Introduction Oracle SQL and PL/SQL

Module 1/Day 1

SELECT Statement:

1. **List tables in your schema and check for existence of DEPT, EMP and SALGERADE tables**

SELECT TABLE_NAME FREOM USER_TABLES;--If we have admin access

SELECT OWNER,TABLE_NAME FROM ALL_TABLES;

SELECT * FROM SCOTT.DEPT; SELECT * FROM SCOTT.EMP; SELECT *FROM SCOTT. SALGERADE;

2. **If these tables do NOT exists – execute the script in the embedded DemoBld.SQL file to create and populate the tables .**

   Executed given script.

3. **List all columns and all rows from DEPT**

SELECT * FROM SCOTT.DEPT;

4. **List all columns and all rows from EMP**

   SELECT * FROM SCOTT.EMP;

5. **List all columns and all rows from SALGRADE**

   SELECT *FROM SCOTT. SALGERADE;

6. **List employee number, name and salary from employee table**

   SELECT EMPNO,ENAME,SAL FROM SCOTT.EMP;

7. **List employee number, name and salary from employee table where salary is > 3000**

SELECT EMPNO,ENAME,SAL FROM SCOTT.EMP

WHERE SAL>3000;

8. **List employees joined after year 1981**

SELECT * FROM SCOTT.EMP e

WHERE e.HIREDATE > '31-DEC-1981'

9. **List all clerks (JOB = 'CLERK')**

SELECT * FROM SCOTT.EMP e WHERE e.JOB='CLERK' ;

10. **List employees in the ascending order of salary**

SELECT * FROM SCOTT.EMP ORDER BY SAL ;

11. **List employees in ascending order of job within descending order of deptno**

 SELECT * FROM SCOTT.EMP ORDER BY JOB , DEPTNO DESC

12. **List distinct departments from employee table**

SELECT DISTINCT DEPTNO FROM SCOTT.EMP;

13. **List distinct jobs in each department from employee table**

SELECT DISTINCT JOBS FROM SCOTT.EMP GROUP BY DEPTNO

14. **List name, salary and annual salary in the descending order of annual salary – annual salary is a computed column – SAL * 12**

    SELECT ENAME,SAL. 12*SAL AS ANNUAL _SALARY FROM SCOTT.EMP ORDER BY

    ANNUAL _SALARY DESC;

15. **List employees whose salary is not in the range of 2000 and 3000**

    SELECT * FROM SCOTT.EMP

    WHERE SAL NOT IN(2000,3000);

16. **List name and the deptno for all employees who are NOT members of departments 10 and 20**

SELECT ENAME,DEPTNO FROM SCOTT.EMP

WHERE DEPTNO NOT IN(10,20);

17. **List employees for whom COMM is not applicable**

SELECT * FROM SCOTT.EMP

WHERE COMM IS NULL;

18. **List employees for whom COMM is applicable**

SELECT * FROM SCOTT.EMP

WHERE COMM IS NOT NULL;

19. **List employees in ascending order of COMM and note how NULLs are sorted**

SELECT * FROM SCOTT.EMP

WHERE COMM IS NOT NULL GROUP BY COMM ;

20. **List employees whose names start with ‘’SMITH’**

SELECT * FROM SCOTT.EMP

WHERE ENAME LIKE 'SMITH%' ;

21. **List employees whose name contain the ‘MI’**

SELECT * FROM SCOTT.EMP

WHERE ENAME LIKE '%MI%';

22. **List employees whose name start with an _ (underscore) char.**

    SELECT * FROM SCOTT.EMP

    WHERE ENAME LIKE '_%';

23. **List all employees joined between two given dates.**

SELECT * FROM SCOTT.EMP WHERE HIREDATE BETWEEN TO_DATE (Date_1 ,’dd-mm-yyy’) AND TO_DATE (Date_2 ,’dd-mm-yyyy’);

24. **List all clerks in deptno 10**

    SELECT * FROM SCOTT.EMP

    WHERE JOB = 'CLERK' AND DEPTNO =10;

25. **List total/sum, maximum, minimum, average of salary from employee table**

    SELECT ENAME,EMPNO,JOB,MGR,HIREDATE,COMM,DEPTNO,SUM(SAL) AS TOTALSAL,MAX(SAL)

    AS MAXSALARY,MIN(SAL) MINSAL,AVG(SAL) AVGSAL FROM SCOTT.EMP;

26. **List average and count of commission of all employees in department 10**

    SELECT AVG(COMM) AVG,COUNT(COMM) FROM SCOTT.EMP

    WHERE DEPTNO=10;

27. **List department wise no of employees and total salary**

    SELECT DEPTNO,SUM(SAL) TOTALSALARY FROM SCOTT.EMP

    GROUP BY DEPTNO;

