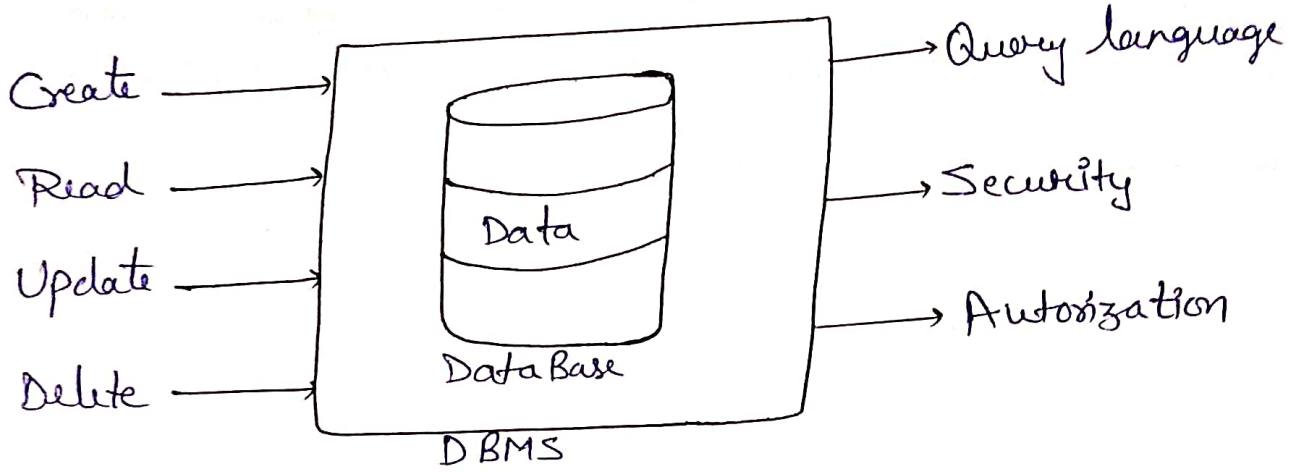
SQL

DataBase Management System



- It is a software which is used to manage and maintain the database.
- Security and Autorization are two main feature that provided by DBMS.
- We use query language to communicate or interact with a database management system.
- Restore data in DBMS in the form of files.

Relation Data Base Management System (RDBMS)

- It is a type of DBMS software. In which we store the data in the form of table.
- We use SQL to communicate or interact with RDBMS.

Tables → It is a logical organization of data which consists of rows and columns.

- Row run horizontally & columns vertically.

Rows/ Records/ tuples

RollNo	Name	Address
1	AB	Delhi
2	CD	Noida → cell

Columns / Attribute / Fields

Relational Model:- It is designed by E.F. CODD.

- In relational model we store the data in the form of tables.
- DBMS which follows relational model becomes RDBMS.
- DBMS which follows rules of E.F.CODD becomes RDBMS.

DBMS ⟶ [Relational Model] ⟶ RDBMS

Rule of E.F. CODD

Rule of E.F.CODD:-

1.) The data entered into a cell must always be a single value data.

2.) According to E.F.CODD we can store the data in multiple tables if needed we can stables a connection between the table with the help of key attribute.

3.) In RDBMS, we store everything in the form of table include meta data.

→ Meta Data:- The details about the data is called as meta data.

Sno	Name	Photo
1	AB	
2	CD	

→ Name - Pic
size - 112KB
formate - JPg
Resolution - 400X300

Meta Data

Name	Size	formate	Resolution
Pic	112KB	JPg	400X300

4.) In RDBMS the data enter into the table can be validate in two steps.

(i) By assigning data type
(ii) By assigning constraints

- Data type are mandatory where as constraints are optional.

Data Type :- It is used to specified or determined the type of data that will be stored in a particular memory location.

Types of data types in SQL -

1.) CHAR
2.) VARCHAR / VARCHAR2
3.) Date
4.) Number
5.) Large Object
   ↳ Character Large Object (CLOB)
   ↳ Binary Large Object (BLOB)

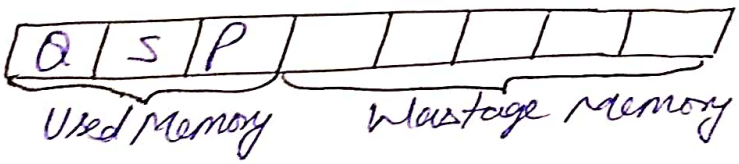
Note :- SQL is not a case sensitive language but it is a case insensitive language.

1.) CHAR :- In CHAR data type we can store A-Z, a-z, 0-9 and special character #, $, @.

- Character must be enclosed with a single quotes.
- When we use CHAR data type we mention size.
- It is used to specified number of character it can store
- The maximum character it can store upto 2000 character
- CHAR data type follows fixed length memory allocation.

Syntax - CHAR (size)

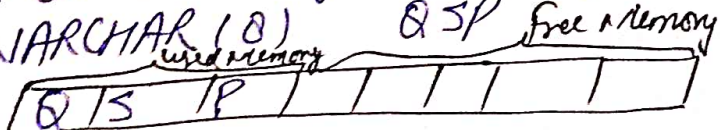
Ex - CHAR(10) 'QSP'



2.) VARCHAR :- In VARCHAR data we can store A-Z, a-z, 0-9 and special character (#, @, $).

- Character must be enclosed with in single quotes whenever we used VARCHAR we must mention size, it can store upto 2000 character.
- It follows variable length memory allocation.

Syntax - VARCHAR (10) 'QSP'



→ VARCHAR2 :- It is an updated version of VARCHAR. It can store upto 4000 character.

3.] Date :- It is used to store data in a particular format. It is used oracle specified format.

Syntax- Date

Formate- DD-MM-YY OR DD-MM-YYYY
23-MAR-23

4.] Number :- It is used to store numeric value.

Syntax- Number (Precision, [Scale])

[] → optional

→ Precision :- It is used to determine the number of integer value.

Ex- Number (5) → ±99999

→ Scale :- It is used to determine the number of digit used to store decimal value with in the precision.

Ex- Number (5,2) → ± 999.99

Number (5,7) → ± 0.0099999

5.] Large Object :-

(i) Character Large Object :- It is used to store characters upto 4GB size

(ii) Binary Large Object :- It is used to store binary values of images, mp3, mp4 documents upto 4GB of size.

Constraints :- It is a rule given to a columns for validation.

Types of Constraints -

1.] Unique
2.] Not Null
3.] Check
4.] Primary Key
5.] Foreign Key

1.] Unique :- It is used to avoid duplicate values into column.

2.] Not Null :- It is used to avoid null.

3.] Check :- It is an extra validation with a condition. If condition is satisfied then value is accepted else rejected.

4.] Primary Key :- It is a constraints which is used to identify a record uniquely from the table.

→ Characteristics of Primary Key :-

- We can have only one primary key in a table.
- Primary key can't accept duplicate or repeated value.
- Primary key can't accept null.
- Primary key is always a combination of unique and not null constraints.

5.] Foreign Key :- It is used to establish the connection between the table.

→ Characteristics of Foreign Key :-

- We can have multiple foreign key in a table.
- Foreign key can accept duplicate or repeated value.
- Foreign key can accept null.
- Foreign key is not a combination of unique and not null constraints.
- For an attribute to become a foreign key. It is mandatory that it must be a primary key in it's own table.

Primary key Unique Not Null S.No Number(3)	Not Null Name Varchar(15)	Check(Length(Pancard) =10) Unique Not Null Pancard Char(10)	Check(Length(phone=10)) Unique Not Null phone Number(10)
121	'Himanshu'	ABCD1234G0	9012345670
122	'Karan'	ADEFG45178	9123456701
123	'Arjun'	BCDEH12345	9113457001

Ex- For foreign key

Primary key → Foreign key ←

Emp id	Name	Salary	D.No	C.No
1	A	12000	10	101
2	B	14000	20	102
3	C	16000	10	101
4	D	18000	20	102
5	E	20000	30	

Child

Primary key

D.No	DName	Loc
10	D1	L1
20	D2	L2
30	D3	L3

Parent

Primary key

C.No	CName	Address
101	C1	A1
102	C2	A2

Parent

Quest- What is the difference between primary key and foreign key?

Ans-

Primary Key	Foreign Key
① Column value can never be null	① Column can accept a null value
② A table can have only one primary key.	② A table can have more than one foreign key.
③ It is a unique attribute, therefore, it can't store duplicate values in relation	③ It can store duplicate values in the foreign key column
④ It can't create a parent child relationship in a table.	④ It can make a parent-child relationship in a table
⑤ If values can't be deleted from the parent table	⑤ Its value can be deleted from the child table

#NULL:- Null is a keyword which is used to represent nothing or empty cell.

