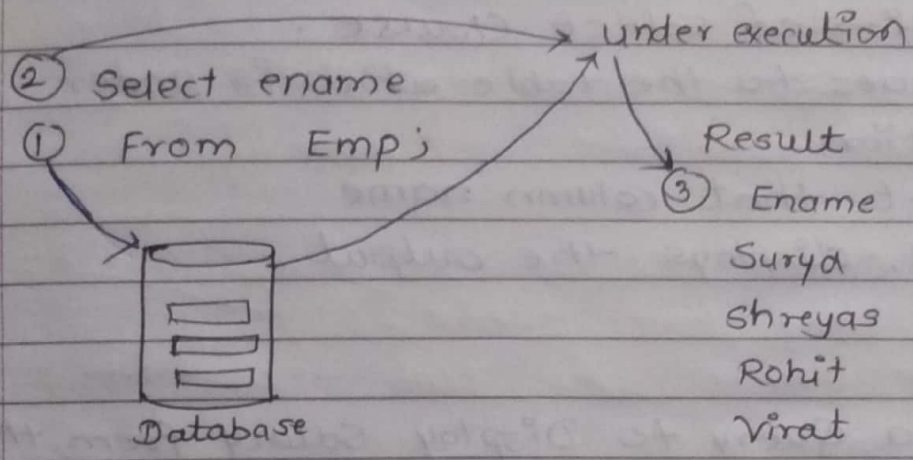
used for managing tab

(★) DQL - Data Query Language

① There are 4 statements.

1. SELECT - Retrieval of the data from table using database.
2. PROJECTION - It is retrieval of data from the table using column name.
3. SELECTION - It is retrieval of data from the table using column name and row name.
4. JOINS - It is a retrieval of data from multiple table simultaneously.

ID	Ename	Sal
1	Surya	1000
2	Shreyas	500
3	Rohit	9000
4	Virat	1000
5	Bumrah	2000



PROJECTION :- Combination of SELECT and FROM clause is called as PROJECTION.

SELECT * [DISTINCT / Column-Name / Expression / ALIAS NAME]
FROM Tablename;

(A) FROM CLAUSE

① It will always execute first.
② We can write multiple table name in from clause

* function of from clause

① It goes to the database.
② Search for the table.
③ put that table under Execution.

PAGE No.
DATE / /

② SELECT CLAUSE

① It executes after from clause.
② It is used to display the output
③ We can write multiple column name in Select clause.

* function of select clause.

① It goes to the table which is under execution.
② select that column name
③ And, displays the output.

Q.] Write a Query to Display Salary from the table Employee.

→ Select Sal
From Emp;

Q) Write a query to display employee name, salary, job

```
SELECT ENAME, SAL, JOB
    FROM EMP;
```

A) NOTE:- select (*) → Asterisk (which is used to display all the records from the table)
from emp;
↳ Tablename

* Write a query to display Ename, Sal, Date of join, Commission from table emp

```
Select Ename, Sal, HireDate, Comm
 From emp;
```

Q) Write a query to display Ename, Dept no, manager from table

```
SELECT ENAME, DEPTNO, MGR
FROM EMP;
```

Q) Write a query to display Dept-no, Dept name, location from table

A) NOTE:- Select * from Tab; } This command is used to display all tables in database

```
SELECT DEPTNO, DNAME, LOC
 FROM DEPT;
```

A) CL SCR (clear screen command)

PAGE No.
DATE / /

(3) Write a query to display employee name, salary, job

```
SELECT ENAME, SAL, JOB
    FROM EMP;
```

(4) NOTE:- select (*) → Asterisk (which is used to display [all] the records from the table)
from emp;
↳ Tablename

* Write a query to display ~~emp~~ Ename, Sal, Date of join, Commission from table emp

```
Select Ename, Sal, HireDate, Comm
  From emp;
```

(5) write a query to display Ename, ~~Sal~~ Dept no, manager from table

```
SELECT ENAME, DEPTNO, MGR
FROM EMP;
```

(6) write a query to display Dept-no, Dept name, location from table

(7) NOTE:- select * from Tab; } This command is used to display all [tables] in database

```
SELECT DEPTNO, DNAME, LOC
  FROM DEPT;
```

(8) CL SCR (clear screen command)

PAGE No.
DATE / /

8.] Select
Write a query to display Deptno without duplicate from Emp table.

⇒ Select Distinct deptno
from emp;

9.] Difference between Unique & Dinstinct

UNIQUE	DISTINCT
i) It is constraint	① It is a keyword
ii) It is used to avoid duplicate while inserting records	② It is used to avoid duplicates in resultant table.

Q.] WAQ to display ename, job, higher date, monthly sal from table.

→ SELECT ENAME, JOB, HIREDATE, SAL
FROM EMP;

Q) WAQ to display ename, job, annual sal from table.

→ SELECT ENAME, JOB, (SAL * 12)
FROM EMP;

Q) WAQ to display ename, job, empno, sal, ~~sal~~ addition of sal & commision

→ ~~SELECT ENAME, SAL, COMM,~~ ~~SAL, SAL~~

SELECT ENAME, JOB, EMPNO, SAL, (SAL+COMM)
FROM EMP;

Expression:-

Any operation which gives some output then it is known as expression.

There are two types of expression.

1. Direct Value
2. Immediate Value

Direct Value	Immediate Value
eg. Sal * 12 (Sal: dynamic, 12: constant)	eg. Sal + comm (Sal: dynamic, comm: dynamic)

Q.] WAQ to display ename, sal, job, 15% hike in their salary from table.

```
SELECT ENAME, SAL, JOB, (SAL * 1.15)
FROM EMP;
```

Q.] WAQ to display ename, job, annual sal, 10% deduction in annual salary from table.

```
SELECT ENAME, JOB, SAL*12, SAL*12*0.9
FROM EMP;
```

Q.] WAQ to display ename, job, monthly sal, annual sal, mid-year sal, Quarter year sal, monthly comm, annual comm, mid year comm, quarter year comm. sum of sal&comm, from table.

```
SELECT ENAME, JOB, SAL, SAL*12, SAL*6,
SAL*3, comm, comm*12, comm*6,
comm*3, SAL+comm
FROM EMP;
```

SET ~~100~~ PAGES 100 LINES 100

PAGE No.
DATE / /

(*) ALIAS NAME

It is an alternative name or temperorory name given to Column.

There are four ways to give Alias name.

① By using space

```
SELECT SAL*12 ANNUAL
FROM EMP;
```

② By using AS Keyword.

```
SELECT SAL*12 AS YEARLY
FROM EMP;
```

③ By using underscore (_)

```
SELECT SAL*12 ANNUAL_SALARY
FROM EMP;
```

④ By using (" ")

```
SELECT SAL*12 "ANNUAL SALARY"
FROM EMP;
```

PAGE No.
DATE / /

8.] WAQ to display of Employeee working as manager
```
SELECT *
FROM EMP
WHERE JOB = 'MANAGER';
```

9.) WAQ to display all details of employee who are ~~high~~ hired before year 87
```
SELECT *
FROM EMP
WHERE HIREDATE < '01-JAN-87';
```

9. WAQ display all details of employee who are working in dp no. 10
```
SELECT *
FROM EMP
WHERE DEPTNO = 10;  OR  DEPTNO = '10';
```

⭐ Characteristics of WHERE CLAUSE
1. It is used to filter the condition
2. It executes row by row.
3. It executes after the execution of FROM clause.
4. We can write multiple condition in WHERE clause
5. we cannot write 'alias' name in where clause.

* OPERATORS *

1. Arithmatic (+, -, *, /)
2. Comparison (=, != or <>)
3. Relational (<, >, <=, >=)
4. logical (AND, OR, NOT)
5. Concatenation (||)
6. Special (IN, NOT IN, BETWEEN, NOT BETWEEN, IS, IS NOT, LIKE, NOT LIKE)
7. Subquery (ALL, ANY, EXISTS, NOT EXISTS)

LOGICAL OPERATOR

1. AND → It is used when both condition needs to be True.
2. OR → It is used when either one condition needs to be true.
3. NOT → It is used to ~~negtec~~ negate the value.

Q] WAQ to display all details of Employee who are working as MANAGER in dept.10

```
SELECT *
FROM EMP
WHERE JOB = 'MANAGER' AND DEPTNO = 10;
```

Q] WAQ TDAE who are working in DEPNO 20,30

```
SELECT *
FROM EMP
WHERE DEPTNO = 20 OR DEPTNO = 30;
```

Q) WAQ ADOE who are working as Analyst, president in Dept 10, 20

```
SELECT *
FROM EMP
WHERE (JOB = 'ANALYST' OR JOB = 'RESIDENT')
        AND (DEPTNO=10 OR DEPTNO = 20);
```

Q) WAQ ADOE who are working as Analyst, CLERK, MANAGER in DEPT 10, 20 ~~getting~~ salary >= 1250

```
SELECT *
FROM EMP
WHERE (JOB = 'ANALYST' OR JOB = 'CLERK' OR
        JOB = 'MANAGER') AND (DEPTNO = 10
        OR DEPTNO=20) AND (SAL >= 1250);
```

Q) WAQ ADOE Who are not working as president

```
SELECT *
FROM EMP
WHERE  JOB != 'PRESIDENT';
        OR
    NOT JOB = 'PRESIDENT';
    OR JOB <> 'PRESIDENT';
```

Q) WAQ ADOE who are working in department 20,30 not As a Analyst, clerk.

```
SELECT *
FROM EMP
WHERE (DEPTNO = 20 OR DEPTNO=30) AND
                NOT (JOB = 'ANALYST' OR JOB='CLE
```

!= (comparison)
<>

Eg (A)
```
SELECT *
FROM EMP
WHERE JOB != 'MANAGER';
OR    JOB <> 'MANAGER'
```

Q] WAQ ADDE who are hired in year 81
```
SELECT *
FROM EMP
WHERE (HIREDATE >= '01-JAN-81') AND
      (HIREDATE <= '31-DEC-81');
```

Q. WAQ ADDE who are working in DEPTNO 10,30 AS MANAGER, SALESMAN
```
SELECT *
FROM EMP
WHERE (DEPTNO = 10 OR DEPTNO = 30) AND
      (JOB = 'MANAGER' OR JOB = 'SALESMAN');
```

Q.] Special Operator

1] IN Operator

① It is a multivalue operator which consist multiple values at RHS single value at RHS.