28. **List total salary Job wise within each department**

SELECT DEPTNO,JOB,SUM(SAL) TOTALSALARY FROM SCOTT.EMP

GROUP BY DEPTNO;

29. **List department wise total salary for deptno 10 and 20 only**

    SELECT DEOTNO,SUM(SAL) TOTALSALARY FROM SCOTT.EMP

    WHERE DEPT IN(10,20)

    GROUP BY DEPTNO;

30. **List department wise total salary where total salary is > 6000**

    SELECT DEPTNO,SUM(SAL) TOTALSAL FROM SCOTT.EMP

    WHERE SUM(SAL) > 6000

    GROUP BY DEPTNO;

31. **SELECT COUNT(*), COUNT(COMM) FROM EMP; - explain why the two counts are different**

COUNT(*) : Gives total number of rows present in Emp table.

COUNT(COMM): Ii gives count of total number of non null comm rows in EMP table.

Sub Query:

1. **List employees whose job is same as that of 'SMITH'**

        SELECT * FROM EMP

    WHERE JOB = (SELECT JOB FROM EMP WHERE ENAME='SMITH')

2. **List employees who have joined after 'ADAM'**

    SELECT * FROM EMP

WHERE HIREDATE = (SELECT HIREDATE FROM EMP WHERE ENAME='ADAM')

3. **List employees who salary is greater than 'SCOTT's salary**

   SELECT * FROM EMP

   WHERE SAL> (SELECT SAL FROM EMP WHERE ENAME='SCOTT')

4. **List employees getting the maximum salary**

   SELECT *FROM EMP ORDER BY SAL DESC

5. **List employees show salary is > the max salary of all employees in deptno 30**

   SELECT * FROM EMP

   WHERE SAL> (SELECT MAX(SAL) FROM EMP WHERE DEPTNO=30)

6. **List all employees whose deptno and Job are same as that of employee with empno 7788.**

   SELECT * FROM EMP WHERE DEPTNO (SELECT DEPTNO FROM EMP WHERE EMPNO=7788)

   AND JOB= (SELECT JOB FROM EMP WHERE EMPNO=7788)

7. **List employee who are not managers**

   SELECT * FROM EMP WHERE JOB NOT IN 'MANAGER'

8. **List all managers**

   SELECT * FROM EMP WHERE JOB='MANAGER'

9. **List all employees who earn(salary) more than the average salary in their own department**

   SELECT * FROM EMP WHERE SAL> (SELECT AVG(SAL) FROM EMP GROUP BY DEPTNO)

10. **List employees whose salary is greater than their manager's salary**

    SELECT * FROM EMP WHERE SAL>(SELECT SAL FROM EMP WHERE JOB='MANAGER')

11. **List details of departments from DEPT table for which there are no employees in EMP table**

    SELECT * FROM DEPT WHERE DEPTNO NOT IN(SELECT DEPTNO FROM EMP)

Module 2/Day 2

Joins:

1. **List employee name, department number and their corresponding department name by joining EMP and DEPT tables**

   SELECT e.ENAME, E.DEPTNO,d. DNAME

   FROM EMPLOYEE E,DEPT d

   WHERE e.DEPTNO=d.DEPTNO;

2. **List employee name and their manager name by joining EMP table to itself**

   Select e.ENAME AS EMP_NAME,e.EMPID AS EMPID

   m.ENAME AS MANAGER_NAME,m.EMPID AS MANAGERID

   FROM EMPLOYEE e JOIN EMPLOYEE m ON e.EMPID=m.EMPID

3. **List employee name, department name and their grade by joining EMP, DEPT and SALGRADE tables**

   Where SALGRADE neither have reference keys like foreign key with EMP and DEPT tables

4. **List employees who work in 'Research' department by joining EMP and DEPT tables**

```
SELECT E.EMPNAME,E.EMPID,D.DEPTNO,D.DNAME FROM
EMPLOYEE E,DEPT D
WHERE  E.DEPTID=D.DEPTID AND D.DNAME='RESEARCH';
```

5. **List all rows from EMP table and only the matching rows from DEPT table – LEFT OUTER JOIN**

```
SELECT * FROM EMP LEFT JOIN DEPT ON EMP.DEPTNO = DEPT.DEPTNO
```

6. **List only matching rows from EMP table and all rows from DEPT table – RIGTH OUTER JOIN**

```
SELECT * FROM EMP RIGHT JOIN DEPT ON EMP.DEPTNO = DEPT.DEPTNO
```

7. **Write a query to perform full outer join between EMP and DEPT tables**

```
SELECT * FROM EMP FULL OUTER JOIN DEPT ON EMP.DEPTNO = DEPT.DEPTNO
```

