SELECT 'Welcome TO PL/SQL Class' FROM dual  
  
SELECT ename||empno FROM emp   
  
  
DECLARE  
a NUMBER;  
b NUMBER;  
BEGIN  
a :=10;  
b :=15;  
  
IF a<b THEN  
  **DBMS\_OUTPUT.put\_line**('A is Smaller');  
END IF;  
IF b< a THEN   
**DBMS\_OUTPUT.put\_line**('B is Smaller');  
END IF;  
END;  
  
  
DECLARE  
a NUMBER;  
b NUMBER;  
BEGIN  
a :=10;  
b :=15;  
  
IF a<b THEN  
  **DBMS\_OUTPUT.put\_line**('A is Smaller');  
ELSE   
**DBMS\_OUTPUT.put\_line**('B is Smaller');  
END IF;  
END;  
  
DECLARE  
a NUMBER;  
b NUMBER;  
BEGIN  
a :=15;  
b :=15;  
IF a= b THEN  
   **DBMS\_OUTPUT.put\_line**('Both are equal');  
ELSE  
   IF a<b THEN  
     **DBMS\_OUTPUT.put\_line**('A is Smaller');  
    ELSE   
    **DBMS\_OUTPUT.put\_line**('B is Smaller');  
   END IF;  
  END IF;    
END;  
  
  
  
DECLARE  
a NUMBER;  
b NUMBER;  
BEGIN  
a :=15;  
b :=15;  
IF a= b THEN  
   **DBMS\_OUTPUT.put\_line**('Both are equal');  
ELSIF a<b THEN  
     **DBMS\_OUTPUT.put\_line**('A is Smaller');  
ELSE   
    **DBMS\_OUTPUT.put\_line**('B is Smaller');  
END IF;  
    
END;  
  
  
BEGIN  
IF 1 THEN  
  **DBMS\_OUTPUT.put\_line**('True');  
ELSE  
  **DBMS\_OUTPUT.put\_line**('False');    
END IF;  
END;  
  
  
  
BEGIN  
IF FALSE THEN  
  **DBMS\_OUTPUT.put\_line**('True');  
ELSE  
  **DBMS\_OUTPUT.put\_line**('False');    
END IF;  
END;  
  
BEGIN  
IF TRUE THEN  
  **DBMS\_OUTPUT.put\_line**('True');  
ELSE  
  **DBMS\_OUTPUT.put\_line**('False');    
END IF;  
END;  
  
BEGIN  
IF null=1 THEN  
  **DBMS\_OUTPUT.put\_line**('True');  
ELSE  
  **DBMS\_OUTPUT.put\_line**('False');    
END IF;  
END;  
  
  
BEGIN  
IF null =null THEN  
  **DBMS\_OUTPUT.put\_line**('True');  
ELSE  
  **DBMS\_OUTPUT.put\_line**('False');    
END IF;  
END;  
  
  
BEGIN  
IF null IS null THEN  
  **DBMS\_OUTPUT.put\_line**('True');  
ELSE  
  **DBMS\_OUTPUT.put\_line**('False');    
END IF;  
END;  
  
  
BEGIN  
IF null THEN  
  **DBMS\_OUTPUT.put\_line**('True');  
ELSE  
  **DBMS\_OUTPUT.put\_line**('False');    
END IF;  
END;  
  
  
  
DECLARE  
i NUMBER:=1;  
BEGIN  
  
LOOP  
  
IF i<=5 THEN  
**DBMS\_OUTPUT.put\_line**('Welcome');  
ELSE  
  EXIT;  
END IF;  
i:=i+1;  
END LOOP;  
  
END;  
  
  
  
  
DECLARE  
i NUMBER:=300;  
BEGIN  
LOOP  
IF i<=400 THEN  
**DBMS\_OUTPUT.put\_line**(i);  
ELSE  
  EXIT;  
END IF;  
i:=i+1;  
END LOOP;  
END;  
  
  
DECLARE  
i NUMBER:=300;  
BEGIN  
LOOP  
  
IF i<=400 THEN  
  IF MOD(i,2) = 0 THEN  
   **DBMS\_OUTPUT.put\_line**(i);  
  END IF;  
    
ELSE  
  EXIT;  
      
END IF;  
i:=i+1  
END LOOP;  
END;  
  
  
  
DECLARE  
i NUMBER:=300;  
BEGIN  
LOOP  
  
IF i<=400 THEN  
  IF MOD(i,2) = 1 THEN  
   **DBMS\_OUTPUT.put\_line**(i);  
  END IF;  
    
ELSE  
  EXIT;  
      
END IF;  
i:=i+1  
END LOOP;  
END;  
  
  
  
DECLARE  
i NUMBER:=300;  
BEGIN  
LOOP  
IF i<=400 THEN  
  IF MOD(i,2) = 0 THEN  
   **DBMS\_OUTPUT.put\_line**(i||'is even');  
   ELSE  
    **DBMS\_OUTPUT.put\_line**(i||'is odd');  
  END IF;  
  
