Basic

SQL

with 80 example queries

CONTENTS

Queries

1) Query to get details of the employees working in department no. 30 as a clerk
2) Query to display the details of employees working in department 30
3) Query to get the details of employees who joined after 1981
4) Write a query to display employee name, designation, half-term salary from emp table
5) Write a query to display details of employees with their annual salary and Half-term salary
6) Write a query to display employee names and their designations with annual salary with 20% deduction, without using minus symbol
7) Write a query to display names of employees with their salary with bonus 200 without using + symbol
8) Write a query to display names and designations of employs who are managers
9) Write a query to display details of employs who are getting salary less than 1500
10) Write a query to display details of employees who are getting salary less their commission
11) Write a Query to display details of employees who are working in department 30 and as salesman or manager
12) Write a query to display details of employees who are not working as clerks
13) Write a query to display the details of the employees who are working in department 30 or 20 and joined in year 1981
14) Write a query to display details of employees who joined in 1982
15) Write a query to display details of employees except the persons getting salary from 1000 to 3000
16) Query to get details of employees whose employee number end with digit 8
17) Write a query to display details of employees joined in deptno 30 and display their names in ascending order
18) Write a query to display the deptno in descending order and the names in each dept in ascending order
19) Write a query to display names of employees whose name contains more than 4 characters
20) Write a query to display names of employees whose name contains odd number of characters
21) Write a query to display the number of times occurrence of character ‘a’ in each name of employees
22) Write a query to display names that which contains character ‘A’ twice in them
23) Write a query to display the details of employees whose name starts and ends with same character

24) Write a query to display details of employees who are having 40 years of experience till date
25) Write a query to display details of employees joined on even months
26) Write a query to display employee details who joined on leap year
27) Write a query to display names & joining dates of employees who joined in 2nd half of the month
28) Query to display employees joined on a Friday
29) Write a query to display the number of employees working at deptno 30
30) Write a query to display the number of employees joined in the year 1981
31) Write a query to display the number of employees working as clerk and number of clerks getting commission
32) Write a query to display number of employees who have salary more than 2000
33) Write a query to display number of clerks in emp table and display number of clerks getting salary more than 1000
34) A query to display the number of employees In each department
35) Query to display the number of employees working in each designation in each department
36) Write a query to display the number of employees working as salesman, manager, analyst
37) Write a query to display number of employees joined in the year 1981
38) Write a query to display number of employees joined in each year
39) Write a query to display repeated salaries from emp
40) Write a query to display non-repeated designation
41) Query to display deptno where the number of people working are more than 3
42) Write a query to display the year where number of employees joined more than 5
43) Write a query to display names of designations where at least 2 persons working in each designation
44) Write a query to display number of employees getting salary more than 1000 in each department and display only department number where at least 3 employees are working
45) Write a query to display total salary paid for each designation should be more than 4000
46) Write a query to display deptno whose maximum salary is more than 4000
47) Write a query to display average salary paid for each designation and display only the designation which gets more than 2000
48) Query to display details of employees from the same deptno as Allen's

49) Write a query to display details of employees who are getting salary more than miller
50) Write a query to display names of employees who joined in the same year as Turner
51) Write a query to display details of employees who are working as Martin's designation and getting salary more than smith
52) Write a query to display details of employees reporting for Blake
53) Write a query to display the employees whose starting character of the name is same as the ending character of the names of employees who are working in empno is 7900
54) Write a query to display employees working in the same department of ford and whose first letter of their names has an odd ASCII value
55) Write a query to display details of employees, who are getting salary more than the salary of the employees working in deptno 20
56) Write a query to display details of employees who are working in the same deptno as King or Allen and working in the designation same as Turner or Jones
57) Write a query to display details of employees who joined in any of the years of Smith or Scott and getting salary more than James and Smith
58) Query to get the last record of a table
59) Query to get the first record of a table
60) Query to get the first 5 records from table emp
61) Query to display every record of the table emp
62) Query to display the 4th record of emp table
63) Standard query to display the nth record from any table
64) Write a query to display the second half of the emp table
65) Write a query to display records of emp table by eliminating last 3 records
66) Write a query to display even records from emp table
67) Write a query to display the middle record of emp table
68) A query to get the middle record
69) Write a query to display the 3rd quarter records of emp table
70) Write a query to display details of employees working in the location DALLAS
71) Write a query to display number of employees working in sales department
72) Query to display employees who are working in same designation as miller
73) Write a query to display names of employees with their reporting Manager's name
74) Write a query to display the number of employees reporting to Blake and King
75) Write a query to display employees, their location, and their reporting managers, it uses both Self Join and Inner Join

76) Write a query to display names of employees with their reporting managers where both employees and reporting manager working in same location

77) Write a query to display names of employees with their reporting managers and their respective hiredate and display only employees who joined before their reporting manager

78) Write a query to display details of employees who are not working as reporting managers using Joins

79) Write a query to display the 4th maximum salary from emp table

80) Write a query to display the names of the employees who are having second maximum number of characters

INTRODUCTION

The first name of SQL was “Sequel”, (Simple English query executable language. SQL was found by Raymond Boyce and Donald Chamberlin. SQL is used to interact or communicate with database i.e., it is used to perform CRUD (Create, Read, Update and Delete) operations and manage or modify, the data which is existing are DBMS type database, standing for ‘Database management system.’ SQL, Structured Query Language is a language that is designed to interact only with RDBMS type databases, standing for ‘Relational Database management system.’ It was certified by American National Standard Institute, ANSI and International Standard Organization, ISO.

IDENTIFIERS

The names provided for columns, tables or variables are called as Identifiers.

- It should also always begin with alphabets, cannot begin with numbers or special characters.
- The names given as an Identifier must always be unique.

DATATYPES

It decides/Specifies the type of data that should be accepted or stored by a variable or a column.

- Char(size)
- Varchar(size)
- Varchar(size)
- Number (precision, scale)
- Date
- Timestamp
- Int
- Float
- LOB (Large Object)
- BLOB (Binary LOB)
- CLOB (Character LOB)

1) Char() – Char(size), If a column is mentioned with char datatype, the column can accept only string or character type data. Anything that is written between single quotes are called string or Character.

- The size of character datatype is 2000 characters
- Character datatype is known as fixed memory allocation datatype.
- "Here the memory is not used in an efficient way, and memory is wasted in character datatype".
- Char datatype is faster in process than a Variable character (Varchar) datatype.

2) Varchar() – Varchar(size), if a column is mentioned with Varchar (variable character) datatype, it can only accept string and character datatype into it.

- Variable character datatype allows size up to 2000 character.
- Variable character datatype is also known as Variable memory allocation.
- In Varchar the memory is used in an efficient and there is no wastage of memory.
- Varchar is slower than char datatype comparatively.

3) Varchar2() – Varchar2(size), It is like varchar except that it can allow size up to 4000.

4) Number() – Number(precision, scale), If a column is mentioned with number datatype, it can only accept only number type data an Input.

- Precision decides total number of digits to be entered as input in a number.
- Scale decides the number of digits after decimal point to be allowed to enter by the user as input.
- The range of precision is from 1 – 38.
- If the number datatype has only been given attribute precision than it can only accept integers as a datatype, to be able to accept decimal numbers too, the number datatype must be described with attributes precision and scale together.

03-12-2022

5) Date – When a column is mentioned date datatype. It can accept only oracle date format as input into it.

Standard oracle format:

DD-MMM-YYYY or

DD-MMM-YY

6) Large Objects - Binary large objects - If a column is mentioned with BLOB, it can only accept large binary type file in it such as .mp3, .mp4, .jpg.

- The range of BLOB if 4 GB.

7) Character Large Object - If a column is mentioned with CLOB, it can only accept large text type files in it, such as .pdf, .txt, .doc, .xml, etc.,

- The range of CLOB is 4 GB.

CONSTRAINTS

Are used to apply some rules and regulations with conditions to validate the data before accepting into tables.

Null – If a column is mentioned with null constraint, the column becomes optional and column can accept Null values, Not Null values, and Duplicate values into it as well.

- It is a default constraint, i.e., if any column is not mentioned with any constraint, Null will be automatically assigned as a default constraint to the column.

Not Null – If a column is mentioned with “Not Null”, the column becomes mandatory for the user to input some value. The column can only accept Not Null values and duplicate values in it.

Unique – For a column mentioned with it, it can only accept different values in to it, and it cannot accept duplicate or repeated values.

- A column with unique constraint can accept only one Null value in it.

Check – It is used to provide conditions to validate the data before accepting into tables.

- The data will be entered or accepted only if the condition is met, or else the data is excluded.

Primary Key - It assigns a Unique Identity value for every row→, primary key access the reference key to make relationship between tables.

- A table can have only one primary key in it.
- Primary key column can accept only unique values and not null values.

Foreign Key - It is a constraint used to build relationship between tables using primary key as reference.

- If a column is mentioned with foreign key, it can only accept the values which is present in reference key column and it can also accept Null value or duplicate values in it.
- If a column must be foreign key, it should be primary key in another table
- A table can contain more than one foreign key in it.

SQL STATEMENTS

A set of Keywords, predefined syntax, and Functions to perform CRUD operations in a Database. There are 5 types of SQL statements;

- DDL (Data Definition Language)
- DML (Data Manipulation Language)
- DCL (Data Control Language)
- TCL (Transaction Control Language)
- DQL (Data Query Language)

10-12-2022

SETTING UP THE SESSION

Set command is used to assign the values to the Variables.

1) Page-size and Line-size is set to get the data in a proper and aligned way.

- -The range of Page-size is from 0-50000
- -The range of line-size is from 1-32767

E.g., set pagesize 100;

Set linesize 100;

2) To know the current user;
E.g., Show user;

3) To switch to other user accounts;
E.g., Connect 'username';
Ed;
r; or /;

4) To change the password of user;
E.g., Alter user 'username' identified by;

5) To know the description of the table;
E.g., Desc 'tablename'; or describe 'Tablename';

24-12-2022

DDL (DATA DEFINITION LANGUAGE)

It is a set of SQL statements and set of Keywords that which defines the structures of Tables, Views, Users, Scheme.

The command statements of DDL

- Create
- Rename
- Alter – Add a column, delete a column, Rename, Modify
- Drop
- Flashback
- Purge
- Truncate

Create

- It is a DDL command statement used for constructing or building structures of Tables, Schemes, Views.

- The columns are assigned with respective data type and constraints in create command statement.

Syntax;

```
Create table Tablename (Column 1 name datatype(size) constraints
                        Column 2 name datatype(size) constraints
                        Column 3 name datatype(size) constraints
                                                    ……);
```

Rename

- It is a DDL Command used to change the name of Table, Columns, Scheme, Views, etc.,

Syntax;

```
Rename existing Tablename to new Tablename;
```

- When a name to an entity is renamed, the old name is deleted from the database and the new name is registered.

Alter

- Alter is a DDL statement used to modify the statements of Tables, views users.
- Using Alter statement, a table can be modified such as adding a column to the table, modifying columns, removing/dropping a column from a table, and renaming of columns in a table.

1) To add a new column

Syntax;

```
Alter table Tablename add (Column 1 name datatype(size), Constraints
                                    Column 2 name datatype(size), Constraints
                                   ……..);
```

2) To modify

Syntax;

```
Alter table Tablename modify (Column 1 name datatype(size), Constraints
                                          Column 2 name datatype(size),
Constraints
                                            ……..);
```

Note: *If you do not need to modify datatype or constraint, then only mention the parameter you want to modify. If we need to change datatype, then only mention datatype. When modifying a column using alter, the same constraint is not supposed to be repeated to the column.*

Before modifying;

```
SQL> desc DECEMBER312022;
Name                                      Null?            Type
 ----------------------------------------- -------- -------------
 EID                                           NOT NULL  NUMBER(10)
 NAME                                      NOT NULL  VARCHAR2(10)
```

Alter table DECEMBER312022
Modify EID number(15);

After modifying;

```
SQL> desc DECEMBER312022;
 Name                                       Null?              Type
 ----------------------------------------- -------- ------------
 EID                                           NOT NULL   NUMBER(15)
 NAME                                          NOT NULL   VARCHAR2(10)
```

31-12-2022

3) To delete a column of a table;

Alter table Tablename
Drop column Columnname;

Drop

- Drop is a DDL command statement used to delete a table including data from database and state in recycle bin.

Flashback

- It is a DDL command statement used to replace or restore the dropped table from recycle bin to database,

Note: *Flashback can be performed only on dropped tables.*

Syntax;

Flashback table Tablename to before drop;

Purge

- It is a DDL command used to delete a table from recycle bin after dropping it which permanently deletes the table from database.

1) To drop and purge simultaneously;

Drop table Tablename purge;

- Purge cannot be performed on a table present in database.

Truncate

- It is a DDL command statement used to erase the entire data present on any table permanently, and the table structure remains same.

DML (DATA MANIPULATION LANGUAGE)

It is a Statement used to modify the data or adding new data or delete a particular data selected from the table.

- Insert
- Update
- Delete

Insert

- It is a DML command statement used to add new records or data into a table.

Syntax;

1) When you do not know the order of the columns, then you need to mention the order of the columns like given below;

```
Insert into Table-name
(column 1 name column 2 name column 3 name ….)
Values (‘Value 1’ ‘Value 2’ ‘Value 3’……);
```

2) Otherwise, when you know the order of the columns;

```
Insert into Table-name
Values (‘Value 1’ ‘Value 2’ ‘Value 3’……);
```

3) Inserting multiple records;

Syntax;

```
Insert all
Into Table-name (column 1 name column 2 name column 3 name …..)
Values (‘Value 1’ ‘Value 2’ ‘Value 3’ ……);
“
“
“
“
“
“
nth record
Select * from dual;
```

- Dual is an Empty space to store all records and execute all at once.

13-02-2023

Update

- Update is a DML command statement used to modify the existing data in a table.

Syntax;

```
Update Table-name
Set column-name = value
Where <filtering-condition>;
```

1) For updating whole record/multiple data

```
Update Table-name
Set column-name = 'value', column-name = 'value'…….
Where <any one filtering-condition in that particular record>;
```

Where

- It is used to filter according to the filtering conditions. And it filters row-by-row.

Note: *If 'Where' line is not mentioned, every row/record will be updated with the same value.*

Order of Execution;

- Update clause
- where clause
- set clause

Rollback

- To undo the DML commands.

```
SQL> select * from std2;

        ID NAME            EMAIL                    MOBILE
---------- --------------- -------------------- ----------
         1 parbakar        p@palknama           1234567899
         2 rolex                                9876543201
         3 sarala          m@m                  8765432199
         6 baby            ABC@123              9999999999
         5 benten          ben@123              2942396421

SQL> update std2
  2  set email=null;

5 rows updated.
```

```
SQL> select * from std2;

        ID NAME            EMAIL                    MOBILE
---------- --------------- -------------------- ----------
         1 parbakar                             1234567899
         2 rolex                                9876543201
         3 sarala                               8765432199
         6 baby                                 9999999999
         5 benten                               2942396421

SQL> rollback;

Rollback complete.

SQL> select * from std2;

        ID NAME            EMAIL                    MOBILE
---------- --------------- -------------------- ----------
         1 parbakar        p@palknama           1234567899
         2 rolex                                9876543201
         3 sarala          m@m                  8765432199
         6 baby            ABC@123              9999999999
         5 benten          ben@123              2942396421

SQL>  update std2
  2  set name='Sunny'
  3  where id=6;

1 row updated.

SQL> select * from std2;

        ID NAME            EMAIL                    MOBILE
---------- --------------- -------------------- ----------
         1 parbakar        p@palknama           1234567899
         2 rolex                                9876543201
         3 sarala          m@m                  8765432199
         6 Sunny           ABC@123              9999999999
         5 benten          ben@123              2942396421

SQL> update std2
  2  set id=4 , name='JONY', email='j@420',mobile=9876543211
  3  where id=3;

1 row updated.

SQL> select * from std2;

        ID NAME            EMAIL                    MOBILE
---------- --------------- -------------------- ----------
         1 parbakar        p@palknama           1234567899
         2 rolex                                9876543201
         4 JONY            j@420                9876543211
         6 Sunny           ABC@123              9999999999
         5 benten          ben@123              2942396421

SQL> update std2
  2  set mobile=1234567890;
update std2
*
ERROR at line 1:
ORA-00001: unique constraint (SCOTT.BIN$EZRIQXkWSgS+qBvmTT35Og==$0) violated
```

Delete

- Delete is a DML command statement used when we require to delete/remove record/records/row/rows from a particular table.

Syntax;

Delete
From 'table-name'
Where <filtering condition>;

Order of Execution;

- From clause
- Where clause
- Delete

```
SQL> delete
  2  from std2
  3  where Name='sunny';

0 rows deleted.

SQL> delete
  2  from std2
  3  where Name='Sunny';

1 row deleted.

SQL> select * from std2;

        ID NAME            EMAIL                    MOBILE
---------- --------------- -------------------- ----------
         1 parbakar        p@palknama           1234567899
         2 rolex                                9876543201
         4 JONY            j@420                9876543211
         5 benten          ben@123              2942396421

SQL> show user;
USER is "SCOTT"
SQL> connect hr;
Enter password: *****
Connected.
SQL> select * from tab;

TNAME                          TABTYPE  CLUSTERID
------------------------------ ------- ----------
REGIONS                        TABLE
COUNTRIES                      TABLE
LOCATIONS                      TABLE
DEPARTMENTS                    TABLE
JOBS                           TABLE
EMPLOYEES                      TABLE
JOB_HISTORY                    TABLE
EMP_DETAILS_VIEW               VIEW
DEMOPY123                      TABLE

9 rows selected.

SQL> select * from std2;
select * from std2
```

```
             *
ERROR at line 1:
ORA-00942: table or view does not exist


SQL> select * from scott.std2;
select * from scott.std2
                    *
ERROR at line 1:
ORA-00942: table or view does not exist
```

DCL (DATA CONTROL LANGUAGE)

Data control language is a statement used to control the flow of data from one user to the other.

- Grant
- Revoke

Grant

- Grant is a DCL statement used to provide permissions to other users to access the tables/data present in the current user.

Syntax;

Grant (DQL statement) →select/delete/update/insert/all
On table-name
To username;

```
SQL> connect scott;
Enter password: *****
Connected.
SQL> grant select on std2 to hr;

Grant succeeded.

SQL> connect hr;
Enter password: *****
Connected.
SQL> select * from scott.std2;

        ID NAME            EMAIL                    MOBILE
---------- --------------- -------------------- ----------
         1 parbakar        p@palknama           1234567899
         2 rolex                                9876543201
         4 JONY            j@420                9876543211
         5 benten          ben@123              2942396421

SQL> delete from scott.std2
  2  where id=5;
delete from scott.std2
                  *
ERROR at line 1:
ORA-01031: insufficient privileges


SQL> connect scott;
```

```
Enter password: *****
Connected.
SQL> grant all on std2 to hr;

Grant succeeded.

SQL> connect hr;
Enter password: *****
Connected.
SQL> select * from scott.std2;

        ID NAME            EMAIL                    MOBILE
---------- --------------- -------------------- ----------
         1 parbakar        p@palknama           1234567899
         2 rolex                                9876543201
         4 JONY            j@420                9876543211
         5 benten          ben@123              2942396421

SQL> delete from scott.std2
  2  where id=5;

1 row deleted.

SQL> select * from scott.std2;

        ID NAME            EMAIL                    MOBILE
---------- --------------- -------------------- ----------
         1 parbakar        p@palknama           1234567899
         2 rolex                                9876543201
         4 JONY            j@420                9876543211
```

Revoke

- Revoke is a DCL command statement used to cancel the permissions to access the data in the current.

