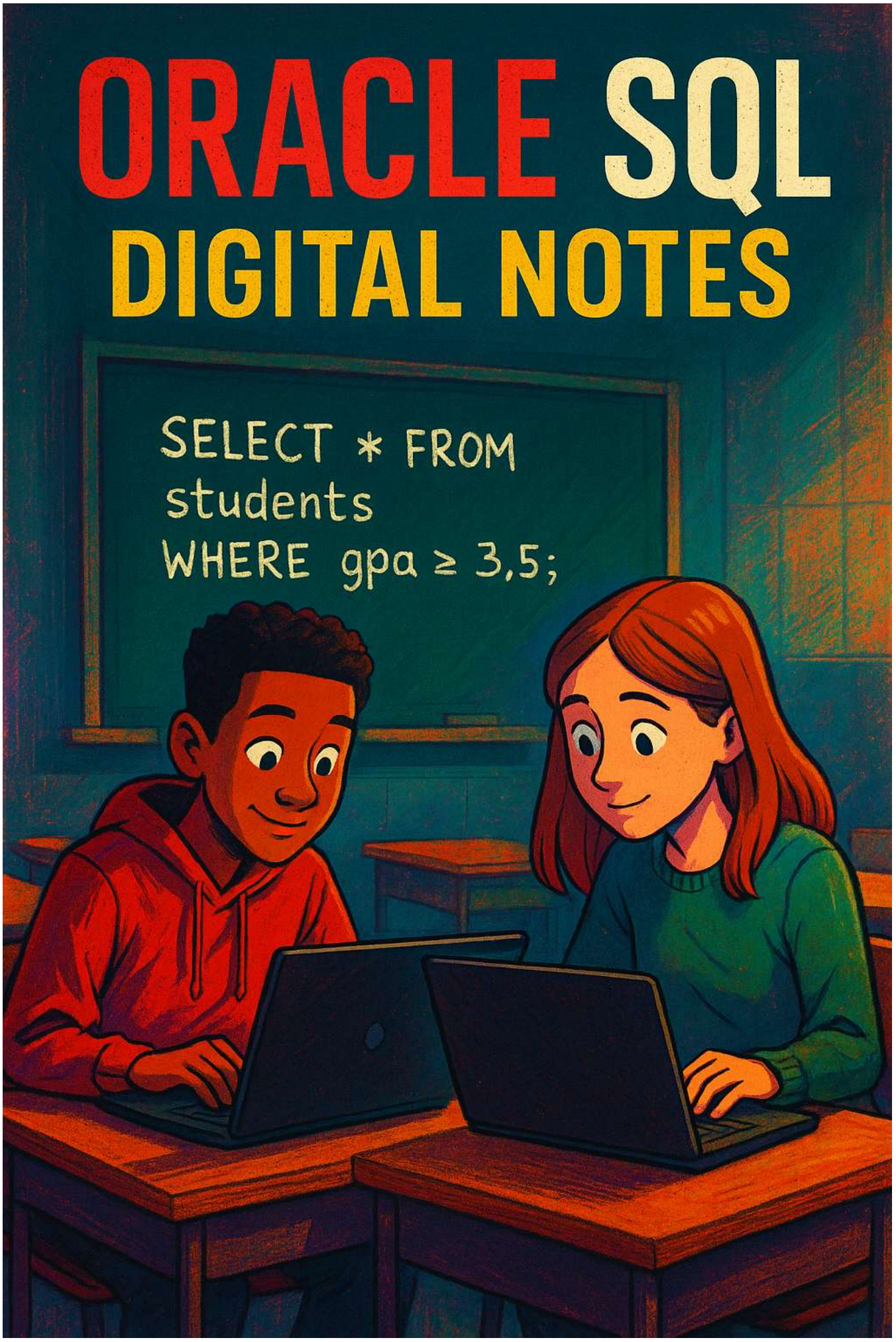
ORACLE SQL
DIGITAL NOTES



INDEX

8	Transaction Control Language (TCL)	- COMMIT - SAVEPOINT - ROLLBACK
9	Data Control Language (DCL)	- GRANT - REVOKE
10	Operators	- Types of Operators - Examples
11	Functions	- Single Row Functions - Multi Row Functions: MAX(), MIN(), SUM(), AVG(), COUNT()
12	Sub Queries	- Introduction - Working - Query Writing - Types: 1. Single Row 2. Multi Row - Nested Subquery
13	Co-Related Sub Queries	- Concept - Working Examples
14	Pseudo Columns	- Introduction - ROWID - ROWNUM - Usage
15	Joins	- What is Join? - Types: • Cartesian Join • Inner Join • Outer Join • Self Join - Examples
16	Normalization	- Introduction - Types of Normal Forms: 1NF, 2NF, 3NF, BCNF - Examples

SOFTWARE

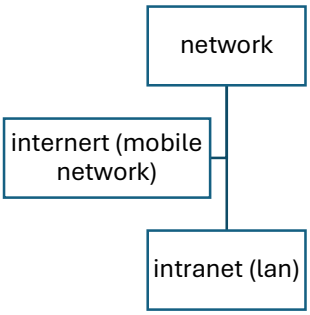
A set of programs that are used to perform some basic operations.

Software is also called an application.

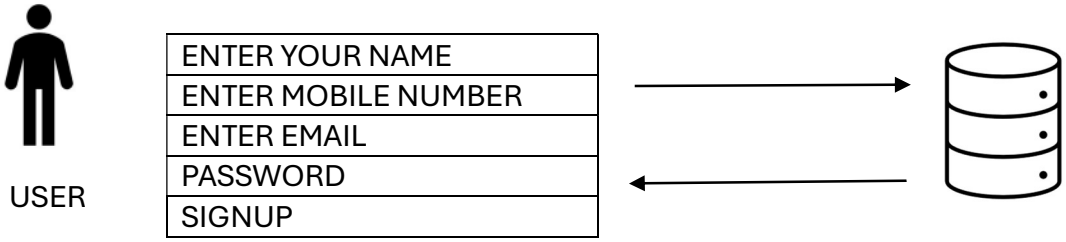
To access each application network connection and database are mandatory .

NETWORK

The connection between two devices is known as a network.



DATABASE



Database is a container and it is used to store useful information in the form of data .

DATA

Data is a raw fact which is used to describe the property of an object

Properties are also known as attributes (columns).

INFORMATION

Collection Of Data Is Known as Information.

FEATURES OF DATABASE

- Databases are mandatory for data security purposes and for data accessing purposes.
- On a database, we can perform basic operations called CRUD operations.
  i) C stands for creating a database in the database server and storing data in the database,
  ii) R stands for retrieving data from the database.
  iii) U stands for updating old data with new data.
  iv) D stands for deleting unwanted data from database.

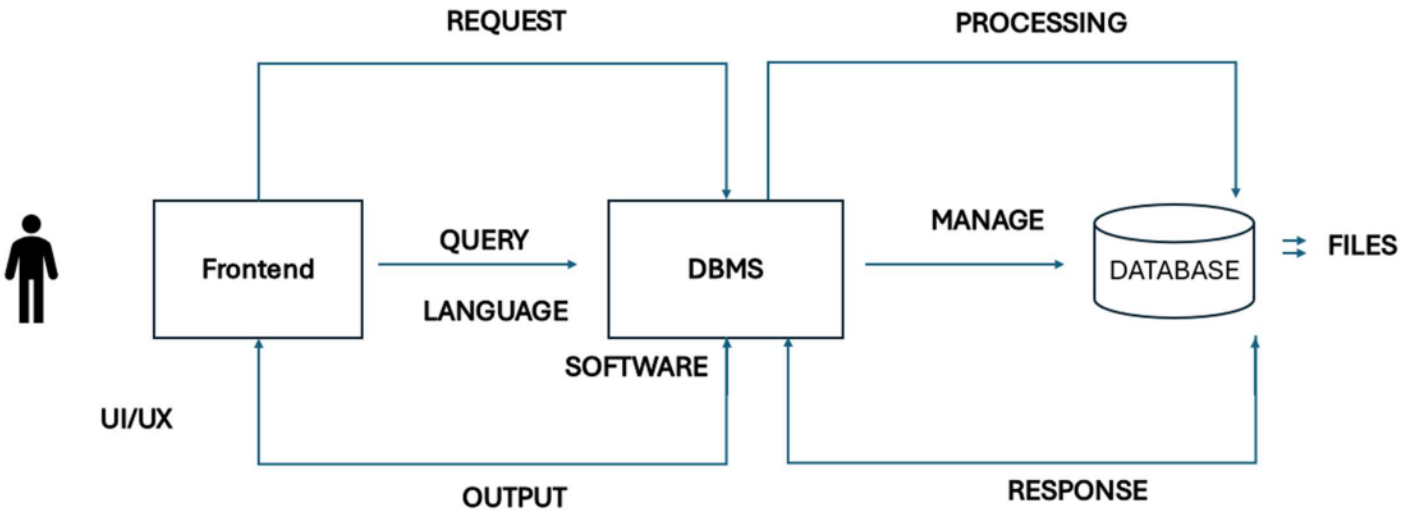
Note :

- ❖ To perform CRUD operations, we need database software.

TYPES OF DATABASE SOFTWARE

- DBMS
- RDBMS
- NOSQL
- HDBMS
- NDBMS
- ODBMS

DBMS



DBMS stands for database management system, a software used to manage databases.

Manage stands for storing data in the database, retrieving data from the database, updating old data with new data and deleting unwanted data, we can perform CRUD operations on the database using DBMS software. To interact with the DBMS, we are using Query language.

In real-life scenarios, we do not use DBMS software because data security is less and retrieving data from the database is very complex.

Using DBMS software, we can't build relations across multiple files.

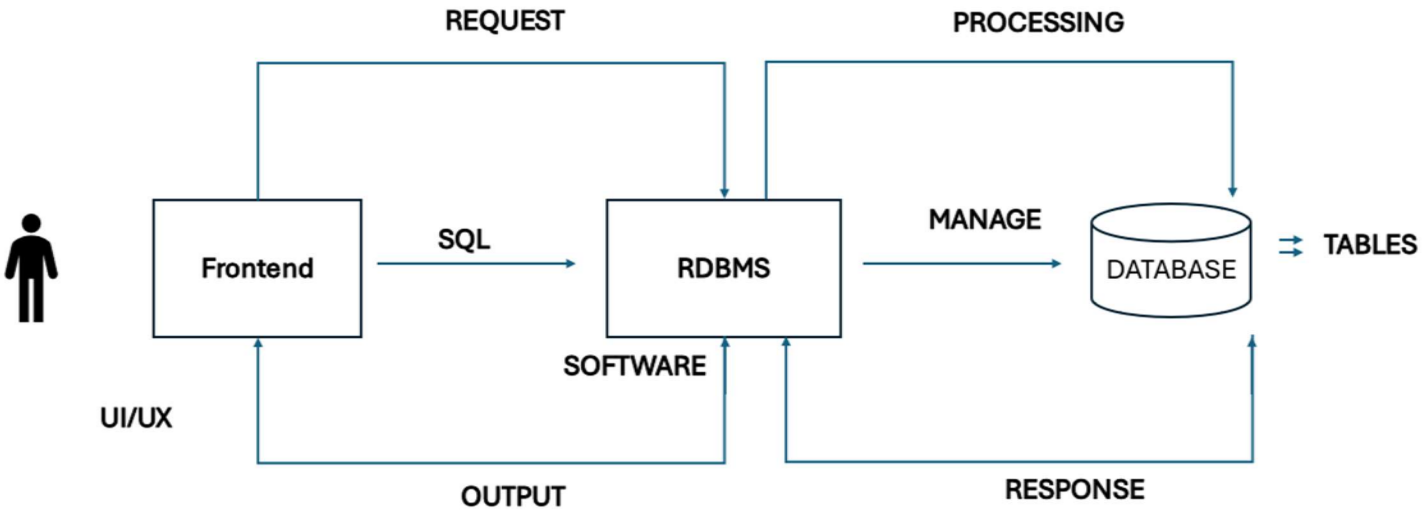
Using DBMS, we can store data in the database in the form of files.

EXAMPLES of DBMS

File Manager

Microsoft Access (.txt, .csv files)

RDBMS



RDBMS stands for relational database management system.

RDBMS is a software used to manage and maintain databases.

We can perform CRUD operations on database using RDBMS software.

To interact with RDBMS we use SQL (Structured Query Language).

Edgar F Codd introduced RDBMS in the year 1970.

Rules towards RDBMS by Edgar F Codd

1. In each and every cell, we have to store single-value data.
2. Data should be stored in the form of tables (rows and columns).
3. Using RDBMS software, we can create multiple tables, and we can build Relations on multiple tables.

ADVANTAGES OF RDBMS

- Data security is more in RDBMS software .
- Storing data and retrieving data from database is easier as compared to other softwares.
- We can build relation on multiple tables.
- Data redundancy is less.
- We can apply normalization on tables.
- It follows data integrity and ACID properties.

EXAMPLES OF RDMS

- Oracle sql
- Mysql
- Microsoft sql
- Postgre sql

Note:

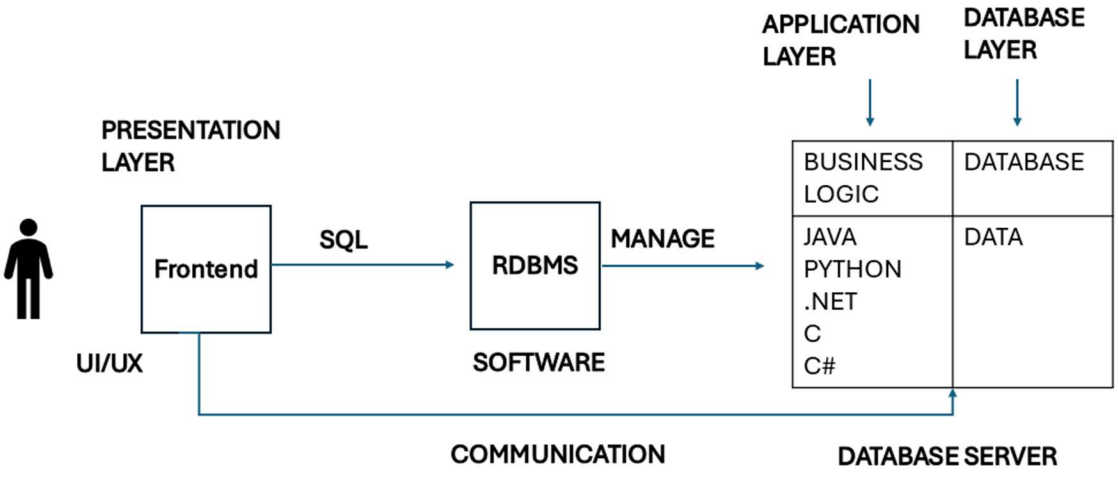
- Tables are also called as objects or entity.
- Rows are also called as records or tuple.
- Columns are also called as files and attributes.

DIFFERENCE BETWEEN DBMS AND RDBMS

DIFFERENCE BETWEEN DBMS AND RDBMS

DBMS	RDBMS
Data is stored in the form of files	Data is stored in the form of tables
Data security is less	Data security is more as compared to DBMS
Retrieving data from the database is complex	Retrieving data from database is easy
Data redundancy is more	Data redundancy is less
We can't apply normalization on files	We can apply normalization on files
We can store small amount of data using DBMS	We can store large amount of data using RDBMS
We can't build relations on multiple tables	We can build relations on multiple tables
To interact with DBMS we use Query Language	To interact with DBMS we use SQL

INTRODUCTION TO SQL



- SQL stands for Structured query language.
- SQL is a language used to communicate with database server with the help of RDBMS software.
- IBM researchers Raymond Boyce and Donald D. Chamberlin introduced SQL in the year 1970.
- In the year 1986, ANSI (American National Standards Institute) adopted SQL .
- SQL is a high level language.
- SQL is not a case sensitive language but the data present inside the table is case sensitive.
- Each and every query must end with a semicolon.
- SQL is also known as sequal query language.
- To manage and maintain database server we can make use of SQL statements.

SQL STATEMENTS

Types of SQL statements :

1. DQL / DRL (Data Query Language / Data Retrieval Language)
2. DDL (Data Definition Language)
3. DML (Data Manipulation Language)
4. TCL (Transaction Control Language)
5. DCL (Data Control Language)

DQL	DDL	DML	TCL	DCL
SELECT	CREATE TABLE	INSERT DATA	COMMIT	COMMIT
PROJECTION	RENAME TABLE	UPDATE DATA	ROLLBACK	REVOKE
SELECTION	DROP TABLE	DELETE DATA	SAVEPOINT	
FROM	FLASHBACK TABLE			
WHERE CLAUSE	PURGE TABLE			
GROUP BY CLAUSE	TRUNCATE TABLE			
HAVING CLAUSE	ALTER TABLE			
ORDER BY CLAUSE				
DISTINCT				
LIMIT				
OFFSET				
JOINS				

Note:

To display output in organized format in Oracle sql we have to use below command

SET PAGES 1000 LINES 1000

OR

SET PAGES 1000

SET LINES 1000

ORACLE SQL EMPLOYEE TABLE

```
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.
```

ORACLE SQL DEPARTMENT TABLE

```
SQL> SELECT * FROM DEPT;

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

DQL (Data Query Language)

Data query language statements are used to retrieve data from database.

Types of DQL:

SELECT

This statement is used to select and retrieve data from the table.

PROJECTION (*)

Projection is used to retrieve data from the database by selecting all the columns.

SELECTION

Selection is used to select the particular column to retrieve data from the database.

FROM

It is a keyword used to specify the table name.

QUERIES ON EMP TABLE

a) Write a query to display all employee information.

```
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.
```

b) Query to display department table information.

```
SQL> SELECT * FROM DEPT;

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

c) Query to display name of the employee along with their designation.

```
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.
```

d) Query to display name of the employee along with their manager name.

```
SQL> SELECT ENAME , MGR FROM EMP;

ENAME             MGR
---------- ----------
SMITH            7902
ALLEN            7698
WARD             7698
JONES            7839
MARTIN           7698
BLAKE            7839
CLARK            7839
SCOTT            7566
KING
TURNER           7698
ADAMS            7788
JAMES            7698
FORD             7566
MILLER           7782

14 rows selected.
```

e) Query to display name of employee along with their salary and joining date.

```
SQL> SELECT ENAME , SAL , HIREDATE FROM EMP;

ENAME             SAL HIREDATE
---------- ---------- ---------
SMITH              800 17-DEC-80
ALLEN             1600 20-FEB-81
WARD              1250 22-FEB-81
JONES             2975 02-APR-81
MARTIN            1250 28-SEP-81
BLAKE             2850 01-MAY-81
CLARK             2450 09-JUN-81
SCOTT             3000 19-APR-87
KING              5000 17-NOV-81
TURNER            1500 08-SEP-81
ADAMS             1100 23-MAY-87
JAMES              950 03-DEC-81
FORD              3000 03-DEC-81
MILLER            1300 23-JAN-82

14 rows selected.
```

f) Query to display name of employee along with their department number and their commission.

```
SQL> SELECT ENAME , DEPTNO , COMM FROM EMP;

ENAME          DEPTNO       COMM
---------- ---------- ----------
SMITH              20
ALLEN              30        300
WARD               30        500
JONES              20
MARTIN             30       1400
BLAKE              30
CLARK              10
SCOTT              20
KING               10
TURNER             30          0
ADAMS              20
JAMES              30
FORD               20
MILLER             10

14 rows selected.
```

g) Query to display name of employee , designation and employee number.

```
SQL> SELECT ENAME , JOB , EMPNO FROM EMP;

ENAME      JOB            EMPNO
---------- --------- ----------
SMITH      CLERK           7369
ALLEN      SALESMAN        7499
WARD       SALESMAN        7521
JONES      MANAGER         7566
MARTIN     SALESMAN        7654
BLAKE      MANAGER         7698
CLARK      MANAGER         7782
SCOTT      ANALYST         7788
KING       PRESIDENT       7839
TURNER     SALESMAN        7844
ADAMS      CLERK           7876
JAMES      CLERK           7900
FORD       ANALYST         7902
MILLER     CLERK           7934

14 rows selected.
```

h) Query to display department name and location from department table.

```
SQL> SELECT DNAME , LOC FROM DEPT;

DNAME          LOC
-------------- -------------
ACCOUNTING     NEW YORK
RESEARCH       DALLAS
SALES          CHICAGO
OPERATIONS     BOSTON
```

WHERE CLAUSE

Where clause is used to give condition to the query and it will execute the output as per the condition.

For one select statement we can use only one select statement.

Syntax

SELECT * FROM TABLENAME

WHERE CONDITION ;

SELECT COLNAME1 , COLNAME2 FROM TABLENAME

WHERE CONDITION ;

a) Query to display smith's information from employee table.

```
SQL> SELECT * FROM EMP WHERE ENAME = 'SMITH' ;

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

b) Query to display name of employee along with their designation for employee who belongs from analyst designation.

```
SQL> SELECT ENAME , JOB FROM EMP WHERE JOB = 'ANALYST' ;

ENAME      JOB
---------- ---------
SCOTT      ANALYST
FORD       ANALYST
```

c) Query to display details of employees who are earning more than 3000.

```
SQL> SELECT * FROM EMP WHERE SAL > 3000 ;

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

d) Query to display details of employees who are earning less than 1000.

```
SQL> SELECT * FROM EMP WHERE SAL < 1000 ;

     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
```

e) Query to display details of employees who belongs from salesman designation.

```
SQL> SELECT * FROM EMP WHERE JOB = 'SALESMAN' ;

     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
```

f) Query to display details of employee who are having manager number 7839.

```
SQL> SELECT * FROM EMP WHERE MGR = 7839 ;

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

g) Query to display details of employee who joined after 1981.

```
SQL> SELECT * FROM EMP WHERE HIREDATE > '31-DEC-1981' ;

     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
```

h) Query to display details of employee who are earning commission less than 1000.

```
SQL> SELECT * FROM EMP WHERE COMM < 1000 ;

     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
      7844 TURNER     SALESMAN         7698 08-SEP-81       1500          0         30
```

ORDER BY

It is used to sort the records either in ascending or descending order .

Syntax

SELECT * FROM TABLENAME

ORDER BY COLNAME ASC/DESC ;

Order by is always written at the end of the query.

Order by execution steps:

1. FROM
2. WHERE
3. SELECT
4. ORDER BY

We can pass multiple columns to order by clause.

a) Query to display name and salary of employee where salary is arranged in descending order .

```
SQL> SELECT ENAME , SAL FROM EMP
  2  ORDER BY SAL DESC ;

ENAME             SAL
---------- ----------
KING             5000
FORD             3000
SCOTT            3000
JONES            2975
BLAKE            2850
CLARK            2450
ALLEN            1600
TURNER           1500
MILLER           1300
WARD             1250
MARTIN           1250
ADAMS            1100
JAMES             950
SMITH             800

14 rows selected.
```

b) Query to display name , salary and department number where salary and department number is arranged in ascending order.

```
SQL> SELECT ENAME , SAL , DEPTNO FROM EMP
  2  ORDER BY SAL , DEPTNO ;

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

14 rows selected.
```

c) Query to display details of manager whose salary is arranged in descending order.

```
SQL> SELECT * FROM EMP
  2  WHERE JOB = 'MANAGER'
  3  ORDER BY SAL DESC ;

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

d) Query to display details of employee arranged in descending order of their name if they are in department number 20.

```
SQL> SELECT * FROM EMP
  2  WHERE DEPTNO = 20
  3  ORDER BY ENAME DESC ;

     EMPNO ENAME      JOB              MGR HIREDATE         SAL       COMM     DEPTNO
---------- ---------- --------- ---------- --------- ---------- ---------- ----------
      7369 SMITH      CLERK           7902 17-DEC-80        800                    20
      7788 SCOTT      ANALYST         7566 19-APR-87       3000                    20
      7566 JONES      MANAGER         7839 02-APR-81       2975                    20
      7902 FORD       ANALYST         7566 03-DEC-81       3000                    20
      7876 ADAMS      CLERK           7788 23-MAY-87       1100                    20
```

e) Query to display details of employee whose hiredate is after 1981 arranged in descending order of their salary and department number.

```
SQL> SELECT * FROM EMP
  2  WHERE HIREDATE > '31-DEC-1981'
  3  ORDER BY SAL DESC , DEPTNO DESC ;

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

f) Query to display name , job and hiredate of employee arranged in the order of their join date.

```
SQL> SELECT ENAME , JOB , HIREDATE FROM EMP
  2  ORDER BY HIREDATE ;

ENAME      JOB       HIREDATE
---------- --------- ---------
SMITH      CLERK     17-DEC-80
ALLEN      SALESMAN  20-FEB-81
WARD       SALESMAN  22-FEB-81
JONES      MANAGER   02-APR-81
BLAKE      MANAGER   01-MAY-81
CLARK      MANAGER   09-JUN-81
TURNER     SALESMAN  08-SEP-81
MARTIN     SALESMAN  28-SEP-81
KING       PRESIDENT 17-NOV-81
JAMES      CLERK     03-DEC-81
FORD       ANALYST   03-DEC-81
MILLER     CLERK     23-JAN-82
SCOTT      ANALYST   19-APR-87
ADAMS      CLERK     23-MAY-87

14 rows selected.
```

EXPRESSION

Anything that gives a result is called an expression

Where 5 + 2 ⟶ 7

5 and 2 are operands, + is an operator.

a) Query to display annual salary of all employees.

```
SQL> SELECT SAL*12 FROM EMP ;

    SAL*12
----------
      9600
     19200
     15000
     35700
     15000
     34200
     29400
     36000
     60000
     18000
     13200
     11400
     36000
     15600

14 rows selected.
```

b) Query to display employee details along with annual salary.

```
SQL> SELECT EMP.* , SAL*12 FROM EMP ;
```

EMPNO	ENAME	JOB	MGR	HIREDATE	SAL	COMM	DEPTNO	SAL*12
7369	SMITH	CLERK	7902	17-DEC-80	800		20	9600
7499	ALLEN	SALESMAN	7698	20-FEB-81	1600	300	30	19200
7521	WARD	SALESMAN	7698	22-FEB-81	1250	500	30	15000
7566	JONES	MANAGER	7839	02-APR-81	2975		20	35700
7654	MARTIN	SALESMAN	7698	28-SEP-81	1250	1400	30	15000
7698	BLAKE	MANAGER	7839	01-MAY-81	2850		30	34200
7782	CLARK	MANAGER	7839	09-JUN-81	2450		10	29400
7788	SCOTT	ANALYST	7566	19-APR-87	3000		20	36000
7839	KING	PRESIDENT		17-NOV-81	5000		10	60000
7844	TURNER	SALESMAN	7698	08-SEP-81	1500	0	30	18000
7876	ADAMS	CLERK	7788	23-MAY-87	1100		20	13200
7900	JAMES	CLERK	7698	03-DEC-81	950		30	11400
7902	FORD	ANALYST	7566	03-DEC-81	3000		20	36000
7934	MILLER	CLERK	7782	23-JAN-82	1300		10	15600

```
14 rows selected.
```

c) Query to display details of employee along with 200 deduction in their salary.

```
SQL> SELECT EMP.* , SAL-200 FROM EMP ;
```

EMPNO	ENAME	JOB	MGR	HIREDATE	SAL	COMM	DEPTNO	SAL-200
7369	SMITH	CLERK	7902	17-DEC-80	800		20	600
7499	ALLEN	SALESMAN	7698	20-FEB-81	1600	300	30	1400
7521	WARD	SALESMAN	7698	22-FEB-81	1250	500	30	1050
7566	JONES	MANAGER	7839	02-APR-81	2975		20	2775
7654	MARTIN	SALESMAN	7698	28-SEP-81	1250	1400	30	1050
7698	BLAKE	MANAGER	7839	01-MAY-81	2850		30	2650
7782	CLARK	MANAGER	7839	09-JUN-81	2450		10	2250
7788	SCOTT	ANALYST	7566	19-APR-87	3000		20	2800
7839	KING	PRESIDENT		17-NOV-81	5000		10	4800
7844	TURNER	SALESMAN	7698	08-SEP-81	1500	0	30	1300
7876	ADAMS	CLERK	7788	23-MAY-87	1100		20	900
7900	JAMES	CLERK	7698	03-DEC-81	950		30	750
7902	FORD	ANALYST	7566	03-DEC-81	3000		20	2800
7934	MILLER	CLERK	7782	23-JAN-82	1300		10	1100

```
14 rows selected.
```

d) query to display details of employee along with 1000 bonus in their annual salary.

```
SQL> SELECT EMP.* , (SAL*12)+1000 FROM EMP ;
```

EMPNO	ENAME	JOB	MGR	HIREDATE	SAL	COMM	DEPTNO	(SAL*12)+1000
7369	SMITH	CLERK	7902	17-DEC-80	800		20	10600
7499	ALLEN	SALESMAN	7698	20-FEB-81	1600	300	30	20200
7521	WARD	SALESMAN	7698	22-FEB-81	1250	500	30	16000
7566	JONES	MANAGER	7839	02-APR-81	2975		20	36700
7654	MARTIN	SALESMAN	7698	28-SEP-81	1250	1400	30	16000
7698	BLAKE	MANAGER	7839	01-MAY-81	2850		30	35200
7782	CLARK	MANAGER	7839	09-JUN-81	2450		10	30400
7788	SCOTT	ANALYST	7566	19-APR-87	3000		20	37000
7839	KING	PRESIDENT		17-NOV-81	5000		10	61000
7844	TURNER	SALESMAN	7698	08-SEP-81	1500	0	30	19000
7876	ADAMS	CLERK	7788	23-MAY-87	1100		20	14200
7900	JAMES	CLERK	7698	03-DEC-81	950		30	12400
7902	FORD	ANALYST	7566	03-DEC-81	3000		20	37000
7934	MILLER	CLERK	7782	23-JAN-82	1300		10	16600

```
14 rows selected.
```

ALIAS

It is an alternate name given to a column or a table .

Alias can be written in two ways :

- ❖ With 'AS' keyword
- ❖ Without 'AS' keyword

If there is a space in the alias name, then the name should be enclosed with double quotes " " .

a) Query to display annual salary of employee.

