CREATED BY AKKI

15-09-2022
THURSDAY

SQL

(STRUCTURED QUERY LANGUAGE)

❖ **DATA:-** Data is raw-fact which describes attributes of an entity.

❖ **DATABASE:-** It is a place or container used to store the data in systematic and organised manner.
- To retrieve data easily and in the fastest way.

❖ **DATABASE OPERATION:-** It is a operation used to achieve database.

- CREATE
- READ
- UPDATE
- DELETE

-Database operation is also known as (CRUD) operation.

❖ **DBMS**:-(DATABASE MANAGEMENT SYSTEM)
- It is a software which is used to maintain and manage the database.
- To communicate with database, we use “Query Language”.
- DBMS also provide security and authorization.

❖ **TYPES OF DBMS**:-
1. OBJECT ORIENTED
2. RDBMS(RELATIONAL DATABASE MANAGEMENT SYSTEM)
3. NETWORK
4. HIERARICAL

❖ **RDBMS(RELATIONAL DATABASE MANAGEMENT SYSTEM):-**

- It is a software which is used to store and manage the data in the form of table.

❖ CHARACTERISTICS OF RDBMS:-

1. If DBMS follows relational (table),then it is known as **RDBMS.**
2. If DBMS follows EF CODD rule,then it is known as **RDBMS.**
3. To communicate with RDBMS we use **(SQL).**

❖ RELATIONAL MODEL:-

1. Relational model is given by EF CODD.
2. In relational model, data is stored in the form of tables.
3. In relational model, table is stored in the form of meta-data.

❖ EF CODD RULE:-

1. Data entered into the cell should be single value data or atomic data.
2. We can store the data in multiple tables and establish connection through their key attributes.
3. To validate the data entered into cell,we use datatype and constraints.
4. Datatype is mandatory,Constraints are of optional.

19-09-2022
MONDAY

❖ **DATATYPES:-**

- Datatype is a type of data or kind of data which is used to store in memory.

❖ **TYPES OF DATATYPES:-**

1. **CHAR**
2. **VARCHAR**
3. **LARGE OBJECT:-**
   - CHAR LARGE OBJECT
   - BINARY LARGE OBJECT
4. **DATE**
5. **NUMBER**

1) CHAR:-

- **SYNTAX:- CHAR(_)**←length
- Maximum Size is 2000
- If input point and final point are same then it is known as **FIXED LENGTH MEMORY ALLOCATION.**
- Ex. Phone no,Aadhar no,PAN no,etc.

2) VARCHAR:-

- **SYNTAX:- VARCHAR(_)**←length
- Maximum Size is 2000
- If input point and final point are different then it is known as **VARIABLE LENGTH MEMORY ALLOCATION.**
- Ex. Name,Address,Designation,etc.
- **In VARCHAR2, maximum size is 4000.**

20-09-2022
TUESDAY

3) LARGE OBJECT:-

1. CHAR LARGE OBJECT:-
   - **SYNTAX:-CLOB;**
   - Its maximum size is **4GB**.

2. BINARY LARGE OBJECT:-
   - **SYNTAX:-BLOB;**
   - **It is a datatype,used to store images,videos,audios,documents,etc. in database after converting into binary values, through java methods or functions.**
   - Its maximum size is **4GB(**In oracle database**)**
   - For **MySQL,** maximum size is **12-14GB.**
   - For **MsSQL,** maximum size is **12-14GB.**

4) DATE:-
   - **SYNTAX:- DATE ;**
   - **IN ORACLE FORMAT,**
   - **'DD-MON-YYYY'(FOR FUTURE OR PAST DATE)**
   - **Ex.** 24-JAN-2001
   - **'DD-MON-YY'(FOR PRESENT DATE)**
   - **Ex.** 20-SEP-22

5) NUMBER :-
   - **SYNTAX:- NUMBER(P[,SCALE]);**
   - Where, P:-precision(1-38 capacity) & Scale:- (0-127 capacity)
   - Ex. 1. NUMBER(7,2) => 99999.99

     2.NUMBER(6,3)=>999.999

22-09-2022
THURSDAY

❖ CONSTRAINTS :-

- It is a set of conditions or rules used to validate the data entered into the cell.

❖ TYPES OF CONSTRAINTS :-

1. **UNIQUE**
2. **NOT NULL**
3. **CHECK**
4. **PRIMARY KEY**
5. **FOREIGN KEY**

1. UNIQUE :-

- It is a constraint used to avoid duplicate data entered into cell.
- **SYNTAX**:- **UNIQUE(_);**

2. NOT NULL:-

- NOT NULL is a constraint which represents something in a cell.
- **SYNTAX:- NOTNULL;**
- **NULL :-** NULL is a keyword which represents nothing in cell.

3. CHECK :-

- It is a extra condition used to validate the data entered into cell.
- **SYNTAX :- CHECK(CONDITION);**
- **EX.** CHECK(FEE>=35000 AND FEE<=40000);

4. PRIMARY KEY :-

- It is a key used to uniquely identify a row or a column in a table.

❖ **CHARACTERISTICS OF PRIMARY KEY:-**

I. It should be unique.
II. It should ne not null.
III. It should be combination of unique and not null.
IV. In a table, there should be only one primary key.
V. Primary key represents the table.
VI. Primary key is not mandatory, designwise it is preferable.

5. FOREIGN KEY :-

- It is a key used to make connection from one table to another table.

❖ **CHARACTERISTICS OF FOREIGN KEY:-**

I. It could be unique.
II. It could be not null.
III. It is not combination of unique and not null.

- Primary key is present in parent table.
- Foreign key is present in child table.
- Primary key of one table should be foreign key of other table.
- Foreign key also known as '**referential integrity constraint**'.

❖ **SQL STATEMENTS:-**

1. **DATA DEFINATION LANGUAGE (DDL)**
   - It helps regarding creation of the table.
     a) Create
     b) Rename
     c) Alter ➔editing
     d) Truncate
     e) Drop

     used to delete (table)

2. **DATA MANIPULATION LANGUAGE (DML)**
   - Insertion of data.
     a) Insert
     b) Update
     c) Delete ➔ about data

3. **TRANSACTION CONTROL LANGUAGE(TCL)**
   - Saving data.
     a) Commit
     b) Roll back
     c) Save point

     Save data

4. **DATA CONTROL LANGUAGE (DCL)**
   - permission to user.
     a) Grant
     b) Revoke

5. **DATA QUERY LANGUAGE (DQL)**
   - To communicate with the table.
     a) Select
     b) Projection
     c) Selection
     d) Joins

❖ DATA QUERY LANGUAGE (DQL):-

1. SELECT :-
   - Retrival of data from the table which is present in database.

2. PROJECTION:-
   - Retrival of data from the table which present in execution area by providing column name.

3. SELECTION :-
   - Retrival of data from the table by providing both column name and row name.

4. JOINS :-
   - Retrival of data from multiple tables simultaneously.

❖ SELECT :-

- From clause :-
- **SYNTAX :- FROM TABLE_NAME ;**

❖ PROJECTION :-

- Select clause :-
- **SYNTAX :- SELECT */[DISTINCT] COL_NAME/EXPRESSION [ALIAS NAME]**

FUNCTIONALITY OF FROM CLAUSE:-

- Goes to database, search for the table name and put it into execution area.

FUNCTIONALITY OF SELECT CLAUSE :-

- Goes to the table which is under execution area, search for column name and display the output.

27-09-2022

TUESDAY

- **Assignment :-Difference betn unique & distinct.**

Set page 100

Set line 100

- **cl scr**:- clear screen;
- ❖ **astring(*)**:- It is a **wild card character** used to display all the details of table.
- ❖ **Expression** :- by considering 2 or more operant and using some operation we can get some output it is known as Expression.
- ❖ **Distinct**:- It is keyword used to display unique value at the time of execution.
- ❖ **Alias name**:- It is a alternate name given to column or expression at the time of execution.

The ways to gave alias name :-

- sal*12:- annual
- sal*12:-"annual sal"
- Ename:-Employee_name
- Sal*6:-As mid_term_sal

1. Write query to display all the details of dept table.
   - ➢ select * from dept;

2. Write query to display all details of salgrade table.
   - ➢ select * from salgrade;

3. Write query to display all the details of bonus table.
   - ➢ output is(no rows selected)

4. WAQTD all the table name present in database.
   - ➢ select*from tab;

5. WAQTD ename,sal,annual sal which is from emp table.
- select ename,sal,sal*12 from emp;

6. WAQTD ename ,monthly sal,mid term sal,quarter sal,1000rs bonus from emp table.
- select ename,sal,sal*12, sal*6,sal*3, sal+1000 from emp;

7. WAQTD ename,sal,job,annual sal,mid term sal, 1000 bonus for mid term sal,1000 deduction for monthly sal .
- select ename,sal,job,sal*12,sal*6,sal*6+1000,sal-1000 from emp;

8. WAQTD ename,sal,annual sal,mid term sal,10% increment for monthly sal.
- select ename,sal,sal*12,sal*6,sal+sal*10/100 from emp;

9. WAQTD ename,sal,20% increment for monthly sal,10% decrement for monthly comm.
- select ename,sal,sal+sal*20/100,comm-comm*10/100 from emp;

10. WAQTD ename,sal,job,30% incr. for monthly sal,35% decrement for monthly comm.
- select ename,sal,job,sal+sal*30/100,comm-comm*35/100 from emp;

11. WAQTD ename,sal,annual sal,10% incr for mid term sal.
- select ename,sal,sal*12,sal*6+sal*6*10/100 from emp;

12. WAQTD ename,sal,mid term sal,25% incr for quarter sal,10% decr for quarter comm.
- select ename,sal,sal*6,sal*3+sal*3*25/100, comm*3-comm*3*10/100 from emp;

13. WAQTD ename,sal which is from emp table.
- select ename,sal from emp;

14. WAQTD unique sal fromemp table.
- select distinct(sal) from emp;

15. WAQTD unique job from emp table.
- select distinct(job) from emp;

16. WAQTD unique dept no from emp.
- select distinct(deptno) from emp;

17. WAQTD ename,sal,job,annual sal,10% incr for mid term sal.
- select ename empname,sal,job,sal*12 “annual sal”,sal*6+sal*6*10/100 from emp;

29-09-2022

THURSDAY

❖ **SELECTION**:-

Retrival of data from the table by providing both column name and row name.

❖ **Where clause**:-

Syntax:- where col_name operator value;

- It is a clause used to filter the condition row by row.

❖ **CHARACTERISTICS OF WHERE CLAUSE**:-

- Where clause used to filter the condition.
- Where clause executes row by row.
- Where clause executes after execution of **from clause.**
- In Where clause, we can write multiple conditions.
- In Where clause, we can't write multi-row function.

❖ **OPERATORS:-**

I. ARITHMETIC OPERATOR(+,-,*,/)
II. RELATIONAL OPERATOR(>,<,=,!=)
III. COMPARISON OPERATOR(>,<,>=,<=)
IV. CONCATINATION(||)
V. SPECIAL OPERATOR:-
   1) In,Not in
   2) Between,Not between
   3) Is,Is not
   4) Like,Not like

VI. SUB-QUERY OPERATOR:-
   - All,Any
   - Exists,Not Exists

VII. LOGICAL OPERATOR:-
   1) AND(*):-

C1	C2	OUTPUT
1	1	1
1	0	0
0	1	0
0	0	0

2) OR(+):-

C1	C2	OUTPUT
1	1	1
1	0	1
0	1	1
0	0	0

18. WAQTD ename,sal,job,if ename is smith.
- select ename,sal,job from emp where ename='SMITH';

19. WAQTD ename,sal,if they are getting sal more than 1000.
- Select ename,sal from emp where sal>1000;

20. WAQTD ename,sal,job if they are working as manager.
- Select ename,sal,job from emp where job='MANAGER';(no rows selected)

21. WAQTD all details of emps if they are working in deptno 10.
- Select*from emp where deptno=10;

22. WAQTD ename,sal,hiredate if they are hired after 1981.
- select ename,sal,hiredate from emp where hiredate> '31-DEC-1981';

23. WAQTD ename,sal,job if they are getting sal more than or equal to 3000.
- select ename,sal,job from emp where sal>=3000;

24. WAQTD empno,ename,sal if empno is 7839.
- select empno,ename,sal from emp where EMPNO=7839;

25. WAQTD all details of emp if they are working as manager in deptno 10.
- select*from emp where job='MANAGER' AND deptno=10;

26. WAQTD ename,job,sal if they working as salesman and getting sal more than 3000.

- select ename,job,sal from emp where job='SALESMAN' AND sal> 3000;

27. WAQTD ENAME, job,empno if they are working as analyst or if they are working as deptno 10.

- select ename,job,deptno from emp where job='ANALYST'OR DEPTNO=10;

30-09-2022

FRIDAY

❖ **IN OPERATOR**:-

- **It is a multi value operator which takes single value in LHS and multiple value in RHS**

SYNTAX:- Col_name IN (v1,v2,v3,……Vn);