- Null doesn't represent 0 or space.
- Any operation performed on a null will result is null.
- NULL doesn't occupy any memory.

#Overview of SQL Statements:-

1.] Data Definition Language (DDL)
2.] Data Manipulation Language (DML)
3.] Transition Contol Language (TCL)
4.] Data Control Language (DCL)
5.] Data Query Language (DQL)

#Data Query Language:- DQL is used to retrive the data from the data list. It has four statements:-

1.] SELECT
2.] PROJECTION
3.] SELECTION
4.] JOIN

1.] SELECT:- It is used to retrive the data from the table and display it.

2.] PROJECTION:- It is a process of retrieving a data by selecting only the columns.

- In projection all the records or values present in a particular column are by default selected.

Syntax - Select * / [distinct] columnName / expression aliasName from TableName;

Order of Execution -

(i) From clause
(ii) Select Clause

Qus- WAQTD of all the employee name

Ans- Select ename
From emp;

Note:- ① From clause start the execution.
② For from clause we can pass table name as a argument.
③ The job of from clause is to go to the database and search for the table and put the table under execution.
④ Select clause will executed after the execution of from clause.
⑤ For select clause we can pass three arguments
→ *
→ Column Name
→ Expression
⑥ The job of select clause is to go to the table under execution and select the column mention.
⑦ Select clause is responsible for preparing the result table.

Asterisk (*) :- It means to select all the column from the table.

Semicolon (;) :- It means end of the query.

To set a page dimension :-

Set pages 100 lines 100;

Qus- WAQTD salary and commission of all the employees

Ans- Select sal, comm
From emp;

Ques:- WAP to display name of the employee along with their date of joining.

Ans- Select ename, hiredate
from emp;

Ques:- WAP to display employee id and department number of all the employees in emp table.

Ans- Select empno, deptno
from emp;

Ques:- WAP to name and designation of all the employee.

Ans- Select ename, job
from emp;

Ques:- WAP to dname and location present in dept table.

Ans- Select dname, loc
from dept;

→ Distinct Clause:- It is used to remove the duplicate or repeated values from the present table.

- Distinct clause has to be used as the first argument to select clause
- We can use multiple columns as an argument to a distinct clause.
- It will remove the combination of columns in which records are duplicated.

Ex- Select distinct job
from emp;

→ Expression:- A statement which gives result is known as expression.

- Expression is a combination of operands & operators

→ Operands:- These are the values, we will pass

→ Operators:- These are the symbols which perform some operation on the operands.

Qus:- WAQTD name and annual salary of the employee.
Ans- Select ename, sal*12
from emp;
OR
Select ename, sal*12 Annual_salary
from emp;
Alias Name

Qus- WAQTD all the details of the employee along with annual salary.
Ans- Select emp.*, sal*12 Annual_salary
from emp;
OR
Select emp.*, sal*12 as "annual salary"
from emp;

Qus- WAQTD employee name & there annual salary with 20% hike.
Ans- Select ename, sal*12 + sal*12/5 Hike_salary
from emp;

Qus:- WAQTD employee name & salary of all employees with 30 deduction.
Ans- Select ename, sal - sal/100 *30 deduct_salary
from emp;

→ Alias Name:- It is an alternate name given to a column or an expression in the result table we can assign name with or without using 'as' keywood.
- Alias name have to be a single string which is seprated by an underscore(_) or enclosed with in a double quotes(" ").

3.) SELECTION:- It is a process of retreiving the data by selecting rows and columns is known as selection.
Syntax - Select * / [Distinct] columnName / Expression AliasName
from TableName
where <if/for condition>

Order of execution-

1.] from clause

2.] where clause

3.] Select clause

→ Where clause:- It is use to filter the record.

Quest- WAQTD employees name whose is getting salary more than 1000.

Ans. Select ename
from emp
where sal > 1000;

Ques- WAQTD name & salary of the employees working in deptment 10.

Ans Select ename, sal
from emp
where deptno = 10;

Quest- WAQTD name & hiredate of an employe hired on 9-Jun-1981.

Ans- Select ename, hiredate
from emp
where hiredate = '9-Jun-81';

Quest- WAQTD details of the employee whose name is miller.

Ans- Select *
from emp
where ename = 'MILLER';

Quest- WAQTD details of the employee hired after 1-Jan-1982

Ans- Select *
from emp
where hiredate > '1-Jan-82';

Quest- WAQTD name, salary & hiredate of the employees who were hired before 1985.

Ans Select ename, sal, hiredate
from emp
where hiredate < '1-Jan-85';

Ques:- WAQTD name of the employee how was hired on Valentien day 2020.

Ans Select ename
from emp
where hiredate = '14-Feb-20';

Operator in SQLF

1) Arithmetic Operator → (+, -, /, *)
2) Concatination Operator → (||)
3) Comparsion Operator → (=, !=)
4) Relational Operator → (<, >, <=, >=)
5) Logical Operator → (OR, AND, NOT)
6) Special Operator
   (i) IN
   (ii) NOT IN
   (iii) BETWEEN
   (iv) NOT BETWEEN
   (v) IS
   (vi) IS NOT
   (vii) LIKE
   (viii) NOT LIKE
7) Sub Query Operator
   (i) ALL
   (ii) ANY
   (iii) EXISTS
   (iv) NOT EXISTS

→ Concatination Operator:- It is used to join string.

Que- WAQTD employee name, salary, deptno who are working as manager.

Ans Select 'Mr' || ename, sal, deptno
from emp
where job = 'MANAGER';

→ Not equal Operator (!=):-

Qus- WAQTD all details of employes who are not working in deptno 10.

Ans- Select *
from emp
where deptno != 10;

Qus- WAQTD all details who are getting salary 3000 or less than 3000.

Ans- Select *
from emp
where sal <= 3000;

Qus- WAQTD all details who are hired 81 and more than 81 year.

Ans- Select *
from emp
where hiredate >= '01-Jan-81';

→ Logical Operator :- We use logical operator to write multiple conditions.

Qus- WAQTD name and deptno along with job or the employee working in deptno 10 and working as manager.

Ans Select ename, deptno, job
from emp
where deptno=10 and job='MANAGER';

Qus- WAQTD name, deptno, salary of the employee working in deptno 20 and earning less than 3000.

Ans- Select ename, deptno, sal
from emp
where deptno=20 and sal<3000;

Qus- WAQTD name and salary of employee if employee earn more than 1250 but less than 3000.

Ans- Select ename, sal
from emp
where sal > 1250 and sal < 3000;

Qus- WAQTD name and deptno of the employee if they work in deptno 10 or 20.

Ans- Select ename, deptno
from emp
where deptno = 10 or deptno = 20;

Qus- WAQTD name and salary and deptno of the employees if employee get more than 1250 Rupess but less than 4000 and work in deptno 20.

Ans- Select ename, sal, deptno
from emp
where sal > 1250 and sal < 4000 and deptno = 20;

Qus- WAQTD name, job and deptno of the employees working as clerk or manager in deptno 10.

Ans- Select ename, job, deptno -
from emp
where (job = 'CLEARK' or job = 'MANAGER') and deptno = 10;

→ Special Operators

(i) IN operators It is a multivalued operator which can accept multiple values at the right hand side.

Syntax - Column Name/Express IN (multiple value)

Qus- WAQTD name & deptno of the employees working in deptno 10 or 30.

Ans- Select ename, deptno
from emp
where deptno in (10, 30);

Qus- WAQTD name & job of the employee working as a clerk or manager or salesman.

Ans- Select ename, job
from emp
where job in ('CLEARK', 'MANAGER', SALESMAN');

Qus- WAQTD employee number, ename & sal of the employee who's empno is 7902 or 7839 & getting salary more than 2900.

Ans- Select empno, ename, sal
from emp
where empno in (7902, 7839) and sal > 2900;

(ii) NOT IN Operator :- It is a multi valued operator which can accept multiple values at the right hand side. It is similar to IN operator instead of selecting is rejecting the value

Qus- WAQTD name & deptno of all employee except the employee working in deptno 10 or 40.

Ans- Select ename, deptno
from emp
where deptno not in (10, 40);

Qus- WAQTD name, deptno and job of employee working in deptno 20 but not as a clerk or manager.

Ans- Select ename, deptno, job
from emp
where deptno = 20 and job not in ('CLERK', 'MANAGER');

(iii) BETWEEN Operator :- It is use whenever we have range of values [start value and stop value]

Syntax- ColumnName BETWEEN lower value and higher value

Qus- WAQTD name & salary of the employees if the employee is earning salary in the range 1000 to 3000.

Ans- Select ename, sal
from emp
where sal between 1000 and 3000.

Qus- WAQTD name and deptno of the employee working in deptno 10 and hired during 1987.