ELSE  
  EXIT;  
      
END IF;  
i:=i+1  
END LOOP;  
END;  
  
  
  
DECLARE  
i NUMBER:=0;  
BEGIN  
WHILE i<100  
LOOP  
  **DBMS\_OUTPUT.put\_line**(i);  
  i:=i+1;  
END LOOP;  
END;  
  
  
  
  
BEGIN  
FOR i IN 1..100  
LOOP  
  **DBMS\_OUTPUT.put\_line**(i);  
END LOOP;  
END;  
  
  
DECLARE  
i NUMBER:=0;  
BEGIN  
<<start\_point>>  
**DBMS\_OUTPUT.put\_line**(i);  
i:=i+1;  
IF i<=5 THEN  
GOTO start\_point;  
END IF;  
END;  
  
  
  
BEGIN  
EXECUTE IMMEDIATE 'CREATE TABLE abc (n NUMBER)';  
END;  
  
DECLARE  
ls VARCHAR2(400):='CREATE TABLE abc (n NUMBER)';  
BEGIN  
EXECUTE IMMEDIATE ls;  
END;  
  
BEGIN  
INSERT INTO abc VALUES(2);  
COMMIT;  
END;  
  
DECLARE  
l NUMBER;  
BEGIN  
SELECT \* INTO l FROM abc;  
**DBMS\_OUTPUT.put\_line**(l);  
END;  
  
DECLARE  
l NUMBER;  
BEGIN  
SELECT \* INTO l FROM abc WHERE 1=2;  
**DBMS\_OUTPUT.put\_line**(l);  
END;  
  
DECLARE  
l NUMBER;  
BEGIN  
SELECT \* INTO l FROM abc WHERE 1=2;  
**DBMS\_OUTPUT.put\_line**(l);  
END;  
  
SELECT \* FROM abc  
  
CREATE TABLE abc (n NUMBER)  
  
DROP TABLE abc  
  
  
DECLARE  
l VARCHAR2(400);  
m VARCHAR2(400);  
BEGIN  
l:='CREATE TABLE ISCONE (n NUMBER)';  
EXECUTE IMMEDIATE l;  
m:='INSERT INTO ISCONE VALUES(1)';  
EXECUTE IMMEDIATE m;  
COMMIT;  
l:='SELECT n FROM iscone';  
EXECUTE IMMEDIATE l INTO m;  
**DBMS\_OUTPUT.put\_line**(m);  
END;  
  
  
  
SELECT \* FROM iscone  
  
DROP TABLE ISCONE  
  
DECLARE  
l\_name VARCHAR2(4);  
BEGIN  
SELECT ename INTO l\_name FROM emp WHERE 1=2;  
EXCEPTION  
WHEN OTHERS THEN  
  **DBMS\_OUTPUT.put\_line**(SQLERRM);   
END;  
  
DECLARE  
l\_name VARCHAR2(4);  
BEGIN  
SELECT ename INTO l\_name FROM emp WHERE empno=2;  
EXCEPTION  
WHEN NO\_DATA\_FOUND THEN  
  **DBMS\_OUTPUT.put\_line**('Employee does not exist');   
WHEN OTHERS THEN  
  **DBMS\_OUTPUT.put\_line**(SQLERRM);   
END;  
  
  
  
DECLARE  
l\_name VARCHAR2(4);  
BEGIN  
SELECT ename INTO l\_name FROM emp WHERE empno=7369;  
EXCEPTION  
WHEN NO\_DATA\_FOUND THEN  
  **DBMS\_OUTPUT.put\_line**('Employee does not exist');   
WHEN OTHERS THEN  
  **DBMS\_OUTPUT.put\_line**(SQLERRM);   
END;  
  
  
DECLARE  
l\_name VARCHAR2(14);  
BEGIN  
SELECT ename INTO l\_name FROM emp WHERE empno=7369;  
EXCEPTION  
WHEN NO\_DATA\_FOUND THEN  
  **DBMS\_OUTPUT.put\_line**('Employee does not exist');   
WHEN OTHERS THEN  
  **DBMS\_OUTPUT.put\_line**(SQLERRM);   
END;  
  
DECLARE  
l\_name VARCHAR2(1);  
BEGIN  
SELECT ename INTO l\_name FROM emp WHERE empno=7369;  
EXCEPTION  
WHEN OTHERS THEN  
  **DBMS\_OUTPUT.put\_line**(SQLERRM);  
WHEN NO\_DATA\_FOUND THEN  
  **DBMS\_OUTPUT.put\_line**('Employee does not exist');    
END;  
  
BEGIN  
  
DECLARE  
l\_name VARCHAR2(10):='Immense Source';  
BEGIN  
SELECT ename INTO l\_name FROM emp ;  
EXCEPTION  
WHEN TOO\_MANY\_ROWS THEN  
  **DBMS\_OUTPUT.put\_line**('ISC'||l\_name);  
WHEN NO\_DATA\_FOUND THEN  
  **DBMS\_OUTPUT.put\_line**('ISC-1');  
WHEN OTHERS THEN  
  **DBMS\_OUTPUT.put\_line**('ISC1-2');      
END;  
EXCEPTION  
WHEN OTHERS THEN  
   **DBMS\_OUTPUT.put\_line**(SQLERRM);  
END;  
  
