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

     EMPNO ENAME      JOB              MGR HIREDATE          SAL        COMM
DEPTNO S
---------- ---------- --------- ---------- --------- ---------- ----------
---------- -
ADDRESS

--------------------

       100 ALI        SALESMAN                                100
20 A


       125 ALI        SALESMAN                               1000
30 A


      7369 SMITH      CLERK           7902 17-DEC-80         800
20 I


      7499 ALLEN      SALESMAN        7698 20-FEB-81        1600        300
30 A


      7521 WARD       SALESMAN        7698 22-FEB-81        1250        500
30 A


      7566 JONES      MANAGER         7839 02-APR-81        2975
20 A


      7654 MARTIN     SALESMAN        7698 28-SEP-81        1250       1400
30 A


      7698 BLAKE      MANAGER         7839 01-MAY-81        2850
30 A


      7782 CLARK      MANAGER         7839 09-JUN-81        2450
10 A
```

```
      7788 SCOTT      ANALYST         7566 19-APR-87       3000
20 A


      7839 KING       PRESIDENT            17-NOV-81       5000
10


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


      7876 ADAMS      CLERK           7788 23-MAY-87       1100
20 I


      7900 JAMES      CLERK           7698 03-DEC-81        950
30 I


      7902 FORD       ANALYST         7566 03-DEC-81      45666
20


      7934 MILLER     CLERK           7782 23-JAN-85       1300
10 I



16 rows selected.

SQL>
SQL> ALTER TABLE EMP DROP COLULMN ADDRESS;
ALTER TABLE EMP DROP COLULMN ADDRESS
                    *
ERROR at line 1:
ORA-00905: missing keyword


SQL> ED
Wrote file afiedt.buf

  1* ALTER TABLE EMP DROP COLUMN ADDRESS
SQL> /

Table altered.
```

```
SQL> SELECT * FROM EMP;

     EMPNO ENAME      JOB              MGR HIREDATE          SAL        COMM
DEPTNO S
---------- ---------- --------- ---------- --------- ---------- ----------
---------- -
       100 ALI        SALESMAN                              100
20 A
       125 ALI        SALESMAN                             1000
30 A
      7369 SMITH      CLERK           7902 17-DEC-80        800
20 I
      7499 ALLEN      SALESMAN        7698 20-FEB-81       1600         300
30 A
      7521 WARD       SALESMAN        7698 22-FEB-81       1250         500
30 A
      7566 JONES      MANAGER         7839 02-APR-81       2975
20 A
      7654 MARTIN     SALESMAN        7698 28-SEP-81       1250        1400
30 A
      7698 BLAKE      MANAGER         7839 01-MAY-81       2850
30 A
      7782 CLARK      MANAGER         7839 09-JUN-81       2450
10 A
      7788 SCOTT      ANALYST         7566 19-APR-87       3000
20 A
      7839 KING       PRESIDENT            17-NOV-81       5000
10
      7844 TURNER     SALESMAN        7698 08-SEP-81       1500           0
30 A
      7876 ADAMS      CLERK           7788 23-MAY-87       1100
20 I
      7900 JAMES      CLERK           7698 03-DEC-81        950
30 I
      7902 FORD       ANALYST         7566 03-DEC-81      45666
20
      7934 MILLER     CLERK           7782 23-JAN-85       1300
10 I

16 rows selected.

SQL> ALTER TABLE EMP DROP COLUMN STATUS;

Table altered.

SQL> SELECT * FROM EMP;

     EMPNO ENAME      JOB              MGR HIREDATE          SAL        COMM
DEPTNO
---------- ---------- --------- ---------- --------- ---------- ----------
----------
       100 ALI        SALESMAN                              100
20
       125 ALI        SALESMAN                             1000
30
      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      45666
20
      7934 MILLER     CLERK           7782 23-JAN-85       1300
10

16 rows selected.

SQL>
SQL>
SQL> DELETE FROM EMP
  2  WHERE HIREDATE IS NULL;

2 rows deleted.

SQL> COMMIT;

Commit complete.

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      45666
20
      7934 MILLER     CLERK           7782 23-JAN-85       1300
10

14 rows selected.

SQL>
SQL>
SQL>
SQL> SELECT * FROM DEP;
SELECT * FROM DEP
              *
ERROR at line 1:
ORA-00942: table or view does not exist


SQL> ED
Wrote file afiedt.buf

  1* SELECT * FROM DEPT
SQL> /

    DEPTNO DNAME          LOC

---------- -------------- -------------

        50 HR             KARACHI

        60 NEW HR         LHR

        10 ACCOUNTING     NEW YORK

        20 RESEARCH       DALLAS

        30 SALES          CHICAGO

        40 OPERATIONS     BOSTON


6 rows selected.

SQL> ED
Wrote file afiedt.buf

  1  BEGIN
  2  ---------TESTING------------
  3  FOR   I IN (SELECT * FROM DEPT) LOOP;
  4  &D(I.DEPTNO||'   '||I.DNAME);
  5   FOR  E IN (SELECT * FROM EMP WHERE DEPTNO=I.DEPTNO);
  6  &D(E.DEPTNO||'   '||E.ENAME||'   '||E.JOB);
  7  END LOOP;
  8  END LOOP;
  9* END;
 10  /
FOR   I IN (SELECT * FROM DEPT) LOOP;
                                    *
ERROR at line 3:
ORA-06550: line 3, column 37:
PLS-00103: Encountered the symbol ";" when expecting one of the following:
begin case declare exit for goto if loop mod null pragma
raise return select update while with <an identifier>
```

```
<a double-quoted delimited-identifier> <a bind variable> <<
close current delete fetch lock insert open rollback
savepoint set sql execute commit forall merge pipe
The symbol "exit" was substituted for ";" to continue.
ORA-06550: line 5, column 53:
PLS-00103: Encountered the symbol ";" when expecting one of the following:
loop
ORA-06550: line 9, column 1:
PLS-00103: Encountered the symbol "END"


SQL> ED
Wrote file afiedt.buf

  1  BEGIN
  2  ---------TESTING------------
  3  FOR   I IN (SELECT * FROM DEPT) LOOP
  4  &D(I.DEPTNO||'    '||I.DNAME);
  5   FOR  E IN (SELECT * FROM EMP WHERE DEPTNO=I.DEPTNO) LOOP
  6  &D(E.DEPTNO||'    '||E.ENAME||'    '||E.JOB);
  7  END LOOP;
  8  END LOOP;
  9* END;
SQL> /
50   HR

60   NEW HR

10   ACCOUNTING

10   CLARK   MANAGER

10   KING   PRESIDENT

10   MILLER   CLERK

20   RESEARCH

20   SMITH   CLERK

20   JONES   MANAGER

20   SCOTT   ANALYST

20   ADAMS   CLERK

20   FORD   ANALYST

30   SALES

30   ALLEN   SALESMAN

30   WARD   SALESMAN

30   MARTIN   SALESMAN

30   BLAKE   MANAGER

30   TURNER   SALESMAN

30   JAMES   CLERK

40   OPERATIONS
```

```
PL/SQL procedure successfully completed.

SQL>
SQL>
SQL>
SQL> ED
Wrote file afiedt.buf

  1  BEGIN
  2  ---------TESTING------------
  3  -----------FROM MASTER TABLE
  4  FOR   I IN (SELECT * FROM DEPT) LOOP
  5  &D(I.DEPTNO||'   '||I.DNAME);
  6  &D('========================');
  7   FOR  E IN (SELECT * FROM EMP WHERE DEPTNO=I.DEPTNO) LOOP
  8  &D(E.DEPTNO||'   '||E.ENAME||'   '||E.JOB);
  9  END LOOP;
 10  END LOOP;
 11* END;
 12  /
50   HR

=========================

60   NEW HR

=========================

10   ACCOUNTING

=========================

10   CLARK   MANAGER

10   KING   PRESIDENT

10   MILLER   CLERK

20   RESEARCH

=========================

20   SMITH   CLERK

20   JONES   MANAGER

20   SCOTT   ANALYST

20   ADAMS   CLERK

20   FORD   ANALYST

30   SALES

=========================

30   ALLEN   SALESMAN

30   WARD   SALESMAN

30   MARTIN   SALESMAN
```

```
30   BLAKE   MANAGER

30   TURNER   SALESMAN

30   JAMES   CLERK

40   OPERATIONS

==========================


PL/SQL procedure successfully completed.

SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL> CL SCR
SQL> ED
Wrote file afiedt.buf

  1  DECLARE (O)
  2  BEGIN   (M)
  3  EXCEPTION  (O)
  4* END   (M)
  5  .
SQL>
SQL> SELECT 5+5 FROM DUAL;

       5+5

----------

        10


SQL> DECLARE (O)
  2  BEGIN   (M)
  3  EXCEPTION  (O)
  4  END   (M)
  5  .
SQL> ED
Wrote file afiedt.buf

  1  /*
  2  DECLARE (O)         (*) PROCEDURE    (*) FUNCTION
  3  BEGIN   (M)
  4  EXCEPTION  (O)
  5  END   (M)
  6* */
  7  /
*/
 *
ERROR at line 6:
ORA-00900: invalid SQL statement


SQL>
SQL>
SQL>
```

```
SQL>
SQL>
SQL>
SQL> SELECT ENAME FROM EMP WHERE EMPNO=7839;

ENAME

----------

KING


SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL> ED
Wrote file afiedt.buf

  1  BEGIN
  2  SELECT ENAME FROM EMP WHERE EMPNO=7839
  3* END;
SQL> /
END;
*
ERROR at line 3:
ORA-06550: line 2, column 40:
PL/SQL: ORA-00933: SQL command not properly ended
ORA-06550: line 2, column 1:
PL/SQL: SQL Statement ignored
ORA-06550: line 3, column 4:
PLS-00103: Encountered the symbol "end-of-file" when expecting one of the following:

begin case declare end exception exit for goto if loop mod
null pragma raise return select update while with
<an identifier> <a double-quoted de


SQL>
SQL>
SQL>
SQL>
SQL> ED
Wrote file afiedt.buf

  1  DECLARE
  2  V_ENAME VARCHAR2(20);
  3  BEGIN
  4  SELECT ENAME INTO V_ENAME FROM EMP WHERE EMPNO=7839;
  5  &D(V_ENAME);
  6* END;
  7  /
KING


PL/SQL procedure successfully completed.

SQL>
SQL>
```

```
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
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      45666
20
      7934 MILLER     CLERK           7782 23-JAN-85       1300
10

14 rows selected.

SQL> L
  1* SELECT * FROM EMP
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL> /

     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      45666
20
      7934 MILLER     CLERK           7782 23-JAN-85       1300
10

14 rows selected.

SQL> CLEAR BUFFER
buffer cleared
SQL> L
SP2-0223: No lines in SQL buffer.
SQL> /
SP2-0103: Nothing in SQL buffer to run.
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL> ED
SP2-0107: Nothing to save.
SQL> BEGIN
  2  .
```

```
SQL> ED
Wrote file afiedt.buf

  1  BEGIN
  2  DBMS_OUTPUT.PUT_LINE('ORACLE');
  3* END;
  4  /
ORACLE


PL/SQL procedure successfully completed.

SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL> ED
Wrote file afiedt.buf

  1  BEGIN
  2  DBMS_OUTPUT.PUT_LINE('ORACLE');
  3  DBMS_OUTPUT.PUT_LINE('SQL');
  4* END;
SQL> /
ORACLE

SQL


PL/SQL procedure successfully completed.

SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL> ED
Wrote file afiedt.buf

  1  BEGIN
  2  DBMS_OUTPUT.PUT_LINE('ORACLE','SQL');
  3  ----DBMS_OUTPUT.PUT_LINE('SQL');
  4* END;
SQL> /
DBMS_OUTPUT.PUT_LINE('ORACLE','SQL');
*
ERROR at line 2:
ORA-06550: line 2, column 1:
PLS-00306: wrong number or types of arguments in call to 'PUT_LINE'
ORA-06550: line 2, column 1:
PL/SQL: Statement ignored
```

```
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL> ED
Wrote file afiedt.buf

  1  BEGIN
  2  DBMS_OUTPUT.PUT_LINE('ORACLE'||'SQL');
  3  ----DBMS_OUTPUT.PUT_LINE('SQL');
  4* END;
SQL> /
ORACLESQL


PL/SQL procedure successfully completed.

SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL> ED
Wrote file afiedt.buf

  1  BEGIN
  2  DBMS_OUTPUT.PUT_LINE('ORACLE'||'    '||'SQL');
  3  ----DBMS_OUTPUT.PUT_LINE('SQL');
  4* END;
SQL> /
ORACLE    SQL


PL/SQL procedure successfully completed.

SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
```

```
SQL>
SQL>
SQL>
SQL>
SQL> ED
Wrote file afiedt.buf

  1  BEGIN
  2  DBMS_OUTPUT.PUT_LINE('ORACLE'||CHR(10)||'SQL');
  3  ----DBMS_OUTPUT.PUT_LINE('SQL');
  4* END;
SQL> /
ORACLE
SQL


PL/SQL procedure successfully completed.

SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL> SELECT CHR(65) FROM DUAL;

C

-

A


SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL> BEGIN
  2  DBMS_OUTPUT.PUT_LINE('ORACLE'||CHR(10)||'SQL');
  3  ----DBMS_OUTPUT.PUT_LINE('SQL');
  4  END;
  5
  6  .ED
  7  .
SQL> ED
Wrote file afiedt.buf

  1  BEGIN
  2  DBMS_OUTPUT.PUT_LINE('ORACLE'||CHR(10)||'SQL');
```

```
  3  DBMS_OUTPUT.PUT_LINE('ORACLE'||CHR(10)||'PLSQL');
  4  ---DBMS_OUTPUT.PUT_LINE('SQL');
  5* END;
  6  /
ORACLE
SQL

ORACLE
PLSQL


PL/SQL procedure successfully completed.

SQL> ED
Wrote file afiedt.buf

  1  BEGIN
  2
DBMS_OUTPUT.PUT_LINE('ORACLE'||CHR(10)||'SQL'||CHR(10)||'ORACLE'||CHR(10)||'PLSQL');
  3  ----DBMS_OUTPUT.PUT_LINE('ORACLE'||CHR(10)||'PLSQL');
  4  ---DBMS_OUTPUT.PUT_LINE('SQL');
  5* END;
SQL> /
ORACLE
SQL
ORACLE
PLSQL

PL/SQL procedure successfully completed.

SQL>
SQL>
SQL>
SQL>
SQL> ED
Wrote file afiedt.buf

  1  BEGIN
  2
DBMS_OUTPUT.PUT_LINE('ORACLE'||CHR(10)||'SQL'||CHR(10)||'ORACLE'||CHR(10)||'PLSQL');
  3* END;
SQL> /
ORACLE
SQL
ORACLE
PLSQL

PL/SQL procedure successfully completed.

SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL> ED
Wrote file afiedt.buf
```

```
  1  DECLARE
  2  A NUMBER := 10;
  3  B NUMBER :=5;
  4  TOTAL NUMBER :=0;
  5  BEGIN
  6  TOTAL := A + B ;
  7  DBMS_OUTPUT.PUT_LINE('VALUE OF A IS...'||A);
  8  DBMS_OUTPUT.PUT_LINE('VALUE OF B IS...'||B);
  9  DBMS_OUTPUT.PUT_LINE('VALUE OF TOTAL IS...'||TOTAL);
 10* END;
 11  /
VALUE OF A IS...10

VALUE OF B IS...5

VALUE OF TOTAL IS...15


PL/SQL procedure successfully completed.

SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL> ED
Wrote file afiedt.buf

  1  DECLARE
  2  A NUMBER := 10;
  3  B NUMBER :=5;
  4  TOTAL NUMBER :=0;
  5  TOTAL := A + B ;
  6  BEGIN
  7  DBMS_OUTPUT.PUT_LINE('VALUE OF A IS...'||A);
  8  DBMS_OUTPUT.PUT_LINE('VALUE OF B IS...'||B);
  9  DBMS_OUTPUT.PUT_LINE('VALUE OF TOTAL IS...'||TOTAL);
 10* END;
 11  /
TOTAL := A + B ;
      *
ERROR at line 5:
ORA-06550: line 5, column 7:
PLS-00103: Encountered the symbol "=" when expecting one of the following:
constant exception <an identifier>
<a double-quoted delimited-identifier> table LONG_ double ref
char time timestamp interval date binary national character
nchar


SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL> ED
Wrote file afiedt.buf

  1  DECLARE
```

```
  2  A NUMBER := 10;
  3  B NUMBER :=5;
  4  TOTAL NUMBER := A + B;
  5  BEGIN
  6  DBMS_OUTPUT.PUT_LINE('VALUE OF A IS...'||A);
  7  DBMS_OUTPUT.PUT_LINE('VALUE OF B IS...'||B);
  8  DBMS_OUTPUT.PUT_LINE('VALUE OF TOTAL IS...'||TOTAL);
  9* END;
 10  /
VALUE OF A IS...10

VALUE OF B IS...5

VALUE OF TOTAL IS...15


PL/SQL procedure successfully completed.

SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL> ED
Wrote file afiedt.buf

  1  DECLARE
  2  A NUMBER := 10;
  3  B NUMBER :=5;
  4  TOTAL NUMBER := 0;
  5  C NUMBER:=20;
  6  BEGIN
  7  TOTAL := A + B ;
  8  DBMS_OUTPUT.PUT_LINE('VALUE OF A IS...'||A);
  9  DBMS_OUTPUT.PUT_LINE('VALUE OF B IS...'||B);
 10  DBMS_OUTPUT.PUT_LINE('VALUE OF TOTAL IS...'||TOTAL);
 11  TOTAL := A + B + C;
 12  DBMS_OUTPUT.PUT_LINE('VALUE OF A IS...'||A);
 13  DBMS_OUTPUT.PUT_LINE('VALUE OF B IS...'||B);
 14  DBMS_OUTPUT.PUT_LINE('VALUE OF C IS...'||C);
 15  DBMS_OUTPUT.PUT_LINE('VALUE OF TOTAL IS...'||TOTAL);
 16* END;
 17  /
VALUE OF A IS...10

VALUE OF B IS...5

VALUE OF TOTAL IS...15

VALUE OF A IS...10

VALUE OF B IS...5

VALUE OF C IS...20

VALUE OF TOTAL IS...35


PL/SQL procedure successfully completed.

SQL>
SQL>
```

```
SQL>
SQL>
SQL>
SQL>
SQL>
SQL> ED
Wrote file afiedt.buf

  1  DECLARE
  2  A NUMBER := 10;
  3  B NUMBER :=5;
  4  TOTAL NUMBER := 0;
  5  C NUMBER:=20;
  6  BEGIN
  7  TOTAL := A + B ;
  8  DBMS_OUTPUT.PUT_LINE('VALUE OF A IS...'||A||CHR(10)||'VALUE OF B IS...'||B);
  9  ---DBMS_OUTPUT.PUT_LINE('VALUE OF B IS...'||B);
 10  DBMS_OUTPUT.PUT_LINE('VALUE OF TOTAL IS...'||TOTAL);
 11  TOTAL := A + B + C;
 12  DBMS_OUTPUT.PUT_LINE('VALUE OF A IS...'||A);
 13  DBMS_OUTPUT.PUT_LINE('VALUE OF B IS...'||B);
 14  DBMS_OUTPUT.PUT_LINE('VALUE OF C IS...'||C);
 15  DBMS_OUTPUT.PUT_LINE('VALUE OF TOTAL IS...'||TOTAL);
 16* END;
SQL> /
VALUE OF A IS...10
VALUE OF B IS...5
VALUE OF TOTAL IS...15

VALUE OF A IS...10

VALUE OF B IS...5

VALUE OF C IS...20

VALUE OF TOTAL IS...35


PL/SQL procedure successfully completed.

SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL> ED
Wrote file afiedt.buf

  1  DECLARE
  2  A NUMBER := &ENTER_NO1;
  3  B NUMBER :=&ENTER_NO2;
  4  TOTAL NUMBER := 0;
  5  C NUMBER:=&ENTER_NO3;
  6  BEGIN
  7  TOTAL := A + B ;
  8  DBMS_OUTPUT.PUT_LINE('VALUE OF A IS...'||A||CHR(10)||'VALUE OF B IS...'||B);
  9  ---DBMS_OUTPUT.PUT_LINE('VALUE OF B IS...'||B);
 10  DBMS_OUTPUT.PUT_LINE('VALUE OF TOTAL IS...'||TOTAL);
 11  TOTAL := A + B + C;
 12  DBMS_OUTPUT.PUT_LINE('VALUE OF A IS...'||A);
```

```
 13  DBMS_OUTPUT.PUT_LINE('VALUE OF B IS...'||B);
 14  DBMS_OUTPUT.PUT_LINE('VALUE OF C IS...'||C);
 15  DBMS_OUTPUT.PUT_LINE('VALUE OF TOTAL IS...'||TOTAL);
 16* END;
SQL> /
Enter value for enter_no1: 5
Enter value for enter_no2: 6
Enter value for enter_no3: 9
VALUE OF A IS...5
VALUE OF B IS...6
VALUE OF TOTAL IS...11

VALUE OF A IS...5

VALUE OF B IS...6

VALUE OF C IS...9

VALUE OF TOTAL IS...20


PL/SQL procedure successfully completed.

SQL> /
Enter value for enter_no1: 45
Enter value for enter_no2: 265
Enter value for enter_no3: 445
VALUE OF A IS...45
VALUE OF B IS...265
VALUE OF TOTAL IS...310

VALUE OF A IS...45

VALUE OF B IS...265

VALUE OF C IS...445

VALUE OF TOTAL IS...755


PL/SQL procedure successfully completed.

SQL>
SQL>
SQL> ED
Wrote file afiedt.buf

  1  DECLARE
  2  A NUMBER := &ENTER_NO1;
  3  B NUMBER :=&ENTER_NO2;
  4  TOTAL NUMBER := 0;
  5  C NUMBER:=&ENTER_NO3;
  6  BEGIN
  7  TOTAL := A + B ;
  8  &D('VALUE OF A IS...'||A||CHR(10)||'VALUE OF B IS...'||B);
  9  ---DBMS_OUTPUT.PUT_LINE('VALUE OF B IS...'||B);
 10  DBMS_OUTPUT.PUT_LINE('VALUE OF TOTAL IS...'||TOTAL);
 11  TOTAL := A + B + C;
 12  DBMS_OUTPUT.PUT_LINE('VALUE OF A IS...'||A);
 13  DBMS_OUTPUT.PUT_LINE('VALUE OF B IS...'||B);
 14  DBMS_OUTPUT.PUT_LINE('VALUE OF C IS...'||C);
 15  DBMS_OUTPUT.PUT_LINE('VALUE OF TOTAL IS...'||TOTAL);
 16* END;
```

```
SQL> /
Enter value for enter_no1: 5
Enter value for enter_no2: 5
Enter value for enter_no3: 6
VALUE OF A IS...5
VALUE OF B IS...5
VALUE OF TOTAL IS...10

VALUE OF A IS...5

VALUE OF B IS...5

VALUE OF C IS...6

VALUE OF TOTAL IS...16


PL/SQL procedure successfully completed.

SQL>
SQL>
SQL> DEFINE
DEFINE _DATE            = "02-FEB-13" (CHAR)
DEFINE _CONNECT_IDENTIFIER = "orcl" (CHAR)
DEFINE _USER            = "SCOTT" (CHAR)
DEFINE _PRIVILEGE       = "" (CHAR)
DEFINE _SQLPLUS_RELEASE = "1002000100" (CHAR)
DEFINE _EDITOR          = "Notepad" (CHAR)
DEFINE _O_VERSION       = "Oracle Database 10g Enterprise Edition Release 10.2.0.1.0
- Production
With the Partitioning, OLAP and Data Mining options" (CHAR)
DEFINE _O_RELEASE       = "1002000100" (CHAR)
DEFINE D                = "DBMS_OUTPUT.PUT_LINE" (CHAR)
DEFINE R                = "RAISE_APPLICATION_ERROR" (CHAR)
DEFINE ERRM             = "INITCAP(SUBSTR(SQLERRM,INSTR(SQLERRM,':',1)+1))" (CHAR)
DEFINE _RC              = "1" (CHAR)
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL> ED
Wrote file afiedt.buf

  1  DECLARE
  2  A NUMBER := &ENTER_NO1;
  3  B NUMBER :=&ENTER_NO2;
  4  TOTAL NUMBER := 0;
  5  C NUMBER:=&ENTER_NO3;
  6  BEGIN
  7  TOTAL := A + B ;
  8  &E('VALUE OF A IS...'||A||CHR(10)||'VALUE OF B IS...'||B);
  9  ---DBMS_OUTPUT.PUT_LINE('VALUE OF B IS...'||B);
 10  DBMS_OUTPUT.PUT_LINE('VALUE OF TOTAL IS...'||TOTAL);
 11  TOTAL := A + B + C;
 12  DBMS_OUTPUT.PUT_LINE('VALUE OF A IS...'||A);
 13  DBMS_OUTPUT.PUT_LINE('VALUE OF B IS...'||B);
 14  DBMS_OUTPUT.PUT_LINE('VALUE OF C IS...'||C);
 15  DBMS_OUTPUT.PUT_LINE('VALUE OF TOTAL IS...'||TOTAL);
 16* END;
SQL> /
```

```
Enter value for enter_no1: 5
Enter value for enter_no2: 5
Enter value for enter_no3: 6
Enter value for e:
('VALUE OF A IS...'||A||CHR(10)||'VALUE OF B IS...'||B);
*
ERROR at line 8:
ORA-06550: line 8, column 1:
PLS-00103: Encountered the symbol "(" when expecting one of the following:
begin case declare end exception exit for goto if loop mod
null pragma raise return select update while with
<an identifier> <a double-quoted delimited-identifier>
<a bind variable> << close current delete fetch lock insert
open rollback savepoint set sql execute commit forall merge
pipe


SQL>
SQL>
SQL>
SQL>
SQL>
SQL> ED
Wrote file afiedt.buf

  1  DECLARE
  2  A NUMBER := &ENTER_NO1;
  3  B NUMBER :=&ENTER_NO2;
  4  TOTAL NUMBER := 0;
  5  C NUMBER:=&ENTER_NO3;
  6  BEGIN
  7  TOTAL := A + B ;
  8  &D('VALUE OF A IS...'||A||CHR(10)||'VALUE OF B IS...'||B);
  9  ---DBMS_OUTPUT.PUT_LINE('VALUE OF B IS...'||B);
 10  DBMS_OUTPUT.PUT_LINE('VALUE OF TOTAL IS...'||TOTAL);
 11  TOTAL := A + B + C;
 12  DBMS_OUTPUT.PUT_LINE('VALUE OF A IS...'||A);
 13  DBMS_OUTPUT.PUT_LINE('VALUE OF B IS...'||B);
 14  DBMS_OUTPUT.PUT_LINE('VALUE OF C IS...'||C);
 15  DBMS_OUTPUT.PUT_LINE('VALUE OF TOTAL IS...'||TOTAL);
 16* END;
SQL> /
Enter value for enter_no1: 5
Enter value for enter_no2: 6
Enter value for enter_no3: 33
VALUE OF A IS...5
VALUE OF B IS...6
VALUE OF TOTAL IS...11

VALUE OF A IS...5

VALUE OF B IS...6

VALUE OF C IS...33

VALUE OF TOTAL IS...44


PL/SQL procedure successfully completed.

SQL>
SQL>
SQL>
```

```
SQL>
SQL>
SQL> ED
Wrote file afiedt.buf

  1  DECLARE
  2  --------------------DECLARATION AREA-----------------
  3  S_NAME VARCHAR2(20):='&NAME';
  4  V_CLASS VARCHAR2(20):='&CLASS';
  5  M_MARK NUMBER :=&MATH_MARKS;
  6  S_MARK NUMBER :=&STADIES;
  7  P_MARK NUMBER :=&PHY_MARK;
  8  U_MARK NUMBER :=&URDU;
  9  E_MARK NUMBER :=&ENGLISH;
 10  TOTAL NUMBER :=0;
 11  BEGIN
 12  ------------------------CALCULATION AREA--------------
 13  TOTAL := M_MARK + S_MARK + P_MARK + U_MARK + E_MARK;
 14  ------------------------DISPLAY AREA------------------
 15  &D(' MARKS SHEET ');
 16  &D('==============='||CHR(10));
 17  &D('STUDENT NAME '||S_NAME);
 18  &D('STUDENT CLASS '||V_CLASS);
 19  &D('MATH MARKS  '||M_MARK);
 20  &D('PHYSICS MARKS  '||P_MARK);
 21  &D('URDU MARKS  '||U_MARK);
 22  &D('ENGLISH MARKS  '||E_MARK);
 23  &D('STUDIES MARKS  '||S_MARK||CHR(10));
 24  &D('TOTAL MARKS .,...'||TOTAL);
 25* END;
 26  /
Enter value for name: ALI
Enter value for class: X
Enter value for math_marks: 56
Enter value for stadies: 35
Enter value for phy_mark: 68
Enter value for urdu: 88
Enter value for english: 48
MARKS SHEET

==============


STUDENT NAME ALI

STUDENT CLASS X

MATH MARKS  56

PHYSICS MARKS  68

URDU MARKS  88

ENGLISH MARKS  48

STUDIES MARKS  35

TOTAL MARKS .,...295


PL/SQL procedure successfully completed.

SQL>
```

```
SQL>
SQL>
SQL>
SQL> ED
Wrote file afiedt.buf

  1  DECLARE
  2  --------------------DECLARATION AREA-----------------
  3  S_NAME VARCHAR2(20):='&NAME';
  4  V_CLASS VARCHAR2(20):='&CLASS';
  5  M_MARK NUMBER :=&MATH_MARKS;
  6  S_MARK NUMBER :=&STADIES;
  7  P_MARK NUMBER :=&PHY_MARK;
  8  U_MARK NUMBER :=&URDU;
  9  E_MARK NUMBER :=&ENGLISH;
 10  TOTAL NUMBER :=0;
 11  PER NUMBER := 0;
 12  BEGIN
 13  ------------------------CALCULATION AREA--------------
 14  TOTAL := M_MARK + S_MARK + P_MARK + U_MARK + E_MARK;
 15  PER :=  TOTAL * 100/500;
 16  ------------------------DISPLAY AREA------------------
 17  &D(' MARKS SHEET ');
 18  &D('==============='||CHR(10));
 19  &D('STUDENT NAME '||S_NAME);
 20  &D('STUDENT CLASS '||V_CLASS);
 21  &D('MATH MARKS  '||M_MARK);
 22  &D('PHYSICS MARKS  '||P_MARK);
 23  &D('URDU MARKS  '||U_MARK);
 24  &D('ENGLISH MARKS  '||E_MARK);
 25  &D('STUDIES MARKS  '||S_MARK||CHR(10));
 26  &D('TOTAL MARKS .,...'||TOTAL);
 27  &D('PERCENTAGE  MARKS .,...'||PER);
 28* END;
 29  /
Enter value for name: 58
Enter value for class: X
Enter value for math_marks: 69
Enter value for stadies: 68
Enter value for phy_mark: 89
Enter value for urdu: 68
Enter value for english: 45
MARKS SHEET

==============


STUDENT NAME 58

STUDENT CLASS X

MATH MARKS  69

PHYSICS MARKS  89

URDU MARKS  68

ENGLISH MARKS  45

STUDIES MARKS  68

TOTAL MARKS .,...339
```

```
PERCENTAGE  MARKS .,...67.8


PL/SQL procedure successfully completed.

SQL>
SQL> ED
Wrote file afiedt.buf

  1  DECLARE
  2  --------------------DECLARATION AREA-----------------
  3  S_NAME VARCHAR2(20):='&NAME';
  4  V_CLASS VARCHAR2(20):='&CLASS';
  5  M_MARK NUMBER :=&MATH_MARKS;
  6  S_MARK NUMBER :=&STADIES;
  7  P_MARK NUMBER :=&PHY_MARK;
  8  U_MARK NUMBER :=&URDU;
  9  E_MARK NUMBER :=&ENGLISH;
 10  TOTAL NUMBER :=0;
 11  PER NUMBER := 0;
 12  BEGIN
 13  ------------------------CALCULATION AREA--------------
 14  TOTAL := M_MARK + S_MARK + P_MARK + U_MARK + E_MARK;
 15  PER :=  TOTAL * 100/500;
 16  ------------------------DISPLAY AREA------------------
 17  &D(' MARKS SHEET ');
 18  &D('==============='||CHR(10));
 19  &D('STUDENT NAME '||S_NAME);
 20  &D('STUDENT CLASS '||V_CLASS);
 21  &D('MATH MARKS  '||M_MARK);
 22  &D('PHYSICS MARKS  '||P_MARK);
 23  &D('URDU MARKS  '||U_MARK);
 24  &D('ENGLISH MARKS  '||E_MARK);
 25  &D('STUDIES MARKS  '||S_MARK||CHR(10));
 26  &D('TOTAL MARKS .,...'||TOTAL);
 27  &D('PERCENTAGE  MARKS .,...'||PER||'%');
 28* END;
SQL> /
Enter value for name: ALI
Enter value for class: X
Enter value for math_marks: 58
Enter value for stadies: 69
Enter value for phy_mark: 58
Enter value for urdu: 57
Enter value for english: 58
MARKS SHEET

==============


STUDENT NAME ALI

STUDENT CLASS X

MATH MARKS  58

PHYSICS MARKS  58

URDU MARKS  57

ENGLISH MARKS  58

STUDIES MARKS  69
```

```
TOTAL MARKS .,...300

PERCENTAGE  MARKS .,...60%


PL/SQL procedure successfully completed.

SQL> ED
Wrote file afiedt.buf

  1  DECLARE
  2  --------------------DECLARATION AREA-----------------
  3  S_NAME VARCHAR2(20):='&NAME';
  4  V_CLASS VARCHAR2(20):='&CLASS';
  5  M_MARK NUMBER :=&MATH_MARKS;
  6  S_MARK NUMBER :=&STADIES;
  7  P_MARK NUMBER :=&PHY_MARK;
  8  U_MARK NUMBER :=&URDU;
  9  E_MARK NUMBER :=&ENGLISH;
 10  TOTAL NUMBER :=0;
 11  PER NUMBER := 0;
 12  BEGIN
 13  ------------------------CALCULATION AREA--------------
 14  TOTAL := M_MARK + S_MARK + P_MARK + U_MARK + E_MARK;
 15  PER :=  TOTAL * 100/500;
 16  ------------------------DISPLAY AREA------------------
 17  &D(' MARKS SHEET ');
 18  &D('==============='||CHR(10));
 19  &D('STUDENT NAME '||S_NAME);
 20  &D('STUDENT CLASS '||V_CLASS);
 21  &D('MATH MARKS  '||M_MARK);
 22  &D('PHYSICS MARKS  '||P_MARK);
 23  &D('URDU MARKS  '||U_MARK);
 24  &D('ENGLISH MARKS  '||E_MARK);
 25  &D('STUDIES MARKS  '||S_MARK||CHR(10));
 26  &D('TOTAL MARKS .,...'||TOTAL);
 27  &D('PERCENTAGE  MARKS .,...'||PER||'%');
 28* END;
SQL>
SQL>
SQL>
SQL>
SQL>
SQL> CREATE TABLE STD(ROLL_NO NUMBER(4),SNAME VARCHAR2(20)
  2  .
SQL> ED
Wrote file afiedt.buf

  1  CREATE TABLE STD
  2  (
  3  ROLL_NO NUMBER(4),
  4  SNAME VARCHAR2(20),
  5  CLASS_NM VARCHAR2(20),
  6  F_ENG NUMBER(3),
  7  F_PHY NUMBER(3),
  8  F_URD NUMBER(3),
  9  F_STD NUMBER(3),
 10  F_MAT NUMBER(3),
 11* );
SQL> /
);
*
```

```
ERROR at line 11:
ORA-00904: : invalid identifier


SQL> ED
Wrote file afiedt.buf

  1  CREATE TABLE STD
  2  (
  3  ROLL_NO NUMBER(4),
  4  SNAME VARCHAR2(20),
  5  CLASS_NM VARCHAR2(20),
  6  F_ENG NUMBER(3),
  7  F_PHY NUMBER(3),
  8  F_URD NUMBER(3),
  9  F_STD NUMBER(3),
 10  F_MAT NUMBER(3),
 11* )
SQL> /
)
*
ERROR at line 11:
ORA-00904: : invalid identifier


SQL> ED
Wrote file afiedt.buf

  1  CREATE TABLE STD
  2  (
  3  ROLL_NO NUMBER(4),
  4  SNAME VARCHAR2(20),
  5  CLASS_NM VARCHAR2(20),
  6  F_ENG NUMBER(3),
  7  F_PHY NUMBER(3),
  8  F_URD NUMBER(3),
  9  F_STD NUMBER(3),
 10  F_MAT NUMBER(3)
 11* )
SQL> /

Table created.

SQL> DESC STD
 Name                                                      Null?    Type
 --------------------------------------------------------- --------
-----------------------------------
 ROLL_NO                                                            NUMBER(4)
 SNAME                                                              VARCHAR2(20)
 CLASS_NM                                                           VARCHAR2(20)
 F_ENG                                                              NUMBER(3)
 F_PHY                                                              NUMBER(3)
 F_URD                                                              NUMBER(3)
 F_STD                                                              NUMBER(3)
 F_MAT                                                              NUMBER(3)

SQL>
SQL>
SQL>
SQL>
SQL> DECLARE
  2  --------------------DECLARATION AREA-----------------
  3  S_NAME VARCHAR2(20):='&NAME';
```

```
  4  V_CLASS VARCHAR2(20):='&CLASS';
  5  M_MARK NUMBER :=&MATH_MARKS;
  6  S_MARK NUMBER :=&STADIES;
  7  P_MARK NUMBER :=&PHY_MARK;
  8  U_MARK NUMBER :=&URDU;
  9  E_MARK NUMBER :=&ENGLISH;
 10  TOTAL NUMBER :=0;
 11  PER NUMBER := 0;
 12  BEGIN
 13  ------------------------CALCULATION AREA--------------
 14  TOTAL := M_MARK + S_MARK + P_MARK + U_MARK + E_MARK;
 15  PER :=  TOTAL * 100/500;
 16  ------------------------DISPLAY AREA------------------
 17  &D(' MARKS SHEET ');
 18  &D('==============='||CHR(10));
 19  &D('STUDENT NAME '||S_NAME);
 20  &D('STUDENT CLASS '||V_CLASS);
 21  &D('MATH MARKS  '||M_MARK);
 22  &D('PHYSICS MARKS  '||P_MARK);
 23  &D('URDU MARKS  '||U_MARK);
 24  &D('ENGLISH MARKS  '||E_MARK);
 25  &D('STUDIES MARKS  '||S_MARK||CHR(10));
 26  &D('TOTAL MARKS .,...'||TOTAL);
 27  &D('PERCENTAGE  MARKS .,...'||PER||'%');
 28  END;
 29  .
SQL> ED
Wrote file afiedt.buf

  1  DECLARE
  2  --------------------DECLARATION AREA-----------------
  3  R_NO NUMBER :=&ROLL_NO;
  4  S_NAME VARCHAR2(20):='&NAME';
  5  V_CLASS VARCHAR2(20):='&CLASS';
  6  M_MARK NUMBER :=&MATH_MARKS;
  7  S_MARK NUMBER :=&STADIES;
  8  P_MARK NUMBER :=&PHY_MARK;
  9  U_MARK NUMBER :=&URDU;
 10  E_MARK NUMBER :=&ENGLISH;
 11  TOTAL NUMBER :=0;
 12  PER NUMBER := 0;
 13  BEGIN
 14  ------------------------CALCULATION AREA--------------
 15  TOTAL := M_MARK + S_MARK + P_MARK + U_MARK + E_MARK;
 16  PER :=  TOTAL * 100/500;
 17  ------------------------DISPLAY AREA------------------
 18  &D(' MARKS SHEET ');
 19  &D('==============='||CHR(10));
 20  &D('ROLL NO IS ...'||R_NO);
 21  &D('STUDENT NAME '||S_NAME);
 22  &D('STUDENT CLASS '||V_CLASS);
 23  &D('MATH MARKS  '||M_MARK);
 24  &D('PHYSICS MARKS  '||P_MARK);
 25  &D('URDU MARKS  '||U_MARK);
 26  &D('ENGLISH MARKS  '||E_MARK);
 27  &D('STUDIES MARKS  '||S_MARK||CHR(10));
 28  &D('TOTAL MARKS .,...'||TOTAL);
 29  &D('PERCENTAGE  MARKS .,...'||PER||'%');
 30  INSERT INTO STD(
 31  ROLL_NO,
 32  SNAME,
 33  CLASS_NM,
 34  F_ENG,
```

```
 35  F_PHY,
 36  F_MAT,
 37  F_STD,
 38  F_URD)
 39  VALUES(
 40  R_NO,
 41  S_NAME,
 42  V_CLASS,
 43  E_MARK,
 44  P_MARK,
 45  M_MARK,
 46  S_MARK,
 47  U_MARK
 48  );
 49  COMMIT;
 50* END;
 51  /
Enter value for roll_no: 101
Enter value for name: ALI
Enter value for class: X
Enter value for math_marks: 58
Enter value for stadies: 69
Enter value for phy_mark: 58
Enter value for urdu: 47
Enter value for english: 58
MARKS SHEET

==============


ROLL NO IS ...101

STUDENT NAME ALI

STUDENT CLASS X

MATH MARKS  58

PHYSICS MARKS  58

URDU MARKS  47

ENGLISH MARKS  58

STUDIES MARKS  69

TOTAL MARKS .,...290

PERCENTAGE  MARKS .,...58%


PL/SQL procedure successfully completed.

SQL> SELECT * FROM STD;

   ROLL_NO SNAME                CLASS_NM                   F_ENG      F_PHY
F_URD      F_STD
---------- -------------------- -------------------- ---------- ----------
---------- ----------
     F_MAT

----------
```

```
       101 ALI                    X                                58         58
47         69
        58



SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL> SELECT OBJECT_NAME FROM USER_OBJECTS;

OBJECT_NAME

--------------------------------------------------------------------------------
----------------
PK_DEPT

DEPT

EMP

PK_EMP

BONUS

SALGRADE

GET_ORD

INS_REC

ADD_EMP

GET_MGR

GET_JOB

EMP_TEST

EMP_HIST

EMP_EXCEPTION

ADD_NEW_EMP

EMP_POSTING

DEL_REC

P1

P1

FORWARD_DEC

FORWARD_DEC
```

OVERPACK

OVERPACK

BODYLESS_PACK

SHOW_TXT

WRITE_TO_FILE

GET_FILE_TXT

TEST_JOB

DO_EXE_IMM

T1

CREATE_TABLE

TEST

SHOW_REC

LOG_EMP_HIST

EMP_VIEW

ADD_DEPT

EMP_TEMP

VU_SAL

VU_MGR

GET_ID

ADD_R

V1

EIMAGE

SET_VDO

GET_EMP_VDO_LEN

EMP_RESUME

SYS_LOB0000052750C00002$$

LOAD_TXT_DATA

CHK_SAL

BIN$6oYvYDsOSLqezFJYs/7haA==$0

EMP_AUDIT

GET_WORDS

```
S1

GET_TAX

EMP_COPY

JOB_IDS

STD


57 rows selected.