Syntax:

Column-name / Exp IN (V1, V2, ... Vn);
(LHS) (RHS)

eg logical op ⇒ deptno = 10 OR deptno = 20 OR deptno = 30

Special ⇒ deptno IN (10, 20, 30);
(Selected)
(=) (OR)

Q.] WAQ ADOE who are working in Dept 10, 20 as president, analyst, clerk getting salary < 5000 (IN)

```
SELECT *
FROM EMP
WHERE ~~((DEPTNO = 10)~~
      ( DEPTNO IN (10, 20) ~~OR~~ AND
            JOB IN ('presiden', 'ANALYST', 'CLERK')
                AND SAL < 5000);
```

Q] NOT IN Operator

① It is a multivalue operator which consist multiple value at RHS & single value at LHS.

② In NOT IN Operator RHS values are Rejected.

Syntax:

Column-name / Exp NOT IN $(V_1, V_2 \ldots\ldots V_n)$;

(Exp = LHS, $(V_1, V_2 \ldots\ldots V_n)$ = RHS)

logical op ⇒ DEPTNO != 10 AND DEPTNO != 20;

Special op ⇒ DEPTNO NOT IN (10, 20);

(NOT IN = (!=), (10, 20) = (AND); Rejected)

Q.] WAQ ADOE JOB, SAL, HIREDATE, COMM. from table of emp are working as manager clerk salesman except those who are working in dept 10, 20 hired after year 80 getting sal >= 1200, AND <= 3000

```
SELECT Ename, sal, hiredate, comm
from emp.  (OR)  JOB IN ('manager', 'clerk', 'salesman')
WHERE (JOB = 'manager' OR JOB = 'clerk' OR JOB = 'salesman')
      AND Deptno not in (10,20) AND
      (HIREDATE > '01-DEC-~~80~~') AND
      (SAL >= 1200 AND SAL <= 3000
```

IN RANGE → AND
OUT OF RANGE → OR

PAGE No.
DATE / /

```
SELECT Ename, JOB, SAL, HIREDATE, comm
From emp
WHERE (JOB IN ('MANAGER', 'CLERK', 'SALESMAN'))
      AND DEPTNO NOT IN (10,20) AND
      HIREDATE > '31-DEC-80' AND
      (SAL >=1200 AND SAL<=3000);
```

* who are working as Manager analyst in dept 10,20 getting sal >= 1250 sal <= 4000 getting comm > 300

```
select *
from emp
where JOB IN ('MANAGER', 'ANALYST') AND
      DEPTNO IN (10,20) AND
      (SAL >= 1250 AND SAL <=4000) AND
      COMM > 300;
```

* WAQTD ADOE who are getting SAL > 1400 SAL < 5000

```
Select *
from emp
where SAL > 1400 AND SAL < 5000;
```

* WAQTD ADOE who are getting SAL < 1250 SAL > 2975

```
Select *
from emp
where SAL < 1250 AND SAL > 2975;
      OR
      SAL > 2975 AND SAL < 1250;
```

* WAQTD ADOE who are getting SAL >= 2850 SAL <= 5000

```
Select *
from emp
where SAL >= 2850 AND SAL <= 5000;
```

PAGE No.
DATE / /

BETWEEN OPERATOR.

① It is used to find output when values are IN RANGE.

Syntax:

Column-name / Exp | BETWEEN | lower Range | AND | Higher Range

Note: In between operator Range of values are also selected.

logical op → Sal >= 2850 AND Sal <= 5000;

special op → Sal BETWEEN 2850 AND 5000;

Q.] WAQTD ADOE who are getting SAL >= 1450 SAL < 3000

```
SELECT *
FROM EMP
WHERE   SAL>=1450 AND SAL<3000;
                OR
        SAL BETWEEN 1450 AND 2999;
```

Q. who are getting comm > 200 comm <= 1500

```
SELECT *
From emp
where   comm BETWEEN 201 AND 1500;
```

NOT BETWEEN

① It is used to get out when values are not in Range.

Syntax:

Column.name/Exp NOT BETWEEN lower Range AND Higher Range.

Apna

PAGE No.
DATE / /

NOTE: In NOT BETWEEN operater range of values are not selected.

logical op ⇒ Sal < 1250 OR Sal > 3000;

special op ⇒ Sal not Between 1250 AND 3000;
complete range will be Rejected

Q. WAQTD ADOE who are getting Sal < 1500
Sal > 2000

```
Select *
From emp
where sal not between 1500 AND 2000;
```

Q] WAQTD ADOE who are getting Sal <= 2450
Sal >= 3000

```
Select *
From emp
Where sal not between 2450 AND 3000;
```
2451 2999

Q.] WAQTD ADOE who are working as Manager salesman president, analyst except those who are working in dept (10,20) hired before year 85 getting sal >= 1500 sal < 5000 and getting comm <= 500 comm >= 1400

```
Select *
from emp
where ( JOB IN ('MAnager', 'Salesman', 'president', 'analyst')
      AND ( NOT DeptNo Not in (10,20)
      AND HIREDATE < '01-JAN-1985'
      AND (SAL BETWEEN 1600 AND 4999)
      AND ( comm NOT BETWEEN 501 AND 1400 ) )
```

Q.] WQTD all details of employee who are getting commision

→ select *
from emp
where comm >=0;

* Is operator: It is used to display null or NOT null values.
* Syntax: Column-name/Exp IS Null/Not Null

Ego. select *
from emp
where comm IS NOT NULL;

Q.] WQTD all details of employee who are having Manager

→ select *
from emp
where MGR IS NOT NULL; 13-rows.

Q.] WQTD all DOE who are not earning any comm

select *
from emp
where comm IS Null; (10 rows)

Q.]

(A) LIKE Operator.

It is used to find some pattern in table

Syntax

Column_name/Exp LIKE 'Pattern_to_Match';

NOTE: Percentile(%) :- It takes n no. of characters at n no. of times

Underscore (_) :- It takes Exactly one character at one time

Q.] WAQTD ADOE whose name starts with Charact A

```
Select *
from emp
where ENAME LIKE 'A%';
```

Q] WAQTD ADOE whose JOB ends with K.

```
Select *
from emp
where JOB LIKE '%K';
```

Q.] WAQTD ADOE whose Name is having 2nd character L

```
Select *
from emp
where Ename LIKE '_L%';
```

Q.] WAQTD ADOE whose JOB is having 'MAN' in it

```
Select *
from emp
where JOB LIKE '%MAN%';
```

500 1200 1250 5000

NOT NULL

PAGE No.
DATE / /

Q.] WAQTD ADOE whose name is having consecutive LL

```
Select *
from emp
where ENAME LIKE '%LL%' ;    '2 rows'
```

Q.] ——— whose name does ends with character R.

```
Select *
from emp
where ENAME NOT LIKE '%R' ;    (12 rows)
```

Q.] WAQTD ADOE who are having atleast one A in there name working as CLERK SALESMAN Analyst manager except those who are working dept (10,20) they are having MANAGER getting SAL ~~qual~~ >= 1250 SAL < 5000 getting comm. ~~less than equ~~ <= 500 comm > 1200 hired in year 81 in month of FEB OR DEC

Imp

```
Select *
from emp
where Ename LIKE '%A%'
    AND JOB IN ('CLERK', 'SALESMAN',
                'ANALYST', 'MANAGER')
AND DEPT NO IN (10, 20)
AND (JOB = 'MANAGER' AND (SAL BETWEEN
     MGR NOT NULL                1250 AND 4999))
AND (comm NOT BETWEEN 501 AND 1200
AND (HIREDATE > '01-FEB-81' AND
     HIREDATE < '28-FEB-81')
OR (HIREDATE > '01-DEC-81' AND
    HIREDATE < '31-DEC-81')
```

FUNCTION

It is a set of code or block of instruction which is used to perform specific task.

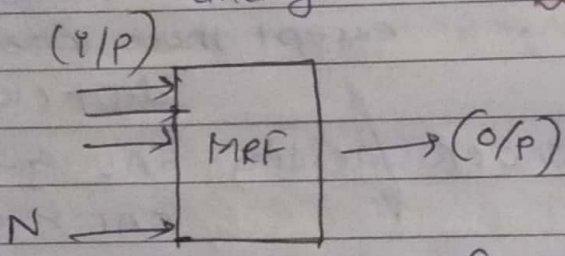
There are two types of functions:

Function

① User defined (PL-SQL) [Procedural Language SQL]

② built in
- ① single Row Function [SRF]
- ② Multi Row function [MRF]

★] Multi-Row function: It takes n no. of input and gives out only one output



There are five types of multirow function

① MAX :→

It is used to obtain max. value from table

Syntax: MAX (column_name / Exp);

② MIN :-

It is used to obtain MIN. value from table.

Syntax: MIN (column_name / Exp);

③ AVG :- It is used to obtain AVG valu from table.

Syntax :- AVG (column_name / Exp);

④ SUM :- It is used to obtain total values or addition of values from table.

Syntax: SUM (column-name / Exp);

⑤ COUNT :- It is used to obtain total no. of values from the table.

