13-3-24 Wednesday.

Week - 1

Execute all DDL, DML, DCL Commands:

SQL Commands:

- create: It is used to create tables in Database.

  Syntax:

  Create table tablename (column1 datatype1, column2 datatype2, column n datatypen);

  Example for creation of student table:

  Create table students (rollno number(10), name varchar (20), department varchar(20));

  Table Created

- desc: desc command is used to describe or display the structure of table.

  Ex: desc students;

Name	Type
ROLL NO	NUMBER (10)
NAME	VARCHAR 2(20)
DEPARTMENT	VARCHAR 2(20)

Similarly, we will create tables for Emp & dept.

→ CREATE TABLE Emp(EmpNO NUMBER(4), Empname VARCHAR(20), Job VARCHAR 2(10), DATE, DOB DATE, Salary NUMBER(8), COMM

```
NUMBER(5), DeptNo NUMBER(4));
Table created.
```

→ Create table Dept (deptno number(4),
dname varchar2(10));

Table created.

```
desc Emp;

Name                              Null?    Type
------------------------------------------------
EMPNO                                      NUMBER(4)
EMPNAME                                    VARCHAR2(20)
JOB                                        VARCHAR2(20)
DOJ                                        DATE
DOB                                        DATE
SALARY                                     NUMBER(8)
COMM                                       NUMBER(5)
DEPTNO                                     NUMBER(4)
```

- Insert : It is used to insert new records into the table.

It has two syntaxes :-

Syntax:

```
Insert into tablename (col 1, col 2, ---- coln)
                values (val 1, val 2, --- val n);
```

Inserting values into all columns.

```
Insert into <table name> values (value 1, value 2,
                --- value n);
```

Example for inserting values into Emp table:

```
INSERT INTO Emp values(101, 'Raju', 'Manager',
   '17-MAR-22', '21-OCT-99', 5000, 500, 24);
1 row created
INSERT INTO Emp values (102, 'Niha', 'Manager',
   '18-MAR-22', '22-OCT-99', 4000, 500, 22);
1 row created
```

Example for inserting values into dept table:

```
INSERT INTO dept values ( 56', 'mca' );
1 row created
INSERT INTO dept values (56, 'mba');
1 row created
INSERT INTO dept values (56, 'Msc');
1 row created
```

- Select :- Select command is used to display the selected rows from the table.

  * :- * symbol gives all values in table.

  Syntax1 : Select * From Emp ;

```
                                        DOJ
EMPNO   ENAME   JOB      EXP   HIREDATE    DOB        SALARY  COMM  DEPT
                                                                     NO
101     Raju    Manager        17-MAR-22   21-OCT-99  5000    500    24
102     Niha    Manager        18-MAR-22   22-OCT-99  4000    500    22
```

dept

Select * From dept;

```
deptno    dname
  56       mba
  57       mca
```

Syntax 2 : Select Columnname from tablename;

Ex: Select EmpNo From Emp;

```
EMPNO
  16
  17
```

Week - 2 20-3-24

Wednesday.

- Alter : Alter command is used to add a new column and also used to modify the existing column to new name.

Syntax 1 :

```
Alter table tablename add colname datatype;
```

Ex : Alter table Emp1 add Emp Number(4);

Syntax 2 :

```
Alter table tablename rename column
                    col_name to new col_name;
```

Ex : Alter table Emp1 rename column Emp-name
                    to Ename;

- Update : Update Command is used to set a value to particular column.

Syntax :

```
Update tablename set col_name1 = value 1
                    where col_name2 = value 2;
```

Ex : Update Emp1 set Exp = 22 where EmpNo = 1;

- Delete : delete command is used to delete particular row of a table.

Syntax :

```
Delete from table name where condition;
```

Ex : delete from Emp1 where EmpNo = 1;

- Truncate : Truncate command is used to delete all rows in the table.
  Syntax : Truncate table table_name;
- Drop : Drop command is used to delete entire structure as well as all rows.
  Syntax :
  Drop table table_name;

DCL :

- Grant : It is used to give user access privileges to a database.

  Ex : Grant select, update on my_table
  to some_user, Another_user;

- Revoke : It is used to take back permission from user.

Ex : Revoke select, update on my_table from
user1, user2;

[illegible]

Week - 3

3 - 4 - 24
Wednesday

Implementation of different types of operators and built-in functions.

- Operators:-
- Arithematic Operators:

Operators:

+ - * / %.