Syntax;

Revoke (DQL statement) →select/delete/update/insert/all
On table-name
from username;

```
SQL> connect scott;
Enter password: *****
Connected.
SQL> revoke select on std2 from hr;

Revoke succeeded.

SQL> revoke select on std2 from hr;
revoke select on std2 from hr
*
ERROR at line 1:
ORA-01927: cannot REVOKE privileges you did not grant


SQL> revoke all on std2 from hr;

Revoke succeeded.

SQL>  revoke all on std2 from hr;
```

```
Revoke succeeded.

SQL> connect hr;
Enter password: *****
Connected.
SQL> select * from scott.std2;
select * from scott.std2
                    *
ERROR at line 1:
ORA-00942: table or view does not exist
```

Note: *To grant access to every user's every table, we can use DBA/ Database administrator command.*

1) To grant

Syntax;

Grant DBA to username;

Note: *For granting the DBA to any user, the granting user should be a DBA/ Database administrator first.*

2) To revoke DBA

Syntax;

Revoke DBA from username;

```
SQL> connect system
Enter password: *****
Connected.
SQL> grant dba to hr;

Grant succeeded.

SQL> connect hr;
Enter password: *****
Connected.
SQL> select * from scott.std2;

        ID NAME            EMAIL                    MOBILE
---------- --------------- -------------------- ----------
         1 parbakar        p@palknama           1234567899
         2 rolex                                9876543201
         4 JONY            j@420                9876543211

SQL> select * from scott.emp;

     EMPNO ENAME      JOB              MGR HIREDATE         SAL        COMM      DEPTNO
---------- ---------- --------- ---------- --------- ---------- ---------- ----------
      7369 SMITH      CLERK           7902 17-DEC-80        800                    20
      7499 ALLEN      SALESMAN        7698 20-FEB-81       1600        300         30
      7521 WARD       SALESMAN        7698 22-FEB-81       1250        500         30
      7566 JONES      MANAGER         7839 02-APR-81       2975                    20
      7654 MARTIN     SALESMAN        7698 28-SEP-81       1250       1400         30
      7698 BLAKE      MANAGER         7839 01-MAY-81       2850                    30
      7782 CLARK      MANAGER         7839 09-JUN-81       2450                    10
      7788 SCOTT      ANALYST         7566 19-APR-87       3000                    20
      7839 KING       PRESIDENT            17-NOV-81       5000                    10
      7844 TURNER     SALESMAN        7698 08-SEP-81       1500          0         30
```

```
      7876 ADAMS      CLERK           7788 23-MAY-87       1100                    20
      7900 JAMES      CLERK           7698 03-DEC-81        950                    30
      7902 FORD       ANALYST         7566 03-DEC-81       3000                    20
      7934 MILLER     CLERK           7782 23-JAN-82       1300                    10

14 rows selected.

SQL> select * from scott.dept;

    DEPTNO DNAME          LOC
---------- -------------- -------------
        10 ACCOUNTING     NEW YORK
        20 RESEARCH       DALLAS
        30 SALES          CHICAGO
        40 OPERATIONS     BOSTON

SQL> connect scott;
Enter password: *****
Connected.
SQL> revoke dba from hr;

Revoke succeeded.

SQL>
```

14-02-2023

TCL (TRANSACTION CONTROL LANGUAGE)

It is a statement used to control the transactions performed by DML command statements such as insert/update/delete. Command statements of TCL are;

- Commit
- Rollback
- Save-point reference-name
- Rollback to save-point name

Commit

- Commit is a TCL command statement used to save the transactions performed by DML command statements permanently to the main database.

Rollback

- Rollback is a command statement used to undo the transactions performed by the DML command statements. Rollback only undo'es the DML statements up-to the lastly committed state and returns the lastly saved data before commit point.

```
SQL*Plus: Release 10.2.0.1.0 - Production on Tue Feb 14 17:57:54 2023

Copyright (c) 1982, 2005, Oracle.  All rights reserved.


Connected to:
Oracle Database 10g Enterprise Edition Release 10.2.0.1.0 - Production
With the Partitioning, OLAP and Data Mining options
```

```
SQL> select * from std2;

        ID NAME            EMAIL                    MOBILE
---------- --------------- -------------------- ----------
         1 parbakar        p@palknama           1234567899
         2 rolex                                9876543201
         4 JONY            j@420                9876543211
         5 sunny           s@12                 9870012345

SQL> insert into std2 values(6,'kattappa','k@sarala',1111122222);

1 row created.

SQL> update std2
  2  set name='joncena'
  3  where id=4;

1 row updated.

SQL> delete from std2
  2  where id=2;

1 row deleted.

SQL> select * from std2;

        ID NAME            EMAIL                    MOBILE
---------- --------------- -------------------- ----------
         1 parbakar        p@palknama           1234567899
         4 joncena         j@420                9876543211
         5 sunny           s@12                 9870012345
         6 kattappa        k@sarala             1111122222

SQL> rollback;

Rollback complete.

SQL> select * from std2;

        ID NAME            EMAIL                    MOBILE
---------- --------------- -------------------- ----------
         1 parbakar        p@palknama           1234567899
         2 rolex                                9876543201
         4 JONY            j@420                9876543211
         5 sunny           s@12                 9870012345

SQL> alter table std2
  2  drop column email;

Table altered.

SQL> select * from std2;

        ID NAME                MOBILE
---------- --------------- ----------
         1 parbakar        1234567899
         2 rolex           9876543201
         4 JONY            9876543211
         5 sunny           9870012345

SQL> rollback;

Rollback complete.

SQL> select * from std2;
```

```
        ID NAME                MOBILE
---------- --------------- ----------
         1 parbakar        1234567899
         2 rolex           9876543201
         4 JONY            9876543211
         5 sunny           9870012345

SQL>  insert into std2 values(6,'kattappa','k@sarala',1111122222);
 insert into std2 values(6,'kattappa','k@sarala',1111122222)
             *
ERROR at line 1:
ORA-00913: too many values


SQL> insert into std2 values(6,'kattappa',1111122222);

1 row created.

SQL> savepoint a;

Savepoint created.

SQL> insert into std2 values(7,'tom',9000000111);

1 row created.

SQL> savepoint b;

Savepoint created.

SQL> insert into std2 values(8,'jilebi',1431434201);

1 row created.

SQL> select * from std2;

        ID NAME                MOBILE
---------- --------------- ----------
         1 parbakar        1234567899
         2 rolex           9876543201
         4 JONY            9876543211
         5 sunny           9870012345
         6 kattappa        1111122222
         7 tom             9000000111
         8 jilebi          1431434201

7 rows selected.

SQL> rollback to b;

Rollback complete.

SQL> select * from std2;

        ID NAME                MOBILE
---------- --------------- ----------
         1 parbakar        1234567899
         2 rolex           9876543201
         4 JONY            9876543211
         5 sunny           9870012345
         6 kattappa        1111122222
         7 tom             9000000111

6 rows selected.

SQL> insert into std2 values(9,'jilebi',1431434201);
```

```
1 row created.

SQL> select * from std2;

        ID NAME                MOBILE
---------- --------------- ----------
         1 parbakar        1234567899
         2 rolex           9876543201
         4 JONY            9876543211
         5 sunny           9870012345
         6 kattappa        1111122222
         7 tom             9000000111
         9 jilebi          1431434201

7 rows selected.

SQL> rollback to b;

Rollback complete.

SQL> select * from std2;

        ID NAME                MOBILE
---------- --------------- ----------
         1 parbakar        1234567899
         2 rolex           9876543201
         4 JONY            9876543211
         5 sunny           9870012345
         6 kattappa        1111122222
         7 tom             9000000111

6 rows selected.

SQL> rollback to a;

Rollback complete.

SQL> select * from std2;

        ID NAME                MOBILE
---------- --------------- ----------
         1 parbakar        1234567899
         2 rolex           9876543201
         4 JONY            9876543211
         5 sunny           9870012345
         6 kattappa        1111122222

SQL> rollback to b;
rollback to b
*
ERROR at line 1:
ORA-01086: savepoint 'B' never established


SQL> commit;

Commit complete.

SQL> rollback to a;
rollback to a
*
ERROR at line 1:
ORA-01086: savepoint 'A' never established


SQL>
```

```
Announcement: "SQL> set pages 100 lines 100; SQL>…"
Hemanth Sriramulu
Created 6:43 PM6:43 PM
SQL> set pages 100 lines 100;
SQL> select * from std2;

        ID NAME            EMAIL                    MOBILE
---------- --------------- -------------------- ----------
         1 parbakar        p@palknama           1234567899
         2 rolex                                9876543201
         4 JONY            j@420                9876543211

SQL> rollback;

Rollback complete.

SQL> select * from std2;

        ID NAME            EMAIL                    MOBILE
---------- --------------- -------------------- ----------
         1 parbakar        p@palknama           1234567899
         2 rolex                                9876543201
         4 JONY            j@420                9876543211

SQL>  commit;

Commit complete.

SQL> insert into std2 values(5,'sunny','s@12',9870012345);

1 row created.

SQL> select * from std2;

        ID NAME            EMAIL                    MOBILE
---------- --------------- -------------------- ----------
         1 parbakar        p@palknama           1234567899
         2 rolex                                9876543201
         4 JONY            j@420                9876543211
         5 sunny           s@12                 9870012345

SQL>
```

Save-point and Rollback

- Savepoint is a TCL command statement used to mark the transactions and helps to perform rollback to a certain marked position of a savepoint.

Note: *If we commit, only the data gets saved to the main database and the savepoints are erased. If we rollback without mentioning any savepoint, everything altered from committed state is undo-ed.*

1
2
4
5
Commit;

6;
Savepoint a;
7;
Savepoint b;
8;
Rollback to b;
Rollback to a;
Rollback;

15-02-2023

DQL (DATA QUERY LANGUAGE)

It is a statement used to retrieve data from the tables. Statements of DQL are;

- Select

The retrieving process using select statement is further divided into three types they are;

- Projection
- Selection
- Joins

Projection

- Projection is a process of retrieving data from the tables using only specified columns.

Syntax;

```
Select Distinct */columnname/Expressions/Aliasing/tablename. *
From tablename;
```

Order of Execution;
- From clause
- Select clause

From Clause

- From clause is used to select the source table from where the data is to be fetched.

Select Clause

- Select clause is used to display the specified columns from the source table.

(Asterisk *)

- If a select statement is mentioned with asterisk/*, by default it selects all the columns from the table, retrieves and dispalys in the console.
- No columns or expressions are allowed when select is mentioned with asterisk/*.

```
Connected to:
Oracle Database 10g Enterprise Edition Release 10.2.0.1.0 - Production
With the Partitioning, OLAP and Data Mining options

SQL> set pages 100 lines 100;
SQL> select * from emp;

     EMPNO ENAME      JOB              MGR HIREDATE         SAL       COMM     DEPTNO
---------- ---------- --------- ---------- --------- ---------- ---------- ----------
      7369 SMITH      CLERK           7902 17-DEC-80        800                    20
      7499 ALLEN      SALESMAN        7698 20-FEB-81       1600        300         30
      7521 WARD       SALESMAN        7698 22-FEB-81       1250        500         30
      7566 JONES      MANAGER         7839 02-APR-81       2975                    20
      7654 MARTIN     SALESMAN        7698 28-SEP-81       1250       1400         30
      7698 BLAKE      MANAGER         7839 01-MAY-81       2850                    30
      7782 CLARK      MANAGER         7839 09-JUN-81       2450                    10
      7788 SCOTT      ANALYST         7566 19-APR-87       3000                    20
      7839 KING       PRESIDENT            17-NOV-81       5000                    10
      7844 TURNER     SALESMAN        7698 08-SEP-81       1500          0         30
      7876 ADAMS      CLERK           7788 23-MAY-87       1100                    20
      7900 JAMES      CLERK           7698 03-DEC-81        950                    30
      7902 FORD       ANALYST         7566 03-DEC-81       3000                    20
      7934 MILLER     CLERK           7782 23-JAN-82       1300                    10

14 rows selected.

SQL> select *,empno from emp;
select *,empno from emp
        *
ERROR at line 1:
ORA-00923: FROM keyword not found where expected


SQL> select ename,job from emp;

ENAME      JOB
---------- ---------
SMITH      CLERK
ALLEN      SALESMAN
WARD       SALESMAN
JONES      MANAGER
MARTIN     SALESMAN
BLAKE      MANAGER
CLARK      MANAGER
SCOTT      ANALYST
KING       PRESIDENT
TURNER     SALESMAN
ADAMS      CLERK
JAMES      CLERK
FORD       ANALYST
MILLER     CLERK

14 rows selected.

SQL> select ename,job,ename,job from emp;

ENAME      JOB       ENAME      JOB
---------- --------- ---------- ---------
SMITH      CLERK     SMITH      CLERK
ALLEN      SALESMAN  ALLEN      SALESMAN
WARD       SALESMAN  WARD       SALESMAN
JONES      MANAGER   JONES      MANAGER
MARTIN     SALESMAN  MARTIN     SALESMAN
BLAKE      MANAGER   BLAKE      MANAGER
```

```
CLARK      MANAGER   CLARK      MANAGER
SCOTT      ANALYST   SCOTT      ANALYST
KING       PRESIDENT KING       PRESIDENT
TURNER     SALESMAN  TURNER     SALESMAN
ADAMS      CLERK     ADAMS      CLERK
JAMES      CLERK     JAMES      CLERK
FORD       ANALYST   FORD       ANALYST
MILLER     CLERK     MILLER     CLERK

14 rows selected.

SQL> select ename,sal,job,deptno
  2  from emp;

ENAME              SAL JOB            DEPTNO
---------- ---------- --------- ----------
SMITH              800 CLERK              20
ALLEN             1600 SALESMAN           30
WARD              1250 SALESMAN           30
JONES             2975 MANAGER            20
MARTIN            1250 SALESMAN           30
BLAKE             2850 MANAGER            30
CLARK             2450 MANAGER            10
SCOTT             3000 ANALYST            20
KING              5000 PRESIDENT          10
TURNER            1500 SALESMAN           30
ADAMS             1100 CLERK              20
JAMES              950 CLERK              30
FORD              3000 ANALYST            20
MILLER            1300 CLERK              10

14 rows selected.

SQL> select from emp;
select from emp
       *
ERROR at line 1:
ORA-00936: missing expression


SQL> select * from em p;
select * from em p
              *
ERROR at line 1:
ORA-00942: table or view does not exist


SQL> select * from emp;

     EMPNO ENAME      JOB              MGR HIREDATE         SAL       COMM     DEPTNO
---------- ---------- --------- ---------- --------- ---------- ---------- ----------
      7369 SMITH      CLERK           7902 17-DEC-80        800                    20
      7499 ALLEN      SALESMAN        7698 20-FEB-81       1600        300         30
      7521 WARD       SALESMAN        7698 22-FEB-81       1250        500         30
      7566 JONES      MANAGER         7839 02-APR-81       2975                    20
      7654 MARTIN     SALESMAN        7698 28-SEP-81       1250       1400         30
      7698 BLAKE      MANAGER         7839 01-MAY-81       2850                    30
      7782 CLARK      MANAGER         7839 09-JUN-81       2450                    10
      7788 SCOTT      ANALYST         7566 19-APR-87       3000                    20
      7839 KING       PRESIDENT            17-NOV-81       5000                    10
      7844 TURNER     SALESMAN        7698 08-SEP-81       1500          0         30
      7876 ADAMS      CLERK           7788 23-MAY-87       1100                    20
      7900 JAMES      CLERK           7698 03-DEC-81        950                    30
      7902 FORD       ANALYST         7566 03-DEC-81       3000                    20
      7934 MILLER     CLERK           7782 23-JAN-82       1300                    10

14 rows selected.
```

Expressions

Expression is a combination of operands and operators to perform some mathematical operations and returns an output.

```
SQL> select sal,sal*12  from emp;

       SAL     SAL*12
---------- ----------
       800       9600
      1600      19200
      1250      15000
      2975      35700
      1250      15000
      2850      34200
      2450      29400
      3000      36000
      5000      60000
      1500      18000
      1100      13200
       950      11400
      3000      36000
      1300      15600

14 rows selected.

SQL> select ename as emplyname , job as designation, sal*12 as "annual sal"
  2  from emp;

EMPLYNAME  DESIGNATI annual sal
---------- --------- ----------
SMITH      CLERK           9600
ALLEN      SALESMAN       19200
WARD       SALESMAN       15000
JONES      MANAGER        35700
MARTIN     SALESMAN       15000
BLAKE      MANAGER        34200
CLARK      MANAGER        29400
SCOTT      ANALYST        36000
KING       PRESIDENT      60000
TURNER     SALESMAN       18000
ADAMS      CLERK          13200
JAMES      CLERK          11400
FORD       ANALYST        36000
MILLER     CLERK          15600

14 rows selected.

SQL> select sal*12 annual sal from emp;
select sal*12 annual sal from emp
                      *
ERROR at line 1:
ORA-00923: FROM keyword not found where expected


SQL> select sal monthsal, sal*12 annual_sal from emp;

  MONTHSAL ANNUAL_SAL
---------- ----------
       800       9600
      1600      19200
      1250      15000
      2975      35700
      1250      15000
```

```
      2850      34200
      2450      29400
      3000      36000
      5000      60000
      1500      18000
      1100      13200
       950      11400
      3000      36000
      1300      15600

14 rows selected.
```

Aliasing

- Aliasing is a process of providing a reference name to columns/ tables/ views or schemas, etc., and it behaves as a temporary name.
- Aliasing is performed using a keyword 'as'.

```
SQL> select ename as emplyname , job as designation, sal*12 as "annual sal"
  2  from emp;

EMPLYNAME  DESIGNATI annual sal
---------- --------- ----------
SMITH      CLERK           9600
ALLEN      SALESMAN       19200
WARD       SALESMAN       15000
JONES      MANAGER        35700
MARTIN     SALESMAN       15000
BLAKE      MANAGER        34200
CLARK      MANAGER        29400
SCOTT      ANALYST        36000
KING       PRESIDENT      60000
TURNER     SALESMAN       18000
ADAMS      CLERK          13200
JAMES      CLERK          11400
FORD       ANALYST        36000
MILLER     CLERK          15600

14 rows selected.

SQL> select sal*12 annual sal from emp;
select sal*12 annual sal from emp
                      *
ERROR at line 1:
ORA-00923: FROM keyword not found where expected


SQL> select sal monthsal, sal*12 annual_sal from emp;

  MONTHSAL ANNUAL_SAL
---------- ----------
       800       9600
      1600      19200
      1250      15000
      2975      35700
      1250      15000
      2850      34200
      2450      29400
      3000      36000
      5000      60000
      1500      18000
      1100      13200
```

```
       950      11400
      3000      36000
      1300      15600

14 rows selected.

SQL> select * from emp;

     EMPNO ENAME      JOB              MGR HIREDATE         SAL        COMM      DEPTNO
---------- ---------- --------- ---------- --------- ---------- ---------- ----------
      7369 SMITH      CLERK           7902 17-DEC-80        800                    20
      7499 ALLEN      SALESMAN        7698 20-FEB-81       1600        300         30
      7521 WARD       SALESMAN        7698 22-FEB-81       1250        500         30
      7566 JONES      MANAGER         7839 02-APR-81       2975                    20
      7654 MARTIN     SALESMAN        7698 28-SEP-81       1250       1400         30
      7698 BLAKE      MANAGER         7839 01-MAY-81       2850                    30
      7782 CLARK      MANAGER         7839 09-JUN-81       2450                    10
      7788 SCOTT      ANALYST         7566 19-APR-87       3000                    20
      7839 KING       PRESIDENT            17-NOV-81       5000                    10
      7844 TURNER     SALESMAN        7698 08-SEP-81       1500          0         30
      7876 ADAMS      CLERK           7788 23-MAY-87       1100                    20
      7900 JAMES      CLERK           7698 03-DEC-81        950                    30
      7902 FORD       ANALYST         7566 03-DEC-81       3000                    20
      7934 MILLER     CLERK           7782 23-JAN-82       1300                    10

14 rows selected.

SQL> select emp.* from emp;

     EMPNO ENAME      JOB              MGR HIREDATE         SAL        COMM      DEPTNO
---------- ---------- --------- ---------- --------- ---------- ---------- ----------
      7369 SMITH      CLERK           7902 17-DEC-80        800                    20
      7499 ALLEN      SALESMAN        7698 20-FEB-81       1600        300         30
      7521 WARD       SALESMAN        7698 22-FEB-81       1250        500         30
      7566 JONES      MANAGER         7839 02-APR-81       2975                    20
      7654 MARTIN     SALESMAN        7698 28-SEP-81       1250       1400         30
      7698 BLAKE      MANAGER         7839 01-MAY-81       2850                    30
      7782 CLARK      MANAGER         7839 09-JUN-81       2450                    10
      7788 SCOTT      ANALYST         7566 19-APR-87       3000                    20
      7839 KING       PRESIDENT            17-NOV-81       5000                    10
      7844 TURNER     SALESMAN        7698 08-SEP-81       1500          0         30
      7876 ADAMS      CLERK           7788 23-MAY-87       1100                    20
      7900 JAMES      CLERK           7698 03-DEC-81        950                    30
      7902 FORD       ANALYST         7566 03-DEC-81       3000                    20
      7934 MILLER     CLERK           7782 23-JAN-82       1300                    10

14 rows selected.
```

Distinct

Distinct is a command statement used in select statement to eliminate all the duplicate values or rows and returns only unique values or records as output.

```
SQL> select distinct deptno
  2  from emp;

    DEPTNO
----------
        30
        20
        10

SQL> select distinct job from emp;

JOB
```

```
---------
CLERK
SALESMAN
PRESIDENT
MANAGER
ANALYST

SQL> select job , deptno from emp;

JOB           DEPTNO
--------- ----------
CLERK             20
SALESMAN          30
SALESMAN          30
MANAGER           20
SALESMAN          30
MANAGER           30
MANAGER           10
ANALYST           20
PRESIDENT         10
SALESMAN          30
CLERK             20
CLERK             30
ANALYST           20
CLERK             10

14 rows selected.

SQL> select distinct job,deptno from emp;

JOB           DEPTNO
--------- ----------
MANAGER           20
PRESIDENT         10
CLERK             10
SALESMAN          30
ANALYST           20
MANAGER           30
MANAGER           10
CLERK             30
CLERK             20

9 rows selected.

SQL> select distinct ename,job ,deptno from emp;

ENAME      JOB           DEPTNO
---------- --------- ----------
BLAKE      MANAGER           30
ALLEN      SALESMAN          30
JONES      MANAGER           20
KING       PRESIDENT         10
TURNER     SALESMAN          30
SMITH      CLERK             20
MARTIN     SALESMAN          30
JAMES      CLERK             30
WARD       SALESMAN          30
CLARK      MANAGER           10
SCOTT      ANALYST           20
ADAMS      CLERK             20
FORD       ANALYST           20
MILLER     CLERK             10

14 rows selected.

SQL> select distinct * from  emp;
```

```
     EMPNO ENAME      JOB              MGR HIREDATE         SAL       COMM     DEPTNO
---------- ---------- --------- ---------- --------- ---------- ---------- ----------
      7844 TURNER     SALESMAN         7698 08-SEP-81       1500          0         30
      7902 FORD       ANALYST          7566 03-DEC-81       3000                    20
      7654 MARTIN     SALESMAN         7698 28-SEP-81       1250       1400         30
      7788 SCOTT      ANALYST          7566 19-APR-87       3000                    20
      7369 SMITH      CLERK            7902 17-DEC-80        800                    20
      7566 JONES      MANAGER          7839 02-APR-81       2975                    20
      7876 ADAMS      CLERK            7788 23-MAY-87       1100                    20
      7900 JAMES      CLERK            7698 03-DEC-81        950                    30
      7521 WARD       SALESMAN         7698 22-FEB-81       1250        500         30
      7698 BLAKE      MANAGER          7839 01-MAY-81       2850                    30
      7782 CLARK      MANAGER          7839 09-JUN-81       2450                    10
      7934 MILLER     CLERK            7782 23-JAN-82       1300                    10
      7499 ALLEN      SALESMAN         7698 20-FEB-81       1600        300         30
      7839 KING       PRESIDENT             17-NOV-81       5000                    10

14 rows selected.

SQL>
```

16-02-2023

OPERATORS

Operators are specific predefined symbols which perform certain function.