```
SQL> SELECT SAL*12 AS ANN_SAL FROM EMP ;

   ANN_SAL
----------
      9600
     19200
     15000
     35700
     15000
     34200
     29400
     36000
     60000
     18000
     13200
     11400
     36000
     15600

14 rows selected.

SQL> SELECT SAL*12 ANN_SAL FROM EMP ;

SQL> SELECT SAL*12 AS "ANNUAL SALARY" FROM EMP;
```

b) Query to display details of employee along with 25% deduction in annual salary.

```
SQL> SELECT EMP.* , SAL*12 AS ANN_SAL , (SAL*12)*0.75 AS NEW_SAL FROM EMP ;

     EMPNO ENAME      JOB              MGR HIREDATE         SAL       COMM     DEPTNO    ANN_SAL    NEW_SAL
---------- ---------- --------- ---------- --------- ---------- ---------- ---------- ---------- ----------
      7369 SMITH      CLERK           7902 17-DEC-80        800                    20       9600       7200
      7499 ALLEN      SALESMAN        7698 20-FEB-81       1600        300         30      19200      14400
      7521 WARD       SALESMAN        7698 22-FEB-81       1250        500         30      15000      11250
      7566 JONES      MANAGER         7839 02-APR-81       2975                    20      35700      26775
      7654 MARTIN     SALESMAN        7698 28-SEP-81       1250       1400         30      15000      11250
      7698 BLAKE      MANAGER         7839 01-MAY-81       2850                    30      34200      25650
      7782 CLARK      MANAGER         7839 09-JUN-81       2450                    10      29400      22050
      7788 SCOTT      ANALYST         7566 19-APR-87       3000                    20      36000      27000
      7839 KING       PRESIDENT            17-NOV-81       5000                    10      60000      45000
      7844 TURNER     SALESMAN        7698 08-SEP-81       1500          0         30      18000      13500
      7876 ADAMS      CLERK           7788 23-MAY-87       1100                    20      13200       9900
      7900 JAMES      CLERK           7698 03-DEC-81        950                    30      11400       8550
      7902 FORD       ANALYST         7566 03-DEC-81       3000                    20      36000      27000
      7934 MILLER     CLERK           7782 23-JAN-82       1300                    10      15600      11700

14 rows selected.
```

c) Query to display details of employee along with 10% hike in salary.

```
SQL> SELECT EMP.* , SAL*12 AS ANN_SAL , (SAL*12)*110 AS NEW_SAL FROM EMP ;

     EMPNO ENAME      JOB              MGR HIREDATE         SAL       COMM     DEPTNO    ANN_SAL    NEW_SAL
---------- ---------- --------- ---------- --------- ---------- ---------- ---------- ---------- ----------
      7369 SMITH      CLERK           7902 17-DEC-80        800                    20       9600    1056000
      7499 ALLEN      SALESMAN        7698 20-FEB-81       1600        300         30      19200    2112000
      7521 WARD       SALESMAN        7698 22-FEB-81       1250        500         30      15000    1650000
      7566 JONES      MANAGER         7839 02-APR-81       2975                    20      35700    3927000
      7654 MARTIN     SALESMAN        7698 28-SEP-81       1250       1400         30      15000    1650000
      7698 BLAKE      MANAGER         7839 01-MAY-81       2850                    30      34200    3762000
      7782 CLARK      MANAGER         7839 09-JUN-81       2450                    10      29400    3234000
      7788 SCOTT      ANALYST         7566 19-APR-87       3000                    20      36000    3960000
      7839 KING       PRESIDENT            17-NOV-81       5000                    10      60000    6600000
      7844 TURNER     SALESMAN        7698 08-SEP-81       1500          0         30      18000    1980000
      7876 ADAMS      CLERK           7788 23-MAY-87       1100                    20      13200    1452000
      7900 JAMES      CLERK           7698 03-DEC-81        950                    30      11400    1254000
      7902 FORD       ANALYST         7566 03-DEC-81       3000                    20      36000    3960000
      7934 MILLER     CLERK           7782 23-JAN-82       1300                    10      15600    1716000

14 rows selected.
```

d) Query to display name , salary and annual salary of employee if employee's annual salary is higher than 20000.

```
SQL> SELECT ENAME , SAL , SAL*12 AS ANN_SAL FROM EMP WHERE SAL*12 > 20000 ;

ENAME             SAL    ANN_SAL
---------- ---------- ----------
JONES            2975      35700
BLAKE            2850      34200
CLARK            2450      29400
SCOTT            3000      36000
KING             5000      60000
FORD             3000      36000

6 rows selected.
```

DUPLICATE TABLE

Syntax

CREATE TABLE NEW-TABLE-NAME

AS

SELECT * FROM OLD-TABLE-NAME ;

```
SQL> CREATE TABLE EMPLOYEE
  2  AS
  3  SELECT * FROM EMP ;

Table created.
```

Note:

Creating duplicate table using aliasing is a permanent change is database.

LITERALS

Types of literals:

1. String ('name')
2. Number (7839)
3. Date ('31-MAR-2025')

Using data directly into the query is known as literals .

Note:

For string and date literals single quotes are mandatory but for number literal single quote are not mandatory while storing and retrieving data .

DUAL TABLE

It is a dummy table in oracle sql and it consists of only one column and one record.

We can make use of dual table to perform some basic operations.

```
SQL> DESC DUAL;
 Name
 ---------------------------------
 DUMMY

SQL> SELECT * FROM DUAL ;

D
-
X

SQL> SELECT 5 * 10 FROM DUAL ;

      5*10
----------
        50
```

DISTINCT

Distinct command is used to filter the columns and it will remove duplicate records from the specified column.

Syntax

```
SELECT DISTINCT(COLNAME) FROM TABLE-NAME ;
```

```
SQL> SELECT DISTINCT(JOB) FROM EMP ;

JOB
---------
CLERK
SALESMAN
PRESIDENT
MANAGER
ANALYST

SQL> SELECT DISTINCT(DEPTNO) FROM EMP;

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

Note:(error)

SELECT DISTINCT(JOB) , DISTINCT(DEPTNO) FROM EMP ;

OUTPUT : ERROR Because job(5 records) ≠ deptno(3 records)

For 1 select statement we can use only one distinct command .

LIMIT

Limit statement is used to retrieve records from table based on the limit value.

Syntax

SELECT * FROM TABLE-NAME

LIMIT VALUE ;

```
SELECT * FROM EMP LIMIT 2 ;
```

empno	ename	job	mgr	hiredate	sal	comm	deptno
7369	SMITH	CLERK	7902	1980-12-17	800.00	NULL	20
7499	ALLEN	SALESMAN	7698	1981-02-20	1600.00	300.00	30

```
SELECT * FROM EMP LIMIT 5;
```

empno	ename	job	mgr	hiredate	sal	comm	deptno
7369	SMITH	CLERK	7902	1980-12-17	800.00	NULL	20
7499	ALLEN	SALESMAN	7698	1981-02-20	1600.00	300.00	30
7521	WARD	SALESMAN	7698	1981-02-22	1250.00	500.00	30
7566	JONES	MANAGER	7839	1981-04-02	2975.00	NULL	20
7654	MARTIN	SALESMAN	7698	1981-09-28	1250.00	1400.00	30

OFFSET

Offset Statement Is Used To skip the records based on the offset value.

Limit statement is mandatory to perform offset statement.

Syntax

```
SELECT  * FROM TABLE-NAME
LIMIT VALUE
OFFSET VALUE ;
```

```
SELECT * FROM EMP LIMIT 5 OFFSET 3;
```

empno	ename	job	mgr	hiredate	sal	comm	deptno
7566	JONES	MANAGER	7839	1981-04-02	2975.00	NULL	20
7654	MARTIN	SALESMAN	7698	1981-09-28	1250.00	1400.00	30
7698	BLAKE	MANAGER	7839	1981-05-01	2850.00	NULL	30
7782	CLARK	MANAGER	7839	1981-06-09	2450.00	NULL	10
7788	SCOTT	ANALYST	7566	1982-12-09	3000.00	NULL	20

```
SELECT * FROM EMP LIMIT 1 OFFSET 13;
```

empno	ename	job	mgr	hiredate	sal	comm	deptno
7934	MILLER	CLERK	7782	1982-01-23	1300.00	NULL	10

a) Query to display starting 7 records from employee table.

```
SELECT * FROM EMP LIMIT 7 ;
```

empno	ename	job	mgr	hiredate	sal	comm	deptno
7369	SMITH	CLERK	7902	1980-12-17	800.00	NULL	20
7499	ALLEN	SALESMAN	7698	1981-02-20	1600.00	300.00	30
7521	WARD	SALESMAN	7698	1981-02-22	1250.00	500.00	30
7566	JONES	MANAGER	7839	1981-04-02	2975.00	NULL	20
7654	MARTIN	SALESMAN	7698	1981-09-28	1250.00	1400.00	30
7698	BLAKE	MANAGER	7839	1981-05-01	2850.00	NULL	30
7782	CLARK	MANAGER	7839	1981-06-09	2450.00	NULL	10

b) Query to display last record from employee table.

```
SELECT * FROM EMP
LIMIT 1
OFFSET (SELECT COUNT(*) – 1 FROM EMP ) ;
```

c) Query to display last 4 records from employee table.

```
SELECT * FROM EMP
LIMIT 4
OFFSET (SELECT COUNT(*) – 4 FROM EMP ) ;
```

GROUP BY CLAUSE, HAVING CLAUSE AND JOINS WILL BE DISCUSSED FURTHER.

DDL (DATA DEFINITION LANGUAGE)

Data definition language statements are used to create structure of the table and it is used to modify the structure of the table.

All DDL statements changes are permanent changes in the database.

TYPES OF DDL STATEMENTS

CREATE TABLE

Syntax:

```
CREATE TABLE TABLENAME
(COLUMN1 DATATYPE(SIZE) CONSTRAINT,
COLUMN2 DATATYPE(SIZE) CONSTRAINT);
```

Create table statement is used to create structure of the table with the help of datatype and constraint.

To create structure of the table datatype are mandatory but constraints are optional.

DATA TYPES

Data types are used to specify which type of data to be stored in particular column.

TYPES OF DATA TYPES

1. NUMBER / INT
2. CHAR
3. VARCHAR
4. VARCHAR2
5. DATE
6. DECIMAL
7. LOB (LARGE OBJECT)
    a. CLOB
    b. BLOB

NUMBER DATATYPE

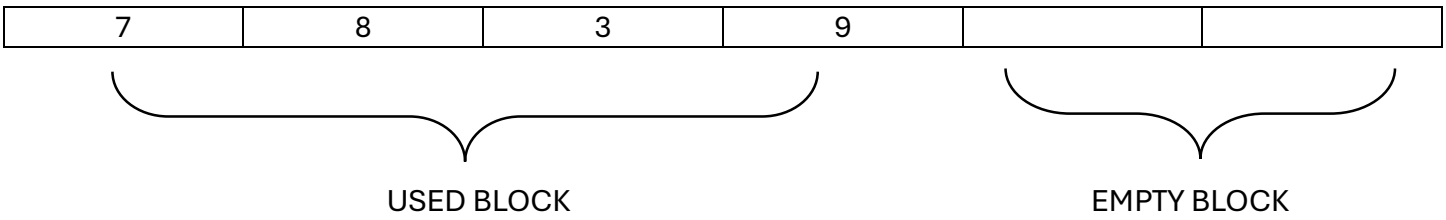
Number datatype is used to store only numeric value (0-9).

Number datatype follows variable memory length allocation (no wastage of memory).

Max size for number datatype is 38 blocks.

```
CREATE TABLE STUDENT
( SID NUMBER(6) ,
 MOBNO NUMBER(10) );
```

Memory Block:



CHAR DATATYPE

(0 – 9 , A – Z , a – z , # , * ……..)

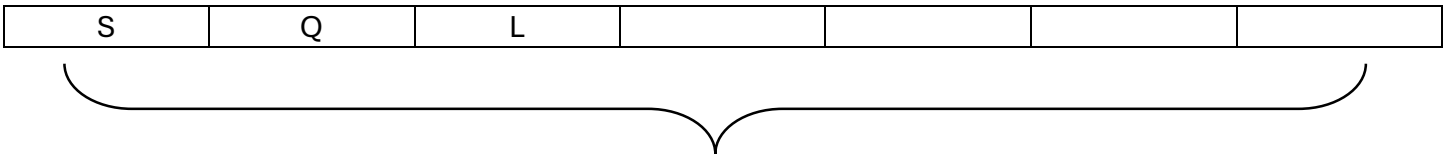
Character datatype is used to store alpha-numeric values along with some special symbols.

Character datatype follows fixed length memory location (wastage of storage)

The max size for character datatype is 2000 blocks.

```
CREATE TABLE STUDENT
( SID NUMBER(6),
 MOBNO NUMBER(10),
 SUBJECT CHAR(8) ) ;
```

Memory Allocation:



USED MEMORY BLOCK

VARCHAR DATATYPE

It is an updated version for character datatype and it is also used to store alpha-numeric values along with symbols.

Varchar datatype follows variable length memory allocation.

Max size for varchar datatype is 2000 blocks.

```
CREATE TABLE STUDENT
( SID NUMBER(6),
 MOBNO NUMBER(10),
 SUBJECT CHAR(8),
 SNAME VARCHAR(9) ) ;
```

Memory Allocation:

USED MEMORY BLOCK EMPTY BLOCK

VARCHAR2 DATATYPE

It is an updated version of varchar datatype and it is used to store alpha-numeric values along with some special symbols.

Varchar2 datatype follows variable length memory allocation.

Maximum size for varchar2 datatype is 4000 blocks

```
CREATE TABLE STUDENT
( SID NUMBER(6),
 MOBNO NUMBER(10),
 SUBJECT CHAR(8),
 SNAME VARCHAR(9),
 EMAIL_ID VARCHAR2(20) ) ;
```

DECIMAL DATATYPE

Decimal datatype is used to store decimal values (float values).

Syntax:

```
DECIMAL ( PRECISION, SCALE)
```

Precision – total no of digits after or before decimal point.

Scale – value after decimal point.

```
CREATE TABLE STUDENT
( SID NUMBER(6),
 MOBNO NUMBER(10),
 SUBJECT CHAR(8),
 SNAME VARCHAR(9),
 EMAIL_ID VARCHAR2(20),
 FEES  DECIMAL(7,2) ) ;
```

LOB (LARGE OBJECT DATATYPE)

CLOB (CHARACTER LOB)

It is used to store large amount of characters upto 4 GB storage.

BLOB (BINARY LOB)

It is used to store binary values of images, videos, and audios upto 4 GB storage.

DATE DATATYPE

Date datatype is used to store data values using date format.

Date format:

1) **DD – MON – YY** - 14 – MAR – 25 (DEFAULT)
2) **DD – MON – YYYY** - 14 – MAR - 2025
3) **DD – MM – YYYY** - 14 – 03 - 2025
4) **FMDAY – MON – YYYY –** FRIDAY – MAR - 2025
5) **DY – MON – YYYY** - FRI – MAR – 2025

```
CREATE TABLE STUDENT
( SID NUMBER(6),
 MOBNO NUMBER(10),
 SUBJECT CHAR(8),
 SNAME VARCHAR(9),
 EMAIL_ID VARCHAR2(20),
 FEES  DECIMAL(7,2) ,
 DOB  DATE ) ;
```

CONSTRAINTS (ATTRIBUTES)

Constraints are used to give additional conditions to the column to store the data.

Types of Constraints:

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

UNIQUE CONSTRAINT

It will not allow duplicate values.

it allows null values.

We can create multiple unique constraints in one table.

NOT NULL CONSTRAINT

It will allow duplicate values.

It will not allow null values.

We can create multiple not null constraints in one table.

```
CREATE TABLE EMPLOYEE
( ENAME  VARCHAR(10)  UNIQUE ,
 SALARY  NUMBER(8)  NOT NULL,
 JOB  VARCHAR(12)  NOT NULL ) ;
```

CHECK CONSTRAINT

It is used to store data in a particular column based on given condition.

```
CREATE TABLE EMPLOYEE
( ENAME  VARCHAR(10)  UNIQUE ,
 SALARY  NUMBER(8)  NOT NULL,
 AGE  NUMBER(2)  CHECK( AGE >= 18 ) ) ;
```

PRIMARY KEY

Primary key is also known as unique representation column in table.

It will not allow duplicate and null values.

We can have only one primary key constraint in one table.

It is an combination of unique and not null constraints.

Primary key is used to build relation on multiple tables with the help of foreign key using common column.

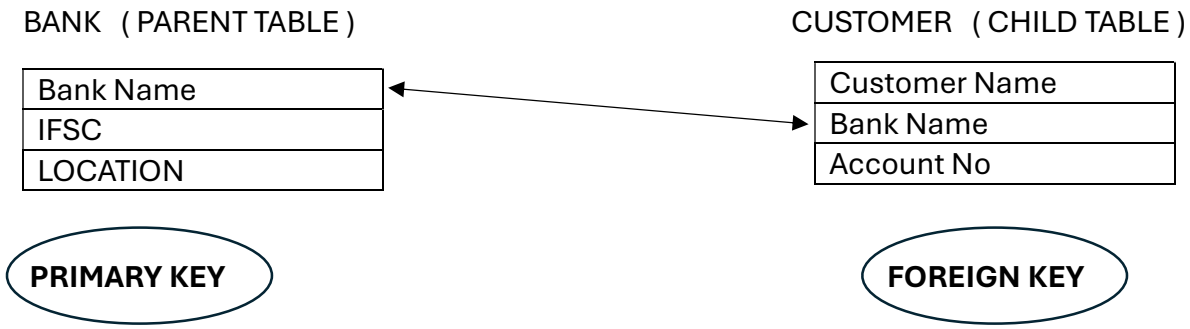
FOREIGN KEY

Foreign key is also known as reference integrity constraint .

The key that we are using for foreign key is references.

It will allow duplicate and null values.

We can create multiple foreign keys in one table.




```
CREATE TABLE BANK
( BANK_NAME  VARCHAR(10)  PRIMARY KEY ,
  IFCS  VARCHAR(10)  UNIQUE ,
  LOCATION  VARCHAR(20)  NOT NULL ) ;
```

```
CREATE TABLE CUSTOMER
( CUSTOMER_NAME  VARCHAR(15)  NOT NULL ,
  BANK_NAME  VARCHAR(10)  REFERENCES  BANK(BANK_NAME) ,
  ACC_NO  NUMBER(12)  UNIQUE  ) ;
```

RENAME TABLE

This statement is used to rename old table name with new table name.

It is a permanent change in database.

Syntax:

```
RENAME  OLD-TABLENAME  TO  NEW-TABLENAME ;
```

```
SQL> RENAME EMP TO EMPLOYEE ;

Table renamed.
```

DROP TABLE

This statement is used to delete structure of the table (used to delete the table).

Deleted table will go under recycle bin for a certain amount of time.

Syntax:

```
DROP TABLE TABLE-NAME ;
```

Syntax to Display Recycle Bin:

```
SHOW RECYCLEBIN
```

```
SQL> DROP TABLE EMPLOYEE ;

Table dropped.

SQL> SHOW RECYCLEBIN
ORIGINAL NAME    RECYCLEBIN NAME                OBJECT TYPE  DROP TIME
---------------- ------------------------------ ------------ -------------------
EMPLOYEE         BIN$QelSwfALQKS8UckIh4WCrw==$0 TABLE        2025-05-04:12:44:33
SQL>
```

FLASHBACK TABLE

It is used to restore deleted table from recycle bin after using drop statement.

Syntax:

```
FLASHBACK TABLE TABLENAME TO BEFORE DROP ;
```

```
SQL> SHOW RECYCLEBIN
ORIGINAL NAME    RECYCLEBIN NAME                OBJECT TYPE  DROP TIME
---------------- ------------------------------ ------------ -------------------
EMPLOYEE         BIN$QelSwfALQKS8UckIh4WCrw==$0 TABLE        2025-05-04:12:44:33
SQL>
SQL> FLASHBACK TABLE EMPLOYEE TO BEFORE DROP ;

Flashback complete.

SQL> SHOW RECYCLEBIN
SQL>
```

PURGE TABLE

Purge statement is used to delete the deleted table permanently from recycle bin.

Syntax:

```
PURGE TABLE TABLE-NAME ;
```

```
SQL> DROP TABLE EMPLOYEE;

Table dropped.

SQL> SHOW RECYCLEBIN
ORIGINAL NAME    RECYCLEBIN NAME                OBJECT TYPE  DROP TIME
---------------- ------------------------------ ------------ -------------------
EMPLOYEE         BIN$in3q7i0+RI6FC5m6HKZWlQ==$0 TABLE        2025-05-04:12:52:47
SQL>
SQL> PURGE TABLE EMPLOYEE ;

Table purged.
```

TRUNCATE TABLE

This statement is used to delete the entire records from specified table but the structure of the table remains the same.

We can't restore deleted records after using truncate.

Syntax:

```
TRUNCATE TABLE TABLE-NAME ;
```

```
SQL> TRUNCATE TABLE EMPLOYEE ;

Table truncated.

SQL> SELECT * FROM EMPLOYEE ;

no rows selected
```

ALTER TABLE

Alter statements are used to modify structure of the table.

1) **ADD SINGLE COLUMN**
   This statement is used to add a single column to the existing table.
   It is a permanent change.

Syntax:

```
ALTER TABLE TABLE-NAME ADD ( COLUMN-NAME DATATYPE(SIZE) ) ;
```

```
SQL> ALTER TABLE EMPLOYEE ADD(MOBNO NUMBER(10)) ;

Table altered.

SQL> SELECT * FROM EMPLOYEE ;

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

2) ADD MULTIPLE COLUMN

This statement is used to add multiple columns at a time to the existing table.

Syntax:

```
ALTER TABLE TABLE-NAME
ADD ( COLUMN1 DATATYPE(SIZE) ,
      ( COLUMN2 DATATYPE(SIZE) ) ;
```

```
SQL> ALTER TABLE EMPLOYEE ADD(EMAIL_ID VARCHAR(20) , AGE NUMBER(2) ) ;

Table altered.

SQL> SELECT * FROM EMPLOYEE ;

     EMPNO ENAME      JOB              MGR HIREDATE         SAL       COMM     DEPTNO      MOBNO EMAIL_ID                AGE
---------- ---------- --------- ---------- --------- ---------- ---------- ---------- ---------- -------------------- ----------
      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.
```

3) DROP SINGLE COLUMN

This statement is used to delete single column from an existing table.

Syntax:

```
ALTER TABLE TABLE-NAME DROP COLUMN COLUMN-NAME ;
```

```
SQL> ALTER TABLE EMPLOYEE DROP COLUMN AGE ;

Table altered.

SQL> SELECT * FROM EMPLOYEE;

     EMPNO ENAME      JOB              MGR HIREDATE         SAL       COMM     DEPTNO      MOBNO EMAIL_ID
---------- ---------- --------- ---------- --------- ---------- ---------- ---------- ---------- ----------
      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.
```

4) DROP MULTIPLE COLUMN

This statement is used to drop multiple columns at a time from an existing table.

Syntax:

```
ALTER TABLE TABLE-NAME DROP ( COLUMN1 , COLUMN2 , COLUMN3 ) ;
```

```
SQL> ALTER TABLE EMPLOYEE DROP(MOBNO , EMAIL_ID) ;

Table altered.

SQL> SELECT * FROM EMPLOYEE ;

     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.
```

5) RENAME COLUMN NAME

This statement is used to rename old column-name to new column-name.

It is a permanent change.

Syntax:

```
ALTER TABLE TABLE-NAME RENAME COLUMN OLD-COLNAME TO NEW-COLNAME ;
```

```
SQL> ALTER TABLE EMPLOYEE RENAME COLUMN SAL TO ANNUAL_SAL ;

Table altered.

SQL> SELECT * FROM EMPLOYEE ;

     EMPNO ENAME      JOB              MGR HIREDATE  ANNUAL_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.
```

6) MODIFY COLUMN DATATYPE

Used to update old data type with new data type.

Syntax:

```
ALTER TABLE TABLE-NAME MODIFY ( COLNAME  NEW-DATA-TYPE(SIZE) ) ;
```

```
SQL> ALTER TABLE EMPLOYEE MODIFY(JOB CHAR(15)) ;

Table altered.
```

7) MODIFY COLUMN CONSTRAINTS

This statement is used to update new constraint to the specified constraint.

Syntax:

```
ALTER TABLE TABLE-NAME MODIFY ( COLUMN NEW-CONSTRAINT ) ;
```

```
SQL> ALTER TABLE EMPLOYEE MODIFY(EMPNO PRIMARY KEY) ;

Table altered.
```

8) **DROP PRIMARY KEY**

This statement is used to delete primary key constraint from an table.

Syntax:

```
ALTER TABLE TABLE-NAME DROP PRIMARY KEY ;
```

```
SQL> ALTER TABLE EMPLOYEE DROP PRIMARY KEY ;

Table altered.
```

DML (DATA MANIPULATION LANGUAGE)

DML statements are used to manage entire database.

Manage stands for storing the data in database, updating old data with new data and deleting unwanted data from the database.

All DML changes are temporary changes in the database and to make them permanent change we need to use Transaction Control Language (TCL) statements.

TYPES OF DML STATEMENTS:

1) **INSERT DATA (INSERT RECORDS)**
2) **UPDATE DATA (UPDATE VALUES)**
3) **DELETE DATA (DELETE RECORDS)**

INSERT DATA

This statement is used to create new records in the specified table.

The statement is used to store data in the database.

SYNTAX:

Syntax1:

```
INSERT INTO TABLE-NAME VALUES( VALUE1 , VALUE2 , VALUE3 ) ;
```

```
SQL> INSERT INTO EMPLOYEE VALUES(7954 , 'SAURAV' , 'CLERK' , '7566' , '18-NOV-83' , 1455 , 200 , 20 );

1 row created.

SQL> SELECT * FROM EMPLOYEE ;

     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
      7954 SAURAV     CLERK             7566 18-NOV-83       1455        200         20

15 rows selected.
```

Syntax2:

stores data in particular column.

INSERT INTO TABLE-NAME(COL1 , COL2) VALUES(VALUE1 , VALUE2) ;

```
SQL> INSERT INTO EMPLOYEE(ENAME , JOB) VALUES('ASHISH' , 'ANALYST' ) ;

1 row created.

SQL> SELECT * FROM EMPLOYEE ;

     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
      7954 SAURAV     CLERK             7566 18-NOV-83       1455        200         20
           ASHISH     ANALYST

16 rows selected.
```

Syntax3:

INSERT INTO TABLE-NAME VALUES(&COL1 , &COL2 , &COL3) ;

Note: & is used to take input from the user.

```
SQL> INSERT INTO EMPLOYEE VALUES (&EMPNO, '&ENAME', '&JOB', &MGR, TO_DATE('&HIREDATE', 'DD-MON-YY'), &SAL, &COMM, &DEPTNO);
Enter value for empno: 7865
Enter value for ename: ANKIT
Enter value for job: SALESMAN
Enter value for mgr: 7782
Enter value for hiredate: 21-MAR-82
Enter value for sal: 2300
Enter value for comm: 80
Enter value for deptno: 10
old   1: INSERT INTO EMPLOYEE VALUES (&EMPNO, '&ENAME', '&JOB', &MGR, TO_DATE('&HIREDATE', 'DD-MON-YY'), &SAL, &COMM, &DEPTNO)
new   1: INSERT INTO EMPLOYEE VALUES (7865, 'ANKIT', 'SALESMAN', 7782, TO_DATE('21-MAR-82', 'DD-MON-YY'), 2300, 80, 10)

1 row created.


SQL> SELECT * FROM EMPLOYEE ;

     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
      7954 SAURAV     CLERK             7566 18-NOV-83       1455        200         20
           ASHISH     ANALYST
      7865 ANKIT      SALESMAN          7782 21-MAR-82       2300         80         10

17 rows selected.
```

Syntax4:

INSERT INTO TABLE-NAME(COL1 , COL2) VALUES((&COL1 , &COL2) ;

```
SQL> INSERT INTO EMPLOYEE(ENAME , JOB) VALUES('&ENAME' , '&JOB');
Enter value for ename: PAUL
Enter value for job: MANAGER
old   1: INSERT INTO EMPLOYEE(ENAME , JOB) VALUES('&ENAME' , '&JOB')
new   1: INSERT INTO EMPLOYEE(ENAME , JOB) VALUES('PAUL' , 'MANAGER')

1 row created.

SQL> SELECT * FROM EMPLOYEE ;

     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
      7954 SAURAV     CLERK             7566 18-NOV-83       1455        200         20
           ASHISH     ANALYST
      7865 ANKIT      SALESMAN          7782 21-MAR-82       2300         80         10
           PAUL       MANAGER

18 rows selected.
```

Syntax5:

Used to store multiple records in multiple tables at a time.

```
INSERT ALL
        INTO TABLE-NAME1 ( COL1 , COL2 ) VALUES ( VALUE1 , VALUE2 )
        INTO TABLE-NAME2 ( COL1 , COL2 , COL3 ) VALUES ( VALUE1 , VALUE2 , VALUE3 )
( SELECT * FROM DUAL ) ;
```

```
SQL> INSERT ALL
  2         INTO EMPLOYEE(ENAME , JOB) VALUES('JOHN' , 'CLERK')
  3         INTO DEPT(DEPTNO , DNAME) VALUES(50 , 'QUALITY')
  4  (SELECT * FROM DUAL);

2 rows created.