Equal operator:- It is single operator which takes single value in LHS and single value in RHS

❖ **NOT IN OPERATOR**:-

- **It is a multivalue operator which takes single value in LHS and multiple value in RHS**

SYNTAX:- Col_name NOT IN (v1,v2,v3,……Vn);

❖ **IS OPERATOR**:-

- **It is a operator used to check null/not null values**

SYNTAX:- Col_name IS NULL/NOT NULL;

❖ **LIKE OPERATOR:-**

- **It is a operator used to check pattern matching**

SYNTAX:- Col_name LIKE 'PATTERN';

❖ **WIDCARD CHARACTER:**-

% :- ANY CHAR

- Any number of time char

_:-

- Any char
- Only one time

1. WAQTD ename sal,deptno,job if they are getting sal more than 2000 in dept no 10 or working as manager.

- select ename,sal,deptno,job
  from emp
  where (sal>2000 AND deptno=10) OR job='MANAGER';

2. WAQTD all details of emp if they are working in dept no 10,20,30.

- select* from emp
  where deptno=10 OR DEPTNO=20 OR DEPTNO=30;
- select* from emp
  where deptno IN (10,20,30);

3. WAQTD ENAME,SAL,if emps are getting sal of 950,1250,1500,3000& 5000 in deptno 10,20,30.

- select ename,sal
  from emp
  where sal IN (950,1250,1500,3000,5000)AND deptno IN(10,20,30);

4. WAQTD ename,job,if they are working as manager,analyst,salesman.

- select ename,job
  from emp
  where job IN ('MANAGER',’ANALYST’,’SALESMAN');

5. WAQTD ename deptno, if emp is not working in deptno 10,20.

- select ename,deptno
  from emp
  where DEPTNO NOT IN (10,20);

6. WAQTD ename,sal,comm,if emps are getting sal more than 2000 & getting some comm.

- select ename,sal,comm
  from emp
  where sal>2000 AND COMM IS not null;

7. WAQTD ename,job ,comm,if emps are working in some job and not getting any comm.

- select ename,job,comm
  from emp
  where job IS NOT NULL AND COMM IS NULL;

8. WAQTD ename,job,comm,deptno,if they are getting sal more than 2000 and getting some comm and working in some dept and hired after 1980.

- select ename,job,comm,deptno
  from emp
  where sal>2000 AND COMM IS NOT NULL AND DEPTNO IS NOT NULL AND HIREDATE>'31-DEC-1980';(no rows selected)

9. WAQTD ename,job,if name starts with A.

- select ename,job
  from emp
  where ename LIKE 'A%';

03-10-2022

MONDAY

- **ASSIGNMENT:-Difference between SQL AND MySQL.**

NO.	SQL	MySQL
1)	SQL is a language to manage database.	MySQL is a database software.
2)	SQL is used to query databases.	MySQL stores the data.
3)	SQL is structured query language.	MySQL is RDBMS (Relational database management system).
4)	SQL commands remains the same.	MySQL software is updated regularly.
5)	SQL commands are used in Oracle, SQL server, DB2, MySQL, etc.	MySQL uses SQL.

1. WAQTD Ename,sal,if name is having at least one'A'.
   - Where ename LIKE '%A%'

2. WAQTD ENAME,job, sal if emp name is having exactly 5 char.
   - Where ename LIKE'_____';

3. WAQTD emp getting sal only 3 digit
   - Where sal LIKE '___';

4. WAQTD ename if name starts with C.
   - Where ename LIKE 'C%';

5. WAQTD if emps getting exactly 4 digits of sal.
   - Where sal LIKE'____';

6. WAQTD ENAME,sal , job if ename starting 2nd char is M.
   - Where ename LIKE '_M%';

7. WAQTD ename,job if the job last before 1 char(2nd last)should be ‘A’.
   - Where job LIKE’%A_’;

8. WAQTD ename,job if job starting 3rd char should be ‘I’.
   - Where job LIKE ‘__I%’;

9. WAQTD ename,hiredate if emps got hire in moth of ‘JAN’.
   - WHERE hiredate LIKE ‘__-JAN-__’;

10. WAQTD ename,hiredate if emps hired in year of 1980.
    - Where hiredate LIKE’%-%-80’;

11. WAQTD they hired on 20th date.
    - Where hiredate LIKE ’20-%-%’;

04-10-2022

TUESDAY

- **% , _** :- **<u>ESCAPE(!)</u>** char used to convert wild card char into ORDINARY char.

    - It is a char used to convert wild card char into SPECIAL char.
    - Refer ex. 3.

1. WAQTD ename , sal, if starting 2nd char should be 'M' in name.

```
Select ename,sal from emp
Where ename LIKE '_M%';
```

2. WAQTD ename,sal if name starts with '$'

```
Select ename,sal from emp
Where ename LIKE '$%';
```

3. WAQTD ename,sal if name start with '%'

```
Select ename,sal from emp
Where ename LIKE '! %%' ESCAPE '!';
```

4. WAQTD ename,sal if name ends with '%'

```
Select ename,sal from emp
Where ename LIKE '%!%' ESCAPE '!';
```

5. WAQTD ename ,job if job 2nd char starting is '_'

```
Select ename,job from emp
Where job LIKE '_ ! _ %' ESCAPE '!';
```

6. WAQTD ename,job if the job last 2nd char is '%'

```
Select ename,job from emp
Where job LIKE '%! % _ ' ESCAPE '!';
```

❖ **BEETWEEN, NOT BETWEEN OPERATOR :-**

- It is operator used to check the ranges from lower value towards higher value.
- **SYNTAX :- col_name BETWEEN X AND Y;**
- **FORMULA :-** >=X AND Y<=

7. WAQTD ename,sal if emps are getting sal more than or equal to 800 and less than or equal to 3000

➢ Select ename,sal from emp
Where SAL BETWEEN 800 AND 3000;

8. WAQTD ename,sal,if they getting sal more than or equal to 1250 and less than or equal to 5000.

➢ Select ename,sal from emp
Where SAL BETWEEN 1250 AND 5000;

9. WAQTD ename,sal,if they getting sal more than 1250 and less than or equal to 5000.

➢ Select ename,sal from emp
Where SAL BETWEEN 1251 AND 5000;

10. WAQTD ename,sal,if they getting sal more than 1500 and less than 5000.

➢ Select ename,sal from emp
Where SAL BETWEEN 1501 AND 4999;

06-10-2022

THURSDAY

1. WAQTD ename,sal if they are getting sal >800 and <=3000
   - SELECT ENAME,SAL FROM EMP
     WHERE SAL BETWEEN 801 AND 3000;

2. WAQTD ename,sal if they are getting sal >800 and <3000
   - SELECT ENAME,SAL FROM EMP
     WHERE SAL BETWEEN 801 AND 2999;

3. WAQTD ename,sal, hiredate if they are hired in year of 1981
   - SELECT ENAME,SAL,HIREDATE FROM EMP
     WHERE HIREDATE BETWEEN '01-JAN-81' AND '31-DEC-81';

4. WAQTD ename,sal if they are getting sal less than 800 and more than 3000
   - SELECT ENAME,SAL FROM EMP
     WHERE SAL NOT BETWEEN 800 AND 3000;

❖ **NOT BETWEEN OPERATOR** :-
   - **SYNTAX**:- COL_NAME NOT BETWEEN X AND Y;
   - **FORMULA** :- <X AND >Y

5. WAQTD ename,sal,comm if they are getting comm less than 300 and more than 1000
   - SELECT ENAME,SAL,COMM FROM EMP
     WHERE COMM NOT BETWEEN 300 AND 1000;

6. WAQTD ename,sal,comm if they are getting comm less than 300 and greater than 3000
   - SELECT ENAME,SAL,COMM FROM EMP
     WHERE COMM NOT BETWEEN 301 AND 2999;

7. WAQTD ename,sal,comm if they are getting sal greater than 1500 and less than 4000
   - SELECT ENAME,SAL,COMM FROM EMP
     WHERE SAL BETWEEN 1500 AND 3000;

8. WAQTD highest sal from emp
   - SELECT MAX(SAL) FROM EMP;

07-10-2022

FRIDAY

❖ **FUNCTIONS** :-

- It is set of instructions or block of code used to give some output.

❖ **TYPES OF FUNCTIONS** :-

1. Built-in function
2. User defined function

1. BUILT-IN FUNCTIONS:-

- Predefined function

2. USER DEFINED FUNCTIONS :-

- Developer function

❖ **TYPES OF BUILD IN FUNCTION :-**

1) SINGLE ROW FUNCTION :-
   - Which takes 'n' number of input and generates 'n' number of output.
   - It executes row by row.

2) MULTI ROW FUNCTION :-
   - Which takes 'n' number input and generates only **'one'** output.
   - It executes group by group.
   - It also known as **'Aggregate function'**.

❖ TYPES OF MULTI ROW FUNCTION:-

1. MAX()
2. MIN()
3. SUM()
4. AVG()
5. COUNT()

08-10-2022
SATURDAY

- ✓ **IN WHERE CLAUSE ,WE CAN'T USE MULTI ROW FUNCTION.**
- ✓ **IN SELECT CLAUSE, WE CAN'T USE BOTH MULTI ROW AND SINGLE NORMAL COLUMN SIMULTANEOUSLY.**

1. WAQTD max,min,avg of sal.
   - SELECT MAX(SAL),MIN(SAL),AVG(SAL) FROM EMP;

2. WAQTD max, min,avg,sal and comm if they are getting some comm.
   - SELECT
     MAX(SAL),MAX(COMM),MIN(SAL),MIN(COMM),AVG(SAL),AVG(COMM)
     FROM EMP
     WHERE COMM IS NOT NULL;

3. WAQTD max,min,sum,avg of sal's if the name starts with 'A'.
   - SELECT MAX(SAL),MIN(SAL),AVG(SAL) FROM EMP
     WHERE ENAME LIKE 'A%';

4. WAQTD no. of emps, where max sal ,min sal if getting sal>2000.
   - SELECT MAX(SAL),MIN(SAL),AVG(SAL) FROM EMP
     WHERE SAL>2000;

5. WAQTD max sal, min sal if they are working in deptno 10,20
   - SELECT MAX(SAL),MIN(SAL) FROM EMP
     WHERE DEPTNO IN (10,20);

6. WAQTD max sal,min sal,if they are getting max sal more than 2000
   - **ERROR.(IN WHERE CLAUSE ,WE CAN'T USE MULTI ROW FUNCTION.)**

7. WAQTD max sal,min sal,deptno if they are working in deptno. 10,20,30
   - **ERROR.(not a single-group group function)**

8. WAQTD max sal,min sal, max comm, min comm if they are working in deptno 10,20,30
   - SELECT MAX(SAL),MIN(SAL),MAX(COMM),MIN(COMM)FROM EMP
     WHERE DEPTNO IN(10,20,30);

9. WAQTD max sal,min sal if they are working in deptno 10,20,30
   - SELECT MAX(SAL),MIN(SAL) FROM EMP
     WHERE DEPTNO IN(10,20,30);

10. WAQTD max sal,min sal in each dept.
    - **Next concept.** SELECT MAX(SAL),MIN(SAL) FROM EMP
      GROUP BY DEPTNO;

❖ **CHARACTERISTICS OF MULTI ROW FUNCTION :-**

1. We can't write multi row function in where clause.
   - Multi row function executes group by group.

2. Along with multi row function it is not possible to write normal column.
   - Need to write only multi row function column.

3. Multi row function is not grouping the null values.
4. We can't nest multi row function.

10-10-2022
MONDAY

❖ **GROUP BY CLAUSE :-**

- Used to create a group or group the records.
- It executes row by row.
- **SYNTAX:- GROUP BY COL_NAME/EXPRESSION;**
- Goes to the table which is under execution area ,search for column name and make a group in row by row manner.

❖ **CHARACTERISTICS OF GROUP BY CLAUSE :-**

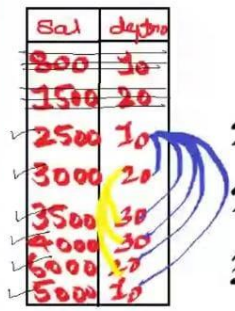
I. Group by clause executes row by row.
II. Group by clause executes after FROM clause but if there is condition it executes after WHERE clause.
III. Group by clause can take multiple column.
IV. The written column in group by clause is known as "GROUP BY EXPRESSION".
V. GROUP BY EXPRESSION is mandatory to write in SELECT clause.
VI. After group by clause, Every clauses execute group by group. (SELECT CLAUSE)

1. WAQTD max sal,min sal in each dept. if they are getting sal more than 2000.

➢ SELECT MAX(SAL),MIN(SAL) FROM EMP
WHERE SAL>2000
GROUP BY DEPTNO;

➢ SELECT MAX(SAL),MIN(SAL),DEPTNO FROM EMP
WHERE SAL>2000
GROUP BY DEPTNO;**(To show deptno. Along with output)**

1.WAQTD MAXSAL,MIN SAL IN EACH DEPT IF THEY ARE GETTING SAL MORE THAN 2000.