DECLARE  
l\_name VARCHAR2(100):='Immense Source';  
BEGIN  
SELECT ename INTO l\_name FROM emp ;  
EXCEPTION  
WHEN TOO\_MANY\_ROWS THEN  
  **DBMS\_OUTPUT.put\_line**('ISC'||l\_name);  
WHEN NO\_DATA\_FOUND THEN  
  **DBMS\_OUTPUT.put\_line**('ISC-1');  
WHEN OTHERS THEN  
  **DBMS\_OUTPUT.put\_line**('ISC1-2');      
END;  
  
select \* from emp  
  
  
DECLARE  
BEGIN  
**DBMS\_OUTPUT.put\_line**(1);  
**DBMS\_OUTPUT.put\_line**(2);  
**DBMS\_OUTPUT.put\_line**(3);  
**DBMS\_OUTPUT.put\_line**(4);  
EXCEPTION  
WHEN INVALID\_NUMBER THEN  
**DBMS\_OUTPUT.put\_line**('Invalid Number');  
WHEN OTHERS THEN  
**DBMS\_OUTPUT.put\_line**('Others');  
END;  
  
DECLARE  
BEGIN  
**DBMS\_OUTPUT.put\_line**(1);  
**DBMS\_OUTPUT.put\_line**(2);  
**DBMS\_OUTPUT.put\_line**('Sum:='||3);  
**DBMS\_OUTPUT.put\_line**(4);  
EXCEPTION  
WHEN INVALID\_NUMBER THEN  
**DBMS\_OUTPUT.put\_line**('Invalid Number');  
WHEN OTHERS THEN  
**DBMS\_OUTPUT.put\_line**('Others');  
END;  
  
  
DECLARE  
BEGIN  
**DBMS\_OUTPUT.put\_line**(1);  
**DBMS\_OUTPUT.put\_line**(2);  
**DBMS\_OUTPUT.put\_line**('Sum:='||3+10);  
**DBMS\_OUTPUT.put\_line**(4);  
EXCEPTION  
WHEN INVALID\_NUMBER THEN  
**DBMS\_OUTPUT.put\_line**('Invalid Number');  
WHEN OTHERS THEN  
**DBMS\_OUTPUT.put\_line**(SQLERRM);  
END;  
  
  
DECLARE  
BEGIN  
**DBMS\_OUTPUT.put\_line**(1);  
**DBMS\_OUTPUT.put\_line**(2);  
**DBMS\_OUTPUT.put\_line**(3+10);  
**DBMS\_OUTPUT.put\_line**(4);  
EXCEPTION  
WHEN INVALID\_NUMBER THEN  
**DBMS\_OUTPUT.put\_line**('Invalid Number');  
WHEN OTHERS THEN  
**DBMS\_OUTPUT.put\_line**(SQLERRM);  
END;  
  
  
DECLARE  
BEGIN  
**DBMS\_OUTPUT.put\_line**(1);  
**DBMS\_OUTPUT.put\_line**(2);  
**DBMS\_OUTPUT.put\_line**('Sum'||(3+10));  
**DBMS\_OUTPUT.put\_line**(4);  
EXCEPTION  
WHEN INVALID\_NUMBER THEN  
**DBMS\_OUTPUT.put\_line**('Invalid Number');  
WHEN OTHERS THEN  
**DBMS\_OUTPUT.put\_line**(SQLERRM);  
END;  
  
  
DECLARE  
ab EXCEPTION;  
BEGIN  
**DBMS\_OUTPUT.put\_line**(1);  
**DBMS\_OUTPUT.put\_line**(2);  
RAISE ab;  
**DBMS\_OUTPUT.put\_line**('Sum'||(3+10));  
**DBMS\_OUTPUT.put\_line**(4);  
EXCEPTION  
WHEN INVALID\_NUMBER THEN  
**DBMS\_OUTPUT.put\_line**('Invalid Number');  
WHEN ab THEN  
  **DBMS\_OUTPUT.put\_line**('My Execption');  
WHEN OTHERS THEN  
**DBMS\_OUTPUT.put\_line**(SQLERRM);  
END;  
  
  
DECLARE  
ab EXCEPTION;  
PRAGMA EXCEPTION\_INIT(ab,-01400);  
BEGIN  
INSERT INTO isc\_not\_null VALUES (1);  
INSERT INTO isc\_not\_null VALUES (null);  
COMMIT;  
EXCEPTION  
WHEN INVALID\_NUMBER THEN  
**DBMS\_OUTPUT.put\_line**('Invalid Number');  
WHEN ab THEN  
  **DBMS\_OUTPUT.put\_line**('My Execption');  
WHEN OTHERS THEN  
**DBMS\_OUTPUT.put\_line**(SQLERRM);  
COMMIT;  
END;  
  