SQL> SELECT * FROM EMPLOYEE;

     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
      7954 SAURAV     CLERK             7566 18-NOV-83        1455        200         20
           ASHISH     ANALYST
      7865 ANKIT      SALESMAN          7782 21-MAR-82        2300         80         10
           PAUL       MANAGER
           JOHN       CLERK

19 rows selected.

SQL> SELECT * FROM DEPT;

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

Syntax6:

Copies one table data into another table but the structure remains the same.

INSERT INTO TARGET-TABLE SELECT * FROM SOURCE-TABLE ;

```
SQL> INSERT INTO EMPLOYEE
  2  SELECT * FROM EMP ;

14 rows created.

SQL> SELECT * FROM EMPLOYEE;
```

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
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

```
28 rows selected.
```

Syntax7:

```
INSERT INTO TARGET-TABLE( COL1 , COL2 )
SELECT COLNAME1 , COLNAME2  FROM SOURCE-TABLE ;
```

```
SQL> INSERT INTO EMPLOYEE(ENAME , SAL)
  2  SELECT ENAME , SAL FROM EMP ;

14 rows created.

SQL> SELECT * FROM EMPLOYEE;

     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
           SMITH                                             800
           ALLEN                                            1600
           WARD                                             1250
           JONES                                            2975
           MARTIN                                           1250
           BLAKE                                            2850
           CLARK                                            2450
           SCOTT                                            3000
           KING                                             5000
           TURNER                                           1500
           ADAMS                                            1100
           JAMES                                             950
           FORD                                             3000
           MILLER                                           1300

28 rows selected.
```

UPDATE STATEMENT

Update statement is used to update old data with new data.

It is a temporary change in the database.

Syntax1:

UPDATE TABLE-NAME SET COL-NAME = NEW VALUE WHERE CONDITION ;

```
SQL> UPDATE EMPLOYEE SET SAL = 7000 WHERE ENAME = 'KING' ;

1 row updated.

SQL> SELECT * FROM EMPLOYEE;

     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       7000                    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.
```

UPDATE TABLE-NAME SET COLNAME = NEW-VALUE ;

```
SQL> UPDATE EMPLOYEE SET JOB = 'ANALYST' ;

14 rows updated.

SQL> SELECT * FROM EMPLOYEE ;

     EMPNO ENAME      JOB              MGR HIREDATE         SAL       COMM     DEPTNO
---------- ---------- --------- ---------- --------- ---------- ---------- ----------
      7369 SMITH      ANALYST         7902 17-DEC-80        800                    20
      7499 ALLEN      ANALYST         7698 20-FEB-81       1600        300         30
      7521 WARD       ANALYST         7698 22-FEB-81       1250        500         30
      7566 JONES      ANALYST         7839 02-APR-81       2975                    20
      7654 MARTIN     ANALYST         7698 28-SEP-81       1250       1400         30
      7698 BLAKE      ANALYST         7839 01-MAY-81       2850                    30
      7782 CLARK      ANALYST         7839 09-JUN-81       2450                    10
      7788 SCOTT      ANALYST         7566 19-APR-87       3000                    20
      7839 KING       ANALYST              17-NOV-81       7000                    10
      7844 TURNER     ANALYST         7698 08-SEP-81       1500          0         30
      7876 ADAMS      ANALYST         7788 23-MAY-87       1100                    20
      7900 JAMES      ANALYST         7698 03-DEC-81        950                    30
      7902 FORD       ANALYST         7566 03-DEC-81       3000                    20
      7934 MILLER     ANALYST         7782 23-JAN-82       1300                    10

14 rows selected.
```

Syntax2:

```
UPDATE TABLE-NAME SET COLNAME1 = NEW-VALUE ,
COLNAME2 = NEW-VALUE WHERE CONDITION ;
```

```
SQL> UPDATE EMPLOYEE SET JOB = 'ANALYST' ,
  2  COMM = 700  WHERE ENAME = 'SMITH' ;

1 row updated.

SQL> SELECT * FROM EMPLOYEE ;

     EMPNO ENAME      JOB              MGR HIREDATE         SAL       COMM     DEPTNO
---------- ---------- --------- ---------- --------- ---------- ---------- ----------
      7369 SMITH      ANALYST         7902 17-DEC-80        800        700         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.
```

CASE FUNCTION

This function is used to update multiple results for multiple conditions.

Syntax:

```
UPDATE TABLE-NAME SET COLNAME =
CASE
        WHEN CONDITION THEN RESULT
        WHEN CONDITION THEN RESULT
ELSE
        DEFAULT RESULT
END;
```

```
SQL> UPDATE EMPLOYEE SET JOB =
  2  CASE
  3     WHEN ENAME = 'KING' THEN 'ANALYST'
  4     WHEN ENAME = 'SMITH' THEN 'SALESMAN'
  5  ELSE
  6  JOB
  7  END,
  8                        SAL =
  9  CASE
 10     WHEN ENAME = 'KING' THEN 6800
 11     WHEN ENAME = 'SMITH' THEN 3400
 12  ELSE
 13  SAL
 14  END;

14 rows updated.

SQL> SELECT * FROM EMPLOYEE ;

     EMPNO ENAME      JOB              MGR HIREDATE         SAL        COMM     DEPTNO
---------- ---------- --------- ---------- --------- ---------- ---------- ----------
      7369 SMITH      SALESMAN         7902 17-DEC-80       3400        700         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       ANALYST               17-NOV-81       6800                    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.
```

DELETE STATEMENT

Delete statement is used to delete record from specified table.

We can restore deleted records using rollback statement.

Syntax:

```
DELETE FROM TABLE-NAME WHERE CONDITION ;
```

```
DELETE TABLE-NAME ;
```

```
SQL> DELETE FROM EMPLOYEE WHERE ENAME = 'KING' ;

1 row deleted.

SQL> SELECT * FROM EMPLOYEE ;

     EMPNO ENAME      JOB              MGR HIREDATE         SAL       COMM     DEPTNO
---------- ---------- --------- ---------- --------- ---------- ---------- ----------
      7369 SMITH      SALESMAN        7902 17-DEC-80       3400        700         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
      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

13 rows selected.

SQL> DELETE EMPLOYEE ;

13 rows deleted.

SQL> SELECT * FROM EMPLOYEE ;

no rows selected
```

TCL (TRANSACTION CONTROL LANGUAGE)

TCL statements are also known as Data Transaction Language Statements (DTL).

All DML changes are temporary changes in the database and to make them permanent change, we need to use TCL statements.

COMMIT

Commit is used to save the DML changes permanently.

Syntax:

COMMIT ;

```
SQL> SELECT * FROM EMPLOYEE ;

no rows selected

SQL> INSERT INTO EMPLOYEE
  2  SELECT * FROM EMP ;

14 rows created.

SQL> COMMIT ;

Commit complete.
```

ROLLBACK

```
SQL> DELETE EMPLOYEE ;

14 rows deleted.

SQL> SELECT * FROM EMPLOYEE ;

no rows selected

SQL> ROLLBACK ;

Rollback complete.

SQL> SELECT * FROM EMPLOYEE ;

     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.
```

SAVEPOINT

This statement is used to restore particular transaction with the help of created save point.

Syntax:

```
SAVEPOINT SAVEPOINT-NAME ;
```

```
ROLLBACK TO SAVEPOINT-NAME ;
```

```
SQL> DELETE FROM EMPLOYEE WHERE SAL < 1500 ;

6 rows deleted.

SQL> SAVEPOINT NEW_SAL ;

Savepoint created.

SQL> DELETE EMPLOYEE ;

8 rows deleted.

SQL> ROLLBACK TO NEW_SAL ;

Rollback complete.

SQL> SELECT * FROM EMPLOYEE ;

     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

8 rows selected.

SQL> ROLLBACK ;

Rollback complete.

SQL> SELECT * FROM EMPLOYEE ;

     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>
```

DCL (DATA CONTROL LANGUAGE)

GRANT

This statement is used to give table access to another user.

Syntax:

```
GRANT SELECT ON TABLE-NAME TO USERNAME ;
```

```
SQL> GRANT SELECT ON EMP TO HR ;

Grant succeeded.


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.
```

REVOKE

This statement is used to take table access back from another user.

Syntax:

```
REVOKE SELECT ON TABLE-NAME FROM USERNAME ;
```

```
SQL> REVOKE SELECT ON EMP FROM HR ;

Revoke succeeded.
```

OPERATORS

Operators are some special symbols that are used to perform operations on operands.

To perform some operation in between two operands, we can make use of operators.

Every data admin and data visualization professional who wants to manage and maintain databases can make use of operators.

TYPES OF OPERATORS:

1) ARITHMETIC OPERATOR
2) RELATIONAL OPERATOR
3) LOGICAL OPERATOR
4) SPECIAL OPERATOR
5) CONCATENATION OPERATOR
6) SET OPERATOR

ARITHMETIC OPERATOR

Arithmetic operator consists of addition, subtraction, multiplication, and division.

All arithmetic operators are used to perform expressions.

Expressions

Combination of two or more value is known as expression.

4 + 5

Sal * 12

We can perform expressions only inside select statement.

QUERIES ON ARITHMETIC OPERATORs

a) query to display name of employee along with 6000 rupees increment in salary.

```
SQL> SELECT ENAME , SAL , SAL+6000 FROM EMP ;

ENAME             SAL   SAL+6000
---------- ---------- ----------
SMITH             800       6800
ALLEN            1600       7600
WARD             1250       7250
JONES            2975       8975
MARTIN           1250       7250
BLAKE            2850       8850
CLARK            2450       8450
SCOTT            3000       9000
KING             5000      11000
TURNER           1500       7500
ADAMS            1100       7100
JAMES             950       6950
FORD             3000       9000
MILLER           1300       7300

14 rows selected.
```

b) query to display name of employee along with 200 rupees decrement in salary.

```
SQL> SELECT ENAME , SAL , SAL-200 FROM EMP ;

ENAME             SAL    SAL-200
---------- ---------- ----------
SMITH             800        600
ALLEN            1600       1400
WARD             1250       1050
JONES            2975       2775
MARTIN           1250       1050
BLAKE            2850       2650
CLARK            2450       2250
SCOTT            3000       2800
KING             5000       4800
TURNER           1500       1300
ADAMS            1100        900
JAMES             950        750
FORD             3000       2800
MILLER           1300       1100

14 rows selected.
```

c) query to display name of employee along with designation and annual salary.

```
SQL> SELECT ENAME , JOB , SAL*12 FROM EMP ;

ENAME      JOB           SAL*12
---------- --------- ----------
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.
```

d) query to display employee name along with their 70% increment in salary.

```
SQL> SELECT ENAME ,SAL , SAL*1.7 FROM EMP ;

ENAME             SAL    SAL*1.7
---------- ---------- ----------
SMITH             800       1360
ALLEN            1600       2720
WARD             1250       2125
JONES            2975     5057.5
MARTIN           1250       2125
BLAKE            2850       4845
CLARK            2450       4165
SCOTT            3000       5100
KING             5000       8500
TURNER           1500       2550
ADAMS            1100       1870
JAMES             950       1615
FORD             3000       5100
MILLER           1300       2210

14 rows selected.
```

e) query to display name of employee along with their designation and their daily salary.

```
SQL> SELECT ENAME , JOB , (SAL*12)/365 FROM EMP ;

ENAME      JOB       (SAL*12)/365
---------- --------- ------------
SMITH      CLERK       26.3013699
ALLEN      SALESMAN    52.6027397
WARD       SALESMAN    41.8958904
JONES      MANAGER     97.8082192
MARTIN     SALESMAN    41.8958904
BLAKE      MANAGER     93.6986301
CLARK      MANAGER     80.5479452
SCOTT      ANALYST      98.630137
KING       PRESIDENT   164.383562
TURNER     SALESMAN    49.3150685
ADAMS      CLERK       36.1643836
JAMES      CLERK       31.2328767
FORD       ANALYST      98.630137
MILLER     CLERK        42.739726

14 rows selected.
```

f) query to display details of employee with their annual salary.

```
SQL> SELECT EMP.* , SAL*12 FROM EMP ;

     EMPNO ENAME      JOB              MGR HIREDATE         SAL       COMM     DEPTNO     SAL*12
---------- ---------- --------- ---------- --------- ---------- ---------- ---------- ----------
      7369 SMITH      CLERK           7902 17-DEC-80        800                    20       9600
      7499 ALLEN      SALESMAN        7698 20-FEB-81       1600        300         30      19200
      7521 WARD       SALESMAN        7698 22-FEB-81       1250        500         30      15000
      7566 JONES      MANAGER         7839 02-APR-81       2975                    20      35700
      7654 MARTIN     SALESMAN        7698 28-SEP-81       1250       1400         30      15000
      7698 BLAKE      MANAGER         7839 01-MAY-81       2850                    30      34200
      7782 CLARK      MANAGER         7839 09-JUN-81       2450                    10      29400
      7788 SCOTT      ANALYST         7566 19-APR-87       3000                    20      36000
      7839 KING       PRESIDENT            17-NOV-81       5000                    10      60000
      7844 TURNER     SALESMAN        7698 08-SEP-81       1500          0         30      18000
      7876 ADAMS      CLERK           7788 23-MAY-87       1100                    20      13200
      7900 JAMES      CLERK           7698 03-DEC-81        950                    30      11400
      7902 FORD       ANALYST         7566 03-DEC-81       3000                    20      36000
      7934 MILLER     CLERK           7782 23-JAN-82       1300                    10      15600

14 rows selected.
```

RELATIONAL OPERATOR

Relational operator consists of < , > , <= , >= , != and =.

All relational operators should be used only inside the WHERE clause.

relational operators are used to give the condition to the query, and it will execute the output based on the given condition.

QUERIES ON RELATIONAL OPERATORS

a) Query to display details of employees who are earning less than

```
SQL> SELECT * FROM EMP WHERE SAL < 1000 ;

     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
```

b) Query to display name of employees , annual salary , if annual salary is more than 37000.

```
SQL> SELECT ENAME , SAL*12 FROM EMP WHERE SAL*12 > 37000 ;

ENAME          SAL*12
---------- ----------
KING            60000
```

c) Query to display details of employees who joined after 1981.

```
SQL> SELECT * FROM EMP WHERE HIREDATE > '31-DEC-1981' ;

     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
```

d) Query to display details of employees who joined before 1981.

```
SQL> SELECT * FROM EMP WHERE HIREDATE < '01-DEC-1981' ;

     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
      7839 KING       PRESIDENT            17-NOV-81       5000                    10
      7844 TURNER     SALESMAN        7698 08-SEP-81       1500          0         30

9 rows selected.
```

e) Query to display smith info from employee table.

```
SQL> SELECT * FROM EMP WHERE ENAME = 'SMITH' ;

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

LOGICAL OPERATOR

Logical operator consists of AND , OR , NOT , operators.

Whenever we are having more than one conditions , we have to use logical operators.

AND OPERATOR

Whenever we are having more than one condition and if all the condition is true then AND operator will execute the output.

OR OPERATOR

Whenever we are having more than one condition and if any one condition is true then OR operator will execute the output.

If we have multiple condition for same column then we have to use OR operator.

NOT OPERATOR

It is used to ignore the condition and we have to use NOT operator along with special operator.

Condition1	Condition2	AND	OR
TRUE	TRUE	TRUE	TRUE
TRUE	FALSE	FALSE	TRUE
FALSE	TRUE	FALSE	TRUE
FALSE	FALSE	FALSE	FALSE

QUERIES ON LOGICAL OPERATOR

a) Query to display details of employee who are earning more than 3000 and are in department 10.

```
SQL> SELECT * FROM EMP WHERE SAL > 3000 AND DEPTNO = 10 ;

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

b) Query to display Smith and Scott information.

```
SQL> SELECT * FROM EMP WHERE ENAME='SMITH' OR ENAME='SCOTT' ;

     EMPNO ENAME      JOB              MGR HIREDATE         SAL       COMM     DEPTNO
---------- ---------- --------- ---------- --------- ---------- ---------- ----------
      7369 SMITH      CLERK           7902 17-DEC-80        800                    20
      7788 SCOTT      ANALYST         7566 19-APR-87       3000                    20
```

c) Query to display Smith and Scott information if they are earning more than 2000.

```
SQL> SELECT * FROM EMP WHERE (ENAME = 'SMITH' OR ENAME='SCOTT') AND SAL > 2000 ;

     EMPNO ENAME      JOB              MGR HIREDATE         SAL       COMM     DEPTNO
---------- ---------- --------- ---------- --------- ---------- ---------- ----------
      7788 SCOTT      ANALYST         7566 19-APR-87       3000                    20
```

d) Query to display details of employee who belongs from salesman and analyst designation.

```
SQL> SELECT * FROM EMP WHERE JOB='SALESMAN' OR JOB='ANALYST' ;

     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
      7788 SCOTT      ANALYST          7566 19-APR-87       3000                    20
      7844 TURNER     SALESMAN         7698 08-SEP-81       1500          0         30
      7902 FORD       ANALYST          7566 03-DEC-81       3000                    20

6 rows selected.
```

e) Query to display salesman information if they joined before 1987.

```
SQL> SELECT * FROM EMP WHERE JOB='SALESMAN' AND HIREDATE < '01-JAN-1987' ;

     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
```

f) Query to display details of employee who belong from analyst designation and working in department number 20.

```
SQL> SELECT * FROM EMP WHERE JOB='ANALYST' AND DEPTNO=20 ;

     EMPNO ENAME      JOB              MGR HIREDATE         SAL       COMM     DEPTNO
---------- ---------- --------- ---------- --------- ---------- ---------- ----------
      7788 SCOTT      ANALYST          7566 19-APR-87       3000                    20
      7902 FORD       ANALYST          7566 03-DEC-81       3000                    20
```

g) Query to display details of employee who joined after 1980 and in department number 20 and 30.

```
SQL> SELECT * FROM EMP WHERE HIREDATE > '31-DEC-1980' AND (DEPTNO = 20 OR 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
      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
      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
      7902 FORD       ANALYST          7566 03-DEC-81       3000                    20

10 rows selected.
```

SPECIAL OPERATOR

Special operators consist of (IN , BETWEEN , LIKE , IS , NOT IN , NOT BETWEEN , NOT LIKE , IS NOT).

All special operators should be used only inside the WHERE clause.

IN OPERATOR

Instead of using OR operator multiple times we can make use of IN operator.

Condition should be for same column.

a) Query to display details of employee who belong from department number 20, 30, 40 .

```
SQL> SELECT * FROM EMP WHERE DEPTNO IN(20,30,40) ;

     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
      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
      7902 FORD       ANALYST         7566 03-DEC-81       3000                    20

11 rows selected.
```

b) Query to display employee details apart from smith and scott.

```
SQL> SELECT * FROM EMP WHERE ENAME NOT IN('SMITH' , 'SCOTT' ) ;

     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
      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

12 rows selected.
```

c) Query to display details of employee who belongs from salesman and analyst designation.

```
SQL> SELECT * FROM EMP WHERE JOB IN('SALESMAN' , 'ANALYST') ;

     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
      7788 SCOTT      ANALYST          7566 19-APR-87       3000                    20
      7844 TURNER     SALESMAN         7698 08-SEP-81       1500          0         30
      7902 FORD       ANALYST          7566 03-DEC-81       3000                    20

6 rows selected.
```

BETWEEN OPERATOR

Whenever we are having values in range we have to use between operator.

a) Query to display details of employee who are earning from 3000 to 5000.

```
SQL> SELECT * FROM EMP WHERE SAL BETWEEN 3000 AND 5000 ;

     EMPNO ENAME      JOB              MGR HIREDATE         SAL        COMM     DEPTNO
---------- ---------- --------- ---------- --------- ---------- ---------- ----------
      7788 SCOTT      ANALYST          7566 19-APR-87       3000                    20
      7839 KING       PRESIDENT             17-NOV-81       5000                    10
      7902 FORD       ANALYST          7566 03-DEC-81       3000                    20
```

b) Query to display details of employee who are earning less than 1000 and more than 3000.

```
SQL> 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
```

c) Query to display details of employee who joined in year 1981.

```
SQL> SELECT * FROM EMP WHERE HIREDATE BETWEEN '01-JAN-1981' AND '31-DEC-1981' ;

     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

10 rows selected.
```

d) Query to display details of employee who joined before 1981 and after 1982.

```
SQL> SELECT * FROM EMP WHERE HIREDATE NOT BETWEEN '01-JAN-1981' AND '31-DEC-1982' ;

     EMPNO ENAME      JOB              MGR HIREDATE         SAL       COMM     DEPTNO
---------- ---------- --------- ---------- --------- ---------- ---------- ----------
      7369 SMITH      CLERK           7902 17-DEC-80        800                    20
      7788 SCOTT      ANALYST         7566 19-APR-87       3000                    20
      7876 ADAMS      CLERK           7788 23-MAY-87       1100                    20
```

e) Query to display details of employee apart from who joined in 1981.

```
SQL> SELECT * FROM EMP WHERE HIREDATE NOT BETWEEN '01-JAN-1981' AND '31-DEC-1981' ;

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

LIKE OPERATOR

Like operator is used for pattern matching ..

We can achieve pattern matching with the help of SQL wild cards.

Types of Wild cards:

- % (percentage) : it derives 0 to n character.
- _ (underscore) : it derives single character.

QUERIES ON LIKE OPERATOR

a) Query to display details of employee whose name start with s.

```
SQL> SELECT * FROM EMP WHERE ENAME LIKE 'S%' ;

     EMPNO ENAME      JOB              MGR HIREDATE         SAL       COMM     DEPTNO
---------- ---------- --------- ---------- --------- ---------- ---------- ----------
      7369 SMITH      CLERK           7902 17-DEC-80        800                    20
      7788 SCOTT      ANALYST         7566 19-APR-87       3000                    20
```

b) Query to display details of employee whose name ends with s.

```
SQL> SELECT * FROM EMP WHERE ENAME LIKE '%S' ;

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

c) Query to display details of employee whose designation contains alphabet s.

```
SQL> SELECT * FROM EMP WHERE JOB LIKE '%S%' ;

     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
      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

7 rows selected.
```

d) Query to display details of employee whose designation contains more than 1 times alphabet s.

```
SQL> SELECT * FROM EMP WHERE JOB LIKE '%S%S%' ;

     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
```

e) Query to display details of employee who are earning 3 digit salary.

```
SQL> SELECT * FROM EMP WHERE SAL LIKE '___' ;

     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
```

f) Query to display details of employee whose designation starting 3rd character is alphabet a.

```
SQL> SELECT * FROM EMP WHERE JOB LIKE '__A%' ;

     EMPNO ENAME      JOB              MGR HIREDATE         SAL       COMM     DEPTNO
---------- ---------- --------- ---------- --------- ---------- ---------- ----------
      7788 SCOTT      ANALYST          7566 19-APR-87       3000                    20
      7902 FORD       ANALYST          7566 03-DEC-81       3000                    20
```

g) Query to display details of employee whose designation ending 2nd character is alphabet r.

```
SQL> SELECT * FROM EMP WHERE JOB LIKE '%R_' ;

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

h) Query to display details of employee except whose name start with s.

```
SQL> SELECT * FROM EMP WHERE ENAME NOT LIKE 'S%' ;

     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
      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

12 rows selected.
```

IS OPERATOR

IT is used to find null values.

QUERIES ON IS OPERATOR

a) Query to display details of employee who are not earning commission.

```
SQL> SELECT * FROM EMP WHERE COMM IS NULL ;

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

10 rows selected.
```

b) Query to display details of employee who are earning commission.

```
SQL> 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
```

c) Query to display details of employee who don't have reporting manager.

```
SQL> SELECT * FROM EMP WHERE MGR IS NULL ;

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

d) Query to display details of employee who have reporting manager.

```
SQL> SELECT * FROM EMP WHERE MGR IS NOT NULL ;

     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
      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

13 rows selected.
```

CONCATENATION OPERATOR

Joining or merging of two or more literals or columns is known as concatenation.

To achieve concatenation we have to use parallel symbol || .

Syntax:

```
SELECT LITERL || LITERAL AS FORMAT  FROM TABLE-NAME ;

SELECT COLNAME1 || COLNAME2 AS FORMAT FROM TABLE-NAME ;
```

a) Query to print output in below format .

SMITH IS EARNING 800

```
SQL> SELECT 'SMITH' || ' IS ' || ' EARNING ' || 800
  2  AS FORMAT FROM DUAL ;

FORMAT
--------------------
SMITH IS  EARNING 800
```

```
SQL> SELECT ENAME || ' IS EARNING ' || SAL
  2  AS FORMAT FROM EMP
  3  WHERE ENAME = 'SMITH' ;

FORMAT
----------------------------------------
SMITH IS EARNING 800
```

SET OPERATOR

Set operator are used to combine the result of two or more select statement.

Syntax:

```
SELECT COLNAME FROM TABLE-NAME 1
< SET OPEARTOR >
SELECT COLNAME FROM TABLE-NAME 2 ;
```

Types of Set Operators:

1. UNION
2. UNION ALL
3. INTERSECT
4. MINUS

Note:

While combining results of two or more select statements using set operator, in every select statement number of columns should be the same.

In every select statement, we have to use similar data type columns.

UNION SET OPERATOR

Union set operator is used to combine the result of two or more select statement.

It will remove duplicate values from all select statements.

It will display matched and unmatched records from all select statements.

```
SELECT DEPTNO FROM EMP
UNION
SELECT DEPTNO FROM DEPT ;
```

```
SQL> SELECT DEPTNO FROM EMP
  2  UNION
  3  SELECT DEPTNO FROM DEPT ;

    DEPTNO
----------
        10
        20
        30
        40
```

UNION ALL SET OPERATOR

Union all is used to combine the results of two or more select statements.

It will not remove duplicate values.

It will display matched and unmatched records from all select statements.

```
SELECT DEPTNO FROM EMP

UNION ALL

SELECT DEPTNO FROM DEPT ;
```

```
SQL> SELECT DEPTNO FROM EMP
  2  UNION ALL
  3  SELECT DEPTNO FROM DEPT ;

    DEPTNO
----------
        20
        30
        30
        20
        30
        30
        10
        20
        10
        30
        20
        30
        20
        10
        10
        20
        30
        40

18 rows selected.
```

INTERSECT SET OPERATOR

It will display only matched records from all select statements.

```
SELECT DEPTNO FROM EMP
INTERSECT
SELECT DEPTNO FROM DEPT;
```

```
SQL> SELECT DEPTNO FROM EMP
  2  INTERSECT
  3  SELECT DEPTNO FROM DEPT ;

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

MINUS SET OPERATOR

It will display unmatched records only from first select statement.

```
SELECT DEPTNO FROM EMP
MINUS
SELECT DEPTNO FROM DEPT;
```

```
SQL> SELECT DEPTNO FROM EMP
  2  MINUS
  3  SELECT DEPTNO FROM DEPT ;

no rows selected
```

```
SELECT DEPTNO FROM DEPT
MINUS
SELECT DEPTNO FROM EMP;
```

```
SQL> SELECT DEPTNO FROM DEPT
  2  MINUS
  3  SELECT DEPTNO FROM EMP ;

    DEPTNO
----------
        40
```

FUNCTIONS

Functions are important features of SQL.

Functions are used to perform calculations on data and to modify particular data.

Types of functions:

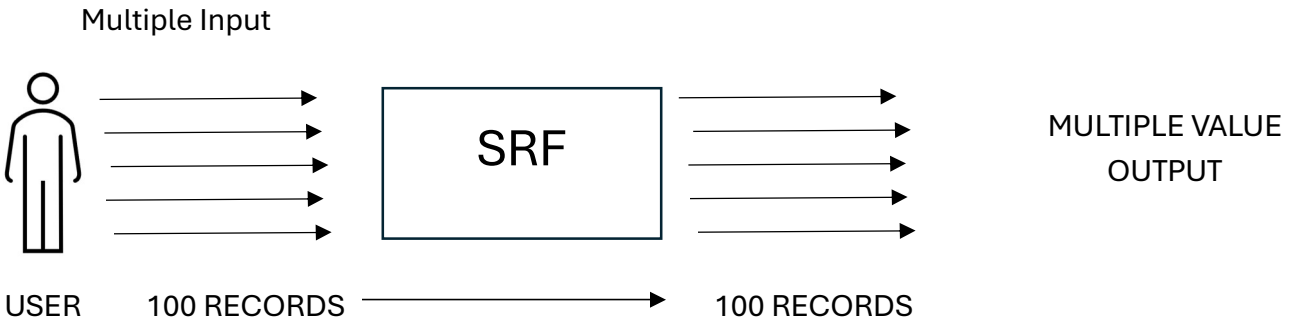
1. **SINGLE ROW FUNCTIONS**
    a. Upper
    b. Lower
    c. Initcap
    d. Length
    e. Substr
    f. Replace
    g. Instr
    h. Ascii
    i. Nvl

2. **MULTIROW FUNCTIONS**
    a. Max
    b. Min
    c. Avg
    d. Sum
    e. Count

3. **DATE FUNCTIONS**
    a. Sysdate
    b. Sys timestamp
    c. To_date
    d. To_char
    e. Last_day
    f. Trunc
    g. Months_between
    h. Add_months
    i. Next_day

4. **WINDOW FUNCTIONS**
    a. Row number
    b. Dense rank
    c. Rank

SINGLE ROW FUNCTIONS



Single row function are also known as scalar functions.

All single row functions will take multiple inputs from user and it will execute multiple value output.

```
SQL> SELECT LOWER(ENAME) FROM EMP ;

LOWER(ENAM
----------
smith
allen
ward
jones
martin
blake
clark
scott
king
turner
adams
james
ford
miller

14 rows selected.
```

All single row functions we can use inside the select statement and the where clause.