Syntax: COUNT (* / column-name / Exp);

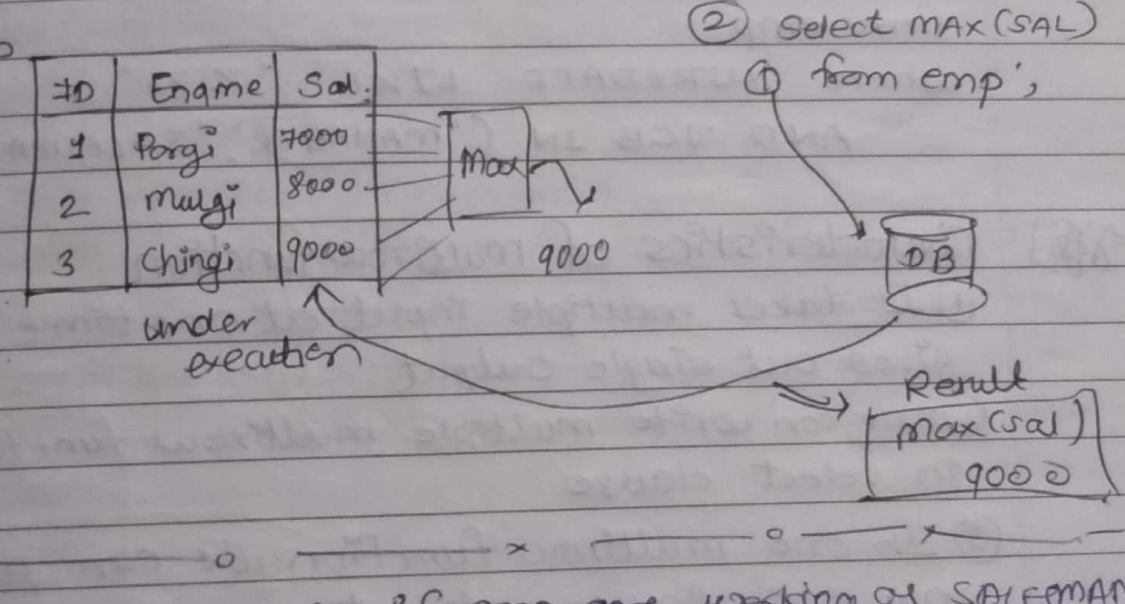
Q] WAQTD MAX SAL from table.

~~select *~~
~~from emp~~
~~where MAX (SAL);~~

Select MAX (SAL)
From emp; (1-row)

Emp

ID	Ename	Sal
1	Pargi	7000
2	Mulgi	8000
3	Chingi	9000



Q] WAQTD MAX SAL if emp are working as SALESMAN

```
select MAX(SAL)
from emp
wher JOB = 'SALESMAN';
```

Q] WAQTD MAX SAL MIN SAL if emp are working as manager, clerk in dept 10,20 hired before year 84

```
select MAX(SAL), MIN(SAL)
from emp
where JOB IN ('MANAGER', 'CLERK')
      AND DEPTNO IN (10,20)
      AND HIREDATE < '01JAN-84';
```

Q.] WAQTD MAX sal max comm avg sal avg comm from table if emp working in dept 20,30 not as a clerk, manager, analyst

```
select MAX(SAL), MAX(comm), AVG(SAL), AVG((comm
from emp
where DEPTNO IN (20,30)
      AND JOB NOT IN ('CLERK', 'MANAGER', 'ANALYST');
```

Q.] WAQTD min sal, total sal, if emp are hired in year 81 working as manager, salesman, clerk

```
select MIN(SAL), SUM(SAL)
from emp
where HIREDATE LIKE '%81'
      AND JOB IN ('MANAGER', 'SALESMAN', 'CLERK');
```

Q.] Characteristics of multirow function

1. It takes multiple input at one time and gives out single output
2. We can write multiple multirow function in select clause
3. In one multirow function we can pass only one Arguement at one time
4. Only in count multirow function we can pass '*' As an arguement
5. at the time of group by clause multirow function executes group by group.
6. multirow function ignores null values

Q] WAQTD MAX (SAL) in dept 10
```
Select MAX(SAL)
from emp
where DEPTNO = 10;
```

Q] WAQTD MAX (SAL) in dept 20
```
select MAX(SAL)
from emp
where DEPTNO = 20;
```

Q] WAQTD MAX (SAL) in dept 30
```
Select MAX (SAL)
from emp
where DEPTNO = 30;
```

Q] WAQTD MAX (SAL) in each dept.
```
Select MAX(SAL)
 from emp
GROUP BY DEPTNO;
```

Group-by Clause

It is use to create group of record

Syntax:- Select [Group-by-Expression]/
[Group-by-function]
```
FROM TABLENAME
[WHERE < FILTER_CONDITION> ]
GROUP BY COLUMN_NAME
```

* Characteristics of group-by clause

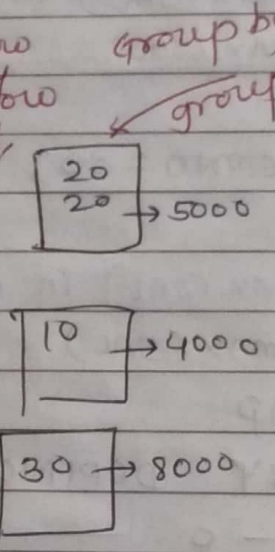
(1) It is use to create group of value

② It always execute row by row
③ where clause is optional for execution of group by ~~name~~ clause
④ we can write group by expression and

Q.] WAQTD max salary in each department if employees are earning salary more than 2975

Emp

Empno.	Ename	Sal	deptno
1	sursuri	700	10
2	chakri	3000	20
3	Pataka	4000	10
4	Paws	2000	20
5	Rocket	900	30
6	lavangi	8000	30
7	Chakli	5000	20
8	kerangi	100	10



(GBG)
④ select max(sal)
① from emp
② where sal) 2975
③ group by deptno
(RBR)

DB

Result

max (sal)
5000
4000
8000

Q.] WAQTD max. salary in each department if employment are not working as clerk

⟹
```
Select max (sal)
from emp
where JOB1 = 'CLERK'
Group by Deptno;
```

* Group by expression
Column name written in group by clause is known as group by expression in select clause

* Group by function
Multi row function used in select clause while using group by clause is known as group by function in select clause

Q] WAQTD max salary, min sal in each job
⇒ select max (sal), min(sal), JOB
from emp
Group by JOB;

Q] WAQTD max salary in each dept if emp are earning sal >= 1500 & sal < 5000 hired before year 87.
select max(sal), deptno
from emp
where (sal BETWEEN 1500 AND 4999)
AND (HIREDATE < '01-JAN-87');
Group by DeptNO;

Q] WAQTD total no. of emp in each job if employees are having character A in their name
select count(ename), JOB
from emp
where ename like '%A%'
Group by JOB;

PAGE No.
DATE / /

Having Clause :-

It is use to filter group condition or multi row function condition.

Syntax :

```
Select [Group-by-Expression] /
                [Group-by-Column]
From Tablename
[WHERE < FILTER_CONDITION >
GROUP BY COLUMN_NAME
HAVING < GROUP_FILTER_CONDITION >
```

PAGE No.
DATE / /

Q] WAQTD total no. of emp who are getting same sal.

④ select count(*), sal
① from emp
② Group by sal
③ having count(*) > 1;

★ ~~select count(*),~~ Row by Row

Emp	EmpNO	Ename	Sal
	1	~~Pooja~~	700
	2	~~Pooj~~	8000
	3	Mulga	700
	4	Mulgi	9000
	5	Chichi	9000

700
700

8000

9000
9000

Result

count *	Sal
2	700
2	9000

Q] WAQTD ~~TNOE~~ with their total Sal if emp are having MGR and they are getting sal < 5000 and they are having max sal >= 3000 and avg sal > 1500

```
Select count(*), sum(sal)
from emp
where MGR is not null AND SAL<5000
Group by JOB
having MAX(SAL)>=3000 AND AVG(SAL)>1500;
```

Where Clause	Having Clause
① It is used to ~~filter~~ Column conditions	① It is used to filter Group conditions or MRF condition
② It executes row by row	② It executes group by group
③ we can write multiple column conditions in where clause	③ we can write multiple MRF cond- in having clause
④ It executes after from clause	④ It executes after group by clause

⑤ Group by clause is not mandatory for execution of where clause.

⑤ Group by clause is mandatory for execution of Having clause.

ORDER BY ;

It is used to order record in ascending or descending manner

Syntax:

```
SELECT [GROUP_BY-EXPRESSION]/[GROUP_BY-FUNCTION]
FROM TABLENAME.
[WHERE < FILTER_CONDITION >]
[GROUP. BY (OLUMN_NAME]
[HAVING < GROUP_FILTER_CONDITION > ]
ORDER BY COLUMN_NAME/EXPRESSION/NUMBER/
                                [ALIAS NAME]
```

* Characteristics of ORDER BY Clause

① It is used to order records in ascending or descending order

② Order by clause is last executable statement

③ By default order by clause order the records in ascending order

④ We can write Alias Name in order by clause

⑤ we can pass number as an arguement in order by clause.