1) SELECT 30 + 20 ;
 50

2) SELECT 60 - 20 ;
 40

3) SELECT 6 * 3 ;
 18

4) SELECT 10/ 5 ;
 2

5) SELECT 18 % 6 ;
 0

- Comparision Operators :

-- Equal to

Syntax : SELECT * FROM Emp Where EmpNO = 107;

Ex :

EMPNO	EMPNAME	JOB	DOJ	DOB	SALARY
107	Nihorika	Manager	22-MAR-22	26-OCT-99	5000

-- Not equal to

Syntax : SELECT * FROM Emp where EmpNO<>105;

Ex:

EMPNO	EMPNAME	JOB	DOJ	DOB	SALARY
101	Raju	Manager	17-MAR-22	21-OCT-99	5000
102	Niha	Manager	18-MAR-22	22-OCT-99	4000
103	Jyo	Manager	19-MAR-22	23-OCT-99	6000
104	Hima	Manager	20-MAR-22	24-OCT-99	3000
106	Hari	Manager	21-MAR-22	25-OCT-99	4000

-- Greater than

Syntax: SELECT * FROM Emp where DEPT>22;

Ex:

EMPNO	EMPNAME	JOB	DOJ	DOB	SALARY
101	Raju	Manager	17-MAR-22	21-OCT-99	5000

COMM	DEPT
500	24

-- Less than

Syntax: SELECT * FROM Emp where DEPT <21;

Ex:

EMPNO	EMPNAME	JOB	DOJ	DOB	SALARY
104	Hima	Manager	20-MAR-22	24-OCT-99	3000

COMM	DEPT
500	20

- Logical Operator :

-- AND

Syntax : SELECT * FROM Emp where EmpNO = 102 AND EmpNAME = 'B';

Ex :

EmpNO	EmpNAME	JOB	DOJ	DOB	SALARY
102	Niha	Manager	18-MAR-22	22-OCT-99	4000

-- OR

Syntax : SELECT * FROM Emp where EmpNO = 106 OR DEPT = 24 ;

Ex :

EmpNO	EmpNAME	JOB	DOJ	DOB	SALARY	COMM	DEPT
101	Raju	Manager	17-MAR-22	21-OCT-99	5000	500	24
106	Hari	Manager	21-MAR-22	25-OCT-99	4000	500	22

-- NOT

Syntax : SELECT * FROM DEPT where Not DEPT NO = 101 ;

Ex :

DEPT NO	DEPTNAME
102	Manager
103	Manager
104	Manager

4. String Function :

-- CONCAT

Syntax : SELECT CONCAT (first-name, ' ', last-name AS Full_name from Emp;

-- SUBSTRING :

Syntax : SELECT SUBSTRING (first- name,1,3) AS Initials from Emp;

Ex : Select DEPTNAME, SUBSTR(DEPTNAME,1,3) AS DEPTNAME SHORT from DEPT;

-- UPPER

Syntax: Select Upper (DEPTNAME) AS DEPTNAME - UPPER from DEPT;

Ex : DEPTNAME_UPPER

MANAGER

-- LOWER

Syntax: Select Lower (DEPTNAME) AS DEPTNAME - LOWER from DEPT;

Ex : DEPTNAME_ LOWER

Manager

5. Mathematical Function.

--ABS

Syntax : Select ABS(-10) AS Absolute-value from DEPT.

Ex : ABSOLUTE_VALUE

```
ABSOLUTE_VALUE
--------------
    10
    10
    10
```

-- SQRT

Syntax : Select SQRT(36) AS Square_root from DEPT;

Ex :
```
SQUARE_ROOT
-----------
     6
     6
     6
```

-- POWER

Syntax : Select POWER(2,3) AS power from DEPT;

Ex :
```
POWER
-----
  8
  8
  8
```

-- ROUND

Syntax : Select ROUND(3.14159, 2) AS Rounded_pi from DEPT;

Ex :
```
ROUNDED_PI
----------
  3.14
  3.14
  3.14
```

6. Aggregate Functions :

-- COUNT

Syntax : Select COUNT(*) AS Dept_name from DEPT;

Ex :
```
DEPT_NAME
---------
    10
```

-- SUM

Syntax: Select sum(SALARY) As Total_salary from DEPT;

Ex:

```
TOTAL_SALARY
------------
 44000
```

-- Avg

Syntax: select Avg(SALARY) As Avg-salary from DEPT;

Ex:

```
AVG_SALARY
----------
   4400
```

-- MAX

Syntax: Select Max(Salary As Max_salary from DEPT;

Ex:

```
MAX_SALARY
----------
 50000
```

Week - 8

* Control Structure :

1) Write a PL/SQL block for the Addition of the Two numbers.

```
SET SERVEROUTPUT ON;
DECLARE
   num1  NUMBER = 10;
   num2   NUMBER = 20;
   SUM   NUMBER ;
   BEGIN
     Sum = num1 + num2 ;
    DBMS_OUTPUT. PUT_LINE ('The sum of'||
              num1|| 'and' || num2|| 'is:"||sum);
    END ;
    /
```

Output :

```
The sum of 10 and 20 is : 30
```

2) Write a PL/SQL block for IF, IF and else condition.

```
SET SERVEROUTPUT ON;
DECLARE
  num1  NUMBER = 10;
  num2  NUMBER = 20;
BEGIN
 -- IF condition
  IF num1 >num2 THEN
   DBMS_OUTPUT. PUT LINE (num1 || 'is greater
                                than' || num2);
```

```
END IF;
-- IF- EISE Condition
IF  num1 < num2  THEN
 DBMS_ OUTPUT. PUT_ LINE (num1 ||' is not less
                than' || num2);
 END IF;
 END;
 /
```