- Arithmetic Operators (+, -, *, /)
- Comparative Operators/ Relational Operators (<, >, <= ,>=, != or <>)
- Logical Operators (AND, OR, NOT)
- Special Operators (IS, IS NOT, IN, NOT IN, BETWEEN, NOT BETWEEN, LIKE, NOT LIKE, ANY, EXIST, NOT EXIST)
- Concatenation Operators (|| →Pipe Operator)
- Set Operators (Union, Union all, Intersect, Minus)

Logical Operators

- These are used to perform operations on Boolean values returned by conditions, and logical operators are used to write multiple conditions in where clause or having clause to filter the rows or groups.

AND operator - AND operator returns a Boolean statement TRUE/ 1 if all the mentioned conditions if satisfied or else it returns Boolean statement FALSE/ 0 if any of the given condition is not satisfied.

E.g,

1) Query to get details of the employees working in department no. 30 as a clerk

Ans.

```
Select * from emp
Where deptno=30 and job='CLERK';
```

EMPNO	ENAME	JOB	MGR	HIREDATE	SAL	COMM	DEPTNO
7900	JAMES	CLERK	7698	03-DEC-81	950		30

TRUTH TABLE

A	B	AND O/P
0	0	0
0	1	0
1	0	0
1	1	1

OR Operator – It returns Boolean statement TRUE/ 1 is any one of the conditions are satisfied or else it returns Boolean statement FALSE/ 0 if all the mentioned are not satisfied

TRUTH TABLE

A	B	OR O/P
0	0	0
0	1	1
1	0	1
1	1	1

NOT Operator - It manipulates the Boolean statement returned by a condition, i.e., if condition is satisfied, NOT operator returns a Boolean statement FALSE, or else if condition is not satisfied, NOT operator returns a Boolean statement TRUE.

TRUTH TABLE

A	NOT O/P
0	1
1	0

Selection

- Selection is a process of retrieving data using both columns and rows from the tables.

Syntax;

```
Select distinct */ columnname/ expressions/ aliasing/ tablename.*
From tablename
Where <filtering condition>;
```

Order of execution;

- From clause
- Where clause
- Select clause

E.g.,

2) Query to display the details of employees working in department 30

Ans.

```
Select * from emp
Where deptno=30;
```

EMPNO	ENAME	JOB	MGR	HIREDATE	SAL	COMM	DEPTNO
7499	ALLEN	SALESMAN	7698	20-FEB-81	1600	300	30
7521	WARD	SALESMAN	7698	22-FEB-81	1250	500	30
7654	MARTIN	SALESMAN	7698	28-SEP-81	1250	1400	30
7698	BLAKE	MANAGER	7839	01-MAY-81	2850		30
7844	TURNER	SALESMAN	7698	08-SEP-81	1500	0	30
7900	JAMES	CLERK	7698	03-DEC-81	950		30

E.g.,

3) Query to get the details of employees who joined after 1981

Ans.

```
select * from emp
where hiredate>'31-dec-81';
```

EMPNO	ENAME	JOB	MGR	HIREDATE	SAL	COMM	DEPTNO
7788	SCOTT	ANALYST	7566	19-APR-87	3000		20
7876	ADAMS	CLERK	7788	23-MAY-87	1100		20
7934	MILLER	CLERK	7782	23-JAN-82	1300		10

4) Write a query to display employee name, designation, half-term salary from emp table

Ans.

```
Select ename, job, sal*6 as Half_Term_Salary
From emp;
```

ENAME	JOB	HALF_TERM_SALARY
SMITH	CLERK	4800
ALLEN	SALESMAN	9600
WARD	SALESMAN	7500
JONES	MANAGER	17850
MARTIN	SALESMAN	7500
BLAKE	MANAGER	17100
CLARK	MANAGER	14700
SCOTT	ANALYST	18000
KING	PRESIDENT	30000
TURNER	SALESMAN	9000
ADAMS	CLERK	6600

```
JAMES      CLERK                5700
FORD       ANALYST             18000
MILLER     CLERK                7800
```

5) Write a query to display details of employees with their annual salary and Half-term salary

Ans.

```
select emp.*, sal*12 as annual_sal, sal*6 as half_term_sal
From emp;
```

EMPNO	ENAME	JOB	MGR	HIREDATE	SAL	COMM	DEPTNO	ANNUAL_SAL	HALF_TERM_SAL
7369	SMITH	CLERK	7902	17-DEC-80	800		20	9600	4800
7499	ALLEN	SALESMAN	7698	20-FEB-81	1600	300	30	19200	9600
7521	WARD	SALESMAN	7698	22-FEB-81	1250	500	30	15000	7500
7566	JONES	MANAGER	7839	02-APR-81	2975		20	35700	17850
7654	MARTIN	SALESMAN	7698	28-SEP-81	1250	1400	30	15000	7500
7698	BLAKE	MANAGER	7839	01-MAY-81	2850		30	34200	17100
7782	CLARK	MANAGER	7839	09-JUN-81	2450		10	29400	14700
7788	SCOTT	ANALYST	7566	19-APR-87	3000		20	36000	18000
7839	KING	PRESIDENT		17-NOV-81	5000		10	60000	30000
7844	TURNER	SALESMAN	7698	08-SEP-81	1500	0	30	18000	9000
7876	ADAMS	CLERK	7788	23-MAY-87	1100		20	13200	6600
7900	JAMES	CLERK	7698	03-DEC-81	950		30	11400	5700
7902	FORD	ANALYST	7566	03-DEC-81	3000		20	36000	18000
7934	MILLER	CLERK	7782	23-JAN-82	1300		10	15600	7800

6) Write a query to display employee names and their designations with annual salary with 20% deduction, without using minus symbol

Ans.

```
select ename, job, sal*12*0.8 as Deducted_Salary
From emp;
```

ENAME	JOB	DEDUCTED_SALARY
SMITH	CLERK	7680
ALLEN	SALESMAN	15360
WARD	SALESMAN	12000
JONES	MANAGER	28560
MARTIN	SALESMAN	12000
BLAKE	MANAGER	27360
CLARK	MANAGER	23520
SCOTT	ANALYST	28800
KING	PRESIDENT	48000
TURNER	SALESMAN	14400
ADAMS	CLERK	10560
JAMES	CLERK	9120
FORD	ANALYST	28800
MILLER	CLERK	12480

7) Write a query to display names of employees with their salary with bonus 200 without using + symbol

Ans.

```
Select ename, sal-(-200) as Bonus
From emp;
```

ENAME	BONUS
SMITH	1000

```
ALLEN            1800
WARD             1450
JONES            3175
MARTIN           1450
BLAKE            3050
CLARK            2650
SCOTT            3200
KING             5200
TURNER           1700
ADAMS            1300
JAMES            1150
FORD             3200
MILLER           1500
```

8) Write a query to display names and designations of employs who are managers

Ans.

```
Select ename, job
From emp
Where job= 'MANAGER';
```

```
ENAME      JOB
---------- ---------
JONES      MANAGER
BLAKE      MANAGER
CLARK      MANAGER
```

9) Write a query to display details of employs who are getting salary less than 1500

Ans.

```
Select emp.*
From emp
Where sal<1500;
```

```
     EMPNO ENAME      JOB              MGR HIREDATE          SAL       COMM     DEPTNO
---------- ---------- --------- ---------- --------- ---------- ---------- ----------
      7369 SMITH      CLERK            7902 17-DEC-80        800                    20
      7521 WARD       SALESMAN         7698 22-FEB-81       1250        500         30
      7654 MARTIN     SALESMAN         7698 28-SEP-81       1250       1400         30
      7876 ADAMS      CLERK            7788 23-MAY-87       1100                    20
      7900 JAMES      CLERK            7698 03-DEC-81        950                    30
      7934   MILLER     CLERK            7782 23-JAN-82       1300                    10
```

10) Write a query to display details of employees who are getting salary less their commission

Ans,

```
Select *
From emp
Where sal<comm;
```

```
     EMPNO ENAME      JOB              MGR HIREDATE          SAL       COMM     DEPTNO
---------- ---------- --------- ---------- --------- ---------- ---------- ----------
```

```
      7654 MARTIN     SALESMAN        7698 28-SEP-81       1250       1400         30
```

17-02-2023

11) Write a Query to display details of employees who are working in department 30 and as salesman or manager

Ans.

```
Select * from emp
Where deptno=30 and (job='SALESMAN' or job='MANAGER');
```

```
     EMPNO ENAME      JOB              MGR HIREDATE          SAL       COMM     DEPTNO
---------- ---------- --------- ---------- --------- ---------- ---------- ----------
      7499 ALLEN      SALESMAN        7698 20-FEB-81       1600        300         30
      7521 WARD       SALESMAN        7698 22-FEB-81       1250        500         30
      7654 MARTIN     SALESMAN        7698 28-SEP-81       1250       1400         30
      7698 BLAKE      MANAGER         7839 01-MAY-81       2850                    30
      7844 TURNER     SALESMAN        7698 08-SEP-81       1500          0         30
```

12) Write a query to display details of employees who are not working as clerks

Ans.

```
Select * from emp
Where NOT job='CLERK';
```

```
  EMPNO ENAME      JOB              MGR HIREDATE          SAL       COMM     DEPTNO
--------- ---------- --------- ---------- --------- ---------- ---------- ----------
     7499 ALLEN      SALESMAN        7698 20-FEB-81       1600        300         30
     7521 WARD       SALESMAN        7698 22-FEB-81       1250        500         30
     7566 JONES      MANAGER         7839 02-APR-81       2975                    20
     7654 MARTIN     SALESMAN        7698 28-SEP-81       1250       1400         30
     7698 BLAKE      MANAGER         7839 01-MAY-81       2850                    30
     7782 CLARK      MANAGER         7839 09-JUN-81       2450                    10
     7788 SCOTT      ANALYST         7566 19-APR-87       3000                    20
     7839 KING       PRESIDENT            17-NOV-81       5000                    10
     7844 TURNER     SALESMAN        7698 08-SEP-81       1500          0         30
     7902 FORD       ANALYST         7566 03-DEC-81       3000                    20
```

Special Operators used in (Where or Having Clause)

IS Operators – It is a special operator used to compare only the NULL values.

Syntax;

```
Select distinct */ columnname/ expressions/ aliasing/ tablename.*
From tablename
Where columnname IS NULL;
```

E.g.,

```
Select * from emp
Where NULL=NULL;  → Error,
```

IS NOT Operator – It is used to compare NOT NULL values irrespective to their datatypes.

Syntax;

```
Select distinct */ columnname/ expressions/ aliasing/ tablename.*
From tablename
Where columnname IS NOT NULL;
```

E.g.,

```
select * from emp
where comm is not null;
```

EMPNO	ENAME	JOB	MGR	HIREDATE	SAL	COMM	DEPTNO
7499	ALLEN	SALESMAN	7698	20-FEB-81	1600	300	30
7521	WARD	SALESMAN	7698	22-FEB-81	1250	500	30
7654	MARTIN	SALESMAN	7698	28-SEP-81	1250	1400	30
7844	TURNER	SALESMAN	7698	08-SEP-81	1500	0	30

IN Operator – It is a special operator to compare multiple values with the field or a column i.e., if the value from the field matches with any of the mentioned values, then it returns a BOOLEAN statement TRUE or else it returns FALSE.

Syntax;

```
Select distinct */ columnname/ expressions/ aliasing/ tablename.*
From tablename
Where columnname IN (value1, value2…. nth value);
```

E.g.,

```
Select * from emp
Where job IN ('SALESMAN', 'MANAGER') AND deptno=30;
```

NOT IN Operator – It is a special operator which is used to compare multiple values with the field or column and it returns a BOOLEAN statement FALSE if the field value matches with any of the mentioned values and excludes the records or elseif the field value of the row is not matching with any of the value then it includes that row.

Syntax;

```
Select distinct */ columnname/ expressions/ aliasing/ tablename.*
From tablename
Where columnname NOT IN (value1, value2…. nth value);
```

E.g.,

```
Select * from emp
```

Where job NOT IN ('ANALYST', 'PRESIDENT');

EMPNO	ENAME	JOB	MGR	HIREDATE	SAL	COMM	DEPTNO
7369	SMITH	CLERK	7902	17-DEC-80	800		20
7499	ALLEN	SALESMAN	7698	20-FEB-81	1600	300	30
7521	WARD	SALESMAN	7698	22-FEB-81	1250	500	30
7566	JONES	MANAGER	7839	02-APR-81	2975		20
7654	MARTIN	SALESMAN	7698	28-SEP-81	1250	1400	30
7698	BLAKE	MANAGER	7839	01-MAY-81	2850		30
7782	CLARK	MANAGER	7839	09-JUN-81	2450		10
7844	TURNER	SALESMAN	7698	08-SEP-81	1500	0	30
7876	ADAMS	CLERK	7788	23-MAY-87	1100		20
7900	JAMES	CLERK	7698	03-DEC-81	950		30
7934	MILLER	CLERK	7782	23-JAN-82	1300		10

13) Write a query to display the details of the employees who are working in department 30 or 20 and joined in year 1981

Ans.

Select * from emp
Where deptno IN (30, 20) and hiredate >= '1-JAN=81' and hiredate<='31-DEC-81';

EMPNO	ENAME	JOB	MGR	HIREDATE	SAL	COMM	DEPTNO
7499	ALLEN	SALESMAN	7698	20-FEB-81	1600	300	30
7521	WARD	SALESMAN	7698	22-FEB-81	1250	500	30
7566	JONES	MANAGER	7839	02-APR-81	2975		20
7654	MARTIN	SALESMAN	7698	28-SEP-81	1250	1400	30
7698	BLAKE	MANAGER	7839	01-MAY-81	2850		30
7844	TURNER	SALESMAN	7698	08-SEP-81	1500	0	30
7900	JAMES	CLERK	7698	03-DEC-81	950		30
7902	FORD	ANALYST	7566	03-DEC-81	3000		20

BETWEEN Operator – It is a special operator that which includes the rows that which contains a field value within the mentioned ranges of lower value and upper value. The range's datatype might be a text or number or a date.

Syntax;

Select distinct */ columnname/ expressions/ aliasing/ tablename.*
From tablename
Where columnname BETWEEN lower_value and upper_value;

E.g.,

Select * from emp
Where sal BETWEEN (1100 and 3000);

EMPNO	ENAME	JOB	MGR	HIREDATE	SAL	COMM	DEPTNO
7499	ALLEN	SALESMAN	7698	20-FEB-81	1600	300	30
7521	WARD	SALESMAN	7698	22-FEB-81	1250	500	30
7566	JONES	MANAGER	7839	02-APR-81	2975		20

```
  7654 MARTIN     SALESMAN        7698 28-SEP-81       1250       1400         30
  7698 BLAKE      MANAGER         7839 01-MAY-81       2850                    30
  7782 CLARK      MANAGER         7839 09-JUN-81       2450                    10
  7788 SCOTT      ANALYST         7566 19-APR-87       3000                    20
  7844 TURNER     SALESMAN        7698 08-SEP-81       1500          0         30
  7876 ADAMS      CLERK           7788 23-MAY-87       1100                    20
  7902 FORD       ANALYST         7566 03-DEC-81       3000                    20
  7934 MILLER     CLERK           7782 23-JAN-82       1300                    10
```

20-02-2023

14) Write a query to display details of employees who joined in 1982

Ans.

```
Select * from emp
Where hiredate between '01-jan-1982' and '31-dec-1982';
```

```
     EMPNO ENAME      JOB              MGR HIREDATE         SAL       COMM     DEPTNO
---------- ---------- --------- ---------- --------- ---------- ---------- ----------
      7934 MILLER     CLERK           7782 23-JAN-82       1300                    10
```

Also some examples, While mentioning BETWEEN for characters, it excludes the upper-value and only includes value previous to the upper value.

```
Select ename from emp
Where ename between 'A' and 'F';

Select ename from emp
Where ename between 'A' and 'G';

Select ename from emp
Where ename between 'AA' and 'AE';
```

NOT BETWEEN Operator – It is a special operator used to get the rows or records that which contains the values out of the mentioned ranges i.e., the records within the mentioned ranges will be excluded and record that contains out of range values will be included.

Syntax;

```
Select distinct */ columnname/ expressions/ aliasing/ tablename.*
From tablename
Where columnname NOT BETWEEN lower_value and upper_value;
```

15) Write a query to display details of employees except the persons getting salary from 1000 to 3000

Ans.

```
Select * from emp
Where sal NOT BETWEEN 1000 and 3000;
```

```
     EMPNO ENAME      JOB              MGR HIREDATE         SAL       COMM     DEPTNO
---------- ---------- --------- ---------- --------- ---------- ---------- ----------
```

```
      7369 SMITH      CLERK           7902 17-DEC-80        800                    20
      7839 KING       PRESIDENT            17-NOV-81       5000                    10
      7900 JAMES      CLERK           7698 03-DEC-81        950                    30
```

E.g.

```
Select ename from emp
Where ename not between 'C' and 'M';
```

```
ENAME
----------
SMITH
ALLEN
WARD
MARTIN
BLAKE
SCOTT
TURNER
ADAMS
MILLER
```

LIKE Operator – It is s special operator used to compare patterns. '_' represents a single character in LIKE operators. '%' represents 0 characters or multiple characters in LIKE operator. Can be used to compare Strings, Numbers or Dates.

Syntax;

```
Select distinct */ columnname/ expressions/ aliasing/ tablename.*
From tablename
Where columnname LIKE 'pattern' ESCAPE 'char';
```

E.g.

```
Select ename from emp
Where ename LIKE 'A____';
```

```
ENAME
----------
ALLEN
ADAMS
```

E.g.

```
Select ename from emp
Where ename LIKE '_A%';
```

```
ENAME
----------
WARD
MARTIN
JAMES
```

E.g.

```
Select ename from emp
Where ename LIKE '%A%';
```

```
ENAME
----------
ALLEN
WARD
MARTIN
BLAKE
CLARK
ADAMS
JAMES
```

E.g.

Select ename from emp
Where ename LIKE '%A%A%';

```
ENAME
----------
ADAMS
```

21-02-2023

16) Query to get details of employees whose employee number end with digit 8

Ans.