```
SQL> SELECT ENAME , LENGTH(ENAME) FROM EMP ;

ENAME      LENGTH(ENAME)
---------- -------------
SMITH                  5
ALLEN                  5
WARD                   4
JONES                  5
MARTIN                 6
BLAKE                  5
CLARK                  5
SCOTT                  5
KING                   4
TURNER                 6
ADAMS                  5
JAMES                  5
FORD                   4
MILLER                 6

14 rows selected.
```

```
SQL> SELECT * FROM EMP WHERE LENGTH(ENAME) = 6 ;

     EMPNO ENAME      JOB              MGR HIREDATE         SAL       COMM     DEPTNO
---------- ---------- --------- ---------- --------- ---------- ---------- ----------
      7654 MARTIN     SALESMAN         7698 28-SEP-81       1250       1400         30
      7844 TURNER     SALESMAN         7698 08-SEP-81       1500          0         30
      7934 MILLER     CLERK            7782 23-JAN-82       1300                    10
```

Combination of columns and single row function is possible.

Types of Single row functions:

UPPER FUNCTION

This function is used to convert all lowercase characters into uppercase.

We can use this function inside select statement and where clause.

Syntax:

```
SELECT UPPER( COLNAME / STRING ) FROM TABLE-NAME ;
```

```
SQL> SELECT UPPER('qspiders') FROM DUAL ;

UPPER('Q
--------
QSPIDERS
```

```
SELECT * FROM TABLE-NAME

WHERE COLNAME = UPPER( VALUE ) ;
```

```
SQL> SELECT * FROM EMP
  2  WHERE ENAME = UPPER('smith') ;

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

LOWER FUNCTION

This function is used to convert all uppercase characters into lowercase.

Syntax:

```
SELECT LOWER( COLNAME / STRING ) FROM TABLE-NAME ;
```

```
SQL> SELECT LOWER('QSPIDERS') FROM DUAL ;

LOWER('Q
--------
qspiders
```

```
SQL> SELECT LOWER(ENAME) FROM EMP ;

LOWER(ENAM
----------
smith
allen
ward
jones
martin
blake
clark
scott
king
turner
adams
james
ford
miller

14 rows selected.
```

INITCAP FUNCTION

This function is used to convert initial first character of string into uppercase and remaining all character in lowercase.

Syntax:

```
SELECT INITCAP( COLNAME / STRING ) FROM TABLE-NAME ;
```

```
SQL> SELECT INITCAP('qSPidEr') FROM DUAL ;

INITCAP
-------
Qspider


SQL> SELECT INITCAP(ENAME) FROM EMP ;

INITCAP(EN
----------
Smith
Allen
Ward
Jones
Martin
Blake
Clark
Scott
King
Turner
Adams
James
Ford
Miller

14 rows selected.
```

LENGTH FUNCTION

This function is used to calculate the number of characters present in literals(string, number, date).

Syntax:

SELECT LENGTH(COLNAME / STRING) FROM TABLE-NAME ;

```
SQL> SELECT LENGTH('QSPIDERS') FROM DUAL;

LENGTH('QSPIDERS')
------------------
                 8

SQL> SELECT LENGTH(ENAME) FROM EMP ;

LENGTH(ENAME)
-------------
            5
            5
            4
            5
            6
            5
            5
            5
            4
            6
            5
            5
            4
            6

14 rows selected.
```

SELECT * FROM TABLE-NAME

WHERE LENGTH (COLNAME) = VALUE ;

```
SQL> SELECT * FROM EMP WHERE LENGTH(ENAME) = 5 ;

     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
      7900 JAMES      CLERK           7698 03-DEC-81        950                    30

8 rows selected.
```

SUBSTR FUNCTION

This function is used to extract string from another string.

Extraction from string happens in forward direction.

Syntax:

```
SELECT SUBSTR( arg1 , arg2 , agr3 ) FROM TABLE-NAME ;
```

arg1 - colname / string

arg2 - starting position

arg3 - length of substring

-8	-7	-6	-5	-4	-3	-2	-1	
Q	**S**	**P**	**I**	**D**	**E**	**R**	**S**	**(L ---→ R)**
1	2	3	4	5	6	7	8	

```
SQL> SELECT SUBSTR('QSPIDERS' , 2 , 4) FROM DUAL ;

SUBS
----
SPID

SQL> SELECT SUBSTR('QSPIDERS' , 2 ) FROM DUAL ;

SUBSTR(
-------
SPIDERS

SQL> SELECT SUBSTR('QSPIDERS' , -4 , 3) FROM DUAL ;

SUB
---
DER

SQL> SELECT SUBSTR('QSPIDERS' , -7) FROM DUAL ;

SUBSTR(
-------
SPIDERS
```

QUERIES

a) Query to display 1st character and last character of each employee name.

```
SQL> SELECT ENAME , SUBSTR(ENAME,1,1) , SUBSTR(ENAME,-1) FROM EMP ;

ENAME      S S
---------- - -
SMITH      S H
ALLEN      A N
WARD       W D
JONES      J S
MARTIN     M N
BLAKE      B E
CLARK      C K
SCOTT      S T
KING       K G
TURNER     T R
ADAMS      A S
JAMES      J S
FORD       F D
MILLER     M R

14 rows selected.
```

b) Query to display starting 50% character of each employee name.

```
SQL> SELECT ENAME , SUBSTR(ENAME , 1 , LENGTH(ENAME)/2 ) FROM EMP ;

ENAME      SUBSTR(ENA
---------- ----------
SMITH      SM
ALLEN      AL
WARD       WA
JONES      JO
MARTIN     MAR
BLAKE      BL
CLARK      CL
SCOTT      SC
KING       KI
TURNER     TUR
ADAMS      AD
JAMES      JA
FORD       FO
MILLER     MIL

14 rows selected.
```

c) Query to display last 50% character of each employee name.

```
SQL> SELECT ENAME , SUBSTR(ENAME , LENGTH(ENAME)/2 +1 ) FROM EMP ;

ENAME      SUBSTR(ENA
---------- ----------
SMITH      ITH
ALLEN      LEN
WARD       RD
JONES      NES
MARTIN     TIN
BLAKE      AKE
CLARK      ARK
SCOTT      OTT
KING       NG
TURNER     NER
ADAMS      AMS
JAMES      MES
FORD       RD
MILLER     LER

14 rows selected.
```

d) Query to display 1st character of employee in lowercase and remaining in uppercase.

```
SQL> SELECT ENAME , LOWER(SUBSTR(ENAME,1,1)) || UPPER(SUBSTR(ENAME,2)) FROM EMP ;

ENAME      LOWER(SUBS
---------- ----------
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

14 rows selected.
```

e) Query to display details of employee whose name start with s without using like operator.

```
SQL> SELECT * FROM EMP WHERE SUBSTR(ENAME,1,1) = 'S' ;

     EMPNO ENAME      JOB              MGR HIREDATE         SAL       COMM     DEPTNO
---------- ---------- --------- ---------- --------- ---------- ---------- ----------
      7369 SMITH      CLERK           7902 17-DEC-80        800                    20
      7788 SCOTT      ANALYST         7566 19-APR-87       3000                    20
```

f) Query to display details of employee whose name end with s .

```
SQL> SELECT * FROM EMP WHERE SUBSTR(ENAME,-1) = 'S' ;

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

REPLACE FUNCTION

It is used to replace old data with new data.

Syntax:

```
SELECT REPLACE( arg1 , arg2 , agr3 ) FROM TABLE-NAME ;
```

arg1 - colname / string

arg2 - old data

arg3 - new data

```
SQL> SELECT REPLACE('QSPIDERS' , 'Q' , 'J') FROM DUAL ;

REPLACE(
--------
JSPIDERS

SQL> SELECT REPLACE('QSPIDERS' , 'Q') FROM DUAL ;

REPLACE
-------
SPIDERS

SQL> SELECT REPLACE('QSPIDERS' , 'J') FROM DUAL ;

REPLACE(
--------
QSPIDERS
```

QUERIES

a) Query to display details of employee whose designation contains only one alphabet s.

```
SQL> SELECT * FROM EMP WHERE
  2  LENGTH(JOB) - LENGTH(REPLACE(JOB,'S')) = 1 ;

     EMPNO ENAME      JOB              MGR HIREDATE         SAL       COMM     DEPTNO
---------- ---------- --------- ---------- --------- ---------- ---------- ----------
      7788 SCOTT      ANALYST         7566 19-APR-87       3000                    20
      7839 KING       PRESIDENT            17-NOV-81       5000                    10
      7902 FORD       ANALYST         7566 03-DEC-81       3000                    20
```

b) Query to display number of alphabet s present in each designation.

```
SQL> SELECT JOB , LENGTH(JOB) - LENGTH(REPLACE(JOB,'S')) FROM EMP ;

JOB       LENGTH(JOB)-LENGTH(REPLACE(JOB,'S'))
--------- ------------------------------------
CLERK                                        0
SALESMAN                                     2
SALESMAN                                     2
MANAGER                                      0
SALESMAN                                     2
MANAGER                                      0
MANAGER                                      0
ANALYST                                      1
PRESIDENT                                    1
SALESMAN                                     2
CLERK                                        0
CLERK                                        0
ANALYST                                      1
CLERK                                        0

14 rows selected.
```

c) Query to display details of employee whose designation contains alphabet s.

```
SQL> SELECT * FROM EMP WHERE
  2  LENGTH(JOB) - LENGTH(REPLACE(JOB,'S')) != 0 ;

     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
      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

7 rows selected.
```

d) Query to display details of employee whose designation contains exactly two time alphabet s.

```
SQL> SELECT * FROM EMP WHERE
  2  LENGTH(JOB) - LENGTH(REPLACE(JOB,'S')) = 2 ;

     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
```

INSTR FUNCTION

instring function is used to find position of the character based on occurrence.

By default, it will find first time present character position if we are not using occurrence.

Syntax:

```
SELECT INSTR ( arg1 , arg2 , agr3 , arg4 ) FROM TABLE-NAME ;
```

arg1 - colname / string

arg2 - character to search

arg3 - length of substring

arg3 - occurance

```
SQL> SELECT INSTR('QSPIDERS' , 'SP' , 2 , 1) FROM DUAL ;

INSTR('QSPIDERS','SP',2,1)
--------------------------
                         2

SQL> SELECT INSTR('QSPIDERS' , 'S' , 1) FROM DUAL ;

INSTR('QSPIDERS','S',1)
-----------------------
                      2

SQL> SELECT INSTR('QSPIDERS' , 'S' , 1 , 2) FROM DUAL ;

INSTR('QSPIDERS','S',1,2)
-------------------------
                        8
```

ASCII FUNCTION (AMERICAN STANDARD CODE FOR INFORMATION INTERCHANGE)

```
SQL> SELECT ASCII('A') FROM DUAL ;

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

SQL> SELECT ASCII('i') FROM DUAL ;

ASCII('I')
----------
       105

SQL> SELECT ASCII('5') FROM DUAL ;

ASCII('5')
----------
        53
```

NVL

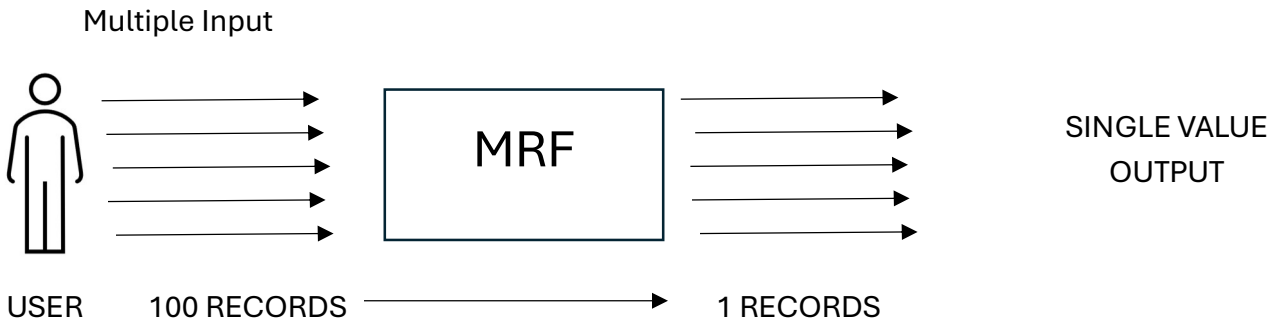
This function is used to replace null value with another value.

```
SQL> SELECT ENAME , SAL , COMM , SAL+NVL(COMM,0) FROM EMP ;

ENAME             SAL       COMM SAL+NVL(COMM,0)
---------- ---------- ---------- ---------------
SMITH             800                       800
ALLEN            1600        300            1900
WARD             1250        500            1750
JONES            2975                      2975
MARTIN           1250       1400            2650
BLAKE            2850                      2850
CLARK            2450                      2450
SCOTT            3000                      3000
KING             5000                      5000
TURNER           1500          0            1500
ADAMS            1100                      1100
JAMES             950                       950
FORD             3000                      3000
MILLER           1300                      1300

14 rows selected.
```

MULTIROW FUNCTIONS



multirow functions are also known as aggregate functions or grouping functions.

All multirow functions will take multiple input from user and it will execute single value output.

Multirow functions can be used inside select statement but we cant use inside the where clause.

Combination of column and multirow functions is not possible.

We will get single group function error and to overcome from that error we have to use group by clause.

Types of Multirow functions:

MAX FUNCTION

This function is used to find highest value from specified column.

Syntax:

```
SELECT MAX( COLNAME ) FROM TABLE-NAME ;
```

```
SQL> SELECT MAX(SAL) FROM EMP ;

  MAX(SAL)
----------
      5000

SQL> SELECT MAX(SAL) FROM EMP WHERE JOB='SALESMAN' ;

  MAX(SAL)
----------
      1600

SQL> SELECT MAX(HIREDATE) FROM EMP;

MAX(HIRED
---------
23-MAY-87
```

MIN FUNCTION

This function is used to find the lowest value from specified column.

Syntax:

```
SELECT MIN( COLNAME ) FROM TABLE-NAME ;
```

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

  MIN(SAL)
----------
       800

SQL> SELECT MIN(SAL) FROM EMP WHERE JOB = 'ANALYST' ;

  MIN(SAL)
----------
      3000
SQL> SELECT MIN(HIREDATE) FROM EMP ;

MIN(HIRED
---------
17-DEC-80
```

AVG FUNCTION

This function is used to calculate the average of multiple values.

Syntax:

```
SELECT AVG( COLNAME ) FROM TABLE-NAME ;
```

```
SQL> SELECT AVG(SAL) FROM EMP ;

  AVG(SAL)
----------
2073.21429

SQL> SELECT AVG(COMM) FROM EMP ;

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

SUM FUNCTION

This function is used to calculate the sum of multiple numeric values.

Syntax:

```
SELECT SUM( COLNAME ) FROM TABLE-NAME ;
```

```
SQL> SELECT SUM(SAL) FROM EMP ;

  SUM(SAL)
----------
     29025

SQL> SELECT SUM(COMM) FROM EMP ;

 SUM(COMM)
----------
      2200
```

COUNT FUNCTION

This function is used to calculate the number of records present in specified columns.

It will ignore null values.

Syntax:

```
SELECT COUNT( COLNAME ) FROM TABLE-NAME ;
```

```
SQL> SELECT COUNT(ENAME) FROM EMP;

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

SQL> SELECT COUNT(DEPTNO) FROM DEPT ;

COUNT(DEPTNO)
-------------
           4
```

GROUP BY CLAUSE

Syntax:

SELECTCOLNAME , MULTIROW-FUNCTION FROM TABLE-NAME

WHERE CLAUSE

GROUP BY CLAUSE ;

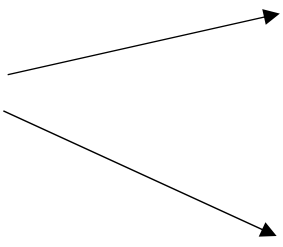
Why we have to use group by clause ?

Combination of column and multirow functions are not possible , we will get single group function error and to overcome from that error we have to use group by clause.

Whatever column we are using inside the select statement along with multirow functions , the same column we have to use inside the group by clause.

what is group by clause ?

ENAME	JOB	SAL
A	CLERK	800
B	SALESMAN	2000
C	SALESMAN	900
D	CLERK	1000



GROUP1 **max = 1000**

ENAME	JOB	SAL
A	CLERK	800
D	CLERK	1000

ENAME	JOB	SAL
B	SALESMAN	2000
C	SALESMAN	900

GROUP2 **max = 2000**

Group by is used to divide the the table into multiple groups and it will execute each output for each group.

Group by clause will create groups based on unique values.

Where clause is not mandatory to execute group by clause.

QUERIES

a) Query to display highest salary given to employee in each designation.

```
SQL> SELECT JOB , MAX(SAL) FROM EMP
  2  GROUP BY JOB ;

JOB         MAX(SAL)
--------- ----------
CLERK           1300
SALESMAN        1600
PRESIDENT       5000
MANAGER         2975
ANALYST         3000
```

b) Query to display highest salary given to employee in each designation apart from salesman.

```
SQL> SELECT JOB , MAX(SAL) FROM EMP
  2  WHERE JOB != 'SALESMAN'
  3  GROUP BY JOB ;

JOB         MAX(SAL)
--------- ----------
CLERK           1300
PRESIDENT       5000
MANAGER         2975
ANALYST         3000
```

HAVING CLAUSE

SELECT COLUMN , MULTIROW-FUNCTION FROM TABLE-NAME

WHERE CLAUSE

GROUP BY CLAUSE

HAVING CLAUSE ;

Why we need to use HAVING clause ?

Multirow function cant be used in where clause , if there is an condition for multirow function we have to use having clause.

What is HAVING clause ?

Having clause is used to filter the groups by giving conditions to multirow functions.

Having clause executes group by clause.

Group by clause is mandatory to use having clause.

Where clause is not mandatory to execute having clause.

QUERIES

a) Query to display highest salary given to employees in each designation , display the output if the highest salary is more than 3000.

```
SQL> SELECT JOB , MAX(SAL) FROM EMP
  2  GROUP BY JOB
  3  HAVING MAX(SAL) > 3000 ;

JOB         MAX(SAL)
--------- ----------
PRESIDENT       5000
```

b) Query to display employees salary that are repeated.

```
SQL> SELECT SAL FROM EMP
  2  GROUP BY SAL
  3  HAVING COUNT(*) > 1 ;

       SAL
----------
      1250
      3000
```

c) Query to display employee designation that are repeated .

```
SQL> SELECT JOB FROM EMP
  2  GROUP BY JOB
  3  HAVING COUNT(*) > 1 ;

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

d) Query to display employees designation that are repeated exactly twice.

```
SQL> SELECT JOB FROM EMP
  2  GROUP BY JOB
  3  HAVING COUNT(*) = 2 ;

JOB
---------
ANALYST
```

DATE FUNCTION

SYSDATE

Syntax:

```
SELECT SYSDATE FROM DUAL;
```

```
SELECT CURRENT_DATE FROM DUAL;
```

```
SQL> SELECT SYSDATE FROM DUAL ;

SYSDATE
---------
06-MAY-25

SQL> SELECT CURRENT_DATE FROM DUAL ;

CURRENT_D
---------
06-MAY-25
```

SYSTIMESTAMP

Syntax:

```
SELECT SYSTIMESTAMP FROM DUAL;
```

```
SYSTIMESTAMP
---------------------------------------
06-MAY-25 09.18.20.988000 PM +05:30
```

TO DATE

This function is used to convert string value into date value.

```
SQL> SELECT TO_DATE('31-DEC-2025' , 'DD-MON-YYYY') - TO_DATE('31-DEC-2024' , 'DD-MON-YYYY') FROM DUAL ;

TO_DATE('31-DEC-2025','DD-MON-YYYY')-TO_DATE('31-DEC-2024','DD-MON-YYYY')
--------------------------------------------------------------------------
                                                                       365
```

MONTHS BETWEEN

```
SQL> SELECT MONTHS_BETWEEN('31-DEC-2025' , '31-DEC-2024') FROM DUAL ;

MONTHS_BETWEEN('31-DEC-2025','31-DEC-2024')
-------------------------------------------
                                         12
```

ADD_MONTHS

```
SQL> SELECT ADD_MONTHS(SYSDATE , 4 ) FROM DUAL ;

ADD_MONTH
---------
06-SEP-25
```

LAST DAY

```
SQL> SELECT LAST_DAY(SYSDATE) FROM DUAL ;

LAST_DAY(
---------
31-MAY-25
```

FIRST DAY

```
SQL> SELECT TRUNC(SYSDATE , 'MONTH' ) FROM DUAL ;

TRUNC(SYS
---------
01-MAY-25

SQL> SELECT TRUNC(SYSDATE , 'YEAR' ) FROM DUAL ;

TRUNC(SYS
---------
01-JAN-25
```

TO CHAR

this function is used to change date format.

```
SQL> SELECT TO_CHAR(SYSDATE , 'DD-MON-YY') FROM DUAL ;

TO_CHAR(S
---------
06-MAY-25

SQL> SELECT TO_CHAR(SYSDATE , 'DD-MON-YYYY') FROM DUAL ;

TO_CHAR(SYS
-----------
06-MAY-2025

SQL> SELECT TO_CHAR(SYSDATE , 'FMDAY') FROM DUAL ;

TO_CHAR(S
---------
TUESDAY

SQL> SELECT TO_CHAR(SYSDATE , 'DY' ) FROM DUAL ;

TO_
---
TUE
```

QUERIES

a) Query to display details of employee who joined in the year1981.

```
SQL> SELECT * FROM EMP WHERE TO_CHAR(HIREDATE , 'YYYY') = 1981 ;

     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

10 rows selected.
```

b) Query to display next day.

```
SQL> SELECT SYSDATE + 1 AS NEXT_DAY FROM DUAL ;

NEXT_DAY
---------
11-MAY-25
```

c) Query to display number of years of experience having each employee.

```
SQL> SELECT ENAME , TO_CHAR(SYSDATE , 'YYYY') - TO_CHAR(HIREDATE , 'YYYY') AS YOE FROM EMP ;

ENAME             YOE
---------- ----------
SMITH              45
ALLEN              44
WARD               44
JONES              44
MARTIN             44
BLAKE              44
CLARK              44
SCOTT              38
KING               44
TURNER             44
ADAMS              38
JAMES              44
FORD               44
MILLER             43

14 rows selected.
```

d) Query to display details of employee who are having 44+ years of experience.

```
SQL> SELECT EMP.* , TO_CHAR(SYSDATE , 'YYYY') - TO_CHAR(HIREDATE , 'YYYY') AS YOE FROM EMP
  2  WHERE TO_CHAR(SYSDATE , 'YYYY') - TO_CHAR(HIREDATE , 'YYYY') > 44 ;

     EMPNO ENAME      JOB              MGR HIREDATE         SAL       COMM     DEPTNO        YOE
---------- ---------- --------- ---------- --------- ---------- ---------- ---------- ----------
      7369 SMITH      CLERK           7902 17-DEC-80        800                    20         45
```

e) Query to display details of employee who joined on Friday.

```
SQL> SELECT EMP.* , TO_CHAR(HIREDATE , 'FMDAY' ) AS DAY FROM EMP
  2  WHERE TO_CHAR(HIREDATE , 'FMDAY') = 'FRIDAY' ;

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

f) Query to display details of employee who joined on Fri.

```
SQL> SELECT EMP.* , TO_CHAR(HIREDATE , 'DY' ) AS DAY FROM EMP
  2  WHERE TO_CHAR(HIREDATE , 'DY' )  = 'FRI' ;

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

WINDOW FUNCTION

Window functions are used to generate ranking to records based on value.

Over clause is mandatory.

TYPES OF WINDOW FUNCTION:

ROW NUMBER

It is used to generate unique values for each and every record.

```
SQL> SELECT ENAME , SAL , ROW_NUMBER()
  2  OVER(ORDER BY SAL DESC) AS RANK
  3  FROM EMP ;

ENAME              SAL       RANK
---------- ---------- ----------
KING             5000          1
FORD             3000          2
SCOTT            3000          3
JONES            2975          4
BLAKE            2850          5
CLARK            2450          6
ALLEN            1600          7
TURNER           1500          8
MILLER           1300          9
WARD             1250         10
MARTIN           1250         11
ADAMS            1100         12
JAMES             950         13
SMITH             800         14

14 rows selected.
```

RANK FUNCTION

It is used to generate ranking for records in non-sequential order.

```
SQL> SELECT ENAME , SAL , RANK()
  2  OVER(ORDER BY SAL DESC) AS RANK
  3  FROM EMP ;

ENAME              SAL       RANK
---------- ---------- ----------
KING             5000          1
FORD             3000          2
SCOTT            3000          2
JONES            2975          4
BLAKE            2850          5
CLARK            2450          6
ALLEN            1600          7
TURNER           1500          8
MILLER           1300          9
WARD             1250         10
MARTIN           1250         10
ADAMS            1100         12
JAMES             950         13
SMITH             800         14

14 rows selected.
```

DENSE RANK FUNCTION

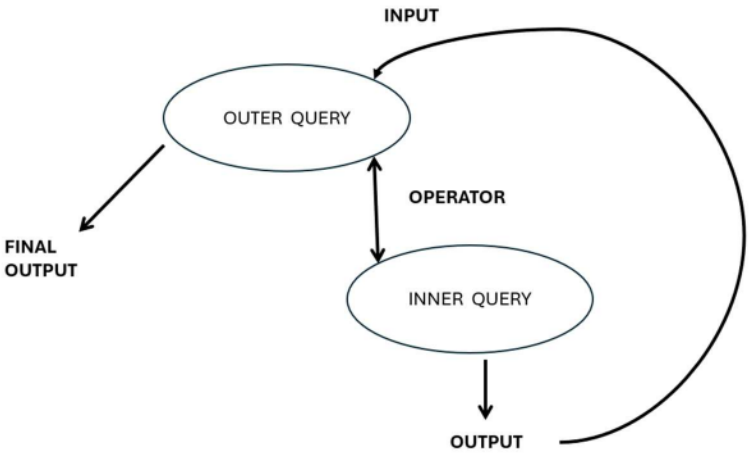
It is used to generate ranking for records in sequence order.

```
SQL> SELECT ENAME , SAL , DENSE_RANK()
  2  OVER(ORDER BY SAL DESC) AS RANK
  3  FROM EMP ;

ENAME                SAL       RANK
---------- ---------- ----------
KING               5000          1
FORD               3000          2
SCOTT              3000          2
JONES              2975          3
BLAKE              2850          4
CLARK              2450          5
ALLEN              1600          6
TURNER             1500          7
MILLER             1300          8
WARD               1250          9
MARTIN             1250          9
ADAMS              1100         10
JAMES               950         11
SMITH               800         12

14 rows selected.
```

SUBQUERY



Query inside another query is known as subquery.

Whenever we are having indirect conditions, we have to use subquery (unknown conditions).

In subquery , inner query is independent.

In subquery , outer query is dependent on output of inner query.

To join inner query and outer query we can make use of operators.

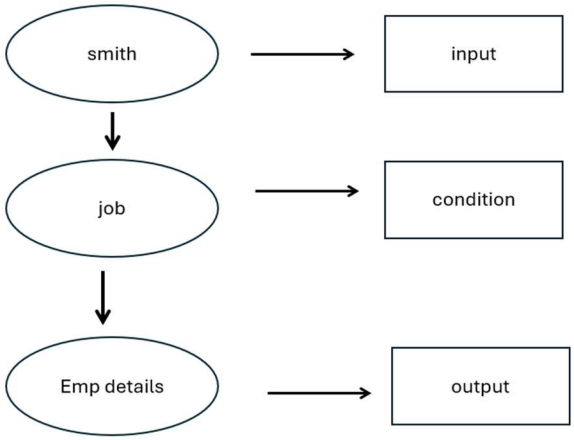
EXECUTION FLOW OF SUBQUERY

First, the inner query will get executed completely , then output of inner query will become input to the outer query and we will get final output from outer query.

Note:

Whatever column we are using inside the where clause of the outer query, the same column we have to use inside the select statement of the inner query.

- Query to display details of the employee who belongs from Smith's designation.



