1) WRITE A PROGRAM TO PRINT HELLO WORLD

```
BEGIN
DBMS_OUTPUT.PUT_LINE ('HELLO WORLD');
END;
/
```

2) WRITE A PROGRAM TO PRINT EVEN NUMBERS FROM 1 TO 100

```
DECLARE
N NUMBER(3) :=0;
BEGIN
WHILE N<=100
LOOP
N :=N+2;
DBMS_OUTPUT.PUT_LINE(N);
END LOOP;
END;
/
```

3) WRITE A PROGRAM TO FIND SUM OF NUMBERS FROM 1 TO 100

```
DECLARE
N NUMBER(3):=1;
S NUMBER(4):=0;
BEGIN

WHILE N<=100
LOOP
S := S+N;
N :=N+1;
END LOOP;
DBMS_OUTPUT.PUT_LINE('THE SUM IS ' | | S);
END;
```

4) WRITE A PROGRAM TO ACCEPT A NUMBER AND FIND SUM OF THE DIGITS

```
DECLARE
N NUMBER(5):=&N;
S NUMBER:=0;
R NUMBER(2):=0;
BEGIN
WHILE N !=0
LOOP
R:=MOD(N,10);
S:=S+R;
N:=TRUNC(N/10);
```

```
END LOOP;
DBMS_OUTPUT.PUT_LINE('SUM OF DIGITS OF GIVEN NUMBER IS '||S);
END;
/
```

5) Write a program to accept a number and print it in reverse order

```
DECLARE
N NUMBER(5):=&N;
REV NUMBER(5):=0;
R NUMBER(5):=0;
BEGIN
WHILE N !=0
LOOP
R:=MOD(N,10);
REV:=REV*10+R;
N:=TRUNC(N/10);
END LOOP;
DBMS_OUTPUT.PUT_LINE('THE REVERSE OF A GIVEN NUMBER IS '||REV);
END;
/
```

6) Write a program accept the value of A,B&C display which is greater

```
DECLARE
A NUMBER(4,2):=&A;
B NUMBER(4,2):=&B;
C NUMBER(4,2):=&C;
BEGIN
IF (A>B AND A>C) THEN
DBMS_OUTPUT.PUT_LINE('A IS GREATER '||''||A);
ELSIF B>C THEN
DBMS_OUTPUT.PUT_LINE('B IS GREATE '||''||B);
ELSE
DBMS_OUTPUT.PUT_LINE('C IS GREATER '||''||C);
END IF;
END;
/
```

7) Write a program accept a string and check whether it is palindrome or not

```
DECLARE
S VARCHAR2(10):='&S';
L VARCHAR2(20);
TEMP VARCHAR2(10);
BEGIN
FOR I IN REVERSE 1..LENGTH(S)
```

```
LOOP
L:=SUBSTR(S,I,1);
TEMP:=TEMP||''||L;
END LOOP;
IF TEMP=S THEN
DBMS_OUTPUT.PUT_LINE(TEMP ||''||' IS PALINDROME');
ELSE
DBMS_OUTPUT.PUT_LINE(TEMP ||''||' IS NOT PALINDROME');
END IF;
END;
/
```

8) WAP to accept the empno and display all the details of emp. If emp does not exist display the message

```
DECLARE
EMPNOV NUMBER:=&EMPNO;
EMPV EMP%ROWTYPE;
BEGIN
SELECT * INTO EMPV FROM EMP WHERE EMPNO=EMPNOV;
DBMS_OUTPUT.PUT_LINE('EMPNO '||EMPV.EMPNO);
DBMS_OUTPUT.PUT_LINE('ENAME '||EMPV.ENAME);
DBMS_OUTPUT.PUT_LINE('JOB '||EMPV.JOB);
DBMS_OUTPUT.PUT_LINE('SALARY '||EMPV.SAL);
DBMS_OUTPUT.PUT_LINE('HIREDATE '||EMPV.HIREDATE);
DBMS_OUTPUT.PUT_LINE('DEPTNO '||EMPV.DEPTNO);
DBMS_OUTPUT.PUT_LINE('MGRNO '||EMPV.MGR);
DBMS_OUTPUT.PUT_LINE('COMMISSION '||EMPV.COMM);
EXCEPTION
WHEN NO_DATA_FOUND THEN
DBMS_OUTPUT.PUT_LINE('EMP NUMBER DOES NOT EXIST');
END;
/
```