4 Select Max(sal), Min(sal), deptno — GBG, GBG, GBG, GBG

1 From Emp

2 Where Sal > 2000 → RBR

3 Group by deptno → RBR

10

2500	10
5000	10

20

3000	20
6000	20

30

3500	30
4000	30

O/P

Max	Min	deptno
5000	2500	10
6000	3000	20
4000	3500	30

11-10-2022
TUESDAY

1. WAQTD all details of emps if they are working in deptno 10,20 and getting some sal between 2000 to 3000 and job ends with man

```
SELECT * FROM EMP
WHERE DEPTNO IN(10,20) AND SAL IS NOT NULL AND SAL BETWEEN 2000 AND 3000 AND JOB LIKE '%MAN';
```

2. WAQTD ename,sal if they are getting sal more than 2000 and less than or equal to 5000

```
SELECT ENAME,SAL FROM EMP
WHERE SAL BETWEEN 2001 AND 5000;
```

3. WAQTD ename,sal,job if they are getting sal less than 300 and more than 1000

```
SELECT ENAME,SAL,JOB FROM EMP
WHERE SAL NOT BETWEEN 300 AND 1000;
```

4. WAQTD ename,sal if they are getting sal less than or equal to 300 and more than 1000

```
SELECT ENAME,SAL,JOB FROM EMP
WHERE SAL NOT BETWEEN 301 AND 1000;
```

5. WAQTD ename,comm if they are getting comm less than or equal to 0 and greater than or equal to 500

```
SELECT ENAME,COMM FROM EMP
WHERE COMM NOT BETWEEN 1 AND 499;
```

6. WAQTD ename,job if the job starts with ‘$’

```
SELECT ENAME,JOB FROM EMP
WHERE JOB LIKE '$%';
```

7. WAQTD ename,job if ename starts with ‘%’

```
SELECT ENAME,JOB FROM EMP
WHERE ENAME LIKE '!%%' ESCAPE'!';
```

8. WAQTD ename,job if the job ends with '_'

```
SELECT ENAME,JOB FROM EMP
WHERE JOB LIKE '%!_' ESCAPE'!';
```

9. WAQTD ename,job if ename starting 2nd char is '_'

```
SELECT ENAME,JOB FROM EMP
WHERE ENAME LIKE '_!_%' ESCAPE'!';
```

10. WAQTD max sal,min sal in each job

```
SELECT MAX(SAL),MIN(SAL),JOB FROM EMP
GROUP BY JOB;
```

11. WAQTD max sal,min sal,ename if ename starts with 'A'

```
SELECT MAX(SAL),MIN(SAL),ENAME FROM EMP
 WHERE ENAME LIKE 'A%'
 GROUP BY ENAME;
```

12-10-2022
WEDNESDAY

❖ **HAVING CLAUSE:-**

- It is clause used to filter the groups.
- **SYNTAX:- HAVING <MRF CONDITION/GROUPED COL_>**

❖ **CHARACTERISTICS OF HAVING CLAUSE:-**

I. It executes group by group.
II. It executes after execution of FROM clause but if there is a condition and group by it will execute after it.
III. In having clause, we can take multiple condition.

❖ **PATH OF EXECUTION :-**

FROM → WHERE → GROUP BY → HAVING → SELECT

❖ **WHERE AND HAVING DIFFERENCE**

1. WAQTD max sal,job which is from emp table.

➢ SELECT MAX(SAL),JOB FROM EMP
GROUP BY JOB;

2. WAQTD max sal,min sal in each dept if they are getting sal more than 1000 and max sal more than 2000

➢ SELECT MAX(SAL),MIN(SAL),DEPTNO FROM EMP
WHERE SAL>1000
GROUP BY DEPTNO
HAVING MAX(SAL)>2000;

3. WAQTD max sal,min sal if max sal is more than 3000

➢ SELECT MAX(SAL),MIN(SAL) FROM EMP
HAVING MAX(SAL)>3000;

4. WAQTD ename,max sal if avg sal is less than 2000

➢ SELECT ENAME,MAX(SAL) FROM EMP
GROUP BY ENAME
HAVING AVG(SAL)<2000

5. WAQTD unique sal without using distinct keyword

- **SELECT SAL FROM EMP**
  GROUP BY SAL;

6. WAQTD no. of emp in each dept

- SELECT COUNT(*),DEPTNO FROM EMP
  GROUP BY DEPTNO;

7. WAQTD NO. OF Emp in each dept if dept is having atleast 4 emps in it

- SELECT COUNT(*),DEPTNO FROM EMP
  GROUP BY DEPTNO
  HAVING COUNT(*)>=4;

8. WAQTD duplicate sal.

- SELECT SAL FROM EMP
  GROUP BY SAL
  HAVING COUNT(SAL)>1;

9. WAQTD repeated job

- SELECT JOB FROM EMP
  GROUP BY JOB
  HAVING COUNT(JOB)>1;

10. WAQTD repeated hiredate.

- SELECT HIREDATE FROM EMP
  GROUP BY HIREDATE
  HAVING COUNT(HIREDATE)>1;

11. WAQTD same sal in same deptno.

- SELECT SAL,DEPTNO FROM EMP
  GROUP BY SAL,DEPTNO
  HAVING COUNT(SAL)>1 AND COUNT(DEPTNO)>1;

12. WAQTD same hiredate on same deptno

- SELECT HIREDATE,DEPTNO FROM EMP
  GROUP BY HIREDATE,DEPTNO
  HAVING COUNT(HIREDATE)>1 AND COUNT(DEPTNO)>1;

❖ **ORDER BY CLAUSE :-**

- It is clause used to **sort** the values, either descending or ascending order.
- **SYNTAX:- ORDER BY COL_NAME/EXPRESSION/MRF/ALIAS NAME ASC/DESC;**

❖ **CHARACTERISTICS OF ORDER BY CLAUSE :-**

1) It executes after execution of SELECT clause.
2) It executes after execute for arranging values
3) In it we can take MRF.
4) Order by clause by default executes in ascending order.

13. WAQTD sal if they are getting sal more than 2000 and arrange the output in decending order

```
SELECT SAL FROM EMP
WHERE SAL>2000
ORDER BY SAL DESC;
```

14. WAQTD annual sal if emps are getting some sal arrange output in ascending order

```
SELECT SAL*12 FROM EMP
WHERE SAL IS NOT NULL
ORDER BY SAL*12 ASC;
```

15. WAQTD max sal,min sal in each dept if they are getting some sal and order the max sal in decending order

```
SELECT MAX(SAL),MIN(SAL),DEPTNO FROM EMP
WHERE SAL IS NOT NULL
GROUP BY DEPTNO
ORDER BY MAX(SAL) DESC;
```

14-10-2022
FRIDAY

❖ **SUB QUERY:-**

- Query inside another query is known as **Sub query**.
- Kinds of sub query: 1) Outer query 2) Inner query

❖ **FUNCTIONALITY OF SUB QUERY :-**

- Inner query is used (to find unknown value) get partial output and gives input to Outer query and it generates complete output(To display output).
- Outer query depends on the inner query.

1. WAQTD ename in decending order.

➢ SELECT ENAME FROM EMP
ORDER BY ename DESC;

2. WAQTD ename,sal arrange ename in decending and sal in ascending.

➢ SELECT ENAME,SAL FROM EMP
ORDER BY ename DESC,SAL ASC;(NO OUTPUT)

3. WAQTD ename,sal if they are getting sal more than 2000'

➢ SELECT ENAME,SAL FROM EMP
WHERE SAL>2000;

4. WAQTD ename ,sal if they are getting sal more than SMITH sal.

➢ SELECT ENAME,SAL FROM EMP
WHERE SAL> (SELECT SAL FROM EMP
WHERE ENAME='SMITH');

5. WAQTD ename,sal,job if they are getting sal more than MILLER sal.

➢ SELECT ENAME,SAL,JOB FROM EMP
WHERE SAL>(SELECT SAL FROM EMP
WHERE ENAME='MILLER');

6. WAQTD ename,sal,job if they working same as ALLEN job.

➢ SELECT ENAME,SAL,JOB FROM EMP
WHERE JOB=(SELECT JOB FROM EMP
WHERE ENAME='ALLEN');

7. WAQTD all details of emp if they hired after SMITH.

```
SELECT * FROM EMP
WHERE HIREDATE>(SELECT HIREDATE FROM EMP
WHERE ENAME='SMITH');
```

8. WAQTD all the details of emps if they working same dept as SCOTT deptno.

```
SELECT * FROM EMP
WHERE DEPTNO=(SELECT DEPTNO FROM EMP
WHERE ENAME='SCOTT');
```

9. WAQTD ename,sal,job if they are getting sal more than ADAMS getting working same as MARTIN job.

```
SELECT ENAME,SAL,JOB FROM EMP
WHERE SAL>(SELECT SAL FROM EMP
WHERE ENAME='ADAMS')AND JOB=(SELECT JOB FROM EMP
WHERE ENAME='MARTIN');
```

10. WAQTD ename,sal,job,deptno if they are getting sal less than KING and job working same as JONES and not getting any comm.

```
SELECT ENAME,JOB,DEPTNO FROM EMP
WHERE SAL<(SELECT SAL FROM EMP
WHERE ENAME='KING')AND JOB=(SELECT JOB FROM EMP
WHERE ENAME='JONES')AND COMM IS NULL;
```

11. WAQTD all details of emps if they are getting sal more than ALLEN sal and working same as MILLER job and in deptno same as SCOTT dept.

```
SELECT * FROM EMP
WHERE SAL>(SELECT SAL FROM EMP
WHERE ENAME='ALLEN')AND JOB=(SELECT JOB FROM EMP
WHERE ENAME='MILLER')AND DEPTNO=(SELECT DEPTNO FROM EMP
WHERE ENAME='SCOTT');
```

16-10-2022
SUNDAY

❖ NESTED SUB QUERY:-

1. WAQTD max sal of emp table.
   - SELECT MAX(SAL) FROM EMP;

2. WAQTD 1st max sal.
   - SELECT MAX(SAL) FROM EMP;

3. WAQTD max sal,min sal,ename if emps are getting sal more than SMITH sal.
   -

4. WAQTD 2nd max sal.
   - SELECT MAX(SAL) FROM EMP
     WHERE SAL<(SELECT MAX(SAL)
     FROM EMP);

5. WAQTD 3rd max sal.
   - SELECT MAX(SAL) FROM EMP
     WHERE SAL<(SELECT MAX(SAL) FROM EMP
     WHERE SAL<(SELECT MAX(SAL) FROM EMP));

6. WAQTD 3rd min sal.
   - SELECT MIN(SAL) FROM EMP
     WHERE SAL>(SELECT MIN(SAL) FROM EMP
     WHERE SAL>(SELECT MIN(SAL) FROM EMP));

7. WAQTD 5th max sal.
   - SELECT MAX(SAL) FROM EMP
     WHERE SAL<(SELECT MAX(SAL) 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 7th min sal.

```
SELECT MIN(SAL) FROM EMP
WHERE SAL>(SELECT MIN(SAL) FROM EMP
WHERE SAL>(SELECT MIN(SAL) FROM EMP
WHERE SAL>(SELECT MIN(SAL) FROM EMP
WHERE SAL>(SELECT MIN(SAL) FROM EMP
WHERE SAL>(SELECT MIN(SAL) FROM EMP
WHERE SAL>(SELECT MIN(SAL) FROM EMP))))));
```

9. WAQTD 9th max sal.

```
SELECT MAX(SAL) FROM EMP
WHERE SAL<(SELECT MAX(SAL) FROM EMP
WHERE SAL<(SELECT MAX(SAL) FROM EMP
WHERE SAL<(SELECT MAX(SAL) FROM EMP
WHERE SAL<(SELECT MAX(SAL) FROM EMP
WHERE SAL<(SELECT MAX(SAL) FROM EMP
WHERE SAL<(SELECT MAX(SAL) FROM EMP
WHERE SAL<(SELECT MAX(SAL) FROM EMP
WHERE SAL<(SELECT MAX(SAL) FROM EMP)))))))));
```

10. WAQTD all the details of emps who is getting 1st max sal.

```
SELECT * FROM EMP
WHERE SAL=(SELECT MAX(SAL) FROM EMP);
```

- **It is possible to nest 255 queries.**

17-10-2022
MONDAY

❖ **NESTED SUB QUERY:- CASE 2**

1. WAQTD ename,job,if the emps is working in research dept.

```
SELECT ENAME,JOB FROM EMP
WHERE DEPTNO=(SELECT DEPTNO FROM DEPT
WHERE DNAME='RESEARCH');
```

2. WAQTD dname,loc of emp MILLER

```
SELECT DNAME,LOC FROM DEPT
WHERE DEPTNO=(SELECT DEPTNO FROM EMP
WHERE ENAME='MILLER');
```

3. WAQTD all details of dept if they are getting sal more than FORD sal

```
SELECT * FROM DEPT
WHERE DEPTNO=(SELECT DEPTNO FROM EMP
WHERE SAL>(SELECT SAL FROM EMP
WHERE ENAME='FORD'));
```

4. WAQTD all the details of dept if they are working same as BLAKE job.

```
SELECT * FROM DEPT
WHERE DEPTNO IN(SELECT DEPTNO FROM EMP
WHERE JOB=(SELECT JOB FROM EMP
WHERE ENAME='BLAKE'));
```

❖ **TYPES OF SUB QUERY:-**

1) SINGLE ROW SUB QUERY:-
   - From inner query,when we get single partial output, then it is known as SINGLE ROW SUB QUERY.
   - Comparison and relational operator are used in it.