Select * from emp
Where empno like '%8';

```
     EMPNO ENAME      JOB              MGR HIREDATE         SAL       COMM     DEPTNO
---------- ---------- --------- ---------- --------- ---------- ---------- ----------
      7698 BLAKE      MANAGER         7839 01-MAY-81       2850                    30
      7788 SCOTT      ANALYST         7566 19-APR-87       3000                    20
```

Other examples,

Select * from emp
Where hiredate like '%81';

```
     EMPNO ENAME      JOB              MGR HIREDATE         SAL       COMM     DEPTNO
---------- ---------- --------- ---------- --------- ---------- ---------- ----------
      7499 ALLEN      SALESMAN        7698 20-FEB-81       1600        300         30
      7521 WARD       SALESMAN        7698 22-FEB-81       1250        500         30
      7566 JONES      MANAGER         7839 02-APR-81       2975                    20
      7654 MARTIN     SALESMAN        7698 28-SEP-81       1250       1400         30
      7698 BLAKE      MANAGER         7839 01-MAY-81       2850                    30
      7782 CLARK      MANAGER         7839 09-JUN-81       2450                    10
      7839 KING       PRESIDENT            17-NOV-81       5000                    10
      7844 TURNER     SALESMAN        7698 08-SEP-81       1500          0         30
      7900 JAMES      CLERK           7698 03-DEC-81        950                    30
      7902 FORD       ANALYST         7566 03-DEC-81       3000                    20
```

Select * from emp
Where hiredate like '%FEB%';

```
     EMPNO ENAME      JOB              MGR HIREDATE         SAL       COMM     DEPTNO
---------- ---------- --------- ---------- --------- ---------- ---------- ----------
      7499 ALLEN      SALESMAN        7698 20-FEB-81       1600        300         30
      7521 WARD       SALESMAN        7698 22-FEB-81       1250        500         30
```

ESCAPE Operator

Escape operator is a clause used in LIKE operator to convert the operators used in LIKE such as '_' or '%' into a regular/normal character and supports to search those characters. Three main escape characters used are (?, $, !).

PID	PNAME
6	A_B_D
1	jerry
2	ALEX _PANDEY
3	ram
4	jack
5	ROSE%MERRY

E.g.,

```
Select * from person
Where pname like '%?_%' escape '?';
```

```
Select * from person
Where pname like '%?_%?_%' escape '?';
```

```
Select * from person
Where pname like '%?%%' escape '?';
```

NOTE: *One escape clause works only for one condition.*

```
Select * from person
Where pname like '%?_%' escape '?' or '%?%%' escape '?';
```

Or

```
Select * from person
Where pname like '%$_%' escape '$' or '%?%%' escape '?';
```

```
Select * from person
Where pname like '%A_%' escape 'A';
```

NOT LIKE Operator – It is a special operator used to compare a pattern with a field of a table and it includes the rows that which does not matches with the mentioned pattern or else it excludes when the pattern is matched with the field of a row.

E.g.,

```
Select * from person
Where pname not like '%?_%' escape '?';
```

```
PID                  PNAME
-------------------  --------------------
1                    jerry
3                    ram
4                    jack
5                    ROSE%MERRY
```

SET Operators

Are used to combine the result set of multiple select statement queries into a single result set and returns the output.

Types of set operators are;

- UNION
- UNION ALL
- INTERSECT
- MINUS

P = [1,2,3,4], Q = [6, 4, 3, 5]

E.g.,

P UNION Q → [1,2,3,4,5,6]

P UNION ALL Q → [1,2,3,4,6,4,3,5]

P INTERSECT Q → [3,4]

P MINUS Q → [1,2]

Q MINUS P → [6,5]

22-02-2023

Syntax;

```
Select distinct */ columnname/ expressions/ aliasing/ tablename.*
From tablename
Where <filtering condition>
Set operator
Select distinct */ columnname/ expressions/ aliasing/ tablename.*
From tablename
Where <filtering condition>;
```

UNION Set Operator - UNION is a set operator that combines result set of multiple select statement queries and returns only unique records and eliminates all the duplicate records or values.

- Union Operator sorts the record by default according to the primary column in the result column.

UNION ALL Set Operator – UNION ALL is a set operator that combines the result sets of multiple select statement queries as it is and returns as a output.

- Union All returns a result set without any elimination of duplicates and no sorting of records is performed.

INTERCEPT Operator – INTERCEPT is a set operator that combines multiple select statement queries and returns only common records of rows as a result.

MINUS – MINUS is a set operator used to combine result set of multiple select statement queries and eliminates all the common records and returns only the left over records from primary query.

Rules to combine result set of queries using set operators;

- All the queries should return same number of columns.
- All the queries should return same ordered datatypes.

E.g.,

```
Select ename, job, deptno from emp
Where job='SALESMAN';
```

```
ENAME      JOB           DEPTNO
---------- --------- ----------
ALLEN      SALESMAN          30
WARD       SALESMAN          30
MARTIN     SALESMAN          30
TURNER     SALESMAN          30
```

```
Select job, deptno, ename from emp
Where deptno=30;
```

```
JOB           DEPTNO ENAME
--------- ---------- ----------
SALESMAN          30 ALLEN
SALESMAN          30 WARD
SALESMAN          30 MARTIN
MANAGER           30 BLAKE
SALESMAN          30 TURNER
CLERK             30 JAMES
```

```
Select ename, job, deptno from emp
```

```
Where job='SALESMAN'
Union
Select ename, job, deptno from emp
Where deptno=30;
```

```
ENAME      JOB           DEPTNO
---------- --------- ----------
ALLEN      SALESMAN          30
BLAKE      MANAGER           30
JAMES      CLERK             30
MARTIN     SALESMAN          30
TURNER     SALESMAN          30
WARD       SALESMAN          30
```

```
Select ename, job, deptno from emp
Where job='SALESMAN'
Union all
Select ename, job, deptno from emp
Where deptno=30;
```

```
ENAME      JOB           DEPTNO
---------- --------- ----------
ALLEN      SALESMAN          30
WARD       SALESMAN          30
MARTIN     SALESMAN          30
TURNER     SALESMAN          30
ALLEN      SALESMAN          30
WARD       SALESMAN          30
MARTIN     SALESMAN          30
BLAKE      MANAGER           30
TURNER     SALESMAN          30
JAMES      CLERK             30
```

```
Select ename, job, deptno from emp
Where job='SALESMAN'
intersect
Select ename, job, deptno from emp
Where deptno=30;
```

```
ENAME      JOB           DEPTNO
---------- --------- ----------
ALLEN      SALESMAN          30
MARTIN     SALESMAN          30
TURNER     SALESMAN          30
WARD       SALESMAN          30
```

```
Select ename, job, deptno from emp
Where job='SALESMAN'
Minus
Select ename, job, deptno from emp
Where deptno=30;
```

```
no row selected
```

```
Select ename, job, deptno from emp
```

```
Where deptno=30
Minus
Select ename, job, deptno from emp
Where job='SALESMAN';

ENAME      JOB           DEPTNO
---------- --------- ----------
BLAKE      MANAGER           30
JAMES      CLERK             30
```

PIPE Operator

- It is used to perform concatenation between one or more data.

E.g.,

```
Select 'Hi '||ename from emp;


'HI'||ENAME
-------------
Hi SMITH
Hi ALLEN
Hi WARD
Hi JONES
Hi MARTIN
Hi BLAKE
Hi CLARK
Hi SCOTT
Hi KING
Hi TURNER
Hi ADAMS
Hi JAMES
Hi FORD
Hi MILLER
```

```
Select 'Hi Mr. '||ename||' congratulations, you have been selected as
'||job||' for our company with your unique ID '||empno||' for a pay of
'||sal|| 'per month ' from emp;


'HIMR.'||ENAME||'CONGRATULATIONS,YOUHAVEBEENSELECTEDAS'||JOB||'FOROURCOMPANYWITH
--------------------------------------------------------------------------------
with your unique ID 7788 for a pay of 3000per month

Hi Mr. KING congratulations, you have been selected as PRESIDENT for our company
 with your unique ID 7839 for a pay of 5000per month

Hi Mr. TURNER congratulations, you have been selected as SALESMAN for our compan
y with your unique ID 7844 for a pay of 1500per month

Hi Mr. ADAMS congratulations, you have been selected as CLERK for our company wi
th your unique ID 7876 for a pay of 1100per month


'HIMR.'||ENAME||'CONGRATULATIONS,YOUHAVEBEENSELECTEDAS'||JOB||'FOROURCOMPANYWITH
--------------------------------------------------------------------------------
```

```
Hi Mr. JAMES congratulations, you have been selected as CLERK for our company wi
th your unique ID 7900 for a pay of 950per month

Hi Mr. FORD congratulations, you have been selected as ANALYST for our company w
ith your unique ID 7902 for a pay of 3000per month

Hi Mr. MILLER congratulations, you have been selected as CLERK for our company w
ith your unique ID 7934 for a pay of 1300per month
```

Order by Clause

- Order by clause is used to perform sorting of the rows/records either in ascending or descending according to the mentioned field or column in order by clause.
- Order by clause sorts the rows/record in ascending order by default.

Syntax;

```
Select distinct */ columnname/ expressions/ aliasing/ tablename.*
From tablename
Where <filtering condition>
Order by columnname's/ expressions ASC/DESC;
```

Order of execution;

- From clause
- Where clause
- Order by clause
- Select clause

E.g.,

```
Select * from emp
Order by sal;
```

EMPNO	ENAME	JOB	MGR	HIREDATE	SAL	COMM	DEPTNO
7369	SMITH	CLERK	7902	17-DEC-80	800		20
7900	JAMES	CLERK	7698	03-DEC-81	950		30
7876	ADAMS	CLERK	7788	23-MAY-87	1100		20
7521	WARD	SALESMAN	7698	22-FEB-81	1250	500	30
7654	MARTIN	SALESMAN	7698	28-SEP-81	1250	1400	30
7934	MILLER	CLERK	7782	23-JAN-82	1300		10
7844	TURNER	SALESMAN	7698	08-SEP-81	1500	0	30
7499	ALLEN	SALESMAN	7698	20-FEB-81	1600	300	30
7782	CLARK	MANAGER	7839	09-JUN-81	2450		10
7698	BLAKE	MANAGER	7839	01-MAY-81	2850		30
7566	JONES	MANAGER	7839	02-APR-81	2975		20
7788	SCOTT	ANALYST	7566	19-APR-87	3000		20
7902	FORD	ANALYST	7566	03-DEC-81	3000		20
7839	KING	PRESIDENT		17-NOV-81	5000		10

24-02-2023

17) Write a query to display details of employees joined in deptno 30 and display their names in ascending order

Ans.

```
Select * from emp
Where deptno=30
Order by ename;
```

```
     EMPNO ENAME      JOB              MGR HIREDATE         SAL       COMM     DEPTNO
---------- ---------- --------- ---------- --------- ---------- ---------- ----------
      7499 ALLEN      SALESMAN         7698 20-FEB-81       1600        300         30
      7698 BLAKE      MANAGER          7839 01-MAY-81       2850                    30
      7900 JAMES      CLERK            7698 03-DEC-81        950                    30
      7654 MARTIN     SALESMAN         7698 28-SEP-81       1250       1400         30
      7844 TURNER     SALESMAN         7698 08-SEP-81       1500          0         30
      7521 WARD       SALESMAN         7698 22-FEB-81       1250        500         30
```

18) Write a query to display the deptno in descending order and the names in each dept in ascending order

Ans.

```
Select ename, deptno from emp
Order by deptno desc, ename;
```

```
ENAME          DEPTNO
---------- ----------
ALLEN              30
BLAKE              30
JAMES              30
MARTIN             30
TURNER             30
WARD               30
ADAMS              20
FORD               20
JONES              20
SCOTT              20
SMITH              20
CLARK              10
KING               10
MILLER             10
```

E.g.,

```
Select ename, job, deptno from emp
Order by deptno desc, job, ename;
```

```
ENAME      JOB           DEPTNO
---------- --------- ----------
JAMES      CLERK             30
BLAKE      MANAGER           30
ALLEN      SALESMAN          30
MARTIN     SALESMAN          30
TURNER     SALESMAN          30
WARD       SALESMAN          30
FORD       ANALYST           20
SCOTT      ANALYST           20
ADAMS      CLERK             20
SMITH      CLERK             20
JONES      MANAGER           20
MILLER     CLERK             10
CLARK      MANAGER           10
KING       PRESIDENT         10
```

FUNCTIONS

- It is a set of codes to perform a specific task.
- There are two types of predefined functions in SQL, they are; Single row functions and Multi row functions.

Single Row functions

- Single row functions accept n number of rows and returns n number of outputs as a result, i.e., the single row function accepts each row from the table and executes for each row individually and returns an output.

Number SRF/ Numeric SRF

1) mod() - Mod() accepts two arguments, which is dividend and divisor and performs the division operation and returns the remainder as output.

Syntax;

mod(dividend, divisor) → `mod(15,4) => 3`

E.g.,

```
Select mod(10,2) from emp;
```

```
MOD(10,2)
----------
         0
         0
         0
         0
         0
         0
         0
         0
         0
         0
         0
        0
         0
         0
```

Select mod(10,2) from dual;

```
MOD(10,2)
----------
         0
```

```
Select * from emp
Where mod(sal, 2)=0;
```

```
     EMPNO ENAME      JOB              MGR HIREDATE         SAL       COMM     DEPTNO
---------- ---------- --------- ---------- --------- ---------- ---------- ----------
      7369 SMITH      CLERK           7902 17-DEC-80        800                    20
```

```
      7499 ALLEN      SALESMAN        7698 20-FEB-81       1600        300         30
      7521 WARD       SALESMAN        7698 22-FEB-81       1250        500         30
      7654 MARTIN     SALESMAN        7698 28-SEP-81       1250       1400         30
      7698 BLAKE      MANAGER         7839 01-MAY-81       2850                    30
      7782 CLARK      MANAGER         7839 09-JUN-81       2450                    10
      7788 SCOTT      ANALYST         7566 19-APR-87       3000                    20
      7839 KING       PRESIDENT            17-NOV-81       5000                    10
      7844 TURNER     SALESMAN        7698 08-SEP-81       1500          0         30
      7876 ADAMS      CLERK           7788 23-MAY-87       1100                    20
      7900 JAMES      CLERK           7698 03-DEC-81        950                    30
      7902 FORD       ANALYST         7566 03-DEC-81       3000                    20
      7934 MILLER     CLERK           7782 23-JAN-82       1300                    10
```

27-02-2023

2) Sqrt() - It accept sone argument and it returns square root value of the mentioned number in the argument.

Syntax;

Sqrt(number) →Square root of the number is returned.

E.g.,

```
Select sqrt(4) from dual;

  SQRT(4)
---------
        2

Select sqrt(8) from dual;

  SQRT(8)
----------
2.82842712
```

3) Power() – It accepts two arguments, base-value and exponent. The power functions returns a number raised to given to given exponent value as a result.

Syntax;

Power(base-value, exponent)→ returns base times base up to exponent number of times.

E.g.,

```
Select power(2,3) from dual;

POWER(2,3)
----------
         8

Select sal, power(sal,2) from emp;

SAL POWER(SAL,2)
       ---------- ------------
       800       640000
      1600      2560000
      1250      1562500
      2975      8850625
      1250      1562500
```

```
      2850       8122500
      2450       6002500
      3000       9000000
      5000      25000000
      1500       2250000
      1100       1210000
       950        902500
      3000       9000000
  1690000
```

4) Abs() – This function accepts a number value as an argument and it returns its absolute value as the result.

Syntax;

abs(number)→ returns the absolute value of the number.

E.g.,

```
Select abs(-10) from dual;

ABS(-10)
--------
      10
```

5) Round() - Round function accepts a number and performs round off to the value according to mentioned index positions and returns as a result.

Index positions of round function is optional, the default value of index position will be 0.

O - 4 → 0

5 - 9 → 1

Syntax;

Round(number, index)

E.g.,

```
Round(234.233);
```

+0 → Everything to the right is converted to zero.

```
 2  3  4  .2  3  3
-3 -2 -1  0  1  2
```

```
234.0000

Round(23874.323, -3);
```

+1 → Everything to the right is converted to zero.

2 3 8 7 4. 3 2 3

5 -4 -3 -2 -1 0 1 2

```
24000.000

SQL> select round(234.2342, -2) from dual;

ROUND(234.2342,-2)
------------------
               200

SQL> select round(9786.23) from dual;

ROUND(9786.23)
--------------
          9786

SQL> select round(723.234,-3) from dual;

ROUND(723.234,-3)
-----------------
             1000
```

6) Trunc() – It accepts a number and it rounds off the value to the nearest lowest value and returns as a result.

0 – 9 → 0

E.g.,

```
SQL> select trunc(2837.23, -2) FROM DUAL;

TRUNC(2837.23,-2)
-----------------
                2800

SQL> select trunc(99.99) from dual;

TRUNC(99.99)
------------
          99

SQL> select trunc(673.23, -2) FROM DUAL;

TRUNC(673.23,-2)
----------------
                600

SQL> select trunc(648.745, -3) from dual;

TRUNC(648.745,-3)
-----------------
                   0
```

2) Character SRF/ Alphabetic SRF

Case Manipulation

1) Upper() – Upper function accepts a character or a string as an argument and converts all lower-case alphabets into uppercase alphabets and returns as result.

E.g.,

```
select upper('KartHik') from dual;

UPPER('
-------
KARTHIK
```

Select upper(pname) from persons;

```
UPPER(PNAME)
--------------------
A_B_D
JERRY
ALEX_PANDEY
RAM
JACK
ROSE%MERRY
```

28-02-2023

E.g.,

```
Select * from person
Where pname='RAM';

      No rows selected

Select * from person
Where upper(pname)='RAM';
PID                  PNAME
---------------------   --------------------
3                    ram
```

2) Lower() – Lower function accepts a character or a string as an argument and converts all the uppercase alphabets into lowercase alphabets.

E.g.,

```
Select lower('PYSPIDERS') from dual;
Lower('PY
---------
pyspiders

Select ename, lower(ename) from emp;

ENAME      LOWER(ENAM
---------- ----------
```

```
SMITH      smith
ALLEN      allen
WARD       ward
JONES      jones
MARTIN     martin
BLAKE      blake
CLARK      clark
SCOTT      scott
KING       king
TURNER     turner
ADAMS      adams
JAMES      james
FORD       ford
MILLER     miller
Select * from emp
```

3) Initcap() – This function accepts character or strings and converts initial character of each word into an uppercase alphabet and all other characters into lowercase alphabets.

Syntax;

Initcap(char/string/column)

E.g.,

Select initcap('rYAN rEYNOLDS') from dual;

```
INITCAP ('RYA
-------------
Ryan Reynolds
```

Select ename, initcap(ename) from emp;

```
ENAME      INITCAP(EN
---------- ----------
SMITH      Smith
ALLEN      Allen
WARD       Ward
JONES      Jones
MARTIN     Martin
BLAKE      Blake
CLARK      Clark
SCOTT      Scott
KING       King
TURNER     Turner
ADAMS      Adams
JAMES      James
FORD       Ford
MILLER     Miller
```

Character Manipulation

4) Length() – Length is a function that accepts one argument and it counts the number of characters present in a string or number or date types of data and returns a positive integer

number as a output. Length considers white space as a character and also includes in count as ‘ ’ also has an ASCII value.

Syntax;

Length(char/string/number/date/column)

E.g.,

select length (‘He LLo H O LLA’) from dual;

```
LENGTH('HELLOHOLLA')
--------------------
                  14
```

Select length(12048023) from dual;

```
LENGTH(12048023)
----------------
               8
```

19) Write a query to display names of employees whose name contains more than 4 characters

Ans.

```
Select * from emp
Where length(ename) > 4;
```

EMPNO	ENAME	JOB	MGR	HIREDATE	SAL	COMM	DEPTNO
7369	SMITH	CLERK	7902	17-DEC-80	800		20
7499	ALLEN	SALESMAN	7698	20-FEB-81	1600	300	30
7566	JONES	MANAGER	7839	02-APR-81	2975		20
7654	MARTIN	SALESMAN	7698	28-SEP-81	1250	1400	30
7698	BLAKE	MANAGER	7839	01-MAY-81	2850		30
7782	CLARK	MANAGER	7839	09-JUN-81	2450		10
7788	SCOTT	ANALYST	7566	19-APR-87	3000		20
7844	TURNER	SALESMAN	7698	08-SEP-81	1500	0	30
7876	ADAMS	CLERK	7788	23-MAY-87	1100		20
7900	JAMES	CLERK	7698	03-DEC-81	950		30
7934	MILLER	CLERK	7782	23-JAN-82	1300		10

20) Write a query to display names of employees whose name contains odd number of characters

Ans.

```
Select * from emp
Where mod(length(ename), 2) = 1;
```

EMPNO	ENAME	JOB	MGR	HIREDATE	SAL	COMM	DEPTNO
7369	SMITH	CLERK	7902	17-DEC-80	800		20
7499	ALLEN	SALESMAN	7698	20-FEB-81	1600	300	30
7566	JONES	MANAGER	7839	02-APR-81	2975		20
7698	BLAKE	MANAGER	7839	01-MAY-81	2850		30
7782	CLARK	MANAGER	7839	09-JUN-81	2450		10
7788	SCOTT	ANALYST	7566	19-APR-87	3000		20
7876	ADAMS	CLERK	7788	23-MAY-87	1100		20
ES	CLERK	7698	03-DEC-81	950		30	

5) Reverse() – It is a function that reverses the string and returns the reversed string.