Ans
```
Select ename, deptno
from emp
where deptno=10 and hiredate between '01-Jan-87' and '31-DEC-8
```

Qus- WAQTD name, sal and hiredate of the employees hired during to 2017 into deptno 20 with a salary greater than 2000

Ans
```
Select ename, sal, hiredate
from emp
where deptno=20 and sal > 2000 and
      hiredate between '01-Jan-17' and '31-DEC-17';
```

Note:- BETWEEN operator works including the range.

(iv) NOT BETWEEN Operator:- It is opposite of BETWEEN.

Syntax- ColumnName NOT BETWEEN lower and higher value

Qus- WAQTD name, salary and hiredate of the employees who were not hired during 17 into deptno 20 with a salary greater than 2000.

Ans.
```
Select ename, sal, hiredate
from emp
where deptno=20 and sal > 2000 and
      hiredate not between '1-Jan-17' and '31-DEC-17';
```

(v) IS operator:- It is used to compare only null.

Syntax- ColumnName is NULL

Qus- WAQTD name of the employee who is not getting salary.

Ans-
```
Select ename
from emp
where sal is null;
```

Qus- WAQTD name of the employee who doesn't get comm.

Ans- Select ename
from emp
where comm is null;

Qus- WAQTD name, salary and commission of the employee if the employee doesn't earned both

Ans- Select ename, sal, comm
from emp
where sal is null and comm is null;

(vi) IS NOT operator — It is used to compare the values with not null.

Qus- WAQTD name of the employee who gets commission.

Ans- Select ename
from emp
where comm is not null;

Qus- WAQTD name, sal and comm of the employee if the employee doesnot earn comission but gets salary.

Ans- Select ename, sal, comm
from emp
where comm is null and sal is not null;

(viii) LIKE Operator — It is used for pattern matching. To achieve pattern matching we use special character

① Percentile (%)
② Underscore (_)

Syntax- ColumnName LIKE 'pattern';

Qus- WAQTD details of employee who name start with S.

Ans. Select *
from emp
where ename like 'S%';

Qus- WAQTD details of the employee who's name end with S.

Ans-
```
Select *
from emp
where ename like '%S';
```

Qus- WAQTD names of the employee who have character S in there name.

Ans-
```
Select ename
from emp
where ename like '%S%';
```

Qus- WAQTD name that starts with J and end with S.

Ans-
```
Select ename
from emp
where ename like 'J%S';
```

Qus- WAQTD name of the employees if the employees has character 'A' as his second character.

Ans-
```
Select ename
from emp
where ename like '_A%';
```

Qus- WAQTD name of the employee if the employee has character A present at least two times

Ans-
```
Select ename
from emp
where ename like '%A%A%';
```

Qus- WAQTD names, sal of the employees if the employees salary last two digit is 50.

Ans-
```
Select ename, sal
from emp
where sal like '%50';
```

Qus- WAQTD name and hiredate of the emp hired in November.

Ans
```
Select ename, hiredate
from emp
where hiredate like '%NOV%';
```

(viii) NOT LIKE Operator:- It is opposite of LIKE operator

Qus- WAQTD name and hiredate of the employees who are not hired in Jan.

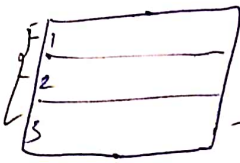
Ans- Select ename, hiredate
from emp
where hiredate like '%JAN%';

Functions:- Function are the block of code or list of instruction which are used to perform a specific task.

There are three main components of a function-
(i) Function Name (ii) Number of arguments (iii) Return type

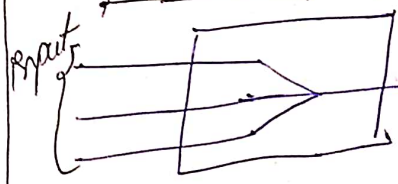
Type of function in SQL-

1.) Single Row function



Syntax- SRF(columnName/Exp)
Ex- Select length(ename)
from emp

2.) Multi Row function/Group function/Aggregate function



Syntax- MRF(ColumnName/Exp)
Ex- Select max(sal)
from emp;

→ Multi Row Function:- It takes all the input at one shot and then execution & provides a single output.

• If we pass 'n' number of inputs to a MRF() it return one output.

* List of MRF()-

① MAX()- It is used to obtain the maximum value present in the column.

(2) MIN():- It is used to obtain the minimum value present in the column.

(3) SUM():- It is used to obtain the summation of value present in the column.

(4) AVG():- It is used to obtain the average of values present in the column.

(5) COUNT():- It is used to obtain the number of values present in the column.

Note:- MRF can accept only one argument that is column name or and expression.

Syntax- MRF(columnName(expression))

- Along with MRF we are not suppose to use any other column name in the select clause.
- MRF ignore the null.
- We can't use a MRF in where clause.
- Count is the only MRF which can accept * as an argument.

Ques:- WAQTD maximum salary given to a manager.

Ans
```
Select max(sal)
  from emp
  where job = 'MANAGER';
```

Ques:- WAQTD total salary given to deptno 20

Ans
```
Select sum(sal)
  from emp
  where deptno = 20;
```

Ques- WAQTD no of employee earning more than 1500 in deptno 20

Ans
```
Select count(ename)
  from emp
  where deptno = 20 and sal > 1500;
```

Ques- WAQTD no. of employee having E in their name.
Ans- Select count(ename)
from emp
where ename like '%E%';

Ques- WAQTD minimum salary given to the employee working as clerk in deptno 10 or 20.
Ans- Select min(sal)
from emp
where job = 'CLERK' and deptno in (10, 20);

Ques- WAQTD no. of employee hired after 1982 & before 1985 in deptno 10 or 30.
Ans Select count(*)
from emp
where hiredate between '1-Jan-83' and '31-DEC-84' and deptno in (10,30);

Ques- WAQTD no. of employee getting commission.
Ans- Select count(comm)
from emp;

Ques- WAQTD max sal given to employee if the employee has character S in the name and works as a manager in dept no 10 with as salary of more than 1800.
Ans- Select max(sal)
from emp
where ename = '%S%' and job = 'MANAGER' and deptno = 10 and sal > 1800;

Grouping:- Group by clause. It is used to group the records.

Syntax- Select group by function name expression/Group
from table name
where <filter-condition>
Group by <Group expression/column name);

Order of Execution -

1. from clause
2. where clause
3. group clause
4. Select clause

Qus- WAQTD no. of employee working in each department
Ans- Select count(*) from emp group by deptno;

OR

Select deptno, count(*) from emp group by deptno;

Note:- Group by clause is used to group the records

- Group by clause execute row by row.
- After the execution of group by clause we get groups.
- Therefore any clause that execute after group by must execute group by group.
- The column name or expression used for grouping can we used in select clause.
- Group by clause can we used without using where clause.

Qus- WAQTD no. of employee working in each department except the employee working as analyst.

Ans-
```
Select deptno, count(*)
from emp
where job = 'ANALYST'
group by deptno;
```

Qus- WAQTD max sal given to each job

Ans-
```
Select job, max(sal)
from emp
group by job;
```

Ques- WAQTD no. of emp working in each job if the employee having 'A' in name.

Ans- Select job, count(ename)
from emp
where ename like '%A%'
group by job;

Ques- WAQTD no of emp getting commission in each dept.

Ans Select comm, count(*)
from emp
group by comm;
OR
Select deptno, count(comm)
from emp
group by deptno;

Filtering :- Having clause. It is used to filter the group.

Syntax- Select group by expression
from table name
where (filter condition)
group by (Group expression / column name)
having (Group, filter condition);

Ques:- WAQTD to find number of employee working in each dept if there are at least 5 employee in each dept.

Ans. Select deptno, count(*)
from emp
group by deptno
having count(*) >=5;

Order of Execution -

1. From clause
2. where clause (Row by Row)
3. Group clause (Row by Row)
4. having clause (Group by Group)
5. Select Clause (Group by Group)

Ques - WAQTD designation in which there are atleast 2 employee present.

Ans
```
Select job, count(job)
from emp
group by job
having count(job) >= 2;
```

Ques - WAQTD the name that are repeated.

Ans
```
Select ename
from emp
group by ename
having count(*) > 1;
```

Ques - WAQTS name that are repeated exactly twice.

Ans
```
Select ename
from emp
group by ename
having count(*) = 2;
```

Ques - WAQTS the salary that is repeated.

Ans
```
Select sal
from emp
group by sal
having count(*) > ;
```

Qus- WAQTD number of employee working in each department having at least 2 employee, having character A or S in their name.

Ans- Select count(*), deptno
from emp
where ename like '%A%' or ename like '%S%'
group by deptno
having count(*) >= 2;

Qus- WAQTD job and total salary of each job if the total salary of each job is greater than 3450.