  
DECLARE  
ab EXCEPTION;  
PRAGMA EXCEPTION\_INIT(ab,-01400);  
BEGIN  
INSERT INTO isc\_not\_null VALUES (1);  
RETURN;  
INSERT INTO isc\_not\_null VALUES (null);  
COMMIT;  
EXCEPTION  
WHEN INVALID\_NUMBER THEN  
**DBMS\_OUTPUT.put\_line**('Invalid Number');  
WHEN ab THEN  
  **DBMS\_OUTPUT.put\_line**('My Execption');  
WHEN OTHERS THEN  
**DBMS\_OUTPUT.put\_line**(SQLERRM);  
COMMIT;  
END;  
  
  
DECLARE  
ab EXCEPTION;  
PRAGMA EXCEPTION\_INIT(ab,-01400);  
BEGIN  
INSERT INTO isc\_not\_null VALUES (1);  
RAISE\_APPLICATION\_ERROR (-20000,'Insert not allowed');  
INSERT INTO isc\_not\_null VALUES (null);  
COMMIT;  
EXCEPTION  
WHEN INVALID\_NUMBER THEN  
**DBMS\_OUTPUT.put\_line**('Invalid Number');  
WHEN ab THEN  
  **DBMS\_OUTPUT.put\_line**(SQLERRM);  
WHEN OTHERS THEN  
**DBMS\_OUTPUT.put\_line**(SQLERRM);  
COMMIT;  
END;  
  
CREATE TABLE isc\_not\_null (n NUMBER NOT NULL )  
  
SELECT \* FROM isc\_not\_null  
  
CREATE OR REPLACE PROCEDURE isc\_sum  
IS  
la NUMBER:=10;  
lb NUMBER:=20;  
lc NUMBER;  
BEGIN  
lc:=la+lb;  
**DBMS\_OUTPUT.put\_line**(lc);  
END;  
  
CREATE OR REPLACE PROCEDURE isc\_sum(la NUMBER,lb IN NUMBER, lc OUT NUMBER)  
IS  
BEGIN  
lc:=la+lb;  
END;  
  
  
CREATE OR REPLACE FUNCTION isc\_sum\_fn(la NUMBER,lb IN NUMBER) RETURN NUMBER  
IS  
lc NUMBER;  
BEGIN  
lc:=la+lb;  
RETURN lc;  
END;  
  
DECLARE  
l NUMBER;  
BEGIN  
**DBMS\_OUTPUT.put\_line**('before'||l);  
l:=isc\_sum\_fn(40,40);  
**DBMS\_OUTPUT.put\_line**('After'||l);  
END;  
  
DECLARE  
l NUMBER;  
BEGIN  
**DBMS\_OUTPUT.put\_line**('before'||l);  
isc\_sum(40,40,l);  
**DBMS\_OUTPUT.put\_line**('After'||l);  
END;  
  
SELECT isc\_sum\_fn(50,20) FROM dual  
  
SELECT \* FROM all\_objects WHERE object\_name='ISC\_SUM'  
  
SELECT \* FROM all\_errors WHERE name='ISC\_SUM\_FN'  
  
DECLARE  
lv NUMBER;  
PROCEDURE isc\_sum(la NUMBER,lb IN NUMBER, lc OUT NUMBER)  
IS  
BEGIN  
lc:=la-lb;  
END;  
  
BEGIN  
isc\_sum(10,20,lv);  
**DBMS\_OUTPUT.put\_line**('After'||lv);  
END;   
  
DECLARE  
a NUMBER:=1;  
b NUMBER:=2;  
C NUMBER;  
BEGIN  
  C:=a;  
  a:=b;  
  b:=c;  
**DBMS\_OUTPUT.put\_line**('a'||a);  
  **DBMS\_OUTPUT.put\_line**('b'||b);  
END;  
  
  
DECLARE  
a NUMBER:=1;  
b NUMBER:=2;  
BEGIN  
  b:=a+b;  
  a:= b-a;  
  b:= b-a;  
**DBMS\_OUTPUT.put\_line**('a'||a);  
  **DBMS\_OUTPUT.put\_line**('b'||b);  
END;  
  
CREATE OR REPLACE PACKAGE isc\_pack  
IS  
PROCEDURE isc\_sum(pa NUMBER,pb NUMBER,xr OUT NUMBER);  
FUNCTION isc\_sum\_f(pa NUMBER,pb NUMBER) RETURN NUMBER;  
END isc\_pack;  
  
  
CREATE OR REPLACE PACKAGE BODY isc\_pack  
IS  
PROCEDURE isc\_sum(pa NUMBER,pb NUMBER,xr OUT NUMBER)  
IS  
BEGIN  
xr:=pa+pb;  
END isc\_sum;  
FUNCTION isc\_sum\_f(pa NUMBER,pb NUMBER) RETURN NUMBER  
IS  
BEGIN  
RETURN (pa+pb);  
END isc\_sum\_f;  
END isc\_pack;  
  