Syntax;
Reverse(char/string/column)

6) Concat() – It accepts two arguments and it performs concatenation between the data present in two arguments and returns as a result.

Syntax;
Concat(arg1, arg2)

E.g.,
Select concat('Oh','Hello') from dual;

```
CONCAT(
-------
OhHello
```

Select empno, ename, concat(empno, ename) from emp;

```
     EMPNO ENAME      CONCAT(EMPNO,ENAME)
---------- ---------- --------------------------------------------------
      7369 SMITH      7369SMITH
      7499 ALLEN      7499ALLEN
      7521 WARD       7521WARD
      7566 JONES      7566JONES
      7654 MARTIN     7654MARTIN
      7698 BLAKE      7698BLAKE
      7782 CLARK      7782CLARK
      7788 SCOTT      7788SCOTT
      7839 KING       7839KING
      7844 TURNER     7844TURNER
      7876 ADAMS      7876ADAMS
      7900 JAMES      7900JAMES
      7902 FORD       7902FORD
      7934 MILLER     7934MILLER
```

Select empno, ename, concat(empno||' ', ename) from emp;

```
     EMPNO ENAME      CONCAT(EMPNO||'',ENAME)
---------- ---------- ------------------------------
      7369 SMITH      7369 SMITH
      7499 ALLEN      7499 ALLEN
      7521 WARD       7521 WARD
      7566 JONES      7566 JONES
      7654 MARTIN     7654 MARTIN
      7698 BLAKE      7698 BLAKE
      7782 CLARK      7782 CLARK
      7788 SCOTT      7788 SCOTT
      7839 KING       7839 KING
      7844 TURNER     7844 TURNER
      7876 ADAMS      7876 ADAMS
      7900 JAMES      7900 JAMES
      7902 FORD       7902 FORD
      7934 MILLER     7934 MILLER
```

```
Select concat(12, 45)*2 from dual;
CONCAT(12,45)*2
---------------
           2490
```

7) Replace() – It a function that accepts 3 arguments and it eliminates the mentioned substring from the original string and replaces with new string or character.

Syntax;

Replace(original string/column, 'substring', 'newstring')

E.g.,

```
select replace('MANGLORE', 'M') from dual;

REPLACE
-------
ANGLORE

select replace('MANGLORE', 'M', 'B') from dual;

REPLACE(
--------
BANGLORE

select replace('MANGLORE', 'MANG', 'CUDDA') from dual;

REPLACE('
---------
CUDDALORE

select replace('MANAMAGALORE', 'M', 'B') from dual;

REPLACE('MAN
------------
BANABAGALORE
```

1-03-2023

21) Write a query to display the number of times occurrence of character 'a' in each name of employees

Ans.

```
Select length(ename) - length(replace(ename, 'A')) from emp;

ENAME      LENGTH(ENAME)-LENGTH(REPLACE(ENAME,'A',''))
---------- -------------------------------------------
SMITH                                                0
ALLEN                                                1
WARD                                                 1
JONES                                                0
MARTIN                                               1
BLAKE                                                1
CLARK                                                1
SCOTT                                                0
KING                                                 0
```

```
TURNER                                                    0
ADAMS                                                     2
JAMES                                                     1
FORD                                                      0
MILLER                                                    0
```

8) Translate() – It is a function that which accepts three arguments and replaces a sequence of characters mentioned second argument with the sequence of characters in third arguments.

Syntax;

Translate(original string/column, 'Sequence of characters to be eliminated', 'sequence of characters to be replaced with')

E.g.,

```
select translate('ENCAPSULATION', 'AEIOU', '12345') from dual;

TRANSLATE('EN
-------------
2NC1PS5L1T34N

select translate('ENCAPSULATION', 'AEIOU', '1') from dual;

TRANSLATE('EN
-------------
NC1PSL1TN


select translate('ENCAPSULATION', 'AEIOU', '') from dual;

T
-

select replace(translate('ENCAPSULATION', 'AEIOU', '1'), '1') from dual;

REPLACE
-------
NCPSLTN


select length('ENCAPSULATION') -
length(replace(translate('ENCAPSULATION', 'AEIOU', '1'), '1')
) from dual;

LENGTH('ENCAPSULATION')-LENGTH(REPLACE(TRANSLATE('ENCAPSULATION','AEIOU','1'),'1'))
-----------------------------------------------------------------------------------
                                                                                  6

select translate('YQWHTIY','YQWHTI','KARTHI') FROM DUAL;

TRANSLA
-------
KARTHIK
```

9) Instr() – This function accepts four arguments, they are argument 1 accepts original string, argument accepts substring to be searched in original string, argument 3 accepts position and it specifies from where the searching is to be started. If the position value if positive it searched from left to right and if the position value is negative it searches from right to left. The argument 4 accepts the occurrence.

- Instr() is used to search a substring or character in it, and if the character or substring is present it returns the index position or else it returns 0.
- Default value of position and occurrence will be 1.

Syntax;

Instr(Original string/column, 'Substring', 'position', 'occurrence')

```
+ve
------------------------>
B   A   N   A   N   A
1   2   3   4   5   6
-6  -5  -4  -3  -2  -1
<-----------------------
                   -ve
```

```
SQL> select instr('BANANA', 'A') from dual;

INSTR('BANANA','A')
-------------------
                  2

SQL> select instr('BANANA', 'NA', 3) from dual;

INSTR('BANANA','NA',3)
----------------------
                     3

SQL> select instr('BANANA', 'NA') from dual;

INSTR('BANANA','NA')
--------------------
                   3

SQL> select instr('BANANA', 'ANA', -1) from dual;

INSTR('BANANA','ANA',-1)
------------------------
                       4

SQL> select instr('BANANA', 'ANA', 1,2) from dual;

INSTR('BANANA','ANA',1,2)
-------------------------
                        4

SQL> select instr('BANANA', 'ANA', -1, 2) from dual;

INSTR('BANANA','ANA',-1,2)
--------------------------
                         2
```

```
SQL>
```

22) Write a query to display names that which contains character ‘A’ twice in them
Ans.

```
select ename from emp
where length(ename) -length(replace(ename, ‘A’)) = 2;
```

or

```
select ename from emp
where instr(ename, ‘A’, 1,2) > 0 AND  instr(ename, ‘A’, 1, 3) = 0;
```

or

```
select ename from emp
where instr(ename, ‘A’, 1,2) > 0 AND  instr(ename, ‘A’, 1, 3) = 0;
ENAME
----------
ADAMS
```

10) Substr() - It is used to extract a part of string from mentioned original string.
Substr() accepts three arguments, argument one accepts original string or a column that which contains characters or strings, argument two accepts position and position defines that extracting position of substring and argument three accepts length and it specifies number of characters to be extracted from the mentioned position.

- Length is an optional argument and if the length is not mentioned, it extracts everything from the right of the mentioned position index.

P	Y	S	P	I	D	E	R	S
1	2	3	4	5	6	7	8	9
-9	-8	-7	-6	-5	-4	-3	-2	-1

---

23) Write a query to display the details of employees whose name starts and ends with same character
Ans.

```
Select * from emp
Where substr(ename, 1,1) = substr(ename, -1);

no rows selected
```

11) Trim() - Trim function is used to eliminate the selected characters either from leading or trailing or from both the sides of the original string. By default, trim function eliminates white spaces from both the sides of a original string.

Syntax;

Trim(leading/trailing/both 'char' from original string/column)

E.g.,

```
select trim(' PY  SPIDERS  ') from dual;

TRIM('PYSPI
-----------
PY  SPIDERS

SQL> select trim ('s' from 'SPIDERS') from dual;

TRIM('S
-------
SPIDERS

SQL> select trim('S' from 'SPIDERS') from dual;

TRIM(
-----
PIDER

SQL> select trim(leading 'S' from 'SPIDERS') from dual;

TRIM(L
------
PIDERS

SQL> select trim(trailing 'S' from 'SPIDERS') from dual;

TRIM(T
------
SPIDER

SQL> select trim('S' from 'ssSSSPIDERSssSSSss') from dual;

TRIM('S'FROM'SSSSS
------------------
ssSSSPIDERSssSSSss

SQL> select trim('S' from 'SSSssPIDERsssSSS') from dual;

TRIM('S'FR
----------
ssPIDERsss
```

Date SRF

1) Add_Months() – Add month function accepts two arguments, argument 1 accepts date value and argument two accepts a number value that specifies number of months to be

added or subtracted from the original date and it returns the date added/ subtracted with selected number of months.

Syntax;

Add_months(date/column, number value)

```
E.g.,
SQL> select add_months('8-mar-2023', 5) from dual;

ADD_MONTH
---------
08-AUG-23

SQL> select add_months('8-mar-2023', -5) from dual;

ADD_MONTH
---------
08-OCT-22

SQL> select add_months('8-mar-2023', 9.7) from dual;

ADD_MONTH
---------
08-DEC-23

SQL>
```

2) Months_between() – It is function that is used to count the number of months between mentioned ranges of dates and returns a number as a result.

Syntax;

Months_between(date/column, date/column)

E.g.,

```
SQL> select months_between('23-feb-2023', '22-mar-2022') from dual;

MONTHS_BETWEEN('23-FEB-2023','22-MAR-2022')
-------------------------------------------
                                 11.0322581

SQL>  select months_between('22-mar-2022', '23-feb-2023') from dual;

MONTHS_BETWEEN('22-MAR-2022','23-FEB-2023')
-------------------------------------------
                                -11.032258
```

10-03-2023

24) Write a query to display details of employees who are having 40 years of experience till date

Ans.

```
Select ename, hiredate, sysdate, months_between(sysdate, hiredate)/12 from
emp
Where months_between(sysdate, hiredate)>40*12;
```

```
ENAME      HIREDATE  SYSDATE   MONTHS_BETWEEN(SYSDATE,HIREDATE)/12
---------- --------- --------- ----------------------------------
SMITH      17-DEC-80 10-MAR-23                          42.2322755
ALLEN      20-FEB-81 10-MAR-23                          42.0575443
WARD       22-FEB-81 10-MAR-23                           42.052168
JONES      02-APR-81 10-MAR-23                          41.9392647
MARTIN     28-SEP-81 10-MAR-23                          41.4527056
BLAKE      01-MAY-81 10-MAR-23                          41.8586196
CLARK      09-JUN-81 10-MAR-23                          41.7537809
KING       17-NOV-81 10-MAR-23                          41.3156088
TURNER     08-SEP-81 10-MAR-23                           41.506469
JAMES      03-DEC-81 10-MAR-23                          41.2699099
FORD       03-DEC-81 10-MAR-23                          41.2699099
MILLER     23-JAN-82 10-MAR-23                          41.1328131
```

3) Last_day() – It accepts a date value as an argument and checks the year and month in the given date and returns the respective last date according to the month.

Syntax;

Last_day(date)

E.g.,

Select last_day('10-feb-2000') from dual;

```
LAST_DAY(
---------
29-FEB-00
```

Select last_day('10-feb-2020') from dual;

```
LAST_DAY(
---------
29-FEB-20
```

4) Extract() – It is a function used to fetch day or month or year individually from a date datatype and returns in the form of numbers.

Syntax;

Extract(day/month/year from to_date('date')/column/sysdate)

E.g.,

```
SQL> select extract(day from to_date('21-jun-1987')) from dual;

EXTRACT(DAYFROMTO_DATE('21-JUN-1987'))
--------------------------------------
                                    21

SQL> select extract(month from to_date('21-jun-1987')) from dual;
```

```
EXTRACT(MONTHFROMTO_DATE('21-JUN-1987'))
----------------------------------------
                                       6

SQL> select extract(year from to_date('21-jun-1987')) from dual;


EXTRACT(YEARFROMTO_DATE('21-JUN-1987'))
---------------------------------------
                                  1987

SQL> select ename, hiredate, extract(year from hiredate) from emp;

ENAME      HIREDATE  EXTRACT(YEARFROMHIREDATE)
---------- --------- -------------------------
SMITH      17-DEC-80                      1980
ALLEN      20-FEB-81                      1981
WARD       22-FEB-81                      1981
JONES      02-APR-81                      1981
MARTIN     28-SEP-81                      1981
BLAKE      01-MAY-81                      1981
CLARK      09-JUN-81                      1981
SCOTT      19-APR-87                      1987
KING       17-NOV-81                      1981
TURNER     08-SEP-81                      1981
ADAMS      23-MAY-87                      1987
JAMES      03-DEC-81                      1981
FORD       03-DEC-81                      1981
MILLER     23-JAN-82                      1982
```

25) Write a query to display details of employees joined on even months

Ans.

```
Select * from emp
Where mod(extract(month from hiredate), 2)=0;
```

EMPNO	ENAME	JOB	MGR	HIREDATE	SAL	COMM	DEPTNO
7369	SMITH	CLERK	7902	17-DEC-80	800		20
7499	ALLEN	SALESMAN	7698	20-FEB-81	1600	300	30
7521	WARD	SALESMAN	7698	22-FEB-81	1250	500	30
7566	JONES	MANAGER	7839	02-APR-81	2975		20
7782	CLARK	MANAGER	7839	09-JUN-81	2450		10
7788	SCOTT	ANALYST	7566	19-APR-87	3000		20
7900	JAMES	CLERK	7698	03-DEC-81	950		30
7902	FORD	ANALYST	7566	03-DEC-81	3000		20

26) Write a query to display employee details who joined on leap year

Ans.

```
Select * from emp
Where (mod(extract(year from hiredate), 400)=0 or mod(extract(year from
hiredate), 4)=0) and mod(extract(year from hiredate), 100)<>0;
```

EMPNO	ENAME	JOB	MGR	HIREDATE	SAL	COMM	DEPTNO
7369	SMITH	CLERK	7902	17-DEC-80	800		20

27) Write a query to display names & joining dates of employees who joined in 2nd half of the month

Ans.

```
Select * from emp
Where extract(day from hiredate)>15;
```

```
     EMPNO ENAME      JOB              MGR HIREDATE         SAL       COMM     DEPTNO
---------- ---------- --------- ---------- --------- ---------- ---------- ----------
      7369 SMITH      CLERK           7902 17-DEC-80        800                    20
      7499 ALLEN      SALESMAN        7698 20-FEB-81       1600        300         30
      7521 WARD       SALESMAN        7698 22-FEB-81       1250        500         30
      7654 MARTIN     SALESMAN        7698 28-SEP-81       1250       1400         30
      7788 SCOTT      ANALYST         7566 19-APR-87       3000                    20
      7839 KING       PRESIDENT            17-NOV-81       5000                    10
      7876 ADAMS      CLERK           7788 23-MAY-87       1100                    20
      7934 MILLER     CLERK           7782 23-JAN-82       1300                    10
```

13-03-2023

4) General SRF - It consists of only two functions, which are Nvl() and Nvl2(), (Null value logic).

1) Nvl() - nvl() functions accepts two arguments if the argument 1 contains NULL value, then the function returns Argument 2 value as a result.
- If argument 1 is NOT NULL then it returns the same value present in argument 1 as a result.

Syntax;
Nvl(arg1, arg2)
Arg --> value/column/expression

If Arg 1 = NULL, then O/P = Arg 2
If Arg 1 = NOT NULL, then O/P = Arg 1

E.g.,
```
Select nvl(null, 1) from dual;
```

```
NVL(NULL,1)
-----------
          1
```

```
Select nvl(10,1) from dual;
```

```
NVL(10,1)
---------
       10
```

```
Select comm, nvl(comm, 0) from emp;
```

```
      COMM NVL(COMM,0)
---------- -----------
                     0
       300         300
       500         500
```

```
                0
      1400      1400
                0
                0
                0
                0
         0      0
                0
                0
                0
                0
```

Select sal, comm, sal+nvl(comm, 0) from emp;

```
       SAL       COMM SAL+NVL(COMM,0)
---------- ---------- ---------------
       800                        800
      1600        300            1900
      1250        500            1750
      2975                       2975
      1250       1400            2650
      2850                       2850
      2450                       2450
      3000                       3000
      5000                       5000
      1500          0            1500
      1100                       1100
       950                        950
      3000                       3000
      1300                       1300
```

2) Nvl2() - It accepts three arguments, if argument 1 contains a NULL value, then the nvl2() function returns argument 3 as output, and if the argument 1 is NOT NULL value, then the nvl2() function returns argument 2 as the output.

Syntax;

Nvl2(arg1, arg2, arg3)

Arg --> value/ column/ expression

If Arg 1 = NULL, then O/P = Arg 3
If Arg 1 = NOT NULL, then O/P = Arg 2

E.g.,

Select nvl2(null, 1, 0) from dual;

```
NVL2(NULL,1,0)
--------------
             0
```

Select nvl2(10, 1, 0) from dual;

```
NVL2(NULL,1,0)
--------------
             1
```

```
Select comm, nvl2(comm, 1, 0) from emp;
```

```
      COMM NVL2(COMM,1,0)
---------- --------------
                        0
       300               1
       500               1
                         0
      1400               1
                         0
                         0
                         0
                         0
         0               1
                         0
                         0
                         0
                         0
```

Conversion SRF

To convert one type of data into another type of data.

14-03-2023

1) to_date() – It converts different types of date formats into Oracle date format, 'DD-MMM-YY' or 'DD-MMM-YYYY' , and stores in the table.

Syntax;

```
to_date(String/ Number, 'format')
```

E.g.,

```
Select to_date('31-03-2023', 'dd-mm-yy') from dual;
```

```
TO_DATE('
---------
31-MAR-23
```

```
Select to_date('2020&21#02', 'yy&dd#mm') from dual;

Select to_date(12202304, 'mm-yy-dd') from dual;
```

2) to_Number() – It converts a string that contains only digits and a special character, into number datatype.

Syntax;

```
to_Number(String, 'format')
```

E.g.,

```
Select to_number('1,23,12,234', '99999999999') from dual;

TO_NUMBER('1,23,12,234','99999999999')
--------------------------------------
                              12312234

Select to_number('12,234.232342', '99999999.999999999') from dual;

TO_NUMBER('12,234.232342','99999999.999999999')
-----------------------------------------------
                                   12234.2323
```

3) to_char()

Day – Return week day of mentioned data
DD – Returns data in number format
MM – Returns month in number format
MON – Returns starting 3 character of month name
MONTH – Returns full month name
YY – Returns last 2 digits of year in number format
YYYY – Returns four digits of the year
Year – Returns year in string format
SP – converts number to string
TH – Returns number with standard units

Syntax;

```
to_char(Date/ Number, 'format')
```

E.g.,

```
Select to_char(to_date('24-mar-2022'), 'day') from dual;

TO_CHAR(T
---------
thursday

Select to_char(sysdate, ' day, dd, mm, mon, month, yy, yyyy, year') from
dual;

TO_CHAR(SYSDATE,'DAY  DD  MM  MON  MONTH  YY  YYYY  YEAR')
-----------------------------------------------------------
tuesday  14  03  mar  march   23  2023 twenty twenty-three

select to_char(sysdate, 'ddsp mmsp yysp yyyysp') from dual;

TO_CHAR(SYSDATE,'DDSPMMSPYYSPYYYYSP')
```

```
--------------------------------------------------
fourteen three twenty-three two thousand twenty-three
```

```
select to_char(sysdate, 'ddth mmth yyth yyyyth') from dual;

TO_CHAR(SYSDATE,'DDTH
---------------------
14th 03rd 23rd 2023rd
```

15-03-2023

28) Query to display employees joined on a Friday

Ans.

```
Select to_char(hiredate, 'day'), length(to_char(hiredate, 'day')) from emp;

TO_CHAR(H LENGTH(TO_CHAR(HIREDATE,'DAY'))
--------- --------------------------------
wednesday                                9
friday                                   9
sunday                                   9
thursday                                 9
monday                                   9
friday                                   9
tuesday                                  9
sunday                                   9
tuesday                                  9
tuesday                                  9
saturday                                 9
thursday                                 9
thursday                                 9
saturday                                 9
```

```
Select * from emp
Where to_char(hiredate, 'day') = 'friday   ';

     EMPNO ENAME      JOB              MGR HIREDATE         SAL       COMM     DEPTNO
---------- ---------- --------- ---------- --------- ---------- ---------- ----------
      7499 ALLEN      SALESMAN        7698 20-FEB-81       1600        300         30
      7698 BLAKE      MANAGER         7839 01-MAY-81       2850                    30
```

```
Select * from emp
Where trim(to_char(hiredate, 'day')) = 'friday';

     EMPNO ENAME      JOB              MGR HIREDATE         SAL       COMM     DEPTNO
---------- ---------- --------- ---------- --------- ---------- ---------- ----------
      7499 ALLEN      SALESMAN        7698 20-FEB-81       1600        300         30
      7698 BLAKE      MANAGER         7839 01-MAY-81       2850                    30
```

Number to Character

```
Select to_char(12423, '99,99,999$') from dual;

TO_CHAR(124
-----------
    $12,423
```

```
Select to_char(231312.235, '99,99,999.99999') from dual;

TO_CHAR(231312.2
----------------
  2,31,312.23500
```

4) ASCII() – It accepts the characters and returns their respective ASCII numbers.