Ans- Select job, sum(sal)
from emp
group by job
having sum(sal) > 3450;

Qus- WAQTD job and total salary of the employee if the employee are earning more than 1500.

Ans- Select job, sum(sal)
from emp
where sal > 1500
group by job;

Qus- WAQTD number of employee earning salary is more than 1200 in each job and the total salary need to pay employee of each job must exceed 3800.

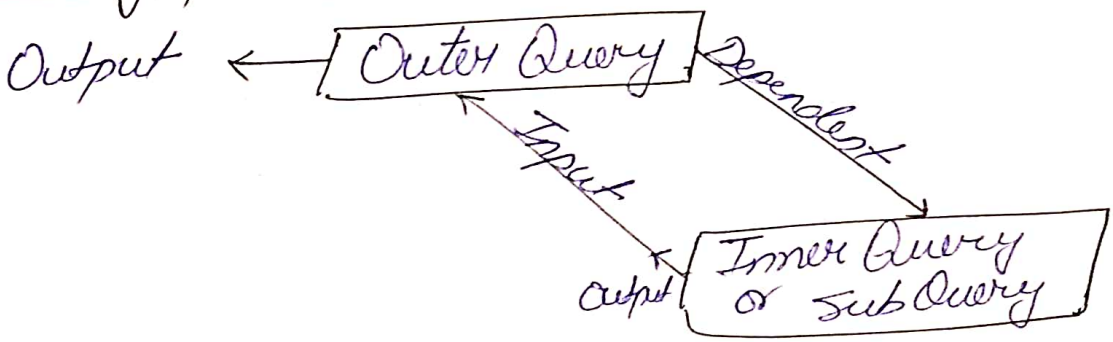
Ans- Select job, count(*), sum(sal)
from emp
where sal > 1200
group by job
having sum(sal) > 3800;

Difference between Where Clause and Having clause

Where Clause	Having Clause
① It is used to filter the record	① It is used to filter the group
② Where clause is execute row by row	② Having clause execute group by group
③ We can't use MRF	③ We can use MRF
④ Where clause execute before group by clause	④ Having clause execute after group by clause.

SubQuery :- A query written inside another query is known as subquery.

Working_principle -



Let us consider two queries, outer query and inner query -

1.) Inner query or sub query execute first and produce an output.
2.) The output of inner query is given as an input to outer query.
3.) The outer query generates the result.
4.) Therefore, we can state that the outer query is dependent on inner query and this is the execution principle of sub query.

→ While or when do we use sub query

Case 1 - Whenever, we have unknown present in the question we use sub query to find unknown.

Ques - WAQTD names of the employees earning less than MILLER.

Ans - Select ename
from emp
where sal<(Select sal
from emp
where ename='MILLER');

Ques - WAQTD names and deptno of the employees working in the same dept as 'SMITH'.

Ans - Select ename, deptno
from emp
where deptno=(Select deptno
from emp
where ename='SMITH');

Ques - WAQTD name and hiredate of the employees if the employees was hired after 'Jones'.

Ans - Select ename, hiredate
from emp
where hiredate > (Select hiredate
from emp
where ename='JONES');

Ques - WAQTD all the details of the employee working in the same designation as 'KING'.

Ans - Select *
from emp
where job=(Select job
from emp
where ename='KING');

Qus- WAQTD name, sal, deptno of the employee if the employees earn more than 2000 and worked in the same dept as 'James'.

Ans
```
Select ename, sal, deptno
from emp
where sal > 2000 and deptno = (Select deptno
                               from emp
                               where ename = 'JAMES');
```

Qus- WAQTD names of the employees working in the same dept as a 'James' and earning salary is more than 'Adam' and working in the same job as 'Miller'.

Ans
```
Select ename
from emp
where deptno = (Select deptno
                from emp
                where ename = 'JAMES') and sal > (Select sal
                                                  from emp
                                                  where ename = 'ADAM')
and job = (Select job
           from emp
           where ename = 'MILLER');
```

Case 2:- When ever the data to be selected and the condition to be executed are present in different table be used as subquery.

Qus- WAQTD dname of the employee who's name is miller.

Ans
```
Select dname
from dept
where deptno = (Select deptno
                from emp
                where ename = 'MILLER');
```

Qus- WAQTD location of ADAMS.

Ans- Select loc
from dept
where deptno = (Select deptno
from emp
where ename = 'ADAMS');

Qus- WAQTD names of the employees working in location Boston.

Ans. Select ename
from emp
where deptno = (Select deptno
from dept
where loc = 'BOSTON');

Qus- WAQTD no. of employees working in dept sales

Ans- Select count(*)
from emp
where deptno = (Select deptno
from dept
where dname = 'SALES');

Qus- WAQTD ename, salary of all the employee earning more than SCOTT and working in dept20.

Ans- Select ename, sal
from emp
where deptno=20 and sal > (Select sal
from emp
where ename='SCOTT');

Qus- WAQTD all the details of the employee working as a manager in dept ACCOUNTING.

Ans- Select *
from emp
where job='MANAGER' and deptno= (Select deptno
from dept
where dname='ACCOUNTING');

Qus- WAQTD details of the employee working in the same designation as MILLER and work in location New York.

Ans. Select *
from emp
where job = (Select job
from emp
where ename = 'MILLER') and
deptno = (select deptno
from dept
where loc = 'NEW YORK');

Qus- WAQTD no. of employees working as a CLERK in the same deptno as SMITH and earning more than KING hired after MARTIN in the location BOSTON.

Ans Select count(*) No_of_Emp
from emp
where job = 'CLERK' and deptno = (Select deptno
from emp
where ename = 'SMITH')
and sal > (Select sal
from emp
where ename = 'KING') and hiredate > (Select hiredate
from emp
where ename = 'MARTIN')
and deptno = (Select deptno
from dept
where loc = 'BOSTON');

MAX & MIN

Qus- WAQTD name of the employee getting maximum sal.

Ans Select ename
from emp
where sal = (select max(sal)
from emp);

Qus- WAQTD name and sal of the Employee getting minimum salary.

Ans- Select ename, sal
from emp
where sal = (Select min(sal)
from emp);

→ Types of Subquery :-

1. Single Row Sub query
2. Multi Row Sub query

Qus- WAQTD deptname and Allen.

A. Select dname
from dept
where deptno = (Select deptno
from emp
where ename = 'ALLEN');

Qus- WAQTD dname of Allen and Smith

Ans- Select dname
from dept
where deptno = (Select deptno
from emp
where ename in ('ALLEN', 'SMITH');

- Here since the subquery return two reports / two values we can't use equal operator we have to use in operator

1.) Single Row SubQuery :- If the subquery returns exactly one value or report we call it as a single row sub query.

- If it return's only one value than we can use the normal operator's or the special operator to compare the value.

2.) Multi Row SubQuery :- If the subquery return more than one records we call it as multi row subquery.

- If it return more than one value than we can not use normal operators.

- We have to use only special operator's to compare the values.

Note:- It is difficult to identify whether a query belong's single or multirow. So, it is always recommended to use special operator to compare the value.

Ques- WAQTD ename and salary of the employees earning more than employee of dept 10.

Ans- Select ename, sal
from emp
where sal > (Select sal
from emp
where deptno = 10);

Here we can't use '>' symbol to compare multiple value

- We can't use in or not in as well as because it is used for equal and not equal to symbols.

Therefore, we have to use subquery operator for comparing ~~the values present at the right hand side.~~ relational operator's such (<, >, >=, <=) etc.

Sub Query Operators:-

1) ALL:- It is special operator used along with a relational operator (<, >, <=, >=) ~~along with a relational~~ to compare the values present at the right hand side.

- All operator return's true if all the value at the R.H.S have satisfied the conditions.

Ques- WAQTD name of the Emp if the employee Earn less than the employee working as salesman.

Ans- Select ename
from emp
where sal < all (Select sal
from emp
where job = 'SALESMAN');

Qus- WAQTD name of the employee if the employee has salary more than 'ADAMS'.

Ans- Select ename
from emp
where sal > all (Select sal
from emp
where ename = 'ADAMS');

2) ANY :- It is special operator used along with a relational operator to compare the value present at R.H.S

- Any operator return true if one of value at R.H.S have satisfied the condition.

Qus- WAQTD name of employee if the employee earn less than at least a salesman.

Ans- Select ename
from emp
where sal < any (Select sal
from emp
where job = 'SALESMAN');

Qus- WAQTD details of the employee working as CLERK and hired before at least a salesman.

Ans- Select *
from emp
where job = 'CLERK' and hiredate < any (Select hiredate
from emp
where job = 'SALESMAN');

Nested Sub Query :- A sub query written inside another subquery is known as nested subquery

- We can nest about 255 sub query.