2) MULTI ROW SUB QUERY:-
   - From inner query, when we get multiple partial output, then it is known as MULTIPLE ROW SUB QUERY.
   - Special operators and sub query operators are used in it.

- **Equal operator is faster than In operator.**

5. WAQTD all the details of emps if they are hired after BLAKE and working in CHICAGO.

```
SELECT * FROM EMP
WHERE HIREDATE>(SELECT HIREDATE FROM EMP
WHERE ENAME='BLAKE') AND
DEPTNO=(SELECT DEPTNO FROM DEPT
WHERE LOC='CHICAGO');
```

6. WAQTD all details of emps if they are working in accounting and getting sal less than KING sal.

```
SELECT * FROM EMP
WHERE DEPTNO=(SELECT DEPTNO FROM DEPT
WHERE DNAME='ACCOUNTING') AND SAL<(SELECT SAL FROM EMP
WHERE ENAME='KING');
```

7. WAQTD all the details of dept if they are getting sal more than SMITH sal and hired after ALLEN and working same as MILLER job and getting sal less than KING sal.

```
SELECT * FROM DEPT
WHERE DEPTNO IN(SELECT DEPTNO FROM EMP
WHERE SAL>(SELECT SAL FROM EMP
WHERE ENAME='SMITH') AND HIREDATE>(SELECT HIREDATE FROM EMP
WHERE ENAME='ALLEN') AND JOB=(SELECT JOB FROM EMP
WHERE ENAME='MILLER') AND SAL<(SELECT SAL FROM EMP
WHERE ENAME='KING'));
```

- **CONDITIONS :-**

1) Inner query **SELECT CLAUSE COL_NAME** should be same as Outer query **WHERE CLAUSE COL_NAME**.

2) Inner query **SELECT CLAUSE COL_NAME datatype** should be same as Outer query **WHERE CLAUSE COL_NAME datatype.**

3) We can't write multiple column in **Inner query as to be compared with outer query where clause**.

18-10-2022
TUESDAY

❖ **ALL OPERATOR:-**

- It is a multi value operator which takes single value at LHS and multiple value at RHS along with comparison operator.
- It is sub query operator.
- All operator is same as **AND operator.**

❖ **ANY OPERATOR:-**

- It is a multi value operator which takes single value at LHS and multiple value at RHS along with comparison operator.
- It is sub query operator.
- All operator is same as **OR operator.**

❖ **NESTED SUB QUERY:-**

1. WAQTD ename,sal,job if they are getting sal more than all the managers sal.

➢ SELECT ENAME,SAL,JOB FROM EMP
WHERE SAL>ALL(SELECT SAL FROM EMP
WHERE JOB='MANAGER');

➢ SELECT ENAME,SAL,JOB FROM EMP
WHERE SAL>(SELECT MAX(SAL) FROM EMP
WHERE JOB='MANAGER'); **(WITHOUT USING SUB QUERY OPERATOR)**

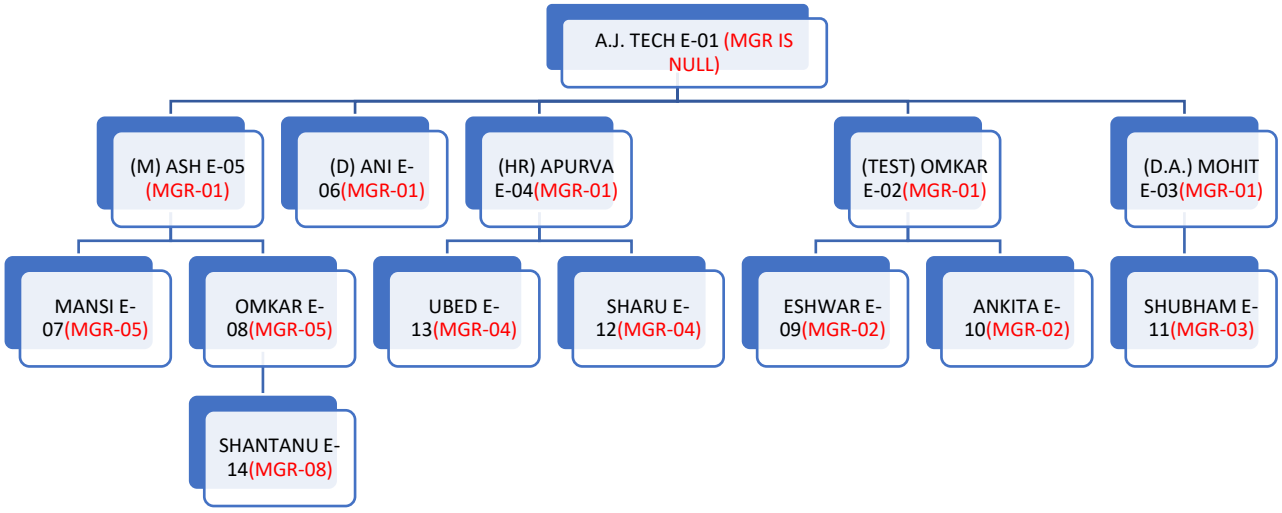
2. WAQTD ename,sal if they are getting sal more than any of the SALESMAN sal.

➢ SELECT ENAME,SAL FROM EMP
WHERE SAL>ANY(SELECT SAL FROM EMP
WHERE JOB='SALESMAN');

➢ SELECT ENAME,SAL FROM EMP
WHERE SAL>(SELECT MIN(SAL) FROM EMP
WHERE JOB='SALESMAN'); **(WITHOUT USING SUB QUERY OPERATOR)**

❖ **RELATIONSHIP BETWEEN REPORTING MANAGER AND EMPLOYEE:-**

NAME OF MGR(MANAGER) ➔ INNER QUERY(MGR) AND OUTER QUERY(EMPNO)
REPORTING TO MGR(EMP) ➔ INNER QUERY(EMPNO) AND OUTER QUERY(MGR)



3. WAQTD ename of reporting manager of SMITH.

➢ SELECT ENAME FROM EMP
WHERE EMPNO=(SELECT MGR FROM EMP
WHERE ENAME='SMITH');

4. WAQTD ename,who is reporting to CLARK.

➢ SELECT ENAME FROM EMP
WHERE MGR=(SELECT EMPNO FROM EMP
WHERE ENAME='CLARK');

5. WAQTD reporting manager name of FORD.

➢ SELECT ENAME FROM EMP
WHERE EMPNO=(SELECT MGR FROM EMP
WHERE ENAME='FORD');

6. WAQTD ename,who is reporting to ANALYST.

```
SELECT ENAME FROM EMP
WHERE MGR IN(SELECT EMPNO FROM EMP
WHERE JOB='ANALYST');
```

7. WAQTD reporting manager name of CLERK.

```
SELECT ENAME
FROM EMP WHERE EMPNO IN(SELECT MGR FROM EMP
WHERE JOB ='CLERK');
```

8. WAQTD reporting manager name if they are hired on JAN.

```
SELECT ENAME FROM EMP
WHERE EMPNO=(SELECT MGR FROM EMP
WHERE HIREDATE LIKE '%JAN%');
```

9. WAQTD ename who is reporting to PRESIDENT.

```
SELECT ENAME FROM EMP
WHERE MGR=(SELECT EMPNO FROM EMP
WHERE JOB='PRESIDENT');
```

10. WAQTD manager's manager name of SMITH.

```
SELECT ENAME FROM EMP
WHERE EMPNO=(SELECT MGR FROM EMP
WHERE EMPNO=(SELECT MGR FROM EMP
WHERE ENAME='SMITH'));
```

11. WAQTD manager's manager's manager name of SMITH.

```
SELECT ENAME FROM EMP
WHERE EMPNO=(SELECT MGR FROM EMP
WHERE EMPNO=(SELECT MGR FROM EMP
WHERE EMPNO=(SELECT MGR FROM EMP
WHERE ENAME='SMITH')));
```

19-10-2022
WEDNESDAY

❖ **JOINS:-**

- **Retrival of data from multiple tables simultaneously.**

❖ **TYPES OF JOINS:- IMP**

1) CROSS JOIN
2) INNER JOIN
3) NATURAL JOIN
4) OUTER JOIN:-
    A) LEFT OUTER JOIN
    B) RIGHT OUTER JOIN
    C) FULL OUTER JOIN
5) SELF JOIN

1) CROSS JOIN:-

- (No condition)
- It is a join used to get combination of one or more tables.
- **SYNTAX:-1) ANSI:** SELECT COL1,COL2,COLN
  FROM TABLE1 CROSS JOIN TABLE2;
  2)ORACLE: SELECT COL1,COL2,COLN
  FROM TABLE1,TABLE2;
- It is also known as **CARTESION JOIN.**

1. WAQTD ename,job sal ,loc if emp is getting sal more than SMITH sal.

➢

20-10-2022
THURSDAY

- **ERROR RECORD:-** Even though there is no connection between 2 tables, cross join generate some output.

2. WAQTD ename,dname which is present in emp table and dept table.

➢ SELECT ENAME,DNAME FROM EMP,DEPT;

2) INNER JOIN:-

- **Only to get matched record.**
- It is a join used to get only the matched record.
- **SYNTAX:- 1. ANSI:-** SELECT COL1,COL2,COLN
  FROM TABLE1 INNER JOIN TABLE2
  ON TABLE1.KEY_ATT=TABLE2.KEY_ATT;

  2.ORACLE:-SELECT COL1,COL2,COLN
  FROM TABLE1,TABLE2
  WHERE TABLE1.KEY_ATT=TABLE2.KEY_ATT;
- Also known as **EQUI JOIN.**

1. WAQTD ename,dname,loc if emp is getting sal more than 2000 in RESEARCH dept.

➢ SELECT ENAME,DNAME,LOC
FROM EMP,DEPT
WHERE EMP.DEPTNO=DEPT.DEPTNO AND SAL>2000
ANDDNAME='RESEARCH';

➢ SELECT ENAME,DNAME,LOC
FROM EMP INNER JOIN DEPT
ON EMP.DEPTNO=DEPT.DEPTNO
WHERE SAL>2000 AND DNAME='RESEARCH';(**USING ANSI SYNTAX**)

2. WAQTD ename,sal,job,loc if ename starts with S and job ends with MAN and loc starts with D.

➢ SELECT ENAME,SAL,JOB,LOC
FROM EMP,DEPT
WHERE EMP.DEPTNO=DEPT.DEPTNO AND ENAME LIKE 'S%' AND JOB LIKE '%MAN' AND LOC LIKE 'D%';

3. WAQTD ename,sal,dname if emps are working in CHICAGO and getting sal more than SMITH sal.

➢ SELECT ENAME,SAL,DNAME
FROM EMP,DEPT
WHERE EMP.DEPTNO=DEPT.DEPTNO AND LOC='CHICAGO' AND
SAL>(SELECT SAL FROM EMP
WHERE ENAME='SMITH');

4. WAQTD all the details of emps along with loc if the emp name starts with A and they are getting sal less than KING sal and working in dname ends with S.