DECLARE  
x NUMBER;  
BEGIN  
isc\_pack.isc\_sum(10,20,x);  
**DBMS\_OUTPUT.put\_line**(x);  
x:=isc\_pack.isc\_sum\_f(40,30);  
**DBMS\_OUTPUT.put\_line**(x);  
END;  
  
  
CREATE OR REPLACE PACKAGE isc\_pack  
IS  
PROCEDURE isc\_sum(pa NUMBER,pb NUMBER,xr OUT NUMBER);  
FUNCTION isc\_sum(pa NUMBER,pb NUMBER) RETURN NUMBER;  
END isc\_pack;  
  
  
CREATE OR REPLACE PACKAGE BODY isc\_pack  
IS  
PROCEDURE isc\_test;  
PROCEDURE isc\_sum(pa NUMBER,pb NUMBER,xr OUT NUMBER)  
IS  
BEGIN  
xr:=pa+pb;  
END isc\_sum;  
FUNCTION isc\_sum(pa NUMBER,pb NUMBER) RETURN NUMBER  
IS  
BEGIN  
isc\_test;  
RETURN (pa+pb);  
END isc\_sum;  
PROCEDURE isc\_test  
IS  
BEGIN  
**DBMS\_OUTPUT.put\_line**('Test');  
END isc\_test;  
END isc\_pack;  
  
DECLARE  
x NUMBER;  
BEGIN  
isc\_pack.isc\_sum(10,20,x);  
**DBMS\_OUTPUT.put\_line**(x);  
x:=isc\_pack.isc\_sum(40,30);  
**DBMS\_OUTPUT.put\_line**(x);  
isc\_pack.isc\_sum(isc\_pack.isc\_sum(40,30),20,x);  
**DBMS\_OUTPUT.put\_line**(x);  
END;  
  
  
  
DECLARE  
l VARCHAR2(20);  
ln NUMBER;  
BEGIN  
SELECT ename INTO l FROM EMP WHERE rownum=1;  
IF SQL%ISOPEN THEN  
**DBMS\_OUTPUT.put\_line**('Opened');  
ELSE  
   **DBMS\_OUTPUT.put\_line**('Not Opened');  
END IF;  
END;  
  
  
DECLARE  
l VARCHAR2(20);  
ln NUMBER;  
BEGIN  
--SELECT ename INTO l FROM EMP WHERE rownum=1;  
IF SQL%ISOPEN THEN  
**DBMS\_OUTPUT.put\_line**('Opened');  
ELSE  
   **DBMS\_OUTPUT.put\_line**('Not Opened');  
END IF;  
END;  
  
  
DECLARE  
l VARCHAR2(20);  
ln NUMBER;  
BEGIN  
SELECT ename INTO l FROM EMP WHERE rownum=1;  
  
IF SQL%FOUND THEN  
**DBMS\_OUTPUT.put\_line**('Found');  
ELSE  
   **DBMS\_OUTPUT.put\_line**('Not Found');  
END IF;  
  
END;  
  
  
DECLARE  
l VARCHAR2(20);  
ln NUMBER;  
BEGIN  
        BEGIN  
         SELECT ename INTO l FROM EMP WHERE 2=1;  
         EXCEPTION  
         WHEN OTHERS THEN   
          null;  
        END;    
  
IF SQL%FOUND THEN  
**DBMS\_OUTPUT.put\_line**('Found');  
ELSE  
   **DBMS\_OUTPUT.put\_line**('Not Found');  
END IF;  
EXCEPTION  
WHEN OTHERS THEN  
  **DBMS\_OUTPUT.put\_line**('Others');  
END;  
  
  
  
DECLARE  
l VARCHAR2(20);  
ln NUMBER;  
BEGIN  
        BEGIN  
         SELECT comm INTO l FROM EMP WHERE empno=7369;  
         EXCEPTION  
         WHEN OTHERS THEN   
          null;  
        END;    
  
IF SQL%FOUND THEN  
**DBMS\_OUTPUT.put\_line**('Found');  
ELSE  
   **DBMS\_OUTPUT.put\_line**('Not Found');  
END IF;  
EXCEPTION  
WHEN OTHERS THEN  
  **DBMS\_OUTPUT.put\_line**('Others');  
END;  
  
  
  
DECLARE  
l VARCHAR2(20);  
ln NUMBER;  
BEGIN  
SELECT comm INTO l FROM EMP WHERE empno=7369;  
ln:=SQL%ROWCOUNT;  
   **DBMS\_OUTPUT.put\_line**(ln);  
EXCEPTION  
WHEN OTHERS THEN  
  **DBMS\_OUTPUT.put\_line**('Others');  
END;  
  
  
DECLARE  
l VARCHAR2(20);  
ln NUMBER;  
BEGIN  
SELECT count(comm) INTO l FROM EMP;  
ln:=SQL%ROWCOUNT;  
   **DBMS\_OUTPUT.put\_line**(ln);  
EXCEPTION  
WHEN OTHERS THEN  
  **DBMS\_OUTPUT.put\_line**('Others');  
END;  
  