SQL> ED
Wrote file afiedt.buf

  1* SELECT OBJECT_NAME FROM USER_OBJECTS
SQL>
SQL> .
SQL> SPOOL OFF
```

```
SQL>
SQL> DECLARE
  2  .
SQL> ED
Wrote file afiedt.buf

  1  DECLARE
  2      --------------------DECLARATION AREA-----------------
  3      V_R_NO NUMBER :=&ROLL_NO;
  4      V_S_NAME VARCHAR2(20);
  5      V_CLASS VARCHAR2(20);
  6      V_M_MARK NUMBER :=0;
  7      V_S_MARK NUMBER :=0;
  8      V_P_MARK NUMBER :=0;
  9      V_U_MARK NUMBER :=0
 10    V_E_MARK NUMBER :=0;
 11    TOTAL NUMBER :=0;
 12    PER NUMBER := 0;
 13     BEGIN
 14  ------------------FETCHING-----------------
 15  SELECT
 16   SNAME,
 17   CLASS_NM,
 18   F_ENG,
 19   F_PHY,
 20   F_URD,
 21   F_STD,
 22   F_MAT     INTO
 23    V_S_NAME,
 24    V_CLASS,
 25    V_E_MARK,
 26    V_P_MARK,
 27    V_U_MARK ,
 28    V_S_MARK,
 29    V_M_MARK  FROM SCOTT.STD WHERE ROLL=V_R_NO;
 30    ----------------------CALCULATION AREA--------------
 31     TOTAL := V_M_MARK + V_S_MARK + V_P_MARK + V_U_MARK + V_E_MARK;
 32     PER :=  TOTAL * 100/500;
 33     ------------------------DISPLAY AREA------------------
 34     &D(' MARKS SHEET ');
 35     &D('==============='||CHR(10));
 36     &D('ROLL NO IS ...'||V_R_NO);
 37     &D('STUDENT NAME '||V_S_NAME);
 38     &D('STUDENT CLASS '||V_CLASS);
 39     &D('MATH MARKS  '||V_M_MARK);
 40     &D('PHYSICS MARKS  '||V_P_MARK);
 41     &D('URDU MARKS  '||V_U_MARK);
 42     &D('ENGLISH MARKS  '||V_E_MARK);
 43     &D('STUDIES MARKS  '||V_S_MARK||CHR(10));
 44     &D('TOTAL MARKS .,...'||TOTAL);
 45    &D('PERCENTAGE  MARKS .,...'||PER||'%');
 46* END;
 47  /
Enter value for roll_no: 101
  V_E_MARK NUMBER :=0;
  *
ERROR at line 10:
ORA-06550: line 10, column 3:
PLS-00103: Encountered the symbol "V_E_MARK" when expecting one of the following:
* & = - + ; < / > at in is mod remainder not rem
<an exponent (**)> <> or != or ~= >= <= <> and or like LIKE2_
LIKE4_ LIKEC_ between || multiset member SUBMULTISET_
The symbol ";" was substituted for "V_E_MARK" to continue.
```

```
SQL>
SQL>
SQL>
SQL> ED
Wrote file afiedt.buf

  1  DECLARE
  2      --------------------DECLARATION AREA-----------------
  3      V_R_NO NUMBER :=&ROLL_NO;
  4      V_S_NAME VARCHAR2(20);
  5      V_CLASS VARCHAR2(20);
  6      V_M_MARK NUMBER :=0;
  7      V_S_MARK NUMBER :=0;
  8      V_P_MARK NUMBER :=0;
  9      V_U_MARK NUMBER :=0
 10    V_E_MARK NUMBER :=0;
 11    TOTAL NUMBER :=0;
 12    PER NUMBER := 0;
 13     BEGIN
 14  ------------------FETCHING-----------------
 15  SELECT
 16   SNAME,
 17   CLASS_NM,
 18   F_ENG,
 19   F_PHY,
 20   F_URD,
 21   F_STD,
 22   F_MAT     INTO
 23    V_S_NAME,
 24    V_CLASS,
 25    V_E_MARK,
 26    V_P_MARK,
 27    V_U_MARK ,
 28    V_S_MARK,
 29    V_M_MARK  FROM SCOTT.STD WHERE ROLL=V_R_NO;
 30    ------------------------CALCULATION AREA--------------
 31     TOTAL := V_M_MARK + V_S_MARK + V_P_MARK + V_U_MARK + V_E_MARK;
 32     PER :=  TOTAL * 100/500;
 33     ------------------------DISPLAY AREA------------------
 34     &D(' MARKS SHEET ');
 35     &D('==============='||CHR(10));
 36     &D('ROLL NO IS ...'||V_R_NO);
 37     &D('STUDENT NAME '||V_S_NAME);
 38     &D('STUDENT CLASS '||V_CLASS);
 39     &D('MATH MARKS  '||V_M_MARK);
 40     &D('PHYSICS MARKS  '||V_P_MARK);
 41     &D('URDU MARKS  '||V_U_MARK);
 42     &D('ENGLISH MARKS  '||V_E_MARK);
 43     &D('STUDIES MARKS  '||V_S_MARK||CHR(10));
 44     &D('TOTAL MARKS .,...'||TOTAL);
 45    &D('PERCENTAGE  MARKS .,...'||PER||'%');
 46* END;
SQL>
SQL> ED
Wrote file afiedt.buf

  1  DECLARE
  2      --------------------DECLARATION AREA-----------------
  3      V_R_NO NUMBER :=&ROLL_NO;
  4      V_S_NAME VARCHAR2(20);
  5      V_CLASS VARCHAR2(20);
```

```
  6      V_M_MARK NUMBER :=0;
  7      V_S_MARK NUMBER :=0;
  8      V_P_MARK NUMBER :=0;
  9      V_U_MARK NUMBER :=0;
 10    V_E_MARK NUMBER :=0;
 11    TOTAL NUMBER :=0;
 12    PER NUMBER := 0;
 13     BEGIN
 14  ------------------FETCHING-----------------
 15  SELECT
 16   SNAME,
 17   CLASS_NM,
 18   F_ENG,
 19   F_PHY,
 20   F_URD,
 21   F_STD,
 22   F_MAT     INTO
 23    V_S_NAME,
 24    V_CLASS,
 25    V_E_MARK,
 26    V_P_MARK,
 27    V_U_MARK ,
 28    V_S_MARK,
 29    V_M_MARK  FROM SCOTT.STD WHERE ROLL=V_R_NO;
 30    ------------------------CALCULATION AREA--------------
 31     TOTAL := V_M_MARK + V_S_MARK + V_P_MARK + V_U_MARK + V_E_MARK;
 32     PER :=  TOTAL * 100/500;
 33     ------------------------DISPLAY AREA------------------
 34     &D(' MARKS SHEET ');
 35     &D('==============='||CHR(10));
 36     &D('ROLL NO IS ...'||V_R_NO);
 37     &D('STUDENT NAME '||V_S_NAME);
 38     &D('STUDENT CLASS '||V_CLASS);
 39     &D('MATH MARKS  '||V_M_MARK);
 40     &D('PHYSICS MARKS  '||V_P_MARK);
 41     &D('URDU MARKS  '||V_U_MARK);
 42     &D('ENGLISH MARKS  '||V_E_MARK);
 43     &D('STUDIES MARKS  '||V_S_MARK||CHR(10));
 44     &D('TOTAL MARKS .,...'||TOTAL);
 45    &D('PERCENTAGE  MARKS .,...'||PER||'%');
 46* END;
SQL> /
Enter value for roll_no: 101
  V_M_MARK  FROM SCOTT.STD WHERE ROLL=V_R_NO;
                               *
ERROR at line 29:
ORA-06550: line 29, column 34:
PL/SQL: ORA-00904: "ROLL": invalid identifier
ORA-06550: line 15, column 1:
PL/SQL: SQL Statement ignored


SQL> ED
Wrote file afiedt.buf

  1  DECLARE
  2      --------------------DECLARATION AREA-----------------
  3      V_R_NO NUMBER :=&ROLL_NO;
  4      V_S_NAME VARCHAR2(20);
  5      V_CLASS VARCHAR2(20);
  6      V_M_MARK NUMBER :=0;
  7      V_S_MARK NUMBER :=0;
  8      V_P_MARK NUMBER :=0;
```

```
  9      V_U_MARK NUMBER :=0;
 10    V_E_MARK NUMBER :=0;
 11    TOTAL NUMBER :=0;
 12    PER NUMBER := 0;
 13     BEGIN
 14  ------------------FETCHING-----------------
 15  SELECT
 16   SNAME,
 17   CLASS_NM,
 18   F_ENG,
 19   F_PHY,
 20   F_URD,
 21   F_STD,
 22   F_MAT     INTO
 23    V_S_NAME,
 24    V_CLASS,
 25    V_E_MARK,
 26    V_P_MARK,
 27    V_U_MARK ,
 28    V_S_MARK,
 29    V_M_MARK  FROM SCOTT.STD WHERE ROLL_NO=R_NO;
 30    ----------------------CALCULATION AREA--------------
 31     TOTAL := V_M_MARK + V_S_MARK + V_P_MARK + V_U_MARK + V_E_MARK;
 32     PER :=  TOTAL * 100/500;
 33     ------------------------DISPLAY AREA------------------
 34     &D(' MARKS SHEET ');
 35     &D('==============='||CHR(10));
 36     &D('ROLL NO IS ...'||V_R_NO);
 37     &D('STUDENT NAME '||V_S_NAME);
 38     &D('STUDENT CLASS '||V_CLASS);
 39     &D('MATH MARKS  '||V_M_MARK);
 40     &D('PHYSICS MARKS  '||V_P_MARK);
 41     &D('URDU MARKS  '||V_U_MARK);
 42     &D('ENGLISH MARKS  '||V_E_MARK);
 43     &D('STUDIES MARKS  '||V_S_MARK||CHR(10));
 44     &D('TOTAL MARKS .,...'||TOTAL);
 45    &D('PERCENTAGE  MARKS .,...'||PER||'%');
 46* END;
SQL> /
Enter value for roll_no: 101
  V_M_MARK  FROM SCOTT.STD WHERE ROLL_NO=R_NO;
                                         *
ERROR at line 29:
ORA-06550: line 29, column 42:
PL/SQL: ORA-00904: "R_NO": invalid identifier
ORA-06550: line 15, column 1:
PL/SQL: SQL Statement ignored


SQL> ED
Wrote file afiedt.buf

  1  DECLARE
  2      --------------------DECLARATION AREA-----------------
  3      V_R_NO NUMBER :=&ROLL_NO;
  4      V_S_NAME VARCHAR2(20);
  5      V_CLASS VARCHAR2(20);
  6      V_M_MARK NUMBER :=0;
  7      V_S_MARK NUMBER :=0;
  8      V_P_MARK NUMBER :=0;
  9      V_U_MARK NUMBER :=0;
 10    V_E_MARK NUMBER :=0;
 11    TOTAL NUMBER :=0;
```

```
 12    PER NUMBER := 0;
 13     BEGIN
 14  ------------------FETCHING-----------------
 15  SELECT
 16   SNAME,
 17   CLASS_NM,
 18   F_ENG,
 19   F_PHY,
 20   F_URD,
 21   F_STD,
 22   F_MAT     INTO
 23    V_S_NAME,
 24    V_CLASS,
 25    V_E_MARK,
 26    V_P_MARK,
 27    V_U_MARK ,
 28    V_S_MARK,
 29    V_M_MARK  FROM SCOTT.STD WHERE ROLL_NO=V_R_NO;
 30    ----------------------CALCULATION AREA--------------
 31     TOTAL := V_M_MARK + V_S_MARK + V_P_MARK + V_U_MARK + V_E_MARK;
 32     PER :=  TOTAL * 100/500;
 33     -----------------------DISPLAY AREA------------------
 34     &D(' MARKS SHEET ');
 35     &D('==============='||CHR(10));
 36     &D('ROLL NO IS ...'||V_R_NO);
 37     &D('STUDENT NAME '||V_S_NAME);
 38     &D('STUDENT CLASS '||V_CLASS);
 39     &D('MATH MARKS  '||V_M_MARK);
 40     &D('PHYSICS MARKS  '||V_P_MARK);
 41     &D('URDU MARKS  '||V_U_MARK);
 42     &D('ENGLISH MARKS  '||V_E_MARK);
 43     &D('STUDIES MARKS  '||V_S_MARK||CHR(10));
 44     &D('TOTAL MARKS .,...'||TOTAL);
 45    &D('PERCENTAGE  MARKS .,...'||PER||'%');
 46* END;
SQL> /
Enter value for roll_no: 101
MARKS SHEET

==============


ROLL NO IS ...101

STUDENT NAME ALI

STUDENT CLASS X

MATH MARKS  58

PHYSICS MARKS  58

URDU MARKS  47

ENGLISH MARKS  58

STUDIES MARKS  69

TOTAL MARKS .,...290

PERCENTAGE  MARKS .,...58%
```

```
PL/SQL procedure successfully completed.

SQL>
SQL>
SQL>
SQL>
SQL> ED
Wrote file afiedt.buf

  1  DECLARE
  2      --------------------DECLARATION AREA-----------------
  3      V_R_NO NUMBER :=&ROLL_NO;
  4      V_S_NAME VARCHAR2(20);
  5      V_CLASS VARCHAR2(2);
  6      V_M_MARK NUMBER :=0;
  7      V_S_MARK NUMBER :=0;
  8      V_P_MARK NUMBER :=0;
  9      V_U_MARK NUMBER :=0;
 10    V_E_MARK NUMBER :=0;
 11    TOTAL NUMBER :=0;
 12    PER NUMBER := 0;
 13     BEGIN
 14  ------------------FETCHING-----------------
 15  SELECT
 16   SNAME,
 17   CLASS_NM,
 18   F_ENG,
 19   F_PHY,
 20   F_URD,
 21   F_STD,
 22   F_MAT     INTO
 23    V_S_NAME,
 24    V_CLASS,
 25    V_E_MARK,
 26    V_P_MARK,
 27    V_U_MARK ,
 28    V_S_MARK,
 29    V_M_MARK  FROM SCOTT.STD WHERE ROLL_NO=V_R_NO;
 30    ------------------------CALCULATION AREA--------------
 31     TOTAL := V_M_MARK + V_S_MARK + V_P_MARK + V_U_MARK + V_E_MARK;
 32     PER :=  TOTAL * 100/500;
 33     ------------------------DISPLAY AREA------------------
 34     &D(' MARKS SHEET ');
 35     &D('==============='||CHR(10));
 36     &D('ROLL NO IS ...'||V_R_NO);
 37     &D('STUDENT NAME '||V_S_NAME);
 38     &D('STUDENT CLASS '||V_CLASS);
 39     &D('MATH MARKS  '||V_M_MARK);
 40     &D('PHYSICS MARKS  '||V_P_MARK);
 41     &D('URDU MARKS  '||V_U_MARK);
 42     &D('ENGLISH MARKS  '||V_E_MARK);
 43     &D('STUDIES MARKS  '||V_S_MARK||CHR(10));
 44     &D('TOTAL MARKS .,...'||TOTAL);
 45    &D('PERCENTAGE  MARKS .,...'||PER||'%');
 46* END;
SQL>
SQL> /
Enter value for roll_no: 101
MARKS SHEET

==============
```

```
ROLL NO IS ...101

STUDENT NAME ALI

STUDENT CLASS X

MATH MARKS  58

PHYSICS MARKS  58

URDU MARKS  47

ENGLISH MARKS  58

STUDIES MARKS  69

TOTAL MARKS .,...290

PERCENTAGE  MARKS .,...58%


PL/SQL procedure successfully completed.

SQL>
SQL>
SQL> ED
Wrote file afiedt.buf

  1  DECLARE
  2      --------------------DECLARATION AREA-----------------
  3      V_R_NO NUMBER :=&ROLL_NO;
  4      V_S_NAME VARCHAR2(2);
  5      V_CLASS VARCHAR2(2);
  6      V_M_MARK NUMBER :=0;
  7      V_S_MARK NUMBER :=0;
  8      V_P_MARK NUMBER :=0;
  9      V_U_MARK NUMBER :=0;
 10    V_E_MARK NUMBER :=0;
 11    TOTAL NUMBER :=0;
 12    PER NUMBER := 0;
 13     BEGIN
 14  ------------------FETCHING-----------------
 15  SELECT
 16   SNAME,
 17   CLASS_NM,
 18   F_ENG,
 19   F_PHY,
 20   F_URD,
 21   F_STD,
 22   F_MAT     INTO
 23    V_S_NAME,
 24    V_CLASS,
 25    V_E_MARK,
 26    V_P_MARK,
 27    V_U_MARK ,
 28    V_S_MARK,
 29    V_M_MARK  FROM SCOTT.STD WHERE ROLL_NO=V_R_NO;
 30    ------------------------CALCULATION AREA--------------
 31     TOTAL := V_M_MARK + V_S_MARK + V_P_MARK + V_U_MARK + V_E_MARK;
 32     PER :=  TOTAL * 100/500;
 33     ------------------------DISPLAY AREA------------------
 34     &D(' MARKS SHEET ');
 35     &D('==============='||CHR(10));
```

```
 36     &D('ROLL NO IS ...'||V_R_NO);
 37     &D('STUDENT NAME '||V_S_NAME);
 38     &D('STUDENT CLASS '||V_CLASS);
 39     &D('MATH MARKS  '||V_M_MARK);
 40     &D('PHYSICS MARKS  '||V_P_MARK);
 41     &D('URDU MARKS  '||V_U_MARK);
 42     &D('ENGLISH MARKS  '||V_E_MARK);
 43     &D('STUDIES MARKS  '||V_S_MARK||CHR(10));
 44     &D('TOTAL MARKS .,...'||TOTAL);
 45    &D('PERCENTAGE  MARKS .,...'||PER||'%');
 46* END;
SQL> /
Enter value for roll_no: 101
DECLARE
*
ERROR at line 1:
ORA-06502: PL/SQL: numeric or value error: character string buffer too small
ORA-06512: at line 15


SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL> ED
Wrote file afiedt.buf

  1  DECLARE
  2      --------------------DECLARATION AREA-----------------
  3      V_R_NO NUMBER :=&ROLL_NO;
  4      V_S_NAME STD.SNAME%TYPE;
  5      V_CLASS VARCHAR2(2);
  6      V_M_MARK NUMBER :=0;
  7      V_S_MARK NUMBER :=0;
  8      V_P_MARK NUMBER :=0;
  9      V_U_MARK NUMBER :=0;
 10    V_E_MARK NUMBER :=0;
 11    TOTAL NUMBER :=0;
 12    PER NUMBER := 0;
 13     BEGIN
 14  ------------------FETCHING-----------------
 15  SELECT
 16   SNAME,
 17   CLASS_NM,
 18   F_ENG,
 19   F_PHY,
 20   F_URD,
 21   F_STD,
 22   F_MAT     INTO
 23    V_S_NAME,
 24    V_CLASS,
 25    V_E_MARK,
 26    V_P_MARK,
 27    V_U_MARK ,
 28    V_S_MARK,
```

```
 29    V_M_MARK  FROM SCOTT.STD WHERE ROLL_NO=V_R_NO;
 30    -----------------------CALCULATION AREA--------------
 31     TOTAL := V_M_MARK + V_S_MARK + V_P_MARK + V_U_MARK + V_E_MARK;
 32     PER :=  TOTAL * 100/500;
 33     -----------------------DISPLAY AREA------------------
 34     &D(' MARKS SHEET ');
 35     &D('==============='||CHR(10));
 36     &D('ROLL NO IS ...'||V_R_NO);
 37     &D('STUDENT NAME '||V_S_NAME);
 38     &D('STUDENT CLASS '||V_CLASS);
 39     &D('MATH MARKS  '||V_M_MARK);
 40     &D('PHYSICS MARKS  '||V_P_MARK);
 41     &D('URDU MARKS  '||V_U_MARK);
 42     &D('ENGLISH MARKS  '||V_E_MARK);
 43     &D('STUDIES MARKS  '||V_S_MARK||CHR(10));
 44     &D('TOTAL MARKS .,...'||TOTAL);
 45    &D('PERCENTAGE  MARKS .,...'||PER||'%');
 46* END;
SQL> /
Enter value for roll_no: 101
MARKS SHEET

==============


ROLL NO IS ...101

STUDENT NAME ALI

STUDENT CLASS X

MATH MARKS  58

PHYSICS MARKS  58

URDU MARKS  47

ENGLISH MARKS  58

STUDIES MARKS  69

TOTAL MARKS .,...290

PERCENTAGE  MARKS .,...58%


PL/SQL procedure successfully completed.

SQL>
SQL>
SQL> ED
Wrote file afiedt.buf

  1  DECLARE
  2      --------------------DECLARATION AREA-----------------
  3      V_R_NO NUMBER :=&ROLL_NO;
  4      V_S_NAME STD.SNAME%TYPE;
  5      V_CLASS STD.CLASS_NM%TYPE;
  6      V_M_MARK STD.F_MAT%TYPE;
  7      V_S_MARK STD.F_STD%TYPE;
  8      V_P_MARK STD.F_PHY%TYPE;
  9      V_U_MARK STD.F_URD%TYPE;
 10    V_E_MARK STD.F_ENG%TYPE;
```

```
 11    TOTAL NUMBER :=0;
 12    PER NUMBER := 0;
 13     BEGIN
 14  ------------------FETCHING-----------------
 15  SELECT
 16   SNAME,
 17   CLASS_NM,
 18   F_ENG,
 19   F_PHY,
 20   F_URD,
 21   F_STD,
 22   F_MAT     INTO
 23    V_S_NAME,
 24    V_CLASS,
 25    V_E_MARK,
 26    V_P_MARK,
 27    V_U_MARK ,
 28    V_S_MARK,
 29    V_M_MARK  FROM SCOTT.STD WHERE ROLL_NO=V_R_NO;
 30    ------------------------CALCULATION AREA--------------
 31     TOTAL := V_M_MARK + V_S_MARK + V_P_MARK + V_U_MARK + V_E_MARK;
 32     PER :=  TOTAL * 100/500;
 33     ------------------------DISPLAY AREA------------------
 34     &D(' MARKS SHEET ');
 35     &D('==============='||CHR(10));
 36     &D('ROLL NO IS ...'||V_R_NO);
 37     &D('STUDENT NAME '||V_S_NAME);
 38     &D('STUDENT CLASS '||V_CLASS);
 39     &D('MATH MARKS  '||V_M_MARK);
 40     &D('PHYSICS MARKS  '||V_P_MARK);
 41     &D('URDU MARKS  '||V_U_MARK);
 42     &D('ENGLISH MARKS  '||V_E_MARK);
 43     &D('STUDIES MARKS  '||V_S_MARK||CHR(10));
 44     &D('TOTAL MARKS .,...'||TOTAL);
 45    &D('PERCENTAGE  MARKS .,...'||PER||'%');
 46* END;
SQL> /
Enter value for roll_no: 101
MARKS SHEET

==============


ROLL NO IS ...101

STUDENT NAME ALI

STUDENT CLASS X

MATH MARKS  58

PHYSICS MARKS  58

URDU MARKS  47

ENGLISH MARKS  58

STUDIES MARKS  69

TOTAL MARKS .,...290

PERCENTAGE  MARKS .,...58%
```

```
PL/SQL procedure successfully completed.

SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL> ED
Wrote file afiedt.buf

  1  DECLARE
  2              V_EMPNO EMP.EMPNO%TYPE:=&EMP_ID;
  3              V_ENAME EMP.ENAME%TYPE;
  4              V_JOB   EMP.JOB%TYPE;
  5              V_SAL   EMP.SAL%TYPE;
  6                      V_DEPTNO EMP.DEPTNO%TYPE;
  7  BEGIN
  8     SELECT ENAME,JOB,SAL,DEPTNO
  9     INTO
 10            V_ENAME,V_JOB,V_SAL,V_DEPTNO
 11  WHERE EMPNO=V_EMPNO;
 12       &D('ENAME IS ...'||V_ENAME);
 13       &D('JOB IS ...'||V_JOB);
 14       &D('SALARY  IS ...'||V_SAL);
 15       &D('DEPTNO IS ...'||V_DEPTNO);
 16  END;
 17* END;
 18  /
Enter value for emp_id:
                V_EMPNO EMP.EMPNO%TYPE:=;
                                       *
ERROR at line 2:
ORA-06550: line 2, column 27:
PLS-00103: Encountered the symbol ";" when expecting one of the following:
( - + case mod new not null <an identifier>
<a double-quoted delimited-identifier> <a bind variable> avg
count current exists max min prior sql stddev sum variance
execute forall merge time timestamp interval date
<a string literal with character set specification>
<a number> <a single-quoted SQL string> pipe
<an alternatively-quoted string literal with character set specification>
<an alternatively-quoted S
ORA-06550: line 11, column 1:
PLS-00103: Encountered the symbol "WHERE" when expecting one of the following:
. ( , % from
ORA-06550: line 12, column 6:
PLS-00103: Encountered the symbol "DBMS_OUTPUT"
ORA-06550: line 12, column 51:
PLS-00103: Encountered the symbol ";" when expecting one of the following:
. ( , * % & - + / at mod remainder rem <an identifier>
<a double-quoted delimited-identifier> <an exponent (**)> as
from into || multiset bulk


SQL>
SQL> ED
Wrote file afiedt.buf

  1  DECLARE
  2              V_EMPNO EMP.EMPNO%TYPE:=&EMP_ID;
```

```
  3             V_ENAME EMP.ENAME%TYPE;
  4             V_JOB   EMP.JOB%TYPE;
  5             V_SAL   EMP.SAL%TYPE;
  6                     V_DEPTNO EMP.DEPTNO%TYPE;
  7  BEGIN
  8     SELECT ENAME,JOB,SAL,DEPTNO
  9     INTO
 10            V_ENAME,V_JOB,V_SAL,V_DEPTNO
 11  WHERE EMPNO=V_EMPNO;
 12       &D('ENAME IS ...'||V_ENAME);
 13       &D('JOB IS ...'||V_JOB);
 14       &D('SALARY  IS ...'||V_SAL);
 15       &D('DEPTNO IS ...'||V_DEPTNO);
 16* END;
SQL> /
Enter value for emp_id: 7788
WHERE EMPNO=V_EMPNO;
*
ERROR at line 11:
ORA-06550: line 10, column 40:
PL/SQL: ORA-00923: FROM keyword not found where expected
ORA-06550: line 8, column 4:
PL/SQL: SQL Statement ignored


SQL> ED
Wrote file afiedt.buf

  1  DECLARE
  2             V_EMPNO EMP.EMPNO%TYPE:=&EMP_ID;
  3             V_ENAME EMP.ENAME%TYPE;
  4             V_JOB   EMP.JOB%TYPE;
  5             V_SAL   EMP.SAL%TYPE;
  6                     V_DEPTNO EMP.DEPTNO%TYPE;
  7  BEGIN
  8     SELECT ENAME,JOB,SAL,DEPTNO
  9     INTO
 10            V_ENAME,V_JOB,V_SAL,V_DEPTNO
 11  FROM SCOTT.EMP
 12  WHERE EMPNO=V_EMPNO;
 13       &D('ENAME IS ...'||V_ENAME);
 14       &D('JOB IS ...'||V_JOB);
 15       &D('SALARY  IS ...'||V_SAL);
 16       &D('DEPTNO IS ...'||V_DEPTNO);
 17* END;
SQL> /
Enter value for emp_id: 7788
ENAME IS ...SCOTT

JOB IS ...ANALYST

SALARY  IS ...3000

DEPTNO IS ...20


PL/SQL procedure successfully completed.

SQL>
SQL>
SQL>
SQL>
SQL>
```

```
SQL>
SQL>
SQL> ED
Wrote file afiedt.buf

  1  DECLARE
  2              V_EMPNO EMP.EMPNO%TYPE:=&EMP_ID;
  3              V_ENAME EMP.ENAME%TYPE;
  4              V_JOB   EMP.JOB%TYPE;
  5              V_SAL   EMP.SAL%TYPE;
  6                      V_DEPTNO EMP.DEPTNO%TYPE;
  7                   V_DNAME  DEPT.DNAME%TYPE;
  8  BEGIN
  9     SELECT ENAME,JOB,SAL,DEPTNO
 10     INTO
 11            V_ENAME,V_JOB,V_SAL,V_DEPTNO
 12  FROM SCOTT.EMP
 13  WHERE EMPNO=V_EMPNO;
 14  SELECT DNAME INTO V_DNAME FROM DEPT
 15  WHERE DEPTNO=V_DEPTNO;
 16       &D('ENAME IS ...'||V_ENAME);
 17       &D('JOB IS ...'||V_JOB);
 18       &D('SALARY  IS ...'||V_SAL);
 19       &D('DEPTNO IS ...'||V_DEPTNO);
 20       &D('DNAME IS ...'||V_DNAME);
 21* END;
 22  /
Enter value for emp_id: 101
DECLARE
*
ERROR at line 1:
ORA-01403: no data found
ORA-06512: at line 9


SQL> E
SP2-0042: unknown command "E" - rest of line ignored.
SQL>
SQL> /
Enter value for emp_id: 7788
ENAME IS ...SCOTT

JOB IS ...ANALYST

SALARY  IS ...3000

DEPTNO IS ...20

DNAME IS ...RESEARCH


PL/SQL procedure successfully completed.

SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL> ED
Wrote file afiedt.buf

  1  DECLARE
```

```
  2             V_EMPNO EMP.EMPNO%TYPE:=&EMP_ID;
  3             V_ENAME EMP.ENAME%TYPE;
  4             V_JOB   EMP.JOB%TYPE;
  5             V_SAL   EMP.SAL%TYPE;
  6                     V_DEPTNO EMP.DEPTNO%TYPE;
  7                  V_DNAME  DEPT.DNAME%TYPE;
  8                   V_GRADE SALGRADE.GRADE%TYPE;
  9  BEGIN
 10     SELECT ENAME,JOB,SAL,DEPTNO
 11     INTO
 12            V_ENAME,V_JOB,V_SAL,V_DEPTNO
 13  FROM SCOTT.EMP
 14  WHERE EMPNO=V_EMPNO;
 15  SELECT DNAME INTO V_DNAME FROM DEPT
 16  WHERE DEPTNO=V_DEPTNO;
 17  SELECT  GRADE INTO V_GRADE FROM SALGRADE
 18  WHERE V_SAL BETWEEN LOSAL AND HISAL;
 19       &D('ENAME IS ...'||V_ENAME);
 20       &D('JOB IS ...'||V_JOB);
 21       &D('SALARY  IS ...'||V_SAL);
 22       &D('DEPTNO IS ...'||V_DEPTNO);
 23       &D('DNAME IS ...'||V_DNAME);
 24       &D('GRADE IS ...'||V_GRADE);
 25* END;
 26  /
Enter value for emp_id: 7788
DECLARE
*
ERROR at line 1:
ORA-01422: exact fetch returns more than requested number of rows
ORA-06512: at line 17


SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL> ED
Wrote file afiedt.buf

SP2-0223: No lines in SQL buffer.
SQL> SELECT * FROM GRADE;
SELECT * FROM GRADE
              *
ERROR at line 1:
ORA-00942: table or view does not exist


SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL> SELECT * FROM SALGRADE;

     GRADE      LOSAL      HISAL
```

```
---------- ---------- ----------

         1        700       1200

         2       1201       1400

         3       1401       2000

         4       2001       3000

         5       3001       9999

         1        700       1200

         2       1201       1400

         3       1401       2000

         4       2001       3000

         5       3001       9999

10 rows selected.

SQL>
SQL>
SQL>
SQL>
SQL>
SQL> /

     GRADE      LOSAL      HISAL

---------- ---------- ----------

         1        700       1200

         2       1201       1400

         3       1401       2000

         4       2001       3000

         5       3001       9999


SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL> ed
Wrote file afiedt.buf

  1  DECLARE
  2              V_EMPNO EMP.EMPNO%TYPE:=&EMP_ID;
  3              V_ENAME EMP.ENAME%TYPE;
  4              V_JOB   EMP.JOB%TYPE;
```

```
  5             V_SAL    EMP.SAL%TYPE;
  6                      V_DEPTNO EMP.DEPTNO%TYPE;
  7                   V_DNAME  DEPT.DNAME%TYPE;
  8                    V_GRADE SALGRADE.GRADE%TYPE;
  9  BEGIN
 10     SELECT ENAME,JOB,SAL,DEPTNO
 11     INTO
 12            V_ENAME,V_JOB,V_SAL,V_DEPTNO
 13  FROM SCOTT.EMP
 14  WHERE EMPNO=V_EMPNO;
 15  SELECT DNAME INTO V_DNAME FROM DEPT
 16  WHERE DEPTNO=V_DEPTNO;
 17  SELECT  GRADE INTO V_GRADE FROM SALGRADE
 18  WHERE V_SAL BETWEEN LOSAL AND HISAL;
 19       &D('ENAME IS ...'||V_ENAME);
 20       &D('JOB IS ...'||V_JOB);
 21       &D('SALARY  IS ...'||V_SAL);
 22       &D('DEPTNO IS ...'||V_DEPTNO);
 23       &D('DNAME IS ...'||V_DNAME);
 24       &D('GRADE IS ...'||V_GRADE);
 25* END;
SQL> /
Enter value for emp_id: 7788
ENAME IS ...SCOTT

JOB IS ...ANALYST

SALARY  IS ...3000

DEPTNO IS ...20

DNAME IS ...RESEARCH

GRADE IS ...4


PL/SQL procedure successfully completed.

SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL> ed
Wrote file afiedt.buf

  1  DECLARE
  2             V_EMPNO EMP.EMPNO%TYPE:=&EMP_ID;
  3             V_ENAME EMP.ENAME%TYPE;
  4             V_JOB   EMP.JOB%TYPE;
  5             V_SAL   EMP.SAL%TYPE;
  6                     V_DEPTNO EMP.DEPTNO%TYPE;
  7                   V_DNAME  DEPT.DNAME%TYPE;
  8                    V_GRADE SALGRADE.GRADE%TYPE;
  9  BEGIN
 10     SELECT E.ENAME,E.JOB,E.SAL,E.DEPTNO,D.dname,G.GRADE
 11     INTO
 12            V_ENAME,V_JOB,V_SAL,V_DEPTNO,V_DNAME,V_GRADE
 13  FROM SCOTT.EMP E  JOIN DEPT D
```

```
 14  ON E.DEPTNO=D.DEPTNO
 15  JOIN SALGRADE G
 16  ON E.SAL BETWEEN HISAL AND LOSAL
 17  AND E.EMPNO=V_EMPNO;
 18  /*SELECT DNAME INTO V_DNAME FROM DEPT
 19  WHERE DEPTNO=V_DEPTNO;
 20  SELECT  GRADE INTO V_GRADE FROM SALGRADE
 21  WHERE V_SAL BETWEEN LOSAL AND HISAL;
 22  */
 23       &D('ENAME IS ...'||V_ENAME);
 24       &D('JOB IS ...'||V_JOB);
 25       &D('SALARY  IS ...'||V_SAL);
 26       &D('DEPTNO IS ...'||V_DEPTNO);
 27       &D('DNAME IS ...'||V_DNAME);
 28       &D('GRADE IS ...'||V_GRADE);
 29* END;
 30  /
Enter value for emp_id: 7788
DECLARE
*
ERROR at line 1:
ORA-01403: no data found
ORA-06512: at line 10


SQL>
SQL>
SQL>
SQL>
SQL> ED
Wrote file afiedt.buf

  1  DECLARE
  2              V_EMPNO EMP.EMPNO%TYPE:=&EMP_ID;
  3              V_ENAME EMP.ENAME%TYPE;
  4              V_JOB   EMP.JOB%TYPE;
  5              V_SAL   EMP.SAL%TYPE;
  6                      V_DEPTNO EMP.DEPTNO%TYPE;
  7                   V_DNAME  DEPT.DNAME%TYPE;
  8                    V_GRADE SALGRADE.GRADE%TYPE;
  9  BEGIN
 10     SELECT E.ENAME,E.JOB,E.SAL,E.DEPTNO,D.dname,G.GRADE
 11     INTO
 12             V_ENAME,V_JOB,V_SAL,V_DEPTNO,V_DNAME,V_GRADE
 13  FROM SCOTT.EMP E  JOIN DEPT D
 14  ON E.DEPTNO=D.DEPTNO
 15  JOIN SALGRADE G
 16  ON E.SAL BETWEEN G.HISAL AND G.LOSAL
 17  WHERE E.EMPNO=V_EMPNO;
 18  /*SELECT DNAME INTO V_DNAME FROM DEPT
 19  WHERE DEPTNO=V_DEPTNO;
 20  SELECT  GRADE INTO V_GRADE FROM SALGRADE
 21  WHERE V_SAL BETWEEN LOSAL AND HISAL;
 22  */
 23       &D('ENAME IS ...'||V_ENAME);
 24       &D('JOB IS ...'||V_JOB);
 25       &D('SALARY  IS ...'||V_SAL);
 26       &D('DEPTNO IS ...'||V_DEPTNO);
 27       &D('DNAME IS ...'||V_DNAME);
 28       &D('GRADE IS ...'||V_GRADE);
 29* END;
 30  /
Enter value for emp_id: 7788
```

```
DECLARE
*
ERROR at line 1:
ORA-01403: no data found
ORA-06512: at line 10


SQL>
SQL>
SQL>
SQL> ED
Wrote file afiedt.buf

  1  DECLARE
  2              V_EMPNO EMP.EMPNO%TYPE:=&EMP_ID;
  3              V_ENAME EMP.ENAME%TYPE;
  4              V_JOB   EMP.JOB%TYPE;
  5              V_SAL   EMP.SAL%TYPE;
  6                      V_DEPTNO EMP.DEPTNO%TYPE;
  7                   V_DNAME  DEPT.DNAME%TYPE;
  8                    V_GRADE SALGRADE.GRADE%TYPE;
  9  BEGIN
 10     SELECT E.ENAME,E.JOB,E.SAL,E.DEPTNO,D.dname,G.GRADE
 11     INTO
 12             V_ENAME,V_JOB,V_SAL,V_DEPTNO,V_DNAME,V_GRADE
 13  FROM SCOTT.EMP E  JOIN DEPT D
 14  ON E.DEPTNO=D.DEPTNO
 15  JOIN SALGRADE G
 16  ON E.SAL BETWEEN G.HISAL AND G.LOSAL
 17  WHERE E.EMPNO=7788 ----V_EMPNO;
 18  /*SELECT DNAME INTO V_DNAME FROM DEPT
 19  WHERE DEPTNO=V_DEPTNO;
 20  SELECT  GRADE INTO V_GRADE FROM SALGRADE
 21  WHERE V_SAL BETWEEN LOSAL AND HISAL;
 22  */
 23       &D('ENAME IS ...'||V_ENAME);
 24       &D('JOB IS ...'||V_JOB);
 25       &D('SALARY  IS ...'||V_SAL);
 26       &D('DEPTNO IS ...'||V_DEPTNO);
 27       &D('DNAME IS ...'||V_DNAME);
 28       &D('GRADE IS ...'||V_GRADE);
 29* END;
SQL> /
Enter value for emp_id: 7478
     DBMS_OUTPUT.PUT_LINE('ENAME IS ...'||V_ENAME);
     *
ERROR at line 23:
ORA-06550: line 23, column 6:
PL/SQL: ORA-00933: SQL command not properly ended
ORA-06550: line 10, column 4:
PL/SQL: SQL Statement ignored


SQL>
SQL>
SQL>
SQL>
SQL> ED
Wrote file afiedt.buf

  1  DECLARE
  2              V_EMPNO EMP.EMPNO%TYPE:=&EMP_ID;
  3              V_ENAME EMP.ENAME%TYPE;
```

```
  4             V_JOB    EMP.JOB%TYPE;
  5             V_SAL    EMP.SAL%TYPE;
  6                      V_DEPTNO EMP.DEPTNO%TYPE;
  7                   V_DNAME  DEPT.DNAME%TYPE;
  8                    V_GRADE SALGRADE.GRADE%TYPE;
  9  BEGIN
 10     SELECT E.ENAME,E.JOB,E.SAL,E.DEPTNO,D.dname,G.GRADE
 11     INTO
 12            V_ENAME,V_JOB,V_SAL,V_DEPTNO,V_DNAME,V_GRADE
 13  FROM SCOTT.EMP E  JOIN DEPT D
 14  ON E.DEPTNO=D.DEPTNO
 15  JOIN SALGRADE G
 16  ON E.SAL BETWEEN G.HISAL AND G.LOSAL
 17  WHERE E.EMPNO=7788 ;  ----V_EMPNO;
 18  /*SELECT DNAME INTO V_DNAME FROM DEPT
 19  WHERE DEPTNO=V_DEPTNO;
 20  SELECT  GRADE INTO V_GRADE FROM SALGRADE
 21  WHERE V_SAL BETWEEN LOSAL AND HISAL;
 22  */
 23       &D('ENAME IS ...'||V_ENAME);
 24       &D('JOB IS ...'||V_JOB);
 25       &D('SALARY  IS ...'||V_SAL);
 26       &D('DEPTNO IS ...'||V_DEPTNO);
 27       &D('DNAME IS ...'||V_DNAME);
 28       &D('GRADE IS ...'||V_GRADE);
 29* END;
SQL> /
Enter value for emp_id: 457
DECLARE
*
ERROR at line 1:
ORA-01403: no data found
ORA-06512: at line 10


SQL>
SQL>
SQL>
SQL>
SQL>
SQL> ED
Wrote file afiedt.buf

  1  DECLARE
  2             V_EMPNO EMP.EMPNO%TYPE:=&EMP_ID;
  3             V_ENAME EMP.ENAME%TYPE;
  4             V_JOB    EMP.JOB%TYPE;
  5             V_SAL    EMP.SAL%TYPE;
  6                      V_DEPTNO EMP.DEPTNO%TYPE;
  7                   V_DNAME  DEPT.DNAME%TYPE;
  8                    V_GRADE SALGRADE.GRADE%TYPE;
  9  BEGIN
 10     SELECT E.ENAME,E.JOB,E.SAL,E.DEPTNO,D.dname,G.GRADE
 11     INTO
 12            V_ENAME,V_JOB,V_SAL,V_DEPTNO,V_DNAME,V_GRADE
 13  FROM EMP E  JOIN DEPT D
 14  ON E.DEPTNO=D.DEPTNO
 15  JOIN SALGRADE G
 16  ON E.SAL BETWEEN G.HISAL AND G.LOSAL
 17  WHERE E.EMPNO=V_EMPNO;
 18  /*SELECT DNAME INTO V_DNAME FROM DEPT
 19  WHERE DEPTNO=V_DEPTNO;
 20  SELECT  GRADE INTO V_GRADE FROM SALGRADE
```

```
 21  WHERE V_SAL BETWEEN LOSAL AND HISAL;
 22  */
 23       &D('ENAME IS ...'||V_ENAME);
 24       &D('JOB IS ...'||V_JOB);
 25       &D('SALARY  IS ...'||V_SAL);
 26       &D('DEPTNO IS ...'||V_DEPTNO);
 27       &D('DNAME IS ...'||V_DNAME);
 28       &D('GRADE IS ...'||V_GRADE);
 29* END;
SQL> /
Enter value for emp_id: 7788
DECLARE
*
ERROR at line 1:
ORA-01403: no data found
ORA-06512: at line 10


SQL>
SQL>
SQL>
SQL> ED
Wrote file afiedt.buf

  1  DECLARE
  2              V_EMPNO EMP.EMPNO%TYPE:=&EMP_ID;
  3              V_ENAME EMP.ENAME%TYPE;
  4              V_JOB   EMP.JOB%TYPE;
  5              V_SAL   EMP.SAL%TYPE;
  6                      V_DEPTNO EMP.DEPTNO%TYPE;
  7                   V_DNAME  DEPT.DNAME%TYPE;
  8                    V_GRADE SALGRADE.GRADE%TYPE;
  9  BEGIN
 10     SELECT E.ENAME,E.JOB,E.SAL,E.DEPTNO,D.dname,G.GRADE
 11     INTO
 12            V_ENAME,V_JOB,V_SAL,V_DEPTNO,V_DNAME,V_GRADE
 13  FROM EMP E,DEPT D,SALGRADE G
 14  WHERE E.DEPTNO=D.DEPTNO
 15  AND E.SAL BETWEEN G.HISAL AND G.LOSAL
 16  AND  E.EMPNO=V_EMPNO;
 17  /*SELECT DNAME INTO V_DNAME FROM DEPT
 18  WHERE DEPTNO=V_DEPTNO;
 19  SELECT  GRADE INTO V_GRADE FROM SALGRADE
 20  WHERE V_SAL BETWEEN LOSAL AND HISAL;
 21  */
 22       &D('ENAME IS ...'||V_ENAME);
 23       &D('JOB IS ...'||V_JOB);
 24       &D('SALARY  IS ...'||V_SAL);
 25       &D('DEPTNO IS ...'||V_DEPTNO);
 26       &D('DNAME IS ...'||V_DNAME);
 27       &D('GRADE IS ...'||V_GRADE);
 28* END;
SQL> /
Enter value for emp_id: 7788
DECLARE
*
ERROR at line 1:
ORA-01403: no data found
ORA-06512: at line 10


SQL>
SQL>
```

```
SQL>
SQL>
SQL> ED
Wrote file afiedt.buf

  1  DECLARE
  2              V_EMPNO EMP.EMPNO%TYPE:=&EMP_ID;
  3              V_ENAME EMP.ENAME%TYPE;
  4              V_JOB   EMP.JOB%TYPE;
  5              V_SAL   EMP.SAL%TYPE;
  6                      V_DEPTNO EMP.DEPTNO%TYPE;
  7                   V_DNAME  DEPT.DNAME%TYPE;
  8                    V_GRADE SALGRADE.GRADE%TYPE;
  9  BEGIN
 10  /*
 11     SELECT E.ENAME,E.JOB,E.SAL,E.DEPTNO,D.dname,G.GRADE
 12     INTO
 13            V_ENAME,V_JOB,V_SAL,V_DEPTNO,V_DNAME,V_GRADE
 14  FROM EMP E,DEPT D,SALGRADE G
 15  WHERE E.DEPTNO= 20
 16  AND E.SAL BETWEEN G.HISAL AND G.LOSAL
 17  AND  E.EMPNO=V_EMPNO;
 18  */
 19   SELECT E.ENAME,E.JOB,E.SAL,E.DEPTNO,D.dname,G.GRADE
 20     INTO
 21            V_ENAME,V_JOB,V_SAL,V_DEPTNO,V_DNAME,V_GRADE
 22  FROM EMP E,DEPT D,SALGRADE G
 23  WHERE E.EMPNO=V_EMPNO
 24  AND E.DEPTNO=D.DEPTNO
 25  AND E.SAL BETWEEN G.LOSAL AND G.HISAL;
 26  /*SELECT DNAME INTO V_DNAME FROM DEPT
 27  WHERE DEPTNO=V_DEPTNO;
 28  SELECT  GRADE INTO V_GRADE FROM SALGRADE
 29  WHERE V_SAL BETWEEN LOSAL AND HISAL;
 30  */
 31       &D('ENAME IS ...'||V_ENAME);
 32       &D('JOB IS ...'||V_JOB);
 33       &D('SALARY  IS ...'||V_SAL);
 34       &D('DEPTNO IS ...'||V_DEPTNO);
 35       &D('DNAME IS ...'||V_DNAME);
 36       &D('GRADE IS ...'||V_GRADE);
 37* END;
 38  /
Enter value for emp_id: 7788
ENAME IS ...SCOTT

JOB IS ...ANALYST

SALARY  IS ...3000

DEPTNO IS ...20

DNAME IS ...RESEARCH

GRADE IS ...4


PL/SQL procedure successfully completed.

SQL>
SQL>
SQL>
SQL>
```

```
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL> ED
Wrote file afiedt.buf

  1  DECLARE
  2              V_EMPNO EMP.EMPNO%TYPE:=&EMP_ID;
  3              V_ENAME EMP.ENAME%TYPE;
  4              V_JOB   EMP.JOB%TYPE;
  5              V_SAL   EMP.SAL%TYPE;
  6                      V_DEPTNO EMP.DEPTNO%TYPE;
  7                   V_DNAME  DEPT.DNAME%TYPE;
  8                    V_GRADE SALGRADE.GRADE%TYPE;
  9  BEGIN
 10  /*
 11     SELECT E.ENAME,E.JOB,E.SAL,E.DEPTNO,D.dname,G.GRADE
 12     INTO
 13             V_ENAME,V_JOB,V_SAL,V_DEPTNO,V_DNAME,V_GRADE
 14  FROM EMP E,DEPT D,SALGRADE G
 15  WHERE E.DEPTNO= 20
 16  AND E.SAL BETWEEN G.HISAL AND G.LOSAL
 17  AND  E.EMPNO=V_EMPNO;
 18  */
 19   SELECT E.ENAME,E.JOB,E.SAL,E.DEPTNO,D.dname,G.GRADE
 20     INTO
 21             V_ENAME,V_JOB,V_SAL,V_DEPTNO,V_DNAME,V_GRADE
 22  FROM EMP E,DEPT D,SALGRADE G
 23  WHERE E.EMPNO=V_EMPNO
 24  AND E.DEPTNO=D.DEPTNO
 25  AND E.SAL BETWEEN G.LOSAL AND G.HISAL;
 26  /*SELECT DNAME INTO V_DNAME FROM DEPT
 27  WHERE DEPTNO=V_DEPTNO;
 28  SELECT  GRADE INTO V_GRADE FROM SALGRADE
 29  WHERE V_SAL BETWEEN LOSAL AND HISAL;
 30  */
 31       &D('ENAME IS ...'||V_ENAME);
 32       &D('JOB IS ...'||V_JOB);
 33       &D('SALARY  IS ...'||V_SAL);
 34       &D('DEPTNO IS ...'||V_DEPTNO);
 35       &D('DNAME IS ...'||V_DNAME);
 36       &D('GRADE IS ...'||V_GRADE);
 37* END;
SQL>
SQL> ED
Wrote file afiedt.buf

  1  DECLARE
  2              V_EMPNO EMP.EMPNO%TYPE:=&EMP_ID;
  3              V_ENAME EMP.ENAME%TYPE;
  4              V_JOB   EMP.JOB%TYPE;
  5              V_SAL   EMP.SAL%TYPE;
  6                      V_DEPTNO EMP.DEPTNO%TYPE;
  7                   V_DNAME  DEPT.DNAME%TYPE;
  8                    V_GRADE SALGRADE.GRADE%TYPE;
  9  BEGIN
 10  /*
```

```
 11     SELECT E.ENAME,E.JOB,E.SAL,E.DEPTNO,D.dname,G.GRADE
 12     INTO
 13             V_ENAME,V_JOB,V_SAL,V_DEPTNO,V_DNAME,V_GRADE
 14   FROM EMP E,DEPT D,SALGRADE G
 15   WHERE E.DEPTNO= 20
 16   AND E.SAL BETWEEN G.HISAL AND G.LOSAL
 17   AND  E.EMPNO=V_EMPNO;
 18   */
 19    SELECT E.ENAME,E.JOB,E.SAL,E.DEPTNO,D.dname,G.GRADE
 20      INTO
 21             V_ENAME,V_JOB,V_SAL,V_DEPTNO,V_DNAME,V_GRADE
 22   FROM EMP E,DEPT D,SALGRADE G
 23   WHERE E.EMPNO=V_EMPNO
 24   AND E.DEPTNO=D.DEPTNO
 25   AND E.SAL BETWEEN G.LOSAL AND G.HISAL;
 26   /*SELECT DNAME INTO V_DNAME FROM DEPT
 27   WHERE DEPTNO=V_DEPTNO;
 28   SELECT  GRADE INTO V_GRADE FROM SALGRADE
 29   WHERE V_SAL BETWEEN LOSAL AND HISAL;
 30   */
 31        &D('ENAME IS ...'||V_ENAME);
 32        &D('JOB IS ...'||V_JOB);
 33        &D('SALARY  IS ...'||V_SAL);
 34        &D('DEPTNO IS ...'||V_DEPTNO);
 35        &D('DNAME IS ...'||V_DNAME);
 36        &D('GRADE IS ...'||V_GRADE);
 37* END;
SQL>
SQL>
SQL> ED
Wrote file afiedt.buf

  1  DECLARE
  2              V_EMPNO EMP.EMPNO%TYPE:=&EMP_ID;
  3              V_ENAME EMP.ENAME%TYPE;
  4              V_JOB   EMP.JOB%TYPE;
  5              V_SAL   EMP.SAL%TYPE;
  6                      V_DEPTNO EMP.DEPTNO%TYPE;
  7  BEGIN
  8   SELECT E.ENAME,E.JOB,E.SAL,E.DEPTNO
  9     INTO
 10             V_ENAME,V_JOB,V_SAL,V_DEPTNO
 11  FROM EMP E
 12  WHERE E.EMPNO=V_EMPNO;
 13        &D('ENAME IS ...'||V_ENAME);
 14        &D('JOB IS ...'||V_JOB);
 15        &D('SALARY  IS ...'||V_SAL);
 16        &D('DEPTNO IS ...'||V_DEPTNO);
 17* END;
 18  /
Enter value for emp_id: 7788
ENAME IS ...SCOTT

JOB IS ...ANALYST

SALARY  IS ...3000

DEPTNO IS ...20


PL/SQL procedure successfully completed.

SQL>
```

```
SQL>
SQL>
SQL>
SQL>
SQL> ED
Wrote file afiedt.buf

  1  DECLARE
  2             V_EMPNO EMP.EMPNO%TYPE:=&EMP_ID;
  3             EMP_REC  EMP%ROWTYPE;
  4  BEGIN
  5   SELECT *
  6     INTO
  7            EMP_REC
  8  FROM EMP E
  9  WHERE E.EMPNO=V_EMPNO;
 10       &D('ENAME IS ...'||EMP_REC.ENAME);
 11       &D('JOB IS ...'||EMP_REC.JOB);
 12       &D('SALARY  IS ...'||EMP_REC.SAL);
 13       &D('DEPTNO IS ...'||EMP_REC.DEPTNO);
 14* END;
 15  /
Enter value for emp_id: 7788
ENAME IS ...SCOTT

JOB IS ...ANALYST

SALARY  IS ...3000

DEPTNO IS ...20


PL/SQL procedure successfully completed.

SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL> ED
Wrote file afiedt.buf

  1  DECLARE
  2             V_EMPNO EMP.EMPNO%TYPE:=&EMP_ID;
  3             EMP_REC  EMP%ROWTYPE;
  4  BEGIN
  5   SELECT ENAME,JOB,SAL,DEPTNO
  6     INTO
  7            EMP_REC
  8  FROM EMP E
  9  WHERE E.EMPNO=V_EMPNO;
 10       &D('ENAME IS ...'||EMP_REC.ENAME);
 11       &D('JOB IS ...'||EMP_REC.JOB);
 12       &D('SALARY  IS ...'||EMP_REC.SAL);
 13       &D('DEPTNO IS ...'||EMP_REC.DEPTNO);
 14* END;
SQL> /
Enter value for emp_id: 7788
FROM EMP E
*
ERROR at line 8:
ORA-06550: line 7, column 19:
```

```
PL/SQL: ORA-00913: too many values
ORA-06550: line 5, column 2:
PL/SQL: SQL Statement ignored


SQL>
SQL>
SQL>
SQL>
SQL> ED
Wrote file afiedt.buf

  1  DECLARE
  2              V_EMPNO EMP.EMPNO%TYPE:=&EMP_ID;
  3              EMP_REC  EMP%ROWTYPE;
  4  BEGIN
  5   SELECT *
  6     INTO
  7            EMP_REC
  8  FROM EMP E
  9  WHERE E.EMPNO=V_EMPNO;
 10       &D('ENAME IS ...'||EMP_REC.ENAME);
 11       &D('JOB IS ...'||EMP_REC.JOB);
 12       &D('SALARY  IS ...'||EMP_REC.SAL);
 13       &D('DEPTNO IS ...'||EMP_REC.DEPTNO);
 14* END;
SQL> /
Enter value for emp_id: 7788
ENAME IS ...SCOTT

JOB IS ...ANALYST

SALARY  IS ...3000

DEPTNO IS ...20


PL/SQL procedure successfully completed.

SQL>
SQL>
SQL>
SQL> ED
Wrote file afiedt.buf

  1   SELECT *
  2  FROM EMP E,DEPT D,SALGRADE G
  3  WHERE  E.DEPTNO=D.DEPTNO
  4* AND E.SAL BETWEEN G.LOSAL AND G.HISAL;
  5  /
AND E.SAL BETWEEN G.LOSAL AND G.HISAL;
                                      *
ERROR at line 4:
ORA-00911: invalid character


SQL>
SQL>
SQL> ED
Wrote file afiedt.buf

  1   SELECT *
  2  FROM EMP E,DEPT D,SALGRADE G
```

```
  3  WHERE  E.DEPTNO=D.DEPTNO
  4* AND E.SAL BETWEEN G.LOSAL AND G.HISAL
SQL> /

     EMPNO ENAME      JOB              MGR HIREDATE          SAL        COMM
DEPTNO     DEPTNO
---------- ---------- --------- ---------- --------- ---------- ----------
---------- ----------
DNAME          LOC                 GRADE      LOSAL      HISAL

-------------- ------------- ---------- ---------- ----------

      7369 SMITH      CLERK            7902 17-DEC-80        800
20         20
RESEARCH       DALLAS                   1        700       1200


      7900 JAMES      CLERK            7698 03-DEC-81        950
30         30
SALES          CHICAGO                  1        700       1200


      7876 ADAMS      CLERK            7788 23-MAY-87       1100
20         20
RESEARCH       DALLAS                   1        700       1200


      7521 WARD       SALESMAN         7698 22-FEB-81       1250        500
30         30
SALES          CHICAGO                  2       1201       1400


      7654 MARTIN     SALESMAN         7698 28-SEP-81       1250       1400
30         30
SALES          CHICAGO                  2       1201       1400


      7934 MILLER     CLERK            7782 23-JAN-85       1300
10         10
ACCOUNTING     NEW YORK                 2       1201       1400


      7844 TURNER     SALESMAN         7698 08-SEP-81       1500          0
30         30
SALES          CHICAGO                  3       1401       2000


      7499 ALLEN      SALESMAN         7698 20-FEB-81       1600        300
30         30
SALES          CHICAGO                  3       1401       2000


      7782 CLARK      MANAGER          7839 09-JUN-81       2450
10         10
ACCOUNTING     NEW YORK                 4       2001       3000
```

```
      7698 BLAKE      MANAGER          7839 01-MAY-81       2850
30         30
SALES          CHICAGO                4       2001       3000


      7566 JONES      MANAGER          7839 02-APR-81       2975
20         20
RESEARCH       DALLAS                 4       2001       3000


      7788 SCOTT      ANALYST          7566 19-APR-87       3000
20         20
RESEARCH       DALLAS                 4       2001       3000


      7839 KING       PRESIDENT             17-NOV-81       5000
10         10
ACCOUNTING     NEW YORK               5       3001       9999



13 rows selected.

SQL> SET LINE 10000
SQL> /

     EMPNO ENAME      JOB              MGR HIREDATE          SAL       COMM
DEPTNO     DEPTNO DNAME          LOC                 GRADE      LOSAL      HISAL
```

---------- ---------- --------- ---------- --------- ---------- ----------
---------- ---------- ---------- ---------- ---------- ---------- ----------

```
      7369 SMITH      CLERK           7902 17-DEC-80        800
20          20 RESEARCH       DALLAS                  1        700       1200
```

```
      7900 JAMES      CLERK           7698 03-DEC-81        950
30          30 SALES          CHICAGO                1        700       1200
```

```
      7876 ADAMS      CLERK           7788 23-MAY-87       1100
20          20 RESEARCH       DALLAS                  1        700       1200
```

```
      7521 WARD       SALESMAN        7698 22-FEB-81       1250        500
30          30 SALES           CHICAGO                    2       1201       1400
```

```
      7654 MARTIN     SALESMAN        7698 28-SEP-81       1250       1400
30          30 SALES          CHICAGO                2       1201       1400
```

```
      7934 MILLER     CLERK           7782 23-JAN-85       1300
10          10 ACCOUNTING     NEW YORK                2       1201       1400
```

```
      7844 TURNER     SALESMAN        7698 08-SEP-81       1500          0
30          30 SALES          CHICAGO                  3       1401       2000
```

```
      7499 ALLEN      SALESMAN        7698 20-FEB-81       1600        300
30          30 SALES          CHICAGO                3       1401       2000
```

```
      7782 CLARK      MANAGER          7839 09-JUN-81       2450
10          10 ACCOUNTING     NEW YORK                4       2001       3000
```

```
      7698 BLAKE      MANAGER         7839 01-MAY-81       2850
30          30 SALES          CHICAGO                4       2001       3000
```

```
      7566 JONES      MANAGER          7839 02-APR-81       2975
20          20 RESEARCH       DALLAS                  4       2001       3000
```

```
      7788 SCOTT      ANALYST         7566 19-APR-87       3000
20          20 RESEARCH       DALLAS                    4       2001       3000
```

```
      7839 KING       PRESIDENT            17-NOV-81       5000
10          10 ACCOUNTING     NEW YORK                5       3001       9999
```

```
13 rows selected.

SQL>
SQL>
SQL>
SQL>
SQL> ED
Wrote file afiedt.buf

  1   SELECT *
  2  FROM EMP E,DEPT D,SALGRADE G
  3  WHERE  E.DEPTNO=D.DEPTNO
  4* AND E.SAL BETWEEN G.LOSAL AND G.HISAL
SQL> .
SQL>
SQL>
SQL>
SQL>
SQL> SELECT * FROM EMP_INFO;

     EMPNO ENAME      JOB              SAL DNAME          LOC                GRADE
```

---------- ---------- --------- ---------- -------------- ------------- ----------

7369 SMITH CLERK 800 RESEARCH DALLAS 1

7900 JAMES CLERK 950 SALES CHICAGO 1

7876	ADAMS	CLERK	1100	RESEARCH	DALLAS	1

7521	WARD	SALESMAN	1250	SALES	CHICAGO	2

7654	MARTIN	SALESMAN	1250	SALES	CHICAGO	2

7934 MILLER CLERK 1300 ACCOUNTING NEW YORK 2

7844 TURNER SALESMAN 1500 SALES CHICAGO 3

7499 ALLEN SALESMAN 1600 SALES CHICAGO 3

7782 CLARK MANAGER 2450 ACCOUNTING NEW YORK 4

7698 BLAKE MANAGER 2850 SALES CHICAGO 4

7566	JONES	MANAGER	2975	RESEARCH	DALLAS	4

7788	SCOTT	ANALYST	3000	RESEARCH	DALLAS	4

7839 KING PRESIDENT 5000 ACCOUNTING NEW YORK 5

```
13 rows selected.

SQL>
SQL>
SQL>
SQL>
SQL>
SQL> ED
Wrote file afiedt.buf

  1  DECLARE
  2  V_EMPNO EMP.EMPNO%TYPE:=&EMPNO;
  3  EMP_REC EMP_INFO%ROWTYPE;
  4  BEGIN
  5  SELECT * INTO EMP_REC  FROM EMP_INFO
  6  WHERE EMPNO=V_EMPNO;
  7  &D('ENAME IS ....'||EMP_REC.ENAME);
  8  &D('JOB IS ....'||EMP_REC.JOB);
  9  &D('DNAME IS ....'||EMP_REC.DNAME);
 10  &D('GRADE IS ....'||EMP_REC.GRADE);
 11* END;
 12  /
Enter value for empno: 7788
ENAME IS ....SCOTT
```

JOB IS ....ANALYST

DNAME IS ....RESEARCH

GRADE IS 4

```
PL/SQL procedure successfully completed.

SQL>
SQL>
```

```
SQL>
SQL>
SQL>
SQL>
SQL> ED
Wrote file afiedt.buf

  1  DECLARE
  2  V_EMPNO EMP.EMPNO%TYPE:=&EMPNO;
  3  EMP_REC EMP_INFO%ROWTYPE;
  4  BEGIN
  5  SELECT * INTO EMP_REC  FROM EMP_INFO
  6  WHERE EMPNO=V_EMPNO;
  7  &D('ENAME IS ....'||EMP_REC.ENAME);
  8  &D('JOB IS ....'||EMP_REC.JOB);
  9  &D('DEPTNO IS ....'||EMP_REC.DEPTNO);
 10  &D('DNAME IS ....'||EMP_REC.DNAME);
 11  &D('GRADE IS ....'||EMP_REC.GRADE);
 12* END;
SQL> /
Enter value for empno: 7839
DBMS_OUTPUT.PUT_LINE('DEPTNO IS ....'||EMP_REC.DEPTNO);
                                       *
ERROR at line 9:
ORA-06550: line 9, column 48:
PLS-00302: component 'DEPTNO' must be declared
ORA-06550: line 9, column 1:
PL/SQL: Statement ignored


SQL>
SQL>
SQL>
SQL>
SQL>
SQL> SELECT * FROM EMP_INFO;

     EMPNO ENAME      JOB              SAL DNAME          LOC                 GRADE
```

---------- ---------- --------- ---------- ------------- ------------ ----------

7369	SMITH	CLERK	800	RESEARCH	DALLAS	1

7900	JAMES	CLERK	950	SALES	CHICAGO	1

7876	ADAMS	CLERK	1100	RESEARCH	DALLAS	1

7521 WARD SALESMAN 1250 SALES CHICAGO 2

7654 MARTIN SALESMAN 1250 SALES CHICAGO 2

7934 MILLER CLERK 1300 ACCOUNTING NEW YORK 2

7844 TURNER SALESMAN 1500 SALES CHICAGO 3

7499	ALLEN	SALESMAN	1600	SALES	CHICAGO	3

7782 CLARK MANAGER 2450 ACCOUNTING NEW YORK 4

7698	BLAKE	MANAGER	2850	SALES	CHICAGO	4

7566 JONES MANAGER 2975 RESEARCH DALLAS 4

7788 SCOTT ANALYST 3000 RESEARCH DALLAS 4

7839 KING PRESIDENT 5000 ACCOUNTING NEW YORK 5

13 rows selected.

SQL> /

EMPNO	ENAME	JOB	SAL	DEPTNO	DNAME	LOC	GRADE

---------- ---------- --------- ---------- ---------- -------------- -------------
----

7369 SMITH CLERK 800 20 RESEARCH DALLAS
1

7900 JAMES CLERK 950 30 SALES CHICAGO
1

7876 ADAMS CLERK 1100 20 RESEARCH DALLAS
1

7521 WARD SALESMAN 1250 30 SALES CHICAGO
2

```
7654 MARTIN     SALESMAN         1250         30 SALES          CHICAGO
 2
```

7934 MILLER CLERK 1300 10 ACCOUNTING NEW YORK
2

7844 TURNER SALESMAN 1500 30 SALES CHICAGO
3

7499 ALLEN SALESMAN 1600 30 SALES CHICAGO
3

7782 CLARK MANAGER 2450 10 ACCOUNTING NEW YORK
4

7698 BLAKE MANAGER 2850 30 SALES CHICAGO
4

7566 JONES MANAGER 2975 20 RESEARCH DALLAS
4

7788 SCOTT ANALYST 3000 20 RESEARCH DALLAS
4

7839 KING PRESIDENT 5000 10 ACCOUNTING NEW YORK

5

```
13 rows selected.

SQL>
SQL>  DECLARE
  2   V_EMPNO EMP.EMPNO%TYPE:=&EMPNO;
  3   EMP_REC EMP_INFO%ROWTYPE;
  4   BEGIN
  5   SELECT * INTO EMP_REC  FROM EMP_INFO
```

```
  6   WHERE EMPNO=V_EMPNO;
  7   &D('ENAME IS ....'||EMP_REC.ENAME);
  8   &D('JOB IS ....'||EMP_REC.JOB);
  9   &D('DEPTNO IS ....'||EMP_REC.DEPTNO);
 10   &D('DNAME IS ....'||EMP_REC.DNAME);
 11   &D('GRADE IS ....'||EMP_REC.GRADE);
 12   END;
 13  /
Enter value for empno: 7788
ENAME IS ....SCOTT
```

```
JOB IS ....ANALYST
```

DEPTNO IS 20

DNAME IS ....RESEARCH

GRADE IS 4

```
PL/SQL procedure successfully completed.

SQL>
SQL>
SQL>
SQL>
SQL>
SQL> /
Enter value for empno: 7788
 DBMS_OUTPUT.PUT_LINE('DEPTNO IS ....'||EMP_REC.DEPTNO);
                                                 *
ERROR at line 9:
ORA-06550: line 9, column 49:
PLS-00302: component 'DEPTNO' must be declared
ORA-06550: line 9, column 2:
PL/SQL: Statement ignored


SQL> ED
Wrote file afiedt.buf

  1   DECLARE
  2   V_EMPNO EMP.EMPNO%TYPE:=&EMPNO;
  3   EMP_REC EMP_INFO%ROWTYPE;
  4   BEGIN
  5   SELECT * INTO EMP_REC  FROM EMP_INFO
  6   WHERE EMPNO=V_EMPNO;
  7   &D('ENAME IS ....'||EMP_REC.ENAME);
  8   &D('JOB IS ....'||EMP_REC.JOB);
  9   &D('DEPTNO IS ....'||EMP_REC.DEPT_ID);
```

```
 10   &D('DNAME IS ....'||EMP_REC.DNAME);
 11   &D('GRADE IS ....'||EMP_REC.GRADE);
 12*  END;
SQL> /
Enter value for empno: 7788
ENAME IS ....SCOTT
```

JOB IS ....ANALYST

DEPTNO IS 20

DNAME IS ....RESEARCH

GRADE IS 4

```
PL/SQL procedure successfully completed.

SQL>
SQL>
SQL>
SQL>
SQL>
SQL> SPOOL OFF
```

```
SQL>
SQL>
SQL> DECLARE
  2  FIRST_NO NUMBER :=&FIRST_NO;
  3  SECOND_NO NUMBER :=&SECOND_NO;
  4  BEGIN
  5  IF FIRST_NO>SECOND_NO THEN
  6  &D.
  7  ED
  8  .
SQL> ED
Wrote file afiedt.buf

  1  DECLARE
  2  FIRST_NO NUMBER :=&FIRST_NO;
  3  SECOND_NO NUMBER :=&SECOND_NO;
  4  BEGIN
  5  IF FIRST_NO>SECOND_NO THEN
  6  &D('HIGHEST NO IS ....'||FIRST_NO);
  7  ELSE
  8  &D('HIGHEST NO IS ....'||SECOND_NO);
  9  END IF;
 10* END;
 11  /
Enter value for first_no: 5
Enter value for second_no: 4
HIGHEST NO IS ....5


PL/SQL procedure successfully completed.

SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL> ED
Wrote file afiedt.buf

  1  DECLARE
  2  FIRST_NO NUMBER :=&FIRST_NO;
  3  SECOND_NO NUMBER :=&SECOND_NO;
  4  BEGIN
  5  IF FIRST_NO>SECOND_NO THEN
  6  &D('HIGHEST NO IS(I M IN TRUE STAT) ....'||FIRST_NO);
  7  ELSE
  8  &D('HIGHEST NO IS(I M IN FALSE STAT) ....'||SECOND_NO);
  9  END IF;
 10* END;
SQL> /
Enter value for first_no: 5
Enter value for second_no: 4
HIGHEST NO IS(I M IN TRUE STAT) ....5


PL/SQL procedure successfully completed.
```

```
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL> /
Enter value for first_no: 8
Enter value for second_no: 9
HIGHEST NO IS(I M IN FALSE STAT) ....9


PL/SQL procedure successfully completed.

SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL> /
Enter value for first_no: 5
Enter value for second_no: 5
HIGHEST NO IS(I M IN FALSE STAT) ....5


PL/SQL procedure successfully completed.

SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL> ED
Wrote file afiedt.buf

  1  DECLARE
  2  FIRST_NO NUMBER :=&FIRST_NO;
  3  SECOND_NO NUMBER :=&SECOND_NO;
  4  BEGIN
  5  IF FIRST_NO>SECOND_NO THEN
  6  &D('HIGHEST NO IS(I M IN 1ST TRUE STAT) ....'||FIRST_NO);
  7  ELSIF SECOND_NO>FIRST_NO THEN
  8  &D('HIGHEST NO IS(I M IN 2N TRUE STAT) ....'||SECOND_NO);
  9  ELSE
 10  ELSE
 11  &D('BOTH ARE EQUALS NUMBERS');
 12  END IF;
 13* END;
SQL> /
Enter value for first_no:
Enter value for second_no:
```

```
FIRST_NO NUMBER :=;
                 *
ERROR at line 2:
ORA-06550: line 2, column 19:
PLS-00103: Encountered the symbol ";" when expecting one of the following:
( - + case mod new not null <an identifier>
<a double-quoted delimited-identifier> <a bind variable> avg
count current exists max min prior sql stddev sum variance
execute forall merge time timestamp interval date
<a string literal with character set specification>
<a number> <a single-quoted SQL string> pipe
<an alternatively-quoted string literal with character set specification>
<an alternatively-quoted S
ORA-06550: line 3, column 20:
PLS-00103: Encountered the symbol ";" when expecting one of the following:
( - + case mod new not null <an identifier>
<a double-quoted delimited-identifier> <a bind variable> avg
count current exists max min prior sql stddev su
ORA-06550: line 10, column 1:
PLS-00103: Encountered the symbol "ELSE" when expecting one of the following:
begin case declare exit for goto if loop mod null pragma
raise return select update while with <a


SQL>
SQL>
SQL> ED
Wrote file afiedt.buf

  1  DECLARE
  2  FIRST_NO NUMBER :=&FIRST_NO;
  3  SECOND_NO NUMBER :=&SECOND_NO;
  4  BEGIN
  5  IF FIRST_NO>SECOND_NO THEN
  6  &D('HIGHEST NO IS(I M IN 1ST TRUE STAT) ....'||FIRST_NO);
  7  ELSIF SECOND_NO>FIRST_NO THEN
  8  &D('HIGHEST NO IS(I M IN 2N TRUE STAT) ....'||SECOND_NO);
  9  ELSE
 10  &D('BOTH ARE EQUALS NUMBERS');
 11  END IF;
 12* END;
 13  /
Enter value for first_no: 5
Enter value for second_no: 4
HIGHEST NO IS(I M IN 1ST TRUE STAT) ....5


PL/SQL procedure successfully completed.

SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL> /
Enter value for first_no: 8
Enter value for second_no: 9
HIGHEST NO IS(I M IN 2N TRUE STAT) ....9


PL/SQL procedure successfully completed.
```

```
SQL>
SQL>
SQL>
SQL>
SQL>
SQL> ED
Wrote file afiedt.buf

  1  DECLARE
  2  FIRST_NO NUMBER :=&FIRST_NO;
  3  SECOND_NO NUMBER :=&SECOND_NO;
  4  BEGIN
  5  IF FIRST_NO>SECOND_NO THEN
  6  &D('HIGHEST NO IS(I M IN 1ST TRUE STAT) ....'||FIRST_NO);
  7  ELSIF SECOND_NO>FIRST_NO THEN
  8  &D('HIGHEST NO IS(I M IN 2N TRUE STAT) ....'||SECOND_NO);
  9  ELSE
 10  &D('BOTH ARE EQUALS NUMBERS');
 11  END IF;
 12* END;
SQL> /
Enter value for first_no: 8
Enter value for second_no: 8
BOTH ARE EQUALS NUMBERS


PL/SQL procedure successfully completed.

SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL> /
Enter value for first_no: 0
Enter value for second_no: 0
BOTH ARE EQUALS NUMBERS


PL/SQL procedure successfully completed.

SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL> ED
Wrote file afiedt.buf

  1  DECLARE
  2  FIRST_NO NUMBER :=&FIRST_NO;
  3  SECOND_NO NUMBER :=&SECOND_NO;
  4  BEGIN
  5  IF FIRST_NO>SECOND_NO THEN
  6  &D('HIGHEST NO IS(I M IN 1ST TRUE STAT) ....'||FIRST_NO);
  7  ELSIF SECOND_NO>FIRST_NO THEN
  8  &D('HIGHEST NO IS(I M IN 2N TRUE STAT) ....'||SECOND_NO);
  9  ELSIF FIRST_NO=0 AND SECOND_NO=0 THEN
```

```
 10  &D('ZERO VALUES');
 11  ELSE
 12  &D('BOTH ARE EQUALS NUMBERS');
 13  END IF;
 14* END;
 15  /
Enter value for first_no: 8
Enter value for second_no: 6
HIGHEST NO IS(I M IN 1ST TRUE STAT) ....8


PL/SQL procedure successfully completed.

SQL>
SQL>
SQL>
SQL>
SQL> /
Enter value for first_no: 7
Enter value for second_no: 9
HIGHEST NO IS(I M IN 2N TRUE STAT) ....9


PL/SQL procedure successfully completed.

SQL>
SQL>
SQL>
SQL> /
Enter value for first_no: 0
Enter value for second_no: 0
ZERO VALUES


PL/SQL procedure successfully completed.

SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL> /
Enter value for first_no: 8
Enter value for second_no: 8
BOTH ARE EQUALS NUMBERS


PL/SQL procedure successfully completed.

SQL>
SQL>
SQL>
SQL>
SQL>
SQL> ED
Wrote file afiedt.buf

  1  DECLARE
  2  FIRST_NO NUMBER :=&FIRST_NO;
  3  SECOND_NO NUMBER :=&SECOND_NO;
  4  THIRD_NO NUMBER :=&THIRD_NO;
  5  BEGIN
```

```
  6  IF FIRST_NO>SECOND_NO  AND FIRST_NO>THIRD_NO THEN
  7  &D('HIGHEST NO IS(I M IN 1ST TRUE STAT) ....'||FIRST_NO);
  8  ELSIF SECOND_NO>FIRST_NO AND SECOND_NO>THIRD_NO THEN
  9  &D('HIGHEST NO IS(I M IN 2N TRUE STAT) ....'||SECOND_NO);
 10  ELSIF THIRD_NO>FIRST_NO AND THIRD_NO>SECOND_NO THEN
 11  &D('HIGHEST NO IS(I M IN 3RD TRUE STAT) ....'||THIRD_NO);
 12  ELSIF FIRST_NO=0 AND SECOND_NO=0 THEN
 13  &D('ZERO VALUES');
 14  ELSE
 15  &D('BOTH ARE EQUALS NUMBERS');
 16  END IF;
 17* END;
 18  /
Enter value for first_no: 7
Enter value for second_no: 8
Enter value for third_no: 9
HIGHEST NO IS(I M IN 3RD TRUE STAT) ....9


PL/SQL procedure successfully completed.

SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL> /
Enter value for first_no: 8
Enter value for second_no: 5
Enter value for third_no: 9
HIGHEST NO IS(I M IN 3RD TRUE STAT) ....9


PL/SQL procedure successfully completed.

SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL> /
Enter value for first_no: 0
Enter value for second_no: 0
Enter value for third_no: 0
ZERO VALUES


PL/SQL procedure successfully completed.

SQL>
SQL>
SQL>
SQL>
SQL>
SQL> ED
Wrote file afiedt.buf

  1  DECLARE
  2  FIRST_NO NUMBER :=&FIRST_NO;
  3  SECOND_NO NUMBER :=&SECOND_NO;
```

```
  4  THIRD_NO NUMBER :=&THIRD_NO;
  5  BEGIN
  6  IF FIRST_NO>SECOND_NO  AND FIRST_NO>THIRD_NO THEN
  7  &D('HIGHEST NO IS(I M IN 1ST TRUE STAT) ....'||FIRST_NO);
  8  ELSIF SECOND_NO>FIRST_NO AND SECOND_NO>THIRD_NO THEN
  9  &D('HIGHEST NO IS(I M IN 2N TRUE STAT) ....'||SECOND_NO);
 10  ELSIF THIRD_NO>FIRST_NO AND THIRD_NO>SECOND_NO THEN
 11  &D('HIGHEST NO IS(I M IN 3RD TRUE STAT) ....'||THIRD_NO);
 12  ELSIF FIRST_NO=0 AND SECOND_NO=0  AND THIRD_NO=0 THEN
 13  &D('ZERO VALUES');
 14  ELSE
 15  &D('ALL ARE EQUALS NUMBERS');
 16  END IF;
 17* END;
SQL> /
Enter value for first_no: 7
Enter value for second_no: 8
Enter value for third_no: 9
HIGHEST NO IS(I M IN 3RD TRUE STAT) ....9


PL/SQL procedure successfully completed.

SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL> ED
Wrote file afiedt.buf

  1  DECLARE
  2  FIRST_NO NUMBER :=&FIRST_NO;
  3  SECOND_NO NUMBER :=&SECOND_NO;
  4  THIRD_NO NUMBER :=&THIRD_NO;
  5  BEGIN
  6  IF FIRST_NO>SECOND_NO  AND FIRST_NO>THIRD_NO THEN
  7  &D('1S HIGHEST NO IS ....'||FIRST_NO);
  8             IF SECON_NO>THIRD_NO THEN
  9              &D('2ND HIGHEST NO IS ....'||SECOND_NO);
 10              &D('3RD HIGHEST NO IS ....'||THIRD_NO);
 11            ELSIF THIRD_NO>SECOND_NO THEN
 12              &D('2ND HIGHEST NO IS ....'||THIRD_NO);
 13              &D('3RD HIGHEST NO IS ....'||SECOND_NO);
 14             ELSE
 15             &('2ND AND 3RD ARE EQUALS ...');
 16             END IF;
 17  ELSIF SECOND_NO>FIRST_NO AND SECOND_NO>THIRD_NO THEN
 18  &D('HIGHEST NO IS(I M IN 2N TRUE STAT) ....'||SECOND_NO);
 19  ELSIF THIRD_NO>FIRST_NO AND THIRD_NO>SECOND_NO THEN
 20  &D('HIGHEST NO IS(I M IN 3RD TRUE STAT) ....'||THIRD_NO);
 21  ELSIF FIRST_NO=0 AND SECOND_NO=0  AND THIRD_NO=0 THEN
 22  &D('ZERO VALUES');
 23  ELSE
 24  &D('ALL ARE EQUALS NUMBERS');
 25  END IF;
 26* END;
 27
 28  /
Enter value for first_no: 100
Enter value for second_no: 9
Enter value for third_no: 8
```

```
          &('2ND AND 3RD ARE EQUALS ...');
          *
ERROR at line 15:
ORA-06550: line 15, column 11:
PLS-00103: Encountered the symbol "&" when expecting one of the following:
begin case declare exit for goto if loop mod null pragma
raise return select update while with <an identifier>
<a double-quoted delimited-identifier> <a bind variable> <<
close current delete fetch lock insert open rollback
savepoint set sql execute commit forall merge pipe
The symbol "mod was inserted before "&" to continue.


SQL> ED
Wrote file afiedt.buf

  1  DECLARE
  2  FIRST_NO NUMBER :=&FIRST_NO;
  3  SECOND_NO NUMBER :=&SECOND_NO;
  4  THIRD_NO NUMBER :=&THIRD_NO;
  5  BEGIN
  6  IF FIRST_NO>SECOND_NO  AND FIRST_NO>THIRD_NO THEN
  7  &D('1S HIGHEST NO IS ....'||FIRST_NO);
  8             IF SECOND_NO>THIRD_NO THEN
  9              &D('2ND HIGHEST NO IS ....'||SECOND_NO);
 10              &D('3RD HIGHEST NO IS ....'||THIRD_NO);
 11            ELSIF THIRD_NO>SECOND_NO THEN
 12              &D('2ND HIGHEST NO IS ....'||THIRD_NO);
 13              &D('3RD HIGHEST NO IS ....'||SECOND_NO);
 14             ELSE
 15             &D('2ND AND 3RD ARE EQUALS ...');
 16             END IF;
 17  ELSIF SECOND_NO>FIRST_NO AND SECOND_NO>THIRD_NO THEN
 18  &D('HIGHEST NO IS(I M IN 2N TRUE STAT) ....'||SECOND_NO);
 19  ELSIF THIRD_NO>FIRST_NO AND THIRD_NO>SECOND_NO THEN
 20  &D('HIGHEST NO IS(I M IN 3RD TRUE STAT) ....'||THIRD_NO);
 21  ELSIF FIRST_NO=0 AND SECOND_NO=0  AND THIRD_NO=0 THEN
 22  &D('ZERO VALUES');
 23  ELSE
 24  &D('ALL ARE EQUALS NUMBERS');
 25  END IF;
 26* END;
 27  /
Enter value for first_no: 100
Enter value for second_no: 50
Enter value for third_no: 40
1S HIGHEST NO IS ....100

2ND HIGHEST NO IS ....50

3RD HIGHEST NO IS ....40


PL/SQL procedure successfully completed.

SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL> /
```

```
Enter value for first_no: 100
Enter value for second_no: 80
Enter value for third_no: 90
1S HIGHEST NO IS ....100

2ND HIGHEST NO IS ....90

3RD HIGHEST NO IS ....80


PL/SQL procedure successfully completed.

SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL> /
Enter value for first_no: 100
Enter value for second_no: 50
Enter value for third_no: 50
1S HIGHEST NO IS ....100

2ND AND 3RD ARE EQUALS ...


PL/SQL procedure successfully completed.

SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL> ED
Wrote file afiedt.buf

  1      DECLARE
  2          --------------------DECLARATION AREA-----------------
  3          V_R_NO NUMBER :=&ROLL_NO;
  4          V_S_NAME STD.SNAME%TYPE;
  5          V_CLASS STD.CLASS_NM%TYPE;
  6          V_M_MARK STD.F_MAT%TYPE;
  7          V_S_MARK STD.F_STD%TYPE;
  8          V_P_MARK STD.F_PHY%TYPE;
  9          V_U_MARK STD.F_URD%TYPE;
 10      V_E_MARK STD.F_ENG%TYPE;
 11      TOTAL NUMBER :=0;
 12       PER NUMBER := 0;
 13        BEGIN
 14     ------------------FETCHING-----------------
 15    SELECT
 16      SNAME,
 17      CLASS_NM,
 18     F_ENG,
 19      F_PHY,
 20      F_URD,
 21     F_STD,
 22     F_MAT     INTO
```

```
 23      V_S_NAME,
 24       V_CLASS,
 25      V_E_MARK,
 26      V_P_MARK,
 27      V_U_MARK ,
 28      V_S_MARK,
 29      V_M_MARK  FROM SCOTT.STD WHERE ROLL_NO=V_R_NO;
 30      -----------------------CALCULATION AREA-------------
 31       TOTAL := V_M_MARK + V_S_MARK + V_P_MARK + V_U_MARK + V_E_MARK;
 32       PER :=  TOTAL * 100/500;
 33       -----------------------DISPLAY AREA-----------------
 34       &D(' MARKS SHEET ');
 35       &D('=============='||CHR(10));
 36       &D('ROLL NO IS ...'||V_R_NO);
 37        &D('STUDENT NAME '||V_S_NAME);
 38       &D('STUDENT CLASS '||V_CLASS);
 39       &D('MATH MARKS  '||V_M_MARK);
 40       &D('PHYSICS MARKS  '||V_P_MARK);
 41       &D('URDU MARKS  '||V_U_MARK);
 42        &D('ENGLISH MARKS  '||V_E_MARK);
 43       &D('STUDIES MARKS  '||V_S_MARK||CHR(10));
 44       &D('TOTAL MARKS .,...'||TOTAL);
 45       &D('PERCENTAGE  MARKS .,...'||PER||'%');
 46*   END;
 47  /
Enter value for roll_no: 101
MARKS SHEET

==============


ROLL NO IS ...101

STUDENT NAME ALI

STUDENT CLASS X

MATH MARKS  58

PHYSICS MARKS  58

URDU MARKS  47

ENGLISH MARKS  58

STUDIES MARKS  69

TOTAL MARKS .,...290

PERCENTAGE  MARKS .,...58%


PL/SQL procedure successfully completed.

SQL> ED
Wrote file afiedt.buf

  1      DECLARE
  2          --------------------DECLARATION AREA-----------------
  3          V_R_NO NUMBER :=&ROLL_NO;
  4          V_S_NAME STD.SNAME%TYPE;
  5          V_CLASS STD.CLASS_NM%TYPE;
  6          V_M_MARK STD.F_MAT%TYPE;
```

```
  7          V_S_MARK STD.F_STD%TYPE;
  8          V_P_MARK STD.F_PHY%TYPE;
  9          V_U_MARK STD.F_URD%TYPE;
 10       V_E_MARK STD.F_ENG%TYPE;
 11       TOTAL NUMBER :=0;
 12        PER NUMBER := 0;
 13      GRADE VARCHAR2(5);
 14         BEGIN
 15      ------------------FETCHING-----------------
 16     SELECT
 17       SNAME,
 18       CLASS_NM,
 19      F_ENG,
 20       F_PHY,
 21       F_URD,
 22      F_STD,
 23      F_MAT     INTO
 24       V_S_NAME,
 25        V_CLASS,
 26       V_E_MARK,
 27       V_P_MARK,
 28       V_U_MARK ,
 29       V_S_MARK,
 30       V_M_MARK  FROM SCOTT.STD WHERE ROLL_NO=V_R_NO;
 31       ------------------------CALCULATION AREA--------------
 32        TOTAL := V_M_MARK + V_S_MARK + V_P_MARK + V_U_MARK + V_E_MARK;
 33        PER :=  TOTAL * 100/500;
 34         IF PER >80 THEN
 35              GRADE := 'A+1';
 36           ELSIF PER BETWEEN 70 AND 79.99 THEN
 37            GRADE := 'A';
 38           ELSIF PER BETWEEN 60 AND 69.99 THEN
 39            GRADE := 'B';
 40           ELSIF PER BETWEEN 50 AND 59.99 THEN
 41            GRADE := 'C';
 42           ELSIF PER BETWEEN 40 AND 49.99 THEN
 43            GRADE := 'D';
 44           ELSE
 45              GRADE := 'FAIL';
 46            END IF;
 47        ------------------------DISPLAY AREA------------------
 48        &D(' MARKS SHEET ');
 49        &D('==============='||CHR(10));
 50        &D('ROLL NO IS ...'||V_R_NO);
 51         &D('STUDENT NAME '||V_S_NAME);
 52        &D('STUDENT CLASS '||V_CLASS);
 53        &D('MATH MARKS  '||V_M_MARK);
 54        &D('PHYSICS MARKS  '||V_P_MARK);
 55        &D('URDU MARKS  '||V_U_MARK);
 56         &D('ENGLISH MARKS  '||V_E_MARK);
 57        &D('STUDIES MARKS  '||V_S_MARK||CHR(10));
 58        &D('TOTAL MARKS .,...'||TOTAL);
 59        &D('PERCENTAGE  MARKS .,...'||PER||'%'||CHR(10));
 60         &D('GRADE IS ...'||GRADE);
 61*    END;
 62  /
Enter value for roll_no: 101
MARKS SHEET

==============


ROLL NO IS ...101
```

```
STUDENT NAME ALI

STUDENT CLASS X

MATH MARKS  58

PHYSICS MARKS  58

URDU MARKS  47

ENGLISH MARKS  58

STUDIES MARKS  69

TOTAL MARKS .,...290

PERCENTAGE  MARKS .,...58%

GRADE IS ...C


PL/SQL procedure successfully completed.

SQL>
SQL>
SQL>
SQL>
SQL> ED
Wrote file afiedt.buf

  1      DECLARE
  2          --------------------DECLARATION AREA-----------------
  3          V_R_NO NUMBER :=&ROLL_NO;
  4          V_S_NAME STD.SNAME%TYPE;
  5          V_CLASS STD.CLASS_NM%TYPE;
  6          V_M_MARK STD.F_MAT%TYPE;
  7          V_S_MARK STD.F_STD%TYPE;
  8          V_P_MARK STD.F_PHY%TYPE;
  9          V_U_MARK STD.F_URD%TYPE;
 10      V_E_MARK STD.F_ENG%TYPE;
 11      TOTAL NUMBER :=0;
 12       PER NUMBER := 0;
 13     GRADE VARCHAR2(5);
 14        BEGIN
 15     ------------------FETCHING-----------------
 16    SELECT
 17      SNAME,
 18      CLASS_NM,
 19     F_ENG,
 20      F_PHY,
 21      F_URD,
 22     F_STD,
 23     F_MAT     INTO
 24      V_S_NAME,
 25       V_CLASS,
 26      V_E_MARK,
 27      V_P_MARK,
 28      V_U_MARK ,
 29      V_S_MARK,
 30      V_M_MARK  FROM SCOTT.STD WHERE ROLL_NO=V_R_NO;
 31  IF V_E_MARK>=40 AND  V_P_MARK>= 40 AND
 32      V_U_MARK>= 40 AND V_S_MARK >=40 AND V_M_MARK>=40
```

```
 33  THEN
 34     ------------------------CALCULATION AREA--------------
 35       TOTAL := V_M_MARK + V_S_MARK + V_P_MARK + V_U_MARK + V_E_MARK;
 36       PER :=  TOTAL * 100/500;
 37        IF PER >80 THEN
 38             GRADE := 'A+1';
 39          ELSIF PER BETWEEN 70 AND 79.99 THEN
 40           GRADE := 'A';
 41          ELSIF PER BETWEEN 60 AND 69.99 THEN
 42           GRADE := 'B';
 43          ELSIF PER BETWEEN 50 AND 59.99 THEN
 44           GRADE := 'C';
 45          ELSIF PER BETWEEN 40 AND 49.99 THEN
 46           GRADE := 'D';
 47          ELSE
 48             GRADE := 'FAIL';
 49            END IF;
 50       ----------------------DISPLAY AREA------------------
 51       &D(' MARKS SHEET ');
 52       &D('==============='||CHR(10));
 53       &D('ROLL NO IS ...'||V_R_NO);
 54        &D('STUDENT NAME '||V_S_NAME);
 55       &D('STUDENT CLASS '||V_CLASS);
 56       &D('MATH MARKS  '||V_M_MARK);
 57       &D('PHYSICS MARKS  '||V_P_MARK);
 58       &D('URDU MARKS  '||V_U_MARK);
 59        &D('ENGLISH MARKS  '||V_E_MARK);
 60       &D('STUDIES MARKS  '||V_S_MARK||CHR(10));
 61       &D('TOTAL MARKS .,...'||TOTAL);
 62       &D('PERCENTAGE  MARKS .,...'||PER||'%'||CHR(10));
 63        &D('GRADE IS ...'||GRADE);
 64  ELSE
 65  &D('STUDENT IS FAIL');
 66  END IF;
 67*   END;
 68  /
Enter value for roll_no: 101
MARKS SHEET

==============


ROLL NO IS ...101

STUDENT NAME ALI

STUDENT CLASS X

MATH MARKS  58

PHYSICS MARKS  58

URDU MARKS  47

ENGLISH MARKS  58

STUDIES MARKS  69

TOTAL MARKS .,...290

PERCENTAGE  MARKS .,...58%

GRADE IS ...C
```

```
PL/SQL procedure successfully completed.

SQL> ED
Wrote file afiedt.buf

  1  DECLARE
  2  OUTER NUMBER :=10;
  3  BEGIN
  4  &d('VALUE OF OUTER IS IN OUTER BLOCK' ||OUTER);
  5* END;
  6  /
VALUE OF OUTER IS IN OUTER BLOCK10


PL/SQL procedure successfully completed.

SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL> ED
Wrote file afiedt.buf

  1  DECLARE
  2  OUTER NUMBER :=10;
  3  BEGIN
  4  &d('VALUE OF OUTER IS IN OUTER BLOCK...' ||OUTER);
  5  ----------------------------NESTED BLOCK-----------------
  6     DECLARE
  7     INNER NUMBER :=30;
  8             BEGIN
  9     &D(VALUE OF INNER IN INNER BLOCK...'||INNER);
 10     END;
 11  -------------------------END OF NESTED BLOCK----------
 12* END;
 13  ED
 14  .
SQL> ED
Wrote file afiedt.buf

  1  DECLARE
  2  OUTER NUMBER :=10;
  3  BEGIN
  4  &d('VALUE OF OUTER IS IN OUTER BLOCK...' ||OUTER);
  5  ----------------------------NESTED BLOCK-----------------
  6     DECLARE
  7     INNER NUMBER :=30;
  8             BEGIN
  9     &D('VALUE OF INNER IN INNER BLOCK...'||INNER);
 10     END;
 11  -------------------------END OF NESTED BLOCK----------
 12* END;
 13  /
VALUE OF OUTER IS IN OUTER BLOCK...10

VALUE OF INNER IN INNER BLOCK...30
```

```
PL/SQL procedure successfully completed.

SQL>
SQL>
SQL>
SQL>
SQL>
SQL> ED
Wrote file afiedt.buf

  1  DECLARE
  2  OUTER NUMBER :=10;
  3  BEGIN
  4  &d('VALUE OF OUTER IS IN OUTER BLOCK...' ||OUTER);
  5  ----------------------------NESTED BLOCK-----------------
  6     DECLARE
  7     INNER NUMBER :=30;
  8             BEGIN
  9         &d('VALUE OF OUTER IN INNER BLOCK...' ||OUTER);
 10     &D('VALUE OF INNER IN INNER BLOCK...'||INNER);
 11     END;
 12  -------------------------END OF NESTED BLOCK----------
 13* END;
SQL> /
VALUE OF OUTER IS IN OUTER BLOCK...10

VALUE OF OUTER IN INNER BLOCK...10

VALUE OF INNER IN INNER BLOCK...30


PL/SQL procedure successfully completed.

SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL> ED
Wrote file afiedt.buf

  1  DECLARE
  2  OUTER NUMBER :=10;
  3  BEGIN
  4  &d('VALUE OF OUTER IS IN OUTER BLOCK...' ||OUTER);
  5  ----------------------------NESTED BLOCK-----------------
  6     DECLARE
  7     INNER NUMBER :=30;
  8             BEGIN
  9         &d('VALUE OF OUTER IN INNER BLOCK...' ||OUTER);
 10     &D('VALUE OF INNER IN INNER BLOCK...'||INNER);
 11     END;
 12  -------------------------END OF NESTED BLOCK----------
 13         &d('VALUE OF OUTER AFTER INNER BLOCK...' ||OUTER);
 14* END;
 15  /
VALUE OF OUTER IS IN OUTER BLOCK...10
```

```
VALUE OF OUTER IN INNER BLOCK...10

VALUE OF INNER IN INNER BLOCK...30

VALUE OF OUTER AFTER INNER BLOCK...10


PL/SQL procedure successfully completed.

SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL> ED
Wrote file afiedt.buf

  1  DECLARE
  2  OUTER NUMBER :=10;
  3  BEGIN
  4  &d('VALUE OF OUTER IS IN OUTER BLOCK...' ||OUTER);
  5  ----------------------------NESTED BLOCK-----------------
  6     DECLARE
  7     INNER NUMBER :=30;
  8             BEGIN
  9         &d('VALUE OF OUTER IN INNER BLOCK...' ||OUTER);
 10     &D('VALUE OF INNER IN INNER BLOCK...'||INNER);
 11     END;
 12  -------------------------END OF NESTED BLOCK----------
 13         &d('VALUE OF OUTER AFTER INNER BLOCK...' ||OUTER);
 14     &D('VALUE OF INNER AFTER INNER BLOCK...'||INNER);
 15* END;
SQL> /
        DBMS_OUTPUT.PUT_LINE('VALUE OF INNER AFTER INNER BLOCK...'||INNER);
                                                                  *
ERROR at line 14:
ORA-06550: line 14, column 62:
PLS-00201: identifier 'INNER' must be declared
ORA-06550: line 14, column 2:
PL/SQL: Statement ignored


SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL> ED
Wrote file afiedt.buf

  1  DECLARE
  2  OUTER NUMBER :=10;
  3  BEGIN
  4  &d('VALUE OF OUTER IS IN OUTER BLOCK...' ||OUTER);
  5  ----------------------------NESTED BLOCK-----------------
  6     DECLARE
  7     INNER NUMBER :=30;
  8             BEGIN
  9         &d('VALUE OF OUTER IN INNER BLOCK...' ||OUTER);
 10     &D('VALUE OF INNER IN INNER BLOCK...'||INNER);
 11     END;
```

```
 12  -------------------------END OF NESTED BLOCK----------
 13         &d('VALUE OF OUTER AFTER INNER BLOCK...' ||OUTER);
 14     &D('VALUE OF INNER AFTER INNER BLOCK...'||INNER);
 15  ----------------------------NESTED BLOCK 2 -----------------
 16     DECLARE
 17     INNER2 NUMBER :=30;
 18             BEGIN
 19         &d('VALUE OF OUTER IN INNER2 BLOCK...' ||OUTER);
 20     &D('VALUE OF INNER IN INNER2 BLOCK...'||INNER2);
 21     END;
 22  -------------------------END OF NESTED BLOCK----------
 23* END;
 24  /
        DBMS_OUTPUT.PUT_LINE('VALUE OF INNER AFTER INNER BLOCK...'||INNER);
                                                                  *
ERROR at line 14:
ORA-06550: line 14, column 62:
PLS-00201: identifier 'INNER' must be declared
ORA-06550: line 14, column 2:
PL/SQL: Statement ignored


SQL> ED
Wrote file afiedt.buf

  1  DECLARE
  2  OUTER NUMBER :=10;
  3  BEGIN
  4  &d('VALUE OF OUTER IS IN OUTER BLOCK...' ||OUTER);
  5  ----------------------------NESTED BLOCK-----------------
  6     DECLARE
  7     INNER NUMBER :=30;
  8             BEGIN
  9         &d('VALUE OF OUTER IN INNER BLOCK...' ||OUTER);
 10     &D('VALUE OF INNER IN INNER BLOCK...'||INNER);
 11     END;
 12  -------------------------END OF NESTED BLOCK----------
 13         &d('VALUE OF OUTER AFTER INNER BLOCK...' ||OUTER);
 14     ---('VALUE OF INNER AFTER INNER BLOCK...'||INNER);
 15  ----------------------------NESTED BLOCK 2 -----------------
 16     DECLARE
 17     INNER2 NUMBER :=30;
 18             BEGIN
 19         &d('VALUE OF OUTER IN INNER2 BLOCK...' ||OUTER);
 20     &D('VALUE OF INNER IN INNER2 BLOCK...'||INNER2);
 21     END;
 22  -------------------------END OF NESTED BLOCK----------
 23* END;
SQL> /
VALUE OF OUTER IS IN OUTER BLOCK...10

VALUE OF OUTER IN INNER BLOCK...10

VALUE OF INNER IN INNER BLOCK...30

VALUE OF OUTER AFTER INNER BLOCK...10

VALUE OF OUTER IN INNER2 BLOCK...10

VALUE OF INNER IN INNER2 BLOCK...30


PL/SQL procedure successfully completed.
```

```
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL> ED
Wrote file afiedt.buf

  1  DECLARE
  2  V_EMPNO NUMBER :=&EMPNO;
  3  V_ENAME EMP.ENAME%TYPE:='&ENAME';
  4  V_JOB   EMP.JOB%TYPE:='&JOB';
  5  V_SAL   EMP.SAL%TYPE:=&SALARY;
  6  V_DEPTNO EMP.DEPTNO%TYPE:=&DEPTNO;
  7  BEGIN
  8  INSERT INTO EMP(EMPNO,ENAME,JOB,SAL,DEPTNO)
  9  VALUES(V_EMPNO,V_ENAME,V_JOB,V_SAL,V_DEPTNO);
 10  COMMIT;
 11  &D('RECORD CREATED WITH EMPNO ..'||V_EMPNO);
 12* END;
 13  /
Enter value for empno: 500
Enter value for ename: ALI
Enter value for job: SALESMAN
Enter value for salary: 1000
Enter value for deptno: 30
RECORD CREATED WITH EMPNO ..500


PL/SQL procedure successfully completed.

SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL> SELECT * FROM EMP
  2  WHERE EMPNO=500;

     EMPNO ENAME      JOB              MGR HIREDATE          SAL        COMM
DEPTNO
---------- ---------- --------- ---------- --------- ---------- ----------
----------
       500 ALI        SALESMAN                              1000
30

SQL>
SQL>
SQL>
SQL>
SQL> DECLARE
  2  V_EMPNO NUMBER :=&EMPNO;
  3  V_ENAME EMP.ENAME%TYPE:='&ENAME';
  4  V_JOB   EMP.JOB%TYPE:='&JOB';
  5  V_SAL   EMP.SAL%TYPE:=&SALARY;
  6  V_DEPTNO EMP.DEPTNO%TYPE:=&DEPTNO;
  7  BEGIN
  8  INSERT INTO EMP(EMPNO,ENAME,JOB,SAL,DEPTNO)
  9  VALUES(V_EMPNO,V_ENAME,V_JOB,V_SAL,V_DEPTNO);
 10  COMMIT;
```

```
 11  &D('RECORD CREATED WITH EMPNO ..'||V_EMPNO);
 12  END;
 13  .
SQL> ED
Wrote file afiedt.buf

  1  DECLARE
  2  V_EMPNO NUMBER :=&EMPNO;
  3  V_ENAME EMP.ENAME%TYPE:='&ENAME';
  4  V_JOB   EMP.JOB%TYPE:='&JOB';
  5  V_SAL   EMP.SAL%TYPE:=&SALARY;
  6  V_DEPTNO EMP.DEPTNO%TYPE:=&DEPTNO;
  7  V_COMM NUMBER :=0;
  8  BEGIN
  9  IF V_JOB='SALESMAN' THEN
 10  V_COMM := V_SAL * 10/100;
 11  END IF;
 12  INSERT INTO EMP(EMPNO,ENAME,JOB,SAL,DEPTNO,COMM)
 13  VALUES(V_EMPNO,V_ENAME,V_JOB,V_SAL,V_DEPTNO,V_COMM);
 14  COMMIT;
 15  &D('RECORD CREATED WITH EMPNO ..'||V_EMPNO);
 16* END;
 17  /
Enter value for empno: 501
Enter value for ename: ALI
Enter value for job: SALESMAN
Enter value for salary: 1000
Enter value for deptno: 30
RECORD CREATED WITH EMPNO ..501


PL/SQL procedure successfully completed.

SQL>
SQL>
SQL>
SQL> SELECT * FROM EMP
  2  WHERE EMPNO=501
  3  ;

     EMPNO ENAME      JOB              MGR HIREDATE          SAL        COMM
DEPTNO
---------- ---------- --------- ---------- --------- ---------- ----------
----------
       501 ALI        SALESMAN                            1000         100
30

SQL>
SQL>
SQL>
SQL> /

     EMPNO ENAME      JOB              MGR HIREDATE          SAL        COMM
DEPTNO
---------- ---------- --------- ---------- --------- ---------- ----------
----------
       501 ALI        SALESMAN                            1000         100
30

SQL> DECLARE
  2  V_EMPNO NUMBER :=&EMPNO;
  3  V_ENAME EMP.ENAME%TYPE:='&ENAME';
  4  V_JOB   EMP.JOB%TYPE:='&JOB';
```

```
  5  V_SAL   EMP.SAL%TYPE:=&SALARY;
  6  V_DEPTNO EMP.DEPTNO%TYPE:=&DEPTNO;
  7  V_COMM NUMBER :=0;
  8  BEGIN
  9  IF V_JOB='SALESMAN' THEN
 10  V_COMM := V_SAL * 10/100;
 11  END IF;
 12  INSERT INTO EMP(EMPNO,ENAME,JOB,SAL,DEPTNO,COMM)
 13  VALUES(V_EMPNO,V_ENAME,V_JOB,V_SAL,V_DEPTNO,V_COMM);
 14  COMMIT;
 15  &D('RECORD CREATED WITH EMPNO ..'||V_EMPNO);
 16  END;
 17  /
Enter value for empno: 502
Enter value for ename: ALI
Enter value for job: MANAGER
Enter value for salary: 4000
Enter value for deptno: 20
RECORD CREATED WITH EMPNO ..502


PL/SQL procedure successfully completed.

SQL> SELECT * FROM EMP
  2  WHERE EMPNO=502
  3
SQL> /

     EMPNO ENAME      JOB               MGR HIREDATE          SAL        COMM
DEPTNO
---------- ---------- --------- ---------- --------- ---------- ----------
----------
       502 ALI        MANAGER                                4000           0
20

SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL> ED
Wrote file afiedt.buf

SP2-0223: No lines in SQL buffer.
SQL>
SQL> SELECT * FROM CAT;

TABLE_NAME                     TABLE_TYPE

------------------------------ -----------

DEPT                           TABLE

EMP                            TABLE

BONUS                          TABLE

SALGRADE                       TABLE
```

```
EMP_TEST                       TABLE

EMP_HIST                       TABLE

EMP_EXCEPTION                  TABLE

TEST                           TABLE

LOG_EMP_HIST                   TABLE

EMP_VIEW                       VIEW

EMP_TEMP                       TABLE

VU_SAL                         VIEW

VU_MGR                         VIEW

EIMAGE                         TABLE

EMP_RESUME                     TABLE

EMP_INFO                       VIEW

BIN$6oYvYDsOSLqezFJYs/7haA==$0 TABLE

EMP_AUDIT                      TABLE

S1                             SEQUENCE

EMP_COPY                       TABLE

JOB_IDS                        TABLE

STD                            TABLE

TEST1                          TABLE


23 rows selected.

SQL> SELECT * FROM EMP_TEST
  2  ;'
  3  /
ERROR:
ORA-01756: quoted string not properly terminated


SQL>
SQL>
SQL> SELECT * FROM EMP_TEST;

     EMPNO ENAME      JOB              MGR HIREDATE          SAL        COMM
DEPTNO I POST_DATE
---------- ---------- --------- ---------- --------- ---------- ----------
---------- - ---------
REV_DATE  ADDRESS

--------- --------------------

      2002 SCOTT      SALESMAN           2                 1000        100
30
```

30	2003	SCOTT	SALESMAN	7788		1000	100
30	2004	SCOTT	SALESM	7788		1000	50
30	2005	SCOTT	SALESMAN	7788		1000	100
20	7369	SMITH	CLERK	7902	17-DEC-80	800	
30	7499	ALLEN	SALESMAN	7698	20-FEB-81	1600	300
30	7521	WARD	SALESMAN	7698	22-FEB-81	1250	500
20	7566	JONES	MANAGER	7839	02-APR-81	2975	
30	7654	MARTIN	SALESMAN	7698	28-SEP-81	1250	1400
30	7698	BLAKE	MANAGER	7839	01-MAY-81	2850	
10	7782	CLARK	MANAGER	7839	09-JUN-81	2450	

```
      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

      3000 SCOTT      SALESMAN                             1000
30 Y 10-DEC-12
10-DEC-12


19 rows selected.

SQL> SELECT DEPTNO,SAL FROM EMP_TEST;

    DEPTNO        SAL

---------- ----------

        30       1000

        30       1000

        30       1000
```

```
        30       1000

        20        800

        30       1600

        30       1250

        20       2975

        30       1250

        30       2850

        10       2450

        20       3000

        10       5000

        30       1500

        20       1100

        30        950

        20       3000

        10       1300

        30       1000

19 rows selected.

SQL>
SQL>
SQL>
SQL>
SQL> ED
Wrote file afiedt.buf

  1  DECLARE
  2  V_DEPTNO NUMBER :=&DEPTNO;
  3  BEGIN
  4     UPDATE EMP_TEST
  5     SET SAL=SAL +100
  6     WHERE DEPTNO=V_DEPTNO;
  7        &D('RECORD UPDATED...');
  8* END;
  9  /
Enter value for deptno: 30
RECORD UPDATED...


PL/SQL procedure successfully completed.

SQL> SELECT DEPTNO,SAL FROM EMP_TEST;

    DEPTNO        SAL

---------- ----------
```

```
        30       1100
        30       1100
        30       1100
        30       1100
        20        800
        30       1700
        30       1350
        20       2975
        30       1350
        30       2950
        10       2450
        20       3000
        10       5000
        30       1600
        20       1100
        30       1050
        20       3000
        10       1300
        30       1100

19 rows selected.

SQL> /

    DEPTNO        SAL
---------- ----------
        30       1100
        30       1100
        30       1100
        30       1100
        20        800
        30       1700
        30       1350
        20       2975
```

```
        30       1350
        30       2950
        10       2450
        20       3000
        10       5000
        30       1600
        20       1100
        30       1050
        20       3000
        10       1300
        30       1100

19 rows selected.
SQL> ROLLBACK;
Rollback complete.
SQL> SELECT DEPTNO,SAL FROM EMP_TEST;
    DEPTNO        SAL
---------- ----------
        30       1000
        30       1000
        30       1000
        30       1000
        20        800
        30       1600
        30       1250
        20       2975
        30       1250
        30       2850
        10       2450
        20       3000
        10       5000
        30       1500
```

```
        20       1100

        30        950

        20       3000

        10       1300

        30       1000


19 rows selected.

SQL>  DECLARE
  2   V_DEPTNO NUMBER :=&DEPTNO;
  3   BEGIN
  4      UPDATE EMP_TEST
  5      SET SAL=SAL +100
  6      WHERE DEPTNO=V_DEPTNO;
  7         &D('RECORD UPDATED...');
  8   END;
  9  /
Enter value for deptno: 55
RECORD UPDATED...


PL/SQL procedure successfully completed.

SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL> ED
Wrote file afiedt.buf

  1   DECLARE
  2   V_DEPTNO NUMBER :=&DEPTNO;
  3   BEGIN
  4      UPDATE EMP_TEST
  5      SET SAL=SAL +100
  6      WHERE DEPTNO=V_DEPTNO;
  7         IF SQL%FOUND THEN
  8         &D('RECORD UPDATED...');
  9         END IF;
 10*  END;
 11  /
Enter value for deptno: 30
RECORD UPDATED...


PL/SQL procedure successfully completed.

SQL> /
Enter value for deptno: 55

PL/SQL procedure successfully completed.

SQL>
SQL>
SQL> ED
Wrote file afiedt.buf
```

```
  1   DECLARE
  2   V_DEPTNO NUMBER :=&DEPTNO;
  3   BEGIN
  4      UPDATE EMP_TEST
  5      SET SAL=SAL +100
  6      WHERE DEPTNO=V_DEPTNO;
  7         IF SQL%FOUND THEN
  8         &D('RECORD UPDATED...');
  9        ELSE
 10       &D('INVALID DEPTNO...');
 11         END IF;
 12*  END;
 13  /
Enter value for deptno: 55
INVALID DEPTNO...


PL/SQL procedure successfully completed.

SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL> ED
Wrote file afiedt.buf

  1   DECLARE
  2   V_DEPTNO NUMBER :=&DEPTNO;
  3   BEGIN
  4      UPDATE EMP_TEST
  5      SET SAL=SAL +100
  6      WHERE DEPTNO=V_DEPTNO;
  7         IF SQL%FOUND THEN
  8         &D('RECORD UPDATED...');
  9        ELSIF SQL%NOTFOUND THEN
 10       &D('INVALID DEPTNO...');
 11         END IF;
 12*  END;
SQL> /
Enter value for deptno: 30
RECORD UPDATED...


PL/SQL procedure successfully completed.

SQL> /
Enter value for deptno: 55
INVALID DEPTNO...


PL/SQL procedure successfully completed.

SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
```

```
SQL> ED
Wrote file afiedt.buf

  1   DECLARE
  2   V_DEPTNO NUMBER :=&DEPTNO;
  3   BEGIN
  4      UPDATE EMP_TEST
  5      SET SAL=SAL +100
  6      WHERE DEPTNO=V_DEPTNO;
  7         IF SQL%FOUND THEN
  8         &D(SQL%ROWCOUNT||'  RECORD UPDATED...');
  9        ELSIF SQL%NOTFOUND THEN
 10       &D('INVALID DEPTNO...');
 11         END IF;
 12*  END;
SQL> /
Enter value for deptno: 30
11  RECORD UPDATED...


PL/SQL procedure successfully completed.

SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL> ED
Wrote file afiedt.buf

  1   DECLARE
  2   V_EMPNO NUMBER :=&EMPNO;
  3   EMP_REC EMP%ROWTYPE;
  4   BEGIN
  5    SELECT * INTO EMP_REC FROM SCOTT.EMP
  6     WHERE EMPNO=V_EMPNO;
  7    IF SQL%FOUND THEN
  8        &D('ENAME IS ...'||EMP_REC.ENAME);
  9        &D('JOB IS ...'||EMP_REC.JOB);
 10        &D('SAL IS ...'||EMP_REC.SAL);
 11         END IF;
 12*  END;
 13  /
Enter value for empno: 7788
ENAME IS ...SCOTT

JOB IS ...ANALYST

SAL IS ...3000


PL/SQL procedure successfully completed.

SQL>
SQL>
SQL>
SQL>
SQL>
```

```
SQL>
SQL>
SQL> ED
Wrote file afiedt.buf

  1   DECLARE
  2   V_EMPNO NUMBER :=&EMPNO;
  3   EMP_REC EMP%ROWTYPE;
  4   BEGIN
  5    SELECT * INTO EMP_REC FROM SCOTT.EMP
  6     WHERE EMPNO=V_EMPNO;
  7    IF SQL%FOUND THEN
  8        &D('ENAME IS ...'||EMP_REC.ENAME);
  9        &D('JOB IS ...'||EMP_REC.JOB);
 10        &D('SAL IS ...'||EMP_REC.SAL);
 11      ELSIF SQL%NOTFOUND THEN
 12     &('RECORD NOT EXIST..');
 13         END IF;
 14*  END;
 15  /
Enter value for empno:
 V_EMPNO NUMBER :=;
                 *
ERROR at line 2:
ORA-06550: line 2, column 19:
PLS-00103: Encountered the symbol ";" when expecting one of the following:
( - + case mod new not null <an identifier>
<a double-quoted delimited-identifier> <a bind variable> avg
count current exists max min prior sql stddev sum variance
execute forall merge time timestamp interval date
<a string literal with character set specification>
<a number> <a single-quoted SQL string> pipe
<an alternatively-quoted string literal with character set specification>
<an alternatively-quoted S
ORA-06550: line 12, column 4:
PLS-00103: Encountered the symbol "&" when expecting one of the following:
begin case declare exit for goto if loop mod null pragma
raise return select update while with <an identifier>
<a double-quoted delimited-identifier> <a


SQL>
SQL> ED
Wrote file afiedt.buf

  1   DECLARE
  2   V_EMPNO NUMBER :=&EMPNO;
  3   EMP_REC EMP%ROWTYPE;
  4   BEGIN
  5    SELECT * INTO EMP_REC FROM SCOTT.EMP
  6     WHERE EMPNO=V_EMPNO;
  7    IF SQL%FOUND THEN
  8        &D('ENAME IS ...'||EMP_REC.ENAME);
  9        &D('JOB IS ...'||EMP_REC.JOB);
 10        &D('SAL IS ...'||EMP_REC.SAL);
 11      ELSIF SQL%NOTFOUND THEN
 12     &D('RECORD NOT EXIST..');
 13         END IF;
 14*  END;
SQL> /
Enter value for empno: 7788
ENAME IS ...SCOTT
```

```
JOB IS ...ANALYST

SAL IS ...3000


PL/SQL procedure successfully completed.

SQL>
SQL>
SQL>
SQL> /
Enter value for empno: 524
 DECLARE
*
ERROR at line 1:
ORA-01403: no data found
ORA-06512: at line 5


SQL>
SQL>
SQL>
SQL>
SQL>
SQL> SELECT * FROM EMP
  2  WHERE EMPNO=45
  3  ;

no rows selected

SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL> SELECT COUNT(*) FROM EMP WHERE DEPTNO=&DEPTNO;
Enter value for deptno: 10

  COUNT(*)

----------

         3


SQL>
SQL>
SQL>
SQL> /
Enter value for deptno: 30

  COUNT(*)

----------

         8


SQL> /
Enter value for deptno: 54
```

```
  COUNT(*)

----------

         0


SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL> ED
Wrote file afiedt.buf

  1* SELECT COUNT(*) FROM EMP WHERE DEPTNO=&DEPTNO
SQL>
SQL>
SQL>
SQL>
SQL> SPOOL OFF
```

```
SQL>
SQL> DECLARE
  2  .
SQL> ED
Wrote file afiedt.buf

  1  DECLARE
  2  V_EMPNO EMP.EMPNO%TYPE:=&EMPNO;
  3  EMP_REC EMP%ROWTYPE;
  4  TOTAL_SAL NUMBER :=0;
  5  V_TAX NUMBER :=0;
  6  V_DED NUMBER :=0;
  7  V_ANN_SAL NUMBER :=0;
  8  BEGIN
  9  SELECT * INTO EMP_REC FROM EMP
 10  WHERE EMPNO=V_EMPNO;
 11  &D(EMP_REC.SAL);
 12* END;
 13  /
Enter value for empno: 7788
3000


PL/SQL procedure successfully completed.

SQL> ED
Wrote file afiedt.buf

  1  DECLARE
  2  V_EMPNO EMP.EMPNO%TYPE:=&EMPNO;
  3  EMP_REC EMP%ROWTYPE;
  4  TOTAL_SAL NUMBER :=0;
  5  V_TAX NUMBER :=0;
  6  V_DED NUMBER :=100;
  7  V_ANN_SAL NUMBER :=0;
  8  V_BONUS NUMBER :=100;
  9  BEGIN
 10  SELECT * INTO EMP_REC FROM EMP
 11  WHERE EMPNO=V_EMPNO;
 12  V_ANN_SAL := EMP_REC.SAL * 12;
 13  IF V_ANN_SAL BETWEEN 10000 AND 12000 THEN
 14  V_TAX := EMP_REC.SAL * 2/100;
 15  ELSIF V_ANN_SAL BETWEEN 12001 AND 14000 THEN
 16  V_TAX := EMP_REC.SAL * 3/100;
 17  ELSIF V_ANN_SAL BETWEEN 14001 AND 16000 THEN
 18  V_TAX := EMP_REC.SAL * 5/100;
 19  ELSIF V_ANN_SAL >16000 THEN
 20  V_TAX := EMP_REC.SAL * 10/100;
 21  END IF;
 22  TOTAL_SAL := (NVL(EMP_REC.SAL,0) + NVL(EMP_REC.COMM,0) + NVL(V_BONUS,0))
 23               -
 24                (V_TAX + V_DED) ;
 25  -----------------DISPLAY---------------
 26  &D('ENAME IS ....'||EMP_REC.ENAME);
 27  &D('DESIGNATION IS ....'||EMP_REC.JOB);
 28  &D('SALARY  IS ....'||EMP_REC.SAL);
 29  &D('COMMISSION IS ....'||EMP_REC.COMM);
 30  &D('DEPTARTMENT IS ....'||EMP_REC.DEPTNO);
 31  &D('TAX IS ....'||V_TAX);
 32  &D('BONUS  IS ....'||V_BONUS);
 33  &D('DEDUCTION IS ....'||V_DED);
 34  &D('NET SALARY IS ....'||TOTAL_SAL);
 35* END;
```

```
 36  /
Enter value for empno: 7788
ENAME IS ....SCOTT

DESIGNATION IS ....ANALYST

SALARY  IS ....3000

COMMISSION IS ....

DEPTARTMENT IS ....20

TAX IS ....300

BONUS  IS ....100

DEDUCTION IS ....100

NET SALARY IS ....2700


PL/SQL procedure successfully completed.

SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL> ED
Wrote file afiedt.buf

  1  DECLARE
  2  V_EMPNO EMP.EMPNO%TYPE:=&EMPNO;
  3  EMP_REC EMP%ROWTYPE;
  4  TOTAL_SAL NUMBER :=0;
  5  V_TAX NUMBER :=0;
  6  V_TAX_RATE NUMBER :=0;
  7  V_DED NUMBER :=100;
  8  V_ANN_SAL NUMBER :=0;
  9  V_BONUS NUMBER :=100;
 10  BEGIN
 11  SELECT * INTO EMP_REC FROM EMP
 12  WHERE EMPNO=V_EMPNO;
 13  V_ANN_SAL := EMP_REC.SAL * 12;
 14  IF V_ANN_SAL BETWEEN 10000 AND 12000 THEN
 15  V_TAX_RATE := 2/100;
 16  ELSIF V_ANN_SAL BETWEEN 12001 AND 14000 THEN
 17  V_TAX_RATE := 3/100;
 18  ELSIF V_ANN_SAL BETWEEN 14001 AND 16000 THEN
 19  V_TAX_RATE := 5/100;
 20  ELSIF V_ANN_SAL >16000 THEN
 21  V_TAX_RATE := 10/100;
 22  END IF;
 23  V_TAX := EMP_REC.SAL * V_TAX_RATE;
 24  TOTAL_SAL := (NVL(EMP_REC.SAL,0) + NVL(EMP_REC.COMM,0) + NVL(V_BONUS,0))
 25               -
 26                (V_TAX + V_DED) ;
 27  -----------------DISPLAY---------------
 28  &D('ENAME IS ....'||EMP_REC.ENAME);
 29  &D('DESIGNATION IS ....'||EMP_REC.JOB);
 30  &D('SALARY  IS ....'||EMP_REC.SAL);
 31  &D('COMMISSION IS ....'||EMP_REC.COMM);
```

```
 32  &D('DEPTARTMENT IS ....'||EMP_REC.DEPTNO);
 33  &D('TAX RATE IS .......'||V_TAX_RATE);
 34  &D('TAX IS ....'||V_TAX);
 35  &D('BONUS  IS ....'||V_BONUS);
 36  &D('DEDUCTION IS ....'||V_DED);
 37  &D('NET SALARY IS ....'||TOTAL_SAL);
 38* END;
 39  /
Enter value for empno: 7788
ENAME IS ....SCOTT

DESIGNATION IS ....ANALYST

SALARY  IS ....3000

COMMISSION IS ....

DEPTARTMENT IS ....20

TAX RATE IS .......1

TAX IS ....300

BONUS  IS ....100

DEDUCTION IS ....100

NET SALARY IS ....2700


PL/SQL procedure successfully completed.

SQL>
SQL>
SQL>
SQL> ED
Wrote file afiedt.buf

  1  DECLARE
  2  V_EMPNO EMP.EMPNO%TYPE:=&EMPNO;
  3  EMP_REC EMP%ROWTYPE;
  4  TOTAL_SAL NUMBER :=0;
  5  V_TAX NUMBER :=0;
  6  V_TAX_RATE NUMBER :=0;
  7  V_DED NUMBER :=100;
  8  V_ANN_SAL NUMBER :=0;
  9  V_BONUS NUMBER :=100;
 10  BEGIN
 11  SELECT * INTO EMP_REC FROM EMP
 12  WHERE EMPNO=V_EMPNO;
 13  V_ANN_SAL := EMP_REC.SAL * 12;
 14  IF V_ANN_SAL BETWEEN 10000 AND 12000 THEN
 15  V_TAX_RATE := 2/100;
 16  ELSIF V_ANN_SAL BETWEEN 12001 AND 14000 THEN
 17  V_TAX_RATE := 3/100;
 18  ELSIF V_ANN_SAL BETWEEN 14001 AND 16000 THEN
 19  V_TAX_RATE := 5/100;
 20  ELSIF V_ANN_SAL >16000 THEN
 21  V_TAX_RATE := 10/100;
 22  END IF;
 23  V_TAX := EMP_REC.SAL * V_TAX_RATE;
 24  TOTAL_SAL := (NVL(EMP_REC.SAL,0) + NVL(EMP_REC.COMM,0) + NVL(V_BONUS,0))
 25               -
```

```
 26                (V_TAX + V_DED) ;
 27  -----------------DISPLAY---------------
 28  &D('ENAME IS ....'||EMP_REC.ENAME);
 29  &D('DESIGNATION IS ....'||EMP_REC.JOB);
 30  &D('SALARY  IS ....'||EMP_REC.SAL);
 31  &D('COMMISSION IS ....'||EMP_REC.COMM);
 32  &D('DEPTARTMENT IS ....'||EMP_REC.DEPTNO);
 33  &D('TAX RATE IS  = '||V_TAX_RATE);
 34  &D('TAX IS ....'||V_TAX);
 35  &D('BONUS  IS ....'||V_BONUS);
 36  &D('DEDUCTION IS ....'||V_DED);
 37  &D('NET SALARY IS ....'||TOTAL_SAL);
 38* END;
SQL> /
Enter value for empno: 7788
ENAME IS ....SCOTT

DESIGNATION IS ....ANALYST

SALARY  IS ....3000

COMMISSION IS ....

DEPTARTMENT IS ....20

TAX RATE IS  = .1

TAX IS ....300

BONUS  IS ....100

DEDUCTION IS ....100

NET SALARY IS ....2700


PL/SQL procedure successfully completed.

SQL>
SQL>
SQL>
SQL>
SQL> ED
Wrote file afiedt.buf

  1  DECLARE
  2  V_EMPNO EMP.EMPNO%TYPE:=&EMPNO;
  3  EMP_REC EMP%ROWTYPE;
  4  TOTAL_SAL NUMBER :=0;
  5  V_TAX NUMBER :=0;
  6  V_TAX_RATE NUMBER :=0;
  7  V_DED NUMBER :=100;
  8  V_ANN_SAL NUMBER :=0;
  9  V_BONUS NUMBER :=100;
 10  BEGIN
 11  SELECT * INTO EMP_REC FROM EMP
 12  WHERE EMPNO=V_EMPNO;
 13  V_ANN_SAL := EMP_REC.SAL * 12;
 14  IF V_ANN_SAL BETWEEN 10000 AND 12000 THEN
 15  V_TAX_RATE := 2;
 16  ELSIF V_ANN_SAL BETWEEN 12001 AND 14000 THEN
 17  V_TAX_RATE := 3;
 18  ELSIF V_ANN_SAL BETWEEN 14001 AND 16000 THEN
```

```
 19  V_TAX_RATE := 5;
 20  ELSIF V_ANN_SAL >16000 THEN
 21  V_TAX_RATE := 10;
 22  END IF;
 23  V_TAX := EMP_REC.SAL * V_TAX_RATE/100;
 24  TOTAL_SAL := (NVL(EMP_REC.SAL,0) + NVL(EMP_REC.COMM,0) + NVL(V_BONUS,0))
 25               -
 26                (V_TAX + V_DED) ;
 27  -----------------DISPLAY---------------
 28  &D('ENAME IS ....'||EMP_REC.ENAME);
 29  &D('DESIGNATION IS ....'||EMP_REC.JOB);
 30  &D('SALARY  IS ....'||EMP_REC.SAL);
 31  &D('COMMISSION IS ....'||EMP_REC.COMM);
 32  &D('DEPTARTMENT IS ....'||EMP_REC.DEPTNO);
 33  &D('TAX RATE IS  = '||V_TAX_RATE);
 34  &D('TAX IS ....'||V_TAX);
 35  &D('BONUS  IS ....'||V_BONUS);
 36  &D('DEDUCTION IS ....'||V_DED);
 37  &D('NET SALARY IS ....'||TOTAL_SAL);
 38* END;
SQL>
SQL> /
Enter value for empno: 7788
ENAME IS ....SCOTT

DESIGNATION IS ....ANALYST

SALARY  IS ....3000

COMMISSION IS ....

DEPTARTMENT IS ....20

TAX RATE IS  = 10

TAX IS ....300

BONUS  IS ....100

DEDUCTION IS ....100

NET SALARY IS ....2700


PL/SQL procedure successfully completed.

SQL>
SQL>
SQL>
SQL>
SQL>
SQL> ED
Wrote file afiedt.buf

  1  DECLARE
  2  V_EMPNO EMP.EMPNO%TYPE:=&EMPNO;
  3  EMP_REC EMP%ROWTYPE;
  4  TOTAL_SAL NUMBER :=0;
  5  V_TAX NUMBER :=0;
  6  V_TAX_RATE NUMBER :=0;
  7  V_DED NUMBER :=100;
  8  V_ANN_SAL NUMBER :=0;
  9  V_BONUS NUMBER :=100;
```

```
 10  BEGIN
 11  SELECT * INTO EMP_REC FROM EMP
 12  WHERE EMPNO=V_EMPNO;
 13  V_ANN_SAL := EMP_REC.SAL * 12;
 14  IF V_ANN_SAL BETWEEN 10000 AND 12000 THEN
 15  V_TAX_RATE := 2;
 16  ELSIF V_ANN_SAL BETWEEN 12001 AND 14000 THEN
 17  V_TAX_RATE := 3;
 18  ELSIF V_ANN_SAL BETWEEN 14001 AND 16000 THEN
 19  V_TAX_RATE := 5;
 20  ELSIF V_ANN_SAL >16000 THEN
 21  V_TAX_RATE := 10;
 22  END IF;
 23  V_TAX := EMP_REC.SAL * V_TAX_RATE/100;
 24  TOTAL_SAL := (NVL(EMP_REC.SAL,0) + NVL(EMP_REC.COMM,0) + NVL(V_BONUS,0))
 25               -
 26                (V_TAX + V_DED) ;
 27  -----------------DISPLAY---------------
 28  &D('ENAME IS ....'||EMP_REC.ENAME);
 29  &D('DESIGNATION IS ....'||EMP_REC.JOB);
 30  &D('SALARY  IS ....'||EMP_REC.SAL);
 31  &D('COMMISSION IS ....'||EMP_REC.COMM);
 32  &D('DEPTARTMENT IS ....'||EMP_REC.DEPTNO);
 33  &D('TAX RATE IS  = '||V_TAX_RATE||'%');
 34  &D('TAX IS ....'||V_TAX);
 35  &D('BONUS  IS ....'||V_BONUS);
 36  &D('DEDUCTION IS ....'||V_DED);
 37  &D('NET SALARY IS ....'||TOTAL_SAL);
 38* END;
SQL> /
Enter value for empno: 7788
ENAME IS ....SCOTT

DESIGNATION IS ....ANALYST

SALARY  IS ....3000

COMMISSION IS ....

DEPTARTMENT IS ....20

TAX RATE IS  = 10%

TAX IS ....300

BONUS  IS ....100

DEDUCTION IS ....100

NET SALARY IS ....2700


PL/SQL procedure successfully completed.

SQL>
SQL>
SQL>
SQL>
SQL>
SQL> ED
Wrote file afiedt.buf

  1  CREATE TABLE EMP_ATTEND
```

```
  2  (
  3  EMPNO NUMBER(4),
  4  TOTAL_DAYS NUMBER(2),
  5  ABSENTS NUMBER (2),
  6  ATTEND_MONTH DATE
  7* )
SQL> /

Table created.

SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL> DESC EMP_ATTEND
 Name                                                  Null?    Type
 ----------------------------------------------------- -------- ------------------------------------
 EMPNO                                                          NUMBER(4)
 TOTAL_DAYS                                                     NUMBER(2)
 ABSENTS                                                        NUMBER(2)
 ATTEND_MONTH                                                   DATE

SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL> INSERT INTO EMP_ATTEND(EMPNO)
  2  SELECT EMPNO FROM EMP;

14 rows created.

SQL>
SQL>
SQL>
SQL> COMMIT;

Commit complete.

SQL> SELECT * FROM EMP_ATTEND;

     EMPNO TOTAL_DAYS    ABSENTS ATTEND_MO

---------- ---------- ---------- ---------

      7369

      7499

      7521

      7566

      7654

      7698
```

```
      7782

      7788

      7839

      7844

      7876

      7900

      7902

      7934


14 rows selected.

SQL>
SQL>
SQL>
SQL>
SQL> UPDATE EMP_ATTEND
  2  SET TOTAL_DAYS=30,
  3  ABSENTS= 5,
  4  ATTEND_MONTH=SYSDATE
  5  WHERE JOB='SALESMAN';
WHERE JOB='SALESMAN'
      *
ERROR at line 5:
ORA-00904: "JOB": invalid identifier


SQL> ED
Wrote file afiedt.buf

  1  UPDATE EMP_ATTEND
  2  SET TOTAL_DAYS=30,
  3  ABSENTS= 5,
  4  ATTEND_MONTH=SYSDATE
  5* WHERE EMPNO>7500
SQL> /

12 rows updated.

SQL>
SQL>
SQL> COMMIT;

Commit complete.

SQL> SELECT * FROM EMP_ATTEND;

     EMPNO TOTAL_DAYS    ABSENTS ATTEND_MO

---------- ---------- ---------- ---------

      7369

      7499
```

```
      7521         30          7 12-FEB-13

      7566         30          5 12-FEB-13

      7654         30          5 12-FEB-13

      7698         30          5 12-FEB-13

      7782         30          8 12-FEB-13

      7788         30          5 12-FEB-13

      7839         30          4 12-FEB-13

      7844         30          5 12-FEB-13

      7876         30         10 12-FEB-13

      7900         30          5 12-FEB-13

      7902         30          5 12-FEB-13

      7934         30          5 12-FEB-13


14 rows selected.

SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL> ED
Wrote file afiedt.buf

  1  DECLARE
  2  V_EMPNO EMP.EMPNO%TYPE:=&EMPNO;
  3  EMP_REC EMP%ROWTYPE;
  4  TOTAL_SAL NUMBER :=0;
  5  V_TAX NUMBER :=0;
  6  V_TAX_RATE NUMBER :=0;
  7  V_ABT NUMBER :=0;
  8  V_DED NUMBER :=0;
  9  V_PER_SAL NUMBER :=0;
 10  V_ANN_SAL NUMBER :=0;
 11  V_BONUS NUMBER :=100;
 12  BEGIN
 13  SELECT * INTO EMP_REC FROM EMP
 14  WHERE EMPNO=V_EMPNO;
 15  SELECT ABSENTS INTO V_ABT FROM EMP_ATTEND
 16  WHERE EMPNO=V_EMPNO;
 17  V_PER_SAL := EMP_REC.SAL/30;
 18  V_DED := V_ABT * V_PER_SAL;
 19  V_ANN_SAL := EMP_REC.SAL * 12;
 20  IF V_ANN_SAL BETWEEN 10000 AND 12000 THEN
 21  V_TAX_RATE := 2;
 22  ELSIF V_ANN_SAL BETWEEN 12001 AND 14000 THEN
 23  V_TAX_RATE := 3;
 24  ELSIF V_ANN_SAL BETWEEN 14001 AND 16000 THEN
 25  V_TAX_RATE := 5;
 26  ELSIF V_ANN_SAL >16000 THEN
 27  V_TAX_RATE := 10;
```

```
 28  END IF;
 29  V_TAX := EMP_REC.SAL * V_TAX_RATE/100;
 30  TOTAL_SAL := (NVL(EMP_REC.SAL,0) + NVL(EMP_REC.COMM,0) + NVL(V_BONUS,0))
 31               -
 32                (V_TAX + V_DED) ;
 33  -----------------DISPLAY---------------
 34  &D('ENAME IS ....'||EMP_REC.ENAME);
 35  &D('DESIGNATION IS ....'||EMP_REC.JOB);
 36  &D('SALARY  IS ....'||EMP_REC.SAL);
 37  &D('COMMISSION IS ....'||EMP_REC.COMM);
 38  &D('DEPTARTMENT IS ....'||EMP_REC.DEPTNO);
 39  &D('TAX RATE IS  = '||V_TAX_RATE||'%');
 40  &D('TAX IS ....'||V_TAX);
 41  &D('BONUS  IS ....'||V_BONUS);
 42  &D('DEDUCTION IS ....'||V_DED);
 43  &D('NET SALARY IS ....'||TOTAL_SAL);
 44* END;
 45  /
Enter value for empno: 7788
ENAME IS ....SCOTT

DESIGNATION IS ....ANALYST

SALARY  IS ....3000

COMMISSION IS ....

DEPTARTMENT IS ....20

TAX RATE IS  = 10%

TAX IS ....300

BONUS  IS ....100

DEDUCTION IS ....500

NET SALARY IS ....2300


PL/SQL procedure successfully completed.

SQL>
SQL> ED
Wrote file afiedt.buf

  1  DECLARE
  2  V_EMPNO EMP.EMPNO%TYPE:=&EMPNO;
  3  EMP_REC EMP%ROWTYPE;
  4  TOTAL_SAL NUMBER :=0;
  5  V_TAX NUMBER :=0;
  6  V_TAX_RATE NUMBER :=0;
  7  V_ABT NUMBER :=0;
  8  V_DED NUMBER :=0;
  9  V_PER_SAL NUMBER :=0;
 10  V_ANN_SAL NUMBER :=0;
 11  V_BONUS NUMBER :=100;
 12  BEGIN
 13  SELECT * INTO EMP_REC FROM EMP
 14  WHERE EMPNO=V_EMPNO;
 15  SELECT ABSENTS INTO V_ABT FROM EMP_ATTEND
 16  WHERE EMPNO=V_EMPNO;
 17  V_PER_SAL := EMP_REC.SAL/30;
```

```
 18  V_DED := V_ABT * V_PER_SAL;
 19  V_ANN_SAL := EMP_REC.SAL * 12;
 20  IF V_ANN_SAL BETWEEN 10000 AND 12000 THEN
 21  V_TAX_RATE := 2;
 22  ELSIF V_ANN_SAL BETWEEN 12001 AND 14000 THEN
 23  V_TAX_RATE := 3;
 24  ELSIF V_ANN_SAL BETWEEN 14001 AND 16000 THEN
 25  V_TAX_RATE := 5;
 26  ELSIF V_ANN_SAL >16000 THEN
 27  V_TAX_RATE := 10;
 28  END IF;
 29  V_TAX := EMP_REC.SAL * V_TAX_RATE/100;
 30  TOTAL_SAL := (NVL(EMP_REC.SAL,0) + NVL(EMP_REC.COMM,0) + NVL(V_BONUS,0))
 31               -
 32                (V_TAX + V_DED) ;
 33  -----------------DISPLAY---------------
 34  &D('ENAME IS ....'||EMP_REC.ENAME);
 35  &D('DESIGNATION IS ....'||EMP_REC.JOB);
 36  &D('TOTAL ABSENTS ...'||V_ABT);
 37  &D('SALARY  IS ....'||EMP_REC.SAL);
 38  &D('COMMISSION IS ....'||EMP_REC.COMM);
 39  &D('DEPTARTMENT IS ....'||EMP_REC.DEPTNO);
 40  &D('TAX RATE IS  = '||V_TAX_RATE||'%');
 41  &D('TAX IS ....'||V_TAX);
 42  &D('BONUS  IS ....'||V_BONUS);
 43  &D('DEDUCTION IS ....'||V_DED);
 44  &D('NET SALARY IS ....'||TOTAL_SAL);
 45* END;
 46  /
Enter value for empno: 7788
ENAME IS ....SCOTT

DESIGNATION IS ....ANALYST

TOTAL ABSENTS ...5

SALARY  IS ....3000

COMMISSION IS ....

DEPTARTMENT IS ....20

TAX RATE IS  = 10%

TAX IS ....300

BONUS  IS ....100

DEDUCTION IS ....500

NET SALARY IS ....2300


PL/SQL procedure successfully completed.

SQL>
SQL>
SQL>
SQL>
SQL> --------SSOHAIL1@HOTMAIL.COM
SQL>
SQL>
SQL>
```

```
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL> ED
Wrote file afiedt.buf

  1  DECLARE
  2  V_EMPNO EMP.EMPNO%TYPE:=&EMPNO;
  3  EMP_REC EMP%ROWTYPE;
  4  TOTAL_SAL NUMBER :=0;
  5  V_TAX NUMBER :=0;
  6  V_TAX_RATE NUMBER :=0;
  7  V_ABT NUMBER :=0;
  8  V_DED NUMBER :=0;
  9  V_PER_SAL NUMBER :=0;
 10  V_ANN_SAL NUMBER :=0;
 11  V_BONUS NUMBER :=100;
 12  BEGIN
 13  SELECT * INTO EMP_REC FROM EMP
 14  WHERE EMPNO=V_EMPNO;
 15  SELECT ABSENTS INTO V_ABT FROM EMP_ATTEND
 16  WHERE EMPNO=V_EMPNO;
 17  V_PER_SAL := EMP_REC.SAL/30;
 18  V_DED := V_ABT * V_PER_SAL;
 19  V_ANN_SAL := EMP_REC.SAL * 12;
 20  IF V_ANN_SAL BETWEEN 10000 AND 12000 THEN
 21  V_TAX_RATE := 2;
 22  ELSIF V_ANN_SAL BETWEEN 12001 AND 14000 THEN
 23  V_TAX_RATE := 3;
 24  ELSIF V_ANN_SAL BETWEEN 14001 AND 16000 THEN
 25  V_TAX_RATE := 5;
 26  ELSIF V_ANN_SAL >16000 THEN
 27  V_TAX_RATE := 10;
 28  END IF;
 29  V_TAX := EMP_REC.SAL * V_TAX_RATE/100;
 30  TOTAL_SAL := (NVL(EMP_REC.SAL,0) + NVL(EMP_REC.COMM,0) + NVL(V_BONUS,0))
 31               -
 32                (V_TAX + V_DED) ;
 33  -----------------DISPLAY---------------
 34  &D('ENAME IS ....'||EMP_REC.ENAME);
 35  &D('DESIGNATION IS ....'||EMP_REC.JOB);
 36  &D('TOTAL ABSENTS ...'||V_ABT);
 37  &D('SALARY  IS ....'||EMP_REC.SAL);
 38  &D('COMMISSION IS ....'||EMP_REC.COMM);
 39  &D('DEPTARTMENT IS ....'||EMP_REC.DEPTNO);
 40  &D('TAX RATE IS  = '||V_TAX_RATE||'%');
 41  &D('TAX IS ....'||V_TAX);
 42  &D('BONUS  IS ....'||V_BONUS);
 43  &D('DEDUCTION IS ....'||V_DED);
 44  &D('NET SALARY IS ....'||TOTAL_SAL);
 45* END;
SQL> /
Enter value for empno: 7839
ENAME IS ....KING

DESIGNATION IS ....PRESIDENT

TOTAL ABSENTS ...4
```

```
SALARY  IS ....5000

COMMISSION IS ....

DEPTARTMENT IS ....10

TAX RATE IS  = 10%

TAX IS ....500

BONUS  IS ....100

DEDUCTION IS ....666.66666666666666666666666666666666668

NET SALARY IS ....3933.33333333333333333333333333333333332


PL/SQL procedure successfully completed.

SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL> ED
Wrote file afiedt.buf

  1  /*
  2    LOOP.
  3               (I) UNCONDITIONAL LOOP  (FOR).
  4               (II) CONDITIONAL LOOP  (WHILE,BASIC LOOP).
  5* */
  6  .
SQL>
SQL> ED
Wrote file afiedt.buf

  1  DECLARE
  2  A NUMBER :=0;
  3  BEGIN
  4  FOR A IN 1..10 LOOP
  5  &D(A);
  6  END LOOP;
  7* END;
  8  /
1

2

3

4

5

6

7
```

```
8

9

10


PL/SQL procedure successfully completed.

SQL>
SQL>
SQL>
SQL> ED
Wrote file afiedt.buf

  1  DECLARE
  2  A NUMBER :=0;
  3  BEGIN
  4  FOR A IN 1..10 LOOP
  5  &D(A);
  6  END LOOP;
  7  &D('AFTER LOOP..');
  8* END;
SQL> /
1

2

3

4

5

6

7

8

9

10

AFTER LOOP..


PL/SQL procedure successfully completed.

SQL> ED
Wrote file afiedt.buf

  1  DECLARE
  2  A NUMBER :=0;
  3  BEGIN
  4  FOR A IN 1..100 LOOP
  5  &D(A);
  6  END LOOP;
  7  &D('AFTER LOOP..');
  8* END;
SQL> /
1
```

```
97

98

99

100

AFTER LOOP..


PL/SQL procedure successfully completed.

SQL> ED
Wrote file afiedt.buf

  1  BEGIN
  2  FOR A IN 1..100 LOOP
  3  &D(A);
  4  END LOOP;
  5  &D('AFTER LOOP..');
  6* END;
  7  /
1

2

3

4

5

6

7

8

9

10

11

12

13

14

15

16

17

18

19

20
```

```
84

85

86

87

88

89

90

91

92

93

94

95

96

97

98

99

100

AFTER LOOP..


PL/SQL procedure successfully completed.

SQL> ED
Wrote file afiedt.buf

  1  BEGIN
  2  FOR A IN 1..10 LOOP
  3  &D(A||TO_CHAR(TO_DATE(A,'J'),'JSP'));
  4  END LOOP;
  5  &D('AFTER LOOP..');
  6* END;
SQL> /
1ONE

2TWO

3THREE

4FOUR

5FIVE

6SIX

7SEVEN
```

```
8EIGHT

9NINE

10TEN

AFTER LOOP..


PL/SQL procedure successfully completed.

SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL> ED
Wrote file afiedt.buf

  1  BEGIN
  2  FOR A IN 1..10 LOOP
  3  &D(A||'  '||TO_CHAR(TO_DATE(A,'J'),'JSP'));
  4  END LOOP;
  5  &D('AFTER LOOP..');
  6* END;
SQL> /
1  ONE

2  TWO

3  THREE

4  FOUR

5  FIVE

6  SIX

7  SEVEN

8  EIGHT

9  NINE

10  TEN

AFTER LOOP..


PL/SQL procedure successfully completed.

SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL> ED
Wrote file afiedt.buf

  1  DECLARE
```

```
  2  S_NO NUMBER :=&STARTING_NO;
  3  E_NO NUMBER :=&ENDING_NO;
  4  BEGIN
  5  FOR A IN S_NO..E_NO LOOP
  6  &D(A||'  '||TO_CHAR(TO_DATE(A,'J'),'JSP'));
  7  END LOOP;
  8  &D('AFTER LOOP..');
  9* END;
 10  /
Enter value for starting_no: 5
Enter value for ending_no: 12
5  FIVE

6  SIX

7  SEVEN

8  EIGHT

9  NINE

10  TEN

11  ELEVEN

12  TWELVE

AFTER LOOP..


PL/SQL procedure successfully completed.

SQL>
SQL>
SQL>
SQL>
SQL>
SQL> /
Enter value for starting_no: 80
Enter value for ending_no: 89
80  EIGHTY

81  EIGHTY-ONE

82  EIGHTY-TWO

83  EIGHTY-THREE

84  EIGHTY-FOUR

85  EIGHTY-FIVE

86  EIGHTY-SIX

87  EIGHTY-SEVEN

88  EIGHTY-EIGHT

89  EIGHTY-NINE

AFTER LOOP..
```

```
PL/SQL procedure successfully completed.

SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL> ED
Wrote file afiedt.buf

  1  DECLARE
  2  S_NO NUMBER :=&STARTING_NO;
  3  E_NO NUMBER :=&ENDING_NO;
  4  BEGIN
  5  FOR A IN S_NO..E_NO LOOP
  6  &D(A||'  '||TO_CHAR(TO_DATE(A,'J'),'JSP'));
  7  END LOOP;
  8  &D('AFTER LOOP..');
  9* END;
SQL>
SQL> /
Enter value for starting_no: 8
Enter value for ending_no: 7
AFTER LOOP..


PL/SQL procedure successfully completed.

SQL>
SQL>
SQL>
SQL>
SQL> ED
Wrote file afiedt.buf

  1  DECLARE
  2  S_NO NUMBER :=&STARTING_NO;
  3  E_NO NUMBER :=&ENDING_NO;
  4  BEGIN
  5  IF S_NO<=E_NO THEN
  6  FOR A IN S_NO..E_NO LOOP
  7  &D(A||'  '||TO_CHAR(TO_DATE(A,'J'),'JSP'));
  8  END LOOP;
  9  ELSE
 10  &D('INVALID NO');
 11  END IF;
 12* END;
 13  /
Enter value for starting_no: 5
Enter value for ending_no: 8
5  FIVE

6  SIX

7  SEVEN

8  EIGHT


PL/SQL procedure successfully completed.
```

```
SQL>
SQL>
SQL>
SQL> /
Enter value for starting_no: 8
Enter value for ending_no: 5
INVALID NO


PL/SQL procedure successfully completed.

SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL> ED
Wrote file afiedt.buf

  1  DECLARE
  2  T_NO NUMBER :=&TABLE_NO;
  3  S_NO NUMBER :=&STARTING_NO;
  4  E_NO NUMBER :=&ENDING_NO;
  5  CNTR NUMBER :=0;
  6  BEGIN
  7  IF S_NO<=E_NO THEN
  8  FOR A IN S_NO..E_NO LOOP
  9  CNTR := T_NO * A;
 10  &D(T_NO||' X '||A||' = '||CNTR );
 11  END LOOP;
 12  ELSE
 13  &D('INVALID NO');
 14  END IF;
 15* END;
 16  /
Enter value for table_no: 5
Enter value for starting_no: 1
Enter value for ending_no: 10
5 X 1 = 5

5 X 2 = 10

5 X 3 = 15

5 X 4 = 20

5 X 5 = 25

5 X 6 = 30

5 X 7 = 35

5 X 8 = 40

5 X 9 = 45

5 X 10 = 50


PL/SQL procedure successfully completed.
```

```
SQL>
SQL>
SQL>
SQL>
SQL> /
Enter value for table_no: 5
Enter value for starting_no: 6
Enter value for ending_no: 8
5 X 6 = 30

5 X 7 = 35

5 X 8 = 40


PL/SQL procedure successfully completed.