9) Write a program to accept the grade and display emps belongs to that grade?

```
DECLARE
GRADEV SALGRADE.GRADE%TYPE:=&GRADE;
CURSOR A IS
SELECT EMP.*,GRADE FROM EMP,SALGRADE WHERE SAL BETWEEN LOSAL AND HISAL AND
GRADE=GRADEV;
B A%ROWTYPE;
BEGIN
OPEN A;
LOOP
FETCH A INTO B;
EXIT WHEN A%NOTFOUND;
DBMS_OUTPUT.PUT_LINE('EMP NO IS ' || B.EMPNO);
```

```
DBMS_OUTPUT.PUT_LINE('ENAME IS ' || B.ENAME);
DBMS_OUTPUT.PUT_LINE('SAL IS ' || B.SAL);
DBMS_OUTPUT.PUT_LINE('MGR NO IS ' || B.MGR);
DBMS_OUTPUT.PUT_LINE('COMM IS ' || B.COMM);
DBMS_OUTPUT.PUT_LINE('HIREDATE IS ' || B.HIREDATE);
DBMS_OUTPUT.PUT_LINE('GRADE IS ' || B.GRADE);
DBMS_OUTPUT.PUT_LINE('EMP JOB IS ' || B.JOB);
DBMS_OUTPUT.PUT_LINE('************************');
END LOOP;
CLOSE A;
END;
/
```

10) Write a program to accept a deptno and display who are working in that dept?

```
DECLARE
DEPTV EMP.DEPTNO%TYPE:=&DEPTNO;
CURSOR A IS
SELECT * FROM EMP WHERE DEPTNO=DEPTV;
B A%ROWTYPE;
BEGIN
OPEN A;
LOOP
FETCH A INTO B;
EXIT WHEN A%NOTFOUND;
DBMS_OUTPUT.PUT_LINE('EMP NO IS ' || B.EMPNO);
DBMS_OUTPUT.PUT_LINE('ENAME IS ' || B.ENAME);
DBMS_OUTPUT.PUT_LINE('SAL IS ' || B.SAL);
DBMS_OUTPUT.PUT_LINE('MGR NO IS ' || B.MGR);
DBMS_OUTPUT.PUT_LINE('COMM IS ' || B.COMM);
DBMS_OUTPUT.PUT_LINE('HIREDATE IS ' || B.HIREDATE);
DBMS_OUTPUT.PUT_LINE('DEPTNO IS ' || B.DEPTNO);
DBMS_OUTPUT.PUT_LINE('EMP JOB IS ' || B.JOB);
DBMS_OUTPUT.PUT_LINE('************************');
END LOOP;
CLOSE A;
END;
/
```

11) Write a program to display all the information of emp table?

```
DECLARE
CURSOR A IS
SELECT * FROM EMP;
B A%ROWTYPE;
BEGIN
OPEN A;
LOOP
```

```
FETCH A INTO B;
EXIT WHEN A%NOTFOUND;
DBMS_OUTPUT.PUT_LINE('EMP NO IS ' || B.EMPNO);
DBMS_OUTPUT.PUT_LINE('ENAME IS ' || B.ENAME);
DBMS_OUTPUT.PUT_LINE('SAL IS ' || B.SAL);
DBMS_OUTPUT.PUT_LINE('MGR NO IS ' || B.MGR);
DBMS_OUTPUT.PUT_LINE('COMM IS ' || B.COMM);
DBMS_OUTPUT.PUT_LINE('HIREDATE IS ' || B.HIREDATE);
DBMS_OUTPUT.PUT_LINE('DEPTNO IS ' || B.DEPTNO);
DBMS_OUTPUT.PUT_LINE('EMP JOB IS ' || B.JOB);
DBMS_OUTPUT.PUT_LINE('*************************');
END LOOP;
CLOSE A;
END;
/
```