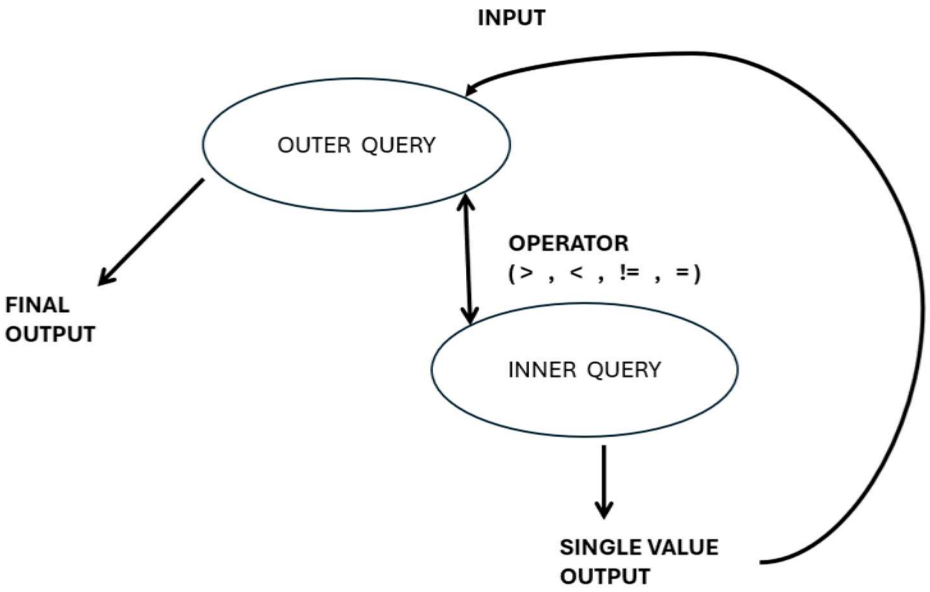
```
SQL> SELECT * FROM EMP WHERE JOB =
  2  (SELECT JOB FROM EMP WHERE ENAME = 'SMITH' ) ;

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

TYPES OF SUBQUERY

1. **NON-COREATED SUBQUERY**
    a. SINGLE ROW
    b. MULTIROW
    c. NESTED
2. **CO-RELATED SUBQUERY**

SINGLE ROW SUBQUERY



In single row subquery from inner query, we will get single value output.

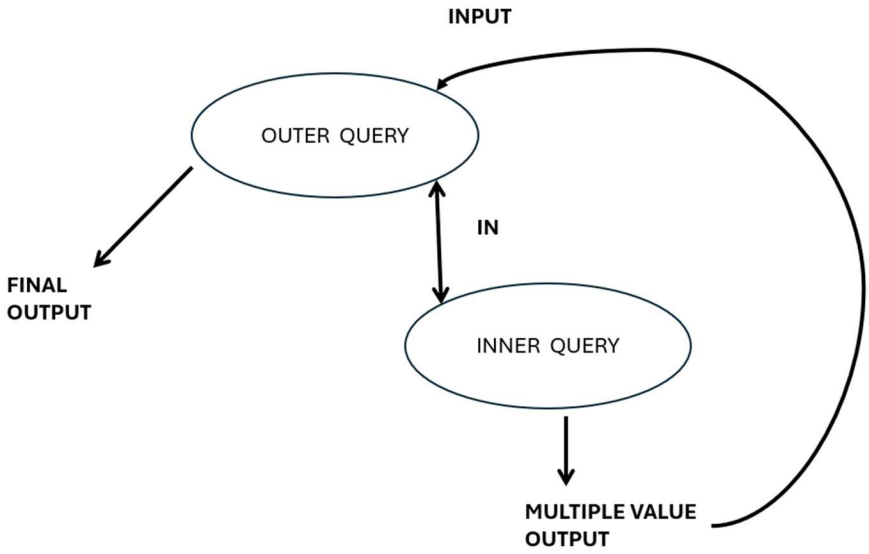
To join the inner query and the outer query, we have to use relational operator.

- Query to display 1st highest salary earning employee.

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

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

MULTI ROW SUBQUERY



In multirow subquery from inner query we will get multiple value output.

To join the inner query and the outer query, we have to use IN operator.

- Query to display details of employee who belongs from smith and martin designation.

```
SQL> SELECT * FROM EMP WHERE JOB IN
  2  (SELECT JOB FROM EMP WHERE ENAME IN('SMITH' , 'MARTIN')) ;

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

8 rows selected.
```

NESTED SUBQUERY

A Query inside another subquery is known as a nested subquery.

QUERIES

a) Query to display second highest salary earning employee.

```
SQL> SELECT * FROM EMP WHERE SAL =
  2  (SELECT MAX(SAL) FROM EMP WHERE SAL <
  3  (SELECT MAX(SAL) FROM EMP )) ;

     EMPNO ENAME      JOB              MGR HIREDATE         SAL       COMM     DEPTNO
---------- ---------- --------- ---------- --------- ---------- ---------- ----------
      7788 SCOTT      ANALYST         7566 19-APR-87       3000                    20
      7902 FORD       ANALYST         7566 03-DEC-81       3000                    20
```

b) Query to display lowest salary earning employee.

```
SQL> SELECT * FROM EMP WHERE SAL =
  2  (SELECT MIN(SAL) FROM EMP) ;

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

c) Query to display 1st highest salary earning employee in salesman designation.

```
SQL> SELECT * FROM EMP WHERE SAL =
  2  (SELECT MAX(SAL) FROM EMP WHERE JOB = 'SALESMAN' ) ;

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

d) Query to display 2nd lowest salary earning employee details.

```
SQL> SELECT * FROM EMP WHERE SAL =
  2  (SELECT MIN(SAL) FROM EMP WHERE SAL >
  3  (SELECT MIN(SAL) FROM EMP )) ;

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

e) Query to display details of employee who are earning more than average salary of employee working as salesman.

```
SQL> SELECT * FROM EMP WHERE SAL >
  2  (SELECT AVG(SAL) FROM EMP WHERE JOB='SALESMAN' ) ;

     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

8 rows selected.
```

f) Query to display details of employee from king department .

```
SQL> SELECT * FROM EMP WHERE DEPTNO =
  2  (SELECT DEPTNO FROM EMP WHERE ENAME = 'KING' );

     EMPNO ENAME      JOB              MGR HIREDATE         SAL       COMM     DEPTNO
---------- ---------- --------- ---------- --------- ---------- ---------- ----------
      7782 CLARK      MANAGER         7839 09-JUN-81       2450                    10
      7839 KING       PRESIDENT            17-NOV-81       5000                    10
      7934 MILLER     CLERK           7782 23-JAN-82       1300                    10
```

g) Query to display department name for smith.

```
SQL> SELECT DNAME FROM DEPT WHERE DEPTNO =
  2  (SELECT DEPTNO FROM EMP WHERE ENAME = 'SMITH' ) ;

DNAME
--------------
RESEARCH
```

h) Query to display employee info for employee who belongs from research department.

```
SQL> SELECT * FROM EMP WHERE DEPTNO =
  2  (SELECT DEPTNO FROM DEPT WHERE DNAME = 'RESEARCH' ) ;

     EMPNO ENAME      JOB              MGR HIREDATE         SAL       COMM     DEPTNO
---------- ---------- --------- ---------- --------- ---------- ---------- ----------
      7369 SMITH      CLERK           7902 17-DEC-80        800                    20
      7566 JONES      MANAGER         7839 02-APR-81       2975                    20
      7788 SCOTT      ANALYST         7566 19-APR-87       3000                    20
      7876 ADAMS      CLERK           7788 23-MAY-87       1100                    20
      7902 FORD       ANALYST         7566 03-DEC-81       3000                    20
```

i) Query to display details of employee who belongs from research and sales department.

```
SQL>  SELECT * FROM EMP WHERE DEPTNO IN
  2   (SELECT DEPTNO FROM DEPT  WHERE DNAME IN('RESEARCH' , 'SALES' )) ;

     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
      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
      7902 FORD       ANALYST         7566 03-DEC-81       3000                    20

11 rows selected.
```

j) Query to display senior most employee joined in 1981.

```
SQL> SELECT * FROM EMP WHERE HIREDATE =
  2  (SELECT MIN(HIREDATE) FROM EMP WHERE TO_CHAR(HIREDATE , 'YYYY') = 1981 ) ;

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

k) Query to display department name for senior most employee joined in 1981.

```
SQL> SELECT DNAME FROM DEPT WHERE DEPTNO =
  2  (SELECT DEPTNO FROM EMP WHERE HIREDATE =
  3  (SELECT MIN(HIREDATE) FROM EMP WHERE TO_CHAR(HIREDATE , 'YYYY') = 1981 )) ;

DNAME
--------------
SALES
```

l) Query to display department name and location for employees who belongs from Smith's designation.

```
SQL> SELECT DNAME , LOC FROM DEPT WHERE DEPTNO IN
  2  (SELECT DEPTNO FROM EMP WHERE JOB =
  3  (SELECT JOB FROM EMP WHERE ENAME = 'SMITH' )) ;

DNAME          LOC
-------------- -------------
SALES          CHICAGO
RESEARCH       DALLAS
ACCOUNTING     NEW YORK
```

m) Query to display department name for employee who are earning more than Martin's salary.

```
SQL> SELECT DNAME FROM DEPT WHERE DEPTNO IN
  2  (SELECT DEPTNO FROM EMP WHERE SAL >
  3  (SELECT SAL FROM EMP WHERE ENAME = 'MARTIN' )) ;

DNAME
--------------
SALES
RESEARCH
ACCOUNTING
```

n) Query to display department name in which employees are working.

```
SQL> SELECT DNAME FROM DEPT WHERE DEPTNO IN
  2  (SELECT DEPTNO FROM EMP) ;

DNAME
--------------
ACCOUNTING
RESEARCH
SALES
```

o) Query to display department name in which employees are not working.

```
SQL> SELECT DNAME FROM DEPT WHERE DEPTNO NOT IN
  2  (SELECT DEPTNO FROM EMP ) ;

DNAME
--------------
OPERATIONS
```

p) Query to display details of employee who joined after king.

```
SQL> SELECT * FROM EMP WHERE HIREDATE >
  2  (SELECT HIREDATE FROM EMP WHERE ENAME = 'KING' ) ;

     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
      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
```

q) Query to display details of employee who joined before king.

```
SQL> SELECT * FROM EMP WHERE HIREDATE <
  2  (SELECT HIREDATE FROM EMP WHERE ENAME = 'KING' ) ;

     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

8 rows selected.
```

r) Query to display the seniormost employee joined before king.

```
SQL> SELECT * FROM EMP WHERE HIREDATE =
  2  (SELECT MIN(HIREDATE) FROM EMP WHERE HIREDATE <
  3  (SELECT HIREDATE FROM EMP WHERE ENAME = 'KING' )) ;

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

s) Query to display department name for seniormost employee joined before king.

```
SQL> SELECT DNAME FROM DEPT WHERE DEPTNO =
  2  (SELECT DEPTNO FROM EMP WHERE HIREDATE =
  3  (SELECT MIN(HIREDATE) FROM EMP WHERE HIREDATE <
  4  (SELECT HIREDATE FROM EMP WHERE ENAME = 'KING' ))) ;

DNAME
--------------
RESEARCH
```

t) Query to display longest employee name details.

```
SQL> SELECT * FROM EMP WHERE LENGTH(ENAME) =
  2  (SELECT MAX(LENGTH(ENAME)) FROM EMP ) ;

     EMPNO ENAME      JOB              MGR HIREDATE         SAL       COMM     DEPTNO
---------- ---------- --------- ---------- --------- ---------- ---------- ----------
      7654 MARTIN     SALESMAN        7698 28-SEP-81       1250       1400         30
      7844 TURNER     SALESMAN        7698 08-SEP-81       1500          0         30
      7934 MILLER     CLERK           7782 23-JAN-82       1300                    10
```

u) Query to display details of employee who are earning similar salary.

```
SQL> SELECT * FROM EMP WHERE SAL IN
  2  (SELECT SAL FROM EMP
  3  GROUP BY SAL
  4  HAVING COUNT(*) > 1 );

     EMPNO ENAME      JOB              MGR HIREDATE         SAL       COMM     DEPTNO
---------- ---------- --------- ---------- --------- ---------- ---------- ----------
      7654 MARTIN     SALESMAN        7698 28-SEP-81       1250       1400         30
      7521 WARD       SALESMAN        7698 22-FEB-81       1250        500         30
      7902 FORD       ANALYST         7566 03-DEC-81       3000                    20
      7788 SCOTT      ANALYST         7566 19-APR-87       3000                    20
```

v) Query to display details of employee who belongs from same designation.

```
SQL> SELECT * FROM EMP WHERE JOB IN
  2  (SELECT JOB FROM EMP
  3  GROUP BY JOB
  4  HAVING COUNT(*) > 1 ) ;

     EMPNO ENAME      JOB              MGR HIREDATE         SAL       COMM     DEPTNO
---------- ---------- --------- ---------- --------- ---------- ---------- ----------
      7934 MILLER     CLERK           7782 23-JAN-82       1300                    10
      7900 JAMES      CLERK           7698 03-DEC-81        950                    30
      7876 ADAMS      CLERK           7788 23-MAY-87       1100                    20
      7369 SMITH      CLERK           7902 17-DEC-80        800                    20
      7844 TURNER     SALESMAN        7698 08-SEP-81       1500          0         30
      7654 MARTIN     SALESMAN        7698 28-SEP-81       1250       1400         30
      7521 WARD       SALESMAN        7698 22-FEB-81       1250        500         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
      7902 FORD       ANALYST         7566 03-DEC-81       3000                    20
      7788 SCOTT      ANALYST         7566 19-APR-87       3000                    20

13 rows selected.
```

w) Query to display department name for employee who belongs from same designation.

```
SQL> SELECT DNAME FROM DEPT WHERE DEPTNO IN
  2  (SELECT DEPTNO FROM EMP WHERE JOB IN
  3  (SELECT JOB FROM EMP
  4  GROUP BY JOB
  5  HAVING COUNT(*) > 1 )) ;

DNAME
--------------
SALES
RESEARCH
ACCOUNTING
```

x) Query to display details of employee who joined on same date.

```
SQL> SELECT * FROM EMP WHERE HIREDATE IN
  2  (SELECT HIREDATE FROM EMP
  3  GROUP BY HIREDATE
  4  HAVING COUNT(*) > 1) ;

     EMPNO ENAME      JOB              MGR HIREDATE         SAL       COMM     DEPTNO
---------- ---------- --------- ---------- --------- ---------- ---------- ----------
      7902 FORD       ANALYST         7566 03-DEC-81       3000                    20
      7900 JAMES      CLERK           7698 03-DEC-81        950                    30
```

y) Query to display department name and location for employee who joined on same date.

```
SQL> SELECT DNAME , LOC FROM DEPT WHERE DEPTNO IN
  2  (SELECT DEPTNO FROM EMP WHERE HIREDATE IN
  3  (SELECT HIREDATE FROM EMP
  4  GROUP BY HIREDATE
  5  HAVING COUNT(*) > 1 )) ;

DNAME          LOC
-------------- -------------
SALES          CHICAGO
RESEARCH       DALLAS
```

z) Query to display department name for employee who belongs from analyst designation.

```
SQL> SELECT DNAME FROM DEPT WHERE DEPTNO IN
  2  (SELECT DEPTNO FROM EMP WHERE JOB = 'ANALYST' ) ;

DNAME
--------------
RESEARCH
```

aa) Query to display department name and location for employee whose designation contains only 1 alphabet s.

```
SQL> SELECT DNAME , LOC FROM DEPT WHERE DEPTNO IN
  2  (SELECT DEPTNO FROM EMP WHERE LENGTH(JOB) - LENGTH(REPLACE(JOB , 'S' )) = 1 ) ;

DNAME          LOC
-------------- -------------
ACCOUNTING     NEW YORK
RESEARCH       DALLAS
```

QUERIES ON EMPLOYEE AND THEIR MANAGER RELATION

Note:

Whenever we are having employee and their manager relation for employee we have to find their manager number and for their manager we have to find their employee number.

OUERIES

a) Query to display details of employee who are working under king.

```
SQL> SELECT * FROM EMP WHERE MGR =
  2  (SELECT EMPNO FROM EMP WHERE ENAME = 'KING' ) ;

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

b) Query to display details of employee who are working under king and blake.

```
SQL> SELECT * FROM EMP WHERE MGR IN
  2  (SELECT EMPNO FROM EMP WHERE ENAME IN('KING' , '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
      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
      7900 JAMES      CLERK            7698 03-DEC-81        950                    30

8 rows selected.
```

c) Query to display Smith's reporting manager information.

```
SQL> SELECT * FROM EMP WHERE EMPNO =
  2  (SELECT MGR FROM EMP WHERE ENAME = 'SMITH' ) ;

     EMPNO ENAME      JOB              MGR HIREDATE         SAL       COMM     DEPTNO
---------- ---------- --------- ---------- --------- ---------- ---------- ----------
      7902 FORD       ANALYST          7566 03-DEC-81       3000                    20
```

d) Query to display Smith's manager's manager details.

```
SQL> SELECT * FROM EMP WHERE EMPNO =
  2  (SELECT MGR FROM EMP WHERE EMPNO =
  3  (SELECT MGR FROM EMP WHERE ENAME = 'SMITH')) ;

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

e) Query to display department name and location for employee who are working under king.

```
SQL> SELECT DNAME , LOC FROM DEPT  WHERE DEPTNO IN
  2  (SELECT DEPTNO FROM EMP WHERE MGR =
  3  (SELECT EMPNO FROM EMP WHERE ENAME = 'KING' )) ;

DNAME          LOC
-------------- -------------
SALES          CHICAGO
RESEARCH       DALLAS
ACCOUNTING     NEW YORK
```

f) Query to display department name for Smith's reporting manager.

```
SQL> SELECT DNAME FROM DEPT WHERE DEPTNO IN
  2  (SELECT DEPTNO FROM EMP WHERE EMPNO =
  3  (SELECT MGR FROM EMP WHERE ENAME = 'SMITH' )) ;

DNAME
--------------
RESEARCH
```

g) Query to display manager details who are having minimum 5 employees working under them.

```
SQL> SELECT * FROM EMP WHERE EMPNO IN
  2  (SELECT MGR FROM EMP
  3  GROUP BY MGR
  4  HAVING COUNT(*) >= 5 ) ;

     EMPNO ENAME      JOB              MGR HIREDATE         SAL       COMM     DEPTNO
---------- ---------- --------- ---------- --------- ---------- ---------- ----------
      7698 BLAKE      MANAGER         7839 01-MAY-81       2850                    30
```

h) Query to display number of employee working under blake.

```
SQL> SELECT COUNT(*) AS NO_OF_EMP FROM EMP WHERE MGR =
  2  (SELECT EMPNO FROM EMP WHERE ENAME = 'BLAKE' ) ;

 NO_OF_EMP
----------
         5
```

i) Query to display employee details who are having employees working under them.(employees who are manager)

```
SQL> SELECT * FROM EMP WHERE EMPNO IN
  2  (SELECT MGR FROM EMP WHERE MGR IS NOT NULL);

     EMPNO ENAME      JOB              MGR HIREDATE         SAL       COMM     DEPTNO
---------- ---------- --------- ---------- --------- ---------- ---------- ----------
      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
      7902 FORD       ANALYST         7566 03-DEC-81       3000                    20

6 rows selected.
```

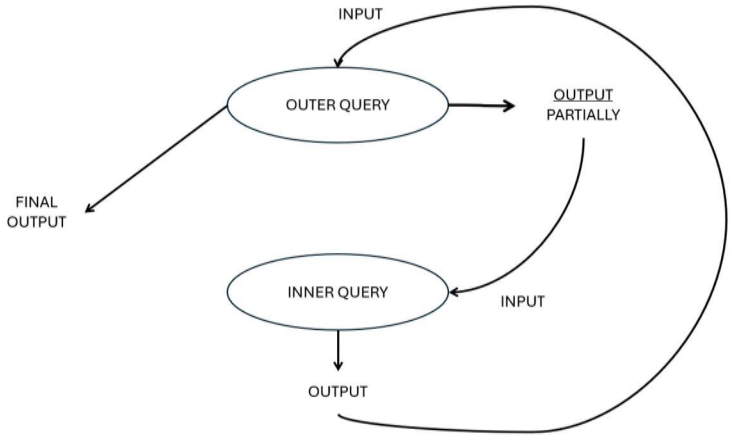
j) Query to display employee details who are not managers(employees who don't have employees working under them).

```
SQL> SELECT * FROM EMP WHERE EMPNO NOT IN
  2  (SELECT MGR FROM EMP WHERE MGR IS NOT NULL);

     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
      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

8 rows selected.
```

CO-RELATED SUBQUERY



in co-related subquery outer query and inner query both are dependent on each other.

EXECUTION FLOW OF CO-RELATED SUBQUERY

In correlated subquery, first outer query will get executed partially, the output of the outer query will become input to the inner query and it will get executed completely. The output of the inner query will become input to the outer query and it will get final output from outer query.

To achieve corelated subquery, we can make use of RowNum column.

RowNum is a pseudo column in Oracle SQL.

Co-related subquery used for row-by-row processing (row-by-row execution)

In co-related subquery inner query will get executed for every row of the outer query.

QUERIES

a) Query to display 3rd highest salary earning employee details.

```
SQL> SELECT * FROM
  2  (SELECT EMP.* , DENSE_RANK() OVER(ORDER BY SAL DESC) AS HSAL FROM EMP )
  3  WHERE HSAL = 3 ;

     EMPNO ENAME      JOB              MGR HIREDATE         SAL       COMM     DEPTNO       HSAL
---------- ---------- --------- ---------- --------- ---------- ---------- ---------- ----------
      7566 JONES      MANAGER         7839 02-APR-81       2975                    20          3
```

b) Query to display the starting 3 highest salaries from employee table.

```
SQL> SELECT SAL FROM
  2  (SELECT EMP.* , DENSE_RANK() OVER(ORDER BY SAL DESC) AS HSAL FROM EMP )
  3  WHERE HSAL <= 3 ;

       SAL
----------
      5000
      3000
      3000
      2975
```

c) Query to display starting 3 highest salary earning employee details.

```
SQL>  SELECT * FROM
  2   (SELECT EMP.* , DENSE_RANK() OVER(ORDER BY SAL DESC) AS HSAL FROM EMP )
  3   WHERE HSAL <= 3 ;

     EMPNO ENAME      JOB              MGR HIREDATE         SAL       COMM     DEPTNO       HSAL
---------- ---------- --------- ---------- --------- ---------- ---------- ---------- ----------
      7839 KING       PRESIDENT            17-NOV-81       5000                    10          1
      7788 SCOTT      ANALYST         7566 19-APR-87       3000                    20          2
      7902 FORD       ANALYST         7566 03-DEC-81       3000                    20          2
      7566 JONES      MANAGER         7839 02-APR-81       2975                    20          3
```

d) Query to display 1st lowest salary earning employee details.

```
SQL> SELECT * FROM
  2  (SELECT EMP.* , DENSE_RANK() OVER(ORDER BY SAL ASC) AS LSAL FROM EMP )
  3  WHERE LSAL = 1 ;

     EMPNO ENAME      JOB              MGR HIREDATE         SAL       COMM     DEPTNO       LSAL
---------- ---------- --------- ---------- --------- ---------- ---------- ---------- ----------
      7369 SMITH      CLERK           7902 17-DEC-80        800                    20          1
```

e) Query to display starting 2 lowest salary earning employee details.

```
SQL>  SELECT * FROM
  2   (SELECT EMP.* , DENSE_RANK() OVER(ORDER BY SAL ASC) AS LSAL FROM EMP )
  3   WHERE LSAL <= 2 ;

     EMPNO ENAME      JOB              MGR HIREDATE         SAL       COMM     DEPTNO       LSAL
---------- ---------- --------- ---------- --------- ---------- ---------- ---------- ----------
      7369 SMITH      CLERK           7902 17-DEC-80        800                    20          1
      7900 JAMES      CLERK           7698 03-DEC-81        950                    30          2
```

f) Query to display 3rd highest salary and 4th lowest salary earning employee details.

```
SQL> SELECT * FROM
  2  (SELECT EMP.* ,
  3  DENSE_RANK() OVER(ORDER BY SAL DESC) AS HSAL ,
  4  DENSE_RANK() OVER(ORDER BY SAL ASC) AS LSAL
  5  FROM EMP )
  6  WHERE HSAL = 3 OR LSAL = 4 ;

     EMPNO ENAME      JOB              MGR HIREDATE         SAL       COMM     DEPTNO       HSAL       LSAL
---------- ---------- --------- ---------- --------- ---------- ---------- ---------- ---------- ----------
      7654 MARTIN     SALESMAN        7698 28-SEP-81       1250       1400         30          9          4
      7521 WARD       SALESMAN        7698 22-FEB-81       1250        500         30          9          4
      7566 JONES      MANAGER         7839 02-APR-81       2975                    20          3         10
```

g) Query to display department name and location for employee who are earning 1st highest salary.

```
SQL> SELECT DNAME , LOC FROM DEPT WHERE DEPTNO IN
  2  (SELECT DEPTNO FROM
  3  (SELECT EMP.* , DENSE_RANK() OVER(ORDER BY SAL DESC) AS HSAL FROM EMP )
  4  WHERE HSAL = 1 ) ;

DNAME          LOC
-------------- -------------
ACCOUNTING     NEW YORK
```

h) Query to display 1st highest salary earning employee details in salesman designation.

```
SQL> SELECT * FROM
  2  (SELECT EMP.* , DENSE_RANK() OVER(ORDER BY SAL DESC) AS HSAL FROM EMP WHERE JOB = 'SALESMAN' )
  3  WHERE HSAL = 1 ;

     EMPNO ENAME      JOB              MGR HIREDATE         SAL       COMM     DEPTNO       HSAL
---------- ---------- --------- ---------- --------- ---------- ---------- ---------- ----------
      7499 ALLEN      SALESMAN        7698 20-FEB-81       1600        300         30          1
```

i) Query to display 1st record from employee table.

```
SQL> SELECT * FROM
  2  (SELECT EMP.* , ROWNUM JSPIDER FROM EMP )
  3  WHERE JSPIDER = 1 ;

     EMPNO ENAME      JOB              MGR HIREDATE         SAL       COMM     DEPTNO    JSPIDER
---------- ---------- --------- ---------- --------- ---------- ---------- ---------- ----------
      7369 SMITH      CLERK           7902 17-DEC-80        800                    20          1
```

j) Query to display starting 3 records from employee table.

```
SQL>  SELECT * FROM
  2   (SELECT EMP.* , ROWNUM JSPIDER FROM EMP )
  3   WHERE JSPIDER <= 3 ;

     EMPNO ENAME      JOB              MGR HIREDATE         SAL       COMM     DEPTNO    JSPIDER
---------- ---------- --------- ---------- --------- ---------- ---------- ---------- ----------
      7369 SMITH      CLERK           7902 17-DEC-80        800                    20          1
      7499 ALLEN      SALESMAN        7698 20-FEB-81       1600        300         30          2
      7521 WARD       SALESMAN        7698 22-FEB-81       1250        500         30          3
```

k) Query to display 2nd and 6th record from employee table.

```
SQL> SELECT * FROM
  2  (SELECT EMP.* , ROWNUM JSPIDER FROM EMP )
  3  WHERE JSPIDER IN(2 , 6) ;

     EMPNO ENAME      JOB              MGR HIREDATE         SAL       COMM     DEPTNO    JSPIDER
---------- ---------- --------- ---------- --------- ---------- ---------- ---------- ----------
      7499 ALLEN      SALESMAN        7698 20-FEB-81       1600        300         30          2
      7698 BLAKE      MANAGER         7839 01-MAY-81       2850                    30          6
```

l) Query to display last record from employee table.

```
SQL> SELECT * FROM
  2  (SELECT EMP.* , ROWNUM JSPIDER FROM EMP )
  3  WHERE JSPIDER  = (SELECT COUNT(*) FROM EMP ) ;

     EMPNO ENAME      JOB              MGR HIREDATE         SAL       COMM     DEPTNO    JSPIDER
---------- ---------- --------- ---------- --------- ---------- ---------- ---------- ----------
      7934 MILLER     CLERK           7782 23-JAN-82       1300                    10         14
```

m) Query to display last 4 records from employee table.

```
SQL> SELECT * FROM
  2  (SELECT EMP.* , ROWNUM JSPIDER FROM EMP )
  3  WHERE JSPIDER  > (SELECT COUNT(*) - 4 FROM EMP ) ;

     EMPNO ENAME      JOB              MGR HIREDATE         SAL       COMM     DEPTNO    JSPIDER
---------- ---------- --------- ---------- --------- ---------- ---------- ---------- ----------
      7876 ADAMS      CLERK           7788 23-MAY-87       1100                    20         11
      7900 JAMES      CLERK           7698 03-DEC-81        950                    30         12
      7902 FORD       ANALYST         7566 03-DEC-81       3000                    20         13
      7934 MILLER     CLERK           7782 23-JAN-82       1300                    10         14
```

n) Query to display even number records from employee table.

```
SQL> SELECT * FROM
  2  (SELECT EMP.* , ROWNUM JSPIDER FROM EMP )
  3  WHERE MOD(JSPIDER , 2) = 0 ;

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

7 rows selected.
```

o) Query to display odd number records from employee table.

```
SQL> SELECT * FROM
  2  (SELECT EMP.* , ROWNUM JSPIDER FROM EMP )
  3  WHERE MOD(JSPIDER , 2) = 1 ;

     EMPNO ENAME      JOB              MGR HIREDATE         SAL       COMM     DEPTNO    JSPIDER
---------- ---------- --------- ---------- --------- ---------- ---------- ---------- ----------
      7369 SMITH      CLERK            7902 17-DEC-80        800                    20          1
      7521 WARD       SALESMAN         7698 22-FEB-81       1250        500         30          3
      7654 MARTIN     SALESMAN         7698 28-SEP-81       1250       1400         30          5
      7782 CLARK      MANAGER          7839 09-JUN-81       2450                    10          7
      7839 KING       PRESIDENT             17-NOV-81       5000                    10          9
      7876 ADAMS      CLERK            7788 23-MAY-87       1100                    20         11
      7902 FORD       ANALYST          7566 03-DEC-81       3000                    20         13

7 rows selected.
```

p) Query to display starting 50% records from employee table.

```
SQL> SELECT * FROM
  2  (SELECT EMP.* , ROWNUM JSPIDER FROM EMP )
  3  WHERE JSPIDER <= (SELECT COUNT(*) * 0.5 FROM EMP ) ;

     EMPNO ENAME      JOB              MGR HIREDATE         SAL       COMM     DEPTNO    JSPIDER
---------- ---------- --------- ---------- --------- ---------- ---------- ---------- ----------
      7369 SMITH      CLERK            7902 17-DEC-80        800                    20          1
      7499 ALLEN      SALESMAN         7698 20-FEB-81       1600        300         30          2
      7521 WARD       SALESMAN         7698 22-FEB-81       1250        500         30          3
      7566 JONES      MANAGER          7839 02-APR-81       2975                    20          4
      7654 MARTIN     SALESMAN         7698 28-SEP-81       1250       1400         30          5
      7698 BLAKE      MANAGER          7839 01-MAY-81       2850                    30          6
      7782 CLARK      MANAGER          7839 09-JUN-81       2450                    10          7

7 rows selected.
```

q) Query to display last 50% records from employee table.

```
SQL>  SELECT * FROM
  2   (SELECT EMP.* , ROWNUM JSPIDER FROM EMP )
  3   WHERE JSPIDER > (SELECT COUNT(*) * 0.5 FROM EMP ) ;

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

7 rows selected.
```

r) Query to display starting 70% records from employee table.

```
SQL> SELECT * FROM
  2  (SELECT EMP.* , ROWNUM JSPIDER FROM EMP )
  3  WHERE JSPIDER <= (SELECT COUNT(*) * 0.7 FROM EMP ) ;

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

9 rows selected.
```

s) Query to display last 30% records from employee table.

```
SQL>  SELECT * FROM
  2   (SELECT EMP.* , ROWNUM JSPIDER FROM EMP )
  3   WHERE JSPIDER > (SELECT COUNT(*) * 0.7 FROM EMP ) ;

     EMPNO ENAME      JOB              MGR HIREDATE         SAL       COMM     DEPTNO    JSPIDER
---------- ---------- --------- ---------- --------- ---------- ---------- ---------- ----------
      7844 TURNER     SALESMAN        7698 08-SEP-81       1500          0         30         10
      7876 ADAMS      CLERK           7788 23-MAY-87       1100                    20         11
      7900 JAMES      CLERK           7698 03-DEC-81        950                    30         12
      7902 FORD       ANALYST         7566 03-DEC-81       3000                    20         13
      7934 MILLER     CLERK           7782 23-JAN-82       1300                    10         14
```

t) Query to display 3rd highest salary earning employee details

```
SQL> SELECT * FROM EMP E1 WHERE
  2  (SELECT COUNT(DISTINCT SAL) FROM EMP E2 WHERE
  3  E2.SAL >= E1.SAL ) = 3 ;

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

E2.SAL	->	->	=>	->	**E1.SAL**	COUNT
5000	800	3000	2975	>=	5000	1
5000	800	3000	2975	>=	800	4
5000	800	3000	2975	>=	3000	2
5000	800	3000	2975	>=	2975	3

u) Query to display 2nd lowest salary earning employee details

```
SQL> SELECT * FROM EMP E1 WHERE
  2  (SELECT COUNT(DISTINCT SAL) FROM EMP E2 WHERE
  3  E2.SAL <= E1.SAL ) = 2 ;

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

v) Query to display name of employee along with their salary and their department name and location for employee who are earning 1st highest salary earning employee.

```
SQL> SELECT ENAME , SAL , DNAME , LOC FROM
  2  EMP JOIN DEPT ON EMP.DEPTNO = DEPT.DEPTNO
  3  AND
  4  SAL =
  5  (SELECT E1.SAL FROM EMP E1 WHERE
  6  (SELECT COUNT(DISTINCT SAL) FROM EMP E2 WHERE
  7  E2.SAL >= E1.SAL ) = 1 ) ;

ENAME                SAL DNAME          LOC
---------- ---------- -------------- -------------
KING               5000 ACCOUNTING     NEW YORK
```

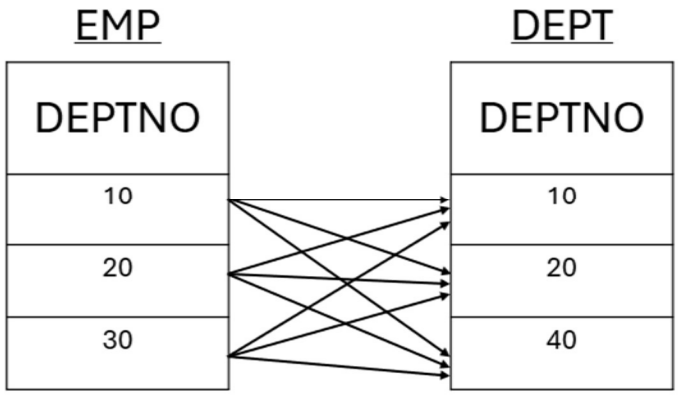
JOINS

Joining or merging of two or more tables and retrieving data from those tables is known as joins.
To achieve joins, common columns are mandatory in both the tables.
We can build relations on multiple table with the help of joins.