⑥ All the records are arranged in ascending order w.r.t primary key of the table

① e.g.
```
Select SAL
FROM EMP
ORDER BY SAL ASC;
```

e.g. ②
```
select sal*12 ANNUAL
from emp
order by annual ASC;
```

e.g ③
```
select *
from emp
order by 2 ASC;
```

case① Data is Unknown Subquery

steps ① find Allen Salary
② greater than
③ display the output

Emp

Empno	Ename	Sal
1	Porga	1500
2	Porgi	7000
3	Allen	1606
4	mulga	5000
5	mulgi	200

② outer
③ Select *
① from emp
② where Sal > (③ select sal
① from emp
② where ename = 'ALLEN');
sal > 1600
inner ①

Q.) WAQTD ADOE who are getting sal < SCOTT

```
Select *
from emp
where sal < (select sal
             from emp
             where Ename = 'SCOTT');
```

Q) WAQTD ADOE who are getting sal > WARD < SCOTT

```
Select *
from emp
where sal > (select sal
             from emp
             where Ename = 'WARD')
      AND sal < (select sal
                 from emp
                 where Ename = 'SCOTT');
```

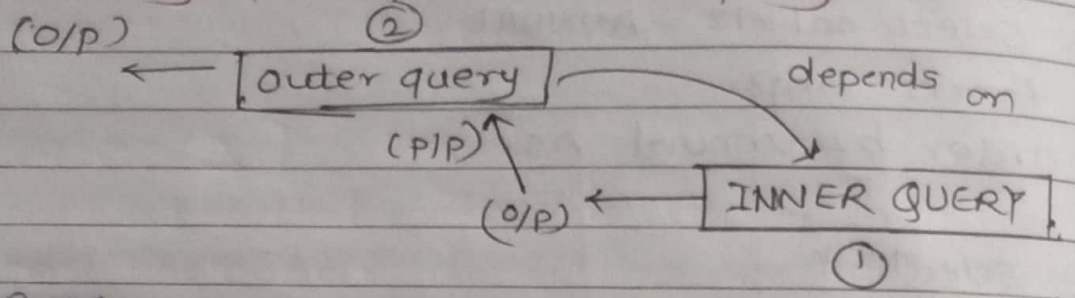
Q.) WQTD ADOE who are hired in year 81 getting sal > ADAMS

```
Select *
from emp
where HIREDATE Like '%81' AND
      Sal > (select sal
             from emp
             where ename = 'ADAMS');
```

(A) Subquery :-

Query written inside another query is called Subquery.

* Working principle of Subquery



① There are two query innerquery & outerquery.
② Innerquery will execute first. It will execute completely and gives output to the outer query.
③ Outer query will execute second it will execute completely and gives complete output of the query
④ From this we can say outer query is depend on inner query. This is known as subquery

Q.] WAQTD ADOE who are getting sal > 2975.

```
Select *
from emp
where sal > 2975;
```

Q.] WAQTD ADOE who are getting sal > Allen

```
Select * from emp
wher ~~from~~ SAL > ( select sal
                      from emp
                      where Ename = 'ALLEN' );
```

case④ Data is Unknown Subquery

steps ① find Allen Salary
② greater than
③ display the output

Emp

Empno	Ename	Sal
1	Porga	1500
2	Porgi	7000
3	Allen	1600
4	mulga	5000
5	mulgi	200

(2) outer
③ Select *
① from emp
② where Sal > (③ select sal
① from emp
② where ename = 'ALLEN');
sal > 1600
inner ①

Q.7 WAQTD ADOE who are getting sal < SCOTT

```
Select *
from emp
where sal < (select sal
             from emp
             where Ename = 'SCOTT');
```

Q) WAQTD ADOE who are getting sal > WARD < SCOTT

```
Select *
from emp
where sal > (select sal
             from emp
             where Ename = 'WARD')
      AND sal < (select sal
                 from emp
                 where Ename = 'SCOTT');
```

Q) WQTD ADOE who are hired in year 81 getting sal > ADAMS

```
Select *
from emp
where HIREDATE Like '%81' AND
      Sal > (select sal
             from emp
             where ename = 'ADAMS');
```

Q) WAQTD ADOE who are working at location Chikago

```
Select *
from emp
where deptno = (Select deptno
                from dept
                where LOC = 'CHICAGO');
```

Case:- 2]

When data to be display is present in one table but condition present in another table then we use subquery.

Q.] WAQTD ADOE who are working in dept Resech

```
Select *
from emp
where deptno = (select Deptno
                from dept
                where Dname = 'RESEARCH');
```

Q.] WAQ Dept details of KING

```
select *
from Dept
where deptno = (select deptno
                from emp
                where ename = 'KING');
```

Q] WAQTD ADOE who are working at loc newyork or delby

```
Select *
from emp
where deptno = (Select deptno
                from dept
                where LOC = 'NEWYork')
OR (Select deptno
    from dept
    where LOC = 'DELHI');
```

OR

```
select *
from emp
where deptno in (select deptno
                 from dept
                 where LOC IN ('NEWYORK','DALLAS'));
```

Types of Subquery

① Single Row Subquery

① When inner query gives single output to the outer query then it is known as single row subquery

② In single row subquery (= / !=) are used (Comparison operator)

eg.
```
select *
from emp
where deptno = (select deptno
                from dept
                where loc = 'CHICAGO');
```

outer query ②

single output

inner query ①

② Multi Row Subquery

① When inner query gives multiple output to the outer query then it is known as multi row subquery.

② In multi row subquery (in, not in) special operator are used.

eg.
```
select *
from emp
where deptno in (select deptno
                 from dept
                 where dname in ('ACCOUNTING',
                                 'RESEARCH'));
```

outer query ②

multiple output

inner query ①

Q] WAQTD MAX (SAL)
select MAX(SAL)
from emp;

Q] WAQTD First MAX (SAL)
Select MAX(SAL)
from emp;

Q.] WAQTD Second MAX(SAL)
Select MAX(SAL)
from emp
~~where sal(select *~~
~~from emp~~
where SAL < (select MAX(SAL)
from emp1);

Q.) WAQTD third minimum sal
Select MIN(SAL)
from emp
where SAL > (select MIN(SAL)
from emp
where SAL > (select MIN(SAL)
from emp));

Q.) WAQTD ADDE who are getting third max sal
Select *
from emp
where SAL = (SELECT MAX(SAL)
From emp
where sal < (Select MAX(SAL)
from emp
where SAL < (select MAX(SAL)
from emp)));

8] WAQTD dept details of emp who is getting 2nd max sal.

```
Select *
from dept
where deptno in (select deptno
                 from emp
                 where sal = (select max(sal)
                              from emp
                              where sal < (select max(SAL)
                                           from emp)));
```

9] WAQTD ADOE who are having MGR

```
Select *
from emp
where MGR is not null;
```

10] WAQTD Name of SMITHS manager

```
Select Ename
from emp
where ~~Ename~~ EmpNo = (select MGR
                         from emp
                         where Ename = 'SMITH');
```

11) WAQTD ADOf SCOTT MAnager.

```
select *
from emp
where EmpNo = (select MGR
               from emp
               where ename = 'SCOTT');
```

8] WAQTD ADO miller's MGR->MGR

```
Select *
from emp
where MGR = (Select empno mgr
             from emp
             where empno = (Select mgr
                            from emp
                            where Ename = 'MILLER'));
```

9.] WAQTD Dept details of martins MGR-MGR

```
Select *
from dept
where deptno = (Select deptno
                from emp
                where empno = (Select mgr
                               from emp
                               where empno =
                               (Select mgr
                                from emp
                                where ename = 'MARTIN')));
```

10.] WAQTD ADOE who are reporting to blake

```
Select *
from emp
where MGR IN (Select empno
              from emp
              where Ename = 'Blake');
```

8] WAQTD ~~the~~ total no of emp reporting to KING

```
select count(*)
from emp
where mgr in (select empno
              from emp
              where ename = 'KING');
```

9] WAQTD ADOE who are reporting to ADAMS MGR → MGR

```
Select *
from emp
where mgr in (select empno
              from emp
              wher Enapno = (select mgr
                             from emp
                             where empno =
                             (select mgr
                              from emp
                              where Ename = 'ADAMS')));
```

10]

Question Solve Employee Manager Relation

Q.1] WAQTD smiths Reporting manager's name

```
Select Ename
from emp
where mgr in (Select empno mgr
              from emp
              where Ename = 'SMITH');
```

FORD

Q.2] WAQTD ADAMS manager's manager name

```
Select Ename
from emp
where empno in (select mgr
                from emp
                where empno in (select mgr
                                from emp
                                where Ename = 'ADAM'));
```

JONES

Q.3] WAQTD Dname of Jones mgr.

```
Select Dname
from dept
where deptno in (select deptno
                 from emp
                 where empno in (Select mgr
                                 from emp
                                 where Ename = 'JONES'));
```

Accounting

Q.4] WAQTD millers managers salary

```
Select Sal
from emp
where empno in (Select mgr
                from emp
                where Ename = 'MILLER');
```

2450

Q.5] WAQTD LOC of smith manager's manager

```
Select LOC
from dept
where deptno in (Select deptno
                 from emp
                 where empno in (Select mgr
                                 from emp
                                 where empno in
                                 (Select mgr
                                  from emp
                                  where Ename = 'SMITH')));
```

DALLAS

Q.6] WAQTD Name of emp reporting to blake

```
Select Ename
from emp
where mgr in (select empno
              from emp
              where Ename = 'BLAKE');
```

(5)

Q.7] WAQTD number of emp reporting to King

```
Select Count(*)
from emp
where mgr in (select empno
              from emp
              where Ename = 'KING');
```

(2)

Q.8] WAQTD details of emp reporting to Jones

```
select *
from emp
where mgr in (select empno
              from emp
              where Ename = 'JONES');
```

(2)

Q9] WAQTD Enames of emp reporting to blakes manager

```
Select Ename
from emp
where mgr in (select empno
              from emp
              where mgr in (select empno
                            from emp
                            where Ename = 'BLAKE'));
```

Q.10] WAQTD Number of emp reporting to fordy manager

```
Select count(*)
from emp
where mgr in (select empno
              from emp
              where mgr in (select mgr
                            from emp
                            where Ename = 'FORD'));
```

OR

```
Select count(*)
from emp
where mgr in (Select mgr
              from emp
              where Ename = 'FORD');
```

Subquery Operators

① ALL operator

① It is multivalue operator which we used in subquery

② It has to be used with relational operator

③ It gives true when inner query returns multiple values and we want all these values to be true

Eg: WAQTD ADOE who are getting sal greater than all salesman.

```
Select *                          All condition needs to be true.
from emp                          (1600, 1250, 1250, 1500)
where sal > ALL (Select sal
                 from emp
                 where JOB = 'SALESMAN');
```

`>` ← Relational operater

`(Select sal from emp where JOB = 'SALESMAN')` ← inner query

② ANY Operator

① It is a multivalue operater which is used in subquery it has to be used with relational operator

② It returns true when any one condition needs to be true.