8. **List employee name, their manager name and their manager's manager name**

```
SELECT DISTINT E.ENAME AS EMPLOYEE,M.MGR AS REPORTS_TO,M.ENAME AS MANAGER
FROM EMPLOYEE E
INNER JOIN EMPLOYEE M ON E.MGR=M.EMPNO;
```

DDL

1. **Create DEPARTMENT table with the following columns with appropriate data type and width**
   a) **Deptno PK**
   b) **Danme**
   c) **Location**

```
CREATE TABLE DEPARTMENT (
DEPTNO          NUMBER(2) NOT NULL,
DNAME           VARCHAR2(14),
 LOC            VARCHAR2(13),
CONSTRAINT DEPT_PRIMARY_KEY PRIMARY KEY (DEPTNO));
```

2. **Create EMPLOYEE table with the following columns with appropriate data type and width**
    a. **empno PK**
    b. **ename not null**
    c. **designation**
    d. **sex**
    e. **basic_salary (> 0 and < 500000)**
    f. **Date of joining**
    g. **Deptno reference deptno of DEPARTMENT table**

```
CREATE TABLE EMPLOYEE (
 EMPNO            NUMBER(4) NOT NULL,
 ENAME            VARCHAR2(10)  NOT NULL ,
 DESIGNATION            VARCHAR2(9),
SEX               VARCHAR(5),
BASIC_SALARY   NUMBER(7) CHECK (BASIC_SALARY>0 AND BASIC_SALARY <500000)
 DATEOFJOINING        DATE,
 DEPTNO           NUMBER(2) NOT NULL,
 CONSTRAINT EMP_FOREIGN_KEY FOREIGN KEY (DEPTNO) REFERENCES DEPT (DEPTNO),
 CONSTRAINT EMP_PRIMARY_KEY PRIMARY KEY (EMPNO));
```

3. **Alter table EMPLOYEE add column commission**

```
ALTER TABLE  EMPLOYEE ADD  COMMISSION NUMBER(7,2);
```

4. **Alter table EMPLOYEE add constraint SEX in (‘M’, ‘F’)**

```
ALTER TABLE   EMPLOYEE
 ADD CONSTRAINT  CHECK _CONSTRAINT CHECK(SEX IN ('M','F');
```

4. **Create Index on ename column of EMPLOYEE table**

```
CREATE INDEX ENAME_INDEX
ON EMPLOYEE(ENAME);
```

5. **Create exact replica of EMPLOYEE table with no data**

```
CREATE TABLE EMP AS SELECT * FROM EMPLOYEE WHERE 1=0;
```

6. Create new table called EX_EMP with columns empno, ename, basic_salary and populate the data from EMPLOYEE table

CREATE TABLE EX_EMP AS SELECT EMPNO, ENAME,BASIC_SALARY;

7. Drop the Index created on ename column.

DROP INDEX ENAME_INDEX ON EMPLOYEE;

DML

1. Insert at least 5 valid rows into DEPARTMENT table and commit the changes

INSERT INTO DEPT VALUES (10,'ACCOUNTING','NEW YORK');

INSERT INTO DEPT VALUES (20,'RESEARCH','DALLAS');

INSERT INTO DEPT VALUES (30,'SALES','CHICAGO');

INSERT INTO DEPT VALUES (40,'OPERATIONS','BOSTON');

INSERT INTO DEPT VALUES (50,'INFRA','NEWCASTLE');

COMMIT;

2. Insert at least 15 valid rows in EMPLOYEE table and commit the changes

INSERT INTO EMPLOYEE(111,'Mahesh','DEVELOPER','F',22000,'15-AUG-2015',10);

INSERT INTO EMPLOYEE(112,'Raju','DEVELOPER','F',22000,'16-AUG-2015',90);

INSERT INTO EMPLOYEE(113,'Raja','ENTERPRENIOUR','M',25000,'15-SEPT-2015',20);

INSERT INTO EMPLOYEE(114,'Rani','TESTER','F',22000,'15-AUG-2015',100);

INSERT INTO EMPLOYEE(1114,'LEELU','DEVELOPER','F',22000,'15-AUG-2013',60);

INSERT INTO EMPLOYEE(1115,'BANDHAVI','DEVELOPER','F',22000,'15-AUG-2015',10);

INSERT INTO EMPLOYEE(1126,'SRI','DEVELOPER','F',22000,'16-AUG-2015',60);

INSERT INTO EMPLOYEE(1137,'MANI','ENTERPRENIOUR','M',25000,'15-SEPT-2015',20);

INSERT INTO EMPLOYEE(1118,'SRINIKA','TESTER','F',22000,'15-AUG-2015',10);

INSERT INTO EMPLOYEE(1119,'LEELA','DEVELOPER','F',22000,'15-AUG-2013',60);

INSERT INTO EMPLOYEE(1011,'Raja','DEVELOPER','F',22000,'15-AUG-2015',10);