Qus- WAQTD second max salary given to an employee.

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

Ques- WAQTD third max sal of the employee.

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

Ques- WAQTD fourth max sal of the employee.

Ans 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))));

Ques- WAQTD third min sal of the employee.

Ans- Select min(sal)
from emp
where sal > (select min(sal)
from emp
where sal > (select min(sal)
from emp));

Ques- WAQTD names of the employee who is getting fourth minimum salary.

Ans- 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))));

Select ename
from emp
where sal =

Employee and Manager Relation

Case 1 -

Qus - WAQTD name of ALLEN manager.

Ans
```
Select ename
from emp
where empno = (select mgr
               from emp
               where ename = 'ALLEN');
```

Qus - WAQTD smith manager manager's name.

Ans
```
Select ename
from emp
where empno = (select mgr
               from emp
               where empno = (select mgr
                              from emp
                              where ename = 'SMITH'));
```

Qus - WAQTD dname of KING manager

Ans
```
Select dname
from dept
where deptno = (select deptno
                from emp
                where empno = (select mgr
                               from emp
                               where ename = 'KING'));
```

Case 2 -

Qus - WAQTD name of the employe reporting to KING.

Ans
```
Select ename, job
from emp
where mgr = (select empno
             from emp
             where ename = 'KING');
```

Qus- WAQTD deptname of the employee reporting to president

Ans- Select dname
from dept
where deptno in (Select deptno
from emp
where mgr in (Select empno
from emp
where job='PRESIDENT'));

#Joins:- The process of retrival of data from multiples table simultaneously is known as join.

#Why/When:- Whenever the attribute is to be selected from both the tables we use joins

→ Types of Joins:-

1. Cartesian Join / Cross Join
2. Inner Join / Equi Join
3. Outer Join
   (i) Right Outer Join
   (ii) Left Outer Join
   (iii) Full Outer Join
4. ~~Self Join~~ Self join
5. Natural Join

1.) Cartesian Join / Cross Join:- In cartesian join a record from table 1 will be merged with all the record of table 2.

Emp

Empno	Ename	deptno
101	E1	10
102	E2	20
103	E3	10

dept

deptno	dname
10	D1
20	D2

Result -

EmpNo	EName	deptno	deptno	dname
101	E1	10	10	D1
101	E1	10	20	D2
102	E2	20	10	D1
102	E2	20	20	D2
103	E3	10	10	D1
103	E3	10	20	D2

- Number of column in the result table will be equivalent to the summation (Σ) of columns present in the both the tables.
  ie, No. of column = no. of col in T1 + no of col in T2
  = 3 + 2
  = 5
- Number of rows in the result table will be equivalent to the product of no. of rows present in the both the tables.
  ie, No. of rows = no of rows in T1 * no. of rows in T2
  = 3 * 2
  = 6

Syntax :-

ANSI → American National Standard institute

ANSI - Syntax :-

```
Select columnName
from Table1 CROSS JOIN Table2;
```

Oracle - Syntax -

```
Select columnName
from Table1, Table2;
```

Qus- WAQTD ename & dname for all the employees.

Ans- Select ename, dname
from emp cross join dept;
OR
Select ename, dname
from emp, dept;

2.) Inner Join:- It is used to obtain only matching records or a record which has a pair.

Join Condition - It is a condition on which the two tables are merged.

Syntax - Table 1. ColumnName = Table 2. ColumnName

Cond" → emp.deptno = dept.deptno

ANSI Syntax for Inner Join -

Select ColumnName
from Table 1 Inner Join Table 2
on <Join condition>;

Oracle syntax -

Select columnName
from Table 1, Table 2
where <Join condition>;

Qus- WAQTD ename and dname for all the employees.

Ans- Select ename, dname
from emp, dept
where emp.deptno = dept.deptno;

Qus- WAQTD ename and loc for all the employees working as manager.

Ans- Select ename, loc
from emp, dept
where emp.deptno = dept.deptno and job='MANAGER';

Qus- WAQTD ename, sal and dname of the employee working as Clerk in deptno 20 with a salary of more than 800.

Ans- Select ename, sal, dname
from emp, dept
where emp.deptno = dept.deptno and job='CLERK' and emp.deptno = 20 and sal > 800;

Qus- WAQTD ename, deptno, dname and loc of the employee earning more than 2000 in New York.

Ans Select ename, ~~deptno~~ emp.deptno, dname, loc
from emp, dept
where dept.deptno = emp.deptno and sal > 2000 and loc='NEW YORK';

3.) Outer Join:- It is used to obtain unmatched records.

(i) Left Outer Join - It is used to obtain unmatched records of left table along with matching record.

Emp

Ename	deptno
A	10
B	Null
C	20
D	Null

dept

deptno	dname
10	D1
20	D2
30	D3
40	D4

Result -

ename	deptno	deptno	dname
A	10	10	D1
C	20	20	D2
B	Null	Null	Null
D	Null	Null	Null

ANSI Syntax -

```
Select columnName
from Table1 Left Outer Join Table2
on <Join condition>;
```

Oracle Syntax -

```
Select ColumnName
from Table1, Table2
where <Table1.ColumnName = Table2.ColumnName(+)>;
```

Qus- WAQTD ename and dname of all the employees even though ~~dont~~ the employee don't work in any dept.

Ans-

```
Select ename, dname
from emp, dept
where emp.deptno = dept.deptno(+);
```

(ii) Right Outer Join :- It is used to obtain unmatched record of right table along with matching records

emp

Ename	deptno
A	10
B	Null
C	20
D	Null

dept

deptno	dname
10	D1
20	D2
30	D3
40	D4

Result-

Ename	deptno	deptno	dname
A	10	10	D1
C	20	20	D2
Null	Null	30	D3
Null	Null	40	D4

ANSI Syntax-

```
Select ColumnName
from Table1 Right Outer Join Table2
on < Join Condition >;
```

Oracle Syntax-

```
Select ColumnName
from Table1, Table2
where < Table1.ColumnName(+) = Table2.columnName >;
```

Ques- WAQTS ename and dname of the employees even though there are no employees in a dept.

Ans-

```
Select ename, dname
from emp, dept
where emp.deptno(+) = dept.deptno;
```

(iii) Full Outer Join:- It is used to obtain unmatched records of both left and right table along with matching records

emp

Ename	deptno
A	10
B	Null
C	20
D	Null

dept

deptno	dname
10	D1
20	D2
30	D3
40	D4

Result-

Ename	deptno	deptno	dname
A	10	10	D1
C	20	20	D2
B	Null	Null	Null
D	Null	Null	Null
Null	Null	30	D3
Null	Null	40	D4

ANSI Syntax-

```
Select ColumnName
from Table1 Full Outer Join Table2
on <Join Condition>;
```

Ques- WAOTD ename and dnames of all the employees even though the employees don't work in any dept and a dept having no employees.

Ans-
```
Select ename, dname
from emp full outer join dept
on emp.deptno = dept.deptno;
```

Ques- WAOTD name and salary of the employees earning more than 'James' in the dept ACCOUNTING.

Ans-
```
Select ename, sal
from emp
where sal > (select sal
             from emp
             where ename='JAMES' and
               deptno= (select deptno
                        from dept
                        where dname='ACCOUNTING'));
```

4.) Self Join:- Joining a label by it self is known as self join.

Why/When:- Whenever the data to select is in the same table but present in different record we use self join

emp E1

Empno	Ename	mgr
101	A	102
102	B	103
103	C	101

emp E2

Empno	Ename	mgr
101	A	102
102	B	103
103	C	101

Condition-

E1. mgr = E2. empno

Result →

Empno	Ename	mgr	empno	Ename	mgr
101	A	102	102	B	103
102	B	103	103	C	101
103	C	101	101	A	102

ANSI Syntax -

Select ColumnName
from Table1-T1 JOIN Table2 T2 → Aliasname
on <Join condition>;

Example- Select *
from emp e1 JOIN emp e2
on e1.mgr = e2.empno;

Oracle Syntax -

Select ColumnName
from table1 T1, table2 T2
where <T1.ColumnName = T2.ColumnName>;

Ex- Select *
from emp e1, emp e2
where e1.mgr = e2.empno;

Qus- WAQTD ename, and manager's name for all the employees.