SQL>
SQL>
SQL>
SQL> /
Enter value for table_no: \5
Enter value for starting_no:
Enter value for ending_no:
T_NO NUMBER :=\5;
              *
ERROR at line 2:
ORA-06550: line 2, column 15:
PLS-00103: Encountered the symbol "\" when expecting one of the following:
( - + case mod new not null <an identifier>
<a double-quoted delimited-identifier> <a bind variable> avg
count current exists max min prior sql stddev sum variance
execute forall merge time timestamp interval date
<a string literal with character set specification>
<a number> <a single-quoted SQL string> pipe
<an alternatively-quoted string literal with character set specification>
<an alternatively-quoted S
ORA-06550: line 3, column 15:
PLS-00103: Encountered the symbol ";" when expecting one of the following:
( - + case mod new not null <an identifier>
<a double-quoted delimited-identifier> <a bind variable> avg
count current exists max min prior sql stddev su
ORA-06550: line 4, column 15:
PLS-00103: Encountered the symbol ";" when expecting one of the following:
( - + case mod new not null <an identifier>
<a double-quoted delimited-identifier> <a bind variable>


SQL> /
Enter value for table_no: 5
Enter value for starting_no: 9
Enter value for ending_no: 5
INVALID NO


PL/SQL procedure successfully completed.

SQL>
SQL>
SQL>
SQL> /
Enter value for table_no: 5
```

```
Enter value for starting_no: 6
Enter value for ending_no: 6
5 X 6 = 30


PL/SQL procedure successfully completed.

SQL>
SQL>
SQL>
SQL> ED
Wrote file afiedt.buf

  1  DECLARE
  2  S_NO NUMBER :=&STARTING_NO;
  3  E_NO NUMBER :=&ENDING_NO;
  4  BEGIN
  5  IF S_NO<=E_NO THEN
  6  FOR A IN S_NO..E_NO LOOP
  7  &D(A);
  8  END LOOP;
  9  ELSE
 10  &D('INVALID NO');
 11  END IF;
 12* END;
 13  /
Enter value for starting_no: 1
Enter value for ending_no: 5
1

2

3

4

5


PL/SQL procedure successfully completed.

SQL>
SQL>
SQL>
SQL>
SQL>
SQL> ED
Wrote file afiedt.buf

  1  DECLARE
  2  S_NO NUMBER :=&STARTING_NO;
  3  E_NO NUMBER :=&ENDING_NO;
  4  RESULT VARCHAR2(100);
  5  BEGIN
  6  IF S_NO<=E_NO THEN
  7  FOR A IN S_NO..E_NO LOOP
  8   RESULT := RESULT ||A||' ';
  9  END LOOP;
 10  &D(RESULT);
 11  ELSE
 12  &D('INVALID NO');
 13  END IF;
 14* END;
```

```
 15  /
Enter value for starting_no: 1
Enter value for ending_no: 5
1 2 3 4 5


PL/SQL procedure successfully completed.

SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL> ED
Wrote file afiedt.buf

  1  DECLARE
  2  S_NO NUMBER :=&STARTING_NO;
  3  E_NO NUMBER :=&ENDING_NO;
  4  R VARCHAR2(100);
  5  BEGIN
  6  IF S_NO<=E_NO THEN
  7  FOR A IN S_NO..E_NO LOOP
  8   R := R ||A||' ';
  9  END LOOP;
 10  &D(R);
 11  ELSE
 12  &D('INVALID NO');
 13  END IF;
 14* END;
SQL> /
Enter value for starting_no: 1
Enter value for ending_no: 10
1 2 3 4 5 6 7 8 9 10


PL/SQL procedure successfully completed.

SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL> ED
Wrote file afiedt.buf

  1  DECLARE
  2  S_NO NUMBER :=&STARTING_NO;
  3  E_NO NUMBER :=&ENDING_NO;
  4  R VARCHAR2(100);
  5  BEGIN
  6  IF S_NO<=E_NO THEN
  7  FOR A IN S_NO..E_NO LOOP
```

```
  8   R := R ||A||' ';
  9  &D(R);
 10  END LOOP;
 11  ELSE
 12  &D('INVALID NO');
 13  END IF;
 14* END;
 15  /
Enter value for starting_no: 1
Enter value for ending_no: 10
1

1 2

1 2 3

1 2 3 4

1 2 3 4 5

1 2 3 4 5 6

1 2 3 4 5 6 7

1 2 3 4 5 6 7 8

1 2 3 4 5 6 7 8 9

1 2 3 4 5 6 7 8 9 10


PL/SQL procedure successfully completed.

SQL>
SQL>
SQL>
SQL>
SQL>
SQL> ED
Wrote file afiedt.buf

  1  BEGIN
  2  FOR A IN 1..10 LOOP
  3  &D(A);
  4  END LOOP;
  5  ELSE
  6  &D('INVALID NO');
  7  END IF;
  8* END;
  9  /
ELSE
*
ERROR at line 5:
ORA-06550: line 5, column 1:
PLS-00103: Encountered the symbol "ELSE" when expecting one of the following:
begin case declare end exception exit for goto if loop mod
null pragma raise return select update while with
<an identifier> <a double-quoted delimited-identifier>
<a bind variable> << close current delete fetch lock insert
open rollback savepoint set sql execute commit forall merge
pipe
ORA-06550: line 8, column 1:
PLS-00103: Encountered the symbol "END"
```

```
SQL> ED
Wrote file afiedt.buf

  1  BEGIN
  2  FOR A IN 1..10 LOOP
  3  &D(A);
  4  END LOOP;
  5  &D('INVALID NO');
  6* END;
  7  /
1

2

3

4

5

6

7

8

9

10

INVALID NO


PL/SQL procedure successfully completed.

SQL>
SQL>
SQL>
SQL> ED
Wrote file afiedt.buf

  1  BEGIN
  2  FOR A IN 1..10 LOOP
  3  IF A MOD 2 =0 THEN
  4  &D('EVEN NO...'||A);
  5  ELSE
  6  &D('ODD NO...'||A);
  7  END IF;
  8  END LOOP;
  9  &D('INVALID NO');
 10* END;
 11  /
ODD NO...1

EVEN NO...2

ODD NO...3

EVEN NO...4

ODD NO...5
```

```
EVEN NO...6

ODD NO...7

EVEN NO...8

ODD NO...9

EVEN NO...10

INVALID NO


PL/SQL procedure successfully completed.

SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL> ED
Wrote file afiedt.buf

  1  DECLARE
  2  SUM_OF_EVEN NUMBER :=0;
  3  SUM_OF_ODD NUMBER :=0;
  4  BEGIN
  5  FOR A IN 1..10 LOOP
  6  IF A MOD 2 =0 THEN
  7  &D('EVEN NO...'||A);
  8  SUM_OF_EVEN  := SUM_OF_EVEN  + A;
  9  ELSE
 10  &D('ODD NO...'||A);
 11  SUM_OF_ODD  := SUM_OF_ODD  + A;
 12  END IF;
 13  END LOOP;
 14  &D('SUM OF EVEN NUMBERS ARE ..'||SUM_OF_EVEN);
 15  &D('SUM OF ODD NUMBERS ARE ..'||SUM_OF_ODD);
 16* END;
 17  /
ODD NO...1

EVEN NO...2

ODD NO...3

EVEN NO...4

ODD NO...5

EVEN NO...6

ODD NO...7

EVEN NO...8

ODD NO...9
```

```
EVEN NO...10

SUM OF EVEN NUMBERS ARE ..30

SUM OF ODD NUMBERS ARE ..25


PL/SQL procedure successfully completed.

SQL>
SQL>
SQL>
SQL>
SQL>
SQL> ED
Wrote file afiedt.buf

  1  BEGIN
  2  FOR A IN 1..300 LOOP
  3  &D(A);
  4  END LOOP;
  5* END;
  6  /
1

2

3

4

5

6

7

8

9

10

11

12

13

14

15

16

17

18

19

20
```

84
85
86
87
88
89
90
91
92
93
94
95
96
97
98
99
100
101
102
103
104
105
106
107
108
109
110
111
112
113
114
115

273
274
275
276
277
278
279
280
281
282
283
284
285
286
287
288
289
290
291
292
293
294
295
296
297
298
299
300

PL/SQL procedure successfully completed.

SQL> ED
Wrote file afiedt.buf

  1 BEGIN

```
  2  FOR A IN 1..300 LOOP
  3  &D(A||'  '||CHR(A) );
  4  END LOOP;
  5* END;
SQL> /
1

2

3

4

5

6

7

8

9

10


11

12

13


14

15

16

17

18

19

20

21

22

23

24

25

26

27

28
```

29
30
31
32
33 !
34 "
35 #
36 $
37 %
38 &
39 '
40 (
41)
42 *
43 +
44 ,
45 -
46 .
47 /
48 0
49 1
50 2
51 3
52 4
53 5
54 6
55 7
56 8
57 9
58 :
59 ;
60 <

61 =
62 >
63 ?
64 @
65 A
66 B
67 C
68 D
69 E
70 F
71 G
72 H
73 I
74 J
75 K
76 L
77 M
78 N
79 O
80 P
81 Q
82 R
83 S
84 T
85 U
86 V
87 W
88 X
89 Y
90 Z
91 [

92 \
93]
94 ^
95 _
96 `
97 a
98 b
99 c
100 d
101 e
102 f
103 g
104 h
105 i
106 j
107 k
108 l
109 m
110 n
111 o
112 p
113 q
114 r
115 s
116 t
117 u
118 v
119 w
120 x
121 y
122 z
123 {

124 |

125 }

126 ~

127

128 €

129

130 ,

131 ƒ

132 „

133 …

134 †

135 ‡

136 ˆ

137 ‰

138 Š

139 ‹

140 Œ

141

142 Ž

143

144

145 ‘

146 ’

147 “

148 ”

149 •

150 –

151 —

152 ˜

153 ™

154 š

155 ›

156 œ

157

158 ž

159 Ÿ

160

161 ¡

162 ¢

163 £

164 ¤

165 ¥

166 ¦

167 §

168 ¨

169 ©

170 ª

171 «

172 ¬

173 -

174 ®

175 ¯

176 °

177 ±

178 ²

179 ³

180 ´

181 µ

182 ¶

183 ·

184 ¸

185 ¹

186 º

187 »
188 ¼
189 ½
190 ¾
191 ¿
192 À
193 Á
194 Â
195 Ã
196 Ä
197 Å
198 Æ
199 Ç
200 È
201 É
202 Ê
203 Ë
204 Ì
205 Í
206 Î
207 Ï
208 Ð
209 Ñ
210 Ò
211 Ó
212 Ô
213 Õ
214 Ö
215 ×
216 Ø
217 Ù

218 Ú
219 Û
220 Ü
221 Ý
222 Þ
223 ß
224 à
225 á
226 â
227 ã
228 ä
229 å
230 æ
231 ç
232 è
233 é
234 ê
235 ë
236 ì
237 í
238 î
239 ï
240 ð
241 ñ
242 ò
243 ó
244 ô
245 õ
246 ö
247 ÷
248 ø
249 ù

250 ú
251 û
252 ü
253 ý
254 þ
255 ÿ
256
257
258
259
260
261
262
263
264
265
266
267
268
269
270
271
272
273
274
275
276
277
278
279

```
280
281
282
283
284
285
286
287
288
289  !
290  "
291  #
292  $
293  %
294  &
295  '
296  (
297  )
298  *
299  +
300  ,

PL/SQL procedure successfully completed.

SQL> ED
Wrote file afiedt.buf

  1  DECLARE
  2  S_ALPH VARCHAR2(1):='&STARTING_ALPHA';
  3  E_ALPH VARCHAR2(1):='&ENDING_ALPHA';
  4  BEGIN
  5  FOR A IN ASC(S_ALPH)..ASC(E_ALPH) LOOP
  6  &D(A||'  '||CHR(A) );
  7  END LOOP;
  8* END;
  9  /
Enter value for starting_alpha: 1
Enter value for ending_alpha:
FOR A IN ASC(S_ALPH)..ASC(E_ALPH) LOOP
         *
ERROR at line 5:
ORA-06550: line 5, column 10:
```

```
PLS-00103: Encountered the symbol "ASC" when expecting one of the following:
( - + case mod new null <an identifier>
<a double-quoted delimited-identifier> <a bind variable>
reverse avg count current max min prior sql stddev sum
variance execute forall merge <a SQL statement> time
timestamp interval date
<a string literal with character set specification>
<a number> <a single-quoted SQL string> pipe
<an alternatively-quoted string literal with character set specification>
<an al
ORA-06550: line 5, column 23:
PLS-00103: Encountered the symbol "ASC" when expecting one of the following:
( - + case mod new null <an identifier>
<a double-quoted delimited-identifier> <a bind variable> avg
count current max min prior sql stddev sum varianc


SQL>
SQL> /
Enter value for starting_alpha: A
Enter value for ending_alpha: D
FOR A IN ASC(S_ALPH)..ASC(E_ALPH) LOOP
         *
ERROR at line 5:
ORA-06550: line 5, column 10:
PLS-00103: Encountered the symbol "ASC" when expecting one of the following:
( - + case mod new null <an identifier>
<a double-quoted delimited-identifier> <a bind variable>
reverse avg count current max min prior sql stddev sum
variance execute forall merge <a SQL statement> time
timestamp interval date
<a string literal with character set specification>
<a number> <a single-quoted SQL string> pipe
<an alternatively-quoted string literal with character set specification>
<an al
ORA-06550: line 5, column 23:
PLS-00103: Encountered the symbol "ASC" when expecting one of the following:
( - + case mod new null <an identifier>
<a double-quoted delimited-identifier> <a bind variable> avg
count current max min prior sql stddev sum varianc


SQL> SELECT CHR(65),ASCII('A') FROM DUAL;

C ASCII('A')

- ----------

A         65


SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL> DECLARE
  2  S_ALPH VARCHAR2(1):='&STARTING_ALPHA';
  3  E_ALPH VARCHAR2(1):='&ENDING_ALPHA';
```

```
  4  BEGIN
  5  FOR A IN ASC(S_ALPH)..ASC(E_ALPH) LOOP
  6  &D(A||'  '||CHR(A) );
  7  END LOOP;
  8  END;
  9  .
SQL> ED
Wrote file afiedt.buf

  1  DECLARE
  2  S_ALPH VARCHAR2(1):='&STARTING_ALPHA';
  3  E_ALPH VARCHAR2(1):='&ENDING_ALPHA';
  4  BEGIN
  5  FOR A IN ASCII(S_ALPH)..ASCII(E_ALPH) LOOP
  6  &D(A||'  '||CHR(A) );
  7  END LOOP;
  8* END;
SQL> /
Enter value for starting_alpha: A
Enter value for ending_alpha: D
65  A

66  B

67  C

68  D


PL/SQL procedure successfully completed.

SQL>
SQL>
SQL>
SQL>
SQL> ED
Wrote file afiedt.buf

  1  DECLARE
  2  S_ALPH VARCHAR2(1):='&STARTING_ALPHA';
  3  E_ALPH VARCHAR2(1):='&ENDING_ALPHA';
  4  BEGIN
  5  FOR A IN ASCII(S_ALPH)..ASCII(E_ALPH) LOOP
  6  &D(CHR(A) );
  7  END LOOP;
  8* END;
SQL> /
Enter value for starting_alpha: A
Enter value for ending_alpha: F
A

B

C

D

E

F


PL/SQL procedure successfully completed.
```

```
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL> /
Enter value for starting_alpha: a
Enter value for ending_alpha: d
a

b

c

d


PL/SQL procedure successfully completed.

SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL> spool off
```

```
SQL>
SQL>
SQL>
SQL> DELCARE
SP2-0042: unknown command "DELCARE" - rest of line ignored.
SQL> DECLARE
  2  .
SQL> ED
Wrote file afiedt.buf

  1  DECLARE
  2  USER_NAME VARCHAR2(100):='&YOUR_NAME';
  3  BEGIN
  4  FOR I IN 1..LENGTH(USER_NAME) LOOP
  5  &D(I);
  6  END LOOP;
  7* END;
  8  /
Enter value for your_name: PAKISTAN
1

2

3

4

5

6

7

8


PL/SQL procedure successfully completed.

SQL>
SQL>
SQL>
SQL> ED
Wrote file afiedt.buf

  1  DECLARE
  2  USER_NAME VARCHAR2(100):='&YOUR_NAME';
  3  LEN NUMBER :=0;
  4  BEGIN
  5  LEN := LENGTH(USER_NAME);
  6  FOR I IN 1..LEN LOOP
  7  &D(I);
  8  END LOOP;
  9* END;
 10  /
Enter value for your_name: ASDF
1

2

3

4
```

```
PL/SQL procedure successfully completed.

SQL>
SQL>
SQL>
SQL>
SQL> ED
Wrote file afiedt.buf

  1  DECLARE
  2  USER_NAME VARCHAR2(100):='&YOUR_NAME';
  3  LEN NUMBER :=0;
  4  BEGIN
  5  LEN := LENGTH(USER_NAME);
  6  &D('LENGTH OF YOUR NAME IS'||LEN);
  7  FOR I IN 1..LEN LOOP
  8  &D(I);
  9  END LOOP;
 10* END;
 11  /
Enter value for your_name: ALI
LENGTH OF YOUR NAME IS3

1

2

3


PL/SQL procedure successfully completed.

SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL> /
Enter value for your_name: PAKISTAN
LENGTH OF YOUR NAME IS8

1

2

3

4

5

6

7

8


PL/SQL procedure successfully completed.

SQL>
```

```
SQL> ED
Wrote file afiedt.buf

  1  DECLARE
  2  USER_NAME VARCHAR2(100):='&YOUR_NAME';
  3  LEN NUMBER :=0;
  4  BEGIN
  5  LEN := LENGTH(USER_NAME);
  6  &D('LENGTH OF YOUR NAME IS..'||LEN);
  7  FOR I IN 1..LEN LOOP
  8  &D(SUBSTR(USER_NAME,1,I));
  9  END LOOP;
 10* END;
SQL> /
Enter value for your_name: PAKISTAN
LENGTH OF YOUR NAME IS..8

P

PA

PAK

PAKI

PAKIS

PAKIST

PAKISTA

PAKISTAN


PL/SQL procedure successfully completed.

SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL> ED
Wrote file afiedt.buf

  1  DECLARE
  2  USER_NAME VARCHAR2(100):='&YOUR_NAME';
  3  LEN NUMBER :=0;
  4  BEGIN
  5  LEN := LENGTH(USER_NAME);
  6  &D('LENGTH OF YOUR NAME IS..'||LEN);
  7  FOR I IN 1..LEN LOOP
  8  &D(SUBSTR(USER_NAME,I,1));
  9  END LOOP;
 10* END;
SQL> /
Enter value for your_name: PAKISTAN
LENGTH OF YOUR NAME IS..8

P
```

```
A

K

I

S

T

A

N


PL/SQL procedure successfully completed.

SQL>
SQL>
SQL>
SQL>
SQL>
SQL> ED
Wrote file afiedt.buf

  1  DECLARE
  2  USER_NAME VARCHAR2(100):='&YOUR_NAME';
  3  LEN NUMBER :=0;
  4  BEGIN
  5  LEN := LENGTH(USER_NAME);
  6  &D('LENGTH OF YOUR NAME IS..'||LEN);
  7  FOR I IN 1..LEN LOOP
  8  &D(SUBSTR(USER_NAME,I,I));
  9  END LOOP;
 10* END;
SQL> /
Enter value for your_name: PAKISTAN
LENGTH OF YOUR NAME IS..8

P

AK

KIS

ISTA

STAN

TAN

AN

N


PL/SQL procedure successfully completed.

SQL>
SQL>
SQL>
SQL>
```

```
SQL>
SQL> ED
Wrote file afiedt.buf

  1  DECLARE
  2  USER_NAME VARCHAR2(100):='&YOUR_NAME';
  3  LEN NUMBER :=0;
  4  BEGIN
  5  LEN := LENGTH(USER_NAME);
  6  &D('LENGTH OF YOUR NAME IS..'||LEN);
  7  FOR I IN 1..LEN LOOP
  8  &D(SUBSTR(USER_NAME,I,1));
  9  END LOOP;
 10* END;
SQL> /
Enter value for your_name: PAKISTAN
LENGTH OF YOUR NAME IS..8

P

A

K

I

S

T

A

N


PL/SQL procedure successfully completed.

SQL>
SQL>
SQL>
SQL> ED
Wrote file afiedt.buf

  1  DECLARE
  2  USER_NAME VARCHAR2(100):='&YOUR_NAME';
  3  LEN NUMBER :=0;
  4  BEGIN
  5  LEN := LENGTH(USER_NAME);
  6  &D('LENGTH OF YOUR NAME IS..'||LEN);
  7  FOR I IN 1..LEN LOOP
  8  IF ASCII(SUBSTR(USER_NAME,I,1)) BETWEEN 65 AND 90 THEN
  9  &D('UPPER CASE NOT ALLOWED..');
 10  ELSE
 11  (SUBSTR(USER_NAME,I,1));
 12  END IF;
 13  END LOOP;
 14* END;
 15  /
Enter value for your_name: pakistan
(SUBSTR(USER_NAME,I,1));
*
ERROR at line 11:
ORA-06550: line 11, column 1:
```

```
PLS-00103: Encountered the symbol "(" when expecting one of the following:
begin case declare exit for goto if loop mod null pragma
raise return select update while with <an identifier>
<a double-quoted delimited-identifier> <a bind variable> <<
close current delete fetch lock insert open rollback
savepoint set sql execute commit forall merge pipe
The symbol "case" was substituted for "(" to continue.
ORA-06550: line 11, column 24:
PLS-00103: Encountered the symbol ";" when expecting one of the following:
* & = - + < / > at in is mod remainder not rem when
<an exponent (**)> <> or != or ~= >= <= <> and or like LIKE2_
LIKE4_ LIKEC_ between overlaps || mult


SQL>
SQL>
SQL>
SQL>
SQL> ed
Wrote file afiedt.buf

  1  DECLARE
  2  USER_NAME VARCHAR2(100):='&YOUR_NAME';
  3  LEN NUMBER :=0;
  4  BEGIN
  5  LEN := LENGTH(USER_NAME);
  6  &D('LENGTH OF YOUR NAME IS..'||LEN);
  7  FOR I IN 1..LEN LOOP
  8  IF ASCII(SUBSTR(USER_NAME,I,1)) BETWEEN 65 AND 90 THEN
  9  &D('UPPER CASE NOT ALLOWED..');
 10  ELSE
 11  &d(SUBSTR(USER_NAME,I,1));
 12  END IF;
 13  END LOOP;
 14* END;
SQL> /
Enter value for your_name: pakistan
LENGTH OF YOUR NAME IS..8

p

a

k

i

s

t

a

n


PL/SQL procedure successfully completed.

SQL>
SQL>
SQL>
SQL> /
Enter value for your_name: pakisTan
```

```
LENGTH OF YOUR NAME IS..8

p

a

k

i

s

UPPER CASE NOT ALLOWED..

a

n


PL/SQL procedure successfully completed.

SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL> ed
Wrote file afiedt.buf

  1  DECLARE
  2  USER_NAME VARCHAR2(100):='&YOUR_NAME';
  3  LEN NUMBER :=0;
  4  RES VARCHAR2(200);
  5  BEGIN
  6  LEN := LENGTH(USER_NAME);
  7  &D('LENGTH OF YOUR NAME IS..'||LEN);
  8  FOR I IN 1..LEN LOOP
  9  IF (ASCII(SUBSTR(USER_NAME,I,1)) BETWEEN 65 AND 90 )
 10  THEN
 11  &D('UPPER CASE NOT ALLOWED..');
 12  ELSE
 13    RES := RES||(SUBSTR(USER_NAME,I,1))||' ';
 14  END IF;
 15  END LOOP;
 16  &D(RES);
 17* END;
 18  /
Enter value for your_name: pakisTan
LENGTH OF YOUR NAME IS..8

UPPER CASE NOT ALLOWED..

p a k i s a n


PL/SQL procedure successfully completed.

SQL>
SQL>
SQL>
```

```
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL> ed
Wrote file afiedt.buf

  1  DECLARE
  2  USER_NAME VARCHAR2(100):='&YOUR_NAME';
  3  LEN NUMBER :=0;
  4  RES VARCHAR2(200);
  5  CNTR NUMBER :=0;
  6  BEGIN
  7  LEN := LENGTH(USER_NAME);
  8  &D('LENGTH OF YOUR NAME IS..'||LEN);
  9  FOR I IN 1..LEN LOOP
 10  IF (ASCII(SUBSTR(USER_NAME,I,1)) BETWEEN 65 AND 90 )
 11  THEN
 12  &D('UPPER CASE NOT ALLOWED..');
 13  CNTR := CNTR +1;
 14  ELSE
 15    RES := RES||(SUBSTR(USER_NAME,I,1))||' ';
 16  END IF;
 17  END LOOP;
 18  IF CNTR =0 THEN
 19  &D(RES);
 20  END IF;
 21* END;
 22  /
Enter value for your_name: pakistan
LENGTH OF YOUR NAME IS..8

p a k i s t a n


PL/SQL procedure successfully completed.

SQL>
SQL>
SQL>
SQL> /
Enter value for your_name: pakisTan
LENGTH OF YOUR NAME IS..8

UPPER CASE NOT ALLOWED..


PL/SQL procedure successfully completed.

SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL> ed
Wrote file afiedt.buf
```

```
  1  DECLARE
  2  USER_NAME VARCHAR2(100):='&YOUR_NAME';
  3  LEN NUMBER :=0;
  4  RES VARCHAR2(200);
  5  BEGIN
  6  LEN := LENGTH(USER_NAME);
  7  &D('LENGTH OF YOUR NAME IS..'||LEN);
  8  RES := SUBSTR(USER_NAME,1,1);
  9  FOR I IN 1..LEN LOOP
 10  IF ASCII(SUBSTR(USER_NAME,I,1))=32 THEN
 11  RES :=  RES ||SUBSTR(USER_NAME,I+1,1);
 12  END IF;
 13  END LOOP;
 14  &D(RES);
 15* END;
 16  /
Enter value for your_name: PAKIST INTER AIR
LENGTH OF YOUR NAME IS..16

PIA


PL/SQL procedure successfully completed.

SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL> ED
Wrote file afiedt.buf

  1  DECLARE
  2  USER_NAME VARCHAR2(100):='&YOUR_NAME';
  3  LEN NUMBER :=0;
  4  RES VARCHAR2(200);
  5  BEGIN
  6  LEN := LENGTH(USER_NAME);
  7  &D('LENGTH OF YOUR NAME IS..'||LEN);
  8  RES := SUBSTR(USER_NAME,1,1);
  9  FOR I IN 1..LEN LOOP
 10  IF ASCII(SUBSTR(USER_NAME,I,1))=32 THEN
 11  RES :=  RES ||SUBSTR(USER_NAME,I+1,1);
 12  &D(RES);
 13  END IF;
 14  END LOOP;
 15  &D(RES);
 16* END;
 17  /
Enter value for your_name: PAKIST INTER AIR
LENGTH OF YOUR NAME IS..16

PI

PIA

PIA
```

```
PL/SQL procedure successfully completed.

SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL> ED
Wrote file afiedt.buf

  1  DECLARE
  2  USER_NAME VARCHAR2(100):='&YOUR_NAME';
  3  LEN NUMBER :=0;
  4  RES VARCHAR2(200);
  5  BEGIN
  6  LEN := LENGTH(USER_NAME);
  7  &D('LENGTH OF YOUR NAME IS..'||LEN);
  8  RES := SUBSTR(USER_NAME,1,1);
  9  &D(RES);
 10  FOR I IN 1..LEN LOOP
 11  IF ASCII(SUBSTR(USER_NAME,I,1))=32 THEN
 12  RES :=  RES ||SUBSTR(USER_NAME,I+1,1);
 13  &D(RES);
 14  END IF;
 15  END LOOP;
 16  ---D(RES);
 17* END;
SQL> /
Enter value for your_name: PAKIST INTER AIR
LENGTH OF YOUR NAME IS..16

P

PI

PIA


PL/SQL procedure successfully completed.

SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL> ED
Wrote file afiedt.buf

  1  BEGIN
  2  FOR I IN 1..10 LOOP
  3  &D(I);
  4  END LOOP;
  5* END;
  6  /
1

2
```

```
3

4

5

6

7

8

9

10


PL/SQL procedure successfully completed.

SQL>
SQL>
SQL>
SQL>
SQL> ED
Wrote file afiedt.buf

  1  BEGIN
  2  FOR I REVERSE IN 1..10 LOOP
  3  &D(I);
  4  END LOOP;
  5* END;
SQL> /
FOR I REVERSE IN 1..10 LOOP
      *
ERROR at line 2:
ORA-06550: line 2, column 7:
PLS-00103: Encountered the symbol "REVERSE" when expecting one of the following:
in
The symbol "REVERSE" was ignored.


SQL>
SQL>
SQL>
SQL> ED
Wrote file afiedt.buf

  1  BEGIN
  2  FOR I  IN REVERSE 1..10 LOOP
  3  &D(I);
  4  END LOOP;
  5* END;
SQL> /
10

9

8

7

6
```

```
5

4

3

2

1


PL/SQL procedure successfully completed.

SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL> ED
Wrote file afiedt.buf

  1  BEGIN
  2  FOR I IN (SELECT * FROM EMP)  LOOP
  3  &D(I.EMPNO||'  '||I.ENAME||'   '||I.JOB||'  '||I.SAL);
  4  END LOOP;
  5* END;
SQL> /
125  DS     100

5454  SMITH   CLERK  900

7499  ALLEN   SALESMAN  1600

7521  WARD   SALESMAN  1250

7566  JONES   MANAGER  2975

7654  MARTIN   SALESMAN  1250

7698  BLAKE   MANAGER  2850

7782  CLARK   MANAGER  2450

7788  SCOTT   ANALYST  3000

7839  KING   PRESIDENT  5000

7844  TURNER   SALESMAN  1500

7876  ADAMS   CLERK  1100

7900  JAMES   CLERK  950

7902  FORD   ANALYST  45666

7934  MILLER   CLERK  1300


PL/SQL procedure successfully completed.

SQL>
```

```
SQL>
SQL>
SQL>
SQL> ED
Wrote file afiedt.buf

  1  BEGIN
  2  FOR I IN (SELECT * FROM EMP E WHERE  E.JOB='&JOB')  LOOP
  3  &D(I.EMPNO||'  '||I.ENAME||'   '||I.JOB||'  '||I.SAL);
  4  END LOOP;
  5* END;
SQL> /
Enter value for job: SALESMAN
7499  ALLEN   SALESMAN  1600

7521  WARD   SALESMAN  1250

7654  MARTIN   SALESMAN  1250

7844  TURNER   SALESMAN  1500


PL/SQL procedure successfully completed.

SQL>
SQL>
SQL>
SQL> ED
Wrote file afiedt.buf

  1  BEGIN
  2  FOR I IN (SELECT * FROM EMP E,DEPT D WHERE  E.DEPTNO=D.DEPTNO)  LOOP
  3  &D(I.EMPNO||'  '||I.ENAME||'   '||I.JOB||'  '||I.SAL||'  '||I.DNAME);
  4  END LOOP;
  5* END;
SQL> /
FOR I IN (SELECT * FROM EMP E,DEPT D WHERE  E.DEPTNO=D.DEPTNO)  LOOP
*
ERROR at line 2:
ORA-06550: line 2, column 1:
PLS-00402: alias required in SELECT list of cursor to avoid duplicate column names
ORA-06550: line 2, column 1:
PL/SQL: Statement ignored


SQL>
SQL>
SQL>
SQL>
SQL>
SQL> ED
Wrote file afiedt.buf

  1  BEGIN
  2  FOR I IN (SELECT E.EMPNO,E.ENAME,E.JOB,E.SAL,D.DNAME FROM EMP E,DEPT D WHERE
E.DEPTNO=D.DEPTNO)  LOOP
  3  &D(I.EMPNO||'  '||I.ENAME||'   '||I.JOB||'  '||I.SAL||'  '||I.DNAME);
  4  END LOOP;
  5* END;
SQL> /
125  DS     100  SALES

5454  SMITH   CLERK  900  RESEARCH
```

```
7499  ALLEN   SALESMAN  1600  SALES

7521  WARD   SALESMAN  1250  SALES

7566  JONES   MANAGER  2975  RESEARCH

7654  MARTIN   SALESMAN  1250  SALES

7698  BLAKE   MANAGER  2850  SALES

7782  CLARK   MANAGER  2450  ACCOUNTING

7788  SCOTT   ANALYST  3000  RESEARCH

7839  KING   PRESIDENT  5000  ACCOUNTING

7844  TURNER   SALESMAN  1500  SALES

7876  ADAMS   CLERK  1100  RESEARCH

7900  JAMES   CLERK  950  SALES

7902  FORD   ANALYST  45666  RESEARCH

7934  MILLER   CLERK  1300  ACCOUNTING


PL/SQL procedure successfully completed.

SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL> ED
Wrote file afiedt.buf

  1  BEGIN
  2  FOR I IN 1..10  LOOP
  3  &D(I);
  4  END LOOP;
  5* END;
  6  /
1

2

3

4

5

6

7

8
```

```
9

10


PL/SQL procedure successfully completed.

SQL>
SQL>
SQL>
SQL> ED
Wrote file afiedt.buf

  1  BEGIN
  2  FOR I IN 1..5  LOOP
  3  &D('OUTER VALUE..'||I);
  4  ------------------NESTED LOOP-------------------
  5            FOR J IN 1..5 LOOP
  6     &D('INNER VALUE........'||J);
  7            END LOOP;
  8  -------------------END OF NESTED LOOP-------------
  9  END LOOP;
 10* END;
 11  /
OUTER VALUE..1

INNER VALUE........1

INNER VALUE........2

INNER VALUE........3

INNER VALUE........4

INNER VALUE........5

OUTER VALUE..2

INNER VALUE........1

INNER VALUE........2

INNER VALUE........3

INNER VALUE........4

INNER VALUE........5

OUTER VALUE..3

INNER VALUE........1

INNER VALUE........2

INNER VALUE........3

INNER VALUE........4

INNER VALUE........5

OUTER VALUE..4
```

```
INNER VALUE........1

INNER VALUE........2

INNER VALUE........3

INNER VALUE........4

INNER VALUE........5

OUTER VALUE..5

INNER VALUE........1

INNER VALUE........2

INNER VALUE........3

INNER VALUE........4

INNER VALUE........5


PL/SQL procedure successfully completed.

SQL>
SQL>
SQL>
SQL> ED
Wrote file afiedt.buf

  1  BEGIN
  2  FOR I IN 65..70  LOOP
  3             FOR J IN 1..5 LOOP
  4              &D(CHR(I)||J);
  5             END LOOP;
  6  END LOOP;
  7* END;
  8  /
A1

A2

A3

A4

A5

B1

B2

B3

B4

B5

C1

C2
```

```
C3

C4

C5

D1

D2

D3

D4

D5

E1

E2

E3

E4

E5

F1

F2

F3

F4

F5


PL/SQL procedure successfully completed.

SQL> ED
Wrote file afiedt.buf

  1  DECLARE
  2  RES VARCHAR2(500);
  3  BEGIN
  4  FOR I IN 65..70  LOOP
  5             FOR J IN 1..5 LOOP
  6               RES := RES||CHR(I)||J||' ';
  7             END LOOP;
  8  &D(RES);
  9  END LOOP;
 10* END;
 11  /
A1 A2 A3 A4 A5

A1 A2 A3 A4 A5 B1 B2 B3 B4 B5

A1 A2 A3 A4 A5 B1 B2 B3 B4 B5 C1 C2 C3 C4 C5

A1 A2 A3 A4 A5 B1 B2 B3 B4 B5 C1 C2 C3 C4 C5 D1 D2 D3 D4 D5

A1 A2 A3 A4 A5 B1 B2 B3 B4 B5 C1 C2 C3 C4 C5 D1 D2 D3 D4 D5 E1 E2 E3 E4 E5
```

```
A1 A2 A3 A4 A5 B1 B2 B3 B4 B5 C1 C2 C3 C4 C5 D1 D2 D3 D4 D5 E1 E2 E3 E4 E5 F1 F2 F3
F4 F5

PL/SQL procedure successfully completed.

SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL> ED
Wrote file afiedt.buf

  1  DECLARE
  2  RES VARCHAR2(500);
  3  BEGIN
  4  FOR I IN 65..70  LOOP
  5             FOR J IN 1..5 LOOP
  6               RES := RES||CHR(I)||J||' ';
  7             END LOOP;
  8  &D(RES);
  9  RES := '';
 10  END LOOP;
 11* END;
 12  /
A1 A2 A3 A4 A5

B1 B2 B3 B4 B5

C1 C2 C3 C4 C5

D1 D2 D3 D4 D5

E1 E2 E3 E4 E5

F1 F2 F3 F4 F5


PL/SQL procedure successfully completed.

SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
```

```
SQL>
SQL>
SQL> ED
Wrote file afiedt.buf

  1  DECLARE
  2  RES VARCHAR2(500);
  3  BEGIN
  4  FOR I IN 1..5  LOOP
  5             FOR J IN 1..5 LOOP
  6               RES := RES||I*J||' ';
  7             END LOOP;
  8  &D(RES);
  9  RES := '';
 10  END LOOP;
 11* END;
SQL> /
1 2 3 4 5

2 4 6 8 10

3 6 9 12 15

4 8 12 16 20

5 10 15 20 25


PL/SQL procedure successfully completed.

SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL> ED
Wrote file afiedt.buf

  1  DECLARE
  2  RES VARCHAR2(500);
  3  BEGIN
  4  FOR I IN 1..10  LOOP
  5             FOR J IN 1..10 LOOP
  6               RES := RES||I*J||' ';
  7             END LOOP;
  8  &D(RES);
  9  RES := '';
 10  END LOOP;
 11* END;
SQL> /
1 2 3 4 5 6 7 8 9 10
```

```
2 4 6 8 10 12 14 16 18 20

3 6 9 12 15 18 21 24 27 30

4 8 12 16 20 24 28 32 36 40

5 10 15 20 25 30 35 40 45 50

6 12 18 24 30 36 42 48 54 60

7 14 21 28 35 42 49 56 63 70

8 16 24 32 40 48 56 64 72 80

9 18 27 36 45 54 63 72 81 90

10 20 30 40 50 60 70 80 90 100


PL/SQL procedure successfully completed.

SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL> ED
Wrote file afiedt.buf

  1  BEGIN
  2  FOR I IN (SELECT * FROM DEPT)  LOOP
  3  &D(I.DEPTNO||'  '||I.DNAME||'  '||I.LOC);
  4  END LOOP;
  5* END;
  6  /
50  HR  KARACHI

60  NEW HR  LHR

10  ACCOUNTING  NEW YORK

20  RESEARCH  DALLAS

30  SALES  CHICAGO

40  OPERATIONS  BOSTON


PL/SQL procedure successfully completed.

SQL>
SQL>
```

```
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL> ED
Wrote file afiedt.buf

  1  BEGIN
  2  FOR I IN (SELECT * FROM DEPT ORDER BY DEPTNO)  LOOP
  3  &D(I.DEPTNO||'  '||I.DNAME||'  '||I.LOC);
  4  END LOOP;
  5* END;
SQL> /
10  ACCOUNTING  NEW YORK

20  RESEARCH  DALLAS

30  SALES  CHICAGO

40  OPERATIONS  BOSTON

50  HR  KARACHI

60  NEW HR  LHR


PL/SQL procedure successfully completed.

SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL> ED
Wrote file afiedt.buf

  1  BEGIN
  2  FOR I IN (SELECT * FROM DEPT ORDER BY DEPTNO)  LOOP
  3  &D(I.DEPTNO||'  '||I.DNAME||'  '||I.LOC);
  4  &D('==================================='||CHR(10));
  5  ---------------------NESTED BLOCK-----------------
  6      FOR J  IN (SELECT * FROM EMP WHERE DEPTNO=I.DEPTNO) LOOP
  7       &D(J.EMPNO||'  '||J.ENAME||' '||J.JOB||'  '||J.SAL);
  8      END LOOP;
  9  --------------------END OF NESTED BLOCK-------------
 10  END LOOP;
 11* END;
 12  /
10  ACCOUNTING  NEW YORK

===================================
```

```
7782  CLARK MANAGER  2450

7839  KING PRESIDENT  5000

7934  MILLER CLERK  1300

20  RESEARCH  DALLAS

==================================

5454  SMITH CLERK  900

7566  JONES MANAGER  2975

7788  SCOTT ANALYST  3000

7876  ADAMS CLERK  1100

7902  FORD ANALYST  45666

30  SALES  CHICAGO

==================================

125  DS   100

7499  ALLEN SALESMAN  1600

7521  WARD SALESMAN  1250

7654  MARTIN SALESMAN  1250

7698  BLAKE MANAGER  2850

7844  TURNER SALESMAN  1500

7900  JAMES CLERK  950

40  OPERATIONS  BOSTON

==================================

50  HR  KARACHI

==================================

60  NEW HR  LHR

==================================


PL/SQL procedure successfully completed.

SQL> ED
Wrote file afiedt.buf

  1  BEGIN
  2  FOR I IN (SELECT * FROM DEPT ORDER BY DEPTNO)  LOOP
  3  &D(CHR(10)||I.DEPTNO||'  '||I.DNAME||'  '||I.LOC);
  4  &D('==================================='||CHR(10));
  5  ----------------------NESTED BLOCK-----------------
  6      FOR J  IN (SELECT * FROM EMP WHERE DEPTNO=I.DEPTNO) LOOP
  7       &D(J.EMPNO||'  '||J.ENAME||' '||J.JOB||'  '||J.SAL);
```

```
  8      END LOOP;
  9  ---------------------END OF NESTED BLOCK-------------
 10  END LOOP;
 11* END;
SQL> /

10  ACCOUNTING  NEW YORK

==================================

7782  CLARK MANAGER  2450

7839  KING PRESIDENT  5000

7934  MILLER CLERK  1300

20  RESEARCH  DALLAS

==================================

5454  SMITH CLERK  900

7566  JONES MANAGER  2975

7788  SCOTT ANALYST  3000

7876  ADAMS CLERK  1100

7902  FORD ANALYST  45666

30  SALES  CHICAGO

==================================

125  DS   100

7499  ALLEN SALESMAN  1600

7521  WARD SALESMAN  1250

7654  MARTIN SALESMAN  1250

7698  BLAKE MANAGER  2850

7844  TURNER SALESMAN  1500

7900  JAMES CLERK  950

40  OPERATIONS  BOSTON

==================================

50  HR  KARACHI

==================================

60  NEW HR  LHR
```

```
==================================


PL/SQL procedure successfully completed.

SQL> ED
Wrote file afiedt.buf

  1  DECLARE
  2  CNTR NUMBER :=0;
  3  BEGIN
  4  FOR I IN (SELECT * FROM DEPT ORDER BY DEPTNO)  LOOP
  5  &D(CHR(10)||I.DEPTNO||'  '||I.DNAME||'  '||I.LOC);
  6  &D('==================================='||CHR(10));
  7  SELECT COUNT(*) INTO CNTR FROM EMP
  8  WHERE DEPTNO=I.DEPTNO;
  9  IF CNTR >0 THEN
 10  ---------------------NESTED BLOCK------------------
 11      FOR J  IN (SELECT * FROM EMP WHERE DEPTNO=I.DEPTNO) LOOP
 12       &D(J.EMPNO||'  '||J.ENAME||' '||J.JOB||'  '||J.SAL);
 13      END LOOP;
 14  --------------------END OF NESTED BLOCK-------------
 15  ELSE
 16  &D('EMPLOYEES NOT EXIST IN THIS DEPTNO ');
 17  END IF;
 18  END LOOP;
 19* END;
 20  /

10  ACCOUNTING  NEW YORK

==================================

7782  CLARK MANAGER  2450

7839  KING PRESIDENT  5000

7934  MILLER CLERK  1300


20  RESEARCH  DALLAS

==================================

5454  SMITH CLERK  900

7566  JONES MANAGER  2975

7788  SCOTT ANALYST  3000

7876  ADAMS CLERK  1100

7902  FORD ANALYST  45666


30  SALES  CHICAGO

==================================

125  DS   100

7499  ALLEN SALESMAN  1600
```

```
7521  WARD SALESMAN  1250

7654  MARTIN SALESMAN  1250

7698  BLAKE MANAGER  2850

7844  TURNER SALESMAN  1500

7900  JAMES CLERK  950


40  OPERATIONS  BOSTON

==================================

EMPLOYEES NOT EXIST IN THIS DEPTNO


50  HR  KARACHI

==================================

EMPLOYEES NOT EXIST IN THIS DEPTNO


60  NEW HR  LHR

==================================

EMPLOYEES NOT EXIST IN THIS DEPTNO


PL/SQL procedure successfully completed.

SQL> ED
Wrote file afiedt.buf

  1  DECLARE
  2  CNTR NUMBER :=0;
  3  TOT_SAL NUMBER :=0;
  4  BEGIN
  5  FOR I IN (SELECT * FROM DEPT ORDER BY DEPTNO)  LOOP
  6  &D(CHR(10)||I.DEPTNO||'  '||I.DNAME||'  '||I.LOC);
  7  &D('==================================='||CHR(10));
  8  SELECT COUNT(*),SUM(SAL) INTO CNTR,TOT_SAL FROM EMP
  9  WHERE DEPTNO=I.DEPTNO;
 10  IF CNTR >0 THEN
 11  &D('TOTAL PAYMENT OF DEPTNO..'||TOT_SAL);
 12  ----------------------NESTED BLOCK-----------------
 13      FOR J  IN (SELECT * FROM EMP WHERE DEPTNO=I.DEPTNO) LOOP
 14       &D(J.EMPNO||'  '||J.ENAME||' '||J.JOB||'  '||J.SAL);
 15      END LOOP;
 16  ---------------------END OF NESTED BLOCK-------------
 17  ELSE
 18  &D('EMPLOYEES NOT EXIST IN THIS DEPTNO ');
 19  END IF;
 20  END LOOP;
 21* END;
 22  /

10  ACCOUNTING  NEW YORK

==================================
```

TOTAL PAYMENT OF DEPTNO..8750

7782 CLARK MANAGER 2450

7839 KING PRESIDENT 5000

7934 MILLER CLERK 1300

20 RESEARCH DALLAS

==========

TOTAL PAYMENT OF DEPTNO..53641

5454 SMITH CLERK 900

7566 JONES MANAGER 2975

7788 SCOTT ANALYST 3000

7876 ADAMS CLERK 1100

7902 FORD ANALYST 45666

30 SALES CHICAGO

==========

TOTAL PAYMENT OF DEPTNO..9500

125 DS 100

7499 ALLEN SALESMAN 1600

7521 WARD SALESMAN 1250

7654 MARTIN SALESMAN 1250

7698 BLAKE MANAGER 2850

7844 TURNER SALESMAN 1500

7900 JAMES CLERK 950

40 OPERATIONS BOSTON

==========

EMPLOYEES NOT EXIST IN THIS DEPTNO

50 HR KARACHI

==========

EMPLOYEES NOT EXIST IN THIS DEPTNO

60 NEW HR LHR

```
==================================

EMPLOYEES NOT EXIST IN THIS DEPTNO


PL/SQL procedure successfully completed.

SQL> ED
Wrote file afiedt.buf

  1  DECLARE
  2  CNTR NUMBER :=0;
  3  TOT_SAL NUMBER :=0;
  4  EMP_CNTR NUMBER :=0;
  5  BEGIN
  6  FOR I IN (SELECT * FROM DEPT ORDER BY DEPTNO)  LOOP
  7  &D(CHR(10)||I.DEPTNO||'  '||I.DNAME||'  '||I.LOC);
  8  &D('==================================='||CHR(10));
  9  SELECT COUNT(*),SUM(SAL) INTO CNTR,TOT_SAL FROM EMP
 10  WHERE DEPTNO=I.DEPTNO;
 11  IF CNTR >0 THEN
 12  &D('TOTAL PAYMENT OF DEPTNO..'||TOT_SAL);
 13  ----------------------NESTED BLOCK-----------------
 14      FOR J  IN (SELECT * FROM EMP WHERE DEPTNO=I.DEPTNO) LOOP
 15        EMP_CNTR := EMP_CNTR +1;
 16       &D(EMP_CNTR||'  '||J.EMPNO||'  '||J.ENAME||' '||J.JOB||'  '||J.SAL);
 17      END LOOP;
 18  ---------------------END OF NESTED BLOCK-------------
 19  ELSE
 20  &D('EMPLOYEES NOT EXIST IN THIS DEPTNO ');
 21  END IF;
 22  END LOOP;
 23* END;
 24  /

10  ACCOUNTING  NEW YORK

==================================

TOTAL PAYMENT OF DEPTNO..8750

1  7782  CLARK MANAGER  2450

2  7839  KING PRESIDENT  5000

3  7934  MILLER CLERK  1300


20  RESEARCH  DALLAS

==================================

TOTAL PAYMENT OF DEPTNO..53641

4  5454  SMITH CLERK  900

5  7566  JONES MANAGER  2975

6  7788  SCOTT ANALYST  3000

7  7876  ADAMS CLERK  1100
```

```
8  7902  FORD ANALYST  45666

30  SALES  CHICAGO

==================================

TOTAL PAYMENT OF DEPTNO..9500

9  125  DS   100

10  7499  ALLEN SALESMAN  1600

11  7521  WARD SALESMAN  1250

12  7654  MARTIN SALESMAN  1250

13  7698  BLAKE MANAGER  2850

14  7844  TURNER SALESMAN  1500

15  7900  JAMES CLERK  950

40  OPERATIONS  BOSTON

==================================

EMPLOYEES NOT EXIST IN THIS DEPTNO

50  HR  KARACHI

==================================

EMPLOYEES NOT EXIST IN THIS DEPTNO

60  NEW HR  LHR

==================================

EMPLOYEES NOT EXIST IN THIS DEPTNO


PL/SQL procedure successfully completed.