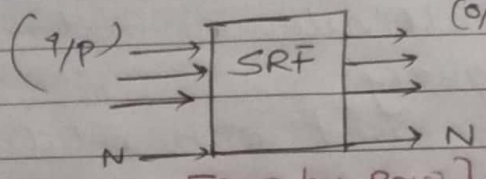
```
select *                          Any one cond^n needs to be true
from emp                          (1600, 1250, 1250, 1500)
where sal > ANY (Select sal
                 from emp
                 where JOB = 'SALESMAN');
```

`>` ← Relational Operater

`(Select sal from emp where JOB = 'SALESMAN')` ← inner query

Single Row function

SRF :- ① It takes n no. of input and gives out n no of output.



[Row by Row]

② Single row function executes Row by Row

③ We can write single row function in Select Clause as well as in where clause.

(1) LOWER :- ① It is used to convert string in lowercase

Syntax :- LOWER ('str')

Eg. Select LOWER ('ZAGAMAGA')
From Dual;
zagamaga

(2) UPPER :- ① It is used to convert string in Uppercase

Syntax :- UPPER ('str')

Eg. Select UPPER ('gspiders')
from Dual;
GSPIDERS

(3) INITCAP ; ① It is used to convert initial of string in uppercase

Syntax - INITCAP ('str');

Eg. select INITCAP ('abhishek');
from dual;
Abhishek

(4) REVERSE :- ① It is used to reverse the string

Syntax :- REVERSE ('str');

eg. Select Reverse ('sun Re')
from dual;

eR nuS

PAGE No.
DATE / /

(5) LENGTH :- It is used to get total character presents in a string.

Syntax - LENGTH ('str')

eg Select length ('MUMBAI')
from dual;

6

(6) WAQTD ADOE whose name consist 4 characters without like operator

Select *
from emp
where Length('Ename') = 4;

(7) MOD :- It is used to find remainder

Syntax :- MOD (V_1, V_2)

eg. Select mod (9,3)
from dual;

0

(7) SYSDATE :- It is used to display present date of system

eg Select SYSDATE
from dual;

07-NOV-22

(8) SYSTIMESTAMP :- It is used to display present date, time & timezone

Select SYSTIMESTAMP
from dual;

07-NOV-22 03.35.24.772000 PM +05:30

(9) NVL (Null Value logic) It is used to overcome the problem occured by null values

Syntax :- NVL (arg1, arg2);

arg1 - Column-name which consist null values

arg2 - Any value which is to be consider at the place of null.

(9) WAQTD Ename, job, sal, comm, sum(sal & comm) from table

```
select Ename, job, sal, comm, sal+comm
from emp;
```

eg.

```
Select Ename, job, sal, com, sal+ NVL(comm,0)
from emp;
```

(A) Concatenation (||) :- It is used to combine two or more strings

```
Select 'MY NAME IS' || ENAME || ' SALARY IS ' || SAL
From emp;
```

(*) Concat function :- It is used to combine two strings.

Syntax: CONCAT(str1, str2)

1) `Select Concat('@SP', 'JSP') from dual;` @SPJSP

2) `Select Concat('@SP', CONCAT('JSP', 'PYSP')) from dual;` @SPJSPPYSP

(10) Replace :- It is used to replace substring from original string with new string

syntax: REPLACE('original_str', 'sub_str', 'new_str');

1) REPLACE('BIRTHDAY', 'B', 'H'); ⇒ HIRTHDAY

2) REPLACE('BIRTHDAY', 'DAY', 'YEAR'); BIRTHYEAR

(11) SUBSTR :- It is used to obtain part of string from original string

SUBSTR('original_str', Position, [Length]);

iii) SUBSTR('MAHARASHTRA', 1, 4); MAHA
iv) SUBSTR('————', 2, 3); AHA
v) SUBSTR('————', -5, 2); SH
vi) SUBSTR('————', 6); ASHTRA
vii) Select SUBSTR('MAHARASHTRA', 11)
from dual; → A

viii) Seled SUBSTR('MAHARASHTRA', 12)
from dual; → (null)

(12) INSTR :- It is used to obtain index value of a string

Syntax:
INSTR('original str', 'sub-str', position, [nth occurance]);

MAMMAPAPPA

i) INSTR('————', 'M', 1, 2); 3
ii) INSTR('————', 'A', 1, 2); 5
iii) INSTR('————', 'm', 3, 1); 3
iv) INSTR('————', 'P', 1, 3); 9
v) INSTR('————', 'A', 10); 10

JOINS

(A) JOINS: Retrieval of data from multiple table simultaneously is known as joins.

* There are 5 types of JOINS
  1. Cartesian Join / Cross Join
  2. Inner Join / Equi Join
  3. Outer Join → (1) left outer join
     (2) Right outer join
     (3) full outer join
  4. Self Join
  5. Natural Join

Cartesian Join

1] ☆ Cross Join: (1) In cartesian join table (1) Record are merged with table (2) records.

Syntax:

ORACLE ⇒ Select column-names from T_1, T_2;

ANSI ⇒ Select column-names from T_1 CROSS JOIN T_2;

(American National Standard Institute)

Boys

BID	Bname	GID
1	Porg a	10
2	Mulga	20
3	Sundra	30

Girls

GID	Gname
10	Porgi
20	mulgi
30	Sundari

(A) Select *
from Boys, Girls;

BID	Bname	GID	GID	Gname
1	Porga	10	10	Porgi
1	Porga	10	20	Mulgi
1	Porga	10	30	Sundri
2	Mulga	20	10	Porgi
2	Mulga	20	20	Mulgi
2	Mulga	20	30	Sundri
3	Sundra	30	10	Porgi
3	Sundra	30	20	Mulgi
3	Sundra	30	30	Sundri

2) (A) Inner Join :- It is used to get match records from both the table.

Syntax:

ORACLE ⇒ Select Col-names
from T_1, T_2
where <Join-condition>;

ANSI ⇒ Select Col-names
from T, INNER JOIN T2
ON <JOIN-condition>;

NOTE:- Join condition is used to get all match records from both table.

<T_1.Column_name = T_2.column_name>

Select *
from Boys, Girls
where Boys.Gid = Girls.Gid';

Boys

BID	Bname	GID
1	Porga	10
2	mulga	20
3	chingta	30

Girls

GID	Gname
10	Porgi
20	Mulgi
30	Chingi

Q.] WAQTD Ename, deptname from Table emp dept
Select Ename, dname
from emp, dept
(14) where emp.deptno = dept.deptno;

Q.] WAQTD Ename, deptname, LOC if emp are working in dept accounting and Research
select Ename, dname, LOC
(8) from emp, dept
where emp.deptno = dept.deptno AND
Dname in ('ACCOUNTING', 'RESEARCH');

Q.] WAQTD Ename, Job, sal, dname if emp sal < 1250 sal > 2975 working at LOC CHICAGO, DALLAS
select Ename, Job, sal, dname
(5) from emp, dept
where emp.deptno = dept.deptno AND
sal Not between 1250 and 2975
AND LOC in ('CHICAGO', 'DALLAS');

empname, dname, sal, hiredate
Q.) if manager, president, analyst, in dept accounting Research getting sal ≥ 1250 sal ≤ 5000 and they are having letter 'A' Dname and they are having mgr.
(21) select Ename, dname, sal, hiredate
from emp, dept
where emp.deptno = dept.deptno AND
JOB in ('MANAGER', 'PRESIDENT', 'ANALYST')
AND Dname in ('ACCOUNTING', 'RESEARCH')
AND sal between 1250 and 5000
AND Dname like '%A%'
AND mgr is not null;

Outer Join

① It is used to get unmatched records

① **Left Outer Join :** It is used to get match records along with unmatch records from left table

ORACLE

```
select column_names
    from T1, T2
where T1.column_name = T2.column_name(+);
```

ANSI

```
Select column_names
from T1 LEFT OUTER JOIN T2
ON <Join-condition>;
```

Boys Left

BID	Bname	GID
1	Porga	10
2	mulga	20
3	Guddu	30
4	chingu	

Girls Right

GID	Gname
10	Porgi
20	Mulgi
30	Guddi

```
Slect *
from Boys, Girls
where Boys.Gid = Girls.Gid (+);
```

Result :

BID	Bname	Gid	Gid	Gname
1	Porga	10	10	Porgi
2	mulga	20	20	Mulgi
3	Guddu	30	30	Guddi
4	chingu	NULL	NULL	NULL

Unmatched record from → left table

② (A) Right Outer Join: It is used to get all match records along with unmatch records of right Table.

ORACLE → Select column-names
from T_1, T_2
where T_1.column-name(+) = T_2 column-name;

ANSI →
Select_column.names
from T_1 RIGHT OUTER JOIN T_2
ON <JOIN-condition>;

Boys

BID	Bname	Gid
1	Porga	10
2	Guddu	20
3	Pinku	30

Girls

Gid	Gname
10	Pargi
20	Guddi
30	Pinky
40	chingi

Select *
from Boys, Girls
where Boys.Gid(+) = Girls.Gid;

Result

BID	Bname	Gid	Gid	Gname
1	Porga	10	10	Pargi
2	Guddu	20	20	Guddi
3	Pinku	30	30	Pinky
Null	Null	Null	40	chingi

Unmatched records from Right table

FULL OUTER JOIN : It is used to get all matched records alongwith unmatch records from both the table.