  
  
DECLARE  
l VARCHAR2(20);  
ln NUMBER;  
BEGIN  
SELECT (comm) INTO l FROM EMP;  
ln:=SQL%ROWCOUNT;  
   **DBMS\_OUTPUT.put\_line**(l);  
EXCEPTION  
WHEN OTHERS THEN  
  **DBMS\_OUTPUT.put\_line**('Others');  
END;  
  
  
DECLARE  
l VARCHAR2(20):=100;  
ln NUMBER;  
BEGIN  
SELECT count(comm) INTO l FROM EMP;  
ln:=SQL%ROWCOUNT;  
   **DBMS\_OUTPUT.put\_line**(l);  
EXCEPTION  
WHEN OTHERS THEN  
  **DBMS\_OUTPUT.put\_line**('Others'||l);  
END;  
  
  
DECLARE  
l number:='100';  
ln NUMBER;  
BEGIN  
SELECT count(comm) INTO l FROM EMP;  
ln:=SQL%ROWCOUNT;  
   **DBMS\_OUTPUT.put\_line**(l);  
EXCEPTION  
WHEN OTHERS THEN  
  **DBMS\_OUTPUT.put\_line**('Others'||l);  
END;  
  
  
BEGIN  
  
DECLARE  
l number:='a';  
ln NUMBER;  
BEGIN  
SELECT count(comm) INTO l FROM EMP;  
ln:=SQL%ROWCOUNT;  
   **DBMS\_OUTPUT.put\_line**(l);  
EXCEPTION  
WHEN OTHERS THEN  
  **DBMS\_OUTPUT.put\_line**('Others'||l);  
END;  
  
EXCEPTION  
WHEN OTHERS THEN  
    **DBMS\_OUTPUT.put\_line**('runtime');  
END;      
  
  
  
  
BEGIN  
  
DECLARE  
l number:='a';  
ln NUMBER;  
BEGIN  
**DBMS\_OUTPUT.put\_line**('Before');  
SELECT empno INTO l FROM EMP;  
ln:=SQL%ROWCOUNT;  
   **DBMS\_OUTPUT.put\_line**(l);  
EXCEPTION  
WHEN OTHERS THEN  
  **DBMS\_OUTPUT.put\_line**('Others'||l);  
END;  
  
EXCEPTION  
WHEN OTHERS THEN  
    **DBMS\_OUTPUT.put\_line**('runtime');  
END;    
  
  
BEGIN  
  
DECLARE  
l number;  
ln NUMBER;  
BEGIN  
**DBMS\_OUTPUT.put\_line**('Before');  
SELECT empno INTO l FROM EMP;  
ln:=SQL%ROWCOUNT;  
   **DBMS\_OUTPUT.put\_line**(l);  
EXCEPTION  
WHEN OTHERS THEN  
  **DBMS\_OUTPUT.put\_line**('Others'||l);  
END;  
  
EXCEPTION  
WHEN OTHERS THEN  
    **DBMS\_OUTPUT.put\_line**('runtime');  
END;    
  
  
select \* from emp  
  
DECLARE  
CURSOR emp\_cur  
IS  
SELECT ename  
      ,empno  
      ,sal  
      ,deptno  
  FROM emp;     
l\_ename emp.ename%TYPE ;    
l\_empno emp.empno%TYPE;  
l\_sal   emp.sal%TYPE;  
l\_deptno emp.deptno%TYPE;  
  
BEGIN  
OPEN emp\_cur;  
FETCH emp\_cur INTO  l\_ename,l\_empno,l\_sal,l\_deptno;  
CLOSE emp\_cur;  
**DBMS\_OUTPUT.put\_line**(l\_ename||l\_empno);  
END;  
  
  
DECLARE  
CURSOR emp\_cur  
IS  
SELECT ename  
      ,empno  
      ,sal  
      ,deptno  
  FROM emp;     
l\_ename emp.ename%TYPE ;    
l\_empno emp.empno%TYPE;  
l\_sal   emp.sal%TYPE;  
l\_deptno emp.deptno%TYPE;  
  
BEGIN  
OPEN emp\_cur;  
FETCH emp\_cur INTO  l\_ename,l\_empno,l\_sal,l\_deptno;  
**DBMS\_OUTPUT.put\_line**(l\_ename||l\_empno);  
FETCH emp\_cur INTO  l\_ename,l\_empno,l\_sal,l\_deptno;  
**DBMS\_OUTPUT.put\_line**(l\_ename||l\_empno);  
FETCH emp\_cur INTO  l\_ename,l\_empno,l\_sal,l\_deptno;  
**DBMS\_OUTPUT.put\_line**(l\_ename||l\_empno);  
CLOSE emp\_cur;  
END;  
  
  
  
DECLARE  
CURSOR emp\_cur  
IS  
SELECT ename  
      ,empno  
      ,sal  
      ,deptno  
  FROM emp;     
l\_ename emp.ename%TYPE ;    
l\_empno emp.empno%TYPE;  
l\_sal   emp.sal%TYPE;  
l\_deptno emp.deptno%TYPE;  
  