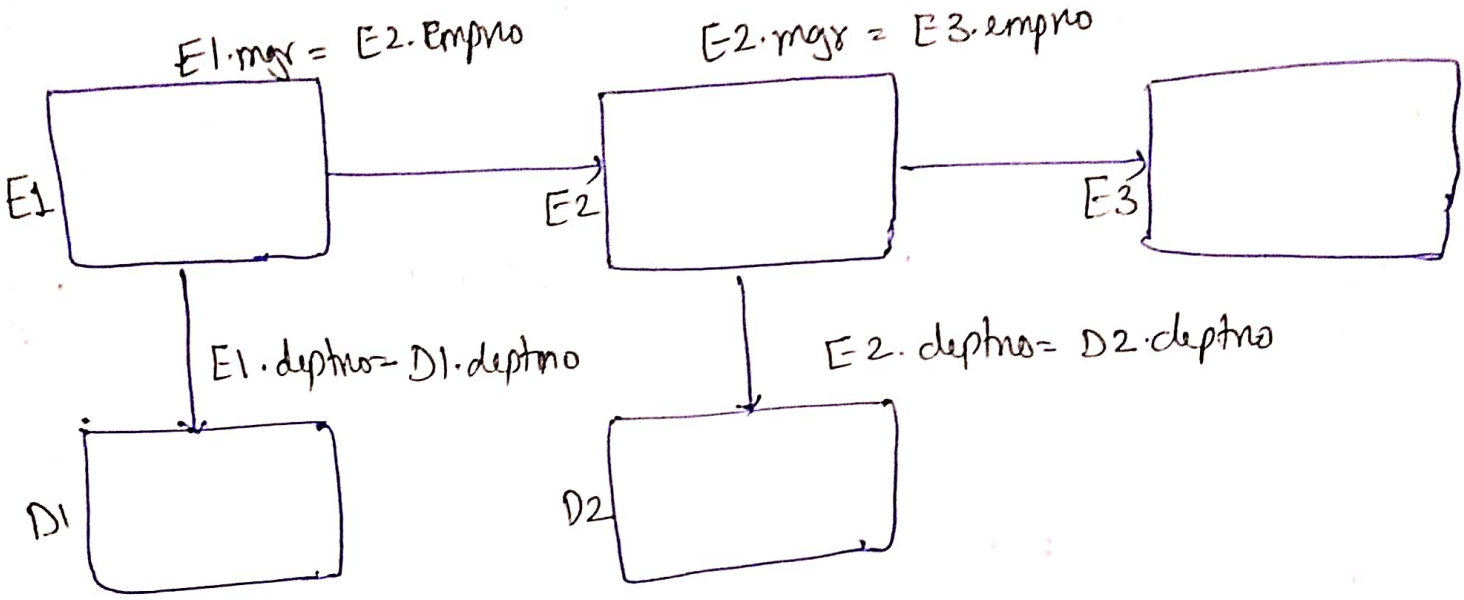
Ans- Select e1.ename, e2.ename
from emp e1, emp e2
where e1.mgr = e2.empno;

Qus- WAQTD ename, sal along with manager's name and manager salary for all the employees.

Ans- Select e1.ename, e1.sal, e2.ename, e2.sal
from emp e1, emp e2
where e1.mgr = e2.empno;

Qus- WAQTD. ename, manager name along with their deptno if employee is working as clerk.

Ans. Select e1.ename, e2.ename, e1.deptno, e2.deptno
from emp e1 JOIN emp e2
on e1.mgr = e2.empno and e1.job = 'CLERK';



Qus- WAQTD ename, manager's name and manager's manager name for all the employees.

Ans Select e1.ename employee, e2.ename manager, e3.ename mgr_manager
from emp e1, emp e2, emp e3
where e1.mgr = e2.empno and e2.mgr = e3.empno;

Qus - WAQTD ename, dname and manager name, manager deptname for all the employees.

Ans- Select e1.ename employee, d1.dname dept_name, e2.ename manager
d2.dname dept_name
from emp e1, emp e2, dept d1, dept d2
where e1.mgr = e2.empno and e2.deptno = d1.deptno
and e2.deptno = d2.deptno;

Note:- To join 'n' number of tables we need to write (n-1) number of join condition.

5.) Natural Join:- It behaves as inner join if there is a relation between the tables as cross join.

ANSI Syntax-

```
Select ColumnName
from Table1 Natural Join Table2;
```

Single Row Function:- single row function execute row by row.
- It takes an input, starts the execution and generate the output then it goes for the next input.
- If we pass 'n' number of input to a single row function it will return 'n' number of output.

→ Length:- It is used to count the number of characters present in the given string.

Syntax - length('String');

Qus- WAQTD number of characters present in SMITH.

A. select length(ename)
from emp
where ename='SMITH';

→ Dual Table:- It is a dummy table which has one column and one row which is used to display the result.

Syntax- Desc dual;

→ Concat():- It is used to join the given two strings.

Syntax- Concat('String1','String2');
from dual;

→ Lower() :- It is used to convert a given string to lower case.

Syntax - lower('string')
from dual;

→ Upper() :- It is used to convert a given string to upper case.

Syntax - upper('string')
from dual;

→ Initcap() :- It convert a given string to initial capital letter case.

Syntax - initcap('string')

→ Reverse() :- It is used to reverse a given string.

Syntax - Reverse('string')

→ Substr() :- It is used to extract a part of string from original string.

Syntax - substr('Original String', position, [length])

Note :- length is not mandatory. If length is not mentioned then consider the complete string.

Ex-
Substr('QSPIDERS', 1) = QSPIDERS
Substr('QSPIDERS', 2, 3) = SPI
Substr('QSPIDERS' 0, 5) = QSPID
Substr('QSPIDERS', 3, 3) = PID
Substr('QSPIDERS', -2, 5) = ERS

Qus- WAQTD abstract first three character of the emp name.

Ans- Select substr(ename, 1, 3)
from emp;

Ques- WAQTD abstract last three characters of the employee name.

Ans select substr(ename, -3)
from emp;

Ques- WAQTD first half characters of the employee name.

Ans select substr(ename, 1, length(ename)/2)
from emp;

→ Replace:- It is used to replace a string with another string in the original string.

Syntax- replace('original string', 'string', ['new string'])

Note:- If the third argument is not mentioned the default value of it is null.

Ex- replace('abhishek', 'a', 'h') → hbhishek
replace('malyalam', 'mal') → yalam

Ques- WAQTD number of times character A present in Malyalam.

Ans select length('MALYALAM') - length(replace('MALYALAM', 'A'))
from dual;

→ Instr() :- It is used to obtain the position in which the string is present in the original string.

- It is used to search for a string in the original string if present it returns the position else it returns zero (0).

Syntax- Instr('original string', 'string', position[, 'occurence'])

Note:- If value of occurence is not mentioned then default value of it is 1.

Ex- instr('MALYALAM', 'A', 1, 1) → 2
instr('MALYALAM', 'A', 1, 3) → 7
instr('MALYALAM', 'A', 3) → 5
instr('MALYALAM', 'YAL', 1) → 4
instr('MALYALAM', 'YAL', 1, 2) → 0

M	A	L	Y	A	L	A	M
1	2	3	4	5	6	7	8

Ques- W/AQTD name of the employees if they have character A in their name

Ans
```
select ename
from emp
where instr(ename, 'A', 1, 1) > 0;
```

Ques- W/AQTD names of the employees if they have character A present at least twice.

Ans
```
select ename
from emp
where instr(ename, 'A', 1, 2) > 0;
```

Ques- W/AQTD ename if they have character A exactly twice.

Ans
```
select ename
from emp
where instr(ename, 'A', 1, 2) > 0 and instr(ename, 'A', 1, 3) = 0;
```

→ MOD:- It is used to obtain modulus or remainder of given number.

Syntax- mod(m, n)

Ques- W/AQTD employee name and empno who have odd employee no.

Ans
```
select ename, empno
from emp
where mod(empno, 2) = 1;
```

Ques- W/AQTD name and sal of employee who earn sal in multiple of 50.

Ans
```
select ename, sal
from emp
where mod(sal, 50) = 0;
```

→ Round() :- It is used to round off the given number based on the scrolled value.

Syntax- round(number[,'scale'])

Ex- Round(7.5) ⇒ 8
Round(7.4) ⇒ 7
Round(6.9) ⇒ 7
Round(7.4, 0) ⇒ 7

- The default value of scale is 0.

Left Side	Right Side
Before the decimal (-ive)	After the decimal (+ve)

- When the scale is negative it indicate the digits before the decimal and the digit count begins from 1.

Ex- Round(7390, -1) ⇒ 7390
Round(7359, -1) ⇒ 7350
Round(796, -1) ⇒ 7970
Round(659, -2) ⇒ 6500
Round(8976.123, -4) ⇒ 90000

- When the scale is positive it indicate the digits after the decimal and digit count begins from the 0.

Ex- Round(7390.12, 0) ⇒ 7390
Round(7390.972, 2) ⇒ 7390.97
Round(123.659, 1) ⇒ 123.7
Round(702.1239, 3) ⇒ 702.124

→ Trunc() :- It is similar to round but it always round off the given number to the lower value.