SQL> ED
Wrote file afiedt.buf

  1  DECLARE
  2  CNTR NUMBER :=0;
  3  TOT_SAL NUMBER :=0;
  4  EMP_CNTR NUMBER :=0;
  5  BEGIN
  6  FOR I IN (SELECT * FROM DEPT ORDER BY DEPTNO)  LOOP
  7  &D(CHR(10)||I.DEPTNO||'  '||I.DNAME||'  '||I.LOC);
  8  &D('==================================='||CHR(10));
  9  SELECT COUNT(*),SUM(SAL) INTO CNTR,TOT_SAL FROM EMP
 10  WHERE DEPTNO=I.DEPTNO;
 11  IF CNTR >0 THEN
 12  &D('TOTAL PAYMENT OF DEPTNO..'||TOT_SAL);
 13  ----------------------NESTED BLOCK-----------------
```

```
 14      FOR J  IN (SELECT * FROM EMP WHERE DEPTNO=I.DEPTNO) LOOP
 15        EMP_CNTR := EMP_CNTR +1;
 16       &D(EMP_CNTR||'  '||J.EMPNO||'  '||J.ENAME||' '||J.JOB||'  '||J.SAL);
 17      END LOOP;
 18  ---------------------END OF NESTED BLOCK-------------
 19  EMP_CNTR:=0;
 20  ELSE
 21  &D('EMPLOYEES NOT EXIST IN THIS DEPTNO ');
 22  END IF;
 23  END LOOP;
 24* END;
 25  /

10  ACCOUNTING  NEW YORK

==================================

TOTAL PAYMENT OF DEPTNO..8750

1  7782  CLARK MANAGER  2450

2  7839  KING PRESIDENT  5000

3  7934  MILLER CLERK  1300


20  RESEARCH  DALLAS

==================================

TOTAL PAYMENT OF DEPTNO..53641

1  5454  SMITH CLERK  900

2  7566  JONES MANAGER  2975

3  7788  SCOTT ANALYST  3000

4  7876  ADAMS CLERK  1100

5  7902  FORD ANALYST  45666


30  SALES  CHICAGO

==================================

TOTAL PAYMENT OF DEPTNO..9500

1  125  DS   100

2  7499  ALLEN SALESMAN  1600

3  7521  WARD SALESMAN  1250

4  7654  MARTIN SALESMAN  1250

5  7698  BLAKE MANAGER  2850

6  7844  TURNER SALESMAN  1500

7  7900  JAMES CLERK  950
```

```
40  OPERATIONS  BOSTON

==================================

EMPLOYEES NOT EXIST IN THIS DEPTNO


50  HR  KARACHI

==================================

EMPLOYEES NOT EXIST IN THIS DEPTNO


60  NEW HR  LHR

==================================

EMPLOYEES NOT EXIST IN THIS DEPTNO


PL/SQL procedure successfully completed.

SQL>
SQL>
SQL>
SQL> ED
Wrote file afiedt.buf

  1  DECLARE
  2  CNTR NUMBER :=0;
  3  TOT_SAL NUMBER :=0;
  4  EMP_CNTR NUMBER :=0;
  5  BEGIN
  6  FOR I IN (SELECT * FROM DEPT ORDER BY DEPTNO)  LOOP
  7  &D(CHR(10)||I.DEPTNO||'  '||I.DNAME||'  '||I.LOC);
  8  &D('==================================='||CHR(10));
  9  SELECT COUNT(*),SUM(SAL) INTO CNTR,TOT_SAL FROM EMP
 10  WHERE DEPTNO=I.DEPTNO;
 11  IF CNTR >0 THEN
 12  &D('TOTAL PAYMENT OF DEPTNO..'||TOT_SAL);
 13  ----------------------NESTED BLOCK-----------------
 14      FOR J  IN (SELECT * FROM EMP WHERE DEPTNO=I.DEPTNO) LOOP
 15        EMP_CNTR := EMP_CNTR +1;
 16    &D(EMP_CNTR||'  '||J.EMPNO||'  '||J.ENAME||' '||J.JOB||'  '||J.SAL);
 17      END LOOP;
 18  ---------------------END OF NESTED BLOCK-------------
 19  EMP_CNTR:=0;
 20  ELSE
 21  &D('EMPLOYEES NOT EXIST IN THIS DEPTNO ');
 22  END IF;
 23  END LOOP;
 24* END;
SQL> /

10  ACCOUNTING  NEW YORK

==================================

TOTAL PAYMENT OF DEPTNO..8750

1  7782  CLARK MANAGER  2450
```

2 7839 KING PRESIDENT 5000

3 7934 MILLER CLERK 1300

20 RESEARCH DALLAS

==========

TOTAL PAYMENT OF DEPTNO..53641

1 5454 SMITH CLERK 900

2 7566 JONES MANAGER 2975

3 7788 SCOTT ANALYST 3000

4 7876 ADAMS CLERK 1100

5 7902 FORD ANALYST 45666

30 SALES CHICAGO

==========

TOTAL PAYMENT OF DEPTNO..9500

1 125 DS 100

2 7499 ALLEN SALESMAN 1600

3 7521 WARD SALESMAN 1250

4 7654 MARTIN SALESMAN 1250

5 7698 BLAKE MANAGER 2850

6 7844 TURNER SALESMAN 1500

7 7900 JAMES CLERK 950

40 OPERATIONS BOSTON

==========

EMPLOYEES NOT EXIST IN THIS DEPTNO

50 HR KARACHI

==========

EMPLOYEES NOT EXIST IN THIS DEPTNO

60 NEW HR LHR

==========

EMPLOYEES NOT EXIST IN THIS DEPTNO

```
PL/SQL procedure successfully completed.

SQL>
SQL>
SQL>
SQL> SPOOL OFF
```

```
SQL>
SQL>
SQL> DECLARE
  2  .
SQL> ED
Wrote file afiedt.buf

  1  DECLARE
  2  A NUMBER :=-1;
  3  BEGIN
  4  IF A>10 THEN
  5  EXIT;
  6  ELSE
  7  A :=A +1;
  8  &D(A);
  9  END IF;
 10* END;
 11  /
EXIT;
*
ERROR at line 5:
ORA-06550: line 5, column 1:
PLS-00376: illegal EXIT statement; it must appear inside a loop
ORA-06550: line 5, column 1:
PL/SQL: Statement ignored


SQL> ED
Wrote file afiedt.buf

  1  DECLARE
  2  A NUMBER :=-1;
  3  BEGIN
  4  LOOP
  5  IF A>10 THEN
  6  EXIT;
  7  ELSE
  8  A :=A +1;
  9  &D(A);
 10  END IF;
 11  END LOOP;
 12* END;
 13  /
0

1

2

3

4

5

6

7

8

9
```

```
10

11


PL/SQL procedure successfully completed.

SQL>
SQL>
SQL>
SQL> ED
Wrote file afiedt.buf

  1  DECLARE
  2  A NUMBER :=0;
  3  BEGIN
  4  LOOP
  5  IF A>10 THEN
  6  EXIT;
  7  ELSE
  8  A :=A +1;
  9  &D(A);
 10  END IF;
 11  END LOOP;
 12* END;
SQL> /
1

2

3

4

5

6

7

8

9

10

11


PL/SQL procedure successfully completed.

SQL> ED
Wrote file afiedt.buf

  1  DECLARE
  2  A NUMBER :=0;
  3  BEGIN
  4  LOOP
  5  IF A>10 THEN
  6  EXIT;
  7  ELSE
  8  &D(A);
  9  A :=A +1;
```

```
 10  END IF;
 11  END LOOP;
 12* END;
 13  /
0

1

2

3

4

5

6

7

8

9

10


PL/SQL procedure successfully completed.

SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL> ED
Wrote file afiedt.buf

  1  DECLARE
  2  A NUMBER :=0;
  3  BEGIN
  4  LOOP
  5  EXIT WHEN A>10
  6  &D(A);
  7  A :=A +1;
  8  END LOOP;
  9* END;
 10  /
DBMS_OUTPUT.PUT_LINE(A);
*
ERROR at line 6:
ORA-06550: line 6, column 1:
PLS-00103: Encountered the symbol "DBMS_OUTPUT" when expecting one of the following:

* & - + ; / at mod remainder rem <an exponent (**)> and or ||
multiset
The symbol "*" was substituted for "DBMS_OUTPUT" to continue.


SQL>
SQL>
SQL>
SQL>
```

```
SQL> ED
Wrote file afiedt.buf

  1  DECLARE
  2  A NUMBER :=0;
  3  BEGIN
  4  LOOP
  5  EXIT WHEN A>10;
  6  &D(A);
  7  A :=A +1;
  8  END LOOP;
  9* END;
SQL> /
0

1

2

3

4

5

6

7

8

9

10


PL/SQL procedure successfully completed.

SQL>
SQL>
SQL> ED
Wrote file afiedt.buf

  1  DECLARE
  2  A NUMBER :=0;
  3  BEGIN
  4  LOOP
  5  EXIT WHEN A>=10
  6  &D(A);
  7  A :=A +1;
  8  END LOOP;
  9* END;
SQL> /
DBMS_OUTPUT.PUT_LINE(A);
*
ERROR at line 6:
ORA-06550: line 6, column 1:
PLS-00103: Encountered the symbol "DBMS_OUTPUT" when expecting one of the following:

* & - + ; / at mod remainder rem <an exponent (**)> and or ||
multiset
The symbol "*" was substituted for "DBMS_OUTPUT" to continue.
```

```
SQL>
SQL>
SQL>
SQL> ED
Wrote file afiedt.buf

  1  DECLARE
  2  A NUMBER :=0;
  3  BEGIN
  4  LOOP
  5  EXIT WHEN A>=10;
  6  &D(A);
  7  A :=A +1;
  8  END LOOP;
  9* END;
SQL> /
0

1

2

3

4

5

6

7

8

9


PL/SQL procedure successfully completed.

SQL>
SQL>
SQL>
SQL>
SQL> ED
Wrote file afiedt.buf

  1  DECLARE
  2  A NUMBER :=0;
  3  BEGIN
  4  LOOP
  5  &D(A);
  6  A :=A +1;
  7  EXIT WHEN A>=10;
  8  END LOOP;
  9* END;
 10  /
0

1

2
```

```
3

4

5

6

7

8

9


PL/SQL procedure successfully completed.

SQL> ED
Wrote file afiedt.buf

  1  DECLARE
  2  A NUMBER :=0;
  3  BEGIN
  4  LOOP
  5  &D(A);
  6  EXIT WHEN A>=10;
  7  A :=A +1;
  8  END LOOP;
  9* END;
 10  /
0

1

2

3

4

5

6

7

8

9

10


PL/SQL procedure successfully completed.

SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL> ED
```

```
Wrote file afiedt.buf

SP2-0223: No lines in SQL buffer.
SQL> ED
SP2-0107: Nothing to save.
SQL> DECLARE
  2  .
SQL> ED
Wrote file afiedt.buf

  1  DECLARE
  2  A NUMBER :=0;
  3  TEST BOOLEAN;
  4  BEGIN
  5  LOOP
  6  &D(A||TEST);
  7  TEST := A>10;
  8  EXIT WHEN TEST;
  9  A :=A +1;
 10  END LOOP;
 11* END;
 12  /
DBMS_OUTPUT.PUT_LINE(A||TEST);
                     *
ERROR at line 6:
ORA-06550: line 6, column 22:
PLS-00306: wrong number or types of arguments in call to '||'
ORA-06550: line 6, column 1:
PL/SQL: Statement ignored


SQL>
SQL>
SQL>
SQL> ED
Wrote file afiedt.buf

  1  DECLARE
  2  A NUMBER :=0;
  3  TEST BOOLEAN;
  4  BEGIN
  5  LOOP
  6  &D(A||'  '||TEST);
  7  TEST := A>10;
  8  EXIT WHEN TEST;
  9  A :=A +1;
 10  END LOOP;
 11* END;
SQL> /
DBMS_OUTPUT.PUT_LINE(A||'  '||TEST);
                     *
ERROR at line 6:
ORA-06550: line 6, column 22:
PLS-00306: wrong number or types of arguments in call to '||'
ORA-06550: line 6, column 1:
PL/SQL: Statement ignored


SQL> ED
Wrote file afiedt.buf

  1  DECLARE
  2  A NUMBER :=&STARTING_VAL;
```

```
  3  TEST BOOLEAN;
  4  BEGIN
  5  LOOP
  6  &D(A);
  7  TEST := A>10;
  8  EXIT WHEN TEST;
  9  A :=A +1;
 10  END LOOP;
 11* END;
SQL> /
Enter value for starting_val: 5
5

6

7

8

9

10

11


PL/SQL procedure successfully completed.

SQL> ED
Wrote file afiedt.buf

  1  DECLARE
  2  A NUMBER :=&STARTING_VAL;
  3  END_NO NUMBER := &ENDING_VAL;
  4  TEST BOOLEAN;
  5  BEGIN
  6  LOOP
  7  &D(A);
  8  TEST := A>END_NO;
  9  EXIT WHEN TEST;
 10  A :=A +1;
 11  END LOOP;
 12* END;
SQL> /
Enter value for starting_val: 5
Enter value for ending_val: 11
5

6

7

8

9

10

11

12
```

```
PL/SQL procedure successfully completed.

SQL> ED
Wrote file afiedt.buf

  1  DECLARE
  2  T_NO NUMBER :=&TABLE_NO;
  3  A NUMBER :=&STARTING_VAL;
  4  END_NO NUMBER := &ENDING_VAL;
  5  TEST BOOLEAN;
  6  CNTR NUMBER :=0;
  7  BEGIN
  8  LOOP
  9  CNTR := T_NO * A;
 10  &D(T_NO||' X '||A||' = '||CNTR);
 11  A :=A +1;
 12  TEST := A>END_NO;
 13  EXIT WHEN TEST;
 14  END LOOP;
 15* END;
 16  /
Enter value for table_no: 5
Enter value for starting_val: 1
Enter value for ending_val: 10
5 X 1 = 5

5 X 2 = 10

5 X 3 = 15

5 X 4 = 20

5 X 5 = 25

5 X 6 = 30

5 X 7 = 35

5 X 8 = 40

5 X 9 = 45

5 X 10 = 50


PL/SQL procedure successfully completed.

SQL> ED
Wrote file afiedt.buf

  1  BEGIN
  2  FOR I IN 1..5 LOOP
  3  &D(I);
  4  END LOOP;
  5* END;
  6  /
1

2

3

4
```

```
5


PL/SQL procedure successfully completed.

SQL> ED
Wrote file afiedt.buf

  1  BEGIN
  2  FOR I IN 1..5 LOOP
  3  &D(I);
  4  END LOOP;
  5  &D(I);
  6* END;
  7  /
DBMS_OUTPUT.PUT_LINE(I);
                     *
ERROR at line 5:
ORA-06550: line 5, column 22:
PLS-00201: identifier 'I' must be declared
ORA-06550: line 5, column 1:
PL/SQL: Statement ignored


SQL>
SQL>
SQL>
SQL> DECLARE
  2  T_NO NUMBER :=&TABLE_NO;
  3  A NUMBER :=&STARTING_VAL;
  4  END_NO NUMBER := &ENDING_VAL;
  5  TEST BOOLEAN;
  6  CNTR NUMBER :=0;
  7  BEGIN
  8  LOOP
  9  CNTR := T_NO * A;
 10  &D(T_NO||' X '||A||' = '||CNTR);
 11  A :=A +1;
 12  TEST := A>END_NO;
 13  EXIT WHEN TEST;
 14  END LOOP;
 15  END;
 16  .
SQL> ED
Wrote file afiedt.buf

  1  DECLARE
  2  T_NO NUMBER :=&TABLE_NO;
  3  A NUMBER :=&STARTING_VAL;
  4  END_NO NUMBER := &ENDING_VAL;
  5  TEST BOOLEAN;
  6  CNTR NUMBER :=0;
  7  BEGIN
  8  LOOP
  9  CNTR := T_NO * A;
 10  &D(T_NO||' X '||A||' = '||CNTR);
 11  A :=A +1;
 12  TEST := A>END_NO;
 13  EXIT WHEN TEST;
 14  END LOOP;
 15  &D(CNTR);
 16* END;
```

```
 17  /
Enter value for table_no:
Enter value for starting_val:
Enter value for ending_val:
T_NO NUMBER :=;
             *
ERROR at line 2:
ORA-06550: line 2, column 15:
PLS-00103: Encountered the symbol ";" when expecting one of the following:
( - + case mod new not null <an identifier>
<a double-quoted delimited-identifier> <a bind variable> avg
count current exists max min prior sql stddev sum variance
execute forall merge time timestamp interval date
<a string literal with character set specification>
<a number> <a single-quoted SQL string> pipe
<an alternatively-quoted string literal with character set specification>
<an alternatively-quoted S


SQL>
SQL> /
Enter value for table_no: 5
Enter value for starting_val: 1
Enter value for ending_val: 10
5 X 1 = 5

5 X 2 = 10

5 X 3 = 15

5 X 4 = 20

5 X 5 = 25

5 X 6 = 30

5 X 7 = 35

5 X 8 = 40

5 X 9 = 45

5 X 10 = 50

50


PL/SQL procedure successfully completed.

SQL> ED
Wrote file afiedt.buf

  1  DECLARE
  2  A NUMBER :=0;
  3  BEGIN
  4  WHILE A<=10 LOOP
  5  &D(A);
  6  A := A+1;
  7  END LOOP;
  8* END;
  9  /
0
```

```
1

2

3

4

5

6

7

8

9

10


PL/SQL procedure successfully completed.

SQL>
SQL>
SQL>
SQL> ED
Wrote file afiedt.buf

  1  DECLARE
  2  A NUMBER :=0;
  3  SQ NUMBER :=0;
  4  CB NUMBER :=0;
  5  BEGIN
  6  WHILE A<=10 LOOP
  7  SQ := A * A;
  8  CB := SQ * A;
  9  &D(A||'  '||SQ||'  '||CB);
 10  A := A+1;
 11  END LOOP;
 12* END;
 13  /
0  0  0

1  1  1

2  4  8

3  9  27

4  16  64

5  25  125

6  36  216

7  49  343

8  64  512

9  81  729

10  100  1000
```

```
PL/SQL procedure successfully completed.

SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL> ED
Wrote file afiedt.buf

  1  DECLARE
  2  A NUMBER :=0;
  3  CNTR NUMBER :=0;
  4  BEGIN
  5  WHILE A<=10 LOOP
  6  CNTR := CNTR +A;
  7  &D(A);
  8  A := A+1;
  9  END LOOP;
 10  &D('SUM OF NUMBER IS ...'||CNTR);
 11* END;
 12  /
0

1

2

3

4

5

6

7

8

9

10

SUM OF NUMBER IS ...55


PL/SQL procedure successfully completed.

SQL> ED
Wrote file afiedt.buf

  1  DECLARE
  2  A NUMBER :=0;
  3  CNTR NUMBER :=10;
  4  BEGIN
  5  WHILE A<=10 LOOP
```

```
  6  &D(A||'  '||CNTR );
  7  A := A+1;
  8  CNTR := CNTR -1;
  9  END LOOP;
 10  &D('SUM OF NUMBER IS ...'||CNTR);
 11* END;
 12  /
0  10

1  9

2  8

3  7

4  6

5  5

6  4

7  3

8  2

9  1

10  0

SUM OF NUMBER IS ...-1


PL/SQL procedure successfully completed.

SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL> ED
Wrote file afiedt.buf

  1  DECLARE
  2  CURSOR MY_CURSOR IS SELECT * FROM EMP;
  3  BEGIN
  4  FOR I IN MY_CURSOR LOOP
  5  &D(I.EMPNO||'  '||I.ENAME||'  '||I.JOB||'  '||I.SAL||'  '||I.DEPTNO);
  6  END LOOP;
  7* END;
  8  /
125  DS    100  30

5454  SMITH  CLERK  900  20

7499  ALLEN  SALESMAN  1600  30

7521  WARD  SALESMAN  1250  30
```

```
7566  JONES  MANAGER  2975  20

7654  MARTIN  SALESMAN  1250  30

7698  BLAKE  MANAGER  2850  30

7782  CLARK  MANAGER  2450  10

7788  SCOTT  ANALYST  3000  20

7839  KING  PRESIDENT  5000  10

7844  TURNER  SALESMAN  1500  30

7876  ADAMS  CLERK  1100  20

7900  JAMES  CLERK  950  30

7902  FORD  ANALYST  45666  20

7934  MILLER  CLERK  1300  10


PL/SQL procedure successfully completed.

SQL>
SQL>
SQL>
SQL>
SQL> ED
Wrote file afiedt.buf

  1  DECLARE
  2  CURSOR MY_CURSOR IS SELECT * FROM EMP;
  3  BEGIN
  4  FOR I IN MY_CURSOR LOOP
  5  &D(I.EMPNO||'  '||I.ENAME||'  '||I.JOB||'  '||I.SAL||'  '||I.DEPTNO);
  6  END LOOP;
  7  FOR I IN MY_CURSOR LOOP
  8  &D(I.EMPNO||'  '||I.ENAME||'  '||I.JOB||'  '||I.SAL||'  '||I.DEPTNO);
  9  END LOOP;
 10* END;
 11  /
125  DS    100  30

5454  SMITH  CLERK  900  20

7499  ALLEN  SALESMAN  1600  30

7521  WARD  SALESMAN  1250  30

7566  JONES  MANAGER  2975  20

7654  MARTIN  SALESMAN  1250  30

7698  BLAKE  MANAGER  2850  30

7782  CLARK  MANAGER  2450  10

7788  SCOTT  ANALYST  3000  20

7839  KING  PRESIDENT  5000  10
```

```
7844  TURNER  SALESMAN  1500  30

7876  ADAMS  CLERK  1100  20

7900  JAMES  CLERK  950  30

7902  FORD  ANALYST  45666  20

7934  MILLER  CLERK  1300  10

125  DS    100  30

5454  SMITH  CLERK  900  20

7499  ALLEN  SALESMAN  1600  30

7521  WARD  SALESMAN  1250  30

7566  JONES  MANAGER  2975  20

7654  MARTIN  SALESMAN  1250  30

7698  BLAKE  MANAGER  2850  30

7782  CLARK  MANAGER  2450  10

7788  SCOTT  ANALYST  3000  20

7839  KING  PRESIDENT  5000  10

7844  TURNER  SALESMAN  1500  30

7876  ADAMS  CLERK  1100  20

7900  JAMES  CLERK  950  30

7902  FORD  ANALYST  45666  20

7934  MILLER  CLERK  1300  10


PL/SQL procedure successfully completed.

SQL> ED
Wrote file afiedt.buf

  1  DECLARE
  2  CURSOR MY_CURSOR IS SELECT * FROM EMP;
  3  BEGIN
  4  FOR I IN (SELECT * FROM EMP) LOOP
  5  &D(I.EMPNO||'  '||I.ENAME||'  '||I.JOB||'  '||I.SAL||'  '||I.DEPTNO);
  6  END LOOP;
  7  &D('AFTER 1ST LOOP'||CHR(10));
  8  FOR I IN (SELECT * FROM EMP) LOOP
  9  &D(I.EMPNO||'  '||I.ENAME||'  '||I.JOB||'  '||I.SAL||'  '||I.DEPTNO);
 10  END LOOP;
 11* END;
 12  /
125  DS    100  30

5454  SMITH  CLERK  900  20

7499  ALLEN  SALESMAN  1600  30
```

```
7521  WARD  SALESMAN  1250  30
7566  JONES  MANAGER  2975  20
7654  MARTIN  SALESMAN  1250  30
7698  BLAKE  MANAGER  2850  30
7782  CLARK  MANAGER  2450  10
7788  SCOTT  ANALYST  3000  20
7839  KING  PRESIDENT  5000  10
7844  TURNER  SALESMAN  1500  30
7876  ADAMS  CLERK  1100  20
7900  JAMES  CLERK  950  30
7902  FORD  ANALYST  45666  20
7934  MILLER  CLERK  1300  10
AFTER 1ST LOOP

125  DS    100  30
5454  SMITH  CLERK  900  20
7499  ALLEN  SALESMAN  1600  30
7521  WARD  SALESMAN  1250  30
7566  JONES  MANAGER  2975  20
7654  MARTIN  SALESMAN  1250  30
7698  BLAKE  MANAGER  2850  30
7782  CLARK  MANAGER  2450  10
7788  SCOTT  ANALYST  3000  20
7839  KING  PRESIDENT  5000  10
7844  TURNER  SALESMAN  1500  30
7876  ADAMS  CLERK  1100  20
7900  JAMES  CLERK  950  30
7902  FORD  ANALYST  45666  20
7934  MILLER  CLERK  1300  10

PL/SQL procedure successfully completed.

SQL> ED
Wrote file afiedt.buf
```

```
  1  DECLARE
  2  CURSOR MY_CURSOR IS SELECT * FROM EMP;
  3  BEGIN
  4  FOR I IN MY_CURSOR LOOP
  5  &D(I.EMPNO||'  '||I.ENAME||'  '||I.JOB||'  '||I.SAL||'  '||I.DEPTNO);
  6  END LOOP;
  7  &D('AFTER 1ST LOOP'||CHR(10));
  8  FOR I IN MY_CURSOR LOOP
  9  &D(I.EMPNO||'  '||I.ENAME||'  '||I.JOB||'  '||I.SAL||'  '||I.DEPTNO);
 10  END LOOP;
 11* END;
SQL> /
125  DS    100  30

5454  SMITH  CLERK  900  20

7499  ALLEN  SALESMAN  1600  30

7521  WARD  SALESMAN  1250  30

7566  JONES  MANAGER  2975  20

7654  MARTIN  SALESMAN  1250  30

7698  BLAKE  MANAGER  2850  30

7782  CLARK  MANAGER  2450  10

7788  SCOTT  ANALYST  3000  20

7839  KING  PRESIDENT  5000  10

7844  TURNER  SALESMAN  1500  30

7876  ADAMS  CLERK  1100  20

7900  JAMES  CLERK  950  30

7902  FORD  ANALYST  45666  20

7934  MILLER  CLERK  1300  10

AFTER 1ST LOOP

125  DS    100  30

5454  SMITH  CLERK  900  20

7499  ALLEN  SALESMAN  1600  30

7521  WARD  SALESMAN  1250  30

7566  JONES  MANAGER  2975  20

7654  MARTIN  SALESMAN  1250  30

7698  BLAKE  MANAGER  2850  30

7782  CLARK  MANAGER  2450  10

7788  SCOTT  ANALYST  3000  20
```

```
7839  KING  PRESIDENT  5000  10

7844  TURNER  SALESMAN  1500  30

7876  ADAMS  CLERK  1100  20

7900  JAMES  CLERK  950  30

7902  FORD  ANALYST  45666  20

7934  MILLER  CLERK  1300  10


PL/SQL procedure successfully completed.

SQL> ED
Wrote file afiedt.buf

  1  DECLARE
  2  CURSOR MY_CURSOR IS SELECT * FROM EMP;
  3  BEGIN
  4  FOR I IN MY_CURSOR LOOP
  5  &D(I.EMPNO||'  '||I.ENAME||'  '||I.JOB||'  '||I.SAL||'  '||I.DEPTNO);
  6  END LOOP;
  7  &D('AFTER 1ST LOOP'||CHR(10));
  8  FOR I IN MY_CURSOR LOOP
  9  &D(I.EMPNO||'  '||I.ENAME||'  '||I.JOB||'  '||I.SAL||'  '||I.DEPTNO);
 10  END LOOP;
 11* END;
SQL> ..
SP2-0042: unknown command ".." - rest of line ignored.
SQL>
SQL>
SQL> SELECT * FROM EMP
  2  WHERE DEPTNO=10;

     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-85       1300
10

SQL> ED
Wrote file afiedt.buf

  1  SELECT * FROM EMP
  2* WHERE DEPTNO=10
SQL>
SQL> ED
Wrote file afiedt.buf

  1  SELECT DEPTNO,SAL FROM EMP
  2* WHERE DEPTNO=10
SQL> /

    DEPTNO        SAL
```

```
---------- ----------

        10       2450

        10       5000

        10       1300


SQL> ED
Wrote file afiedt.buf

  1  UPDATE EMP
  2  SET SAL =SAL +1000
  3* WHERE DEPTNO=10
SQL> /

3 rows updated.

SQL>
SQL>
SQL> SELECT DEPTNO,SAL FROM EMP
  2  WHERE DEPTNO=10
  3  ;

    DEPTNO        SAL

---------- ----------

        10       3450

        10       6000

        10       2300


SQL>
SQL>
SQL>
SQL> ROLLBACK;

Rollback complete.

SQL>
SQL>
SQL> ED
Wrote file afiedt.buf

  1* ROLLBACK
SQL>
SQL> SELECT DEPTNO,SAL FROM EMP
  2  WHERE DEPTNO=10
  3  ;

    DEPTNO        SAL

---------- ----------

        10       2450

        10       5000

        10       1300
```

```
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL> SELECT DEPTNO,SAL FROM FROM
  2  WHERE SAL =950;
SELECT DEPTNO,SAL FROM FROM
                       *
ERROR at line 1:
ORA-00903: invalid table name


SQL>
SQL>
SQL>
SQL> ED
Wrote file afiedt.buf

  1  SELECT ENAME,DEPTNO,SAL FROM EMP
  2* WHERE SAL =950
SQL> /

ENAME           DEPTNO        SAL

---------- ---------- ----------

JAMES               30        950


SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL> ED
Wrote file afiedt.buf

  1  SELECT ENAME,DEPTNO,SAL FROM EMP
  2* WHERE SAL =950
SQL>
SQL> ED
Wrote file afiedt.buf

  1  UPDATE  EMP
  2  SET SAL =1000
  3* WHERE SAL =950
SQL> /

1 row updated.

SQL> ED
Wrote file afiedt.buf
```

```
  1  UPDATE  EMP
  2  SET SAL =1000
  3* WHERE SAL =950
SQL>
SQL> COMMIT;

Commit complete.

SQL>
SQL>
SQL>
SQL>
SQL>  DECLARE
  2   CURSOR MY_CURSOR IS SELECT * FROM EMP;
  3   BEGIN
  4   FOR I IN MY_CURSOR LOOP
  5   &D(I.EMPNO||'  '||I.ENAME||'  '||I.JOB||'  '||I.SAL||'  '||I.DEPTNO);
  6   END LOOP;
  7   &D('AFTER 1ST LOOP'||CHR(10));
  8   FOR I IN MY_CURSOR LOOP
  9   &D(I.EMPNO||'  '||I.ENAME||'  '||I.JOB||'  '||I.SAL||'  '||I.DEPTNO);
 10   END LOOP;
 11   END;
 12  .
SQL> ED
Wrote file afiedt.buf

  1   DECLARE
  2   CURSOR MY_CURSOR IS SELECT * FROM EMP;
  3   BEGIN
  4   FOR I IN MY_CURSOR LOOP
  5   &D(I.EMPNO||'  '||I.ENAME||'  '||I.JOB||'  '||I.SAL||'  '||I.DEPTNO);
  6   END LOOP;
  7*  END;
  8  /
125  DS    100  30

5454  SMITH  CLERK  900  20

7499  ALLEN  SALESMAN  1600  30

7521  WARD  SALESMAN  1250  30

7566  JONES  MANAGER  2975  20

7654  MARTIN  SALESMAN  1250  30

7698  BLAKE  MANAGER  2850  30

7782  CLARK  MANAGER  2450  10

7788  SCOTT  ANALYST  3000  20

7839  KING  PRESIDENT  5000  10

7844  TURNER  SALESMAN  1500  30

7876  ADAMS  CLERK  1100  20

7900  JAMES  CLERK  1000  30

7902  FORD  ANALYST  45666  20
```

```
7934  MILLER  CLERK  1300  10


PL/SQL procedure successfully completed.

SQL>
SQL>
SQL>
SQL> ED
Wrote file afiedt.buf

  1   DECLARE
  2   CURSOR MY_CURSOR IS SELECT * FROM EMP;
  3    C1 MY_CURSOR%ROWTYPE;
  4   BEGIN
  5    OPEN MY_CURSOR;
  6  LOOP
  7    FETCH MY_CURSOR INTO C1;
  8  EXIT WHEN MY_CURSOR%NOTFOUND;
  9   &D(C1.EMPNO||'  '||C1.ENAME||'  '||C1.JOB||'  '||C1.SAL||'  '||C1.DEPTNO);
 10  END LOOP;
 11*  END;
 12  /
125  DS    100  30

5454  SMITH  CLERK  900  20

7499  ALLEN  SALESMAN  1600  30

7521  WARD  SALESMAN  1250  30

7566  JONES  MANAGER  2975  20

7654  MARTIN  SALESMAN  1250  30

7698  BLAKE  MANAGER  2850  30

7782  CLARK  MANAGER  2450  10

7788  SCOTT  ANALYST  3000  20

7839  KING  PRESIDENT  5000  10

7844  TURNER  SALESMAN  1500  30

7876  ADAMS  CLERK  1100  20

7900  JAMES  CLERK  1000  30

7902  FORD  ANALYST  45666  20

7934  MILLER  CLERK  1300  10


PL/SQL procedure successfully completed.

SQL>
SQL>
SQL> ED
Wrote file afiedt.buf

  1   DECLARE
  2   CURSOR MY_CURSOR IS SELECT * FROM EMP;
```

```
  3    C1 MY_CURSOR%ROWTYPE;
  4   BEGIN
  5    OPEN MY_CURSOR;
  6  LOOP
  7    FETCH MY_CURSOR INTO C1;
  8  EXIT WHEN MY_CURSOR%NOTFOUND;
  9   &D(C1.EMPNO||'  '||C1.ENAME||'  '||C1.JOB||'  '||C1.SAL||'  '||C1.DEPTNO);
 10  END LOOP;
 11  CLOSE    MY_CURSOR;
 12*  END;
 13  /
125  DS    100  30

5454  SMITH  CLERK  900  20

7499  ALLEN  SALESMAN  1600  30

7521  WARD  SALESMAN  1250  30

7566  JONES  MANAGER  2975  20

7654  MARTIN  SALESMAN  1250  30

7698  BLAKE  MANAGER  2850  30

7782  CLARK  MANAGER  2450  10

7788  SCOTT  ANALYST  3000  20

7839  KING  PRESIDENT  5000  10

7844  TURNER  SALESMAN  1500  30

7876  ADAMS  CLERK  1100  20

7900  JAMES  CLERK  1000  30

7902  FORD  ANALYST  45666  20

7934  MILLER  CLERK  1300  10


PL/SQL procedure successfully completed.

SQL>
SQL>
SQL>
SQL> ED
Wrote file afiedt.buf

  1   DECLARE
  2   CURSOR MY_CURSOR IS SELECT * FROM EMP WHERE DEPTNO=&DEPTNO
  3    C1 MY_CURSOR%ROWTYPE;
  4   BEGIN
  5    OPEN MY_CURSOR;
  6  LOOP
  7    FETCH MY_CURSOR INTO C1;
  8  EXIT WHEN MY_CURSOR%NOTFOUND;
  9   &D(C1.EMPNO||'  '||C1.ENAME||'  '||C1.JOB||'  '||C1.SAL||'  '||C1.DEPTNO);
 10  END LOOP;
 11  CLOSE    MY_CURSOR;
 12*  END;
SQL> /
```

```
Enter value for deptno: 10
  C1 MY_CURSOR%ROWTYPE;
  *
ERROR at line 3:
ORA-06550: line 3, column 3:
PL/SQL: ORA-00933: SQL command not properly ended
ORA-06550: line 2, column 22:
PL/SQL: SQL Statement ignored


SQL> ED
Wrote file afiedt.buf

  1   DECLARE
  2   CURSOR MY_CURSOR IS SELECT * FROM EMP WHERE DEPTNO=&DEPTNO;
  3    C1 MY_CURSOR%ROWTYPE;
  4   BEGIN
  5    OPEN MY_CURSOR;
  6  LOOP
  7    FETCH MY_CURSOR INTO C1;
  8  EXIT WHEN MY_CURSOR%NOTFOUND;
  9   &D(C1.EMPNO||'  '||C1.ENAME||'  '||C1.JOB||'  '||C1.SAL||'  '||C1.DEPTNO);
 10  END LOOP;
 11  CLOSE    MY_CURSOR;
 12*  END;
SQL> /
Enter value for deptno: 10
7782  CLARK  MANAGER  2450  10

7839  KING  PRESIDENT  5000  10

7934  MILLER  CLERK  1300  10


PL/SQL procedure successfully completed.

SQL> /
Enter value for deptno: 30
125  DS    100  30

7499  ALLEN  SALESMAN  1600  30

7521  WARD  SALESMAN  1250  30

7654  MARTIN  SALESMAN  1250  30

7698  BLAKE  MANAGER  2850  30

7844  TURNER  SALESMAN  1500  30

7900  JAMES  CLERK  1000  30


PL/SQL procedure successfully completed.

SQL> SPOOL OFF
```

```
SQL>
SQL>
SQL>
SQL> DECLARE
  2  .
SQL> ED
Wrote file afiedt.buf

  1  DECLARE
  2  V_EMPNO NUMBER :=&EMPNO;
  3  EMP_REC EMP%ROWTYPE;
  4  BEGIN
  5  SELECT * INTO EMP_REC FROM SCOTT.EMP
  6  WHERE EMPNO=V_EMPNO;
  7  &D(EMP_REC.ENAME ||' '||EMP_REC.JOB||'  '||EMP_REC.SAL||'  '||EMP_REC.DEPTNO);
  8* END;
  9  /
Enter value for empno: 7788
SCOTT ANALYST  3000  20


PL/SQL procedure successfully completed.

SQL>
SQL>
SQL>
SQL>
SQL> ED
Wrote file afiedt.buf

  1  DECLARE
  2  V_EMPNO NUMBER :=&EMPNO;
  3  EMP_REC EMP%ROWTYPE;
  4  BEGIN
  5  SELECT * INTO EMP_REC FROM SCOTT.EMP
  6  WHERE EMPNO=V_EMPNO;
  7  &D(EMP_REC.ENAME ||' '||EMP_REC.JOB||'  '||EMP_REC.SAL||'  '||EMP_REC.DEPTNO);
  8* END;
SQL>
SQL> /
Enter value for empno: 7839
KING PRESIDENT  5000  10


PL/SQL procedure successfully completed.

SQL> /
Enter value for empno: 4578
DECLARE
*
ERROR at line 1:
ORA-01403: no data found
ORA-06512: at line 5


SQL>
SQL>
SQL>
SQL> ED
Wrote file afiedt.buf

  1  DECLARE
  2  V_EMPNO NUMBER :=&EMPNO;
```

```
  3  EMP_REC EMP%ROWTYPE;
  4  BEGIN
  5  SELECT * INTO EMP_REC FROM SCOTT.EMP
  6  WHERE EMPNO=V_EMPNO;
  7  &D(EMP_REC.ENAME ||' '||EMP_REC.JOB||'  '||EMP_REC.SAL||'  '||EMP_REC.DEPTNO);
  8  EXCEPTION
  9    WHEN NO_DATA_FOUND THEN
 10     &D('RECORD NOT EXIST ...');
 11* END;
 12  /
Enter value for empno: 4578
RECORD NOT EXIST ...


PL/SQL procedure successfully completed.

SQL>
SQL>
SQL> ED
Wrote file afiedt.buf

  1  DECLARE
  2  V_EMPNO NUMBER :=&EMPNO;
  3  EMP_REC EMP%ROWTYPE;
  4  BEGIN
  5  SELECT * INTO EMP_REC FROM SCOTT.EMP
  6  WHERE EMPNO=V_EMPNO;
  7  &D(EMP_REC.ENAME ||' '||EMP_REC.JOB||'  '||EMP_REC.SAL||'  '||EMP_REC.DEPTNO);
  8  EXCEPTION
  9    WHEN NO_DATA_FOUND THEN
 10     &D('RECORD NOT EXIST ...');
 11    WHEN OTHERS THEN
 12  &D(SQLCODE||'   '||SQLERRM);
 13* END;
 14  /
Enter value for empno: 45786
RECORD NOT EXIST ...


PL/SQL procedure successfully completed.

SQL>
SQL>
SQL>
SQL> ED
Wrote file afiedt.buf

  1  DECLARE
  2  V_EMPNO EMP.EMPNO%TYPE :=&EMPNO;
  3  EMP_REC EMP%ROWTYPE;
  4  BEGIN
  5  SELECT * INTO EMP_REC FROM SCOTT.EMP
  6  WHERE EMPNO=V_EMPNO;
  7  &D(EMP_REC.ENAME ||' '||EMP_REC.JOB||'  '||EMP_REC.SAL||'  '||EMP_REC.DEPTNO);
  8  EXCEPTION
  9    WHEN NO_DATA_FOUND THEN
 10     &D('RECORD NOT EXIST ...');
 11    WHEN OTHERS THEN
 12  &D(SQLCODE||'   '||SQLERRM);
 13* END;
SQL> /
Enter value for empno: 45789
DECLARE
```

```
*
ERROR at line 1:
ORA-06502: PL/SQL: numeric or value error: number precision too large
ORA-06512: at line 2


SQL>
SQL>
SQL>
SQL> ED
Wrote file afiedt.buf

  1  DECLARE
  2  V_EMPNO EMP.EMPNO%TYPE :=&EMPNO;
  3  EMP_REC EMP%ROWTYPE;
  4  BEGIN
  5  SELECT * INTO EMP_REC FROM SCOTT.EMP
  6  WHERE EMPNO=V_EMPNO;
  7  &D(EMP_REC.ENAME ||' '||EMP_REC.JOB||'  '||EMP_REC.SAL||'  '||EMP_REC.DEPTNO);
  8  EXCEPTION
  9    WHEN NO_DATA_FOUND THEN
 10     &D('RECORD NOT EXIST ...');
 11    WHEN OTHERS THEN
 12  &D(SQLCODE||'   '||SQLERRM||'  '||'MY OTHER ERROR');
 13* END;
SQL> /
Enter value for empno: 45789
DECLARE
*
ERROR at line 1:
ORA-06502: PL/SQL: numeric or value error: number precision too large
ORA-06512: at line 2


SQL>
SQL>
SQL>
SQL>
SQL> ED
Wrote file afiedt.buf

  1  DECLARE
  2  V_EMPNO EMP.EMPNO%TYPE :=&EMPNO;
  3  EMP_REC EMP%ROWTYPE;
  4  BEGIN
  5  SELECT * INTO EMP_REC FROM SCOTT.EMP
  6  WHERE EMPNO=V_EMPNO;
  7  &D(EMP_REC.ENAME ||' '||EMP_REC.JOB||'  '||EMP_REC.SAL||'  '||EMP_REC.DEPTNO);
  8  EXCEPTION
  9    WHEN NO_DATA_FOUND THEN
 10     &D('RECORD NOT EXIST ...');
 11    WHEN OTHERS THEN
 12  &D(SQLCODE||'   '||SQLERRM||'  '||'MY OTHER ERROR');
 13* END;
SQL> /
Enter value for empno: 4578
RECORD NOT EXIST ...


PL/SQL procedure successfully completed.

SQL> ED
Wrote file afiedt.buf
```

```
  1  DECLARE
  2  V_EMPNO EMP.EMPNO%TYPE :=&EMPNO;
  3  EMP_REC EMP%ROWTYPE;
  4  BEGIN
  5  SELECT * INTO EMP_REC FROM SCOTT.EMP
  6  WHERE EMPNO=V_EMPNO
  7  &D(EMP_REC.ENAME ||' '||EMP_REC.JOB||'  '||EMP_REC.SAL||'  '||EMP_REC.DEPTNO);
  8  EXCEPTION
  9    WHEN NO_DATA_FOUND THEN
 10     &D('RECORD NOT EXIST ...');
 11    WHEN OTHERS THEN
 12  &D(SQLCODE||'   '||SQLERRM||'  '||'MY OTHER ERROR');
 13* END;
SQL> /
Enter value for empno: 4578
DBMS_OUTPUT.PUT_LINE(EMP_REC.ENAME ||' '||EMP_REC.JOB||'  '||EMP_REC.SAL||'
'||EMP_REC.DEPTNO);
*
ERROR at line 7:
ORA-06550: line 6, column 21:
PL/SQL: ORA-00933: SQL command not properly ended
ORA-06550: line 5, column 1:
PL/SQL: SQL Statement ignored


SQL> ED
Wrote file afiedt.buf

  1  DECLARE
  2  V_EMPNO EMP.EMPNO%TYPE :=&EMPNO;
  3  EMP_REC EMP%ROWTYPE;
  4  BEGIN
  5  SELECT * INTO EMP_REC FROM SCOTT.EMP
  6  WHERE EMPNO=V_EMPNO;
  7  &D(EMP_REC.ENAME ||' '||EMP_REC.JOB||'  '||EMP_REC.SAL||'  '||EMP_REC.DEPTNO);
  8  EXCEPTION
  9    WHEN NO_DATA_FOUND THEN
 10  NULL;
 11        --D('RECORD NOT EXIST ...');
 12    WHEN OTHERS THEN
 13  &D(SQLCODE||'   '||SQLERRM||'  '||'MY OTHER ERROR');
 14* END;
SQL> /
Enter value for empno: 4578

PL/SQL procedure successfully completed.

SQL> ED
Wrote file afiedt.buf

  1  DECLARE
  2  V_JOB EMP.JOB%TYPE :='&JOB';
  3  EMP_REC EMP%ROWTYPE;
  4  BEGIN
  5  SELECT * INTO EMP_REC FROM SCOTT.EMP
  6  WHERE JOB=V_JOB;
  7  &D(EMP_REC.ENAME ||' '||EMP_REC.JOB||'  '||EMP_REC.SAL||'  '||EMP_REC.DEPTNO);
  8  EXCEPTION
  9    WHEN NO_DATA_FOUND THEN
 10   &D('RECORD NOT EXIST ...');
 11    WHEN OTHERS THEN
 12  &D(SQLCODE||'   '||SQLERRM||'  '||'MY OTHER ERROR');
```

```
 13* END;
SQL> /
Enter value for job: PRESIDENT
KING PRESIDENT  5000  10


PL/SQL procedure successfully completed.

SQL> /
Enter value for job: SR_CLERK
RECORD NOT EXIST ...


PL/SQL procedure successfully completed.

SQL>
SQL>
SQL>
SQL> /
Enter value for job: SALESMAN
-1422   ORA-01422: exact fetch returns more than requested number of rows  MY OTHER
ERROR

PL/SQL procedure successfully completed.

SQL>
SQL>
SQL>
SQL>
SQL> ED
Wrote file afiedt.buf

  1  DECLARE
  2  V_JOB EMP.JOB%TYPE :='&JOB';
  3  EMP_REC EMP%ROWTYPE;
  4  BEGIN
  5  SELECT * INTO EMP_REC FROM SCOTT.EMP
  6  WHERE JOB=V_JOB;
  7  &D(EMP_REC.ENAME ||' '||EMP_REC.JOB||'  '||EMP_REC.SAL||'  '||EMP_REC.DEPTNO);
  8  EXCEPTION
  9    WHEN NO_DATA_FOUND THEN
 10   &D('RECORD NOT EXIST ...');
 11  WHEN TOO_MANY_ROWS THEN
 12  &D('CAN NOT DISPALY RECORD WITH METHOD ...');
 13    WHEN OTHERS THEN
 14  &D(SQLCODE||'   '||SQLERRM||'  '||'MY OTHER ERROR');
 15* END;
 16  /
Enter value for job: SALESMAN
CAN NOT DISPALY RECORD WITH METHOD ...


PL/SQL procedure successfully completed.