ANSI

```
Select Column-name
from T1 Full outer join T2
ON <Join-condition>;
```

No oracle Syntax.

Boys

BID	Bname	Gid
1	Porga	10
2	Mulga	20
3	Sundra	30
4	Gudda	

Girls

Gid	Gname
10	Pargi
20	Mulgi
30	Sundri
40	Pinky

BID	Bname	Gid	Gid	Gname.	
1	Porga	10	10	Pargi	All matched records from both tables
2	Mulga	20	20	Mulgi	
3	Sundra	30	30	Sundri	
4	Gudda	null	null	null	→ unmatched record from left table
null	null	null	40	Pinky	

```
Select *
from boys full outer join girls
On Boys.Gid = Girls.Gid;
```

```
Select *
from Emp Full outer join Dept
On Emp.deptno = dept.deptno;
```

PAGE NO.
DATE / /

(4) WAQTD ADOE who are having mgr

```
Select *
from emp
where mgr is not null;
```

5) WAQTD dept details of miller.

```
Select *
from dept
where deptno in (Select deptno
                 from emp
                 where Ename = 'miller');
```

6) WAQTD manager name of miller.

```
Select Ename
from emp
where empno in (Select mgr
                from emp
                where Ename = 'MILLER');
```

7) WAQTD Ename, mgr name of miller.

Self Join:

① When data to be displayed is present in same table but different rows we use self join.

② Joining table by itself is known as self Join.

ORACLE

```
Select Column_names
from Tablename1 T1, Tablename2 T2
where <Join_condition>
```

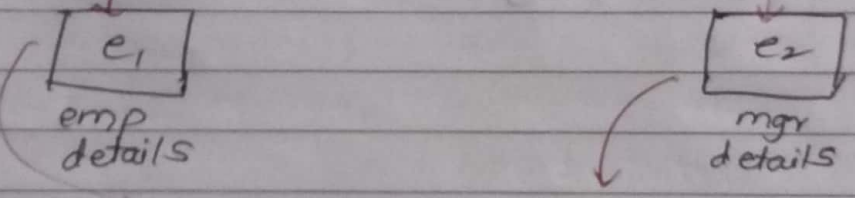
ANSI

```
select Column-names
from Tablename1 T1 Join Tablename2 T2
ON <Join-condition>;
```

Ename, manager name of miller.

emp

e1.mgr = e2.empno




```
Select e1.ename , e2.ename
from   emp e1 , emp e2
where  e1.mgr = e2.empno  AND

       e1.ename = 'MILLER';
```

Q] WAQTD ename, job, mgrname, mgr job from table

```
Select e1.ename, e1.job, e2.ename, e2.job
from emp e1, emp e2
where e1.mgr = e2.empno ;
      e1.mgr = e2.empno
```

e1 — emp details

e2 — mgr details

Q] WAQTD e.ename, e.sal, mgr name, mgr sal if emp are working dept 10, 20 and mgr are Hired in year 81

```
Select e1.ename, e1.sal, e2.ename, e2.sal
from emp e1, emp e2
where e1.mgr = e2.empno AND
      ~~e1.job~~ in e1.deptno in (10,20) AND
      e2.hiredate like '%81' ;
```

Q] WAQTD ename, e job, ~~hir~~ e hiredate, m name, m job if employees are working as clerk, Salesman and mgr getting sal > 2000

```
Select e1.ename, e1.job, e1.hiredate, e2.ename
                                      e2.job
from emp e1, emp e2
where e1.mgr = e2.empno AND
      e1.job in ('CLERK', 'SALESMAN') AND
      e2.sal > 2000 ;
```

Q] WAQTD e name e job e sal m name m sal if emp & mgr are getting same sal

Q] WAQTD e name mgr name mgr mgrs name e job and mgr job

Q] WAQTD e name e dept name mgr name mgr deptname

Q] WAQTD e name e job if emp is having character 'E' in their name and mgr are hired before year 85

(1)
```
Select e1.ename, e1.job, e1.sal, e2.ename,
       e2.sal
from emp e1, emp e2
where e1.mgr = e2.empno AND
      e1.sal = e2.sal ;
```

(2)
```
Select e1.ename, e2.ename, e3.ename
       e1.job, e2.job
from emp e1, emp e2, emp e3
where e1.mgr = e2.empno AND
      e2.mgr = e3.empno ;
```

PAGE NO.
DATE / /

③ select e1.ename, e3.dname, e2.ename,
e2.dname
from emp e1, emp e2, dept e3
where e1.mgr = e2.empno AND
e2.deptno = e3.deptno;

④ Select e1.ename, e1.job
from emp e1, emp e2
where e1.mgr = e2.empno AND
⑥ e2.hiredate < '01-JAN-85'; AND
e1.ename like '%E%';

⑪ select e1.ename, e1.job, e1.sal, e2.ename,
e2.sal
from emp e1 join emp e2
~~set~~ on e1.mgr = e2.empno
where e1.sal = e2.sal;

⑫ select e1.ename, e2.ename, e3.ename,
e1.job, e2.job
from emp e1 join emp e2 join emp e3
ON e1.mgr = e2.empno AND
e2.mgr = e3.empno;

PAGE No.
DATE / /

Natural JOIN:

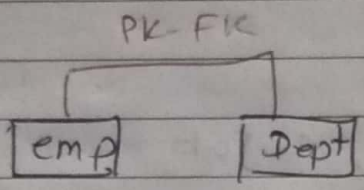
It is used when we dont know table structure.

NO ORACLE SYNTAX

ANSI ⇒

```
select column-name
from T1 NATURAL JOIN T2;
```

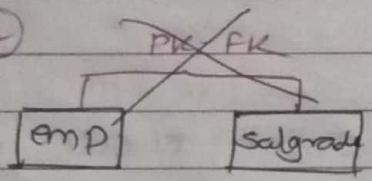
Case 1




```
Select *
from emp natural join dept;
```

All the matched records (inner join)

Case 2




```
Select *
from emp natural join salegrade
```

All the error records (Cartesian join)

Theory: Case 1

1. When two tables are having primary key foreign key relationship after applying natural join we will get all matched records as same as inner join.

PAGE No.
DATE / /

(11) Theory Case (2)

① If two tables are not having primary key foreign key relationship after applying natural join we get all error records as Cartesian Join

(12) Pseudo Columns :

① This are hidden columns to display pseudo column we have to call them explicitly
② This are the false columns
③ There are two pseudo column present in every table.
① RowId ② RowNum.

(13) RowID :-

① It is unique
② It is not null
③ Combination of unique and not null is known as primary but we cannot consider RowID as primary key because it is generated at the time of insertion.
④ RowID consist 18 characters.

(14) RowNum :-

① It is unique
② It is not null
③ Combination of unique and not null is known as primary key but we cannot consider RowNum as primary key

because it is generated at the time of execution

④ RowNum are nothing but serial num which always starts from 1.

Q.] WAQTD top 5 records from the table.

```
Select *
from emp
where Rownum <= 5 ;
```

Q.] WAQTD top 9 records from table

```
Select *
from emp
where Rownum <= 9 ;
```

Q.] WAQTD first record from table

```
Select *
from emp
where Rownum = 1 ;
```

Q.] WAQTD second record from table

```
Select *
from emp
where rownum = 2 ;
```

```
Select *
from (select RowNum slno, Emp.*
      from emp)
where slno = 2 ;
```

This inner query written to change the name of rownum by using alias name.

NOTE:- once rownum condition is rejected it will not move on to next record.

Emp

Slno (RowNum)	Empno	Ename	Sal
1	111	Kalpya	700
2	112	Anu	900
3	113	Prachi	1000
4	114	Paushni	1200
5	115	Aadika	5000
6	116	Ravi	100

Result

Slno	Empno	Ename	Sal
2	112	Anu	900

Q) WAQTD 3, 5, 7, 9 records from table.

```
select *
from (select RowNum slno, Emp.*
      from emp)
where slno in (3,5,7,9);
```

Q) WAQTD third record from bottom.

```
select *
from ( select RowNum slno, Emp.*
       from emp
       order By empno Desc)
where slno = 3;
```

Q) WAQ: 3rd max sal using subquery.

```
Select *
from ( select RowNum slno, Emp.*
       from emp
       order By sal Desc)
where slno = 3;
```

map

```
Select max(sal)
 from emp
where sal < (select max(sal)
             from emp where sal < (select max(sal)
                                   from emp));
```

Ans

STEPS :-

1. remove duplicates by using distinct
2. Order by desc (to find nth max)
3. Assign the rownum
4. display the output.

```
Select sal
from (select rownum slno, sal
      from (select distinct sal
            from emp
            order by sal desc))
where slno = 3;
```

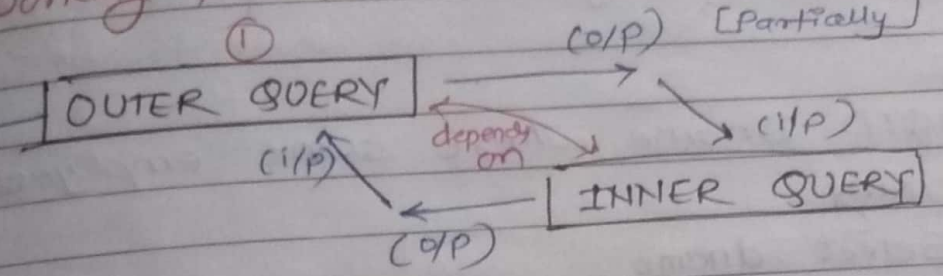
Q.] WAQTD 1st 2nd 3rd min sal using precedence

```
select *
from (select rownum slno, sal
      from (select distinc sal
            from emp
            order by sal))
where slno in (1, 2, 3);
```

Correlated Subquery

Outer Query depend on inner query and inner query is depend on outer query then it is known as Correlated subquery.

working principle of correlated subquery



① There are two query's outer query and inner query.
② In correlated subquery outer query will execute first. It will execute partially.
③ That partial output will be given to inner query as an input.
④ Inner query will execute second it will execute completely
⑤ Output of inner query is will be given to outer query
⑥ Outer query will execute once again this time it will execute completely.
⑦ From this we can say outer query is depend on inner query and viceversa.
⑧ Outer query and inner query are interdependent i.e, depends on each other.