Syntax- Trunc(Number, [scale]);

Ex- trunc(456749.23, -4) ⇒ 450000

→ Month_Between :- It is used to obtain the number of months present between the given two dates.

Syntax- months_between (date1, date2)

Notes:- Sysdate :- It is used to obtain today date

Ex- Select sysdate
From dual;

current_date :- It is also used to obtain today's date

Ex- Select current_date
From dual;

Systimestamp :- It is used to obtain time, date and time zone.

Ex- Select systimestamp
From dual;

Example- Select (months_between(sysdate, hiredate)) || 'months'
From emp
where ename = 'SMITH';

→ Last_day() :- It is used to obtain the last day in the given day.

Syntax- last_day(date)

Ex- Select last_day(sysdate)
From dual;

Ex- Select last_day('12-Feb-2023')
From dual;

→ To_char() :- It is used to convert the given date into string based on the given format model.

Syntax- To_char(date, 'format_model')

Ex- Select To_char(sysdate, 'HH24')
From dual;

Ex-21 Select To_char (sysdate, 'DD-MM-YY')
from dual;

Ques- WAQTD details of the employees who was hired on monday.
Ans- select *
from emp
where to_char(hiredate, 'DAY') = 'MONDAY';

Ques- WAQTD details of the employee hired on wednesday at 10AM.
Ans- Select *
from emp
where to_char(hiredate, 'DAY') = 'WEDNESDAY' and
to_char (hiredate, "HH24') = 10;

NVL (Null Value Logic) :- NVL is used to eleminate the side effect of using null in arithmetic operations.

Syntax- NVL (Argument1, Argument2)

- Argument1 :- Here write any column or expression which can result in null.
- Argument2 :- Here we write a numeric value which will be substituted.

Ques- WAQTD total salary of the employee.
Ans- Select ename, sal + nvl(comm, 0)
from emp;

Order by clause :- It is used to sort the records either in descending order or in descending order.

Note :- By default its ascending order.

Ex- Select *
from emp
order by sal;

Ex- Select *
From emp
order by sal desc;

Qust- WAQTD total details of the employee and display it in ascending order by employee name.

Ans Select *
From emp
order by ename asc;

Qust WAQTD ename, sal and display it in descending order by salary.

Ans- Select ename, sal
From emp
order by sal desc;

Qust WAQTD details of the employees in ascending order by department no, salary.

Ans Select *
From emp
order by deptno, sal;

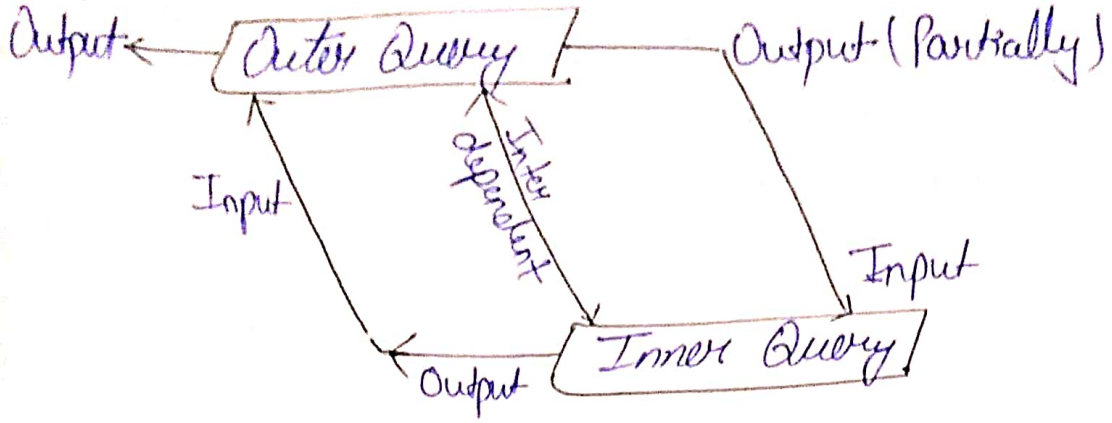
OR

Select *
From emp
Order by deptno → ascending order, sal desc; → descending order

Co-Related Subquery: A query written inside another query such that the outer query and the inner query are dependent on each other. This is known as co-related subquery.

Let's, consider two queries inner & outer query respectively.

Working Principle



1.) Outer query execute first but partially.
2.) The partially executed output is given as an input to the inner query.
3.) The inner query execute completely and generates an output.
4.) The output of inner query is given as an input to the outer query and outer query produce the result/ output.
5.) Therefore, we can state that the outer query and the inner query both are inter dependent.

Note:- • In co-related subquery a join condition is a must and must be written only in the inner query.
• Co-related subquery works with the principle of both subquery and joins.

Q. Write a query to display dname in which there are employee working.

Ans.
```
select dname
from dept d
where d.deptno in (select deptno
                   from empe
                   where d.deptno = e.deptno);
```

Ques- WAQTD dname in which there are no employee working.

Ans- Select dname
from dept d
where d.deptno not in (Select deptno
from emp e
where d.deptno = e.deptno);

Exists and Not Exists Operator

→ Exists- Exists operator is a unary operator which can accept one operand towards RHS and that operand has to be a co-related subquery.

• Exists operator returns true if the subquery return any value other than null.

Ex- Select dname
from dept d
where exists (select deptno
from emp e
where d.deptno = e.deptno);

→ Not Exists- Not Exists operator return true if the subquery return null.

Ex- Select dname
from dept d
where not exists (select deptno
from emp e
where d.deptno = e.deptno);

To Find nth maximum and minimum value

Q- Select *
from emp e1
where (select count(distinct sal)
from emp e2
where e2.sal > e1.sal) = N-1;

Que- WAQTD ename and sal who is getting 3rd minimum salary.

Ans- Select ename, sal
from emp e1
where (select count(distinct sal)
from emp e2
where e2.sal < e1.sal) = 2;

Que- WAQTD ename, sal who are getting 2nd, 4th, 5th and 7th max salary.

Ans- Select ename, sal
from emp e1
where (select count(distinct sal)
from emp e2
where empe2 > empe1) in (1,3,4,6);

DDL (Data Definition Language) :- DDL is used to construct and object in the database and deals with the structure of object.

- It has five statements are -

1.) Create
2.) Rename
3.) Alter
4.) Truncate
5.) Drop

1.) **Create:-** It is used to build or construct an object.
- Object or entity can be a table or a view (Virtual table)

- How to create a table-

(i) Name of the table → tables cannot have same name
(ii) Number of columns
(iii) Names of columns
(iv) Assign data type for the column
(v) Assign constraints [not mandatory]

Syntax-
```
Create table tableName(
    ColumnName1 datatype(size),
    ColumnName2 data type(size),
        :
    ColumnName n data type(size));
```

Note:- desc tablename; → describe the table details.

Ex-
```
Create table first(Sr-no number(3) primary key,
    name varchar(15) not null,
    phone number(10) not null check(length(phone)=10),
    address varchar(20),
    addhaar card. number(12),
    deptno number(3);
Constraints deptno_fk foreign key(deptno) reference dept1(deptno);
```

2.) **Rename:-** It is used to change the name of the object (table).

Syntax- Rename tablename to New Name;

3.) Alter :- It is used to modify the structure of the table.

→ To add a column :-

Syntax- Alter table tableName
add columnName datatype(size) constraints;

→ To drop a column :-

Syntax- Alter table tableName
drop column ColumnName;

→ To rename a columnName :-

Syntax- Alter table tableName
rename column ColumnName to newColumnName;

→ To modify the data type :-

Syntax- Alter table tableName
modify ColumnName newdatatype;

→ To modify not null constraints :-

Syntax- Alter table tableName
modify columnName existingdatatype [NULL]/NOTNULL;

4.) Truncate :- It is used to remove all the record from the table permanently.

Syntax- Truncate table tablename;

5.) Drop :- It is used to remove the table from the database.

Syntax- DROP table tablename;

→ To recover the table :-

Syntax- flashback Table tablename
to before DROP;

→ To delete the table from bin folder:-

Syntax- PURGE Table tablename;

Note:- DDL statement are auto commit statements.

Data Manipulation Language (DML):- It is used to manipulate the object by performing insertion, updating and deletion.

1.) Insert:- It is used to insert or create records in the table.

Syntax- INSERT INTO TableName values ($V_1, V_2, V_3 \dots V_n$);

2.) Update:- It is used to modify and existing values.

Syntax- Update TableName
SET column1 = value, column2 = value ...
[where statement];

Qus- WAQT Update department no. of rohan kumar to 10.