BEGIN  
OPEN emp\_cur;  
LOOP  
FETCH emp\_cur INTO  l\_ename,l\_empno,l\_sal,l\_deptno;  
IF emp\_cur%NOTFOUND THEN  
   exit;  
END IF;  
**DBMS\_OUTPUT.put\_line**(l\_ename||l\_empno);  
END LOOP;  
CLOSE emp\_cur;  
END;  
  
  
  
DECLARE  
CURSOR emp\_cur  
IS  
SELECT ename name  
      ,empno  
      ,sal  
      ,deptno  
  FROM emp;     
BEGIN  
FOR  emp\_cur\_rec IN emp\_Cur  
LOOP  
**DBMS\_OUTPUT.put\_line**(emp\_cur\_rec.name ||emp\_cur\_rec.empno);  
END LOOP;  
END;  
  
CREATE OR REPLACE PROCEDURE isc\_cur(p\_deptno IN NUMBER)  
IS  
CURSOR emp\_cur (c\_deptno NUMBER)  
IS  
SELECT ename name  
      ,empno  
      ,sal  
      ,deptno  
  FROM emp  
WHERE deptno=c\_deptno;      
BEGIN  
FOR  emp\_cur\_rec IN emp\_Cur(p\_deptno)  
LOOP  
**DBMS\_OUTPUT.put\_line**(emp\_cur\_rec.name ||emp\_cur\_rec.empno);  
END LOOP;  
END;  
  
BEGIN  
isc\_cur(30);  
END;  
  
  
CREATE OR REPLACE PROCEDURE isc\_cur  
IS  
CURSOR emp\_cur (c\_deptno NUMBER)  
IS  
SELECT ename name  
      ,empno  
      ,sal  
      ,deptno  
  FROM emp  
WHERE deptno=c\_deptno;      
BEGIN  
FOR  emp\_cur\_rec IN emp\_Cur(30)  
LOOP  
**DBMS\_OUTPUT.put\_line**(emp\_cur\_rec.name ||emp\_cur\_rec.empno);  
END LOOP;  
END;  
  
  
BEGIN  
isc\_cur;  
END;  
  
CREATE OR REPLACE PROCEDURE isc\_ref(p\_column\_name VARCHAR2)  
IS  
lst VARCHAR2(4000);  
l\_result VARCHAR(4000);  
BEGIN  
lst :='SELECT '||p\_column\_name||' FROM emp WHERE rownum=1';  
EXECUTE IMMEDIATE lst INTO l\_result;   
**DBMS\_OUTPUT.put\_line**(l\_result);  
END;  
  
  
  
BEGIN  
isc\_ref('DEPTNO');  
END;  
  
  
SELECT SAL FROM emp  
  
CREATE OR REPLACE PROCEDURE isc\_ref(p\_column\_name VARCHAR2)  
IS  
TYPE l\_ref IS REF CURSOR;  
l\_cur l\_ref ;  
lst VARCHAR2(4000);  
l\_result VARCHAR(4000);  
BEGIN  
lst :='SELECT '||p\_column\_name||' FROM emp ';  
OPEN l\_cur FOR lst;  
LOOP  
FETCH l\_cur INTO  l\_result;  
IF l\_cur%NOTFOUND THEN  
   EXIT;  
END IF;  
**DBMS\_OUTPUT.put\_line**(l\_result);     
END LOOP;  
END;  
  
  
  
BEGIN  
isc\_ref('ENAME');  
END;  
  
  
SELECT SAL FROM emp  
  
CREATE OR REPLACE PROCEDURE isc\_ref(p\_column\_name VARCHAR2,p\_deptno NUMBER)  
IS  
TYPE l\_ref IS REF CURSOR;  
l\_cur l\_ref ;  
lst VARCHAR2(4000);  
l\_result VARCHAR(4000);  
BEGIN  
lst :='SELECT '||p\_column\_name||' FROM emp WHERE deptno = '||p\_deptno;  
OPEN l\_cur FOR lst;  
LOOP  
FETCH l\_cur INTO  l\_result;  
IF l\_cur%NOTFOUND THEN  
   EXIT;  
END IF;  
**DBMS\_OUTPUT.put\_line**(l\_result);     
END LOOP;  
END;  
  
  
  
BEGIN  
isc\_ref('ENAME',20);  
END;  
  
  
BEGIN  
FOR i IN (SELECT \*  FROM EMP)  
LOOP  
**DBMS\_OUTPUT.put\_line**(i.ename);  
END LOOP;  
END;  
  
CREATE OR REPLACE TRIGGER isc\_St\_trg\_bfi  
BEFORE INSERT ON isc\_st  
BEGIN  
**DBMS\_OUTPUT.put\_line**('Before Insert');  
END;   
  
  
CREATE OR REPLACE TRIGGER isc\_St\_trg\_afi  
AFTER INSERT ON isc\_st  
BEGIN  
**DBMS\_OUTPUT.put\_line**('After Insert');  
END;   
  