SQL>
SQL>
SQL>
SQL>
SQL> ED
Wrote file afiedt.buf

  1  DECLARE
  2  V_JOB EMP.JOB%TYPE :='&JOB';
```

```
  3  EMP_REC EMP%ROWTYPE;
  4  BEGIN
  5  SELECT * INTO EMP_REC FROM SCOTT.EMP
  6  WHERE JOB=V_JOB;
  7  &D(EMP_REC.ENAME ||' '||EMP_REC.JOB||'  '||EMP_REC.SAL||'  '||EMP_REC.DEPTNO);
  8  EXCEPTION
  9    WHEN OTHERS THEN
 10  &D(SQLCODE||'   '||SQLERRM||'  '||'MY OTHER ERROR');
 11    WHEN NO_DATA_FOUND THEN
 12   &D('RECORD NOT EXIST ...');
 13  WHEN TOO_MANY_ROWS THEN
 14  &D('CAN NOT DISPALY RECORD WITH METHOD ...');
 15* END;
 16  /
Enter value for job: SALESMAN
  WHEN OTHERS THEN
  *
ERROR at line 9:
ORA-06550: line 9, column 3:
PLS-00370: OTHERS handler must be last among the exception handlers of a block
ORA-06550: line 0, column 0:
PL/SQL: Compilation unit analysis terminated


SQL>
SQL>
SQL>
SQL>
SQL> ED
Wrote file afiedt.buf

  1  DECLARE
  2  V_JOB EMP.JOB%TYPE :='&JOB';
  3  EMP_REC EMP%ROWTYPE;
  4  BEGIN
  5  SELECT * INTO EMP_REC FROM SCOTT.EMP
  6  WHERE JOB=V_JOB;
  7  &D(EMP_REC.ENAME ||' '||EMP_REC.JOB||'  '||EMP_REC.SAL||'  '||EMP_REC.DEPTNO);
  8  EXCEPTION
  9    WHEN OTHERS THEN
 10  &D(SQLCODE||'   '||SQLERRM||'  '||'MY OTHER ERROR');
 11* END;
SQL> /
Enter value for job: SALESMAN
-1422   ORA-01422: exact fetch returns more than requested number of rows  MY OTHER
ERROR

PL/SQL procedure successfully completed.

SQL>
SQL>
SQL>
SQL> ED
Wrote file afiedt.buf

  1  DECLARE
  2  V_JOB EMP.JOB%TYPE :='&JOB';
  3  EMP_REC EMP%ROWTYPE;
  4  BEGIN
  5  SELECT * INTO EMP_REC FROM SCOTT.EMP
  6  WHERE JOB=V_JOB;
  7  &D(EMP_REC.ENAME ||' '||EMP_REC.JOB||'  '||EMP_REC.SAL||'  '||EMP_REC.DEPTNO);
  8  EXCEPTION
```

```
  9    WHEN NO_DATA_FOUND THEN
 10   &D('RECORD NOT EXIST ...');
 11    WHEN TOO_MANY_ROWS THEN
 12   &D('CAN NOT DISPALY RECORD WITH METHOD ...');
 13    WHEN OTHERS THEN
 14  &D(SQLCODE||'   '||SQLERRM||'  '||'MY OTHER ERROR');
 15* END;
 16  /
Enter value for job: SALESMAN
CAN NOT DISPALY RECORD WITH METHOD ...


PL/SQL procedure successfully completed.

SQL>
SQL>
SQL>
SQL>
SQL> ED
Wrote file afiedt.buf

  1  DECLARE
  2  V_JOB EMP.JOB%TYPE :='&JOB';
  3  EMP_REC EMP%ROWTYPE;
  4  BEGIN
  5  SELECT * INTO EMP_REC FROM SCOTT.EMP
  6  WHERE JOB=V_JOB;
  7  &D(EMP_REC.ENAME ||' '||EMP_REC.JOB||'  '||EMP_REC.SAL||'  '||EMP_REC.DEPTNO);
  8  EXCEPTION
  9    WHEN NO_DATA_FOUND THEN
 10   &D('RECORD NOT EXIST ...');
 11    WHEN TOO_MANY_ROWS THEN
 12  -------------------------NESTED BLOCK--------------------
 13             DECLARE
 14             CURSOR C1 IS SELECT * FROM EMP WHERE JOB=V_JOB;
 15             BEGIN
 16             FOR I IN C1 LOOP
 17             &D(I.ENAME ||' '||I.JOB||'  '||I.SAL||'  '||I.DEPTNO);
 18             END LOOP;
 19             EXCEPTION
 20                  WHEN OTHERS THEN
 21             &D(SQLCODE||'   '||SQLERRM||'  '||'MY OTHER ERROR');
 22             END;
 23  ----------------------END OF NESTED BLOCK-------------------
 24   WHEN OTHERS THEN
 25  &D(SQLCODE||'   '||SQLERRM||'  '||'MY OTHER ERROR');
 26* END;
 27  /
Enter value for job: SALESMAN
ALLEN SALESMAN  1600  30

WARD SALESMAN  1250  30

MARTIN SALESMAN  1250  30

TURNER SALESMAN  1500  30


PL/SQL procedure successfully completed.

SQL>
SQL>
SQL> ED
```

```
Wrote file afiedt.buf

  1  DECLARE
  2  V_JOB EMP.JOB%TYPE :='&JOB';
  3  EMP_REC EMP%ROWTYPE;
  4  BEGIN
  5  SELECT * INTO EMP_REC FROM SCOTT.EMP
  6  WHERE JOB=V_JOB;
  7  &D(EMP_REC.ENAME ||' '||EMP_REC.JOB||'  '||EMP_REC.SAL||'  '||EMP_REC.DEPTNO);
  8  EXCEPTION
  9    WHEN NO_DATA_FOUND THEN
 10   &D('RECORD NOT EXIST ...');
 11    WHEN TOO_MANY_ROWS THEN
 12  &D('I M IN TOO_MANY_ROWS BLOCK..');
 13  -------------------------NESTED BLOCK--------------------
 14             DECLARE
 15             CURSOR C1 IS SELECT * FROM EMP WHERE JOB=V_JOB;
 16             BEGIN
 17             FOR I IN C1 LOOP
 18             &D(I.ENAME ||' '||I.JOB||'  '||I.SAL||'  '||I.DEPTNO);
 19             END LOOP;
 20             EXCEPTION
 21                  WHEN OTHERS THEN
 22             &D(SQLCODE||'   '||SQLERRM||'  '||'MY OTHER ERROR');
 23             END;
 24  ----------------------END OF NESTED BLOCK-------------------
 25   WHEN OTHERS THEN
 26  &D(SQLCODE||'   '||SQLERRM||'  '||'MY OTHER ERROR');
 27* END;
 28
 29  /
Enter value for job: SALESMAN
I M IN TOO_MANY_ROWS BLOCK..

ALLEN SALESMAN  1600  30

WARD SALESMAN  1250  30

MARTIN SALESMAN  1250  30

TURNER SALESMAN  1500  30


PL/SQL procedure successfully completed.

SQL>
SQL>
SQL>
SQL>
SQL> CREATE TABLE EMP_EXCEPTION
  2
SQL>
SQL> ED
Wrote file afiedt.buf

  1  CREATE TABLE EMP_EXCEPTION
  2  (
  3  TRN_DATE DATE,
  4  ERR_CODE VARCHAR2(100),
  5  ERR_MSG VARCHAR2(200)
  6* )
  7  /
CREATE TABLE EMP_EXCEPTION
```

```
            *
ERROR at line 1:
ORA-00955: name is already used by an existing object


SQL> DROP TABLE EMP_EXCEPTION
  2  ;

Table dropped.

SQL>  CREATE TABLE EMP_EXCEPTION
  2   (
  3   TRN_DATE DATE,
  4   ERR_CODE VARCHAR2(100),
  5   ERR_MSG VARCHAR2(200)
  6   );

Table created.

SQL> DESC EMP_EXCEPTION
 Name                                                  Null?    Type
 ----------------------------------------------------- -------- ------------
------------------------------------
 TRN_DATE                                                       DATE
 ERR_CODE                                                       VARCHAR2(100)
 ERR_MSG                                                        VARCHAR2(200)

SQL>
SQL>
SQL>
SQL>
SQL> DECLARE
  2  V_JOB EMP.JOB%TYPE :='&JOB';
  3  EMP_REC EMP%ROWTYPE;
  4  BEGIN
  5  SELECT * INTO EMP_REC FROM SCOTT.EMP
  6  WHERE JOB=V_JOB;
  7  &D(EMP_REC.ENAME ||' '||EMP_REC.JOB||'  '||EMP_REC.SAL||'  '||EMP_REC.DEPTNO);
  8  EXCEPTION
  9    WHEN NO_DATA_FOUND THEN
 10   &D('RECORD NOT EXIST ...');
 11    WHEN TOO_MANY_ROWS THEN
 12  &D('I M IN TOO_MANY_ROWS BLOCK..');
 13  --------------------------NESTED BLOCK--------------------
 14             DECLARE
 15             CURSOR C1 IS SELECT * FROM EMP WHERE JOB=V_JOB;
 16             BEGIN
 17             FOR I IN C1 LOOP
 18             &D(I.ENAME ||' '||I.JOB||'  '||I.SAL||'  '||I.DEPTNO);
 19             END LOOP;
 20             EXCEPTION
 21                  WHEN OTHERS THEN
 22             &D(SQLCODE||'   '||SQLERRM||'  '||'MY OTHER ERROR');
 23             END;
 24  ----------------------END OF NESTED BLOCK-------------------
 25   WHEN OTHERS THEN
 26  &D(SQLCODE||'   '||SQLERRM||'  '||'MY OTHER ERROR');
 27  END;
 28
 29  .
SQL> ED
Wrote file afiedt.buf
```

```
  1  DECLARE
  2  V_JOB EMP.JOB%TYPE :='&JOB';
  3  EMP_REC EMP%ROWTYPE;
  4  BEGIN
  5  SELECT * INTO EMP_REC FROM SCOTT.EMP
  6  WHERE JOB=V_JOB;
  7  &D(EMP_REC.ENAME ||' '||EMP_REC.JOB||'  '||EMP_REC.SAL||'  '||EMP_REC.DEPTNO);
  8  EXCEPTION
  9  /*
 10    WHEN NO_DATA_FOUND THEN
 11   &D('RECORD NOT EXIST ...');
 12    WHEN TOO_MANY_ROWS THEN
 13  &D('I M IN TOO_MANY_ROWS BLOCK..');
 14  -------------------------NESTED BLOCK--------------------
 15             DECLARE
 16             CURSOR C1 IS SELECT * FROM EMP WHERE JOB=V_JOB;
 17             BEGIN
 18             FOR I IN C1 LOOP
 19             &D(I.ENAME ||' '||I.JOB||'  '||I.SAL||'  '||I.DEPTNO);
 20             END LOOP;
 21             EXCEPTION
 22                  WHEN OTHERS THEN
 23             &D(SQLCODE||'   '||SQLERRM||'  '||'MY OTHER ERROR');
 24             END;
 25  ----------------------END OF NESTED BLOCK--------------------
 26  */
 27   WHEN OTHERS THEN
 28  &D(SQLCODE||'   '||SQLERRM||'  '||'MY OTHER ERROR');
 29  INSERT INTO EMP_EXCEPTION
 30  VALUES(SYSDATE,SQLCODE,SQLERRM);
 31  COMMIT;
 32* END;
 33  /
Enter value for job: 1454
VALUES(SYSDATE,SQLCODE,SQLERRM);
                       *
ERROR at line 30:
ORA-06550: line 30, column 24:
PL/SQL: ORA-00984: column not allowed here
ORA-06550: line 29, column 1:
PL/SQL: SQL Statement ignored


SQL> ED
Wrote file afiedt.buf

  1  DECLARE
  2  V_JOB EMP.JOB%TYPE :='&JOB';
  3  EMP_REC EMP%ROWTYPE;
  4  V_ERR VARCHAR2(100);
  5  V_MSG VARCHAR2(200);
  6  BEGIN
  7  SELECT * INTO EMP_REC FROM SCOTT.EMP
  8  WHERE JOB=V_JOB;
  9  &D(EMP_REC.ENAME ||' '||EMP_REC.JOB||'  '||EMP_REC.SAL||'  '||EMP_REC.DEPTNO);
 10  EXCEPTION
 11  /*
 12    WHEN NO_DATA_FOUND THEN
 13   &D('RECORD NOT EXIST ...');
 14    WHEN TOO_MANY_ROWS THEN
 15  &D('I M IN TOO_MANY_ROWS BLOCK..');
 16  -------------------------NESTED BLOCK--------------------
 17             DECLARE
```

```
 18             CURSOR C1 IS SELECT * FROM EMP WHERE JOB=V_JOB;
 19             BEGIN
 20             FOR I IN C1 LOOP
 21             &D(I.ENAME ||' '||I.JOB||'  '||I.SAL||'  '||I.DEPTNO);
 22             END LOOP;
 23             EXCEPTION
 24                  WHEN OTHERS THEN
 25             &D(SQLCODE||'   '||SQLERRM||'  '||'MY OTHER ERROR');
 26             END;
 27  ----------------------END OF NESTED BLOCK--------------------
 28  */
 29   WHEN OTHERS THEN
 30  &D(SQLCODE||'   '||SQLERRM||'  '||'MY OTHER ERROR');
 31  V_ERR := SQLCODE;
 32  V_MSG := SQLERRM;
 33  INSERT INTO EMP_EXCEPTION
 34  VALUES(SYSDATE,V_ERR,V_MSG);
 35  COMMIT;
 36* END;
 37  /
Enter value for job: 4578
100   ORA-01403: no data found  MY OTHER ERROR


PL/SQL procedure successfully completed.

SQL>
SQL>
SQL> SELECT * FROM EMP_EXCEPTION;

TRN_DATE

---------

ERR_CODE

--------------------------------------------------------------------------------
----------------
ERR_MSG

--------------------------------------------------------------------------------
----------------
19-FEB-13

100

ORA-01403: no data found




SQL>
SQL>
SQL>
SQL>
SQL> ED
Wrote file afiedt.buf

  1* SELECT * FROM EMP_EXCEPTION
SQL>
SQL> DECLARE
  2  V_JOB EMP.JOB%TYPE :='&JOB';
  3  EMP_REC EMP%ROWTYPE;
```

```
  4  V_ERR VARCHAR2(100);
  5  V_MSG VARCHAR2(200);
  6  BEGIN
  7  SELECT * INTO EMP_REC FROM SCOTT.EMP
  8  WHERE JOB=V_JOB;
  9  &D(EMP_REC.ENAME ||' '||EMP_REC.JOB||'  '||EMP_REC.SAL||'  '||EMP_REC.DEPTNO);
 10  EXCEPTION
 11  /*
 12    WHEN NO_DATA_FOUND THEN
 13   &D('RECORD NOT EXIST ...');
 14    WHEN TOO_MANY_ROWS THEN
 15  &D('I M IN TOO_MANY_ROWS BLOCK..');
 16  -------------------------NESTED BLOCK--------------------
 17             DECLARE
 18             CURSOR C1 IS SELECT * FROM EMP WHERE JOB=V_JOB;
 19             BEGIN
 20             FOR I IN C1 LOOP
 21             &D(I.ENAME ||' '||I.JOB||'  '||I.SAL||'  '||I.DEPTNO);
 22             END LOOP;
 23             EXCEPTION
 24                  WHEN OTHERS THEN
 25             &D(SQLCODE||'   '||SQLERRM||'  '||'MY OTHER ERROR');
 26             END;
 27  ----------------------END OF NESTED BLOCK-------------------
 28  */
 29   WHEN OTHERS THEN
 30  &D(SQLCODE||'   '||SQLERRM||'  '||'MY OTHER ERROR');
 31  V_ERR := SQLCODE;
 32  V_MSG := SQLERRM;
 33  INSERT INTO EMP_EXCEPTION
 34  VALUES(SYSDATE,V_ERR,V_MSG);
 35  COMMIT;
 36  END;
 37  .
SQL> ED
Wrote file afiedt.buf

  1  DECLARE
  2  V_JOB EMP.JOB%TYPE :='&JOB';
  3  EMP_REC EMP%ROWTYPE;
  4  V_ERR VARCHAR2(100);
  5  V_MSG VARCHAR2(200);
  6  BEGIN
  7  SELECT * INTO EMP_REC FROM SCOTT.EMP
  8  WHERE JOB=V_JOB;
  9  &D(EMP_REC.ENAME ||' '||EMP_REC.JOB||'  '||EMP_REC.SAL||'  '||EMP_REC.DEPTNO);
 10  EXCEPTION
 11  /*
 12    WHEN NO_DATA_FOUND THEN
 13   &D('RECORD NOT EXIST ...');
 14    WHEN TOO_MANY_ROWS THEN
 15  &D('I M IN TOO_MANY_ROWS BLOCK..');
 16  -------------------------NESTED BLOCK--------------------
 17             DECLARE
 18             CURSOR C1 IS SELECT * FROM EMP WHERE JOB=V_JOB;
 19             BEGIN
 20             FOR I IN C1 LOOP
 21             &D(I.ENAME ||' '||I.JOB||'  '||I.SAL||'  '||I.DEPTNO);
 22             END LOOP;
 23             EXCEPTION
 24                  WHEN OTHERS THEN
 25             &D(SQLCODE||'   '||SQLERRM||'  '||'MY OTHER ERROR');
 26             END;
```

```
 27  ----------------------END OF NESTED BLOCK--------------------
 28  */
 29   WHEN OTHERS THEN
 30  &D(SQLCODE||'   '||'('||SQLERRM||')');
 31  V_ERR := SQLCODE;
 32  V_MSG := SQLERRM;
 33  INSERT INTO EMP_EXCEPTION
 34  VALUES(SYSDATE,V_ERR,V_MSG);
 35  COMMIT;
 36* END;
SQL> /
Enter value for job: 445
100   (ORA-01403: no data found)


PL/SQL procedure successfully completed.

SQL>
SQL>
SQL> ED
Wrote file afiedt.buf

  1  DECLARE
  2  V_JOB EMP.JOB%TYPE :='&JOB';
  3  EMP_REC EMP%ROWTYPE;
  4  V_ERR VARCHAR2(100);
  5  V_MSG VARCHAR2(200);
  6  BEGIN
  7  SELECT * INTO EMP_REC FROM SCOTT.EMP
  8  WHERE JOB=V_JOB;
  9  &D(EMP_REC.ENAME ||' '||EMP_REC.JOB||'  '||EMP_REC.SAL||'  '||EMP_REC.DEPTNO);
 10  EXCEPTION
 11  /*
 12    WHEN NO_DATA_FOUND THEN
 13   &D('RECORD NOT EXIST ...');
 14    WHEN TOO_MANY_ROWS THEN
 15  &D('I M IN TOO_MANY_ROWS BLOCK..');
 16  -------------------------NESTED BLOCK--------------------
 17             DECLARE
 18             CURSOR C1 IS SELECT * FROM EMP WHERE JOB=V_JOB;
 19             BEGIN
 20             FOR I IN C1 LOOP
 21             &D(I.ENAME ||' '||I.JOB||'  '||I.SAL||'  '||I.DEPTNO);
 22             END LOOP;
 23             EXCEPTION
 24                  WHEN OTHERS THEN
 25             &D(SQLCODE||'   '||SQLERRM||'  '||'MY OTHER ERROR');
 26             END;
 27  ----------------------END OF NESTED BLOCK--------------------
 28  */
 29   WHEN OTHERS THEN
 30  &D(SQLCODE||'   '||'('||SUBSTR(SQLERRM,11)||')');
 31  V_ERR := SQLCODE;
 32  V_MSG := SQLERRM;
 33  INSERT INTO EMP_EXCEPTION
 34  VALUES(SYSDATE,V_ERR,V_MSG);
 35  COMMIT;
 36* END;
SQL> /
Enter value for job: 45
100   ( no data found)
```

```
PL/SQL procedure successfully completed.

SQL>
SQL>
SQL>
SQL>
SQL> ED
Wrote file afiedt.buf

  1  CREATE OR REPLACE PROCEDURE MY_CODE IS
  2  ---DECLARE
  3  V_JOB EMP.JOB%TYPE :='JOB';
  4  EMP_REC EMP%ROWTYPE;
  5  V_ERR VARCHAR2(100);
  6  V_MSG VARCHAR2(200);
  7  BEGIN
  8  SELECT * INTO EMP_REC FROM SCOTT.EMP
  9  WHERE JOB=V_JOB;
 10  &D(EMP_REC.ENAME ||' '||EMP_REC.JOB||'  '||EMP_REC.SAL||'  '||EMP_REC.DEPTNO);
 11  EXCEPTION
 12  /*
 13    WHEN NO_DATA_FOUND THEN
 14   &D('RECORD NOT EXIST ...');
 15    WHEN TOO_MANY_ROWS THEN
 16  &D('I M IN TOO_MANY_ROWS BLOCK..');
 17  -------------------------NESTED BLOCK--------------------
 18             DECLARE
 19             CURSOR C1 IS SELECT * FROM EMP WHERE JOB=V_JOB;
 20             BEGIN
 21             FOR I IN C1 LOOP
 22             &D(I.ENAME ||' '||I.JOB||'  '||I.SAL||'  '||I.DEPTNO);
 23             END LOOP;
 24             EXCEPTION
 25                  WHEN OTHERS THEN
 26             &D(SQLCODE||'   '||SQLERRM||'  '||'MY OTHER ERROR');
 27             END;
 28  ----------------------END OF NESTED BLOCK--------------------
 29  */
 30   WHEN OTHERS THEN
 31  &D(SQLCODE||'   '||'('||SUBSTR(SQLERRM,11)||')');
 32  V_ERR := SQLCODE;
 33  V_MSG := SQLERRM;
 34  INSERT INTO EMP_EXCEPTION
 35  VALUES(SYSDATE,V_ERR,V_MSG);
 36  COMMIT;
 37* END;
SQL> /

Procedure created.

SQL> ED
Wrote file afiedt.buf

  1  DECLARE
  2  V_JOB EMP.JOB%TYPE :='JOB';
  3  EMP_REC EMP%ROWTYPE;
  4  V_ERR VARCHAR2(100);
  5  V_MSG VARCHAR2(200);
  6  BEGIN
  7  SELECT * INTO EMP_REC FROM SCOTT.EMP
  8  WHERE JOB=V_JOB;
  9  &D(EMP_REC.ENAME ||' '||EMP_REC.JOB||'  '||EMP_REC.SAL||'  '||EMP_REC.DEPTNO);
 10  EXCEPTION
```

```
 11    WHEN NO_DATA_FOUND THEN
 12   &D('RECORD NOT EXIST ...');
 13    WHEN TOO_MANY_ROWS THEN
 14  &D('I M IN TOO_MANY_ROWS BLOCK..');
 15  -------------------------NESTED BLOCK--------------------
 16             DECLARE
 17             CURSOR C1 IS SELECT * FROM EMP WHERE JOB=V_JOB;
 18             BEGIN
 19             FOR I IN C1 LOOP
 20             &D(I.ENAME ||' '||I.JOB||'  '||I.SAL||'  '||I.DEPTNO);
 21             END LOOP;
 22             EXCEPTION
 23                  WHEN OTHERS THEN
 24             &D(SQLCODE||'   '||SQLERRM||'  '||'MY OTHER ERROR');
 25             END;
 26  ----------------------END OF NESTED BLOCK-------------------
 27   WHEN OTHERS THEN
 28  &D(SQLCODE||'   '||'('||SUBSTR(SQLERRM,11)||')');
 29  V_ERR := SQLCODE;
 30  V_MSG := SQLERRM;
 31  INSERT INTO EMP_EXCEPTION
 32  VALUES(SYSDATE,V_ERR,V_MSG);
 33  COMMIT;
 34* END;
 35  /
RECORD NOT EXIST ...


PL/SQL procedure successfully completed.

SQL> ED
Wrote file afiedt.buf

  1  DECLARE
  2  V_JOB EMP.JOB%TYPE :='&JOB';
  3  EMP_REC EMP%ROWTYPE;
  4  V_ERR VARCHAR2(100);
  5  V_MSG VARCHAR2(200);
  6  BEGIN
  7  SELECT * INTO EMP_REC FROM SCOTT.EMP
  8  WHERE JOB=V_JOB;
  9  &D(EMP_REC.ENAME ||' '||EMP_REC.JOB||'  '||EMP_REC.SAL||'  '||EMP_REC.DEPTNO);
 10  EXCEPTION
 11    WHEN NO_DATA_FOUND THEN
 12   &D('RECORD NOT EXIST ...');
 13    WHEN TOO_MANY_ROWS THEN
 14  &D('I M IN TOO_MANY_ROWS BLOCK..');
 15  -------------------------NESTED BLOCK--------------------
 16             DECLARE
 17             CURSOR C1 IS SELECT * FROM EMP WHERE JOB=V_JOB;
 18             BEGIN
 19             FOR I IN C1 LOOP
 20             &D(I.ENAME ||' '||I.JOB||'  '||I.SAL||'  '||I.DEPTNO);
 21             END LOOP;
 22             EXCEPTION
 23                  WHEN OTHERS THEN
 24             &D(SQLCODE||'   '||SQLERRM||'  '||'MY OTHER ERROR');
 25             END;
 26  ----------------------END OF NESTED BLOCK-------------------
 27   WHEN OTHERS THEN
 28  &D(SQLCODE||'   '||'('||SUBSTR(SQLERRM,11)||')');
 29  V_ERR := SQLCODE;
 30  V_MSG := SQLERRM;
```

```
 31  INSERT INTO EMP_EXCEPTION
 32  VALUES(SYSDATE,V_ERR,V_MSG);
 33  COMMIT;
 34* END;
SQL> /
Enter value for job: SALESMAN
I M IN TOO_MANY_ROWS BLOCK..

ALLEN SALESMAN  1600  30

WARD SALESMAN  1250  30

MARTIN SALESMAN  1250  30

TURNER SALESMAN  1500  30


PL/SQL procedure successfully completed.

SQL>
SQL>
SQL> /
Enter value for job: DADS
RECORD NOT EXIST ...


PL/SQL procedure successfully completed.

SQL>
SQL> /
Enter value for job: PRESIDENT
KING PRESIDENT  5000  10


PL/SQL procedure successfully completed.

SQL>
SQL> ED
Wrote file afiedt.buf

  1  DECLARE
  2  V_JOB EMP.JOB%TYPE :='&JOB';
  3  EMP_REC EMP%ROWTYPE;
  4  V_ERR VARCHAR2(100);
  5  V_MSG VARCHAR2(200);
  6  BEGIN
  7  SELECT * INTO EMP_REC FROM SCOTT.EMP
  8  WHERE JOB=V_JOB;
  9  &D(EMP_REC.ENAME ||' '||EMP_REC.JOB||'  '||EMP_REC.SAL||'  '||EMP_REC.DEPTNO);
 10  EXCEPTION
 11    WHEN NO_DATA_FOUND THEN
 12   &D('RECORD NOT EXIST ...');
 13    WHEN TOO_MANY_ROWS THEN
 14  &D('I M IN TOO_MANY_ROWS BLOCK..');
 15  -------------------------NESTED BLOCK--------------------
 16             DECLARE
 17                SUM_SAL NUMBER :=0;
 18             CURSOR C1 IS SELECT * FROM EMP WHERE JOB=V_JOB;
 19              SELECT SUM(SAL) INTO SUM_SAL FROM EMP
 20         WHERE JOB=V_JOB;
 21             BEGIN
 22             FOR I IN C1 LOOP
 23             &D(I.ENAME ||' '||I.JOB||'  '||I.SAL||'  '||I.DEPTNO);
```

```
 24             END LOOP;
 25             &D('TOTAL SALARY FOR THIS JOB IS...'||SUM_SAL);
 26             EXCEPTION
 27                  WHEN OTHERS THEN
 28             &D(SQLCODE||'   '||SQLERRM||'  '||'MY OTHER ERROR');
 29             END;
 30  ----------------------END OF NESTED BLOCK--------------------
 31   WHEN OTHERS THEN
 32  &D(SQLCODE||'   '||'('||SUBSTR(SQLERRM,11)||')');
 33  V_ERR := SQLCODE;
 34  V_MSG := SQLERRM;
 35  INSERT INTO EMP_EXCEPTION
 36  VALUES(SYSDATE,V_ERR,V_MSG);
 37  COMMIT;
 38* END;
 39  /
Enter value for job: SALESMAN
            SELECT SUM(SAL) INTO SUM_SAL FROM EMP
            *
ERROR at line 19:
ORA-06550: line 19, column 13:
PLS-00103: Encountered the symbol "SELECT" when expecting one of the following:
begin function package pragma procedure subtype type use
<an identifier> <a double-quoted delimited-identifier> form
current cursor
The symbol "begin" was substituted for "SELECT" to continue.
ORA-06550: line 31, column 2:
PLS-00103: Encountered the symbol "WHEN" when expecting one of the following:
begin case declare end exception exit for goto if loop mod
null pragma raise return select update while with
<an identifier> <a double-quoted delimite
ORA-06550: line 38, column 4:
PLS-00103: Encountered the symbol "end-of-file" when expecting one of the following:

begin case declare end exit for goto if loop mod null pragma
raise return select update when while with <an identifier>
<a double-quoted delimi


SQL>
SQL> ED
Wrote file afiedt.buf

  1  DECLARE
  2  V_JOB EMP.JOB%TYPE :='&JOB';
  3  EMP_REC EMP%ROWTYPE;
  4  V_ERR VARCHAR2(100);
  5  V_MSG VARCHAR2(200);
  6  BEGIN
  7  SELECT * INTO EMP_REC FROM SCOTT.EMP
  8  WHERE JOB=V_JOB;
  9  &D(EMP_REC.ENAME ||' '||EMP_REC.JOB||'  '||EMP_REC.SAL||'  '||EMP_REC.DEPTNO);
 10  EXCEPTION
 11    WHEN NO_DATA_FOUND THEN
 12   &D('RECORD NOT EXIST ...');
 13    WHEN TOO_MANY_ROWS THEN
 14  &D('I M IN TOO_MANY_ROWS BLOCK..');
 15  -------------------------NESTED BLOCK--------------------
 16             DECLARE
 17                SUM_SAL NUMBER :=0;
 18             CURSOR C1 IS SELECT * FROM EMP WHERE JOB=V_JOB;
 19              SELECT SUM(SAL) INTO SUM_SAL FROM EMP
 20         WHERE JOB=V_JOB;
```

```
 21             BEGIN
 22             FOR I IN C1 LOOP
 23             &D(I.ENAME ||' '||I.JOB||'  '||I.SAL||'  '||I.DEPTNO);
 24             END LOOP;
 25             &D('TOTAL SALARY FOR THIS JOB IS...'||SUM_SAL);
 26             EXCEPTION
 27                  WHEN OTHERS THEN
 28             &D(SQLCODE||'   '||SQLERRM||'  '||'MY OTHER ERROR');
 29             END;
 30  ----------------------END OF NESTED BLOCK--------------------
 31   WHEN OTHERS THEN
 32  &D(SQLCODE||'   '||'('||SUBSTR(SQLERRM,11)||')');
 33  V_ERR := SQLCODE;
 34  V_MSG := SQLERRM;
 35  INSERT INTO EMP_EXCEPTION
 36  VALUES(SYSDATE,V_ERR,V_MSG);
 37  COMMIT;
 38* END;
SQL>
SQL> /
Enter value for job: SALESMAN
            SELECT SUM(SAL) INTO SUM_SAL FROM EMP
            *
ERROR at line 19:
ORA-06550: line 19, column 13:
PLS-00103: Encountered the symbol "SELECT" when expecting one of the following:
begin function package pragma procedure subtype type use
<an identifier> <a double-quoted delimited-identifier> form
current cursor
The symbol "begin" was substituted for "SELECT" to continue.
ORA-06550: line 31, column 2:
PLS-00103: Encountered the symbol "WHEN" when expecting one of the following:
begin case declare end exception exit for goto if loop mod
null pragma raise return select update while with
<an identifier> <a double-quoted delimite
ORA-06550: line 38, column 4:
PLS-00103: Encountered the symbol "end-of-file" when expecting one of the following:

begin case declare end exit for goto if loop mod null pragma
raise return select update when while with <an identifier>
<a double-quoted delimi


SQL> ED
Wrote file afiedt.buf

  1  DECLARE
  2  V_JOB EMP.JOB%TYPE :='&JOB';
  3  EMP_REC EMP%ROWTYPE;
  4  V_ERR VARCHAR2(100);
  5  V_MSG VARCHAR2(200);
  6  BEGIN
  7  SELECT * INTO EMP_REC FROM SCOTT.EMP
  8  WHERE JOB=V_JOB;
  9  &D(EMP_REC.ENAME ||' '||EMP_REC.JOB||'  '||EMP_REC.SAL||'  '||EMP_REC.DEPTNO);
 10  EXCEPTION
 11    WHEN NO_DATA_FOUND THEN
 12   &D('RECORD NOT EXIST ...');
 13    WHEN TOO_MANY_ROWS THEN
 14  &D('I M IN TOO_MANY_ROWS BLOCK..');
 15  -------------------------NESTED BLOCK--------------------
 16             DECLARE
 17                SUM_SAL NUMBER :=0;
```

```
 18             CURSOR C1 IS SELECT * FROM EMP WHERE JOB=V_JOB;
 19             BEGIN
 20               SELECT SUM(SAL) INTO SUM_SAL FROM EMP
 21         WHERE JOB=V_JOB;
 22             FOR I IN C1 LOOP
 23             &D(I.ENAME ||' '||I.JOB||'  '||I.SAL||'  '||I.DEPTNO);
 24             END LOOP;
 25             &D('TOTAL SALARY FOR THIS JOB IS...'||SUM_SAL);
 26             EXCEPTION
 27                  WHEN OTHERS THEN
 28             &D(SQLCODE||'   '||SQLERRM||'  '||'MY OTHER ERROR');
 29             END;
 30  ----------------------END OF NESTED BLOCK-------------------
 31   WHEN OTHERS THEN
 32  &D(SQLCODE||'   '||'('||SUBSTR(SQLERRM,11)||')');
 33  V_ERR := SQLCODE;
 34  V_MSG := SQLERRM;
 35  INSERT INTO EMP_EXCEPTION
 36  VALUES(SYSDATE,V_ERR,V_MSG);
 37  COMMIT;
 38* END;
 39  /
Enter value for job: SALESMAN
I M IN TOO_MANY_ROWS BLOCK..

ALLEN SALESMAN  1600  30

WARD SALESMAN  1250  30

MARTIN SALESMAN  1250  30

TURNER SALESMAN  1500  30

TOTAL SALARY FOR THIS JOB IS...5600


PL/SQL procedure successfully completed.

SQL>
SQL>
SQL> ED
Wrote file afiedt.buf

  1  DECLARE
  2  V_JOB EMP.JOB%TYPE :='&JOB';
  3  ---EMP_REC EMP%ROWTYPE;
  4  -------------PLSQL RECORDS----------------------
  5  TYPE E_REC IS RECORD
  6  (V_ENAME VARCHAR2(20),V_JOB VARCHAR2(20),V_SAL NUMBER,V_DEPTNO NUMBER);
  7  EMP_REC E_REC;
  8  V_ERR VARCHAR2(100);
  9  V_MSG VARCHAR2(200);
 10  BEGIN
 11  SELECT * INTO EMP_REC FROM SCOTT.EMP
 12  WHERE JOB=V_JOB;
 13  &D(EMP_REC.ENAME ||' '||EMP_REC.JOB||'  '||EMP_REC.SAL||'  '||EMP_REC.DEPTNO);
 14  EXCEPTION
 15    WHEN NO_DATA_FOUND THEN
 16   &D('RECORD NOT EXIST ...');
 17    WHEN TOO_MANY_ROWS THEN
 18  &D('I M IN TOO_MANY_ROWS BLOCK..');
 19  -------------------------NESTED BLOCK--------------------
 20             DECLARE
```

```
 21               SUM_SAL NUMBER :=0;
 22             CURSOR C1 IS SELECT * FROM EMP WHERE JOB=V_JOB;
 23             BEGIN
 24               SELECT SUM(SAL) INTO SUM_SAL FROM EMP
 25          WHERE JOB=V_JOB;
 26             FOR I IN C1 LOOP
 27             &D(I.ENAME ||' '||I.JOB||'  '||I.SAL||'  '||I.DEPTNO);
 28             END LOOP;
 29             &D('TOTAL SALARY FOR THIS JOB IS...'||SUM_SAL);
 30             EXCEPTION
 31                  WHEN OTHERS THEN
 32             &D(SQLCODE||'   '||SQLERRM||'  '||'MY OTHER ERROR');
 33             END;
 34  ----------------------END OF NESTED BLOCK-------------------
 35   WHEN OTHERS THEN
 36  &D(SQLCODE||'   '||'('||SUBSTR(SQLERRM,11)||')');
 37  V_ERR := SQLCODE;
 38  V_MSG := SQLERRM;
 39  INSERT INTO EMP_EXCEPTION
 40  VALUES(SYSDATE,V_ERR,V_MSG);
 41  COMMIT;
 42* END;
 43  /
Enter value for job: PRESIDENT
SELECT * INTO EMP_REC FROM SCOTT.EMP
                      *
ERROR at line 11:
ORA-06550: line 11, column 23:
PL/SQL: ORA-00947: not enough values
ORA-06550: line 11, column 1:
PL/SQL: SQL Statement ignored
ORA-06550: line 13, column 30:
PLS-00302: component 'ENAME' must be declared
ORA-06550: line 13, column 1:
PL/SQL: Statement ignored


SQL> ED
Wrote file afiedt.buf

  1  DECLARE
  2  V_JOB EMP.JOB%TYPE :='&JOB';
  3  ---EMP_REC EMP%ROWTYPE;
  4  -------------PLSQL RECORDS----------------------
  5  TYPE E_REC IS RECORD
  6  (V_ENAME VARCHAR2(20),V_JOB VARCHAR2(20),V_SAL NUMBER,V_DEPTNO NUMBER);
  7  EMP_REC E_REC;
  8  V_ERR VARCHAR2(100);
  9  V_MSG VARCHAR2(200);
 10  BEGIN
 11  SELECT ENAME,JOB,SAL,DEPTNO INTO EMP_REC FROM SCOTT.EMP
 12  WHERE JOB=V_JOB;
 13  &D(EMP_REC.ENAME ||' '||EMP_REC.JOB||'  '||EMP_REC.SAL||'  '||EMP_REC.DEPTNO);
 14  EXCEPTION
 15    WHEN NO_DATA_FOUND THEN
 16   &D('RECORD NOT EXIST ...');
 17    WHEN TOO_MANY_ROWS THEN
 18  &D('I M IN TOO_MANY_ROWS BLOCK..');
 19  -------------------------NESTED BLOCK--------------------
 20             DECLARE
 21               SUM_SAL NUMBER :=0;
 22             CURSOR C1 IS SELECT * FROM EMP WHERE JOB=V_JOB;
 23             BEGIN
```

```
 24              SELECT SUM(SAL) INTO SUM_SAL FROM EMP
 25         WHERE JOB=V_JOB;
 26             FOR I IN C1 LOOP
 27             &D(I.ENAME ||' '||I.JOB||'  '||I.SAL||'  '||I.DEPTNO);
 28             END LOOP;
 29             &D('TOTAL SALARY FOR THIS JOB IS...'||SUM_SAL);
 30             EXCEPTION
 31                  WHEN OTHERS THEN
 32             &D(SQLCODE||'   '||SQLERRM||'  '||'MY OTHER ERROR');
 33             END;
 34  ----------------------END OF NESTED BLOCK-------------------
 35   WHEN OTHERS THEN
 36  &D(SQLCODE||'   '||'('||SUBSTR(SQLERRM,11)||')');
 37  V_ERR := SQLCODE;
 38  V_MSG := SQLERRM;
 39  INSERT INTO EMP_EXCEPTION
 40  VALUES(SYSDATE,V_ERR,V_MSG);
 41  COMMIT;
 42* END;
SQL> /
Enter value for job: PRESIDENT
DBMS_OUTPUT.PUT_LINE(EMP_REC.ENAME ||' '||EMP_REC.JOB||'  '||EMP_REC.SAL||'
'||EMP_REC.DEPTNO);
                            *
ERROR at line 13:
ORA-06550: line 13, column 30:
PLS-00302: component 'ENAME' must be declared
ORA-06550: line 13, column 1:
PL/SQL: Statement ignored


SQL>
SQL>
SQL> ED
Wrote file afiedt.buf

  1  DECLARE
  2  V_JOB EMP.JOB%TYPE :='&JOB';
  3  ---EMP_REC EMP%ROWTYPE;
  4  ------------PLSQL RECORDS----------------------
  5  TYPE E_REC IS RECORD
  6  (V_ENAME VARCHAR2(20),V_JOB VARCHAR2(20),V_SAL NUMBER,V_DEPTNO NUMBER);
  7  EMP_REC E_REC;
  8  V_ERR VARCHAR2(100);
  9  V_MSG VARCHAR2(200);
 10  BEGIN
 11  SELECT ENAME,JOB,SAL,DEPTNO INTO EMP_REC FROM SCOTT.EMP
 12  WHERE JOB=V_JOB;
 13  &D(EMP_REC.V_ENAME ||' '||EMP_REC.V_JOB||'  '||EMP_REC.V_SAL||'
'||EMP_REC.V_DEPTNO);
 14  EXCEPTION
 15    WHEN NO_DATA_FOUND THEN
 16   &D('RECORD NOT EXIST ...');
 17    WHEN TOO_MANY_ROWS THEN
 18  &D('I M IN TOO_MANY_ROWS BLOCK..');
 19  -------------------------NESTED BLOCK--------------------
 20             DECLARE
 21                SUM_SAL NUMBER :=0;
 22             CURSOR C1 IS SELECT * FROM EMP WHERE JOB=V_JOB;
 23             BEGIN
 24               SELECT SUM(SAL) INTO SUM_SAL FROM EMP
 25         WHERE JOB=V_JOB;
 26             FOR I IN C1 LOOP
```

```
 27            &D(I.ENAME ||' '||I.JOB||'  '||I.SAL||'  '||I.DEPTNO);
 28            END LOOP;
 29            &D('TOTAL SALARY FOR THIS JOB IS...'||SUM_SAL);
 30            EXCEPTION
 31                 WHEN OTHERS THEN
 32            &D(SQLCODE||'   '||SQLERRM||'  '||'MY OTHER ERROR');
 33            END;
 34  ----------------------END OF NESTED BLOCK-------------------
 35   WHEN OTHERS THEN
 36  &D(SQLCODE||'   '||'('||SUBSTR(SQLERRM,11)||')');
 37  V_ERR := SQLCODE;
 38  V_MSG := SQLERRM;
 39  INSERT INTO EMP_EXCEPTION
 40  VALUES(SYSDATE,V_ERR,V_MSG);
 41  COMMIT;
 42* END;
SQL> /
Enter value for job: PRESIDENT
KING PRESIDENT  5000  10


PL/SQL procedure successfully completed.

SQL>
SQL>
SQL>
SQL>
SQL> ED
Wrote file afiedt.buf

  1  DECLARE
  2  V_JOB EMP.JOB%TYPE :='&JOB';
  3  ---EMP_REC EMP%ROWTYPE;
  4  -------------PLSQL RECORDS----------------------
  5  TYPE E_REC IS RECORD
  6  (V_ENAME VARCHAR2(20)
  7  ,V_JOB VARCHAR2(20),
  8  V_SAL NUMBER,
  9  V_DEPTNO NUMBER);
 10  EMP_REC E_REC;
 11  V_ERR VARCHAR2(100);
 12  V_MSG VARCHAR2(200);
 13  BEGIN
 14  SELECT ENAME,JOB,SAL,DEPTNO INTO EMP_REC FROM SCOTT.EMP
 15  WHERE JOB=V_JOB;
 16  &D(EMP_REC.V_ENAME ||' '||EMP_REC.V_JOB||'  '||EMP_REC.V_SAL||'
'||EMP_REC.V_DEPTNO);
 17  EXCEPTION
 18    WHEN NO_DATA_FOUND THEN
 19   &D('RECORD NOT EXIST ...');
 20    WHEN TOO_MANY_ROWS THEN
 21  &D('I M IN TOO_MANY_ROWS BLOCK..');
 22  -------------------------NESTED BLOCK-------------------
 23            DECLARE
 24               SUM_SAL NUMBER :=0;
 25            CURSOR C1 IS SELECT * FROM EMP WHERE JOB=V_JOB;
 26            BEGIN
 27              SELECT SUM(SAL) INTO SUM_SAL FROM EMP
 28         WHERE JOB=V_JOB;
 29            FOR I IN C1 LOOP
 30            &D(I.ENAME ||' '||I.JOB||'  '||I.SAL||'  '||I.DEPTNO);
 31            END LOOP;
 32            &D('TOTAL SALARY FOR THIS JOB IS...'||SUM_SAL);
```

```
 33             EXCEPTION
 34                  WHEN OTHERS THEN
 35             &D(SQLCODE||'   '||SQLERRM||'  '||'MY OTHER ERROR');
 36             END;
 37  ----------------------END OF NESTED BLOCK-------------------
 38   WHEN OTHERS THEN
 39  &D(SQLCODE||'   '||'('||SUBSTR(SQLERRM,11)||')');
 40  V_ERR := SQLCODE;
 41  V_MSG := SQLERRM;
 42  INSERT INTO EMP_EXCEPTION
 43  VALUES(SYSDATE,V_ERR,V_MSG);
 44  COMMIT;
 45* END;
 46  .
SQL>
SQL>
SQL> ED
Wrote file afiedt.buf

  1  DECLARE
  2  V_JOB EMP.JOB%TYPE :='&JOB';
  3  ---EMP_REC EMP%ROWTYPE;
  4  -------------PLSQL RECORDS----------------------
  5  TYPE E_REC IS RECORD
  6  (V_ENAME VARCHAR2(20)
  7  ,V_JOB VARCHAR2(20),
  8  V_SAL NUMBER,
  9  V_DEPTNO NUMBER);
 10  EMP_REC E_REC;
 11  V_ERR VARCHAR2(100);
 12  V_MSG VARCHAR2(200);
 13  BEGIN
 14  SELECT ENAME,JOB,SAL,DEPTNO INTO EMP_REC FROM SCOTT.EMP
 15  WHERE JOB=V_JOB;
 16  &D(EMP_REC.V_ENAME ||' '||EMP_REC.V_JOB||'  '||EMP_REC.V_SAL||'
'||EMP_REC.V_DEPTNO);
 17  EXCEPTION
 18    WHEN NO_DATA_FOUND THEN
 19   &D('RECORD NOT EXIST ...');
 20    WHEN TOO_MANY_ROWS THEN
 21  &D('I M IN TOO_MANY_ROWS BLOCK..');
 22  -------------------------NESTED BLOCK--------------------
 23             DECLARE
 24                SUM_SAL NUMBER :=0;
 25             CURSOR C1 IS SELECT * FROM EMP WHERE JOB=V_JOB;
 26             BEGIN
 27               SELECT SUM(SAL) INTO SUM_SAL FROM EMP
 28         WHERE JOB=V_JOB;
 29             FOR I IN (SELECT * FROM EMP WHERE JOB=V_JOB) LOOP
 30             &D(I.ENAME ||' '||I.JOB||'  '||I.SAL||'  '||I.DEPTNO);
 31             END LOOP;
 32             &D('TOTAL SALARY FOR THIS JOB IS...'||SUM_SAL);
 33             EXCEPTION
 34                  WHEN OTHERS THEN
 35             &D(SQLCODE||'   '||SQLERRM||'  '||'MY OTHER ERROR');
 36             END;
 37  ----------------------END OF NESTED BLOCK-------------------
 38   WHEN OTHERS THEN
 39  &D(SQLCODE||'   '||'('||SUBSTR(SQLERRM,11)||')');
 40  V_ERR := SQLCODE;
 41  V_MSG := SQLERRM;
 42  INSERT INTO EMP_EXCEPTION
 43  VALUES(SYSDATE,V_ERR,V_MSG);
```

```
 44  COMMIT;
 45* END;
SQL> /
Enter value for job: SALESMAN
I M IN TOO_MANY_ROWS BLOCK..

ALLEN SALESMAN  1600  30

WARD SALESMAN  1250  30

MARTIN SALESMAN  1250  30

TURNER SALESMAN  1500  30

TOTAL SALARY FOR THIS JOB IS...5600


PL/SQL procedure successfully completed.