Ans- Update student
SET deptno = 10
where name = 'ROHAN KUMAR';

Qus- WAQT Update address and contact number of mohan kumar to Banglore, 8171000000 respectively.

Ans Update student
SET address = 'Banglore', contact = 8171000000
where name = 'MOHAN KUMAR';

3.) Delete:- It is used to remove a particular record from the table.

Syntax- Delete from tableName
[where statement];

Qus- WAQT Delete Rohan kumar from the tabele.
Ans Delete from student
where name = 'ROHAN KUMAR';

Transition Control Language (TCL) :- It is used to control the transition done on the database.

- The DML operation performed on the database are known as transition. such as insertion, updating and deletion.
- It has three statements:-

1.) Commit
2.) Roll back
3.) Save Point

1.) Commit :- This statement is used to save the transition into the database.

Syntax- commit;

2.) Roll back :- This statement is used to obtain only the saved data from the database.

- It will bring you to the point where you have committed for the last time.

Syntax :- roll back;

3.) Save Point :- This statement is used to mark the position or restoration point.

Syntax- Savepoint savepointName;

Data Control Language (DCL) :- This statement is used to control the flow of data between the users. It has two statement

1.) Grant
2.) Revoke

1.) Grant:- This statement is used to give permission to a user.

Syntax- GRANT SQL Statement
ON tableName
to User;

Ex-

Connect
User-name : SCOTT
Password : tiger
Connected.

Grant select
on Emp
to HR;
Grant succeeded.

Connect
Username : HR
Password : tiger
Connected

Select *
from SCOTT.Emp;

!Show user;
User is HR

Command to show the user.

2.) Revoke:- This statement is used take back the permission from the user.

Syntax- Revoke SQL Statement
ON table name
from user;

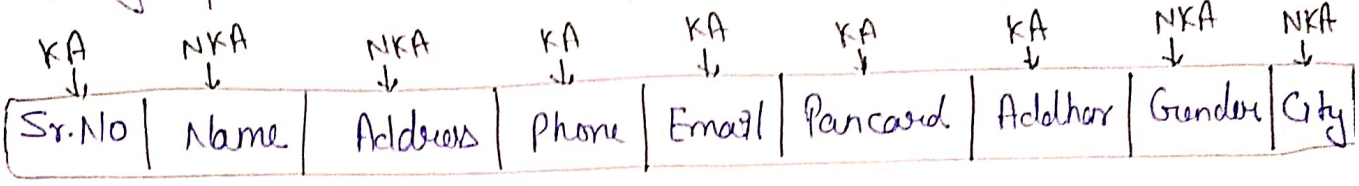
Keys:- Keys is a attribute or set of attribute which is used uniquely identifies a records from the table.

- We have different type of keys

1.) Keys attribute or Candidate key
2.) Non-key attribute

3.) Prime key attribute or Primary key
4.) Non-prime key attribute
5.) Composite key attribute
6.) Super key attribute
7.) Foreign key attribute

1.) Key Attribute or Candidate key :- An attribute that is used to identify a records uniquely from the tabel is known as a key attribute (KA)



2.) Non-key attribute :- All the attribute except the key attribute is known as non-key attribute (NKA)

3.) Prime key attribute or Primary key :- Among with key attribute and attribute choosen to be the main attribute to identify the records uniquely from the table is known as prime key attribute.

4.) Non-prime key attribute :- All the key attribute or candidate key except the prime key attribute is known as non-prime key attribute.

5.) Composite key attribute :- It is a combination of two or more non-key attribute which is used to identify a record uniquely from the table.

6.) Super key attribute :- It is a set of all the key attribute.

7.) Foreign key attribute :- It behaves like an attribute of another entity that is used to represent the relationship.

Functional Dependencies:- Let us consider a relation of two attribute X and Y respectively.

- For example- let us consider that X determines Y or we can also say that Y is dependent of X.

Ex-

KA

X	Y
1	A
2	A

Types of functional dependencies:-

1.) Total Functional dependencies - If all the attributes in the relation are determined by a single attribute which is a key attribute and then there exist total functional dependencies.

- In total functional dependencies we don't have redundancy, Anomaly.

Redundancy:- The unwanted repeation of data present in the table is known as redundancy

Anomaly:- The side effect that occurid during DML operation is known as Anomaly.

Ex- update emp
set deptno = 50
where deptno = 10; } Update Anomaly

Ex- delete ~~emp~~ from dept
where deptno = 10; } delete Anomaly

Example- Let us consider a relation of four attribute A, B, C and D where A is the key attribute. then A determined B, A determined C and A determine D.

ie.. | A | B | C | D |

$A \rightarrow B$

$A \rightarrow C$

$A \rightarrow D$

$A \rightarrow \{B, C, D\}$

Therefore, there exist total functional dependencies.

2.) Partial functional dependencies :- For partial functional dependencies to exist there must be a composite key relation.

- One of the key attribute in the composite key relation determined another attribute seperatly and then there exist partial functional dependencies.
- In partial functional dependencies we have redundancy and anomaly.

Ex- Let us consider a relation with four attributes A, B, C and D in which A and B are the composite key attributes.

A	B	C	D

$AB \rightarrow C$
$AB \rightarrow D$
$B \rightarrow D$ → Partial functional dependency

Therefore, there exist partial functional dependency.

3.) Transitive functional dependencies :- An attribute is determined by a non key attribute which is internally determined by a key attribute then there exist transitive functional dependencies.

- In transitive functional dependencies we have redundancy and anomaly.

Ex-

A	B	C	D

$A \rightarrow C$
$C \rightarrow D$ → Transitive functional dependency.

Let us consider a relation of four attribute A, B, C and D where A is the key attribute.

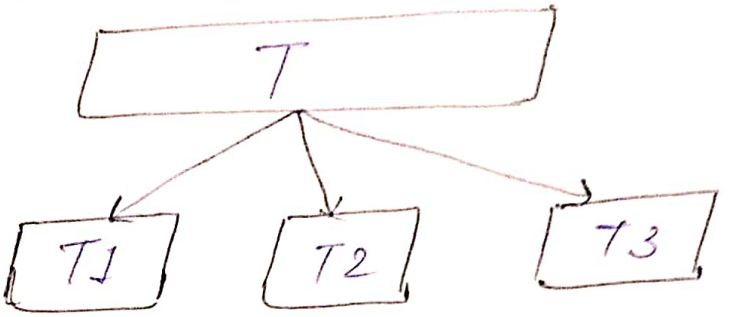
Normalization :- It is the process of reducing a large table into smaller table in order to remove redundancy and anomaly by identifying these functional dependencies is known as normalization.

OR

The process of decomposing a large table into a smaller table is known as normalization.

OR

Reducing a table to its normal form is known as normalization



Q- What is normal form?

Ans- A table without redundancy and anomaly are said to be normal form.

Levels of Normal form:-

1.) First normal form (1NF)
2.) Second normal form (2NF)
3.) Third normal form (3NF)
4) Boyce-Codd normal form (BCNF)

Note:- If any table or entity is reduced to third normal form then table is said to be normalized.

1.) First Normal Form:- • No duplicate Records.
• Multi valued data should not be present.

Ex. OSP

Sid	SName	Course
101	A	SQL, Java
102	B	Testing, SQL
103	C	Java, Testing

⇒

OSP

Sid	SName	C1	C2	C3
101	A	SQL	Java	
102	B	SQL		Testing
103	C		Java	Testing

2.) Second Normal form:-
• Table should be in 1NF.
• Table should not have partial functional dependencies.

Ex-

Emp

composite key

Empno	Ename	sal	deptno	dname	Loc
101	A	1000	10	D1	L1
102	B	2000	20	D2	L2
103	C	3000	10	D1	L1
104	D	4000	20	D2	L2

(empno, deptno) → (ename, sal, dname, Loc)

deptno → (dname, loc)

empno → ename, sal

Empno	Ename	sal	deptno
101	A	1000	10
102	B	2000	20
103	C	3000	10
104	D	4000	20

deptno	dname	Loc
10	D1	L1
20	D2	L2
10	D1	L1
20	D2	L2

3.) Third Normal form

- Table should be in 2NF.
- Table should not have transitive functional dependency

Ex- Emp → (eid, ename, sal, pincode, state, country)

eid → ename, sal

pincode → state, country

R1 → (Eid, ename, sal, pincode) (pincode: FK)

R2 → (pincode, state, country)

Note-

First Normal form (1NF)
→ Single automic value in each row have uniquely identify
↓
Second Normal form (2NF)
→ Satisfy the 1NF condition
→ Partial functional dependancy must be remove from the table.
↓
Third Normal form (3NF)
→ Satisfy all condition of 2NF.

~~or Transitive functional dependency~~

→ Transitive functional dependency of Non-key attributes on key column must be remove