➢ SELECT EMP.*,LOC FROM EMP,DEPT
WHERE EMP.DEPTNO=DEPT.DEPTNO AND ENAME LIKE 'A%' AND
SAL<(SELECT SAL FROM EMP
WHERE ENAME='KING') AND DNAME LIKE '%S';

5. WAQTD max sal,min sal,dname if emps are getting sal more than 2000.

➢ SELECT MAX(SAL),MIN(SAL),DNAME FROM EMP,DEPT
WHERE EMP.DEPTNO=DEPT.DEPTNO AND SAL>2000
GROUP BY DNAME;

- **DESC EMP:- NAME,NULL?,TYPE**
- **SPOOL➔SPOOL FILE➔ (TO SAVE SQL COMMAND)**
- **spool "c:\sql20.txt"**
- **spool off; (AFTER WORK DONE)**

3) NATURAL JOIN:-

- Whenever we don't know the table structure , use **NATURAL JOIN.**
- **SYNTAX:- 1. ANSI:-** SELECT COL1,COL2,COLN
  FROM TABLE1 NATURAL JOIN TABLE2;
  2.ORACLE:- NO SYNTAX.(for natural join there is no oracle syntax because it is same as cross join)

CASE 1:- If naturally there is connection between 2 tables,natural join give output of **INNER JOIN.**

CASE2:- If naturally there is no connection between 2 tables,natural join give output of **CROSS JOIN.**

1. WAQTD ename,dname which is from emp and dept table.

➢ SELECT ENAME,DNAME FROM EMP NATURAL JOIN DEPT;

2. WAQTD sname,ename which is from student table and emp table.

➢ SELECT SNAME,ENAME FROM STUDENT NATURAL JOIN EMP;

21-10-2022

FRIDAY

4) OUTER JOIN:- IMP

- **It is a join used to display both matched and unmatched record.**

a) LEFT OUTER JOIN:-

- **It is join used to display both matched and unmatched record of left side table.**
- **SYNTAX:-**
- **1) ANSI:-**

```
SELECT COL1,COL2,COLN
FROM TABLE1 LEFT OUTER JOIN TABLE2
ON TABLE1.KEY_ATTR=TABLE2.KEY_ATTR;
```

2)ORACLE:-

```
SELECT COL1,COL2,COLN
FROM TABLE1 , TABLE2
WHERE TABLE1.KEY_ATTR=TABLE2.KEY_ATTR(+);
```

b) RIGHT OUTER JOIN:-

- **It is join used to display both matched and unmatched record of RIGHT side table.**
- **SYNTAX:-**
- **1) ANSI:-**

```
SELECT COL1,COL2,COLN
FROM TABLE1 RIGHT OUTER JOIN TABLE2
ON TABLE1.KEY_ATTR=TABLE2.KEY_ATTR;
```

2)ORACLE:-

```
SELECT COL1,COL2,COLN
FROM TABLE1 , TABLE2
WHERE TABLE1.KEY_ATTR(+)=TABLE2.KEY_ATTR;
```

EX. WAQTD DNAME,ENAME CHECK IF ALL LOC HAVE SOME EMPS IN IT OR NOT.(ANSI)

- SELECT DNAME,ENAME,LOC
  FROM DEPT LEFT OUTER JOIN EMP
  ON DEPT.DEPTNO=EMP.DEPTNO;
- SELECT DNAME,ENAME,LOC
  FROM EMP RIGHT OUTER JOIN DEPT
  ON DEPT.DEPTNO=EMP.DEPTNO;

EX. WAQTD DNAME,ENAME CHECK IF ALL LOC HAVE SOME EMPS IN IT OR NOT.(ORACLE)

- SELECT ENAME,DNAME,LOC
  FROM DEPT,EMP
  WHERE DEPT.DEPTNO=EMP.DEPTNO(+);(**LOJ**)
- SELECT ENAME,DNAME,LOC
  FROM DEPT,EMP
  WHERE EMP.DEPTNO(+)=DEPT.DEPTNO;(**ROJ**)

1. WAQTD cname,pname check if all the product are purchased from some customer or not**.(ANSI)**

- SELECT CNAME,PNAME
  FROM PROD LEFT OUTER JOIN CUST
  ON PROD.CUSTID=CUST.CUSTID;
- SELECT CNAME,PNAME
  FROM CUST RIGHT OUTER JOIN PROD
  ON PROD.CUSTID=CUST.CUSTID;

2. WAQTD sname,cname check if all the company is hiring student or not**.(ANSI)**

- SELECT CNAME,SNAME
  FROM PLACEMENT LEFT OUTER JOIN STUDENT
  ON PLACEMENT.STUDID=STUDENT.STUDID;
- SELECT CNAME,SNAME
  FROM STUDENT RIGHT OUTER JOIN PLACEMENT
  ON PLACEMENT.STUDID=STUDENT.STUDID;

3. WAQTD cname,pname check if all the product are purchased from some customer or not**.(ORACLE)**

- SELECT CNAME,PNAME
  FROM PROD ,CUST
  WHERE PROD.CUSTID=CUST.CUSTID(+);
- SELECT CNAME,PNAME
  FROM PROD ,CUST
  WHERE CUST.CUSTID(+)=PROD.CUSTID;

4. WAQTD sname,cname check if all the company is hiring student or not**.(ORACLE)**

- SELECT CNAME,SNAME
  FROM PLACEMENT ,STUDENT
  WHERE PLACEMENT.STUDID=STUDENT.STUDID(+);
- SELECT CNAME,SNAME
  FROM STUDENT , PLACEMENT
  ON STUDENT.STUDID(+)=PLACEMENT.STUDID;

c) FULL OUTER JOIN:-

- **It is a join used to display both the table matched and un matched record.**
- **SYNTAX:-**
- **1) ANSI:-** SELECT COL1,COL2,COLN
  FROM TABLE1 FULL OUTER JOIN TABLE2
  ON TABLE1.KEY_ATTR=TABLE2.KEY_ATTR;

2)ORACLE:- FOR FULL OUTER JOIN , THERE IS NO ORACLE SYNTAX.

5. WAQTD ename,dname check if both emp and dept is having some workers in it or not.

- SELECT ENAME,DNAME
  FROM EMP FULL OUTER JOIN DEPT
  ON EMP.DEPTNO=DEPT.DEPTNO;

22-10-2022

SATURDAY

1. WAQTD reporting manager name of SMITH.

- SELECT ENAME FROM EMP
  WHERE EMPNO=(SELECT MGR FROM EMP
  WHERE ENAME='SMITH');

5) SELF JOIN:-

- **Retrival of data within same table but the output is present in different rows.**
- **SYNTAX:-**
  1) **ANSI:-** SELECT COL1,COL2,COLN
     FROM TABLE1[ALIASNAME] JOIN TABLE2[ALIASNAME]
     ON [ALIASNAME].COLNAME=[ALIASNAME].COLNAME;

2)**ORACLE:-** SELECT COL1,COL2,COLN

FROM TABLE1[ALIASNAME], TABLE2[ALIASNAME]

WHERE [ALIASNAME].COLNAME=[ALIASNAME].COLNAME;

2. WAQTD ename,Manager name from emp table.

- SELECT E1.ENAME,M1.ENAME
  FROM EMP E1,EMP M1
  WHERE E1.MGR=M1.EMPNO;

3. WAQTD ename,mname,esal,msal,ejob,mjob if emps are getting sal more than 2000 and manager are working as ANALYST.

- SELECT E1.ENAME,M1.ENAME,E1.SAL,M1.SAL,E1.JOB,M1.JOB
  FROM EMP E1,EMP M1
  WHERE E1.MGR=M1.EMPNO AND E1.SAL>2000 AND M1.JOB='ANALYST';

4. WAQTD ename,mname,esal,msal,edeptno,mdeptno if the emps job starts with M and mname ends with G.

```
SELECT E1.ENAME,M1.ENAME,E1.SAL,M1.SAL,E1.DEPTNO,M1.DEPTNO
FROM EMP E1,EMP M1
WHERE E1.MGR=M1.EMPNO AND E1.JOB LIKE'M%' AND M1.ENAME
LIKE'%G';
```

5. WAQTD ename,mname,esal,msal if emps are getting some sal and manager's sal is more than SMITH sal.

```
SELECT E1.ENAME,M1.ENAME,E1.SAL,M1.SAL
FROM EMP E1,EMP M1
WHERE E1.MGR=M1.EMPNO AND E1.SAL IS NOT NULL AND
M1.SAL>(SELECT SAL FROM EMP
WHERE ENAME='SMITH');
```

6. WAQTD emp max sal,manager max sal in each dept of both.

```
SELECT MAX(E1.SAL),MAX(M1.SAL),E1.DEPTNO,M1.DEPTNO
FROM EMP E1,EMP M1
WHERE E1.MGR=M1.EMPNO
GROUP BY E1.DEPTNO,M1.DEPTNO;
```

7. WAQTD ename,mname,mdeptname.

```
SELECT E1.ENAME,M1.ENAME,D1.DNAME
FROM EMP E1,EMP M1,DEPT D1
WHERE E1.MGR=M1.EMPNO AND M1.DEPTNO=D1.DEPTNO;
```

8. WAQTD ename,mname,mloc if manager is working in CHICAGO.

```
SELECT E1.ENAME,M1.ENAME,D1.LOC
FROM EMP E1,EMP M1,DEPT D1
WHERE E1.MGR=M1.EMPNO AND M1.DEPTNO=D1.DEPTNO AND
D1.LOC='CHICAGO';
```

23-10-2022

SUNDAY

- **IN ANSI SYNTAX, MORE THAN 2 TABLES IN FROM CLAUSE IS NOT POSSIBLE.**

1. WAQTD ename,mname,esal,msal, if manager name is KING and emps who are not getting any comm.

➢ SELECT E1.ENAME,M1.ENAME,E1.SAL,M1.SAL FROM EMP E1 JOIN EMP M1
ON E1.MGR=M1.EMPNO
WHERE M1.ENAME='KING' AND E1.COMM IS NULL;

2. WAQTD ename,mname,edname,if emps are working in RESEARCH dept.

➢ SELECT E1.ENAME,M1.ENAME,D1.DNAME FROM EMP E1 JOIN EMP M1
ON E1.MGR=M1.EMPNO INNER JOIN DEPT D1
ON E1.DEPTNO=D1.DEPTNO
WHERE D1.DNAME='RESEARCH';

3. WAQTD manager’s manager’s manager name of SMITH.

➢ SELECT E1.ENAME,M1.ENAME,M2.ENAME FROM EMP E1 JOIN EMP M1
ON E1.MGR=M1.EMPNO JOIN EMP M2
ON M1.MGR=M2.EMPNO
WHERE E1.ENAME='SMITH';

4. WAQTD manager’s manager’s manager’s manager name of SMITH.

➢ SELECT E1.ENAME,M1.ENAME,M2.ENAME,M3.ENAME FROM EMP E1 JOIN EMP M1
ON E1.MGR=M1.EMPNO JOIN EMP M2
ON M1.MGR=M2.EMPNO JOIN EMP M3
ON M2.MGR=M3.EMPNO
WHERE E1.ENAME='SMITH';

❖ **PSEUDO COLUMN:-**

- **It is a false column which is present in database for every table.**
- If we want ,we need to invocate.

❖ **TYPES OF PSEUDO COLUMN:-**

1) ROW ID
2) ROW NUM

1) ROW ID:-

- **It is pseudo column which is present in database.**
- SELECT ROWID,EMP.* FROM EMP;

❖ **CHARACTERSTICS OF ROW ID:-**

1. It is unique.
2. It is not null.
3. It is combination of unique and not null.
4. Row id can act as Primary key of table not as foreign key.
5. Row id is generated at the time of insertion of data.
6. Row id is static in nature.(constant)

2) ROW NUM:-

- **It is pseudo column which is present in database.**
- SELECT ROWNUM,EMP.* FROM EMP;

❖ **CHARACTERISTICS OF ROW NUM:-**

1. Row num is integer.
2. Row num always starts with 1.
3. Row num is dynamic in nature.
4. Row num is generated at the time of execution.

1. WAQTD top 5 rows of emp table.
   - SELECT ROWNUM,EMP.* FROM EMP
     WHERE ROWNUM<6;

2. WAQTD exactly 1st row of emp table.
   - SELECT ROWNUM,EMP.* FROM EMP
     WHERE ROWNUM=1;

3. WAQTD exactly 2nd row of emp table.
   -

28-10-2022

FRIDAY

1. WAQTD exactly 2nd row of emp table.

```
SELECT *
FROM (SELECT ROWNUM R1,EMP.*
FROM EMP)
WHERE R1=2;
```

2. WAQTD Exactly 10th row of emp table.

```
SELECT *
FROM (SELECT ROWNUM R1,EMP.*
FROM EMP)
WHERE R1=10;
```

3. WAQTD 2,4,6,8,10 row of emp table.

```
SELECT *
FROM (SELECT ROWNUM R1,EMP.*
FROM EMP)
WHERE R1 IN(2,4,6,8,10);
```

4. WAQTD bottom 3 rows of emp table.

```
SELECT *
FROM (SELECT ROWNUM R1,EMP.* FROM EMP)
WHERE R1>(SELECT COUNT(*)-3 FROM EMP);
```

5. WAQTD bottom 5 rows.

```
SELECT * FROM (SELECT ROWNUM R1,EMP.* FROM EMP)
WHERE R1>(SELECT COUNT(*)-5 FROM EMP);
```

29-10-2022

SATURDAY

❖ FIND NTH MAX AND NTH MIN BY USING ROWNUM:-

1. WAQTD 5th max sal.

➢
```
SELECT *
FROM(SELECT ROWNUM R1,SAL
FROM (SELECT DISTINCT(SAL)
FROM EMP
ORDER BY SAL DESC))
WHERE R1=5;
```

2. WAQTD 10th min sal.

➢
```
SELECT *
FROM (SELECT ROWNUM R1,SAL
FROM (SELECT DISTINCT(SAL)
FROM EMP
ORDER BY SAL ASC))
WHERE R1=10;
```

3. WAQTD 2,4,6,8,10 max sal.

➢
```
SELECT *
FROM (SELECT ROWNUM R1,SAL
FROM (SELECT DISTINCT(SAL)
FROM EMP
ORDER BY SAL DESC))
WHERE R1 IN(2,4,6,8,10);
```

4. WAQTD 5th max sal and 10th min sal.

- SELECT *
  FROM (SELECT ROWNUM R1,SAL
  FROM(SELECT DISTINCT(SAL)
  FROM EMP
  ORDER BY SAL DESC)) E1,
  (SELECT ROWNUM R2,SAL
  FROM(SELECT DISTINCT(SAL)
  FROM EMP
  ORDER BY SAL ASC)) E2
  WHERE E1.R1=5 AND E2.R2=10;

❖ **SINGLE ROW FUNCTION:-**

- **It is a function which takes n no. of inputs and generate n no. of output.**

❖ **TYPES OF SINGLE ROW FUNCTION:-**

1. CHARACTER SINGLE ROW FUNCTION
2. NUMBER SINGLE ROW FUNCTION
3. DATE SINGLE ROW FUNCTION
4. SPECIAL SINGLE ROW FUNCTION

1. **CHARACTER SINGLE ROW FUNCTION:-**

1) **UPPER():-** makes all letters **CAPITAL** letter.
2) **LOWER():-** makes all letters **SMALL** letter.
3) **INITCAP():-** makes first or initial letter capital.
4) **REVERSE():-** reverses the letters.
5) **LENGTH():-** gives length of char.