Output :

10 is less than 20

3) Write a PL/SQL block for implementation of loops.

```
 SET SERVEROUTPUT ON;
 DECLARE
     i NUMBER = 1;
 BEGIN
  -- WHILE LOOP
  WHILE i<=5 LOOP
     DBMS_OUTPUT.PUT_LINE('Value of i:' ||i);
     i = i+1;
    END LOOP;
    -- FOR loop
    FOR j IN 1...5 LOOP
    DBMS_ OUTPUT. PUT LINE ('Value of j:'||j);
    END LOOP;
    END;
    /
```

OUTPUT:

```
value of i : 1
value of i : 2
value of i : 3
value of i : 4
value of i : 5
value of j : 1
value of j : 2
value of j : 3
value of j : 4
value of j : 5
```

4) Write a PL/SQL block for greatest of 3 numbers using IF and ELSE IF.

```
SET SERVEROUTPUT ON;
DECLARE
   num1 NUMBER = 10;
   num2 NUMBER = 20;
   num3 NUMBER = 15;
   greatest NUMBER;
BEGIN
 IF num1 >= num2 AND num1>=num3 THEN
  ~~THEN~~ greatest= num1;
ELSE IF num2>=num1 AND num2>=num3 THEN
greatest = num2;
 ELSE
  greatest = num3;
  END IF;
  DBMS_OUTPUT.PUT_LINE ('The greatest
      number among'|| num1||','||num2||',and'||
      num3|| 'is:' || greatest);
END;
```

Output :

```
The greatest number using among
10, 20 and 15 is : 20
```

[illegible]

Week - 9

Exception Handling - Implement the following with respect to exception handling. Raising Exception, User Defined Exceptions, Pre - Defined Exceptions.

1. Raising Exception :

```
DECLARE
  V-num1 NUMBER := 10;
  V-num2 NUMBER := 0;
BEGIN
  IF V-num2 = 0 THEN
    RAISE ZERO_DIVIDE ;
  ELSE
    DBMS_OUTPUT.PUT LINE('Result:'|| v_num1
                                      /v_num2);
  END IF ;
EXCEPTION
  WHEN ZERO_DIVIDE THEN
    DBMS_OUTPUT.PUT LINE ('Error: Division
                                      by zero');
  END ;
  /
```

Output : Error : Division by Zero

Week - 3 10-4-24
Wednesday.

Implementation of different types of joins with examples:

Join: Join means to combine the records from two or more tables in database.

Types Of SQL Join:

1) INNER JOIN:

It selects records that have matching values in both tables as long as the condition is satisfied.

Syntax:

```
Select Emp. EmpNAME, project. DEPARTMENT
 from Emp
 Inner Join Project
 On Project. EmpNO = Emp. EmpNO;
```

Ex:

EMPNAME	DEPARTMENT
Jyothi	Development
Hari	IT
Niha	Non - IT

2) LEFT JOIN:

It returns all values from left table and matching values from right table.

Syntax :

```
Select Emp. EmpNAME, Project.
DEPARTMENT from Emp
Left join project
On project.EmpNo = Emp.EmpNO;
```

Ex:

EmpNAME	DEPARTMENT
Jyothi	Development
Hari	IT
Niha	Non-IT
Hima	
Deepu	

3) RIGHT JOIN:

It returns all the values from right table and matched values from left table.

Syntax :

```
Select Emp. EmpNAME, project.
DEPARTMENT. from Emp
Right join project
On project. EmpNO = Emp. EmpNO;
```

Ex:

EmpNAME	DEPATMENT
Jyothi	Development
Hari	IT
Niha	Non-IT

4) FULL JOIN:

It is the result of the combination of both left & right outer join.

syntax:

```
Select Emp. EmpNAME, project, DEPARTMENT
from Emp
Full join project
On project. EmpNO = Emp. EmpNO ;
```

Ex:

EmpNAME	DEPARTMENT
Jyothi	Development
Hari	IT
Niha	Non-IT
Siri	NULL
Sai	NULL

5) CROSS JOIN:

It returns all matching records from both tables whether the other table matches or not.

Syntax:

```
Select Emp. EmpNAME, Project. project_No
from EMP
cross join project ;
```

Ex:

EMPNAME	DEPARTM- PROJECT_NO
Jyothi	1
Hari	1
Niha	1
Deepu	1
Siri	1
Sai	1
Jyothi	2
Hari	2
Niha	2
Deepu	2
Siri	2
Sai	2
Jyothi	3
Hari	3
Niha	3
Deepu	3
Siri	3
Sai	3
Jyothi	4
Hari	4
Niha	4
Deepu	4
Siri	4
Sai	4