12) Write a program to accept a range of salary (that is lower boundary and higher boundary) and print the details of emps along with loc,grade and exp?

```
DECLARE
LOSALV SALGRADE.LOSAL%TYPE:=&LOSAL;
HISALV SALGRADE.HISAL%TYPE:=&HISAL;
EXP NUMBER(5,2);
CURSOR A IS
SELECT EMP.*,LOC,GRADE FROM EMP,DEPT,SALGRADE WHERE
EMP.DEPTNO=DEPT.DEPTNO
AND SAL BETWEEN LOSALV AND HISALV
AND SAL BETWEEN LOSAL AND HISAL;
B A%ROWTYPE;
BEGIN
OPEN A;
LOOP
FETCH A INTO B;
EXIT WHEN A%NOTFOUND;
EXP:=MONTHS_BETWEEN(SYSDATE,B.HIREDATE)/12;
DBMS_OUTPUT.PUT_LINE('EMP NO IS ' || B.EMPNO);
DBMS_OUTPUT.PUT_LINE('ENAME IS ' || B.ENAME);
DBMS_OUTPUT.PUT_LINE('EMP JOB IS ' || B.JOB);
DBMS_OUTPUT.PUT_LINE('LOC IS ' || B.LOC);
DBMS_OUTPUT.PUT_LINE('EXP IS ' || EXP);
DBMS_OUTPUT.PUT_LINE('GRADE IS ' || B.GRADE);
DBMS_OUTPUT.PUT_LINE('*************************');
END LOOP;
CLOSE A;
END;
/
```

13) Write a function to accept the empno and return exp with minimum 3 decimal?

```
CREATE OR REPLACE FUNCTION E_DETAILS(EMPNOV NUMBER) RETURN NUMBER
IS
HIREDATEV EMP.HIREDATE%TYPE;
EXP NUMBER(6,3);
BEGIN
SELECT HIREDATE INTO HIREDATEV FROM EMP WHERE EMPNO=EMPNOV;
EXP:=MONTHS_BETWEEN(SYSDATE,HIREDATEV)/12;
RETURN EXP;
END;
/
```

14) Write a database trigger halt the transaction on Sunday on EMP table

```
CREATE OR REPLACE TRIGGER SUN_TRI
AFTER INSERT OR UPDATE OR DELETE ON EMP
DECLARE
DY VARCHAR2(200);
BEGIN
DY:=TO_CHAR(SYSDATE,'DY');
IF DY='SUN' THEN
RAISE_APPLICATION_ERROR(-20005,'TODAY IS SUNDAY TRANSACTION NOT ALLOWED
TODAY');
END IF;
END;
/
```

15) Write a database trigger halt the transaction of USER SCOTT on table EMP

```
CREATE OR REPLACE TRIGGER SCOTT_TRI
BEFORE INSERT OR UPDATE OR DELETE ON EMP
BEGIN
IF USER = 'SCOTT' THEN
RAISE_APPLICATION_ERROR(-20006,'TRANSACTION NOT ALLOWED FOR SCOTT');
END IF;
END;
/
```

16) Write a database trigger stroe the username ,type of transaction ,date of transaction and time of transaction of table emp into the table EMP_LOG

```
CREATE OR REPLACE TRIGGER TRANS_TYPE
AFTER INSERT OR UPDATE OR DELETE ON EMP
```

```
DECLARE
V VARCHAR2(50);
BEGIN
IF INSERTING THEN
V:='I';
ELSIF UPDATING THEN
V:='U';
ELSE
V:='D';
END IF;
INSERT INTO EMP_LOG VALUES (USER,V,SYSDATE,TO_CHAR(SYSDATE,'HH:MI:SS'));
END;
/
```

17) Write a database trigger store the deleted data of EMP table in EMPDEL table

```
CREATE OR REPLACE TRIGGER DEL_TRI
BEFORE DELETE ON EMP
FOR EACH ROW
BEGIN
INSERT INTO EMPDEL
VALUES
(:OLD.EMPNO,:OLD.ENAME,:OLD.JOB,:OLD.MGR,:OLD.HIREDATE,:OLD.SAL,:OLD.COMM,
:OLD.DEPTNO,SYSDATE,TO_CHAR(SYSDATE,'HH:MI:SS));
END;
/
```

18) Write a database trigger halt the transaction of EMP table if the deptno is does not exist in the dept table

```
CREATE OR REPLACE TRIGGER DEPT_NO
BEFORE INSERT OR UPDATE OR DELETE ON EMP
FOR EACH ROW
DECLARE
DNO NUMBER:=0;
BEGIN
SELECT COUNT(*) INTO DNO FROM DEPT WHERE DEPTNO=:NEW.DEPTNO;
DBMS_OUTPUT.PUT_LINE(DNO);
IF DNO=0 THEN
RAISE_APPLICATION_ERROR(-20009,'DEPTNO NOT EXIST IN DEPT TABLE....');
END IF;
END;
/
```