CREATE OR REPLACE TRIGGER isc\_St\_trg\_afi  
AFTER UPDATE ON isc\_st  
BEGIN  
**DBMS\_OUTPUT.put\_line**('After update');  
END;   
  
  
  
CREATE OR REPLACE TRIGGER isc\_St\_trg\_afir  
AFTER UPDATE ON isc\_st  
FOR EACH ROW  
BEGIN  
**DBMS\_OUTPUT.put\_line**('After update');  
END;   
  
CREATE OR REPLACE TRIGGER isc\_St\_trg\_afi  
AFTER INSERT ON isc\_st  
DECLARE  
PRAGMA AUTONOMOUS\_TRANSACTION;  
BEGIN  
**DBMS\_OUTPUT.put\_line**('After Insert');  
INSERT INTO emp (empno,ename) VALUES (12,'ABC');  
COMMIT;  
END;   
  
  
INSERT INTO isc\_st VALUES (20,'B');  
  
select \* from emp  
  
delete from isc\_st  
  
select \* from isc\_st  
`  
UPDATE isc\_st set name='B'  
  
DELETE isc\_St;  
  
CREATE OR REPLACE TRIGGER isc\_St\_trg\_afd  
AFTER DELETE ON isc\_st  
FOR EACH ROW  
DECLARE  
PRAGMA AUTONOMOUS\_TRANSACTION;  
BEGIN  
**DBMS\_OUTPUT.put\_line**('After Insert');  
INSERT INTO emp (empno,ename) VALUES (:OLD.n,:OLD.name);  
COMMIT;  
END;   
  
  
CREATE OR REPLACE VIEW isc\_v as select e.ename,d.dname,e.empno,d.deptno from emp e,dept d wheree.deptno=d.deptno  
  
select \* from isc\_v  
  
INSERT INTO isc\_v values('A','DSD',123,70)  
  
  
CREATE OR REPLACE TRIGGER isc\_instdof  
INSTEAD OF INSERT ON isc\_v  
DECLARE  
PRAGMA AUTONOMOUS\_TRANSACTION;  
BEGIN  
INSERT INTO dept (deptno,dname) VALUES (:NEW.deptno,:NEW.dname);  
INSERT INTO emp (empno,ename,deptno) VALUES (:NEW.empno,:NEW.ename,:NEW.deptno);  
COMMIT;  
END;  
  
select \* from emp  
  
select \* from dept    
  
  
DECLARE  
TYPE ld IS TABLE OF NUMBER INDEX BY BINARY\_INTEGER;  
l\_num LD;  
BEGIN  
FOR i IN 1..100  
LOOP  
l\_num(i):=i+100;  
END LOOP;  
  
FOR i IN l\_num.FIRST..l\_num.LAST  
LOOP  
**DBMS\_OUTPUT.put\_line**(l\_num(i));  
END LOOP;  
  
END;  
  
  
  
DECLARE  
TYPE ld IS TABLE OF NUMBER INDEX BY BINARY\_INTEGER;  
l\_num LD;  
BEGIN  
l\_num(900) := 780;  
l\_num(-50):=200;  
**DBMS\_OUTPUT.put\_line**(l\_num(900));  
END;  
  
  
DECLARE  
TYPE lr IS RECORD (BATCH\_NAME VARCHAR2(30),NO\_OF\_STUDENTS NUMBER);  
TYPE ld IS TABLE OF NUMBER INDEX BY BINARY\_INTEGER;  
l\_num LD;  
l\_record LR;  
BEGIN  
l\_record.batch\_name :='Sun Rises';  
l\_record.no\_of\_students :=10;   
l\_record.batch\_name :='Mumbai Indians';  
l\_record.no\_of\_students :=14;   
**DBMS\_OUTPUT.put\_line**(l\_record.batch\_name);  
END;  
  
  
DECLARE  
TYPE lr IS RECORD (BATCH\_NAME VARCHAR2(30),NO\_OF\_STUDENTS NUMBER);  
TYPE ld IS TABLE OF lr INDEX BY BINARY\_INTEGER;  
l\_num LD;  
l\_record LR;  
BEGIN  
l\_num(10).batch\_name :='Immense Rises';  
l\_num(101).batch\_name :='Sun Rises';  
l\_num(101).no\_of\_students :=10;   
l\_num(50).batch\_name :='Mumbai Indians';  
l\_num(51).no\_of\_students :=14;   
  
FOR i IN l\_num.FIRST..l\_num.LAST  
LOOP  
   BEGIN  
   **DBMS\_OUTPUT.put\_line**('Batch\_name:='||l\_num(i).batch\_name||' And No Of Stdunts := '|| l\_num(i).no\_of\_students);  
   EXCEPTION  
   WHEN NO\_DATA\_FOUND THEN  
     null;  
   END;    
END LOOP;  
**DBMS\_OUTPUT.put\_line**(l\_num(10).batch\_name);  
EXCEPTION  
WHEN NO\_DATA\_FOUND THEN  
   null;  
END;