Types of JOINS :

1) **CROSS JOIN (CARTESIAN JOIN)**
2) **INNER JOIN**
3) **NON–EQUI JOIN**
4) **NATURAL JOIN**
5) **OUTER JOIN**
    a. **LEFT OUTER JOIN**
    b. **RIGHT OUTER JOIN**
    c. **FULL OUTER JOIN**
6) **SELF JOIN**

CROSS JOIN



OUTPUT

10	10	20	10	30	10
10	12	20	20	30	20
10	40	20	40	30	40

In cross join each record of table1 will be matched with each records of another table and it will display matched and unmatched records from all the tables.
In real time scenario to build relation on tables we are not using cross join because it will display incorrect output.

ORACLE Syntax:

```
SELECT * FROM TABLE-NAME1 , TABLE-NAME2 ;
```

```
SQL> SELECT * FROM EMP , DEPT ;

     EMPNO ENAME      JOB              MGR HIREDATE         SAL       COMM     DEPTNO     DEPTNO DNAME          LOC
---------- ---------- --------- ---------- --------- ---------- ---------- ---------- ---------- -------------- -------------
      7369 SMITH      CLERK           7902 17-DEC-80        800                    20         10 ACCOUNTING     NEW YORK
      7499 ALLEN      SALESMAN        7698 20-FEB-81       1600        300         30         10 ACCOUNTING     NEW YORK
      7521 WARD       SALESMAN        7698 22-FEB-81       1250        500         30         10 ACCOUNTING     NEW YORK
      7566 JONES      MANAGER         7839 02-APR-81       2975                    20         10 ACCOUNTING     NEW YORK
      7654 MARTIN     SALESMAN        7698 28-SEP-81       1250       1400         30         10 ACCOUNTING     NEW YORK
      7698 BLAKE      MANAGER         7839 01-MAY-81       2850                    30         10 ACCOUNTING     NEW YORK
      7782 CLARK      MANAGER         7839 09-JUN-81       2450                    10         10 ACCOUNTING     NEW YORK
      7788 SCOTT      ANALYST         7566 19-APR-87       3000                    20         10 ACCOUNTING     NEW YORK
      7839 KING       PRESIDENT            17-NOV-81       5000                    10         10 ACCOUNTING     NEW YORK
      7844 TURNER     SALESMAN        7698 08-SEP-81       1500          0         30         10 ACCOUNTING     NEW YORK
      7876 ADAMS      CLERK           7788 23-MAY-87       1100                    20         10 ACCOUNTING     NEW YORK
      7900 JAMES      CLERK           7698 03-DEC-81        950                    30         10 ACCOUNTING     NEW YORK
      7902 FORD       ANALYST         7566 03-DEC-81       3000                    20         10 ACCOUNTING     NEW YORK
      7934 MILLER     CLERK           7782 23-JAN-82       1300                    10         10 ACCOUNTING     NEW YORK
      7369 SMITH      CLERK           7902 17-DEC-80        800                    20         20 RESEARCH       DALLAS
      7499 ALLEN      SALESMAN        7698 20-FEB-81       1600        300         30         20 RESEARCH       DALLAS
      7521 WARD       SALESMAN        7698 22-FEB-81       1250        500         30         20 RESEARCH       DALLAS
      7566 JONES      MANAGER         7839 02-APR-81       2975                    20         20 RESEARCH       DALLAS
      7654 MARTIN     SALESMAN        7698 28-SEP-81       1250       1400         30         20 RESEARCH       DALLAS
      7698 BLAKE      MANAGER         7839 01-MAY-81       2850                    30         20 RESEARCH       DALLAS
      7782 CLARK      MANAGER         7839 09-JUN-81       2450                    10         20 RESEARCH       DALLAS
      7788 SCOTT      ANALYST         7566 19-APR-87       3000                    20         20 RESEARCH       DALLAS
      7839 KING       PRESIDENT            17-NOV-81       5000                    10         20 RESEARCH       DALLAS
      7844 TURNER     SALESMAN        7698 08-SEP-81       1500          0         30         20 RESEARCH       DALLAS
      7876 ADAMS      CLERK           7788 23-MAY-87       1100                    20         20 RESEARCH       DALLAS
      7900 JAMES      CLERK           7698 03-DEC-81        950                    30         20 RESEARCH       DALLAS
      7902 FORD       ANALYST         7566 03-DEC-81       3000                    20         20 RESEARCH       DALLAS
      7934 MILLER     CLERK           7782 23-JAN-82       1300                    10         20 RESEARCH       DALLAS
      7369 SMITH      CLERK           7902 17-DEC-80        800                    20         30 SALES          CHICAGO
      7499 ALLEN      SALESMAN        7698 20-FEB-81       1600        300         30         30 SALES          CHICAGO
      7521 WARD       SALESMAN        7698 22-FEB-81       1250        500         30         30 SALES          CHICAGO
      7566 JONES      MANAGER         7839 02-APR-81       2975                    20         30 SALES          CHICAGO
      7654 MARTIN     SALESMAN        7698 28-SEP-81       1250       1400         30         30 SALES          CHICAGO
      7698 BLAKE      MANAGER         7839 01-MAY-81       2850                    30         30 SALES          CHICAGO
      7782 CLARK      MANAGER         7839 09-JUN-81       2450                    10         30 SALES          CHICAGO
      7788 SCOTT      ANALYST         7566 19-APR-87       3000                    20         30 SALES          CHICAGO
      7839 KING       PRESIDENT            17-NOV-81       5000                    10         30 SALES          CHICAGO
      7844 TURNER     SALESMAN        7698 08-SEP-81       1500          0         30         30 SALES          CHICAGO
      7876 ADAMS      CLERK           7788 23-MAY-87       1100                    20         30 SALES          CHICAGO
      7900 JAMES      CLERK           7698 03-DEC-81        950                    30         30 SALES          CHICAGO
      7902 FORD       ANALYST         7566 03-DEC-81       3000                    20         30 SALES          CHICAGO
      7934 MILLER     CLERK           7782 23-JAN-82       1300                    10         30 SALES          CHICAGO
      7369 SMITH      CLERK           7902 17-DEC-80        800                    20         40 OPERATIONS     BOSTON
      7499 ALLEN      SALESMAN        7698 20-FEB-81       1600        300         30         40 OPERATIONS     BOSTON
      7521 WARD       SALESMAN        7698 22-FEB-81       1250        500         30         40 OPERATIONS     BOSTON
      7566 JONES      MANAGER         7839 02-APR-81       2975                    20         40 OPERATIONS     BOSTON
      7654 MARTIN     SALESMAN        7698 28-SEP-81       1250       1400         30         40 OPERATIONS     BOSTON
      7698 BLAKE      MANAGER         7839 01-MAY-81       2850                    30         40 OPERATIONS     BOSTON
      7782 CLARK      MANAGER         7839 09-JUN-81       2450                    10         40 OPERATIONS     BOSTON
      7788 SCOTT      ANALYST         7566 19-APR-87       3000                    20         40 OPERATIONS     BOSTON
      7839 KING       PRESIDENT            17-NOV-81       5000                    10         40 OPERATIONS     BOSTON
      7844 TURNER     SALESMAN        7698 08-SEP-81       1500          0         30         40 OPERATIONS     BOSTON
      7876 ADAMS      CLERK           7788 23-MAY-87       1100                    20         40 OPERATIONS     BOSTON
      7900 JAMES      CLERK           7698 03-DEC-81        950                    30         40 OPERATIONS     BOSTON
      7902 FORD       ANALYST         7566 03-DEC-81       3000                    20         40 OPERATIONS     BOSTON
      7934 MILLER     CLERK           7782 23-JAN-82       1300                    10         40 OPERATIONS     BOSTON

56 rows selected.
```

ANSI Syntax:

```
SELECT * FROM TABLE-NAME1 CROSS JOIN TABLE-NAME2 ;
```

```
SQL> SELECT * FROM EMP CROSS JOIN DEPT ;

     EMPNO ENAME      JOB              MGR HIREDATE         SAL       COMM     DEPTNO     DEPTNO DNAME          LOC
---------- ---------- --------- ---------- --------- ---------- ---------- ---------- ---------- -------------- -------------
      7369 SMITH      CLERK           7902 17-DEC-80        800                    20         10 ACCOUNTING     NEW YORK
      7499 ALLEN      SALESMAN        7698 20-FEB-81       1600        300         30         10 ACCOUNTING     NEW YORK
      7521 WARD       SALESMAN        7698 22-FEB-81       1250        500         30         10 ACCOUNTING     NEW YORK
      7566 JONES      MANAGER         7839 02-APR-81       2975                    20         10 ACCOUNTING     NEW YORK
      7654 MARTIN     SALESMAN        7698 28-SEP-81       1250       1400         30         10 ACCOUNTING     NEW YORK
      7698 BLAKE      MANAGER         7839 01-MAY-81       2850                    30         10 ACCOUNTING     NEW YORK
      7782 CLARK      MANAGER         7839 09-JUN-81       2450                    10         10 ACCOUNTING     NEW YORK
      7788 SCOTT      ANALYST         7566 19-APR-87       3000                    20         10 ACCOUNTING     NEW YORK
      7839 KING       PRESIDENT            17-NOV-81       5000                    10         10 ACCOUNTING     NEW YORK
      7844 TURNER     SALESMAN        7698 08-SEP-81       1500          0         30         10 ACCOUNTING     NEW YORK
      7876 ADAMS      CLERK           7788 23-MAY-87       1100                    20         10 ACCOUNTING     NEW YORK
      7900 JAMES      CLERK           7698 03-DEC-81        950                    30         10 ACCOUNTING     NEW YORK
      7902 FORD       ANALYST         7566 03-DEC-81       3000                    20         10 ACCOUNTING     NEW YORK
      7934 MILLER     CLERK           7782 23-JAN-82       1300                    10         10 ACCOUNTING     NEW YORK
      7369 SMITH      CLERK           7902 17-DEC-80        800                    20         20 RESEARCH       DALLAS
      7499 ALLEN      SALESMAN        7698 20-FEB-81       1600        300         30         20 RESEARCH       DALLAS
      7521 WARD       SALESMAN        7698 22-FEB-81       1250        500         30         20 RESEARCH       DALLAS
      7566 JONES      MANAGER         7839 02-APR-81       2975                    20         20 RESEARCH       DALLAS
      7654 MARTIN     SALESMAN        7698 28-SEP-81       1250       1400         30         20 RESEARCH       DALLAS
      7698 BLAKE      MANAGER         7839 01-MAY-81       2850                    30         20 RESEARCH       DALLAS
      7782 CLARK      MANAGER         7839 09-JUN-81       2450                    10         20 RESEARCH       DALLAS
      7788 SCOTT      ANALYST         7566 19-APR-87       3000                    20         20 RESEARCH       DALLAS
      7839 KING       PRESIDENT            17-NOV-81       5000                    10         20 RESEARCH       DALLAS
      7844 TURNER     SALESMAN        7698 08-SEP-81       1500          0         30         20 RESEARCH       DALLAS
      7876 ADAMS      CLERK           7788 23-MAY-87       1100                    20         20 RESEARCH       DALLAS
      7900 JAMES      CLERK           7698 03-DEC-81        950                    30         20 RESEARCH       DALLAS
      7902 FORD       ANALYST         7566 03-DEC-81       3000                    20         20 RESEARCH       DALLAS
      7934 MILLER     CLERK           7782 23-JAN-82       1300                    10         20 RESEARCH       DALLAS
      7369 SMITH      CLERK           7902 17-DEC-80        800                    20         30 SALES          CHICAGO
      7499 ALLEN      SALESMAN        7698 20-FEB-81       1600        300         30         30 SALES          CHICAGO
      7521 WARD       SALESMAN        7698 22-FEB-81       1250        500         30         30 SALES          CHICAGO
      7566 JONES      MANAGER         7839 02-APR-81       2975                    20         30 SALES          CHICAGO
      7654 MARTIN     SALESMAN        7698 28-SEP-81       1250       1400         30         30 SALES          CHICAGO
      7698 BLAKE      MANAGER         7839 01-MAY-81       2850                    30         30 SALES          CHICAGO
      7782 CLARK      MANAGER         7839 09-JUN-81       2450                    10         30 SALES          CHICAGO
      7788 SCOTT      ANALYST         7566 19-APR-87       3000                    20         30 SALES          CHICAGO
      7839 KING       PRESIDENT            17-NOV-81       5000                    10         30 SALES          CHICAGO
      7844 TURNER     SALESMAN        7698 08-SEP-81       1500          0         30         30 SALES          CHICAGO
      7876 ADAMS      CLERK           7788 23-MAY-87       1100                    20         30 SALES          CHICAGO
      7900 JAMES      CLERK           7698 03-DEC-81        950                    30         30 SALES          CHICAGO
      7902 FORD       ANALYST         7566 03-DEC-81       3000                    20         30 SALES          CHICAGO
      7934 MILLER     CLERK           7782 23-JAN-82       1300                    10         30 SALES          CHICAGO
      7369 SMITH      CLERK           7902 17-DEC-80        800                    20         40 OPERATIONS     BOSTON
      7499 ALLEN      SALESMAN        7698 20-FEB-81       1600        300         30         40 OPERATIONS     BOSTON
      7521 WARD       SALESMAN        7698 22-FEB-81       1250        500         30         40 OPERATIONS     BOSTON
      7566 JONES      MANAGER         7839 02-APR-81       2975                    20         40 OPERATIONS     BOSTON
      7654 MARTIN     SALESMAN        7698 28-SEP-81       1250       1400         30         40 OPERATIONS     BOSTON
      7698 BLAKE      MANAGER         7839 01-MAY-81       2850                    30         40 OPERATIONS     BOSTON
      7782 CLARK      MANAGER         7839 09-JUN-81       2450                    10         40 OPERATIONS     BOSTON
      7788 SCOTT      ANALYST         7566 19-APR-87       3000                    20         40 OPERATIONS     BOSTON
      7839 KING       PRESIDENT            17-NOV-81       5000                    10         40 OPERATIONS     BOSTON
      7844 TURNER     SALESMAN        7698 08-SEP-81       1500          0         30         40 OPERATIONS     BOSTON
      7876 ADAMS      CLERK           7788 23-MAY-87       1100                    20         40 OPERATIONS     BOSTON
      7900 JAMES      CLERK           7698 03-DEC-81        950                    30         40 OPERATIONS     BOSTON
      7902 FORD       ANALYST         7566 03-DEC-81       3000                    20         40 OPERATIONS     BOSTON
      7934 MILLER     CLERK           7782 23-JAN-82       1300                    10         40 OPERATIONS     BOSTON

56 rows selected.
```

INNER JOIN

ORACLE Syntax:

```
SELECT * FROM TABLE-NAME1 , TABLE-NAME2
WHERE TABLE-NAME1.COMMON-COLUMN = TABLE-NAME2.COMMON-COLUMN ;
```

```
SQL> SELECT * FROM EMP , DEPT
  2  WHERE EMP.DEPTNO = DEPT.DEPTNO ;

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

14 rows selected.
```

ANSI Syntax:

```
SELECT * FROM TABLE-NAME1 JOIN TABLE-NAME2
ON TABLE-NAME1.COMMON-COLUMN = TABLE-NAME2.COMMON-COLUMN ;
```

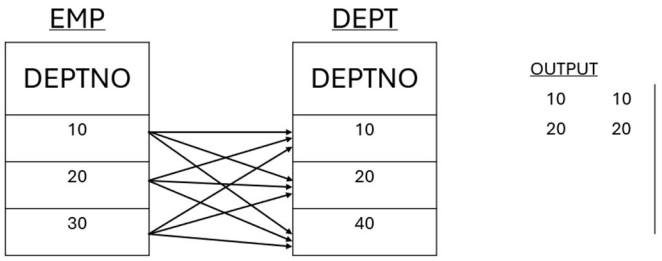
```
SQL> SELECT * FROM EMP JOIN DEPT
  2  ON EMP.DEPTNO = DEPT.DEPTNO ;

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

14 rows selected.
```

in inner join each record of table1 will be matched with each records of other tables and it will display only matched records from all the tables.

In real time scenario to build relations on multiple tables we can make use of inner join.



NON–EQUI JOIN

ORACLE Syntax:

```
SELECT * FROM TABLE-NAME1 , TABLE-NAME2
WHERE TABLE-NAME1.COMMON-COLUMN != TABLE-NAME2.COMMON-COLUMN ;
```

```
SQL> SELECT * FROM EMP , DEPT
  2  WHERE EMP.DEPTNO != DEPT.DEPTNO ;

     EMPNO ENAME      JOB              MGR HIREDATE         SAL       COMM     DEPTNO     DEPTNO DNAME          LOC
---------- ---------- --------- ---------- --------- ---------- ---------- ---------- ---------- -------------- -----------
      7369 SMITH      CLERK           7902 17-DEC-80        800                    20         10 ACCOUNTING     NEW YORK
      7499 ALLEN      SALESMAN        7698 20-FEB-81       1600        300         30         10 ACCOUNTING     NEW YORK
      7521 WARD       SALESMAN        7698 22-FEB-81       1250        500         30         10 ACCOUNTING     NEW YORK
      7566 JONES      MANAGER         7839 02-APR-81       2975                    20         10 ACCOUNTING     NEW YORK
      7654 MARTIN     SALESMAN        7698 28-SEP-81       1250       1400         30         10 ACCOUNTING     NEW YORK
      7698 BLAKE      MANAGER         7839 01-MAY-81       2850                    30         10 ACCOUNTING     NEW YORK
      7788 SCOTT      ANALYST         7566 19-APR-87       3000                    20         10 ACCOUNTING     NEW YORK
      7844 TURNER     SALESMAN        7698 08-SEP-81       1500          0         30         10 ACCOUNTING     NEW YORK
      7876 ADAMS      CLERK           7788 23-MAY-87       1100                    20         10 ACCOUNTING     NEW YORK
      7900 JAMES      CLERK           7698 03-DEC-81        950                    30         10 ACCOUNTING     NEW YORK
      7902 FORD       ANALYST         7566 03-DEC-81       3000                    20         10 ACCOUNTING     NEW YORK
      7499 ALLEN      SALESMAN        7698 20-FEB-81       1600        300         30         20 RESEARCH       DALLAS
      7521 WARD       SALESMAN        7698 22-FEB-81       1250        500         30         20 RESEARCH       DALLAS
      7654 MARTIN     SALESMAN        7698 28-SEP-81       1250       1400         30         20 RESEARCH       DALLAS
      7698 BLAKE      MANAGER         7839 01-MAY-81       2850                    30         20 RESEARCH       DALLAS
      7782 CLARK      MANAGER         7839 09-JUN-81       2450                    10         20 RESEARCH       DALLAS
      7839 KING       PRESIDENT            17-NOV-81       5000                    10         20 RESEARCH       DALLAS
      7844 TURNER     SALESMAN        7698 08-SEP-81       1500          0         30         20 RESEARCH       DALLAS
      7900 JAMES      CLERK           7698 03-DEC-81        950                    30         20 RESEARCH       DALLAS
      7934 MILLER     CLERK           7782 23-JAN-82       1300                    10         20 RESEARCH       DALLAS
      7369 SMITH      CLERK           7902 17-DEC-80        800                    20         30 SALES          CHICAGO
      7566 JONES      MANAGER         7839 02-APR-81       2975                    20         30 SALES          CHICAGO
      7782 CLARK      MANAGER         7839 09-JUN-81       2450                    10         30 SALES          CHICAGO
      7788 SCOTT      ANALYST         7566 19-APR-87       3000                    20         30 SALES          CHICAGO
      7839 KING       PRESIDENT            17-NOV-81       5000                    10         30 SALES          CHICAGO
      7876 ADAMS      CLERK           7788 23-MAY-87       1100                    20         30 SALES          CHICAGO
      7902 FORD       ANALYST         7566 03-DEC-81       3000                    20         30 SALES          CHICAGO
      7934 MILLER     CLERK           7782 23-JAN-82       1300                    10         30 SALES          CHICAGO
      7369 SMITH      CLERK           7902 17-DEC-80        800                    20         40 OPERATIONS     BOSTON
      7499 ALLEN      SALESMAN        7698 20-FEB-81       1600        300         30         40 OPERATIONS     BOSTON
      7521 WARD       SALESMAN        7698 22-FEB-81       1250        500         30         40 OPERATIONS     BOSTON
      7566 JONES      MANAGER         7839 02-APR-81       2975                    20         40 OPERATIONS     BOSTON
      7654 MARTIN     SALESMAN        7698 28-SEP-81       1250       1400         30         40 OPERATIONS     BOSTON
      7698 BLAKE      MANAGER         7839 01-MAY-81       2850                    30         40 OPERATIONS     BOSTON
      7782 CLARK      MANAGER         7839 09-JUN-81       2450                    10         40 OPERATIONS     BOSTON
      7788 SCOTT      ANALYST         7566 19-APR-87       3000                    20         40 OPERATIONS     BOSTON
      7839 KING       PRESIDENT            17-NOV-81       5000                    10         40 OPERATIONS     BOSTON
      7844 TURNER     SALESMAN        7698 08-SEP-81       1500          0         30         40 OPERATIONS     BOSTON
      7876 ADAMS      CLERK           7788 23-MAY-87       1100                    20         40 OPERATIONS     BOSTON
      7900 JAMES      CLERK           7698 03-DEC-81        950                    30         40 OPERATIONS     BOSTON
      7902 FORD       ANALYST         7566 03-DEC-81       3000                    20         40 OPERATIONS     BOSTON
      7934 MILLER     CLERK           7782 23-JAN-82       1300                    10         40 OPERATIONS     BOSTON

42 rows selected.
```

ANSI Syntax:

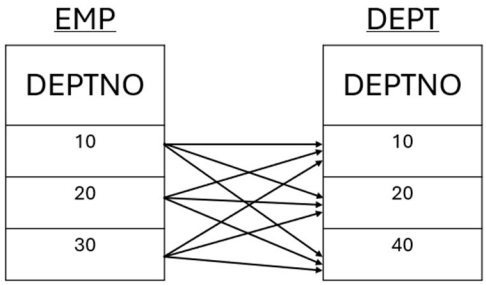
SELECT * FROM TABLE-NAME1 JOIN TABLE-NAME2
ON TABLE-NAME1.COMMON-COLUMN ! = TABLE-NAME2.COMMON-COLUMN ;

```
SQL> SELECT * FROM EMP JOIN DEPT
  2  ON EMP.DEPTNO != DEPT.DEPTNO ;
```

EMPNO	ENAME	JOB	MGR	HIREDATE	SAL	COMM	DEPTNO	DEPTNO	DNAME	LOC
7369	SMITH	CLERK	7902	17-DEC-80	800		20	10	ACCOUNTING	NEW YORK
7499	ALLEN	SALESMAN	7698	20-FEB-81	1600	300	30	10	ACCOUNTING	NEW YORK
7521	WARD	SALESMAN	7698	22-FEB-81	1250	500	30	10	ACCOUNTING	NEW YORK
7566	JONES	MANAGER	7839	02-APR-81	2975		20	10	ACCOUNTING	NEW YORK
7654	MARTIN	SALESMAN	7698	28-SEP-81	1250	1400	30	10	ACCOUNTING	NEW YORK
7698	BLAKE	MANAGER	7839	01-MAY-81	2850		30	10	ACCOUNTING	NEW YORK
7788	SCOTT	ANALYST	7566	19-APR-87	3000		20	10	ACCOUNTING	NEW YORK
7844	TURNER	SALESMAN	7698	08-SEP-81	1500	0	30	10	ACCOUNTING	NEW YORK
7876	ADAMS	CLERK	7788	23-MAY-87	1100		20	10	ACCOUNTING	NEW YORK
7900	JAMES	CLERK	7698	03-DEC-81	950		30	10	ACCOUNTING	NEW YORK
7902	FORD	ANALYST	7566	03-DEC-81	3000		20	10	ACCOUNTING	NEW YORK
7499	ALLEN	SALESMAN	7698	20-FEB-81	1600	300	30	20	RESEARCH	DALLAS
7521	WARD	SALESMAN	7698	22-FEB-81	1250	500	30	20	RESEARCH	DALLAS
7654	MARTIN	SALESMAN	7698	28-SEP-81	1250	1400	30	20	RESEARCH	DALLAS
7698	BLAKE	MANAGER	7839	01-MAY-81	2850		30	20	RESEARCH	DALLAS
7782	CLARK	MANAGER	7839	09-JUN-81	2450		10	20	RESEARCH	DALLAS
7839	KING	PRESIDENT		17-NOV-81	5000		10	20	RESEARCH	DALLAS
7844	TURNER	SALESMAN	7698	08-SEP-81	1500	0	30	20	RESEARCH	DALLAS
7900	JAMES	CLERK	7698	03-DEC-81	950		30	20	RESEARCH	DALLAS
7934	MILLER	CLERK	7782	23-JAN-82	1300		10	20	RESEARCH	DALLAS
7369	SMITH	CLERK	7902	17-DEC-80	800		20	30	SALES	CHICAGO
7566	JONES	MANAGER	7839	02-APR-81	2975		20	30	SALES	CHICAGO
7782	CLARK	MANAGER	7839	09-JUN-81	2450		10	30	SALES	CHICAGO
7788	SCOTT	ANALYST	7566	19-APR-87	3000		20	30	SALES	CHICAGO
7839	KING	PRESIDENT		17-NOV-81	5000		10	30	SALES	CHICAGO
7876	ADAMS	CLERK	7788	23-MAY-87	1100		20	30	SALES	CHICAGO
7902	FORD	ANALYST	7566	03-DEC-81	3000		20	30	SALES	CHICAGO
7934	MILLER	CLERK	7782	23-JAN-82	1300		10	30	SALES	CHICAGO
7369	SMITH	CLERK	7902	17-DEC-80	800		20	40	OPERATIONS	BOSTON
7499	ALLEN	SALESMAN	7698	20-FEB-81	1600	300	30	40	OPERATIONS	BOSTON
7521	WARD	SALESMAN	7698	22-FEB-81	1250	500	30	40	OPERATIONS	BOSTON
7566	JONES	MANAGER	7839	02-APR-81	2975		20	40	OPERATIONS	BOSTON
7654	MARTIN	SALESMAN	7698	28-SEP-81	1250	1400	30	40	OPERATIONS	BOSTON
7698	BLAKE	MANAGER	7839	01-MAY-81	2850		30	40	OPERATIONS	BOSTON
7782	CLARK	MANAGER	7839	09-JUN-81	2450		10	40	OPERATIONS	BOSTON
7788	SCOTT	ANALYST	7566	19-APR-87	3000		20	40	OPERATIONS	BOSTON
7839	KING	PRESIDENT		17-NOV-81	5000		10	40	OPERATIONS	BOSTON
7844	TURNER	SALESMAN	7698	08-SEP-81	1500	0	30	40	OPERATIONS	BOSTON
7876	ADAMS	CLERK	7788	23-MAY-87	1100		20	40	OPERATIONS	BOSTON
7900	JAMES	CLERK	7698	03-DEC-81	950		30	40	OPERATIONS	BOSTON
7902	FORD	ANALYST	7566	03-DEC-81	3000		20	40	OPERATIONS	BOSTON
7934	MILLER	CLERK	7782	23-JAN-82	1300		10	40	OPERATIONS	BOSTON

```
42 rows selected.
```

in inner join each record of table1 will be matched with each records of other tables and it will display only unmatched records from all the tables.