PAGE No.
DATE / /

Condition to write correlated Subquery

① Query should be in Subquery format
② In where clause of inner query there should be one join condition
③ One of the join condition should belong to outer Query

Q] WAQTD Dname where some employees are working.

Outer Query ①
③ Select dname
① from dept (10, 20, 30)
② where deptno in (Select deptno
10 = 10
20 = 20
30 = 30
① from emp
② where emp.deptno = dept.deptno);
10 = 10
20 = 20
30 = 30
40 X
inner query ②

OPERATORS

① Exists : ① It is unary operator which has to be used with correlated subquery
② It gives true if there are some values present at RHS

```
Select Dname
from dept
where Exists (select Deptno
              from emp
              where emp.deptno = dept.deptno);
```

Accounting
Research
Sales

PAGE No.
DATE / /

Not Exists

① It is a unary operator which has to be used with correlated subquery
② It gives true if inner query returns no value or null
③ It gives true if there is no value present in R.H.S.

```
Select Dname
from dept
where Not exists (select deptno
                  from emp
                  where emp.deptno = dept.deptno);
```

output: operation

Q) WAQTD 3rd max sal. using subquery and pseudo column.

```
Select max(sal)          ---- subquery
 from emp
where sal < (select max(sal)
             from emp
             where sal < (select max(sal)
                          from emp));
```

pseudo column.

```
Select sal
from (select rownum slno, sal
      from (select distinct sal
            from emp
            Order by sal desc))
where slno = 3;
```

If 3rd max = 2
2nd max. = 1

less than max

(★) nth max by using correlated subquery

3rd max

Sal
500
800
900
200
800

E1

Sal
500
800
900
200
800

E2

outer ①
Select E1.Sal
from emp e1 (0,1,1,2,3)
where (Select count (distinct E2.Sal)
from emp e2
where E1.sal < E2.Sal) = 2

inner ②

Subquery
Join condition.

500 < 500 ×
< 800 ✓
② < 900 ✓
< 200 ×
< 800 ✓

800 < 500 ×
< 800 ×
① < 900 ✓
< 200 ×
< 800 ×

900 < 500 ×
< 800 ×
0 < 900 ×
< 200 ×
< 800 ×

200 < 500 ✓
< 800 ✓
③ < 900 ✓
< 200 ×
< 800 ✓

800 < 500 ×
< 800 ×
① < 900 ✓
< 200 ×
< 800 ×

SQL STATEMENTS

PAGE NO
DATE / /

I] DATA DEFINATION LANGUAGE

① CREATE:

SYNTAX: CREATE TABLE table-name
(COLUMN-NAME1 DATATYPE NOT NULL/ NULL,
COLUMN-NAME2 DATATYPE NOT NULL/ NULL,
.
.
COLUMN_NAMEn DATATYPE NOT NULL/ NULL,
CONSTRAINT constraint_ref_name UNIQUE(COLUMN_NAME),
CONSTRAINT constraint_ref_name CHECK(CONDITION),
CONSTRAINT constraint_ref_name PRIMARY KEY (COLUMN_NAME),
CONSTRAINT constraint_ref_name FOREIGN KEY(COLUMN_NAME).
REFERENCES parent_table_name (COLUMN_NAME)
);

② RENAME :-

SYNTAX:
RENAME current_table_name TO New_name;

③ ALTER :-

SYNTAX:

① TO ADD A COLUMN
ALTER TABLE table_name
ADD COLUMN_NAME DATATYPE [NULL/NOT NULL];

② TO DROP A COLUMN.
ALTER TABLE table_name
DROP COLUMN COLUMN_NAME;

③ TO CHANGE THE DATATYPE
ALTER TABLE table_name
MODIFY COLUMN_NAME new_datatype;

④ TO CHANGE THE NOT NULL CONSTRAINT

```
ALTER TABLE table_name
MODIFY COLUMN_NAME existing-datatype NULL / NOT NULL;
```

⑤ TO RENAME THE COLUMN

```
ALTER TABLE table_name
RENAME COLUMN current_name TO new_name;
```

⑥ TO MODIFY CONSTRAINTS

a) ALTER TABLE table_name
ADD CONSTRAINT constraint_ref_name UNIQUE(column_name

b) ALTER TABLE table_name
ADD CONSTRAINT constraint_ref_name CHECK (condition);

c) ALTER TABLE table_name
ADD CONSTRAINT constraint_ref_name PRIMARY KEY (column_name);

d) ALTER TABLE table_name
ADD CONSTRAINT constraint_ref_name FOREIGN KEY (column_name
REFERENCES parent_table_name (column_name);

⑦ TO DROP / DISABLE / ENABLE A CONSTRAINT;

```
ALTER TABLE table_name
DROP / DISABLE / ENABLE CONSTRAINT constraint_ref_name;
```

⑭ TRUNCATE;

SYNTAX: TRUNCATE TABLE table_name;

⑮ DROP;

SYNTAX: DROP TABLE table_name;

TO RECOVER THE TABLE: (only in ORACLE)

SYNTAX FLASHBACK TABLE table-name
TO BEFORE DROP ;
[RENAME TO new-name]

TO DROP THE TABLE FROM RECYCLE BIN

SYNTAX PURGE TABLE table-name;

II] DATA MANIPULATION LANGUAGE

① INSERT

SYNTAX ① INSERT INTO table-name VALUES (V1, V2, .., Vn);

② INSERT INTO table-name (COL1, COL2, ... COLn)
VALUES (V1, V2, ..., Vn);

← OR →

INSERT INTO table-name (COL1, COL2, ... COLn)
VALUES (& COL1, & COL2, ... & COLn);

③ INSERT INTO table-name
SELECT statement;

② UPDATE

SYNTAX: UPDATE table-name
SET COL1 = V1, COL2 = V2, ... COLn = Vn
[WHERE <filter-condition>];

③ DELETE

SYNTAX: DELETE
FROM table-name
[WHERE <filter-condition>];

PAGE No.
DATE / /

III] TRANSACTION CONTROL LANGUAGE

★★ (1) COMMIT ;-

SYNTAX : COMMIT ;

★★ (2) SAVEPOINT :

SYNTAX :. SAVEPOINT savepoint-name ;

★★ (3) ROLLBACK ;

SYNTAX :. ROLLBACK ;

⟹ ROLLBACK TO SAVEPOINT

SYNTAX → ROLLBACK TO savepoint-name ;

IV] DATA CONTROL LANGUAGE :

★★ (1) GRANT :

SYNTAX :. GRANT sql-statement ON table-name
TO user-name ;

★★ (2) REVOKE ;

SYNTAX :- REVOKE sql-statement ON table-name
FROM user-name ;

Table Planning
Data Definition Language (DDL)

1. Tablename
2. no. of columns
3. DataType
4. Constraint

1. CREATE → Shopping

	Number(3)	Varchar(20)	Number(8,2)	Varchar(20)	Number(4)	Varchar(20)
	PID	Pname	Price	Location	Quantity	Company
PK	Unique + NN	NN	NN	NN	NN	NN
			check (price > 0)		check (Quantity)	

Constraint clause name: PID+PK | Price-check | Quntity-check

```
CREATE TABLE SHOPPING
( PID NUMBER(3) NOT NULL,
  PNAME VARCHAR(20) NOT NULL,
  PRICE NUMBER(8,2) NOT NULL,
  LOCATION VARCHAR(20) NOT NULL,
  QUANTITY NUMBER(4) NOT NULL,
  COMPANY VARCHAR(20) NOT NULL,
  CONSTRAINT PID_PK PRIMARY KEY(PID),
  CONSTRAINT PRICE_CHEK CHECK(PRICE>0),
  CONSTRAINT QUANTITY_CHECK CHECK(QUANTITY>0)
  );
```

2. RENAME

```
RENAME SHOPPING TO QMART;
```

③ ALTER

① To add a column

```
ALTER TABLE SMART
ADD XYZ VARCHAR(10) NOT NULL;
```

DESC TABLE-NAME;

This command is used to display column names before inserting records.

② To drop a column

```
ALTER TABLE SMART
DROP COLUMN XYZ;
```

③ To change the datatype

```
ALTER TABLE SMART
MODIFY LOCATION CHAR(10);
```

④ To change the not null constraint

```
ALTER TABLE SMART
MODIFY COMPANY VARCHAR(20) NULL;
```

⑤ To Rename the column

```
ALTER TABLE SMART
RENAME COLUMN COMPANY TO BRAND
```

QMART

PID	Pname	Price	location	Quantity	Brand
1	shoes	80000	Pune	1	Nike
2	Biscuit	5	Dholakpur	2	Parleg
3	watch	10000	Mumbai	1	Sonata
4	Soap	10	Dmart	5	lifebuoy
5	whiskey	3700	Alkabar	10	Red label
6	noodles	14	BKC	50	maggi
7	chakli	20	Thane	30	Rambandhu.