Syntax;
Ascii('char')

E.g.,
```
Select ascii('A') from dual;

ASCII('A')
----------
        65

Select ascii('Adams') from dual;

ASCII('ADAMS')
--------------
            65

Select ascii(1) from dual;

  ASCII(1)
----------
        49

Select ascii('1') from dual;

  ASCII(1)
----------
        49
```

5) CHR() – It accepts any ASCII number and returns the respective character.

Syntax;
Char(ASCII number)

E.g.,
```
Select chr(74) from dual;

C
-
J
```

Multi Row functions/ Group Functions/ Aggregate functions

MRF accepts n number of groups and returns n number of outputs, i.e., MR functions accepts multiple rows as a group and executes and returns a single output for each group of rows.

- Multi-Row functions / Group functions are not allowed in Where clause.
- Multi-Row functions / Group functions does not consider NULL values.

Syntax;

```
Select (group function)/ Group by columns (or) expressions
From Tablename
Where <filtering conditions>;
```

List of Functions;

- Count(*/ Column/ expression)
- Max(column/expression)
- Min(column/expression)
- Sum(column/expression)
- Avg(column/expression)

1) Count() – It is a group function that counts the number of rows and values from each group and returns a positive integer number as an Output.

Syntax;

```
Count(*/ Column/ expression)
```

E.g.,

```
Select count(*) from emp;
```

```
  COUNT(*)
----------
        14
```

```
Select count(ename) from emp;
```

```
COUNT(ENAME)
------------
          14
```

```
Select count(comm) from emp;
```

```
COUNT(COMM)
-----------
```

```
         4
```

29) Write a query to display the number of employees working at deptno 30

Ans,
Select count(*)
From emp
Where deptno='30';

```
  COUNT(*)
----------
         6
```

30) Write a query to display the number of employees joined in the year 1981
Ans.

```
Select count(*)
From emp
Where to_char(hiredate, 'yyyy') = '1981';
```

```
  COUNT(*)
----------
10
```

31) Write a query to display the number of employees working as clerk and number of clerks getting commission
Ans.

```
Select count(*), count(comm) from emp
Where job='CLERK';
```

```
  COUNT(*) COUNT(COMM)
---------- -----------
         4           0
```

or

```
Select count(*) from emp
Where job='SALESMEN' and comm>0;
```

```
  COUNT(*)
----------
         0
```

20-02-2023

32) Write a query to display number of employees who have salary more than 2000
Ans.

```
Select count(*) from emp
```

```
Where sal>2000;

  COUNT(*)
----------
         6
```

33) Write a query to display number of clerks in emp table and display number of clerks getting salary more than 1000

Ans.

```
Select count(*) from emp
Where job = 'CLERK' AND sal>1000;

  COUNT(*)
----------
         2

Select count(*) from (
Select * from emp
Where job = 'CLERK'
intersect
Select * from emp
Where sal>1000);

  COUNT(*)
----------
         2
```

2) max() – Max function is a group function that accepts fields or columns as arguments and checks all the values and returns the maximum value from the field as the result.

Syntax;

```
Max(column/Expressions)
```

E.g.,

```
Select max(sal), max(hiredate), max(ename) from emp;

  MAX(SAL) MAX(HIRED MAX(ENAME)
---------- --------- ----------
      5000   -MAY-87 WARD
```

3) min() – Min() function is a group function that accepts fields or columns as arguments and checks all the values and returns the minimum value from the field as the result.

Syntax;

```
min(column/Expressions)
```

E.g.,

```
Select min(sal), max(hiredate), max(ename) from emp;
```

```
  MIN(SAL) MAX(HIRED MAX(ENAME)
---------- --------- ----------
       800 -MAY-87 WARD
```

4) Sum() – Sum function accepts only number datatype fields and adds all the values in the field and returns a total value as result.

Syntax;

```
Sum(column/ expression)
```

E.g.,

```
Select sum(comm), sum(sal) from emp;
```

```
SUM(COMM)   SUM(SAL)
---------- ----------
  29025
```

5) Avg() – It accepts a field adds all the values and divides the total value by the number of values added and returns the average value as a result.

Syntax;

```
Avg(column/ expression)
```

```
E.g., select avg(comm) from emp;
```

```
AVG(COMM)
----------
       550
```

```
Select sum(comm)/sum(nvl2(comm, 1,0)) from emp;
```

Group by clause

It is used to make groups according to the required fields that is group by clause all the rows that which contains similar values in the mentioned field and made as a group.

Syntax;

```
Select [distinct] group functions / group by column/ expressions
From tablename
[where <filtering conditions>]
Group by columns/expressions/SRF
Order by group functions/columns/expressions    asc/desc;
```

Order of execution;

- From clause
- Where clause
- Group by clause
- Order by clause
- Select

21-03-2023

34) A query to display the number of employees In each department

Ans.

```
Select deptno, count(*) from emp
Group by deptno;
```

```
    DEPTNO   COUNT(*)
---------- ----------
        30          6
        20          5
        10          3
```

35) Query to display the number of employees working in each designation in each department

Ans.

```
Select job, deptno, count(*) from emp
Group by job, deptno;
```

```
JOB           DEPTNO   COUNT(*)
--------- ---------- ----------
MANAGER           20          1
PRESIDENT         10          1
CLERK             10          1
SALESMAN          30          4
ANALYST           20          2
MANAGER           30          1
MANAGER           10          1
CLERK             30          1
CLERK             20          2
```

36) Write a query to display the number of employees working as salesman, manager, analyst

Ans.

```
Select job, count(*) from emp
Where job='SALESMAN' or job='MANAGER' or job='ANALYST'
Group by job;
```

Or

```
Select job, count(*) from emp
```

```
Where job in ('SALESMEN',  'MANAGER', 'ANALYST')
Group by job;
```

```
JOB         COUNT(*)
--------- ----------
SALESMAN           4
MANAGER            3
ANALYST            2
```

37) Write a query to display number of employees joined in the year 1981

Ans.

```
Select extract(year from hiredate), count(*) from emp
Where extract(year from hiredate) = '1981'
Group by extract(year from hiredate);
```

```
EXTRACT(YEARFROMHIREDATE)   COUNT(*)
------------------------- ----------
                     1981         10
```

38) Write a query to display number of employees joined in each year

Ans.

```
Select extract(year from hiredate), count(*) from emp
Group by extract(year from hiredate);
```

```
EXTRACT(YEARFROMHIREDATE)   COUNT(*)
------------------------- ----------
                     1982          1
                     1987          2
                     1980          1
                     1981         10
```

39) Write a query to display repeated salaries from emp

Ans.

```
Select sal, count(*) from emp
Group by sal
Having count(*) > 1;
```

```
       SAL   COUNT(*)
---------- ----------
      1250          2
      3000          2
```

40) Write a query to display non-repeated designation

Ans.

```
Select job, count(*) from emp
Group by job
Having count(*) < 2;
```

```
JOB         COUNT(*)
--------- ----------
```

```
PRESIDENT          1
```

24_03_2023

Having Clause

Having clause is used to filter groups and it filters group by group.

Syntax;

```
Select [distinct] group functions / group by column/ expressions
From tablename
[where <filtering conditions>]
Group by columns/expressions/SRF
Having <group filtering conditions>
Order by group functions/columns/expressions    asc/desc;
```

Difference between where clause and having clause

Where	Having
1. Where clause is used to filter rows. 2. Where clause does not allow group functions. 3. Group by clause is not mandatory to perform filtration of rows using where clause. 4. Where clause executes before group by clause.	1. Having clause is used to filter groups. 2. Having clause supports and allows group functions in it. 3. Group by clause is mandatory before performing having clause. 4. Having clause executes after group by clause.

41) Query to display deptno where the number of people working are more than 3

Ans.

```
Select deptno, count(*) from emp
Group by deptno
Having count(*) > 3;
```

```
    DEPTNO   COUNT(*)
---------- ----------
        30          6
        20          5
```

42) Write a query to display the year where number of employees joined more than 5

Ans.

```
Select extract(year from hiredate) from emp
Group by extract(year from hiredate)
```

```
Having count(*) > 5;

EXTRACT(YEARFROMHIREDATE)
-------------------------
                     1981
```

43) Write a query to display names of designations where at least 2 persons working in each designation

Ans.

```
Select job from emp
Group by job
Having count(*)>=2;

JOB
---------
CLERK
SALESMAN
MANAGER
ANALYST
```

44) Write a query to display number of employees getting salary more than 1000 in each department and display only department number where at least 3 employees are working

Ans.

```
Select deptno, count(*) from emp
Where sal > 1000
Group by deptno
Having count(*) >= 3;

    DEPTNO   COUNT(*)
---------- ----------
        30          5
        20          4
        10          3
```

45) Write a query to display total salary paid for each designation should be more than 4000

Ans.

```
Select job, sum(sal) from emp
Group by job
Having sum(sal) > 4000;

JOB         SUM(SAL)
--------- ----------
CLERK           4150
SALESMAN        5600
PRESIDENT       5000
MANAGER         8275
ANALYST         6000
```

46) Write a query to display deptno whose maximum salary is more than 4000

Ans.

```
Select deptno, max(sal) from emp
```

```
Group by deptno
Having max(sal) > 4000;
```

```
    DEPTNO   MAX(SAL)
---------- ----------
        10       5000
```

47) Write a query to display average salary paid for each designation and display only the designation which gets more than 2000

Ans.

```
Select job, avg(sal) from emp
Group by job
Having avg(sal) > 2000;
```

```
JOB         AVG(SAL)
--------- ----------
PRESIDENT       5000
MANAGER   2758.33333
ANALYST         3000
```

Sub Query

A query written inside of another query is called as a sub query.

Types of Sub Queries;

Single Row Sub Query – If inner query of a query returns only one value as a result, then it is known as Single Row sub query.

E.g.,

48) Query to display details of employees from the same deptno as Allen's

Ans.

```
Select * from emp  ← Outer Query
Where deptno = (select deptno from emp where ename = 'ALLEN'); ←Inner query
```

49) Write a query to display details of employees who are getting salary more than miller

Ans.

```
Select * from emp
Where sal > ( select sal from emp where ename = 'MILLER');
```

```
     EMPNO ENAME      JOB              MGR HIREDATE         SAL       COMM     DEPTNO
---------- ---------- --------- ---------- --------- ---------- ---------- ----------
      7499 ALLEN      SALESMAN         7698 20-FEB-81       1600        300         30
      7566 JONES      MANAGER          7839 02-APR-81       2975                    20
      7698 BLAKE      MANAGER          7839 01-MAY-81       2850                    30
      7782 CLARK      MANAGER          7839 09-JUN-81       2450                    10
      7788 SCOTT      ANALYST          7566 19-APR-87       3000                    20
      7839 KING       PRESIDENT             17-NOV-81       5000                    10
```

```
      7844 TURNER     SALESMAN        7698 08-SEP-81       1500          0         30
      7902 FORD       ANALYST         7566 03-DEC-81       3000                    20
```

50) Write a query to display names of employees who joined in the same year as Turner
Ans.

```
Select ename from emp
Where extract(year from hiredate) = (select extract(year from hiredate) from
emp Where ename = 'TURNER');
```

```
ENAME
----------
ALLEN
WARD
JONES
MARTIN
BLAKE
CLARK
KING
TURNER
JAMES
FORD
```

51) Write a query to display details of employees who are working as Martin's designation and getting salary more than smith
Ans.

```
Select * from emp
Where job = (select job from emp where ename = 'MARTIN') and sal > (select
sal from emp where ename = 'SMITH');
```

```
     EMPNO ENAME      JOB              MGR HIREDATE          SAL       COMM     DEPTNO
---------- ---------- --------- ---------- --------- ---------- ---------- ----------
      7499 ALLEN      SALESMAN        7698 20-FEB-81       1600        300         30
      7521 WARD       SALESMAN        7698 22-FEB-81       1250        500         30
      7654 MARTIN     SALESMAN        7698 28-SEP-81       1250       1400         30
      7844 TURNER     SALESMAN        7698 08-SEP-81       1500          0         30
```

52) Write a query to display details of employees reporting for Blake
Ans.

```
Select * from emp
Where mgr = (select empno from emp where ename ='BLAKE');
```

```
     EMPNO ENAME      JOB              MGR HIREDATE          SAL       COMM     DEPTNO
---------- ---------- --------- ---------- --------- ---------- ---------- ----------
      7499 ALLEN      SALESMAN        7698 20-FEB-81       1600        300         30
      7521 WARD       SALESMAN        7698 22-FEB-81       1250        500         30
      7654 MARTIN     SALESMAN        7698 28-SEP-81       1250       1400         30
      7844 TURNER     SALESMAN        7698 08-SEP-81       1500          0         30
      7900 JAMES      CLERK           7698 03-DEC-81        950                    30
```

27_03_2023

53) Write a query to display the employees whose starting character of the name is same as the ending character of the names of employees who are working in empno is 7900

Ans.

```
Select * from emp
Where substr(ename, 1, 1) = (select substr(ename, -1) from emp where empno = 7900);
```

54) Write a query to display employees working in the same department of ford and whose first letter of their names has an odd ASCII value

Ans.

```
Select ename from emp
Where deptno = (select deptno from emp where ename='FORD')
And mod(ASCII(ename), 2) = 1:
```

Multi Row Sub Query

Multi-Row subquery in a subquery inner query returns more than 1 value as a result, then it is known as a Multi Row subquery. Multi-Row subquery cannot be compared with the normal comparative operators, the results of a multi row subquery should be compared with combination of comparative and special subquery operators (ALL/ANY).

55) Write a query to display details of employees, who are getting salary more than the salary of the employees working in deptno 20

Ans.

```
Select * from emp
Where sal>all(select sal from emp where deptno=20);
```

EMPNO	ENAME	JOB	MGR	HIREDATE	SAL	COMM	DEPTNO
7839	KING	PRESIDENT		17-NOV-81	5000		10

56) Write a query to display details of employees who are working in the same deptno as King or Allen and working in the designation same as Turner or Jones

Ans.

```
Select * from emp
Where deptno = any(select deptno from emp where ename in ('KING' ,'ALLEN'))
and job = any(select job from emp where ename in ('TURNER', 'JONES'));
```

EMPNO	ENAME	JOB	MGR	HIREDATE	SAL	COMM	DEPTNO
7499	ALLEN	SALESMAN	7698	20-FEB-81	1600	300	30
7521	WARD	SALESMAN	7698	22-FEB-81	1250	500	30
7654	MARTIN	SALESMAN	7698	28-SEP-81	1250	1400	30

```
      7844 TURNER     SALESMAN        7698 08-SEP-81       1500          0         30
      7698 BLAKE      MANAGER         7839 01-MAY-81       2850                    30
      7782 CLARK      MANAGER         7839 09-JUN-81       2450                    10
```

57) Write a query to display details of employees who joined in any of the years of Smith or Scott and getting salary more than James and Smith

Ans.

```
Select * from emp
Where extract(year from hiredate) = any(select extract(year from hiredate)
from emp where ename = ANY('SMITH', 'SCOTT')) and sal > all(select sal from
emp where ename = ANY('JAMES', 'SMITH'));
```

```
     EMPNO ENAME      JOB              MGR HIREDATE          SAL       COMM     DEPTNO
---------- ---------- --------- ---------- --------- ---------- ---------- ----------
      7876 ADAMS      CLERK           7788 23-MAY-87       1100                    20
      7788 SCOTT      ANALYST         7566 19-APR-87       3000                    20
```

```
select * from emp
where sal > any(select sal from emp where ename in
('JONES','ALLEN','MARTIN'));
```

```
    EMPNO ENAME      JOB              MGR HIREDATE          SAL       COMM     DEPTNO
--------- ---------- --------- ---------- --------- ---------- ---------- ----------
     7839 KING       PRESIDENT               17-NOV-81       5000                    10
     7902 FORD       ANALYST         7566 03-DEC-81       3000                    20
     7788 SCOTT      ANALYST         7566 19-APR-87       3000                    20
     7566 JONES      MANAGER         7839 02-APR-81       2975                    20
     7698 BLAKE      MANAGER         7839 01-MAY-81       2850                    30
     7782 CLARK      MANAGER         7839 09-JUN-81       2450                    10
     7499 ALLEN      SALESMAN        7698 20-FEB-81       1600        300         30
     7844 TURNER     SALESMAN        7698 08-SEP-81       1500          0         30
     7934 MILLER     CLERK           7782 23-JAN-82       1300                    10
```

28_03_2023

Pseudo Columns/ Ghost Columns

- Pseudo columns are the columns created during creation of a table.
- rowID 18 character address assigned for each and every row of a table.
- rowID is always unique and fixed.
- rowNum is a field that which provides or assigns a sequential numbers for each row of a table.
- rowNum works only on 3 conditions.
    1) Rownum = 1 ; This condition returns the firs record of a table.
    2) Rownum <= number or rownum < number ; This condition includes the records from the first row of a table to the mentioned ranges.
    3) Rownum >=1 ; This condition returns all the records from the table.

58) Query to get the last record of a table

Ans.

```
Select * from emp
Where rowid = (select max(rowid) from emp);
```

EMPNO	ENAME	JOB	MGR	HIREDATE	SAL	COMM	DEPTNO
7934	MILLER	CLERK	7782	23-JAN-82	1300		10

59) Query to get the first record of a table

Ans.

```
Select * from emp
Where rowid = (select min(rowid) from emp);
```

EMPNO	ENAME	JOB	MGR	HIREDATE	SAL	COMM	DEPTNO
7369	SMITH	CLERK	7902	17-DEC-80	800		20

Or

```
Select * from emp
Where rownum = 1;
```

EMPNO	ENAME	JOB	MGR	HIREDATE	SAL	COMM	DEPTNO
7369	SMITH	CLERK	7902	17-DEC-80	800		20

60) Query to get the first 5 records from table emp

Ans.

```
Select * from emp
Where rownum<=5;
```

Or

```
Select * from emp
Where rownum<6;
```

EMPNO	ENAME	JOB	MGR	HIREDATE	SAL	COMM	DEPTNO
7369	SMITH	CLERK	7902	17-DEC-80	800		20
7499	ALLEN	SALESMAN	7698	20-FEB-81	1600	300	30
7521	WARD	SALESMAN	7698	22-FEB-81	1250	500	30
7566	JONES	MANAGER	7839	02-APR-81	2975		20
7654	MARTIN	SALESMAN	7698	28-SEP-81	1250	1400	30

61) Query to display every record of the table emp

Ans.

```
Select * from emp
```

```
Where rownum>=1;
```

EMPNO	ENAME	JOB	MGR	HIREDATE	SAL	COMM	DEPTNO
7369	SMITH	CLERK	7902	17-DEC-80	800		20
7499	ALLEN	SALESMAN	7698	20-FEB-81	1600	300	30
7521	WARD	SALESMAN	7698	22-FEB-81	1250	500	30
7566	JONES	MANAGER	7839	02-APR-81	2975		20
7654	MARTIN	SALESMAN	7698	28-SEP-81	1250	1400	30
7698	BLAKE	MANAGER	7839	01-MAY-81	2850		30
7782	CLARK	MANAGER	7839	09-JUN-81	2450		10
7788	SCOTT	ANALYST	7566	19-APR-87	3000		20
7839	KING	PRESIDENT		17-NOV-81	5000		10
7844	TURNER	SALESMAN	7698	08-SEP-81	1500	0	30
7876	ADAMS	CLERK	7788	23-MAY-87	1100		20
7900	JAMES	CLERK	7698	03-DEC-81	950		30
7902	FORD	ANALYST	7566	03-DEC-81	3000		20
7934	MILLER	CLERK	7782	23-JAN-82	1300		10

In-Line Sub Query or Multi-Column Sub Query

In-Line subqueries are used to create a temporary table with the result-sets of select statement.