INSERT INTO EMPLOYEE(1012,'SRIl','DEVELOPER','F',22000,'16-AUG-2015',20);

INSERT INTO EMPLOYEE(1013,'MANI','ENTERPRENIOUR','M',25000,'15-SEPT-2015',20);

INSERT INTO EMPLOYEE(1011,'SRI','TESTER','F',22000,'15-AUG-2015',90);

INSERT INTO EMPLOYEE(1011,'ROja','DEVELOPER','F',22000,'15-AUG-2013',60);

Commit;

3. **Update basic_salary by 10% for employees in deptno 10 and 20 and commit the changes**

   UPDATE EMPLOYEE

   SET BASIC_SALARY=(0.1* BASIC_SALARY)

   WHERE DEPTID IN(10,20);

   COMMIT;

4. **Update basic_salary and commission by 10% and 2% for all employees for whom commission is currently applicable and commit the changes**

   UPDATE EMPLOYEE

   SET BASIC_SALARY=(0.1* BASIC_SALARY),COMMISION=(0.02* COMMISSION);

   COMMIT;

5. **Update the designation of given employee to MANAGER based of given employee number and commit the changes.**

   UPDATE EMPLOYEE

   SET DESIGNATION=' MANAGER' WHERE EMPNO=111;

   COMMIT;

6. **Delete employees joined before a given year and commit changes**

   DELETE EMPLOYEE WHERE DATEOFJOINING< TO_CHAR(DATEOFJOINING,'DD-MMM-YY')

7. **Delete all rows from employee tables.**

   DELETE FROM EMPLOYEE;

8. **Query employee table**

   SELECT * FROM EMPLOYEE;

9. **ROLLBACK**

   ROLLBACK;

10. **Query employee table**

    SELECT * FROM EMPLOYEE;

11. **Delete all rows from employee tables permanently using appropriate DDL command**

    DELETE FROM EMPLOYEE;

    COMMIT;

Procedure and Functions:

1. Create a procedure named DISP_EMP_DETAILS with 3 parameters → one [ie. iEmpNo] is an "IN" mode and other two are [ie. sGrade and sSalary] "OUT" mode parameters. The procedure should retrieve the Grade (this is

column of SALGRADE table) and Salary for the specified employee number [ie. iEmpNo] by joining EMP and SALGRADE table and assign the retrieved values to the "OUT" mode parameters.

Note.:

It should display an appropriate error message if the specified employee number

does not exist in "Employee" table.

Call the created procedure using Bind variable and print the details.

2. Create a procedure named DISPLAY_RECORDS which accepts the P_JOB as a parameter and display all the employees (empno, sal, deptno, job) from the "EMP" table matching the given P_JOB in the following format :-

EmployeeNumber	Salary	DepartmentNumber	Job
XXXXXXXX	99,999	99	CLERK
XXXXXXXX	9,999	12	CLERK

Note.:

It should display an appropriate error message if there are no employees with the given JOB

```
PROCEDURE DISPLAY_RECORDS (
         P_JOB IN NUMBER,
         EMPNO  OUT NUMBER,
         SAL OUT NUMBER,
         DEPNO IN NUMBER
        JOB OUT VARCHAR2(20))

 IS
   CURSOR c1(JOB) IS
     SELECT EMPNO,SAL,DEPTNO,JOB FROM EMPLOYEE
       WHERE P_JOB = JOB;

   BEGIN
             OPEN C1,
             LOOP
DISPLAY_RECORDS (P_JOB)
DBMS.OUTPUT.PUTLINE('EmployeeNumber' ||EMPNO ||' Salary' || SAL || 'DepartmentNumber' || DEPTNO || 'Job' ||
JOB )
      CLOSE c1;
END LOOP
   END;
END PROCEDURE DISPLAY_RECORDS;
```

3. Create a function named GET_EMP_ANNSAL which accepts employee Number as a parameter and Returns Annual Salary of the given employee from EMP table if the record exist otherwise returns -1 (formula to computer ANNUAL_SALARY = SAL * 12)

```
CREATE FUNCTION GET_EMP_ANNSAL
(EMPNO IN NUMBER(7))
RETURN NUMBER
IS
SAL NUMBER;
CUURSOR C1

SELECT (SAL*12)  AS ANNUAL_SALARY FROM EMPLOYEE
WHERE EMPNO= NUMBER;
BEGIN
OPEN C1;
FETCH C1 INTO NUMBER;

IF C1%NOTFOUND THEN
NUMBER :=-1;
ENDIF;
CLOSE NUMBER;
EXCEPTION
    RAISE APPLICATION ERROR(-1,'RECORD DOES NOT EXIST -'||SQLCODE||' -ERROR- '||SQLERRM);
END;
```