✓ **Dual is a dummy table present in the database.**

EX.

1. SELECT UPPER('sundra') FROM DUAL;
2. SELECT LOWER('SMITH') FROM EMP; **(14 TIMES SAME OUTPUT)**
3. SELECT INITCAP('SUNDRA') FROM DUAL;
4. SELECT LENGTH('SUNDRA') FROM DUAL;

1. WAQTD ename,if ename is having exactly 5 character.

➢ SELECT ENAME FROM EMP
WHERE LENGTH(ENAME)=5;

2. WAQTD ename,job,sal,if they are getting sal exactly in 3 digits.
   - SELECT ENAME,JOB,SAL FROM EMP
     WHERE LENGTH(SAL)=3;

3. WAQTD palindrome name.
   - SELECT ENAME FROM EMP
     WHERE REVERSE(ENAME)=ENAME;

✓ **PALINDROME :- Names which is same when reversed.**

4. WAQTD LAST CHAR IS UPPER.
   - SELECT REVERSE(INITCAP(REVERSE(ENAME))) FROM EMP;

2. NUMBER SINGLE ROW FUNCTION:-

1. SQUARE ROOT:- Takes square root of number.
   Ex. SELECT SQRT(16) FROM DUAL;

2. POWER:- Takes power of number.
   Ex. SELECT POWER(4,2) FROM DUAL;

3. ABSOLUTE:- Converts negative into positive.
   Ex. SELECT ABS(-98.2) FROM DUAL;

01-11-2022

TUESDAY

✓ **DIFFERENCE BETWEEN TRUNC AND TRUNCATE.**

❖ **SINGLE ROW FUNCTION :-**

1) **MOD SINGLE ROW FUNCTION(ARG1,ARG2):-**
   - MOD(a,b)=0;
   - If a<b, then ans is “a”.
   - MOD gives remainder.
   - **EX.** SELECT MOD(10,2) FROM DUAL;

2) **ROUND SINGLE ROW FUNCTION(ARG1,ARG2):-**
   - **EX.** SELECT ROUND(87.376,1) FROM DUAL;
     SELECT ROUND(87.526,1) FROM DUAL;
     SELECT ROUND(87.656,1) FROM DUAL;
     SELECT ROUND(87.656,2) FROM DUAL;
     SELECT ROUND(87.656,-1) FROM DUAL;
   - SELECT ROUND(87.656,-2) FROM DUAL;
   - SELECT ROUND(847.656,-2) FROM DUAL;
   - SELECT ROUND(-847.656,-2) FROM DUAL;
   - SELECT ROUND(-667.656,-2) FROM DUAL;

   - **(98.235,1)**
   - **-2 -1 0 1 2 (num position)**

3) **TRUNC SINGLE ROW FUNCTION(ARG1,ARG2):-**
   - Used to Delete decimal value.
   - **EX.** SELECT TRUNC(78.237,1) FROM DUAL;
   - SELECT TRUNC(78.987,1) FROM DUAL;

❖ **NULL VALUE FUNCTION(ARG1,ARG2) :- (NVL)**

- It is a null value function used to convert null with zero.
- **SYNTAX:-** NVL(ARG1,ARG2);
- If arg1 is not null,take arg1.
- If arg1 is null, take arg2.

❖ **NVL2(ARG1,ARG2,ARG3**):-

- **EX.** SELECT ENAME,SAL+NVL2(COMM,COMM,0) FROM EMP;
- If arg1 is not null,take arg2.
- If arg1 is null, take arg3.

1. WAQTD ename,sal addition with comm.

➢ SELECT ENAME,SAL+NVL(COMM,0) FROM EMP;

❖ **DATE SINGLE ROW FUNCTION:-**

- **EX.** SELECT SYSDATE FROM DUAL; **(SYSTEM DATE)**
- SELECT SYSTIMESTAMP FROM DUAL;

1. **TO_CHAR(DATE,'FORM'):-**
   - **FOR YEAR…….**
     - **EX.** SELECT TO_CHAR(SYSDATE,'YYYY') FROM DUAL;
     - WHERE TO_CHAR(SYSDATE,'YYYY');
     - WHERE TO_CHAR(SYSDATE,'YY');
     - WHERE TO_CHAR(SYSDATE,'YEAR');
   - **FOR MONTH……..**
     - **EX.** SELECT TO_CHAR(SYSDATE,'MON') FROM DUAL;
     - WHERE TO_CHAR(SYSDATE,'MON');
     - WHERE TO_CHAR(SYSDATE,'MM');
     - WHERE TO_CHAR(SYSDATE,'MONTH');
   - **FOR DAY……..**
     - **EX.** SELECT TO_CHAR(HIREDATE,'D') FROM EMP; **(FOR SHOWING DAY** (WAAR)**)**
     - SELECT * FROM EMP
       WHERE TO_CHAR(HIREDATE,'D')=1;

02-11-2022

WEDNESDAY

❖ **CONCATINATION:-**

- It is operator used to merge 2 or more string .

✓ **Information:-** It is collection of meaningful data.

- **EX.** SELECT 'THE ENAME is'|| ENAME FROM EMP;
  SELECT 'THE ENAME is'|| ENAME || 'AND GETTING SAL OF'|| SAL FROM EMP;

1. WAQTD the ename is X working as Y and in deptno Z.

➢ SELECT 'THE ENAME IS '||ENAME||'WORKING AS '||JOB||'IN DEPTNO '||DEPTNO FROM EMP;

✓ DIFFERENCE BETWEEN CONCATINATION AND CONCAT(ARG1,ARG2).

❖ **CONCAT(ARG1,ARG2):-**

- **The empname is X.**
  SELECT CONCAT('THE EMPNAME IS ',ENAME) FROM EMP;
- **The emp name is X and getting sal of Y.**
  SELECT CONCAT('THE EMPNAME IS ',CONCAT(ENAME, CONCAT('AND GETTING SAL',SAL))) FROM EMP;

1. WAQTD the ename is X working as Y and in deptno Z.

➢ SELECT CONCAT('THE EMPNAME IS ',CONCAT(ENAME, CONCAT('AND GETTING SAL',CONCAT(SAL, CONCAT('IN DEPTNO ',DEPTNO))))) FROM EMP;

❖ **SUBSTR(ARG1,ARG2,ARG3):-**

- **It is single row function, which is used to display some part of string.**

- **(ARG1,ARG2,ARG3)**

A.S. POSI. LENGTH

EX.

1. SUBSTR('QSPIDER',1,1)=Q
2. SUBSTR('QSPIDER',-8,1)=Q
3. SUBSTR('ANEKAL',-6,3)=ANE
4. SUBSTR('ANEKAL',4,3)=KAL
5. SUBSTR('ANEKAL',-3,3)=KAL

1. WAQTD Ename,job if name starting 2nd char is "M".

➢ SELECT ENAME,JOB FROM EMP
WHERE SUBSTR(ENAME,2,1)='M';

2. WAQTD Ename,job if name starts WITH VOWELS.

➢ SELECT ENAME,JOB FROM EMP
WHERE SUBSTR(ENAME,1,1)IN('A','E','I','O','U');

3. WAQTD ename,job if the job ends with consonents.

➢ SELECT ENAME,JOB FROM EMP
WHERE SUBSTR(JOB,-1,1) NOT IN('A','E','I','O','U');

4. WAQTD ename,job if the job ends with "L".

➢ SELECT ENAME,JOB FROM EMP
WHERE SUBSTR(JOB,-1,1)='L';

❖ **INSTR(ARG1,ARG2,ARG3,ARG4):-**

- **It is single row function, which is used to display POSITION OF part of string.**
- **(ARG1,ARG2,ARG3,ARG4)**

A.S. CHAR START NO.OF OCCURANCE

EX.

1. INSTR('SMITH','I',1,1)=3
2. INSTR('ALLEN','L',1,2)=3
3. INSTR('ALLEN','L',-1,1)=3
4. INSTR('ALLEN','L',-2,2)=2
5. INSTR('ALLEN','L',1,2)=3
6. INSTR('THE EMPNAME IS ALLEN','E',1,4)=19

❖ **REPLACE SINGLE ROW FUNCTION:- REPLACE(ARG1.ARG2.ARG3)**

- **(ARG1,ARG2,ARG3)**

A.S. STR REPLACE STR

EX.

1. REPLACE('QSPIDERS','Q','J')=JSPIDERS
2. REPLACE('QSPIDERS','D','$')=QSPI$ERS
3. REPLACE('QSPIDERS','S','$')=Q$PIDER$
4. REPLACE('QSPIDERS','QSP','J')=JIDERS

1. WAQTD EVERY ENAME 1ST CHAR REPLACE WITH '$'.
   - SELECT REPLACE(ENAME,SUBSTR(ENAME,1,1),'$') FROM EMP;
   - SELECT '$'||SUBSTR(ENAME,2) FROM EMP;

03-11-2022

THURSDAY

- **WAQTD centre row of emp table.**
  - SELECT *
    FROM(SELECT ROWNUM R1,EMP.* FROM EMP)
    WHERE R1 IN(SELECT COUNT(*)/2 FROM EMP);

- **WAQTD REPLACE LAST CHAR WITH '$' AND 1ST CHAR WITH '@'**
  - SELECT REVERSE('$'||SUBSTR((REVERSE(('@'||SUBSTR(ENAME,2)))),2)) FROM EMP;
  - SELECT '@'|| SUBSTR(ENAME,2,LENGTH(ENAME)-2)||'$' FROM EMP;

- **WAQTD REPLACE LAST CHAR WITH '$' AND BEFORE LAST CHAR, ALL CHAR IN SMALL.**
  - SELECT LOWER(SUBSTR(ENAME,1,LENGTH(ENAME)-1))||'$' FROM EMP;

- **WAQTD S$IT@.**
  - SELECT SUBSTR(ENAME,1,1)||'$'||SUBSTR(ENAME,3,LENGTH(ENAME)-3)||'@' FROM EMP;

- **WAQTD SIH.**
  - SELECT SUBSTR(ENAME,1,1)||'$'||
    SUBSTR(ENAME,3,LENGTH(ENAME)-4)||'$'||SUBSTR(ENAME,-1,1) FROM EMP;

- **EXPERIENCE OF EMPLOYEE.**
  - SELECT TO_CHAR(SYSDATE,'YYYY')-TO_CHAR(HIREDATE,'YYYY') FROM EMP;

    - ✓ DESC TABLE_NAME;(TO DESCRIBE TABLE);

❖ SQL STATEMENTS:-

❖ DML(DATA MANIPULATION LANGUAGE):-

1) **INSERT:-**
   - **EX.** INSERT INTO STUD1
     VALUES('S04','MACHA',1234567890,'MACHA@GMAIL.COM',2022);

2) **UPDATE:-**
   - **EX.** UPDATE STUD1
     SET SMAIL='MACHA@MACHI.COM'
     WHERE STUDID='S04';

3) **DELETE:-**
   - **EX.** DELETE
     FROM STUD1
     WHERE STUDID='S03';

04-11-2022

FRIDAY

❖ DDL(DATA DEFINATION LANGUAGE):-

1) CREATE:-

- **EX.** CREATE TABLE CUSTOMER
  (
  CUSTID CHAR(5) NOT NULL,
  CUSTNAME VARCHAR(20) NOT NULL,
  CCONC CHAR(10) NOT NULL,
  CEMAIL VARCHAR(30) NOT NULL,
  CADD VARCHAR(200) NOT NULL,
  CONSTRAINT PK101 PRIMARY KEY(CUSTID),
  CONSTRAINT UK101 UNIQUE(CCONC),
  CONSTRAINT CK101 CHECK(LENGTH(CCONC)=10),
  CONSTRAINT UK102 UNIQUE(CEMAIL),
  CONSTRAINT CK102 CHECK(INSTR(CEMAIL,'@',1,1)>1)
  );