OUTPUT

				30	10
10	20	20	10	30	20
10	40	20	40	30	40

NATURAL JOIN

Syntax:

```
SELECT * FROM TABLE-NAME1 NATURAL JOIN TABLE-NAME2 ;
```

```
SQL> SELECT * FROM EMP NATURAL JOIN DEPT ;

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

14 rows selected.
```

Natural join will display only matched records from all the tables and eliminate duplicate columns.

LEFT OUTER JOIN

Left outer join will display unmatched records from left table and matched records from both the tables.

ORACLE Syntax:

```
SELECT * FROM TABLE-NAME1 , TABLE-NAME2
WHERE TABLE-NAME1.COMMON-COLUMN = TABLE-NAME2.COMMON-COLUMN (+) ;
```

```
SQL> SELECT * FROM EMP , DEPT
  2  WHERE EMP.DEPTNO = DEPT.DEPTNO(+) ;

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

14 rows selected.
```

```
SQL> SELECT * FROM EMP , DEPT
  2  WHERE DEPT.DEPTNO = EMP.DEPTNO(+) ;

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

15 rows selected.
```

ANSI Syntax:

SELECT * FROM TABLE-NAME1 LEFT OUTER JOIN TABLE-NAME2
ON TABLE-NAME1.COMMON-COLUMN = TABLE-NAME2.COMMON-COLUMN ;

```
SQL> SELECT * FROM EMP LEFT OUTER JOIN DEPT
  2  ON EMP.DEPTNO = DEPT.DEPTNO ;

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

14 rows selected.

SQL> SELECT * FROM DEPT LEFT OUTER JOIN EMP
  2  ON EMP.DEPTNO = DEPT.DEPTNO ;

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

15 rows selected.
```

RIGHT OUTER JOIN

Left outer join will display unmatched records from left table and matched records from both the tables.

ORACLE Syntax:

```
SELECT * FROM TABLE-NAME1 , TABLE-NAME2
WHERE TABLE-NAME1.COMMON-COLUMN (+) = TABLE-NAME2.COMMON-COLUMN ;
```

```
SQL> SELECT * FROM EMP , DEPT
  2  WHERE EMP.DEPTNO(+) = DEPT.DEPTNO ;

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

15 rows selected.

SQL> SELECT * FROM EMP , DEPT
  2  WHERE DEPT.DEPTNO(+) = EMP.DEPTNO ;

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

14 rows selected.
```

ANSI Syntax:

```
SELECT * FROM TABLE-NAME1 RIGHT OUTER JOIN TABLE-NAME2
ON TABLE-NAME1.COMMON-COLUMN = TABLE-NAME2.COMMON-COLUMN ;
```

```
SQL> SELECT * FROM EMP RIGHT OUTER JOIN DEPT
  2  ON EMP.DEPTNO = DEPT.DEPTNO ;

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

15 rows selected.

SQL>
SQL> SELECT * FROM DEPT RIGHT OUTER JOIN EMP
  2  ON EMP.DEPTNO = DEPT.DEPTNO ;

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

14 rows selected.
```

FULL OUTER JOIN

Full outer join will display matched records and all the remaining unmatched records from all the tables.

ORACLE Syntax:

```
SELECT * FROM TABLE-NAME1 , TABLE-NAME2
WHERE TABLE-NAME1.COMMON-COLUMN (+) = TABLE-NAME2.COMMON-COLUMN (+) ;
```

OUTPUT : ERROR

ANSI Syntax:

```
SELECT * FROM TABLE-NAME1 FULL OUTER JOIN TABLE-NAME2
ON TABLE-NAME1.COMMON-COLUMN  = TABLE-NAME2.COMMON-COLUMN  ;
```

```
SQL> SELECT * FROM EMP FULL OUTER JOIN DEPT
  2  ON EMP.DEPTNO = DEPT.DEPTNO ;

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

15 rows selected.
```

QUERIES

a) Query to display name of employee along with their salary and location for employee who are earning more than 2000.

```
SQL> SELECT ENAME , SAL , LOC FROM EMP JOIN DEPT
  2  ON EMP.DEPTNO = DEPT.DEPTNO
  3  AND SAL > 2000 ;

ENAME             SAL LOC
---------- ---------- -------------
JONES            2975 DALLAS
BLAKE            2850 CHICAGO
CLARK            2450 NEW YORK
SCOTT            3000 DALLAS
KING             5000 NEW YORK
FORD             3000 DALLAS

6 rows selected.
```

b) Query to display employee info along with their department info for employee who belongs from research department.

```
SQL> SELECT * FROM EMP JOIN DEPT
  2  ON EMP.DEPTNO = DEPT.DEPTNO
  3  AND DNAME = 'RESEARCH' ;

     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
```

c) Query to display employee info along with their department for employee who belongs to department number 10.

```
SQL> SELECT * FROM EMP JOIN DEPT
  2  ON EMP.DEPTNO = DEPT.DEPTNO
  3  AND DEPTNO = 10 ;
AND DEPTNO = 10
    *
ERROR at line 3:
ORA-00918: column ambiguously defined
```

Here, You Will Get an Ambiguity Error Regarding From Which table to choose DeptNo. So, Define Which Table You Want to Choose DeptNo 10 from.

```
SQL> SELECT * FROM EMP JOIN DEPT
  2  ON EMP.DEPTNO = DEPT.DEPTNO
  3  AND EMP.DEPTNO = 10 ;

     EMPNO ENAME      JOB              MGR HIREDATE         SAL       COMM     DEPTNO     DEPTNO DNAME          LOC
---------- ---------- --------- ---------- --------- ---------- ---------- ---------- ---------- -------------- ----------
      7782 CLARK      MANAGER         7839 09-JUN-81       2450                    10         10 ACCOUNTING     NEW YORK
      7839 KING       PRESIDENT            17-NOV-81       5000                    10         10 ACCOUNTING     NEW YORK
      7934 MILLER     CLERK           7782 23-JAN-82       1300                    10         10 ACCOUNTING     NEW YORK
```

d) Query to display name of employee along with their department number and department name for employee who belongs from research and sale department.

```
SQL> SELECT ENAME , DEPTNO , DNAME FROM EMP JOIN DEPT
  2  ON EMP.DEPTNO = DEPT.DEPTNO
  3  AND DNAME IN('SALES' , 'RESEARCH') ;
SELECT ENAME , DEPTNO , DNAME FROM EMP JOIN DEPT
               *
ERROR at line 1:
ORA-00918: column ambiguously defined
```

Here, You Will Get an Ambiguity Error Regarding From Which table to choose DeptNo. So, Define Which Table You Want to Choose DeptNo from.

```
SQL> SELECT ENAME , EMP.DEPTNO , DNAME FROM EMP JOIN DEPT
  2  ON EMP.DEPTNO = DEPT.DEPTNO
  3  AND DNAME IN('SALES' , 'RESEARCH') ;

ENAME          DEPTNO DNAME
---------- ---------- --------------
SMITH              20 RESEARCH
ALLEN              30 SALES
WARD               30 SALES
JONES              20 RESEARCH
MARTIN             30 SALES
BLAKE              30 SALES
SCOTT              20 RESEARCH
TURNER             30 SALES
ADAMS              20 RESEARCH
JAMES              30 SALES
FORD               20 RESEARCH

11 rows selected.
```

e) Query to display employee info along with their department name and location for employee who belongs from smith's designation.

```
SQL> SELECT EMP.* , DNAME , LOC FROM EMP JOIN DEPT
  2  ON EMP.DEPTNO = DEPT.DEPTNO
  3  AND JOB = (SELECT JOB FROM EMP WHERE ENAME = 'SMITH' ) ;

     EMPNO ENAME      JOB              MGR HIREDATE         SAL       COMM     DEPTNO DNAME          LOC
---------- ---------- --------- ---------- --------- ---------- ---------- ---------- -------------- ----------
      7369 SMITH      CLERK           7902 17-DEC-80        800                    20 RESEARCH       DALLAS
      7876 ADAMS      CLERK           7788 23-MAY-87       1100                    20 RESEARCH       DALLAS
      7900 JAMES      CLERK           7698 03-DEC-81        950                    30 SALES          CHICAGO
      7934 MILLER     CLERK           7782 23-JAN-82       1300                    10 ACCOUNTING     NEW YORK
```

f) Query to display employee information along with their department info for employee who are earning more than Scott.

```
SQL> SELECT * FROM EMP JOIN DEPT
  2  ON EMP.DEPTNO = DEPT.DEPTNO
  3  AND SAL > (SELECT SAL FROM EMP WHERE ENAME = 'SCOTT' ) ;

     EMPNO ENAME      JOB              MGR HIREDATE         SAL       COMM     DEPTNO     DEPTNO DNAME          LOC
---------- ---------- --------- ---------- --------- ---------- ---------- ---------- ---------- -------------- ----------
      7839 KING       PRESIDENT            17-NOV-81       5000                    10         10 ACCOUNTING     NEW YORK
```

g) Query to display employee information along with their department information for employee who are earning similar salary in research department.

```
SQL> SELECT * FROM EMP JOIN DEPT
  2  ON EMP.DEPTNO = DEPT.DEPTNO
  3  AND SAL IN (SELECT SAL FROM EMP GROUP BY SAL HAVING COUNT(*) > 1 )
  4  AND DNAME = 'RESEARCH' ;

     EMPNO ENAME      JOB              MGR HIREDATE         SAL       COMM     DEPTNO     DEPTNO DNAME          LOC
---------- ---------- --------- ---------- --------- ---------- ---------- ---------- ---------- -------------- -------
      7902 FORD       ANALYST         7566 03-DEC-81       3000                    20         20 RESEARCH       DALLAS
      7788 SCOTT      ANALYST         7566 19-APR-87       3000                    20         20 RESEARCH       DALLAS
```

h) Query to display 3rd highest salary earning employee information along with their department information.

```
SQL> SELECT * FROM EMP JOIN DEPT
  2  ON EMP.DEPTNO = DEPT.DEPTNO
  3  AND SAL =
  4  (SELECT SAL FROM
  5  (SELECT EMP.* , DENSE_RANK() OVER(ORDER BY SAL DESC) AS HSAL FROM EMP )
  6  WHERE HSAL = 3 ) ;

     EMPNO ENAME      JOB              MGR HIREDATE         SAL       COMM     DEPTNO     DEPTNO DNAME          LOC
---------- ---------- --------- ---------- --------- ---------- ---------- ---------- ---------- -------------- -------
      7566 JONES      MANAGER         7839 02-APR-81       2975                    20         20 RESEARCH       DALLAS
```

SELF JOIN

Joining the table by a part of it own is known as self join.
Whenever we want to match records in same table we can make use of self join.
To achieve self join aliasing is mandatory.

In self join aliasing for table : **TableName AliasName**
EX : **EMP E1**

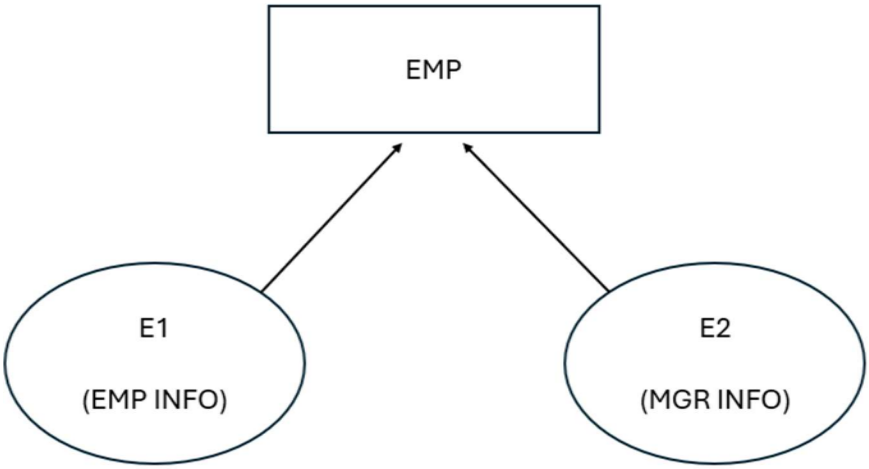
In self join aliasing for columns : **AliasName . ColumnName**
EX : **E1.SAL**

ORACLE Syntax:

```
SELECT AliasName . ColumnName
FROM TableName1 AliasName1 , TableName2 AliasName2
WHERE AliasName1 . Common-Column = AliasName2 . Common-Column ;
```

ANSI Syntax:

```
SELECT AliasName . ColumnName
FROM TableName1 AliasName1 JOIN TableName2 AliasName2
ON AliasName1 . Common-Column = AliasName2 . Common-Column ;
```



E1.SAL -> EMP SALARY
E2.SAL -> MANAGER SALARY

E1.JOB -> EMP DESIGNATION
E2.JOB -> MANAGER DESIGNATION

QUERIES ON EMPLOYEE AND THEIR MANAGER RELATION USING SELF JOIN

Note:

whenever we are having employee and their manager relation , for employee we have to find their manager number and for manager we have to find their employee number.

a) Query to display name of the employee along with their manager name.

```
SQL> SELECT E1.ENAME , E2.ENAME AS MANAGER
  2  FROM EMP E1 JOIN EMP E2
  3  ON E1.MGR = E2.EMPNO ;

ENAME      MANAGER
---------- ----------
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

13 rows selected.
```

b) Query to display employee name along with their manager name and their salaries. For employee , display the output if employee is earning more that their manager's salary.

```
SQL> SELECT E1.ENAME AS EMP , E2.ENAME AS MANAGER , E1.SAL AS EMP_SAL , E2.SAL AS MGR_SAL
  2  FROM EMP E1 JOIN EMP E2
  3  ON E1.MGR = E2.EMPNO
  4  AND E1.SAL > E2.SAL ;

EMP        MANAGER       EMP_SAL    MGR_SAL
---------- ---------- ---------- ----------
SCOTT      JONES            3000       2975
FORD       JONES            3000       2975
```

c) Query to display employee name along with their manager name and their designation. For employee display the output if employee and their manager belongs from same designation.

```
SQL> SELECT E1.ENAME , E2.ENAME , E1.JOB , E2.JOB
  2  FROM EMP E1 JOIN EMP E2
  3  ON E1.MGR = E2.EMPNO
  4  AND E1.JOB = E2.JOB ;

no rows selected
```

d) Query to display employee name and their manager name along with their joining date. For employee display the output if employee joined before their manager.

```
SQL> SELECT E1.ENAME AS EMP , E1.HIREDATE , E2.ENAME AS MGR ,  E2.HIREDATE
  2  FROM EMP E1 JOIN EMP E2
  3  ON E1.MGR = E2.EMPNO
  4  AND E1.HIREDATE < E2.HIREDATE ;

EMP        HIREDATE  MGR        HIREDATE
---------- --------- ---------- ---------
SMITH      17-DEC-80 FORD       03-DEC-81
ALLEN      20-FEB-81 BLAKE      01-MAY-81
WARD       22-FEB-81 BLAKE      01-MAY-81
JONES      02-APR-81 KING       17-NOV-81
BLAKE      01-MAY-81 KING       17-NOV-81
CLARK      09-JUN-81 KING       17-NOV-81

6 rows selected.
```

e) Query to display employee information along with their manager's information. For employee display the output if employee are earning commission.

```
SQL> SELECT * FROM EMP E1 JOIN EMP E2
  2  ON E1.MGR = E2.EMPNO
  3  AND E1.COMM IS NOT NULL ;
```

EMPNO	ENAME	JOB	MGR	HIREDATE	SAL	COMM	DEPTNO	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
7521	WARD	SALESMAN	7698	22-FEB-81	1250	500	30	7698	BLAKE	MANAGER	7839	01-MAY-81	2850		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	7698	BLAKE	MANAGER	7839	01-MAY-81	2850		30

f) Query to display employee information along with their manager's information , for employee if employee joined in year 1981.

```
SQL> SELECT * FROM EMP E1 JOIN EMP E2
  2  ON E1.MGR = E2.EMPNO
  3  AND TO_CHAR(E1.HIREDATE , 'YYYY' ) = 1981 ;
```

EMPNO	ENAME	JOB	MGR	HIREDATE	SAL	COMM	DEPTNO	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
7521	WARD	SALESMAN	7698	22-FEB-81	1250	500	30	7698	BLAKE	MANAGER	7839	01-MAY-81	2850		30
7566	JONES	MANAGER	7839	02-APR-81	2975		20	7839	KING	PRESIDENT		17-NOV-81	5000		10
7654	MARTIN	SALESMAN	7698	28-SEP-81	1250	1400	30	7698	BLAKE	MANAGER	7839	01-MAY-81	2850		30
7698	BLAKE	MANAGER	7839	01-MAY-81	2850		30	7839	KING	PRESIDENT		17-NOV-81	5000		10
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	7698	BLAKE	MANAGER	7839	01-MAY-81	2850		30
7900	JAMES	CLERK	7698	03-DEC-81	950		30	7698	BLAKE	MANAGER	7839	01-MAY-81	2850		30
7902	FORD	ANALYST	7566	03-DEC-81	3000		20	7566	JONES	MANAGER	7839	02-APR-81	2975		20

```
9 rows selected.
```

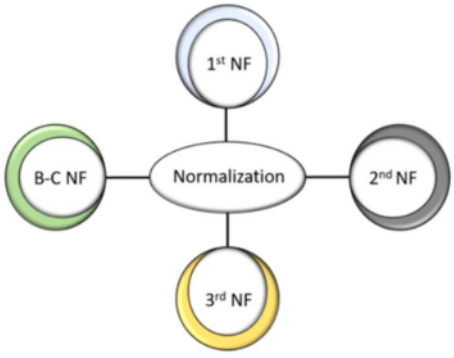
NORMALIZATION

It is the process of reducing the redundancy of data in the table and also improving the data integrity. It involves breaking down large tables into smaller, related tables to ensure efficient data management and consistency.
It usually divides a large table into smaller ones, so it is more efficient. In 1970 the First Normal Form was defined by *Edgar F Codd* and eventually, other Normal Forms were defined.

we may face many issues such as:

1) **Insertion Anomaly**:
   Occurs when you **cannot insert data into a table without adding unrelated data**.
   Example: You can't add a new department to a table unless there is at least one employee in that department.
2) **Update Anomaly**:
   Happens when **changing a value in one place requires changes in multiple places** due to redundant data.
   Example: Updating a department name in one row but forgetting to do it in others causes inconsistent data.
3) **Deletion Anomaly**:
   Occurs when **deleting data also removes other valuable data unintentionally**.
   Example: Deleting the only employee in a department also deletes information about the department itself.

An **anomaly** in databases refers to an **unexpected problem or inconsistency** that arises when inserting, updating, or deleting data. These typically occur in poorly designed databases (usually when normalization is not applied properly. These anomalies can lead to **data inconsistency**, loss of data, or the need to input unnecessary data.



Redundancy means **storing the same data more than once** in a table or database.
Redundancy can cause:

1. **Wasted storage space**
2. **DML Anomaly**

THE DIFFERENT TYPES OF NORMAL FORMS:

1) **1NF**
2) **2NF**
3) **3NF**
4) **BCNF / 3.5 NF**

1st NORMAL FORM (1NF)

In this Normal Form, we tackle the problem of atomicity. Here atomicity means values in the table should not be further divided. In simple terms, a single cell cannot hold multiple values. If a table contains a composite or multi-valued attribute, it violates the First Normal Form.

RULES FOR 1NF:

1. **Atomicity**:

   Each column must contain **atomic (indivisible)** values. No repeating groups, arrays, or multiple values in a single cell.
2. **Uniqueness of Rows**:

   Each row must be **uniquely identifiable**, typically by using a **primary key**.
3. **Single-Valued Attributes**:

   Each field should contain **only one value** per record — **no multi-valued attributes**.

EMPLOYEE_ID	NAME	PHONE_NUMBER
101	Alice	9876543210, 8765432109
102	Bob	9988776655

In the above table we can see that the table is not in 1NF, because it contains more than 1 value in a single cell which violates the 1NF rule.

EMPLOYEE_ID	NAME	PHONE_NUMBER
101	Alice	9876543210
101	Alice	8765432109
102	Bob	9988776655

The above table is atomic and is in 1NF.

2nd NORMAL FORM (2NF)

The first condition in the 2nd NF is that the table has to be in 1st NF. The table also should not contain partial dependency. Here partial dependency means the proper subset of candidate keys determines a non-prime attribute. To understand in a better way lets look at the below example.

RULES FOR 2NF:

1. Be in **First Normal Form (1NF).**

2. Have **no partial dependencies** — meaning non-prime attributes must be fully functionally dependent on the entire candidate key, not just part of it.

Candidate Key:

A **candidate key** is **a minimal set of attributes** that can **uniquely identify each row** in a table.

- A table can have **multiple candidate keys**.
- **One of them is chosen** to be the **primary key**.

Example:

For a table:

STUDENT_COURSE (StudentID, CourseID, StudentName, CourseName)

- Possible **Candidate Key**: (StudentID, CourseID) – because the combination uniquely identifies each row.

Non-prime Attribute:

- A **non-prime attribute** is an attribute that is **not part of any candidate key**.
- These are the **dependent or descriptive fields** in the table.

Example:

Using the same table:

STUDENT_COURSE (StudentID, CourseID, StudentName, CourseName)

- Candidate Key: (StudentID, CourseID)
- Non-prime attributes: StudentName, CourseName

Partial Dependency:

A **partial dependency** occurs when a **non-prime attribute** depends on **only part of a candidate key**, not the whole key.

EMPID	DEPTID	OFFICELOCATION
101	D1	New York
102	D1	New York
103	D2	Chicago

Assumed Composite Key: (EmpID, DeptID)

Now let's check for **partial dependency**.

- **OfficeLocation** depends only on DeptID, not the full composite key (EmpID, DeptID).

  ➤ This **is a partial dependency**.
- If a department changes its office location, we'll have to update multiple rows.
- Redundancy can lead to update anomalies.
- This structure violates 2NF.

Solution: Split the table into two:

EMPID	DEPTID
101	D1
102	D1
103	D2

DEPTID	OFFICELOCATION
D1	New York
D2	Chicago

Result:

- All non-prime attributes now **fully depend on the primary key** of their respective tables.
- Partial dependency is removed ⇒ **2NF achieved**

3rd NORMAL FORM (3NF)

The same rule applies as before i.e, the table has to be in 2NF before proceeding to 3NF. The other condition is there should be no transitive dependency for non-prime attributes. That means non-prime attributes (which doesn't form a candidate key) should not be dependent on other non-prime attributes in a given table. So a transitive dependency is a functional dependency in which X → Z (X determines Z) indirectly, by virtue of X → Y and Y → Z (where it is not the case that Y → X)

RULES FOR 3NF:

1. It is already in Second Normal Form (2NF).

2. There are no transitive functional dependencies of non-prime attributes on candidate keys.

Transitive Dependency

If attribute **A** determines **B**, and **B** determines **C**,

then **A indirectly determines C**.

This is a **transitive dependency**: A → B → C

In database terms:

- **A** is usually a **candidate key**
- **B** is a **non-prime attribute**
- **C** is also a **non-prime attribute**

This violates 3NF **if C depends on A through B**, not directly.

STUDENTID	NAME	SUBJECTID	SUBJECT	ADDRESS
1	Alex	101	SQL	Goa
2	Barry	102	JAVA	Bengaluru
3	Clair	103	C++	Delhi
4	David	102	JAVA	Kochi

Assume Student ID is the **Primary Key**.

The dependencies are:

- Student ID → Student Name, Subject ID, Address (directly related to student)
- Subject ID → Subject

Student ID → Subject ID → Subject

This is a **transitive dependency**, and it **violates 3NF**.

StudentID → SubjectID

SubjectID → Subject

So: StudentID → Subject (indirectly) → ❌ violates 3NF

3NF Solution: Break the table into two

1. STUDENT Table

STUDENTID	NAME	SUBJECTID	ADDRESS
1	Alex	101	Goa
2	Barry	102	Bengaluru
3	Clair	103	Delhi
4	David	102	Kochi

2. SUBJECT Table

SUBJECTID	SUBJECT
101	SQL
102	JAVA
103	C++

Result:

- No indirect (transitive) dependency.
- All non-key attributes depend **only on the primary key**.
- Now the data is in **Third Normal Form (3NF)**.

BCNF (BOYCE-CODD NORMAL FORM)

This is also known as 3.5 NF. Its the higher version 3NF and was developed by Raymond F. Boyce and Edgar F. Codd to address certain types of anomalies which were not dealt with 3NF.

Before proceeding to BCNF the table has to satisfy 3rd Normal Form.

In BCNF if every functional dependency **A → B**, then **A** has to be the **Super Key** of that particular table.

Rules for BCNF

1. **The table must be in Third Normal Form (3NF)**

2. **For every functional dependency A → B:**

   A must be a super key

   That is:

   - The left-hand side (A) **must uniquely identify** every row in the table.
   - If **A is not a super key**, the dependency violates BCNF.

- Functional Dependency (FD): A → B means if you know A, you can find B.
- Super Key: A set of attributes that can uniquely identify a row.

COURSE	INSTRUCTOR	ROOM
DBMS	John	101
OS	Mary	102
DBMS	John	103

Here,

Dependencies:

- Course → Instructor ✅
- Room → Course ❌

But **Room is not a super key**, and it determines Course ⇒ violates BCNF.

Solution for BCNF: Break it into two tables.

COURSE Table

COURSE	INSTRUCTOR
DBMS	John
OS	Mary

ROOM Table

ROOM	COURSE
101	DBMS
102	OS
103	DBMS

Now, all functional dependencies have **super keys** on the left → **BCNF achieved**

QUICK RECAP TABLE

Normal Form	Requirement
1NF	No repeating groups or arrays (atomic values)
2NF	No partial dependency (must be in 1NF)
3NF	No transitive dependency (must be in 2NF)
BCNF	Every determinant must be a **super key** (must be in 3NF)

NOTE:

1. Super Key

A **super key** is **any combination of attributes** that can uniquely identify a row in a table.

- It **may contain extra attributes** not necessary for uniqueness.
- Example: {StudentID}, {StudentID, StudentName}, {StudentID, CourseID} (if StudentID alone is enough).

2. Candidate Key

A **candidate key** is a **minimal super key**, meaning it has **no unnecessary attributes**.

- It **uniquely identifies each record**.
- There can be **multiple candidate keys** in a table.
- Example: If both Email and RollNumber uniquely identify a student, both are candidate keys.

3. Primary Key

A **primary key** is the **chosen candidate key** that acts as the **main identifier** for rows in the table.

- Only **one primary key** per table.
- It **cannot be NULL**.
- Example: Choosing RollNumber as the primary key among multiple candidate keys.

4. Alternate Key

An **alternate key** is any **candidate key that is not selected** as the primary key.

- Example: If RollNumber is the primary key, then Email becomes an alternate key.

5. Foreign Key

A **foreign key** is an attribute in one table that **refers to the primary key** of another table.

- It is used to **establish relationships** between tables.
- Example: DeptID in the EMP table referring to DeptID in the DEPT table.

6. Composite Key

A **composite key** is a **primary key made up of two or more columns**.

- Used when **no single column can uniquely identify** a record.
- Example: (StudentID, CourseID) in a Student_Course table.

7. Unique Key

A **unique key** ensures that all values in the column(s) are **unique across the table**, but unlike a primary key, **it can contain NULL values** (but only one NULL per column in many databases).

- Example: Email can be a unique key if every email is different.

SUMMARY TABLE:

Key Type	Uniquely Identifies Rows	Can Be Null	Can Be Multiple	Used for Relationships
Super Key	✅	✅	✅	❌
Candidate Key	✅	❌	✅	❌
Primary Key	✅	❌	❌	❌
Alternate Key	✅	❌	✅ (as many as needed)	❌
Foreign Key	❌ (in own table)	✅	✅	✅
Composite Key	✅	❌	✅ (type of primary key)	✅
Unique Key	✅	✅ (usually 1 NULL allowed)	✅	❌

QUERIES FOR PRACTICE

Data Query Language (DQL)

1. List employees who earn more than the average salary of all employees hired before 01-JAN-1981.
2. Display employee name, job, and hiredate for those who joined within 60 days of their manager.
3. List departments where the average salary exceeds the company's overall average salary.
4. Retrieve the names of employees whose salaries are above their department's average.
5. List employees who joined earlier than their manager and earn more than their manager.
6. Display employee details along with the department name and location for departments in NEW YORK or DALLAS.
7. Show the job title with the highest number of employees.
8. Find employees who do not share their job with anyone else.
9. List employees whose names contain exactly 5 characters and end with 'R'.
10. Display the second and third highest paid employees in department 30.