① INSERT INTO QMART VALUES (1, 'SHOES', 80000, 'PUNE', 1, 'NIKE');

② INSERT INTO QMART ~~VALUES~~ (PID, PNAME, PRICE, LOCATION, QUANTITY, BRAND)
VALUES (3, 'WATCH', 1000, 'MUMBAI', 1, 'SONATA');

③ INSERT INTO QMART (PID, PNAME, PRICE, LOCATION, QUANTITY, BRAND)
VALUES (&PID, &PNAME, &PRICE, &LOCATION, &QUANTITY, &BRAND);

Insert from one table to another

INSERT INTO target_table_name (col1, col2, ... coln)
Select col, col2 .... coln
from current_table_name
where <condition>;

INSERT INTO BONUS (ENAME, JOB, SAL, COMM)
SELECT ENAME, JOB, SAL, COMM
FROM EMP
WHERE ENAME = 'SMITH';

COMMIT
to save the table;

UPDATE:

```
UPDATE GMART
SET QUANTITY = 10
WHERE PID IN (1,2,3,4);
```

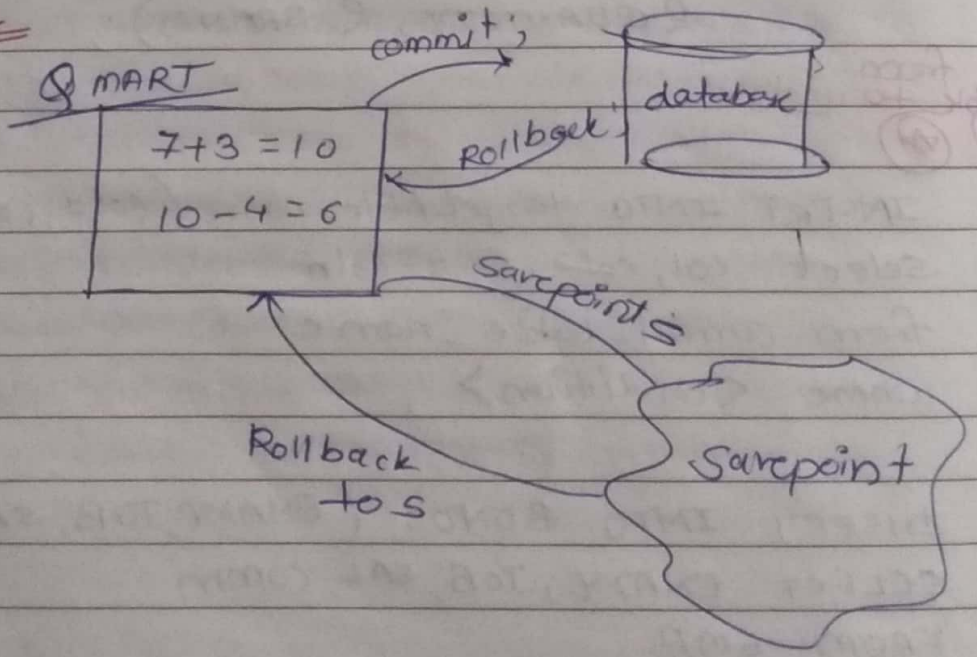
DELETE:

```
Delete
from GMART
where PID IN (4,6,1);
```

ROLL BACK

```
ROLLBACK;
```

TCL



PAGE NO.
DATE / /

⑧ SAVEPOINT S;

④ DROP
⑤ DROP TABLE SMART;

⑫ SHOW RECYCLEBIN;
This command is used to display Recycle Bin

⑥ Flashback
Flashback Table SMART TO BEFORE DROP;

⑦ TRUNCATE
TRUNCATE TABLE SMART;

DELETE	DROP	TRUNCATE
① It is DML command	① It is DDL command	① It is DDL command
② It is used to delete particular records from table	② It is used to DROP table. Table will be stored inside recycle bin	② It is used to truncate table. All records will be removed permanently without altering its table structure
③ we can get back deleted records	③ we can get back dropped table.	③ we cannot get back truncated records.
④ Roll back is used	④ Flashback is used	—

(A) GRANT

STEP (1) Username : HR
Password : TIGER.

STEP (2) select *
from Tab;

STEP (3) select *
from EMP;
(Here HR will get error HR dont have permission to access employee table)
To give permission we have to connect to SCOTT database

STEP (4) CONNECT;
Username : SCOTT
password : TIGER

STEP (5) GRANT SELECT ON EMP TO HR;

STEP (6) CONNECT;
username : HR
password : TIGER

STEP (7) select *
from SCOTT.EMP;

REVOKE: ① connect;
username: HR
password: TIGER

STEP ② REVOKE SELECT ON EMP FROM HR;

STEP ③ CONNECT;
Enter username: HR
password: TIGER

STEP ④ select *
from SCOTT.EMP;

(A) WAQTD Details of emp if the empname starts with B

(1)
```
Select *
from emp
where INSTR('ENAME', 1, 1) = 'B';
      SUBSTR
```

(2) WAQTD details of emp if the empname starts with B (without using Like OP)

```
Select *
from emp
where INSTR(ENAME, 'B', 1, 1) = 1;
```

(3) WAQTD JOB IN REVERSE Format if the job ends with man.

```
Select Reverse(JOB)
 from emp
where JOB in (select JOB
                from emp
               where SUBSTR(JOB, -3, 3) = 'MAN');
```

(4) WAQTD Emp name in lowercase if name starts with K.

```
Select Lower(Ename)
 from emp
where Ename in (select Ename
                  from emp
                 where SUBSTR(Ename, 1, 1) = 'K');
```

(5) WAQTD No. of emp who are getting 3 digit Comm or name having 6 characters

```
select count(*)
from emp
where length(comm) = 3 OR
   length(Ename) = 6;
```

⑥ JSPIDERS - QSPIDERS

```
Select Replace('JSPIDERS', 'J', 'Q')
from dual;
```

PAGE No:
DATE / /

Attribute:-

Property of an Object is called Attribute.

① **Key Attribute:-** To uniquely identify records we select some attributes those attributes are called as key attributes
e.g. SID, mail id, ph.no.

② **Non-key attribute:-** All the attribute except key attribute are called as non-key attribute.
eg Sname, age, address, Branch etc.

③ **Prime Key Attribute:-** Among all the key attribute one attribute one attribute is selected to represent the table is called as prime-key Attribute.
e.g. SID (Primary key)

④ **Non-prime key Attribute:-** All attribute except prime key attribute are called as non-prime key attribute
e.g. mail id, phno.

⑤ **Composite key Attribute:-** Combination of two or more non-key attribute are called composite key Attribute
e.g. Sname, age, add.

⑥ **Super key Attribute:-** Set of all key attributes are called as Super key Attribute.

e.g {SID, mail.id, ph-no}

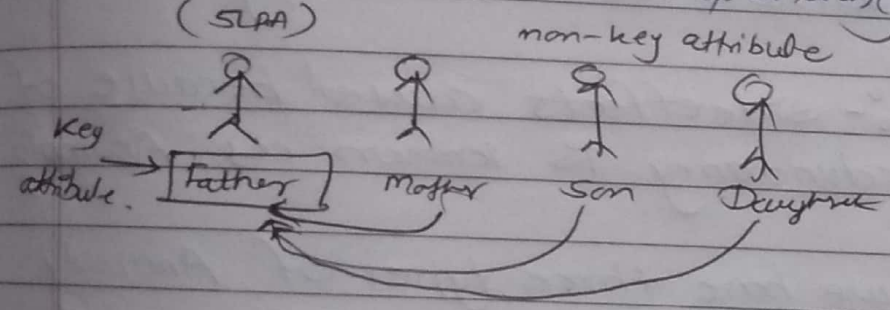
(7) foriegn Key Attribute? : Attribute which helps to connect two tables is known as foriegn key Attribute.

e.g. emp → deptno
dept → deptno

Dependency :-

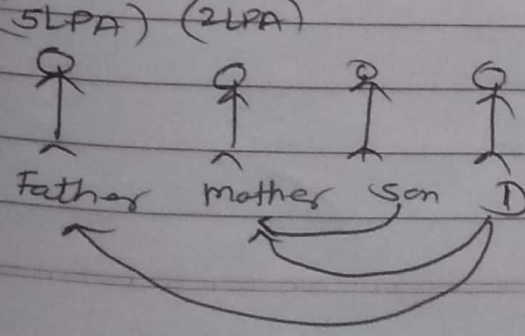
1) Total Functional Dependency :

When all the attributes are depend on one key attribute then it is called as Total functional dependency.



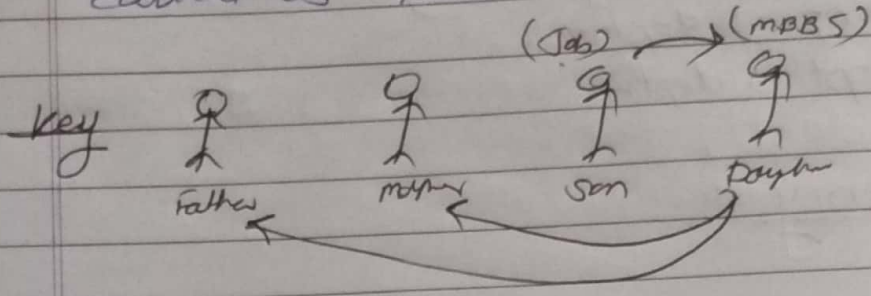
2) Partial Functional Dependency :

① It consist composite key attributes ② when non-key attribute is depend on another attribute which is part of composite key attribute is known as Partial functional dependency



3) (A) Transitive functional Dependency

When non-key attribute is depend on another non-key attribute which is depend on key attribute then it is called as Transitive Functional Dependency.



(A2) Redundancy

Repeatation of unwanted data is known as Redundancy.

Anamoly :- Sideeffects caused because of redundancy is known as Anamoly

(*) In that we have three types of Anamoly

1. Insertion Anamoly
2. Updation Anamoly
3. Deletion Anamoly