2) RENAME:-

- **EX.** RENAME CUSTOMER TO CUST;

3) ALTER:- (ADD COLUMN)

- **SYNTAX:-** ALTER TABLE TABLE_NAME
  ADD COL_NAME DATATYPE NULL/NOT NULL;
- **EX.** ALTER TABLE CUST
  ADD CGEN CHAR(6) NOT NULL; **(FOR GENDER)**

  (RENAME COLUMN)
- **EX.** ALTER TABLE CUST
  RENAME COLUMN CUSTNAME TO CNAME;

(DELETE)

- **EX.** ALTER TABLE CUST
  DROP COLUMN ABC;

(CHANGE DATATYPE)

- **EX.** ALTER TABLE CUST
  MODIFY CADD VARCHAR(50);

(CHANGE NULL/NOT NULL)

- **EX.** ALTER TABLE CUST
  MODIFY CEMAIL VARCHAR(30) NOT NULL;

(ADD CONSTRAINT)

- **EX.** ALTER TABLE CUST
  ADD CONSTRAINT CK1003 CHECK(LENGTH(CNAME)>=2);

(DROP CONSTRAINT)

- **EX.** ALTER TABLE TABLE_NAME
  DROP CONSTRAINT CONSTRAINT_REF_NAME;

❖ **SELECT * FROM ALL_CONSTRAINTS**
WHERE TABLE_NAME='TABLE_NAME'; **(TO FIND CONSTRAINT_REF)**

05-11-2022
SATURDAY

❖ **FLASHBACK:- (RETRIEVE TABLE FROM BIN)**

- **EX.** FLASHBACK TABLE ACUST TO BEFORE DROP;
- **SYNTAX:-** FLASHBACK TABLE TABLE_NAME TO BEFORE DROP;

✓ **NOTE:-** DDL is auto commit statement, DML is commit statement .

❖ **TRANSACTION CONTROL LANGUAGE:-**

1) COMMIT:-
   - It is statement used to save all the transaction of DML.

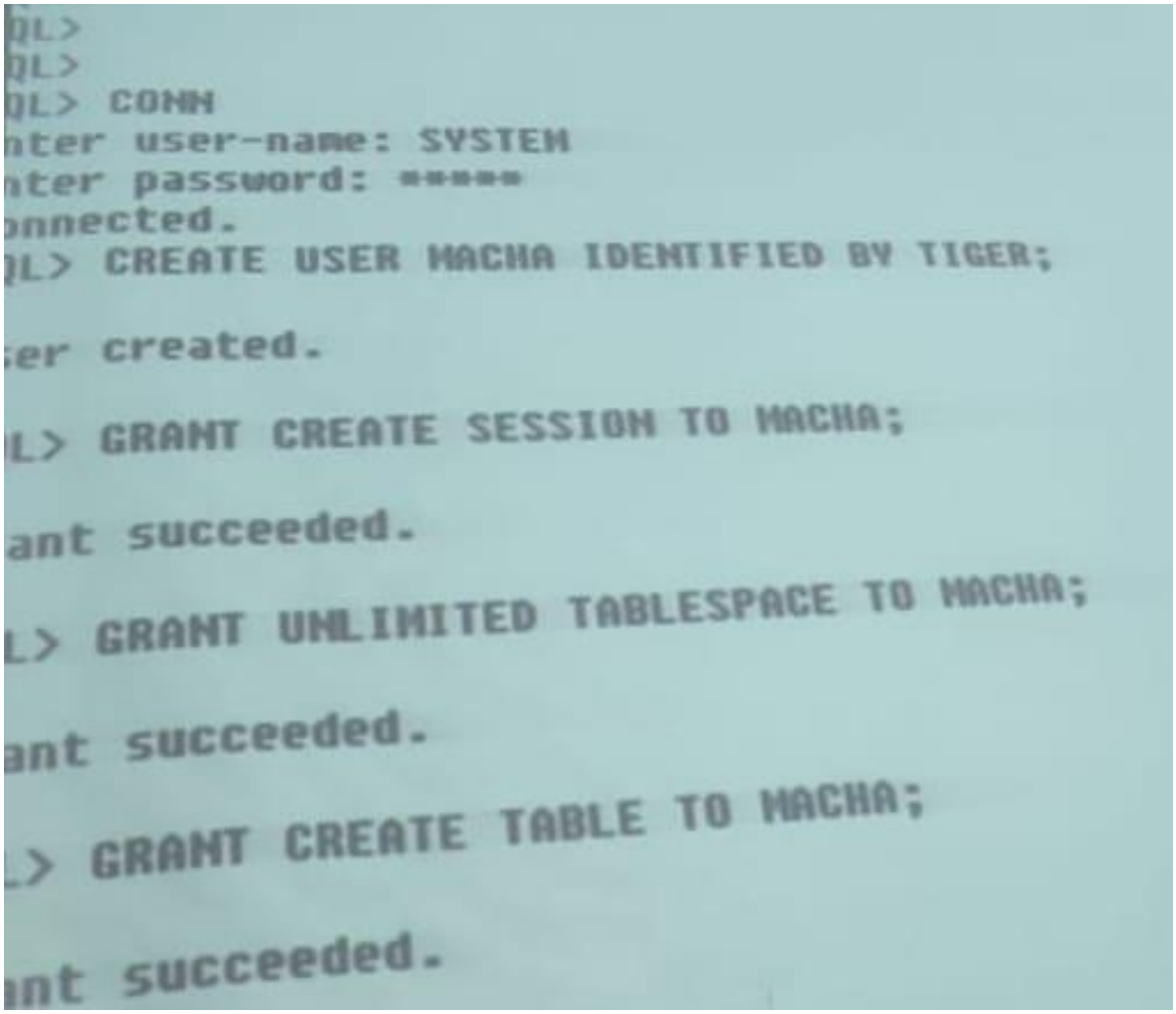
2) ROLLBACK:-
   - It is statement used to get back the deleted and updated data before commit.

3) SAVE POINT:-
   - It is the statement used to get back the data from the table without commit.
   - **Ex.**
   - SAVEPOINT S1;
   - INSERT INTO STUD1
     VALUES('S05','GULDU','9123456789','GULDU@GMAIL.COM','2021');
   - SAVEPOINT S2;
   - UPDATE STUD1
     SET SNAME='KACHADA'
     WHERE STUDID='S02';
   - SAVEPOINT S3;
   - DELETE FROM STUD1
     WHERE STUDID='S01';

❖ **DATA CONTROL LANGUAGE (DCL):-**

1) GRANT:-
   - Give permission to user.
   - **EX.**
     GRANT SELECT ON EMP TO MACHA;
     SELECT * FROM SCOTT.EMP;

2) Revoke:-
   - Take back permission to user.
   - **EX.**
     REVOKE SELECT ON EMP FROM MACHA;



07-11-2022
MONDAY

❖ NORMALISATION:-

❖ TYPES OF ATTRIBUTES :-

1. KEY ATTRIBUTE
2. NON KEY ATTRIBUTE
3. PRIME KEY ATTRIBUTE
4. NON PRIME KEY ATTRIBUTE
5. SUPER KEY ATTRIBUTE
6. COMPOSITE KEY ATTRIBUTE
7. FOREIGN KEY ATTRIBUTE

1) KEY ATTRIBUTE:-

- Also known as 'candidate key'.
- It is the unique attribute present in the table.
- **Ex.**
  STUD SNAME SCON SLOC SBRANCH YOP SEMAIL

2) NON KEY ATTRIBUTE:-

- Non key attributes are not unique.
- **Ex.**
  SNAME,SLOC,SBRANCH,YOP

3) PRIME KEY ATTRIBUTE:-

- Key's which is eligible for becoming a primary key but among them only one can be selected as prime key attribute.
- **Ex.**
  STUDID,SCON,SEMAIL

4) **NON PRIME KEY ATTRIBUTE:-**
   - All the remaining key's which satisfy the characteristics of primary key other than prime key attribute.
   - **Ex.**
     SCON,SEMAIL

5) **SUPER KEY ATTRIBUTE:-**
   - Combination of 2 or more key attributes.
   - **Ex.**
   - SCON,SEMAIL,STUDID+SEMAIL+SCON

6) **COMPOSITE KEY ATTRIBUTE:-**
   - Combination of 2 or more non key attributes.
   - **Ex.**
     SNAME,SLOC,SBRANCH,SNAME+SLOC+SBRANCH

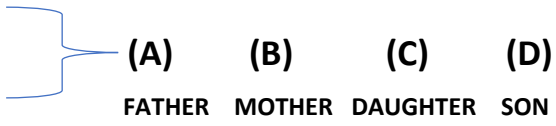
7) **FOREIGN KEY:-**
   - It is used to have connection between 2 tables i.e. from one table to another table.

❖ **TYPES OF DEPENDANCIES:-**

1. TOTAL FUNCTIONAL DEPENDANCIES
2. PARTIAL FUNCTIONAL DEPENDANCIES
3. TRANSITIVE FUNCTIONAL DEPENDANCIES

1) TOTAL FUNCTIONAL DEPENDANCIES:-

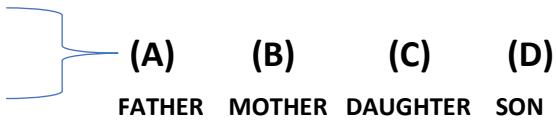
- **A➔B**
  A➔C
  A➔D



- A defines B/B is dependant on A
  A defines C/C is dependant on A
  A defines D/D is dependant on A
- All the attributes depends on one attribute.

2) PARTIAL FUNCTIONAL DEPENDANCIES:-

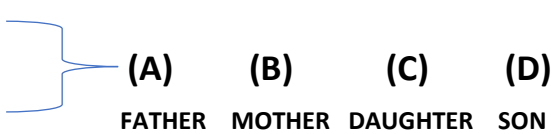
- **A➔C**
  B➔D
  B➔C



- A defines C/C is dependant on A
  B defines D/D is dependant on B
  B defines C/C is dependant on B
- It means composite key attribute depends on 2 attributes.

3) TRANSITIVE FUNCTIONAL DEPENDANCIES:-

- **A➔B**
  A➔C
  C➔D



- A defines B/B is dependant on A
  A defines C/C is dependant on A
  C defines D/D is dependant on C
  (D is indirectly defined by A)
- Attribute indirectly depends on another attribute.
- One non key attribute depends on another non key attribute.

❖ **REDUNDANCIES/REDUNDANCY:-**

- Repeattion of data or duplication of data.

❖ **ANOMALIES:-**

- They can be termed as side effect that we can have in the form of Insert,Update,Delete.

	TFD	**PFD**	**TRFD**
	A → B	A → C	A → B
	A → C	B → D	A → C
	A → D	B → C	C → D A---> D
REDUNDANCY	NO	YES	YES
ANOMALIES	NO	YES	YES

❖ **NORMALISATION:-**

- **Process of reducting a bigger table into smaller table by identifying redundancy and anomalies.**
  OR
  It is a process of reducting bigger table into smaller table by identifying repetations and side effects.