Operators

11. Find all employees whose commission is greater than 20% of their salary.
12. Display names of employees whose job is either 'SALESMAN' or 'CLERK', and who were hired in 1981.

13. Show employees who are not managers but earn more than 1500 and less than 3000.
14. List employees with the same salary as someone in another department.
15. Show names of employees where name doesn’t contain the letter 'A' and ends with 'E'.

Functions in SQL

Single Row

16. Display ENAME, number of characters in ENAME, and reversed ENAME.
17. Convert HIREDATE to the day of the week.
18. Show all employees and format their SAL with a dollar sign and comma.
19. Display EMPNO padded with zeros to 6 digits.
20. Show the month and year of hiredate in MM-YYYY format.

Multi Row

21. Get the average salary for each job in each department.
22. List the total salary and number of employees in departments where job is ‘MANAGER’.
23. For each department, find the job with the highest average salary.
24. Show the department which has the lowest total salary.
25. List each job and how much it contributes to the total payroll (as percentage).

Sub Queries

Single Row

26. Find employees earning more than the employee with the job 'CLERK' who earns the most.
27. Display the employee who has been with the company the longest.
28. Find the employee who earns the third highest salary.
29. Show employees who joined after the employee ‘FORD’.
30. List employees whose salary is greater than their manager’s salary.

Multi Row

31. Display employees whose salary is greater than any SALESMAN.
32. Show employees who work in departments located in DALLAS or CHICAGO.
33. List names of employees who share their manager with at least two others.
34. Find all employees whose salary matches another employee’s salary.
35. Display names of employees who have the same job as someone in another department.

Nested

36. List departments where all employees earn above the average company salary.
37. Display employees who are not the highest paid in their department.
38. Show employees whose salary is higher than the average of employees who have the same job.
39. List employees who earn the maximum commission in the company.
40. Find departments that have employees with multiple jobs.

Pseudo Columns

41. Display the top 3 highest paid employees using ROWNUM and subquery logic.
42. Use ROWID to find duplicate records (assume duplicates are possible).
43. Display employee with the 5th highest salary using ROWNUM.
44. Show employees along with ROWNUM after sorting them by hiredate descending.
45. Retrieve names of the first employee hired in each department using a correlated subquery and ROWNUM.

Joins

Cartesian

46. Find all department-employee combinations and filter only mismatched DEPTNOs.

Inner

47. Display employee names, job, department name, and location for employees earning more than 2500.
48. Show all employees and their department's name who joined in 1981.
49. List all employees along with their department details, but only if that department has more than 2 employees.
50. Get the department name and location for the employee with the highest salary.

Outer

51. Display all departments and their employees, including those departments with no employees.
52. Find departments that currently have no employees assigned (use LEFT OUTER JOIN).

Self-Join

53. List employees along with the name and job of their manager.
54. Display employees who have the same job as their manager.
55. Find employees who report to a manager hired before 01-JAN-1981.
56. Show the hierarchical chain: employee → manager → manager's manager.

Co-Related Sub Query

57. List employees whose salary is greater than the average salary of their department.
58. Display employees who are the only one in their department with their job title.
59. List employees whose hire date is the earliest among their department peers.
60. For each employee, show the number of colleagues in their department.
61. Show employees who have more subordinates than their own manager.
62. Display employees who do not share their salary with any other employee.
63. Find employees whose salary is less than the average salary of their job group.
64. List employees with the highest salary in their department but not in the company.

Data Definition Language (DDL)

65. Create a new table EMP_AUDIT with columns EMPNO, ENAME, and AUDIT_DATE.
66. Create a backup table EMP_BKP using the structure of EMP and copy only employees in department 30.
67. Alter EMP table to add a column LAST_UPDATED of type DATE.

68. Modify the datatype of COMM column to NUMBER(8,2).
69. Drop the column MGR from EMP.
70. Rename the DEPT table to DEPARTMENT_MASTER.
71. Truncate all data from the EMP_BKP table.
72. Drop the EMP_BKP table permanently.
73. Add a NOT NULL constraint on the SAL column.
74. Add a foreign key constraint from EMP.DEPTNO to DEPT.DEPTNO.

Data Manipulation Language (DML)

75. Insert a new employee in department 10 with job as ‘CONSULTANT’ and salary 4000.
76. Update the salary of all SALESMAN to 10% more than their current salary.
77. Delete employees who have been with the company for more than 20 years.
78. Insert into EMP a copy of all MANAGERS with 10% less salary into a test table.
79. Update the commission of all employees in department 30 to the average commission.
80. Delete employees who share their job title with less than 2 others.
81. Insert a new department with no employees.
82. Copy all CLERK details into a separate backup table.
83. Increase salary by 15% for employees who earn less than department average.
84. Update job title to ‘SENIOR CLERK’ for clerks with more than 5 years of experience.

Transaction Control Language (TCL)

85. Start a transaction, insert a record, and rollback.
86. Start a transaction, update 3 salaries, set a savepoint after 1st update, rollback to it.
87. Insert multiple rows into EMP and commit only after verifying the count.
88. Perform update, delete, insert, and rollback to see impact.
89. Commit a salary increase for managers only, while rolling back the same for analysts.

Data Control Language (DCL)

(Assume user roles exist in your environment)

90. Grant SELECT, UPDATE on EMP to user ‘DATA_ENTRY’.
91. Revoke DELETE on EMP from user ‘HR_CLERK’.
92. Grant all privileges on DEPT to user ‘DEPT_ADMIN’.
93. Revoke all access from user ‘INTERN’.
94. Grant only SELECT on EMP where DEPTNO = 30 (via view or role).
95. Create a view for READONLY_EMP and grant SELECT on it.
96. Create a role called 'HR_VIEWER' and grant SELECT on EMP, DEPT.
97. Grant INSERT, UPDATE on EMP to user ‘MANAGER’, but not DELETE.
98. Create a role ‘DATA_AUDITOR’ and assign read-only access.
99. Revoke EXECUTE on stored procedures from user ‘TEMP’.
100. Grant UPDATE on SAL to only users in role ‘PAYROLL_EDITOR’.

Advanced DQL and Filtering (Continued)

101. Find employees who were hired before their manager and earn a highet salary.
102. Display the top two highest earners from each department.
103. List employees who have the same salary as someone from another department but not the same job.

104. Show the earliest hired employee for each job title.
105. Display the total number of employees hired each year.
106. Find out which department has the most distinct job titles.
107. List the employees whose names start and end with the same letter.
108. Display employee name and number of days they have been with the company.
109. Show departments where more than 50% of employees are SALESMAN.
110. Find the most common hire month across all employees.

Nested and Correlated Logic (Advanced)

111. List employees whose salary is higher than their department's average but lower than the overall average.
112. Display employees who joined before all their department colleagues.
113. Show employees who earn more than anyone in department 10 but less than anyone in department 30.
114. List all employees who have a colleague with the exact same hiredate.
115. Show the name of the department with the highest paid employee.
116. Display employees who joined in the same year as their manager.
117. List employees who have more than one manager in the data (due to data anomalies).
118. Show all departments where the highest salary is below the company average.
119. Display employees who earn exactly the median salary (use ranking logic).
120. List managers who manage only one employee.

Joins: Cross Cases

121. Show employee name, manager name, and department name in one row.
122. List employees along with department details and also show “No Department” if DEPTNO is missing.
123. Show each employee’s peer count (colleagues in the same department).
124. Show each department and the count of jobs represented.
125. Show the average salary per department along with the department name and location.
126. Display employees who earn more than the highest paid employee in another department.
127. List departments that have all job types represented in the company.
128. Find out if any manager manages employees from more than one department.
129. Show the department with the lowest average commission.
130. Show employees who work in departments that employ fewer than 3 people.

Ranking and Analytic Patterns (via subqueries or alternatives)

131. Rank employees by salary within their department.
132. Find the 2nd highest paid SALESMAN.
133. Show all employees except the one with the lowest salary per department.
134. List the top earning job title in each department.
135. Show the oldest and newest employees per job title.
136. Identify departments where salaries vary widely (max - min > 2000).
137. List employees whose salary is closest to the department average.
138. Find departments where all employees were hired in the same year.

139. Show jobs that exist in all departments.

140. Display the difference in hiredate between each employee and their manager.

Pattern and Condition Filters

141. Show employees who joined on the last day of a month.

142. List names of employees whose names include at least 3 vowels.

143. Show all employees hired on weekends.

144. Display employees who don't have a commission value but are SALESMAN.

145. List jobs that have only one person in the company.

146. Display departments that don't have any managers.

147. List employees who are managers but also report to someone.

148. Show names where ENAME contains the substring 'LL' or 'TT'.

149. Find employees whose salary is a perfect multiple of 100.

150. List names of employees that are palindromes (e.g., 'BOB', if it existed).

DDL + Data Design Tasks (Theory + Practice)

151. Create a new table EMP_COPY with only those in department 20 or 30.

152. Add a CHECK constraint on SAL to ensure it's greater than 0.

153. Create a view showing EMPNO, ENAME, SAL, DEPTNO where DEPTNO = 10.

154. Rename column JOB in EMP to DESIGNATION.

155. Add a new column STATUS to EMP to indicate if salary > average (YES/NO).

156. Drop column COMM from EMP_COPY.

157. Create a table MANAGERS with only those employees who are managers.

158. Add a UNIQUE constraint on ENAME.

159. Alter DEPT to add a CREATED_DATE column.

160. Clone EMP and DEPT tables into EMP_2025 and DEPT_2025 respectively.

DML Scenarios Continued

161. Insert into EMP_COPY only employees whose job is either ‘CLERK’ or ‘ANALYST’.

162. Update all CLERKs' salaries to match the average clerk salary.

163. Delete all employees who joined before 01-JAN-1981.

164. Set COMM to NULL for all SALESMAN in department 30.

165. Increase salary by 20% for employees hired before 1982.

166. Copy all employees who have no manager into a separate table.

167. Insert an employee with no DEPTNO and then fix the foreign key later.

168. Update salaries of employees with common salary values (duplicates) by 5%.

169. Remove employees from departments that have fewer than 2 people.

170. Insert into EMP_AUDIT the EMPNO and ENAME of anyone who gets deleted.

TCL Scenarios Advanced

171. Start a transaction, update, savepoint, insert, rollback to savepoint.

172. Delete 2 employees, set a savepoint between, rollback one.

173. Update the same record in two sessions (simulate a lock if supported).

174. Show how commit finalizes a DELETE even if rollback comes after.

175. Use SAVEPOINT before and after a TRUNCATE (note behavior).

DCL Scenarios Continued

176. Grant INSERT, UPDATE on EMP to a role ‘HR_EDITOR’.
177. Revoke GRANT OPTION from user 'USER1'.
178. Grant SELECT on only certain columns of EMP using a view.
179. Create a view EMP_NO_COMM showing only employees with NULL COMM.
180. Grant and revoke SELECT on this view to user ‘INTERN’.

Logical Thinking / Puzzle Queries

181. Show the top 3 jobs with the highest number of employees.
182. List employees hired in a leap year.
183. Find the difference between the highest and lowest salary in each department.
184. Identify departments that don’t employ any ANALYST.
185. List departments where all employees earn above 1000.
186. Display employees who’ve stayed more than 40% of the company's age.
187. Show departments where the manager is also the highest paid.
188. Identify duplicate JOB and SAL combinations.
189. List managers who have fewer than 3 direct reports.
190. Display employee names that are also substrings of other names.

Bonus Conceptual or "Scenario-Based"

191. Write a query to display employee hierarchy (manager → employee) in one table.
192. Find the DEPT with highest avg salary but no employee hired in 1981.
193. List employees who share both job and salary with someone in another department.

194. Show the maximum gap in salary between two consecutive employees (when ordered).

195. List jobs where the number of employees is even.

196. Show employees who joined after SCOTT and earn less than FORD.

197. Find employees who are earning exactly the department's average salary.

198. Display employees grouped by job and sorted by count descending.

199. List departments where the manager's salary is lower than someone they manage.

200. List the total salary for departments that employ at least one CLERK.

ADVANCE QUERIES

Subqueries (Basic to Multi-row)

1. Find employees whose salary is equal to the **maximum salary** in the company.
2. List all employees whose salary is **greater than the average salary** of all employees.
3. Show employees who work in the **same department as 'SCOTT'**.
4. Find employees who have the **same job** as employee 'SMITH'.
5. Display employees who earn **less than the lowest salary** in department 10.
6. Find the department number of the employee who has the **highest commission**.
7. List employees whose job title exists in **more than one department**.
8. Show employees who are not in any department **with an employee named 'KING'**.
9. Display employees whose salary matches **any salary in department 20**.
10. List the names of employees who joined **on the same date as another employee**.

- **Nested Subqueries**

11. Find the **second highest salary** using nested subqueries.
12. Display employees earning the **minimum salary among those earning more than average**.
13. Show the names of departments that **contain the top 3 highest paid employees**.
14. List employees who earn more than the employee who earns **less than 'FORD' but more than 'ALLEN'**.
15. Display the name and salary of the **third highest paid employee** using nested subqueries.
16. Show employees in departments where the **average salary is below the global average salary**.
17. List all employees whose job title appears in **exactly two departments**.
18. Find the employees who earn the same as the **highest paid employee in department 30**.
19. Display employees who belong to departments where **KING doesn't work**.
20. Show names of employees whose salary is more than **average of department whose location is DALLAS**.

- **Correlated Subqueries**

21. List employees whose salary is **higher than the average salary of their own department**.
22. Show employees who earn **more than their own manager**.
23. Find employees who are the **only ones with their job in their department**.
24. List employees who have **at least one colleague in their department** earning more.
25. Display employees who are **not the highest earners in their department**.

26. Find employees whose **hiredate is the earliest among their department**.
27. List employees who **earn less than any colleague in their department**.
28. Display employees who are **hired before their manager**.
29. Show employees who have the **same job as someone earning more**.
30. List employees who **don't share their job title with anyone else in the company**.

- **Ranking with DENSE_RANK / Salary Analysis**

31. Use DENSE_RANK() to assign ranks to employees by salary.
32. Display employees who are **ranked 1st, 2nd, or 3rd in salary** across the company.
33. Rank employees **by salary within each department** using DENSE_RANK().
34. Show **department-wise second highest salary earners**.
35. List employees whose **salary rank is tied with someone in another department**.
36. Find employees with **rank 1 salary in each department** (highest paid).
37. Display employees whose salary rank is **greater than 3** in their department.
38. Find employees with the **same rank in salary as 'SCOTT'** in their department.
39. Show job titles where **at least two employees have the same salary rank**.
40. List the **top 2 paid employees per job title** using DENSE_RANK.

- **Combined Subquery + Ranking + Correlation Logic**

41. Show employees who are **not the highest or lowest earners** in their department.
42. Display employees who are ranked **second in their department salary-wise**.
43. Find employees earning **more than the average salary of departments with fewer than 3 employees**.

44. List employees whose salary is **greater than all SALESMEN** but less than any ANALYST.
45. Show the **name, job, and salary of employees** who are in departments where no one earns more than them.
46. Display employees who are **not in the top 3 paid employees overall**.
47. List employees whose salary is greater than the **minimum of other departments but not their own**.
48. Show names of employees who **earn the same as the median salary** (simulate with subqueries).
49. List employees who earn less than someone but **have the same job title**.
50. Display all employees who earn **more than the salary of their department's average + 500**.

PUZZLE ZONE – EMP & DEPT MYSTERIES

(Solve using subqueries, correlated subqueries, and analytic functions. Use paper, think hard, no peeking.)

- **1. "Better Than the Boss"**

List all employees who earn more than their own manager.

- **2. "Only Me in My Kind"**

Show employees who are the **only ones** in their department with their job title.

- **3. "The Twin Salary Paradox"**

Find all employees who **share the same salary** with at least one person in another department.

- **4. "Missing Middle"**

List employees who are **neither the highest nor the lowest** earners in their department.

- **5. "RANK of Shadows"**

Find the **second highest** paid employee in each department **using DENSE_RANK()**.

- **6. "Clash of the Titans"**

Show the names of the **two highest-paid employees in the company**, even if they earn the same.

- **7. “Mirror Manager”**

List employees who **joined before their manager** and earn more than them.

- **8. “No One Like Me”**

Find employees who have **no peers** in the company with the same job and salary.

- **9. “Time Capsule”**

Who is the **earliest hired employee** per job?

- **10. “Kingless Realms”**

Show departments that have **no employees managed by KING**.

- **11. “Job Jumper”**

Which job title exists in **more than 2 departments**?

- **12. “Hirestorm”**

Find months with the **highest number of hires**.

- **13. “The Equalizer”**

List employees whose salary is **equal to the average salary of someone else's department**.

- **14. “Rank Jumpers”**

Who are the **top 3 highest-paid employees** across all departments, skipping ties?

- **15. “Not In My Backyard”**

Show employees whose salary **does not match** anyone else in the same department.

- **16. “Manager Lite”**

List managers who **manage only one person**.

- **17. “Ghost Dept”**

Identify departments that have **no employees assigned**.

- **18. “Perfect Pair”**

Find **employee pairs** who have the **same job, same salary**, but **work in different departments**.

- **19. “Level Field”**

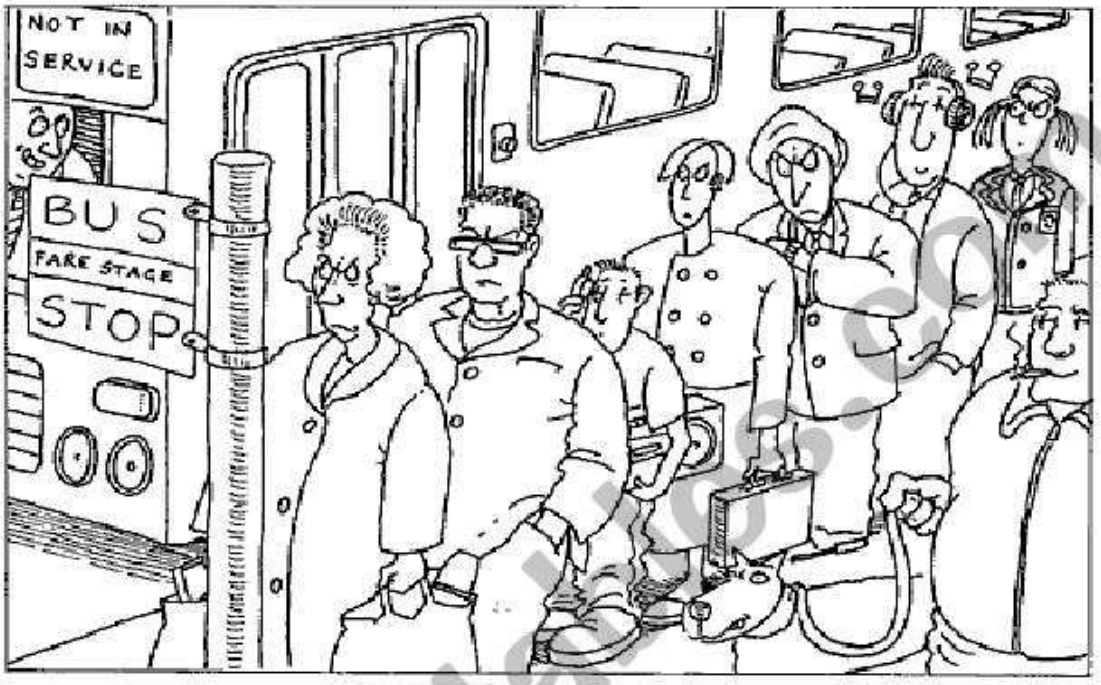
Who are the **lowest paid employees** in each job category?

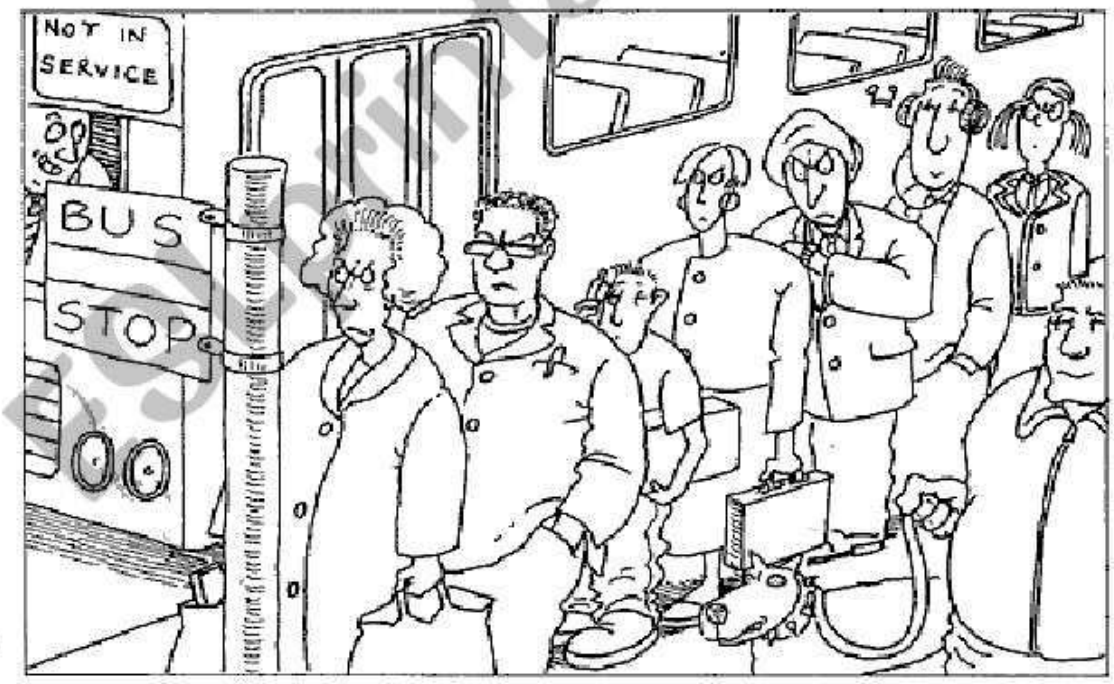
- **20. “Inner Circle”**

List employees whose salaries are **between the min and max of their department**, but not equal to either.

Spot the differences

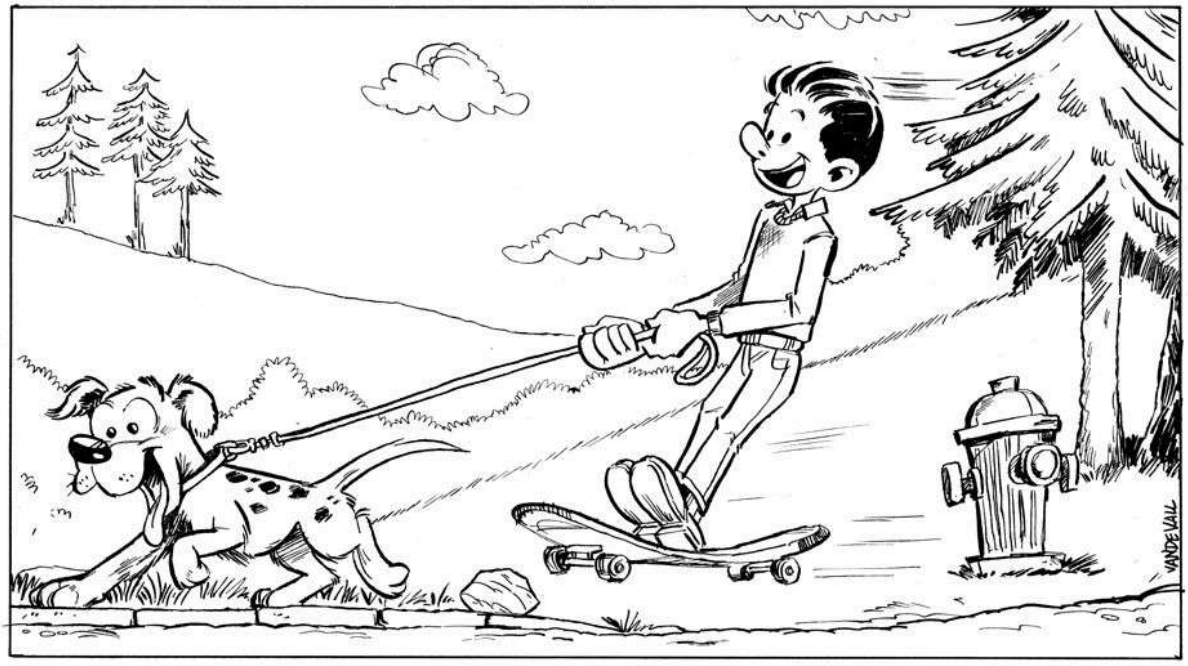
Compare the two pictures and describe the differences between them.

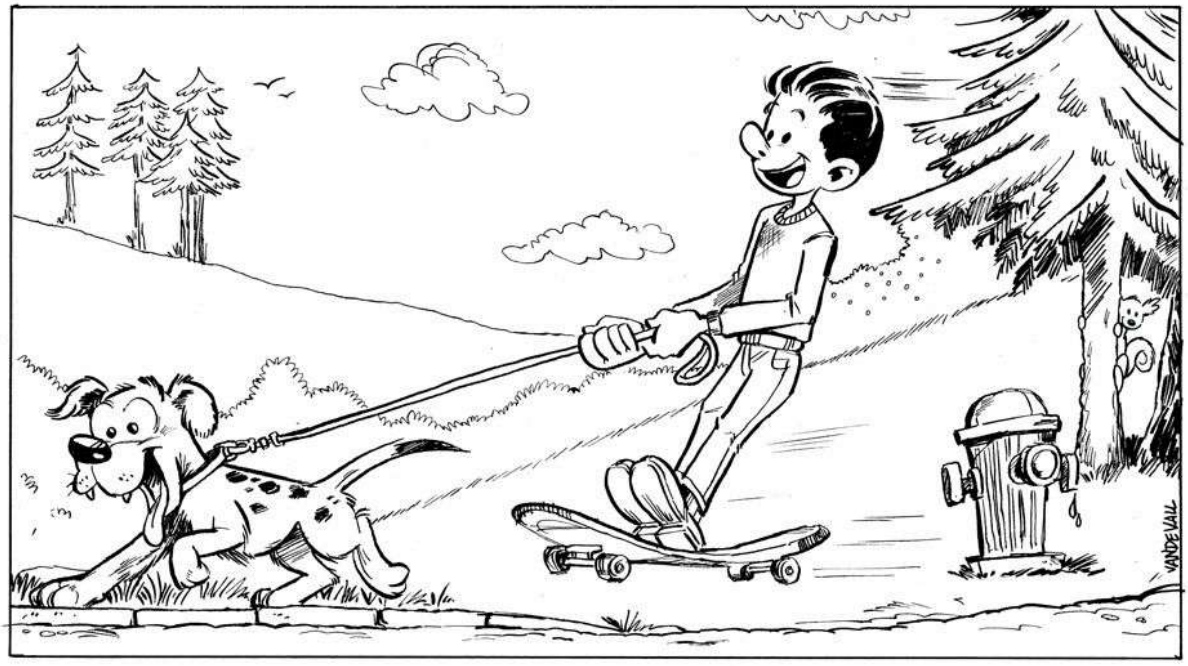




Spot the Difference!

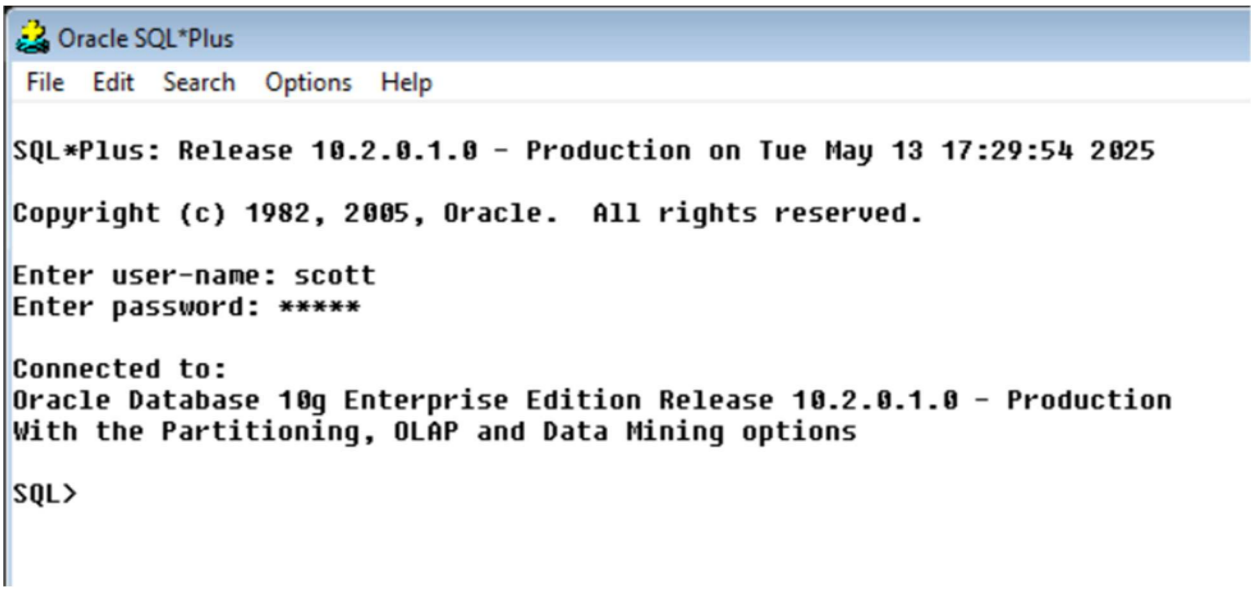
Find 10 differences between the 2 pictures!





SOFTWARE USED

Oracle SQL 10g



MySQL Workbench