Syntax;

```
Select distinct/*/columns/expressions/tablename.* from (select statement)
reference _tablename
Where <filtering-conditions>
Group by
Order by
```

62) Query to display the 4th record of emp table

Ans.

```
Select * from (select * from emp where rownum<=4 order by rownum desc)
Where rownum = 1;
```

EMPNO	ENAME	JOB	MGR	HIREDATE	SAL	COMM	DEPTNO
7566	JONES	MANAGER	7839	02-APR-81	2975		20

63) Standard query to display the nth record from any table

Ans.

```
Select * from (select * from tablename where rownum<=n order by rownum desc)
Where rownum = 1;
```

Or

```
Select * from (Select rownum alias_name_for_rownum, emp.* from emp)
Where  alias_name_for_rownum = n;
```

E.g.,

```
Select * from (select rownum rn, emp.* from emp)
Where rn = 4;
```

RN	EMPNO	ENAME	JOB	MGR	HIREDATE	SAL	COMM	DEPTNO
4	7566	JONES	MANAGER	7839	02-APR-81	2975		20

64) Write a query to display the second half of the emp table

Ans.

```
Select * from (select rownum rn, emp.* from emp)
Where rn > (select max(rn)/2 from ( select rownum rn, emp.* from emp));
```

or

```
Select * from (select rownum rn, emp.* from emp)
Where rn > (select count(*)/2 from emp);
```

RN	EMPNO	ENAME	JOB	MGR	HIREDATE	SAL	COMM	DEPTNO
8	7788	SCOTT	ANALYST	7566	19-APR-87	3000		20
9	7839	KING	PRESIDENT		17-NOV-81	5000		10
10	7844	TURNER	SALESMAN	7698	08-SEP-81	1500	0	30
11	7876	ADAMS	CLERK	7788	23-MAY-87	1100		20
12	7900	JAMES	CLERK	7698	03-DEC-81	950		30
13	7902	FORD	ANALYST	7566	03-DEC-81	3000		20
14	7934	MILLER	CLERK	7782	23-JAN-82	1300		10

65) Write a query to display records of emp table by eliminating last 3 records

Ans.

```
Select rownum, emp.* from emp
where rownum <= ((select count(*) from emp)-3);
```

ROWNUM	EMPNO	ENAME	JOB	MGR	HIRE-DATE	SAL	COMM	DEPTNO
1	7369	SMITH	CLERK	7902	17-DEC-80	800		20
2	7499	ALLEN	SALESMAN	7698	20-FEB-81	1600	300	30
3	7521	WARD	SALESMAN	7698	22-FEB-81	1250	500	30
4	7566	JONES	MANAGER	7839	02-APR-81	2975		20
5	7654	MARTIN	SALESMAN	7698	28-SEP-81	1250	1400	30
6	7698	BLAKE	MANAGER	7839	01-MAY-81	2850		30
7	7782	CLARK	MANAGER	7839	09-JUN-81	2450		10
8	7788	SCOTT	ANALYST	7566	19-APR-87	3000		20
9	7839	KING	PRESIDENT		17-NOV-81	5000		10
10	7844	TURNER	SALESMAN	7698	08-SEP-81	1500	0	30
11	7876	ADAMS	CLERK	7788	23-MAY-87	1100		20

66) Write a query to display even records from emp table

Ans.

```
Select * from ( select rownum rn, emp.* from emp)
Where mod(rn, 2) = 0;
```

```
        RN      EMPNO ENAME      JOB              MGR HIREDATE         SAL       COMM     DEPTNO
---------- ---------- ---------- --------- ---------- --------- ---------- ---------- --------
--
         2       7499 ALLEN      SALESMAN        7698 20-FEB-81       1600        300         30
         4       7566 JONES      MANAGER         7839 02-APR-81       2975                    20
         6       7698 BLAKE      MANAGER         7839 01-MAY-81       2850                    30
         8       7788 SCOTT      ANALYST         7566 19-APR-87       3000                    20
        10       7844 TURNER     SALESMAN        7698 08-SEP-81       1500          0         30
        12       7900 JAMES      CLERK           7698 03-DEC-81        950                    30
        14       7934 MILLER     CLERK           7782 23-JAN-82       1300                    10
```

67) Write a query to display the middle record of emp table

Ans.

```
Select * from (select rownum rn, emp.*from emp)
Where rn in ((select (count(*)/2+1) from emp),( select count(*)/2 from emp));
```

```
        RN      EMPNO ENAME      JOB              MGR HIREDATE         SAL       COMM     DEPTNO
---------- ---------- ---------- --------- ---------- --------- ---------- ---------- --------
--
         7       7782 CLARK      MANAGER         7839 09-JUN-81       2450                    10
         8       7788 SCOTT      ANALYST         7566 19-APR-87       3000                    20
```

68) A query to get the middle record

Ans.

```
Select * from (select rownum rn, emp.* from emp)
Where rn in ((select count(*)/2 from emp),(select count(*)/2+1 from emp)) and
mod((select count(*) from emp),2) = 0
Union
Select * from (select rownum rn, emp.* from emp)
Where rn = (select round(count(*)/2) from emp) and mod((select count(*) from
emp),2) = 1;
```

69) Write a query to display the 3rd quarter records of emp table

Ans.

```
Select * from (select rownum rn, emp.* from emp)
Where rn > (select count(*) from emp)*0.5
And rn<=(select count(*) from emp)*0.75;
```

First quarter;

```
Select * from (select rownum rn, emp.* from emp)
Where rn > (select count(*) from emp)*0.0
And rn<=(select count(*) from emp)*0.25;
```

Creating a duplicate table.

Syntax;

```
Create table Duplicate_tablename as (select * from tablename);
```

Joins

It is a process of retrieving data from multiple tables simultaneously i.e., tables are joined together with their matched records and data is retrieved from the joined tables.

31_03_2023

Creating the relationships between 2 tables.

Synatx;

```
Create table table_name(col1 datatype constraints, col2 datatype constraints,
foreign key (col1/col2) references parent_table_name
(reference_column_name));
```

E.g.,

```
Create table branch(BID number(3) primary key, Bname varchar(10) not null,
HOB varchar(10) null);
```

```
Name                                                      Null?    Type
 ----------------------------------------------------- -------- --------------
 BID                                                    NOT NULL NUMBER(3)
 BNAME                                                  NOT NULL VARCHAR2(10)
 HOB                                                             VARCHAR2(10)
```

```
Create table std(SID number(2) primary key, Sname varchar(10) not null, BID
number(3) , foreign key(bid) references branch(bid));
```

```
Name                                                      Null?    Type
 ----------------------------------------------------- -------- -------------
 SID                                                    NOT NULL NUMBER(2)
 SNAME                                                  NOT NULL VARCHAR2(10)
 BID                                                             NUMBER(3)
```

```
Insert all
Into branch values(420, 'ECE', 'Kattapa')
Into branch values(143, 'CSE', 'Sunny')
Into branch values(840, 'ME', 'Johnny')
Into branch values(600, 'CIVIL', 'Mia')
Select * from dual;
```

```
       BID BNAME      HOB
---------- ---------- ----------
       420 ECE        Kattapa
       143 CSE        Sunny
       840 ME         Johnny
       600 CIVIL      Mia
```

```
Insert all
Into std values(1, 'Jack', 143)
Into std values(2, 'maggie', 840)
Into std values(3, 'Dani',null)
Into std values(4, 'Raju', 420)
Into std values(5, 'Rock', 840)
Into std values(6, 'ABD', null)
Select * from dual;
```

```
       SID SNAME             BID
---------- ---------- ----------
         1 Jack              143
         2 maggie            840
         3 Dani
         4 Raju              420
         5 Rock              840
         6 ABD
```

Types of Joins

Cross Join/ Cartesian Join – It is used to join multiple tables and matches each and every row of a table with all the rows of another table and returns a cross product of rows as a result.

Syntax;

```
ANSI Universal Syntax;
Select distinct */ Columns/ Expressions
From table1_name cross join table2_name
Where < filtering-condition >;

Oracle Syntax;
Select distinct */ Columns/ Expressions
From table1_name, table2_name
Where < filtering-condition >;
```

E.g.,

```
Select * from dept cross join branch;
```

```
    DEPTNO DNAME          LOC                BID BNAME      HOB
---------- -------------- ------------- ---------- ---------- ----------
        10 ACCOUNTING     NEW YORK            420 ECE        Kattapa
        10 ACCOUNTING     NEW YORK            143 CSE        Sunny
        10 ACCOUNTING     NEW YORK            840 ME         Johnny
        10 ACCOUNTING     NEW YORK            600 CIVIL      Mia
        20 RESEARCH       DALLAS              420 ECE        Kattapa
        20 RESEARCH       DALLAS              143 CSE        Sunny
        20 RESEARCH       DALLAS              840 ME         Johnny
        20 RESEARCH       DALLAS              600 CIVIL      Mia
        30 SALES          CHICAGO             420 ECE        Kattapa
        30 SALES          CHICAGO             143 CSE        Sunny
```

```
        30 SALES          CHICAGO              840 ME         Johnny
        30 SALES          CHICAGO              600 CIVIL      Mia
        40 OPERATIONS     BOSTON               420 ECE        Kattapa
        40 OPERATIONS     BOSTON               143 CSE        Sunny
        40 OPERATIONS     BOSTON               840 ME         Johnny
        40 OPERATIONS     BOSTON               600 CIVIL      Mia
```

Select * from std, branch;

```
       SID SNAME              BID        BID BNAME      HOB
---------- ---------- ---------- ---------- ---------- ----------
         1 Jack              143        420 ECE        Kattapa
         2 maggie            840        420 ECE        Kattapa
         3 Dani                         420 ECE        Kattapa
         4 Raju              420        420 ECE        Kattapa
         5 Rock              840        420 ECE        Kattapa
         6 ABD                          420 ECE        Kattapa
         1 Jack              143        143 CSE        Sunny
         2 maggie            840        143 CSE        Sunny
         3 Dani                         143 CSE        Sunny
         4 Raju              420        143 CSE        Sunny
         5 Rock              840        143 CSE        Sunny
         6 ABD                          143 CSE        Sunny
         1 Jack              143        840 ME         Johnny
         2 maggie            840        840 ME         Johnny
         3 Dani                         840 ME         Johnny
         4 Raju              420        840 ME         Johnny
         5 Rock              840        840 ME         Johnny
         6 ABD                          840 ME         Johnny
         1 Jack              143        600 CIVIL      Mia
         2 maggie            840        600 CIVIL      Mia
         3 Dani                         600 CIVIL      Mia
         4 Raju              420        600 CIVIL      Mia
         5 Rock              840        600 CIVIL      Mia
         6 ABD                          600 CIVIL      Mia
```

Inner Join – It joins multiple tables and returns only matched records from the tables as a result and excludes all the unmatched records

Syntax;

ANSI universal syntax;
Select distinct */ Columns/ Expressions
From table1_name inner join table2_name
On <joining-condition> and / or <filtering condition> ;

Oracle syntax;
Select distinct */ Columns/ Expressions
From table1_name, table2_name
Where < Joining-condition> and/or < filtering-condition >;

E.g.,
Select * from std, branch

```
Where std.bid = branch.bid;

       SID SNAME              BID        BID BNAME      HOB
---------- ---------- ---------- ---------- ---------- ----------
         1 Jack              143        143 CSE        Sunny
         2 maggie            840        840 ME         Johnny
         4 Raju              420        420 ECE        Kattapa
         5 Rock              840        840 ME         Johnny
```

1/04/2023

E.g.,

```
Select std.* from std, branch
Where std.bid = branch.bid and bname = 'ECE';

       SID SNAME              BID
---------- ---------- ----------
         4 Raju              420
```

70) Write a query to display details of employees working in the location DALLAS

Ans.

```
Select * from emp, dept
Where emp.deptno = dept.deptno and loc = 'DALLAS';

     EMPNO ENAME      JOB              MGR HIREDATE         SAL       COMM     DEPTNO     DEPTNO DNAME          LOC
---------- ---------- --------- ---------- --------- ---------- ---------- ---------- ---------- --- 
      7369 SMITH      CLERK           7902 17-DEC-80        800                    20         20 RESEARCH       DALLAS
      7566 JONES      MANAGER         7839 02-APR-81       2975                    20         20 RESEARCH       DALLAS
      7788 SCOTT      ANALYST         7566 19-APR-87       3000                    20         20 RESEARCH       DALLAS
      7876 ADAMS      CLERK           7788 23-MAY-87       1100                    20         20 RESEARCH       DALLAS
      7902 FORD       ANALYST         7566 03-DEC-81       3000                    20         20 RESEARCH       DALLAS
```

71) Write a query to display number of employees working in sales department

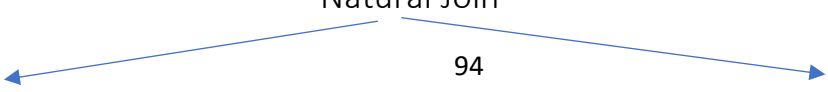
Ans.

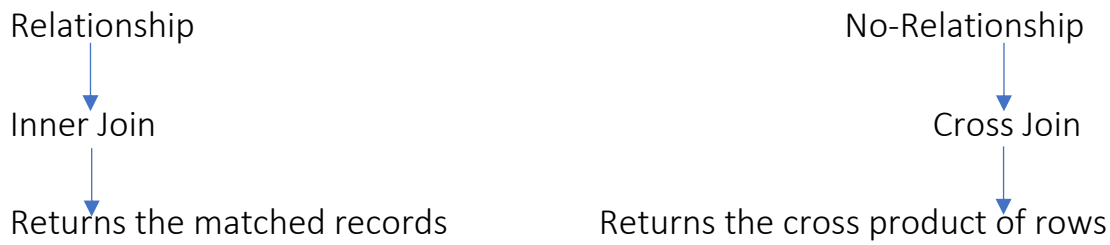
```
Select count(*) from emp, dept
Where emp.deptno  = dept.deptno and dname = 'SALES';

  COUNT(*)
----------
         6
```

Natural join

- It joins multiple tables and returns only the matched records as a result if both the tables contain a common column and common data.
- Natural join performs cross join and returns cross product of rows as a result if the tables do not contain any relationship between them.

Natural Join





Syntax;

ANSI syntax;
Select distinct */ Column/ Expressions
From table1_name natural join table2_name
On <joining-condition> and/or Where <filtering-condition>

E.g.,
Select * from std natural join branch;

BID	SID	SNAME	BNAME	HOB
143	1	Jack	CSE	Sunny
840	2	maggie	ME	Johnny
420	4	Raju	ECE	Kattapa
840	5	Rock	ME	Johnny

Select * from branch natural join dept;

BID	BNAME	HOB	DEPTNO	DNAME	LOC
420	ECE	Kattapa	10	ACCOUNTING	NEW YORK
420	ECE	Kattapa	20	RESEARCH	DALLAS
420	ECE	Kattapa	30	SALES	CHICAGO
420	ECE	Kattapa	40	OPERATIONS	BOSTON
143	CSE	Sunny	10	ACCOUNTING	NEW YORK
143	CSE	Sunny	20	RESEARCH	DALLAS
143	CSE	Sunny	30	SALES	CHICAGO
143	CSE	Sunny	40	OPERATIONS	BOSTON
840	ME	Johnny	10	ACCOUNTING	NEW YORK
840	ME	Johnny	20	RESEARCH	DALLAS
840	ME	Johnny	30	SALES	CHICAGO
840	ME	Johnny	40	OPERATIONS	BOSTON
600	CIVIL	Mia	10	ACCOUNTING	NEW YORK
600	CIVIL	Mia	20	RESEARCH	DALLAS
600	CIVIL	Mia	30	SALES	CHICAGO
600	CIVIL	Mia	40	OPERATIONS	BOSTON

Outer Join

- Outer join joins multiple tables and returns matched and unmatched records from the tables.
- Outer/inner join can be performed between tables Only When there is a relationship between tables.
  Outer join is further divided into 3 types.

Left Outer Join – It joins tables and returns matched records from both the tables and unmatched records from left table.

Syntax;

```
ANSI Syntax;
Select distinct */ column/ expression
From table1_name left outer join table2_name……
On <joining-condition> and/or <filtering-condition>
where <filtering-condition>;
```

E.g.,

```
Select * from std s left join branch b
On s.bid = b.bid;
```

```
       SID SNAME              BID        BID BNAME      HOB
---------- ---------- ---------- ---------- ---------- ---------
         1 Jack               143        143 CSE        Sunny
         2 maggie             840        840 ME         Johnny
         3 Dani               null       nullnull       null
         4 Raju               420        420 ECE        Kattapa
         5 Rock               840        840 ME         Johnny
         6 ABD                null       nullnull       null
```

```
Oracle syntax;
Select distinct */ column/ expression
From table1_name, table2_name……
Where <joining-condition> and/or <filtering-condition>;
Table1_col = Table2_col (+)
```

E.g.,

```
Select * from std s, branch b
Where s.bid = b.bid(+);
```

```
       SID SNAME              BID        BID BNAME      HOB
---------- ---------- ---------- ---------- ---------- ---------
         1 Jack               143        143 CSE        Sunny
         2 maggie             840        840 ME         Johnny
         3 Dani               null       nullnull       null
         4 Raju               420        420 ECE        Kattapa
         5 Rock               840        840 ME         Johnny
         6 ABD                null       nullnull       null
```

Right outer Join – It joins the tables and returns both matched and unmatched records, unmatched records from right table and only matched records from left table.

Syntax;

```
ANSI Syntax;
Select distinct */ column/ expression
From table1_name right outer join table2_name……
On <joining-condition> and/or <filtering-condition>
where <filtering-condition>;

Oracle syntax;
Select distinct */ column/ expression
From table1_name, table2_name……
Where < Table1_col (+) = Table2_col > and/or <filtering-condition>;
```

E.g.,

```
Select * from std s right join branch b
On s.bid=b.bid;
```

Or

```
Select * from std s, branch b
Where s.bid (+) = b.bid;
```

```
       SID SNAME              BID         BID BNAME      HOB
---------- ---------- ---------- ---------- ---------- ----------
         1 Jack               143        143 CSE        Sunny
         4 Raju               420        420 ECE        Kattapa
                                         600 CIVIL      Mia
         5 Rock               840        840 ME         Johnny
         2 maggie             840        840 ME         Johnny
```

Full outer join – It joins multiple tables and returns matched and unmatched records from both left and right tables as a result.