❖ **TYPES OF NORMALISATION:-**

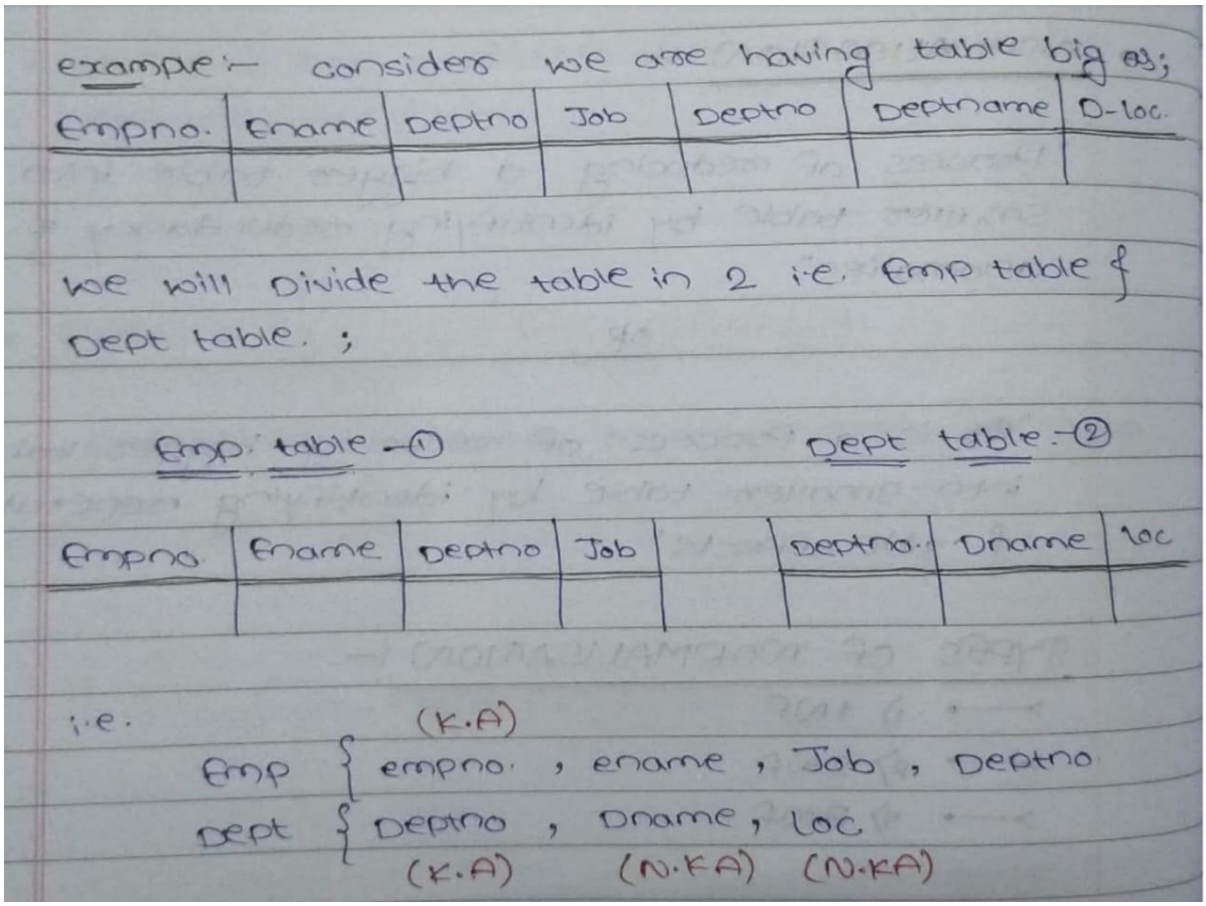
1. 1NF(NORMAL FORM)
2. 2NF
3. 3NF

1) 1NF:-

- Data entered in a cell should be single value data or atomic data.
- If the table is unconditional, convert into ‘total functional dependancies’.

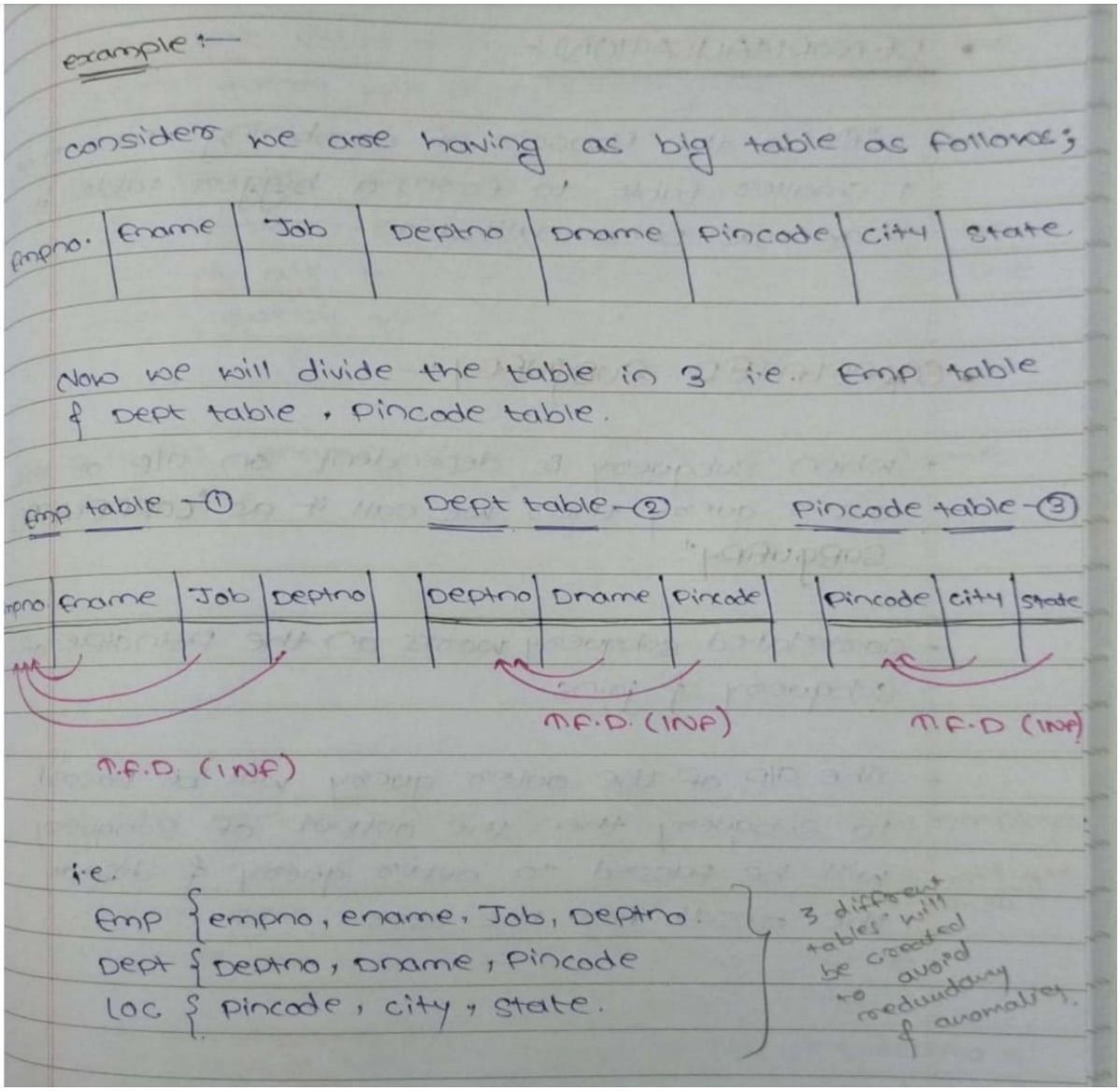
2) 2NF:-

- If table is under 2NF, first it should be complete with 1NF.
- If the table is having ‘partial functional dependancy’,it convert into ‘total functional dependancies’.



3) 3NF:-

- If table is under 3NF, it should be complete 2NF.
- If the table is having ‘transitive functional dependancy’, convert and make into ‘total functional dependancy’.



❖ **DISADVANTAGE OF NORMALISATION:-**

- The only minor disadvantage is that we have to write complex queries as we have more number of tables to be accessed.

❖ **DENORMALISATION:-**

- It is the process of combining more than 1 smaller table to form a bigger table it is known as **DENORMALISATION.**

08-11-2022
TUESDAY

❖ **NOT EXISTS OPERATOR:-**

- It is multivalue operator used to check the partial output.

❖ **CORRELATED SUBQUERY:-**

- **Inner query is dependent on outer query and outer query is dependent on inner query.**
- When subquery is dependent on output of the outer query then we call it as '**CORRELATED SUBQUERY**'.
- Correlated subquery works on principle of subquery and joins.
- The output of the outer query will be passed to subquery then the output of subquery will be passed to outer query and display the result for the user.

1) WAQTD dname which does not have any emp in it.

➢ SELECT DNAME FROM DEPT
WHERE NOT EXISTS(SELECT DEPTNO FROM EMP
WHERE EMP.DEPTNO=DEPT.DEPTNO);

2) WAQTD dname if the dept is having some emps in it.

➢ SELECT DNAME FROM DEPT
WHERE EXISTS(SELECT DEPTNO FROM EMP
WHERE EMP.DEPTNO=DEPT.DEPTNO);

❖ **ACID PROPERTY:-**

- In order to maintain consistency in a database, before and after the transaction, certain properties are followed. These are called **ACID** properties.

ATOMICITY:

Atomicity is also known as the 'All or nothing rule'.

* Each transaction is considered as one unit and either runs to completion or is not executed at all.

CONSISTENCY:

This means that integrity constraints must be maintained so that the database is consistent before and after the transaction. It refers to the correctness of a database. Referring to the example above,

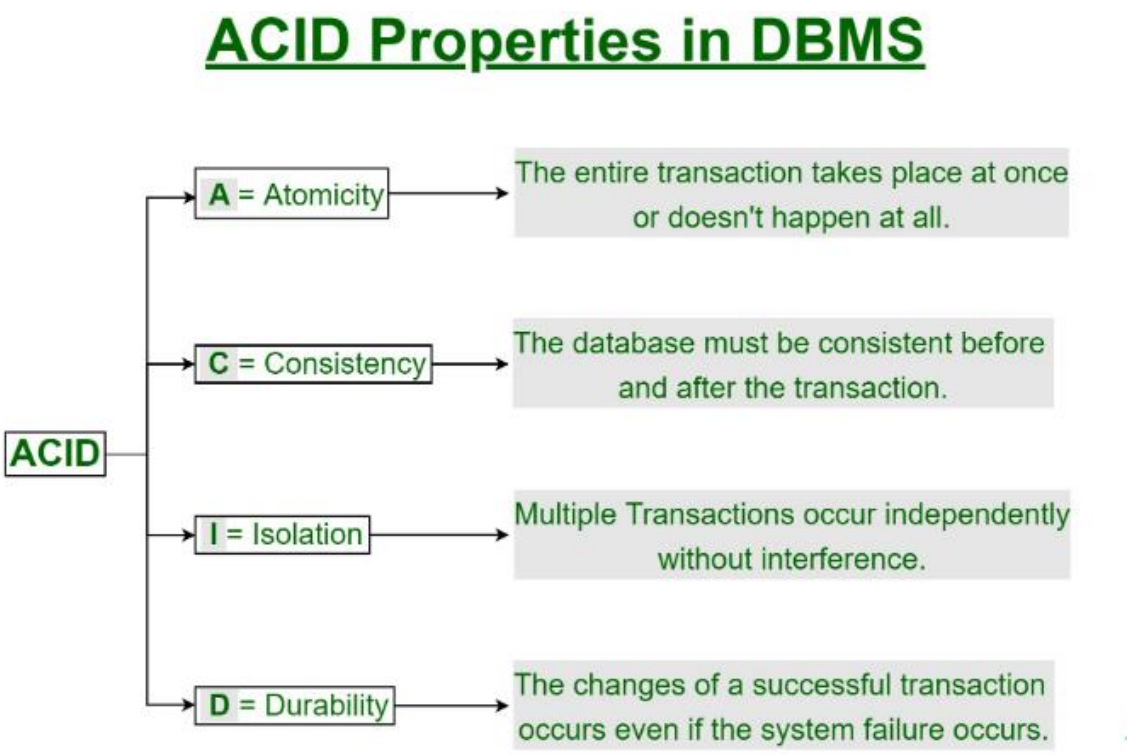
The total amount before and after the **transaction must be maintained the same.**

ISOLATION:

This property ensures that multiple transactions can occur concurrently without leading to the inconsistency of database state. Transactions occur **independently** without interference. Changes occurring in a particular transaction will not be visible to any other transaction until that particular change in that transaction is written to memory or has been committed. This property ensures that the execution of transactions concurrently will result in a state that is equivalent to a state achieved these were executed serially in some order.

DURABILITY:

This property ensures that once the transaction has completed execution, the updates and modifications to the database are stored in and written to disk and they persist even if a system failure occurs. These updates now become **permanent and are stored in non-volatile** memory. The effects of the transaction, thus, are never lost.



09-11-2022
WEDNESDAY

❖ **VIEW:-**

- A view is a virtual table which consists of a subset of data contained in a table.
- Views are not virtually present, and it takes less space to store.
- View can have data of one or more tables combined, and it is depending on the relationship.
- **EX.**
  CREATE VIEW FILE_NAME AS
  (QUERY……);
- **API:**- application programming interface.
- **SELECT * FROM VIEW NAME;** (to display saved query)

❖ **INDEX:-**

- An index is performance tuning method of allowing faster retrieval of records from the table. An index creates an entry for each value and it will be faster to retrieve data.
- THREE TYPES:

1) UNIQUE INDEX
2) CLUSTERED INDEX:- PROP.(PRIMARY KEY),ROWID
3) NON CLUSTERED INDEX:- PROP.(FOREIGN KEY)

1) UNIQUE INDEX.

- This indexing does not allow the field to have duplicate values if the column is unique indexed. Unique index can be applied automatically when primary key is defined.

2) CLUSTERED INDEX.

- This type of index reorders the physical order of the table and search based on the key values. Each table can have only one clustered index.

3) NON CLUSTERED INDEX.

- Non Clustered Index does not alter the physical order of the table and maintains logical order of data. Each table can have 999 non clustered indexes.

❖ TRIGGER

- A DB trigger is a code or programs that automatically execute with response to some event on a table or view in a database.
- Mainly, trigger helps to maintain the integrity of the database.
- Example: When a new student is added to the student database, new records should be created in the related tables like Exam, Score and Attendance tables.