Syntax;

```
ANSI Syntax;
Select distinct */ column/ expression
From table1_name full join table2_name……
On <joining-condition> and/or <filtering-condition>
where <filtering-condition>;
```

E.g.,

```
Select * from std s full join branch b
On s.bid = b.bid;
```

```
       SID SNAME              BID        BID BNAME      HOB
---------- ---------- ---------- ---------- ---------- -------
         1 Jack               143        143 CSE        Sunny
         2 maggie             840        840 ME         Johnny
         3 Dani
         4 Raju               420        420 ECE        Kattapa
         5 Rock               840        840 ME         Johnny
         6 ABD
                                         600 CIVIL      Mia
```

Self Join – Joining a table by itself is known as Self join.

```
ANSI Syntax;
Select distinct */ column/ expression
From table1_name reference_name1 join table1_name reference_name2……
On <joining-condition> and/or <filtering-condition>
where <filtering-condition>;
```

```
Oracle syntax;
Select distinct */ column/ expression
From table1_name reference_name1,  table1_name reference_name2……
Where <joining-condition> and/or <filtering-condition>;
```

E.g.,

```
Select e1.empno, e1.ename, e1.job, e2.empno, e2.ename, e2.job
From emp e1, emp e2;
```

```
     EMPNO ENAME      JOB              EMPNO ENAME      JOB
---------- ---------- --------- ---------- ---------- ---------
      7369 SMITH      CLERK           7369 SMITH      CLERK
      7369 SMITH      CLERK           7499 ALLEN      SALESMAN
      7369 SMITH      CLERK           7521 WARD       SALESMAN
      7369 SMITH      CLERK           7566 JONES      MANAGER
      7369 SMITH      CLERK           7654 MARTIN     SALESMAN
      7369 SMITH      CLERK           7698 BLAKE      MANAGER
      7369 SMITH      CLERK           7782 CLARK      MANAGER
      7369 SMITH      CLERK           7788 SCOTT      ANALYST
      7369 SMITH      CLERK           7839 KING       PRESIDENT
      7369 SMITH      CLERK           7844 TURNER     SALESMAN
      7369 SMITH      CLERK           7876 ADAMS      CLERK
      7369 SMITH      CLERK           7900 JAMES      CLERK
      7369 SMITH      CLERK           7902 FORD       ANALYST
      7369 SMITH      CLERK           7934 MILLER     CLERK
      7499 ALLEN      SALESMAN        7369 SMITH      CLERK
      7499 ALLEN      SALESMAN        7499 ALLEN      SALESMAN
      7499 ALLEN      SALESMAN        7521 WARD       SALESMAN
      7499 ALLEN      SALESMAN        7566 JONES      MANAGER
```

7499	ALLEN	SALESMAN	7654	MARTIN	SALESMAN
7499	ALLEN	SALESMAN	7698	BLAKE	MANAGER
7499	ALLEN	SALESMAN	7782	CLARK	MANAGER
7499	ALLEN	SALESMAN	7788	SCOTT	ANALYST
7499	ALLEN	SALESMAN	7839	KING	PRESIDENT
7499	ALLEN	SALESMAN	7844	TURNER	SALESMAN
7499	ALLEN	SALESMAN	7876	ADAMS	CLERK
7499	ALLEN	SALESMAN	7900	JAMES	CLERK
7499	ALLEN	SALESMAN	7902	FORD	ANALYST
7499	ALLEN	SALESMAN	7934	MILLER	CLERK
7521	WARD	SALESMAN	7369	SMITH	CLERK
7521	WARD	SALESMAN	7499	ALLEN	SALESMAN
7521	WARD	SALESMAN	7521	WARD	SALESMAN
7521	WARD	SALESMAN	7566	JONES	MANAGER
7521	WARD	SALESMAN	7654	MARTIN	SALESMAN
7521	WARD	SALESMAN	7698	BLAKE	MANAGER
7521	WARD	SALESMAN	7782	CLARK	MANAGER
7521	WARD	SALESMAN	7788	SCOTT	ANALYST
7521	WARD	SALESMAN	7839	KING	PRESIDENT
7521	WARD	SALESMAN	7844	TURNER	SALESMAN
7521	WARD	SALESMAN	7876	ADAMS	CLERK
7521	WARD	SALESMAN	7900	JAMES	CLERK
7521	WARD	SALESMAN	7902	FORD	ANALYST
7521	WARD	SALESMAN	7934	MILLER	CLERK
7566	JONES	MANAGER	7369	SMITH	CLERK
7566	JONES	MANAGER	7499	ALLEN	SALESMAN
7566	JONES	MANAGER	7521	WARD	SALESMAN
7566	JONES	MANAGER	7566	JONES	MANAGER
7566	JONES	MANAGER	7654	MARTIN	SALESMAN
7566	JONES	MANAGER	7698	BLAKE	MANAGER
7566	JONES	MANAGER	7782	CLARK	MANAGER
7566	JONES	MANAGER	7788	SCOTT	ANALYST
7566	JONES	MANAGER	7839	KING	PRESIDENT
7566	JONES	MANAGER	7844	TURNER	SALESMAN
7566	JONES	MANAGER	7876	ADAMS	CLERK
7566	JONES	MANAGER	7900	JAMES	CLERK
7566	JONES	MANAGER	7902	FORD	ANALYST
7566	JONES	MANAGER	7934	MILLER	CLERK
7654	MARTIN	SALESMAN	7369	SMITH	CLERK
7654	MARTIN	SALESMAN	7499	ALLEN	SALESMAN
7654	MARTIN	SALESMAN	7521	WARD	SALESMAN
7654	MARTIN	SALESMAN	7566	JONES	MANAGER
7654	MARTIN	SALESMAN	7654	MARTIN	SALESMAN
7654	MARTIN	SALESMAN	7698	BLAKE	MANAGER
7654	MARTIN	SALESMAN	7782	CLARK	MANAGER
7654	MARTIN	SALESMAN	7788	SCOTT	ANALYST
7654	MARTIN	SALESMAN	7839	KING	PRESIDENT
7654	MARTIN	SALESMAN	7844	TURNER	SALESMAN
7654	MARTIN	SALESMAN	7876	ADAMS	CLERK
7654	MARTIN	SALESMAN	7900	JAMES	CLERK
7654	MARTIN	SALESMAN	7902	FORD	ANALYST
7654	MARTIN	SALESMAN	7934	MILLER	CLERK
7698	BLAKE	MANAGER	7369	SMITH	CLERK
7698	BLAKE	MANAGER	7499	ALLEN	SALESMAN
7698	BLAKE	MANAGER	7521	WARD	SALESMAN
7698	BLAKE	MANAGER	7566	JONES	MANAGER
7698	BLAKE	MANAGER	7654	MARTIN	SALESMAN
7698	BLAKE	MANAGER	7698	BLAKE	MANAGER
7698	BLAKE	MANAGER	7782	CLARK	MANAGER
7698	BLAKE	MANAGER	7788	SCOTT	ANALYST
7698	BLAKE	MANAGER	7839	KING	PRESIDENT
7698	BLAKE	MANAGER	7844	TURNER	SALESMAN
7698	BLAKE	MANAGER	7876	ADAMS	CLERK
7698	BLAKE	MANAGER	7900	JAMES	CLERK
7698	BLAKE	MANAGER	7902	FORD	ANALYST
7698	BLAKE	MANAGER	7934	MILLER	CLERK
7782	CLARK	MANAGER	7369	SMITH	CLERK

```
      7782 CLARK      MANAGER         7499 ALLEN      SALESMAN
      7782 CLARK      MANAGER         7521 WARD       SALESMAN
      7782 CLARK      MANAGER         7566 JONES      MANAGER
      7782 CLARK      MANAGER         7654 MARTIN     SALESMAN
      7782 CLARK      MANAGER         7698 BLAKE      MANAGER
      7782 CLARK      MANAGER         7782 CLARK      MANAGER
      7782 CLARK      MANAGER         7788 SCOTT      ANALYST
      7782 CLARK      MANAGER         7839 KING       PRESIDENT
      7782 CLARK      MANAGER         7844 TURNER     SALESMAN
      7782 CLARK      MANAGER         7876 ADAMS      CLERK
      7782 CLARK      MANAGER         7900 JAMES      CLERK
      7782 CLARK      MANAGER         7902 FORD       ANALYST
      7782 CLARK      MANAGER         7934 MILLER     CLERK
      7788 SCOTT      ANALYST         7369 SMITH      CLERK
      7788 SCOTT      ANALYST         7499 ALLEN      SALESMAN
      7788 SCOTT      ANALYST         7521 WARD       SALESMAN
      7788 SCOTT      ANALYST         7566 JONES      MANAGER
      7788 SCOTT      ANALYST         7654 MARTIN     SALESMAN
      7788 SCOTT      ANALYST         7698 BLAKE      MANAGER
      7788 SCOTT      ANALYST         7782 CLARK      MANAGER
      7788 SCOTT      ANALYST         7788 SCOTT      ANALYST
      7788 SCOTT      ANALYST         7839 KING       PRESIDENT
      7788 SCOTT      ANALYST         7844 TURNER     SALESMAN
      7788 SCOTT      ANALYST         7876 ADAMS      CLERK
      7788 SCOTT      ANALYST         7900 JAMES      CLERK
      7788 SCOTT      ANALYST         7902 FORD       ANALYST
      7788 SCOTT      ANALYST         7934 MILLER     CLERK
      7839 KING       PRESIDENT       7369 SMITH      CLERK
      7839 KING       PRESIDENT       7499 ALLEN      SALESMAN
      7839 KING       PRESIDENT       7521 WARD       SALESMAN
      7839 KING       PRESIDENT       7566 JONES      MANAGER
      7839 KING       PRESIDENT       7654 MARTIN     SALESMAN
      7839 KING       PRESIDENT       7698 BLAKE      MANAGER
      7839 KING       PRESIDENT       7782 CLARK      MANAGER
      7839 KING       PRESIDENT       7788 SCOTT      ANALYST
      7839 KING       PRESIDENT       7839 KING       PRESIDENT
      7839 KING       PRESIDENT       7844 TURNER     SALESMAN
      7839 KING       PRESIDENT       7876 ADAMS      CLERK
      7839 KING       PRESIDENT       7900 JAMES      CLERK
      7839 KING       PRESIDENT       7902 FORD       ANALYST
      7839 KING       PRESIDENT       7934 MILLER     CLERK
      7844 TURNER     SALESMAN        7369 SMITH      CLERK
      7844 TURNER     SALESMAN        7499 ALLEN      SALESMAN
      7844 TURNER     SALESMAN        7521 WARD       SALESMAN
      7844 TURNER     SALESMAN        7566 JONES      MANAGER
      7844 TURNER     SALESMAN        7654 MARTIN     SALESMAN
      7844 TURNER     SALESMAN        7698 BLAKE      MANAGER
      7844 TURNER     SALESMAN        7782 CLARK      MANAGER
      7844 TURNER     SALESMAN        7788 SCOTT      ANALYST
      7844 TURNER     SALESMAN        7839 KING       PRESIDENT
      7844 TURNER     SALESMAN        7844 TURNER     SALESMAN
      7844 TURNER     SALESMAN        7876 ADAMS      CLERK
      7844 TURNER     SALESMAN        7900 JAMES      CLERK
      7844 TURNER     SALESMAN        7902 FORD       ANALYST
      7844 TURNER     SALESMAN        7934 MILLER     CLERK
      7876 ADAMS      CLERK           7369 SMITH      CLERK
      7876 ADAMS      CLERK           7499 ALLEN      SALESMAN
      7876 ADAMS      CLERK           7521 WARD       SALESMAN
      7876 ADAMS      CLERK           7566 JONES      MANAGER
      7876 ADAMS      CLERK           7654 MARTIN     SALESMAN
      7876 ADAMS      CLERK           7698 BLAKE      MANAGER
      7876 ADAMS      CLERK           7782 CLARK      MANAGER

     EMPNO ENAME      JOB              EMPNO ENAME      JOB
---------- ---------- --------- ---------- ---------- ---------
      7876 ADAMS      CLERK           7788 SCOTT      ANALYST
      7876 ADAMS      CLERK           7839 KING       PRESIDENT
```

```
      7876 ADAMS      CLERK           7844 TURNER     SALESMAN
      7876 ADAMS      CLERK           7876 ADAMS      CLERK
      7876 ADAMS      CLERK           7900 JAMES      CLERK
      7876 ADAMS      CLERK           7902 FORD       ANALYST
      7876 ADAMS      CLERK           7934 MILLER     CLERK
      7900 JAMES      CLERK           7369 SMITH      CLERK
      7900 JAMES      CLERK           7499 ALLEN      SALESMAN
      7900 JAMES      CLERK           7521 WARD       SALESMAN
      7900 JAMES      CLERK           7566 JONES      MANAGER
      7900 JAMES      CLERK           7654 MARTIN     SALESMAN
      7900 JAMES      CLERK           7698 BLAKE      MANAGER
      7900 JAMES      CLERK           7782 CLARK      MANAGER
      7900 JAMES      CLERK           7788 SCOTT      ANALYST
      7900 JAMES      CLERK           7839 KING       PRESIDENT
      7900 JAMES      CLERK           7844 TURNER     SALESMAN
      7900 JAMES      CLERK           7876 ADAMS      CLERK
      7900 JAMES      CLERK           7900 JAMES      CLERK
      7900 JAMES      CLERK           7902 FORD       ANALYST
      7900 JAMES      CLERK           7934 MILLER     CLERK
      7902 FORD       ANALYST         7369 SMITH      CLERK
      7902 FORD       ANALYST         7499 ALLEN      SALESMAN
      7902 FORD       ANALYST         7521 WARD       SALESMAN
      7902 FORD       ANALYST         7566 JONES      MANAGER
      7902 FORD       ANALYST         7654 MARTIN     SALESMAN
      7902 FORD       ANALYST         7698 BLAKE      MANAGER
      7902 FORD       ANALYST         7782 CLARK      MANAGER
      7902 FORD       ANALYST         7788 SCOTT      ANALYST
      7902 FORD       ANALYST         7839 KING       PRESIDENT
      7902 FORD       ANALYST         7844 TURNER     SALESMAN
      7902 FORD       ANALYST         7876 ADAMS      CLERK
      7902 FORD       ANALYST         7900 JAMES      CLERK
      7902 FORD       ANALYST         7902 FORD       ANALYST
      7902 FORD       ANALYST         7934 MILLER     CLERK
      7934 MILLER     CLERK           7369 SMITH      CLERK
      7934 MILLER     CLERK           7499 ALLEN      SALESMAN
      7934 MILLER     CLERK           7521 WARD       SALESMAN
      7934 MILLER     CLERK           7566 JONES      MANAGER
      7934 MILLER     CLERK           7654 MARTIN     SALESMAN
      7934 MILLER     CLERK           7698 BLAKE      MANAGER
      7934 MILLER     CLERK           7782 CLARK      MANAGER
      7934 MILLER     CLERK           7788 SCOTT      ANALYST
      7934 MILLER     CLERK           7839 KING       PRESIDENT
      7934 MILLER     CLERK           7844 TURNER     SALESMAN
      7934 MILLER     CLERK           7876 ADAMS      CLERK
      7934 MILLER     CLERK           7900 JAMES      CLERK
      7934 MILLER     CLERK           7902 FORD       ANALYST
      7934 MILLER     CLERK           7934 MILLER     CLERK

196 rows selected.
```

72) Query to display employees who are working in same designation as miller

Ans.

```
Select e1.empno, e1.ename, e1.job, e2.empno, e2.ename, e2.job
From emp e1 join emp e2
On e1.ename = 'MILLER' and e1.job = e2.job;
```

```
     EMPNO ENAME      JOB             EMPNO ENAME      JOB
---------- ---------- --------- ---------- ---------- ---------
      7934 MILLER     CLERK           7369 SMITH      CLERK
      7934 MILLER     CLERK           7876 ADAMS      CLERK
      7934 MILLER     CLERK           7900 JAMES      CLERK
      7934 MILLER     CLERK           7934 MILLER     CLERK
```

04/04/2023

73) Write a query to display names of employees with their reporting Manager's name

Ans.

```
Select e1.ename Employees, e2.ename Reporting_ from emp e1, emp e2 Where
e1.mgr = e2.empno;
```

```
EMPLOYEES  REPORTING_
---------- ----------
SMITH      FORD
ALLEN      BLAKE
WARD       BLAKE
JONES      KING
MARTIN     BLAKE
BLAKE      KING
CLARK      KING
SCOTT      JONES
TURNER     BLAKE
ADAMS      SCOTT
JAMES      BLAKE
FORD       JONES
MILLER     CLARK
```

```
Select e1.ename Employees, e2.ename Reporting_ from emp e1, emp e2 Where
e1.mgr = e2.empno(+);
```

```
EMPLOYEES  REPORTING_
---------- ----------
SMITH      FORD
ALLEN      BLAKE
WARD       BLAKE
JONES      KING
MARTIN     BLAKE
BLAKE      KING
CLARK      KING
SCOTT      JONES
KING
TURNER     BLAKE
ADAMS      SCOTT
JAMES      BLAKE
FORD       JONES
MILLER     CLARK
```

74) Write a query to display the number of employees reporting to Blake and King

Ans.

```
Select e2.ename, count(*)
From emp e1, emp e2
Where e1.mgr = e2.empno
Group by e2.ename
Having e2.ename in ('BLAKE', 'KING');
```

```
ENAME      COUNT(*)
---------- ----------
```

```
BLAKE               5
KING                3
```

75) Write a query to display employees, their location, and their reporting managers, It uses both Self Join and Inner Join

Ans.

```
Select e1.ename, loc, e2.ename
From emp e1, emp e2, dept d
where e1.mgr = e2.empno and e1.deptno = d.deptno;
```

```
ENAME      LOC            ENAME
---------- -------------- ---------
SMITH      DALLAS         FORD
ALLEN      CHICAGO        BLAKE
WARD       CHICAGO        BLAKE
JONES      DALLAS         KING
MARTIN     CHICAGO        BLAKE
BLAKE      CHICAGO        KING
CLARK      NEW YORK       KING
SCOTT      DALLAS         JONES
TURNER     CHICAGO        BLAKE
ADAMS      DALLAS         SCOTT
JAMES      CHICAGO        BLAKE
FORD       DALLAS         JONES
MILLER     NEW YORK       CLARK
```

```
Select e1.ename, d1.loc, e2.ename, d2.loc
From emp e1, dept d1, emp e2, dept d2
Where e1.mgr = e2.empno and e1.deptno = d1.deptno and e2.deptno = d2.deptno;
```

```
ENAME      LOC            ENAME      LOC
---------- -------------- ---------- -------------
SMITH      DALLAS         FORD       DALLAS
ALLEN      CHICAGO        BLAKE      CHICAGO
WARD       CHICAGO        BLAKE      CHICAGO
JONES      DALLAS         KING       NEW YORK
MARTIN     CHICAGO        BLAKE      CHICAGO
BLAKE      CHICAGO        KING       NEW YORK
CLARK      NEW YORK       KING       NEW YORK
SCOTT      DALLAS         JONES      DALLAS
TURNER     CHICAGO        BLAKE      CHICAGO
ADAMS      DALLAS         SCOTT      DALLAS
JAMES      CHICAGO        BLAKE      CHICAGO
FORD       DALLAS         JONES      DALLAS
MILLER     NEW YORK       CLARK      NEW YORK
```

76) Write a query to display names of employees with their reporting managers where both employees and reporting manager working in same location

Ans.

```
Select e1.ename, d1.loc, e2.ename, d2.loc
From emp e1, dept d1, emp e2, dept d2
Where e1.mgr = e2.empno and e1.deptno = d1.deptno and e2.deptno = d2.deptno
and d1.loc = d2.loc;
```

```
ENAME      LOC            ENAME      LOC
---------- ------------- ---------- -------------
SMITH      DALLAS         FORD       DALLAS
ALLEN      CHICAGO        BLAKE      CHICAGO
WARD       CHICAGO        BLAKE      CHICAGO
MARTIN     CHICAGO        BLAKE      CHICAGO
CLARK      NEW YORK       KING       NEW YORK
SCOTT      DALLAS         JONES      DALLAS
TURNER     CHICAGO        BLAKE      CHICAGO
ADAMS      DALLAS         SCOTT      DALLAS
JAMES      CHICAGO        BLAKE      CHICAGO
FORD       DALLAS         JONES      DALLAS
MILLER     NEW YORK       CLARK      NEW YORK
```

77) Write a query to display names of employees with their reporting managers and their respective hiredate and display only employees who joined before their reporting manager

Ans.

```
Select e1.ename, e1.hiredate, e2.hiredate, e2.hiredate
From emp e1, emp e2
Where e1.mgr = e2.empno and e1.hiredate < e2.hiredate;
```

```
ENAME      HIREDATE  HIREDATE  HIREDATE
---------- --------- --------- ---------
SMITH      17-DEC-80 03-DEC-81 03-DEC-81
ALLEN      20-FEB-81 01-MAY-81 01-MAY-81
WARD       22-FEB-81 01-MAY-81 01-MAY-81
JONES      02-APR-81 17-NOV-81 17-NOV-81
BLAKE      01-MAY-81 17-NOV-81 17-NOV-81
CLARK      09-JUN-81 17-NOV-81 17-NOV-81
```

78) Write a query to display details of employees who are not working as reporting managers using Joins

Ans.

```
Select e1.ename, e2.ename from emp e1, emp e2
Where e1.mgr=e2.empno(+) and e2.ename is NULL;
```

```
ENAME      ENAME
---------- ----------
           SMITH
           ALLEN
           WARD
           MARTIN
           TURNER
           ADAMS
           JAMES
           MILLER
```

Co-related Sub Query

Co-related subquery is used to compare each and every value of a table with all the values of another table. In co-related subqueries, outer query first partially and the results of outer query is compared with all the values of inner query and if the conditions are satisfied the outer query returns the result.

Syntax;

79) Write a query to display the 4th maximum salary from emp table

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)));
```

Or

To get the 4th maximum salary using co-related sub query.

```
Select distinct sal from emp e1
Where 4=(select count(distinct sal) from emp e2 where e1.sal<=e2.sal);
```

```
       SAL
----------
      2850
```

Or we can use;

```
select * from (select t1.*,  rownum rn from (select distinct sal from emp
order by sal desc) t1)
where rn = 4;
```

80) Write a query to display the names of the employees who are having second maximum number of characters

Ans.

```
Select ename from emp e1
Where 2=(select count(distinct length(ename)) from emp e2 where
length(e1.ename)<=length(e2.ename));
```

```
ENAME
-----
SMITH
ALLEN
JONES
BLAKE
CLARK
SCOTT
ADAMS
JAMES
```