SQL>
SQL>
SQL>
SQL> SPOOL OFF
```

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

     EMPNO ENAME      JOB               MGR HIREDATE          SAL        COMM
DEPTNO
---------- ---------- --------- ---------- --------- ---------- ----------
----------
       345 ALI        SASD              125

      5454 SMITH      CLERK            7902 17-DEC-80        900
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       1000
30
      7902 FORD       ANALYST          7566 03-DEC-81      45666
20
      7934 MILLER     CLERK            7782 23-JAN-85       1300
10

15 rows selected.

SQL> INSERT INTO EMP(EMPNO,ENAME,JOB,SAL,DEPTNO)
  2  VALUES(500,USER,'SALESMAN',1000,30);

1 row created.

SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL> ED
Wrote file afiedt.buf

  1  DECLARE
  2  V_EMPNO EMP.EMPNO%TYPE :=&EMPNO;
  3  V_ENAME EMP.ENAME%TYPE:='&ENAME';
  4  V_JOB   EMP.JOB%TYPE:='&JOB';
  5  V_SAL   EMP.SAL%TYPE:=&SAL;
  6  V_DEPTNO EMP.DEPTNO%TYPE:=&DEPTNO;
  7  BEGIN
  8  INSERT INTO EMP(EMPNO,ENAME,JOB,SAL,DEPTNO)
```

```
  9  VALUES(V_EMPNO,V_ENAME,V_JOB,V_SAL,V_DEPTNO);
 10  COMMIT;
 11  &D('RECORD CREATED WITH EMPNO '||V_EMPNO);
 12* END;
 13  /
Enter value for empno: 501
Enter value for ename: ALI
Enter value for job: SALESMAN
Enter value for sal: 1000
Enter value for deptno: 30
RECORD CREATED WITH EMPNO 501


PL/SQL procedure successfully completed.

SQL> /
Enter value for empno: 502
Enter value for ename: ALI
Enter value for job: SALESMAN
Enter value for sal: 1000
Enter value for deptno: 30
RECORD CREATED WITH EMPNO 502


PL/SQL procedure successfully completed.

SQL> /
Enter value for empno: 502
Enter value for ename: ALI
Enter value for job: SALESMAN
Enter value for sal: 1200
Enter value for deptno: 30
DECLARE
*
ERROR at line 1:
ORA-00001: unique constraint (SCOTT.PK_EMP) violated
ORA-06512: at line 8


SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL> ED
Wrote file afiedt.buf

  1  DECLARE
  2  V_EMPNO EMP.EMPNO%TYPE :=&EMPNO;
  3  V_ENAME EMP.ENAME%TYPE:='&ENAME';
  4  V_JOB   EMP.JOB%TYPE:='&JOB';
  5  V_SAL   EMP.SAL%TYPE:=&SAL;
  6  V_DEPTNO EMP.DEPTNO%TYPE:=&DEPTNO;
  7  BEGIN
  8  INSERT INTO EMP(EMPNO,ENAME,JOB,SAL,DEPTNO)
  9  VALUES(V_EMPNO,V_ENAME,V_JOB,V_SAL,V_DEPTNO);
 10  COMMIT;
 11  &D('RECORD CREATED WITH EMPNO '||V_EMPNO);
 12  EXCEPTION
 13  WHEN DUP_VAL_ON_INDEX  THEN
 14  &D('DUPLICATION RECORDS');
 15* END;
```

```
 16  /
Enter value for empno: 502
Enter value for ename: ALI
Enter value for job: SALESMAN
Enter value for sal: 1123
Enter value for deptno: 30
DUPLICATION RECORDS


PL/SQL procedure successfully completed.

SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL> ED
Wrote file afiedt.buf

  1  DECLARE
  2  V_EMPNO EMP.EMPNO%TYPE :=&EMPNO;
  3  V_ENAME EMP.ENAME%TYPE:='&ENAME';
  4  V_JOB   EMP.JOB%TYPE:='&JOB';
  5  V_SAL   EMP.SAL%TYPE:=&SAL;
  6  V_DEPTNO EMP.DEPTNO%TYPE:=&DEPTNO;
  7  BEGIN
  8  INSERT INTO EMP(EMPNO,ENAME,JOB,SAL,DEPTNO)
  9  VALUES(V_EMPNO,V_ENAME,V_JOB,V_SAL,V_DEPTNO);
 10  COMMIT;
 11  &D('RECORD CREATED WITH EMPNO '||V_EMPNO);
 12  EXCEPTION
 13  ----WHEN DUP_VAL_ON_INDEX  THEN
 14  ----D('DUPLICATION RECORDS');
 15  WHEN OTHERS THEN
 16  &D(SQLCODE||'   '||SQLERRM);
 17* END;
 18  /
Enter value for empno: 502
Enter value for ename: ASL
Enter value for job: SALESMAN
Enter value for sal: 122
Enter value for deptno: 30
-1   ORA-00001: unique constraint (SCOTT.PK_EMP) violated


PL/SQL procedure successfully completed.

SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL> ED
Wrote file afiedt.buf

  1  DECLARE
  2  V_EMPNO EMP.EMPNO%TYPE :=&EMPNO;
  3  V_ENAME EMP.ENAME%TYPE:='&ENAME';
  4  V_JOB   EMP.JOB%TYPE:='&JOB';
  5  V_SAL   EMP.SAL%TYPE:=&SAL;
  6  V_DEPTNO EMP.DEPTNO%TYPE:=&DEPTNO;
```

```
  7  BEGIN
  8  INSERT INTO EMP(EMPNO,ENAME,JOB,SAL,DEPTNO)
  9  VALUES(V_EMPNO,V_ENAME,V_JOB,V_SAL,V_DEPTNO);
 10  COMMIT;
 11  &D('RECORD CREATED WITH EMPNO '||V_EMPNO);
 12  EXCEPTION
 13  WHEN OTHERS THEN
 14  &D(SQLCODE||'   '||SQLERRM);
 15  WHEN DUP_VAL_ON_INDEX  THEN
 16  &D('DUPLICATION RECORDS');
 17* END;
 18  /
Enter value for empno: 502
Enter value for ename: ASD
Enter value for job: SALES
Enter value for sal: 100
Enter value for deptno: 30
WHEN OTHERS THEN
*
ERROR at line 13:
ORA-06550: line 13, column 1:
PLS-00370: OTHERS handler must be last among the exception handlers of a block
ORA-06550: line 0, column 0:
PL/SQL: Compilation unit analysis terminated


SQL>
SQL>
SQL>
SQL>
SQL>
SQL> ED
Wrote file afiedt.buf

  1  DECLARE
  2  V_EMPNO EMP.EMPNO%TYPE :=&EMPNO;
  3  V_ENAME EMP.ENAME%TYPE:='&ENAME';
  4  V_JOB   EMP.JOB%TYPE:='&JOB';
  5  V_SAL   EMP.SAL%TYPE:=&SAL;
  6  V_DEPTNO EMP.DEPTNO%TYPE:=&DEPTNO;
  7  BEGIN
  8  INSERT INTO EMP(EMPNO,ENAME,JOB,SAL,DEPTNO)
  9  VALUES(V_EMPNO,V_ENAME,V_JOB,V_SAL,V_DEPTNO);
 10  COMMIT;
 11  &D('RECORD CREATED WITH EMPNO '||V_EMPNO);
 12  EXCEPTION
 13  WHEN DUP_VAL_ON_INDEX  THEN
 14  ------------------------NESTED  BLOCK-----------------
 15  DECLARE
 16  ID NUMBER :=0;
 17  BEGIN
 18  SELECT MAX(EMPNO)+1 INTO ID FROM SCOTT.EMP;
 19  INSERT INTO EMP(EMPNO,ENAME,JOB,SAL,DEPTNO)
 20  VALUES(ID,V_ENAME,V_JOB,V_SAL,V_DEPTNO);
 21  COMMIT;
 22  &D('RECORD CREATED WITH EMPNO '||ID ||' INSTEAD OF '||V_EMPNO);
 23  END;
 24  ----------------------------END OF NESTED BLOCK-------------
 25  WHEN OTHERS THEN
 26  &D(SQLCODE||'   '||SQLERRM);
 27* END;
 28  /
Enter value for empno: 502
```

```
Enter value for ename: ALI
Enter value for job: SALESMAN
Enter value for sal: 1000
Enter value for deptno: 30
RECORD CREATED WITH EMPNO 7935 INSTEAD OF 502


PL/SQL procedure successfully completed.

SQL>
SQL>
SQL>
SQL>
SQL>
SQL> /
Enter value for empno: 502
Enter value for ename: ADSF
Enter value for job: ALI
Enter value for sal: SALESMAN
Enter value for deptno: 30
V_SAL   EMP.SAL%TYPE:=SALESMAN;
                      *
ERROR at line 5:
ORA-06550: line 5, column 23:
PLS-00201: identifier 'SALESMAN' must be declared
ORA-06550: line 5, column 9:
PL/SQL: Item ignored
ORA-06550: line 9, column 30:
PLS-00320: the declaration of the type of this expression is incomplete or malformed

ORA-06550: line 9, column 30:
PL/SQL: ORA-00904: "V_SAL": invalid identifier
ORA-06550: line 8, column 1:
PL/SQL: SQL Statement ignored
ORA-06550: line 20, column 25:
PLS-00320: the declaration of the type of this expression is incomplete or malformed

ORA-06550: line 20, column 25:
PL/SQL: ORA-00904: "V_SAL": invalid identifier
ORA-06550: line 19, column 1:
PL/SQL: SQL Statement ignored


SQL>
SQL>
SQL> //
Enter value for empno: 7935
Enter value for ename: ALI
Enter value for job: SALESMAN
Enter value for sal: 1000
Enter value for deptno: 30
RECORD CREATED WITH EMPNO 7936 INSTEAD OF 7935


PL/SQL procedure successfully completed.

SQL>
SQL>
SQL>
SQL>
SQL> /
Enter value for empno: 7938
Enter value for ename: KAMRAN
```

```
Enter value for job: SALESMAN
Enter value for sal: 1000
Enter value for deptno: 30
RECORD CREATED WITH EMPNO 7938


PL/SQL procedure successfully completed.

SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL> ED
Wrote file afiedt.buf

  1  DECLARE
  2  V_EMPNO EMP.EMPNO%TYPE :=&EMPNO;
  3  V_ENAME EMP.ENAME%TYPE:='&ENAME';
  4  V_JOB   EMP.JOB%TYPE:='&JOB';
  5  V_SAL   EMP.SAL%TYPE:=&SAL;
  6  V_DEPTNO EMP.DEPTNO%TYPE:=&DEPTNO;
  7  NEW_ID NUMBER :=0;
  8  BEGIN
  9  SELECT NVL(MAX(EMPNO),1000)+1 INTO NEW_ID FROM SCOTT.EMP;
 10  IF V_EMPNO=NEW_ID
 11  INSERT INTO EMP(EMPNO,ENAME,JOB,SAL,DEPTNO)
 12  VALUES(V_EMPNO,V_ENAME,V_JOB,V_SAL,V_DEPTNO);
 13  COMMIT;
 14  &D('RECORD CREATED WITH EMPNO (VALID)'||V_EMPNO);
 15  ELSE
 16  INSERT INTO EMP(EMPNO,ENAME,JOB,SAL,DEPTNO)
 17  VALUES(NEW_ID,V_ENAME,V_JOB,V_SAL,V_DEPTNO);
 18  COMMIT;
 19  &D('RECORD CREATED WITH EMPNO (INVALID)'||NEW_ID);
 20  END IF;
 21  EXCEPTION
 22  WHEN DUP_VAL_ON_INDEX  THEN
 23  ------------------------NESTED  BLOCK-----------------
 24  DECLARE
 25  ID NUMBER :=0;
 26  BEGIN
 27  SELECT MAX(EMPNO)+1 INTO ID FROM SCOTT.EMP;
 28  INSERT INTO EMP(EMPNO,ENAME,JOB,SAL,DEPTNO)
 29  VALUES(ID,V_ENAME,V_JOB,V_SAL,V_DEPTNO);
 30  COMMIT;
 31  &D('RECORD CREATED WITH EMPNO (DUPLICAITON) '||ID ||' INSTEAD OF '||V_EMPNO);
 32  END;
 33  ----------------------------END OF NESTED BLOCK-------------
 34  WHEN OTHERS THEN
 35  &D(SQLCODE||'   '||SQLERRM);
 36* END;
 37  /
Enter value for empno: 7935
Enter value for ename: ALI
Enter value for job: SALESMAN
Enter value for sal: 1000
Enter value for deptno: 30
INSERT INTO EMP(EMPNO,ENAME,JOB,SAL,DEPTNO)
*
```

```
ERROR at line 11:
ORA-06550: line 11, column 1:
PLS-00103: Encountered the symbol "INSERT" when expecting one of the following:
. ( * @ % & - + / at mod remainder rem then
<an exponent (**)> and or || multiset
The symbol "then" was substituted for "INSERT" to continue.


SQL>
SQL>
SQL> ED
Wrote file afiedt.buf

  1  DECLARE
  2  V_EMPNO EMP.EMPNO%TYPE :=&EMPNO;
  3  V_ENAME EMP.ENAME%TYPE:='&ENAME';
  4  V_JOB   EMP.JOB%TYPE:='&JOB';
  5  V_SAL   EMP.SAL%TYPE:=&SAL;
  6  V_DEPTNO EMP.DEPTNO%TYPE:=&DEPTNO;
  7  NEW_ID NUMBER :=0;
  8  BEGIN
  9  SELECT NVL(MAX(EMPNO),1000)+1 INTO NEW_ID FROM SCOTT.EMP;
 10  IF V_EMPNO=NEW_ID  THEN
 11  INSERT INTO EMP(EMPNO,ENAME,JOB,SAL,DEPTNO)
 12  VALUES(V_EMPNO,V_ENAME,V_JOB,V_SAL,V_DEPTNO);
 13  COMMIT;
 14  &D('RECORD CREATED WITH EMPNO (VALID)'||V_EMPNO);
 15  ELSE
 16  INSERT INTO EMP(EMPNO,ENAME,JOB,SAL,DEPTNO)
 17  VALUES(NEW_ID,V_ENAME,V_JOB,V_SAL,V_DEPTNO);
 18  COMMIT;
 19  &D('RECORD CREATED WITH EMPNO (INVALID)'||NEW_ID);
 20  END IF;
 21  EXCEPTION
 22  WHEN DUP_VAL_ON_INDEX  THEN
 23  -------------------------NESTED  BLOCK-----------------
 24  DECLARE
 25  ID NUMBER :=0;
 26  BEGIN
 27  SELECT MAX(EMPNO)+1 INTO ID FROM SCOTT.EMP;
 28  INSERT INTO EMP(EMPNO,ENAME,JOB,SAL,DEPTNO)
 29  VALUES(ID,V_ENAME,V_JOB,V_SAL,V_DEPTNO);
 30  COMMIT;
 31  &D('RECORD CREATED WITH EMPNO (DUPLICAITON) '||ID ||' INSTEAD OF '||V_EMPNO);
 32  END;
 33  ----------------------------END OF NESTED BLOCK-------------
 34  WHEN OTHERS THEN
 35  &D(SQLCODE||'   '||SQLERRM);
 36* END;
SQL> /
Enter value for empno: 7935
Enter value for ename: ALI
Enter value for job: SALESMAN
Enter value for sal: 100
Enter value for deptno: 30
RECORD CREATED WITH EMPNO (INVALID)7939


PL/SQL procedure successfully completed.

SQL>
SQL>
SQL>
```

```
SQL> /
Enter value for empno: 7940
Enter value for ename: ALI
Enter value for job: SALESMAN
Enter value for sal: 1000
Enter value for deptno: 30
RECORD CREATED WITH EMPNO (VALID)7940


PL/SQL procedure successfully completed.

SQL>
SQL>
SQL>
SQL>
SQL> /
Enter value for empno: 7945
Enter value for ename: ALI
Enter value for job: SALESMAN
Enter value for sal: 1000
Enter value for deptno: 30
RECORD CREATED WITH EMPNO (INVALID)7941


PL/SQL procedure successfully completed.

SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL> ED
Wrote file afiedt.buf

  1  DECLARE
  2  V_EMPNO EMP.EMPNO%TYPE :=&EMPNO;
  3  V_ENAME EMP.ENAME%TYPE:='&ENAME';
  4  V_JOB   EMP.JOB%TYPE:='&JOB';
  5  V_SAL   EMP.SAL%TYPE:=&SAL;
  6  V_DEPTNO EMP.DEPTNO%TYPE:=&DEPTNO;
  7  NEW_ID NUMBER :=0;
  8  BEGIN
  9  IF V_EMPNO<=8000 THEN
 10  SELECT NVL(MAX(EMPNO),1000)+1 INTO NEW_ID FROM SCOTT.EMP;
 11  IF V_EMPNO=NEW_ID  THEN
 12  INSERT INTO EMP(EMPNO,ENAME,JOB,SAL,DEPTNO)
 13  VALUES(V_EMPNO,V_ENAME,V_JOB,V_SAL,V_DEPTNO);
 14  COMMIT;
 15  &D('RECORD CREATED WITH EMPNO (VALID)'||V_EMPNO);
 16  ELSE
 17  INSERT INTO EMP(EMPNO,ENAME,JOB,SAL,DEPTNO)
 18  VALUES(NEW_ID,V_ENAME,V_JOB,V_SAL,V_DEPTNO);
 19  COMMIT;
 20  &D('RECORD CREATED WITH EMPNO (INVALID)'||NEW_ID);
 21  END IF;
 22  ELSE
 23  &D('OUT OF RANGE EMPNO');
 24  END IF;
 25  EXCEPTION
 26  WHEN DUP_VAL_ON_INDEX  THEN
```

```
 27  ------------------------NESTED  BLOCK-----------------
 28  DECLARE
 29  ID NUMBER :=0;
 30  BEGIN
 31  SELECT MAX(EMPNO)+1 INTO ID FROM SCOTT.EMP;
 32  INSERT INTO EMP(EMPNO,ENAME,JOB,SAL,DEPTNO)
 33  VALUES(ID,V_ENAME,V_JOB,V_SAL,V_DEPTNO);
 34  COMMIT;
 35  &D('RECORD CREATED WITH EMPNO (DUPLICAITON) '||ID ||' INSTEAD OF '||V_EMPNO);
 36  END;
 37  ----------------------------END OF NESTED BLOCK-------------
 38  WHEN OTHERS THEN
 39  &D(SQLCODE||'   '||SQLERRM);
 40* END;
 41  /
Enter value for empno: 8001
Enter value for ename: AI
Enter value for job: SALESMAN
Enter value for sal: 1000
Enter value for deptno: 30
OUT OF RANGE EMPNO


PL/SQL procedure successfully completed.

SQL>
SQL> /
Enter value for empno: 7940
Enter value for ename: KAMRAN
Enter value for job: SALESMAN
Enter value for sal: 1000
Enter value for deptno: 90
-2291   ORA-02291: integrity constraint (SCOTT.FK_DEPTNO) violated - parent key not
found

PL/SQL procedure successfully completed.

SQL>
SQL>
SQL>
SQL>
SQL>
SQL> ED
Wrote file afiedt.buf

  1  DECLARE
  2  ---------------USER DEFINE EXCEPTION WITH CODE------------------
  3  MASTER_NOT_FOUND EXCEPTION;
  4  PRAGMA EXCEPTION_INIT(MASTER_NOT_FOUND,-2291);
  5  V_EMPNO EMP.EMPNO%TYPE :=&EMPNO;
  6  V_ENAME EMP.ENAME%TYPE:='&ENAME';
  7  V_JOB   EMP.JOB%TYPE:='&JOB';
  8  V_SAL   EMP.SAL%TYPE:=&SAL;
  9  V_DEPTNO EMP.DEPTNO%TYPE:=&DEPTNO;
 10  NEW_ID NUMBER :=0;
 11  BEGIN
 12  IF V_EMPNO<=8000 THEN
 13  SELECT NVL(MAX(EMPNO),1000)+1 INTO NEW_ID FROM SCOTT.EMP;
 14  IF V_EMPNO=NEW_ID  THEN
 15  INSERT INTO EMP(EMPNO,ENAME,JOB,SAL,DEPTNO)
 16  VALUES(V_EMPNO,V_ENAME,V_JOB,V_SAL,V_DEPTNO);
 17  COMMIT;
 18  &D('RECORD CREATED WITH EMPNO (VALID)'||V_EMPNO);
```

```
 19  ELSE
 20  INSERT INTO EMP(EMPNO,ENAME,JOB,SAL,DEPTNO)
 21  VALUES(NEW_ID,V_ENAME,V_JOB,V_SAL,V_DEPTNO);
 22  COMMIT;
 23  &D('RECORD CREATED WITH EMPNO (INVALID)'||NEW_ID);
 24  END IF;
 25  ELSE
 26  &D('OUT OF RANGE EMPNO');
 27  END IF;
 28  EXCEPTION
 29  WHEN DUP_VAL_ON_INDEX  THEN
 30  ------------------------NESTED  BLOCK-----------------
 31  DECLARE
 32  ID NUMBER :=0;
 33  BEGIN
 34  SELECT MAX(EMPNO)+1 INTO ID FROM SCOTT.EMP;
 35  INSERT INTO EMP(EMPNO,ENAME,JOB,SAL,DEPTNO)
 36  VALUES(ID,V_ENAME,V_JOB,V_SAL,V_DEPTNO);
 37  COMMIT;
 38  &D('RECORD CREATED WITH EMPNO (DUPLICAITON) '||ID ||' INSTEAD OF '||V_EMPNO);
 39  END;
 40  ----------------------------END OF NESTED BLOCK-------------
 41  WHEN MASTER_NOT_FOUND THEN
 42  &D('MASTER KEY NOT FOUND....');
 43  WHEN OTHERS THEN
 44  &D(SQLCODE||'   '||SQLERRM);
 45* END;
 46  /
Enter value for empno: 7945
Enter value for ename: ALI
Enter value for job: SALESMNA
Enter value for sal: 1000
Enter value for deptno: 90
MASTER KEY NOT FOUND....


PL/SQL procedure successfully completed.

SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL> ED
Wrote file afiedt.buf

  1  DECLARE
  2  ---------------USER DEFINE EXCEPTION WITH CODE-----------------
  3  MASTER_NOT_FOUND EXCEPTION;
  4  PRAGMA EXCEPTION_INIT(MASTER_NOT_FOUND,-2291);
  5  V_EMPNO EMP.EMPNO%TYPE :=&EMPNO;
  6  V_ENAME EMP.ENAME%TYPE:='&ENAME';
  7  V_JOB   EMP.JOB%TYPE:='&JOB';
  8  V_SAL   EMP.SAL%TYPE:=&SAL;
  9  V_DEPTNO EMP.DEPTNO%TYPE:=&DEPTNO;
 10  NEW_ID NUMBER :=0;
 11  BEGIN
 12  IF V_EMPNO<=8000 THEN
 13  SELECT NVL(MAX(EMPNO),1000)+1 INTO NEW_ID FROM SCOTT.EMP;
```

```
 14  IF V_EMPNO=NEW_ID  THEN
 15  &D('TRUE CONDITION');
 16  INSERT INTO EMP(EMPNO,ENAME,JOB,SAL,DEPTNO)
 17  VALUES(V_EMPNO,V_ENAME,V_JOB,V_SAL,V_DEPTNO);
 18  COMMIT;
 19  &D('RECORD CREATED WITH EMPNO (VALID)'||V_EMPNO);
 20  ELSE
 21  &D('FALSE CONDITION');
 22  INSERT INTO EMP(EMPNO,ENAME,JOB,SAL,DEPTNO)
 23  VALUES(NEW_ID,V_ENAME,V_JOB,V_SAL,V_DEPTNO);
 24  COMMIT;
 25  &D('RECORD CREATED WITH EMPNO (INVALID)'||NEW_ID);
 26  END IF;
 27  ELSE
 28  &D('OUT OF RANGE EMPNO');
 29  END IF;
 30  EXCEPTION
 31  WHEN DUP_VAL_ON_INDEX  THEN
 32  ------------------------NESTED  BLOCK-----------------
 33  DECLARE
 34  ID NUMBER :=0;
 35  BEGIN
 36  SELECT MAX(EMPNO)+1 INTO ID FROM SCOTT.EMP;
 37  INSERT INTO EMP(EMPNO,ENAME,JOB,SAL,DEPTNO)
 38  VALUES(ID,V_ENAME,V_JOB,V_SAL,V_DEPTNO);
 39  COMMIT;
 40  &D('RECORD CREATED WITH EMPNO (DUPLICAITON) '||ID ||' INSTEAD OF '||V_EMPNO);
 41  END;
 42  ----------------------------END OF NESTED BLOCK-------------
 43  WHEN MASTER_NOT_FOUND THEN
 44  &D('MASTER KEY NOT FOUND....');
 45  WHEN OTHERS THEN
 46  &D(SQLCODE||'   '||SQLERRM);
 47* END;
SQL> /
Enter value for empno: 7945
Enter value for ename: ALI
Enter value for job: SALESM
Enter value for sal: 234
Enter value for deptno: 90
FALSE CONDITION

MASTER KEY NOT FOUND....


PL/SQL procedure successfully completed.

SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL> ED
Wrote file afiedt.buf

  1  DECLARE
  2  ---------------USER DEFINE EXCEPTION WITH CODE------------------
  3  MASTER_NOT_FOUND EXCEPTION;
  4  PRAGMA EXCEPTION_INIT(MASTER_NOT_FOUND,-2291);
  5  V_EMPNO EMP.EMPNO%TYPE :=&EMPNO;
  6  V_ENAME EMP.ENAME%TYPE:='&ENAME';
```

```
  7  V_JOB   EMP.JOB%TYPE:='&JOB';
  8  V_SAL   EMP.SAL%TYPE:=&SAL;
  9  V_DEPTNO EMP.DEPTNO%TYPE:=&DEPTNO;
 10  NEW_ID NUMBER :=0;
 11  BEGIN
 12  IF V_EMPNO<=8000 THEN
 13  SELECT NVL(MAX(EMPNO),1000)+1 INTO NEW_ID FROM SCOTT.EMP;
 14  IF V_EMPNO=NEW_ID  THEN
 15  &D('TRUE CONDITION');
 16  INSERT INTO EMP(EMPNO,ENAME,JOB,SAL,DEPTNO)
 17  VALUES(V_EMPNO,V_ENAME,V_JOB,V_SAL,V_DEPTNO);
 18  COMMIT;
 19  &D('RECORD CREATED WITH EMPNO (VALID)'||V_EMPNO);
 20  ELSE
 21  &D('FALSE CONDITION');
 22  INSERT INTO EMP(EMPNO,ENAME,JOB,SAL,DEPTNO)
 23  VALUES(NEW_ID,V_ENAME,V_JOB,V_SAL,V_DEPTNO);
 24  COMMIT;
 25  &D('RECORD CREATED WITH EMPNO (INVALID)'||NEW_ID);
 26  END IF;
 27  ELSE
 28  &D('OUT OF RANGE EMPNO');
 29  END IF;
 30  EXCEPTION
 31  WHEN DUP_VAL_ON_INDEX  THEN
 32  ------------------------NESTED  BLOCK-----------------
 33  DECLARE
 34  ID NUMBER :=0;
 35  BEGIN
 36  SELECT MAX(EMPNO)+1 INTO ID FROM SCOTT.EMP;
 37  INSERT INTO EMP(EMPNO,ENAME,JOB,SAL,DEPTNO)
 38  VALUES(ID,V_ENAME,V_JOB,V_SAL,V_DEPTNO);
 39  COMMIT;
 40  &D('RECORD CREATED WITH EMPNO (DUPLICAITON) '||ID ||' INSTEAD OF '||V_EMPNO);
 41  END;
 42  ----------------------------END OF NESTED BLOCK-------------
 43  WHEN MASTER_NOT_FOUND THEN
 44  INSERT INTO DEPTNO(DEPTNO,DNAME)
 45  VALUES(V_DEPTNO,'UPDATE_REQ');
 46  COMMIT;
 47  INSERT INTO EMP(EMPNO,ENAME,JOB,SAL,DEPTNO)
 48  VALUES(NEW_ID,V_ENAME,V_JOB,V_SAL,V_DEPTNO);
 49  COMMIT;
 50  &D('RECORD CREATED WITH EMPNO (INVALID DEPTNO)'||NEW_ID);
 51  WHEN OTHERS THEN
 52  &D(SQLCODE||'   '||SQLERRM);
 53* END;
 54  /
Enter value for empno: 7945
Enter value for ename: ALI
Enter value for job: SALESMNA
Enter value for sal: 1000
Enter value for deptno: 41
INSERT INTO DEPTNO(DEPTNO,DNAME)
            *
ERROR at line 44:
ORA-06550: line 44, column 13:
PL/SQL: ORA-00942: table or view does not exist
ORA-06550: line 44, column 1:
PL/SQL: SQL Statement ignored


SQL>
```

```
SQL>
SQL>
SQL>
SQL> ED
Wrote file afiedt.buf

  1  DECLARE
  2  ---------------USER DEFINE EXCEPTION WITH CODE------------------
  3  MASTER_NOT_FOUND EXCEPTION;
  4  PRAGMA EXCEPTION_INIT(MASTER_NOT_FOUND,-2291);
  5  V_EMPNO EMP.EMPNO%TYPE :=&EMPNO;
  6  V_ENAME EMP.ENAME%TYPE:='&ENAME';
  7  V_JOB   EMP.JOB%TYPE:='&JOB';
  8  V_SAL   EMP.SAL%TYPE:=&SAL;
  9  V_DEPTNO EMP.DEPTNO%TYPE:=&DEPTNO;
 10  NEW_ID NUMBER :=0;
 11  BEGIN
 12  IF V_EMPNO<=8000 THEN
 13  SELECT NVL(MAX(EMPNO),1000)+1 INTO NEW_ID FROM SCOTT.EMP;
 14  IF V_EMPNO=NEW_ID  THEN
 15  &D('TRUE CONDITION');
 16  INSERT INTO EMP(EMPNO,ENAME,JOB,SAL,DEPTNO)
 17  VALUES(V_EMPNO,V_ENAME,V_JOB,V_SAL,V_DEPTNO);
 18  COMMIT;
 19  &D('RECORD CREATED WITH EMPNO (VALID)'||V_EMPNO);
 20  ELSE
 21  &D('FALSE CONDITION');
 22  INSERT INTO EMP(EMPNO,ENAME,JOB,SAL,DEPTNO)
 23  VALUES(NEW_ID,V_ENAME,V_JOB,V_SAL,V_DEPTNO);
 24  COMMIT;
 25  &D('RECORD CREATED WITH EMPNO (INVALID)'||NEW_ID);
 26  END IF;
 27  ELSE
 28  &D('OUT OF RANGE EMPNO');
 29  END IF;
 30  EXCEPTION
 31  WHEN DUP_VAL_ON_INDEX  THEN
 32  ------------------------NESTED  BLOCK-----------------
 33  DECLARE
 34  ID NUMBER :=0;
 35  BEGIN
 36  SELECT MAX(EMPNO)+1 INTO ID FROM SCOTT.EMP;
 37  INSERT INTO EMP(EMPNO,ENAME,JOB,SAL,DEPTNO)
 38  VALUES(ID,V_ENAME,V_JOB,V_SAL,V_DEPTNO);
 39  COMMIT;
 40  &D('RECORD CREATED WITH EMPNO (DUPLICAITON) '||ID ||' INSTEAD OF '||V_EMPNO);
 41  END;
 42  ----------------------------END OF NESTED BLOCK-------------
 43  WHEN MASTER_NOT_FOUND THEN
 44  INSERT INTO DEPT(DEPTNO,DNAME)
 45  VALUES(V_DEPTNO,'UPDATE_REQ');
 46  COMMIT;
 47  INSERT INTO EMP(EMPNO,ENAME,JOB,SAL,DEPTNO)
 48  VALUES(NEW_ID,V_ENAME,V_JOB,V_SAL,V_DEPTNO);
 49  COMMIT;
 50  &D('RECORD CREATED WITH EMPNO (INVALID DEPTNO)'||NEW_ID);
 51  WHEN OTHERS THEN
 52  &D(SQLCODE||'   '||SQLERRM);
 53* END;
SQL> /
Enter value for empno: 7945
Enter value for ename: ALI
Enter value for job: SALESMAN
```

```
Enter value for sal: 1000
Enter value for deptno: 41
FALSE CONDITION

RECORD CREATED WITH EMPNO (INVALID DEPTNO)7942


PL/SQL procedure successfully completed.

SQL>
SQL>
SQL>
SQL>
SQL>
SQL> SELECT * FROM DEPT;

    DEPTNO DNAME          LOC

---------- -------------- -------------

        41 UPDATE_REQ

        50 HR             KARACHI

        60 NEW HR         LHR

        10 ACCOUNTING     NEW YORK

        20 RESEARCH       DALLAS

        30 SALES          CHICAGO

        40 OPERATIONS     BOSTON


7 rows selected.

SQL>
SQL>
SQL>
SQL>
SQL> INSERT INTO DEPT(DEPTNO,DNAME)
  2  VALUES(99,'OTHERS');

1 row created.

SQL> COMMIT;

Commit complete.

SQL> ED
Wrote file afiedt.buf

  1* COMMIT
SQL>
SQL> .
SQL>
SQL>
SQL>  DECLARE
  2   ---------------USER DEFINE EXCEPTION WITH CODE------------------
  3   MASTER_NOT_FOUND EXCEPTION;
  4   PRAGMA EXCEPTION_INIT(MASTER_NOT_FOUND,-2291);
  5   V_EMPNO EMP.EMPNO%TYPE :=&EMPNO;
```

```
  6   V_ENAME EMP.ENAME%TYPE:='&ENAME';
  7   V_JOB   EMP.JOB%TYPE:='&JOB';
  8   V_SAL   EMP.SAL%TYPE:=&SAL;
  9   V_DEPTNO EMP.DEPTNO%TYPE:=&DEPTNO;
 10   NEW_ID NUMBER :=0;
 11   BEGIN
 12   IF V_EMPNO<=8000 THEN
 13   SELECT NVL(MAX(EMPNO),1000)+1 INTO NEW_ID FROM SCOTT.EMP;
 14   IF V_EMPNO=NEW_ID  THEN
 15   &D('TRUE CONDITION');
 16   INSERT INTO EMP(EMPNO,ENAME,JOB,SAL,DEPTNO)
 17   VALUES(V_EMPNO,V_ENAME,V_JOB,V_SAL,V_DEPTNO);
 18   COMMIT;
 19   &D('RECORD CREATED WITH EMPNO (VALID)'||V_EMPNO);
 20   ELSE
 21   &D('FALSE CONDITION');
 22   INSERT INTO EMP(EMPNO,ENAME,JOB,SAL,DEPTNO)
 23   VALUES(NEW_ID,V_ENAME,V_JOB,V_SAL,V_DEPTNO);
 24   COMMIT;
 25   &D('RECORD CREATED WITH EMPNO (INVALID)'||NEW_ID);
 26   END IF;
 27   ELSE
 28   &D('OUT OF RANGE EMPNO');
 29   END IF;
 30   EXCEPTION
 31   WHEN DUP_VAL_ON_INDEX  THEN
 32   ------------------------NESTED  BLOCK-----------------
 33   DECLARE
 34   ID NUMBER :=0;
 35   BEGIN
 36   SELECT MAX(EMPNO)+1 INTO ID FROM SCOTT.EMP;
 37   INSERT INTO EMP(EMPNO,ENAME,JOB,SAL,DEPTNO)
 38   VALUES(ID,V_ENAME,V_JOB,V_SAL,V_DEPTNO);
 39   COMMIT;
 40   &D('RECORD CREATED WITH EMPNO (DUPLICAITON) '||ID ||' INSTEAD OF '||V_EMPNO);
 41   END;
 42   ----------------------------END OF NESTED BLOCK-------------
 43   WHEN MASTER_NOT_FOUND THEN
 44   INSERT INTO DEPT(DEPTNO,DNAME)
 45   VALUES(V_DEPTNO,'UPDATE_REQ');
 46   COMMIT;
 47   INSERT INTO EMP(EMPNO,ENAME,JOB,SAL,DEPTNO)
 48   VALUES(NEW_ID,V_ENAME,V_JOB,V_SAL,V_DEPTNO);
 49   COMMIT;
 50   &D('RECORD CREATED WITH EMPNO (INVALID DEPTNO)'||NEW_ID);
 51   WHEN OTHERS THEN
 52   &D(SQLCODE||'   '||SQLERRM);
 53   END;
 54
 55  .
SQL> ED
Wrote file afiedt.buf

  1   DECLARE
  2   ---------------USER DEFINE EXCEPTION WITH CODE-----------------
  3   MASTER_NOT_FOUND EXCEPTION;
  4   PRAGMA EXCEPTION_INIT(MASTER_NOT_FOUND,-2291);
  5   V_EMPNO EMP.EMPNO%TYPE :=&EMPNO;
  6   V_ENAME EMP.ENAME%TYPE:='&ENAME';
  7   V_JOB   EMP.JOB%TYPE:='&JOB';
  8   V_SAL   EMP.SAL%TYPE:=&SAL;
  9   V_DEPTNO EMP.DEPTNO%TYPE:=&DEPTNO;
 10   NEW_ID NUMBER :=0;
```

```
 11   BEGIN
 12   IF V_EMPNO<=8000 THEN
 13   SELECT NVL(MAX(EMPNO),1000)+1 INTO NEW_ID FROM SCOTT.EMP;
 14   IF V_EMPNO=NEW_ID  THEN
 15   &D('TRUE CONDITION');
 16   INSERT INTO EMP(EMPNO,ENAME,JOB,SAL,DEPTNO)
 17   VALUES(V_EMPNO,V_ENAME,V_JOB,V_SAL,V_DEPTNO);
 18   COMMIT;
 19   &D('RECORD CREATED WITH EMPNO (VALID)'||V_EMPNO);
 20   ELSE
 21   &D('FALSE CONDITION');
 22   INSERT INTO EMP(EMPNO,ENAME,JOB,SAL,DEPTNO)
 23   VALUES(NEW_ID,V_ENAME,V_JOB,V_SAL,V_DEPTNO);
 24   COMMIT;
 25   &D('RECORD CREATED WITH EMPNO (INVALID)'||NEW_ID);
 26   END IF;
 27   ELSE
 28   &D('OUT OF RANGE EMPNO');
 29   END IF;
 30   EXCEPTION
 31   WHEN DUP_VAL_ON_INDEX  THEN
 32   ----------------------NESTED  BLOCK-----------------
 33   DECLARE
 34   ID NUMBER :=0;
 35   BEGIN
 36   SELECT MAX(EMPNO)+1 INTO ID FROM SCOTT.EMP;
 37   INSERT INTO EMP(EMPNO,ENAME,JOB,SAL,DEPTNO)
 38   VALUES(ID,V_ENAME,V_JOB,V_SAL,V_DEPTNO);
 39   COMMIT;
 40   &D('RECORD CREATED WITH EMPNO (DUPLICAITON) '||ID ||' INSTEAD OF '||V_EMPNO);
 41   END;
 42   ----------------------------END OF NESTED BLOCK-------------
 43   WHEN MASTER_NOT_FOUND THEN
 44   INSERT INTO EMP(EMPNO,ENAME,JOB,SAL,DEPTNO)
 45   VALUES(NEW_ID,V_ENAME,V_JOB,V_SAL,99);
 46   COMMIT;
 47   &D('RECORD CREATED WITH EMPNO (INVALID DEPTNO)'||NEW_ID);
 48   WHEN OTHERS THEN
 49   &D(SQLCODE||'   '||SQLERRM);
 50*  END;
 51  /
Enter value for empno: 7945
Enter value for ename: ALI
Enter value for job: SALESMAN
Enter value for sal: 1000
Enter value for deptno: 42
FALSE CONDITION

RECORD CREATED WITH EMPNO (INVALID DEPTNO)7943


PL/SQL procedure successfully completed.

SQL>
SQL>
SQL>
SQL>
SQL> SELECT * FROM EMP
  2  WHERE EMPNO=7943;

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

```
----------
      7943 ALI        SALESMAN                                1000
99

SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL> ED
Wrote file afiedt.buf

  1  SELECT * FROM EMP
  2* WHERE EMPNO=7943
SQL> ED
Wrote file afiedt.buf

  1  DECLARE
  2      FIRST_NO NUMBER :=&FIRST_NO;
  3      SECOND_NO NUMBER :=&SECOND_NO;
  4  BEGIN
  5      IF FIRST_NO>SECOND_NO THEN
  6           &D('HIGHEST NO IS(I M IN 1ST TRUE STAT) ....'||FIRST_NO);
  7      ELSIF SECOND_NO>FIRST_NO THEN
  8           &D('HIGHEST NO IS(I M IN 2N TRUE STAT) ....'||SECOND_NO);
  9      ELSE
 10    &D('BOTH ARE EQUALS NUMBERS');
 11     END IF;
 12*   END;
SQL> /
Enter value for first_no: 5
Enter value for second_no: 6
HIGHEST NO IS(I M IN 2N TRUE STAT) ....6


PL/SQL procedure successfully completed.

SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL> /
Enter value for first_no: 7
Enter value for second_no: 2
HIGHEST NO IS(I M IN 1ST TRUE STAT) ....7


PL/SQL procedure successfully completed.

SQL>
SQL>
SQL>
SQL>
SQL>
SQL> /
Enter value for first_no: 5
Enter value for second_no: 5
BOTH ARE EQUALS NUMBERS


PL/SQL procedure successfully completed.
```

```
SQL>
SQL>
SQL>
SQL>
SQL>
SQL> /
Enter value for first_no: A
Enter value for second_no: 5
    FIRST_NO NUMBER :=A;
                     *
ERROR at line 2:
ORA-06550: line 2, column 23:
PLS-00201: identifier 'A' must be declared
ORA-06550: line 2, column 14:
PL/SQL: Item ignored
ORA-06550: line 5, column 8:
PLS-00320: the declaration of the type of this expression is incomplete or malformed

ORA-06550: line 5, column 5:
PL/SQL: Statement ignored


SQL>
SQL>
SQL>
SQL> ED
Wrote file afiedt.buf

  1  DECLARE
  2  ----------------USER DEFINE EXCEPION WITH OUT CODE---------
  3  MY_EXCEPTION EXCEPTION;
  4      FIRST_NO NUMBER :=&FIRST_NO;
  5      SECOND_NO NUMBER :=&SECOND_NO;
  6  BEGIN
  7      IF FIRST_NO>SECOND_NO THEN
  8           &D('HIGHEST NO IS(I M IN 1ST TRUE STAT) ....'||FIRST_NO);
  9      ELSIF SECOND_NO>FIRST_NO THEN
 10           &D('HIGHEST NO IS(I M IN 2N TRUE STAT) ....'||SECOND_NO);
 11      ELSE
 12       RAISE MY_EXCEPTION;
 13     END IF;
 14  EXCEPTION
 15  WHEN MY_EXCEPTION THEN
 16  &D('BOTH ARE EQUALS NUMBERS');
 17*   END;
 18  /
Enter value for first_no: 5
Enter value for second_no: 5
BOTH ARE EQUALS NUMBERS


PL/SQL procedure successfully completed.

SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL> ED
Wrote file afiedt.buf
```

```
  1     DECLARE
  2      V_EMPNO NUMBER :=&EMPNO;
  3      EMP_REC EMP%ROWTYPE;
  4      BEGIN
  5      SELECT * INTO EMP_REC FROM SCOTT.EMP
  6      WHERE EMPNO=V_EMPNO;
  7      &D(EMP_REC.ENAME ||' '||EMP_REC.JOB||'  '||EMP_REC.SAL||'
'||EMP_REC.DEPTNO);
  8*   END;
  9  /
Enter value for empno: 7788
SCOTT ANALYST  3000  20


PL/SQL procedure successfully completed.

SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL> /
Enter value for empno: 7847
   DECLARE
*
ERROR at line 1:
ORA-01403: no data found
ORA-06512: at line 5


SQL>
SQL>
SQL>
SQL>
SQL> ED
Wrote file afiedt.buf

  1     DECLARE
  2      V_EMPNO NUMBER :=&EMPNO;
  3      EMP_REC EMP%ROWTYPE;
  4      BEGIN
  5      SELECT * INTO EMP_REC FROM SCOTT.EMP
  6      WHERE EMPNO=V_EMPNO;
  7      &D(EMP_REC.ENAME ||' '||EMP_REC.JOB||'  '||EMP_REC.SAL||'
'||EMP_REC.DEPTNO);
  8     EXCEPTION
  9        WHEN NO_DATA_FOUND THEN
 10       &D('RECORD NOT EXIST ...');
 11*   END;
 12  /
Enter value for empno: 7458
RECORD NOT EXIST ...


PL/SQL procedure successfully completed.

SQL>
SQL>
SQL>
```

```
SQL>
SQL>
SQL>
SQL>
SQL> SPOOL OFF
```

```
SQL>
SQL>
SQL> DECLARE
  2  .
SQL> ED
Wrote file afiedt.buf

  1  DECLARE
  2  CURSOR C1 IS SELECT * FROM EMP;
  3  BEGIN
  4  FOR I IN C1 LOOP
  5  &D(I.EMPNO||'  '||I.ENAME||'  '||I.JOB||'  '||I.SAL||'  '||I.DEPTNO);
  6  END LOOP;
  7* END;
  8  /
7945  ALI  SALESMAN  1000  30

7947  ALI  SALESMAN  1205  30

7949  KAMRAN  SALESMAN  1000  30

5454  SMITH  CLERK  900  20

7499  ALLEN  SALESMAN  1600  30

7521  WARD  SALESMAN  1250  30

7566  JONES  MANAGER  2975  20

7654  MARTIN  SALESMAN  1250  30

7698  BLAKE  MANAGER  2850  30

7782  CLARK  MANAGER  2450  10

7788  SCOTT  ANALYST  3000  20

7839  KING  PRESIDENT  5000  10

7844  TURNER  SALESMAN  1500  30

7876  ADAMS  CLERK  1100  20

7900  JAMES  CLERK  1000  30

7902  FORD  ANALYST  45666  20

7934  MILLER  CLERK  1300  10


PL/SQL procedure successfully completed.

SQL> ED
Wrote file afiedt.buf

  1  DECLARE
  2  TYPE EMP_TAB IS TABLE OF VARCHAR2(2000) INDEX BY BINARY_INTEGER;
  3  EI EMP_TAB;
  4  CNTR BINARY_INTEGER:=0;
  5  BEGIN
  6  FOR I IN (SELECT * FROM EMP) LOOP
  7  &D(I.EMPNO||'  '||I.ENAME||'  '||I.JOB||'  '||I.SAL||'  '||I.DEPTNO);
  8  END LOOP;
```

```
  9* END;
 10  /
7945  ALI  SALESMAN  1000  30

7947  ALI  SALESMAN  1205  30

7949  KAMRAN  SALESMAN  1000  30

5454  SMITH  CLERK  900  20

7499  ALLEN  SALESMAN  1600  30

7521  WARD  SALESMAN  1250  30

7566  JONES  MANAGER  2975  20

7654  MARTIN  SALESMAN  1250  30

7698  BLAKE  MANAGER  2850  30

7782  CLARK  MANAGER  2450  10

7788  SCOTT  ANALYST  3000  20

7839  KING  PRESIDENT  5000  10

7844  TURNER  SALESMAN  1500  30

7876  ADAMS  CLERK  1100  20

7900  JAMES  CLERK  1000  30

7902  FORD  ANALYST  45666  20

7934  MILLER  CLERK  1300  10


PL/SQL procedure successfully completed.

SQL>
SQL>
SQL> ED
Wrote file afiedt.buf

  1  DECLARE
  2  TYPE EMP_TAB IS TABLE OF VARCHAR2(2000) INDEX BY BINARY_INTEGER;
  3  EI EMP_TAB;
  4  CNTR BINARY_INTEGER:=0;
  5  BEGIN
  6  ----------------FETCHING FROM DATABASE----------------
  7  FOR I IN (SELECT * FROM EMP) LOOP
  8  CNTR := CNTR + 1;
  9  EI(CNTR):=I.EMPNO||' '||I.ENAME||' '||I.JOB||' '||I.SAL||'
'||I.DEPTNO||CHR(10);
 10  END LOOP;
 11  ----------------------DISPLAY----------------------
 12  FOR J  IN 1..CNTR LOOP
 13  &D(EI(J));
 14  END LOOP;
 15* END;
 16  /
7945 ALI SALESMAN 1000 30
```

```
7947 ALI SALESMAN 1205 30

7949 KAMRAN SALESMAN 1000 30

5454 SMITH CLERK 900 20

7499 ALLEN SALESMAN 1600 30

7521 WARD SALESMAN 1250 30

7566 JONES MANAGER 2975 20

7654 MARTIN SALESMAN 1250 30

7698 BLAKE MANAGER 2850 30

7782 CLARK MANAGER 2450 10

7788 SCOTT ANALYST 3000 20

7839 KING PRESIDENT 5000 10

7844 TURNER SALESMAN 1500 30

7876 ADAMS CLERK 1100 20

7900 JAMES CLERK 1000 30

7902 FORD ANALYST 45666 20

7934 MILLER CLERK 1300 10


PL/SQL procedure successfully completed.

SQL> ED
Wrote file afiedt.buf

  1  DECLARE
  2  TYPE EMP_TAB IS TABLE OF VARCHAR2(2000) INDEX BY BINARY_INTEGER;
  3  EI EMP_TAB;
  4  CNTR BINARY_INTEGER:=0;
  5  BEGIN
  6  ----------------FETCHING FROM DATABASE----------------
  7  FOR I IN (SELECT * FROM EMP) LOOP
  8  CNTR := CNTR + 1;
  9  EI(CNTR):=I.EMPNO||' '||I.ENAME||' '||I.JOB||' '||I.SAL||'
'||I.DEPTNO||CHR(10);
 10  END LOOP;
 11  ------------------------DISPLAY-----------------------
 12  FOR J  IN 1..COUNT(EI)   LOOP
 13  &D(EI(J));
 14  END LOOP;
 15* END;
SQL> /
FOR J  IN 1..COUNT(EI)   LOOP
           *
ERROR at line 12:
ORA-06550: line 12, column 14:
PLS-00204: function or pseudo-column 'COUNT' may be used inside a SQL statement only

ORA-06550: line 12, column 1:
PL/SQL: Statement ignored
```

```
SQL>
SQL>
SQL>
SQL> ED
Wrote file afiedt.buf

  1  DECLARE
  2  TYPE EMP_TAB IS TABLE OF VARCHAR2(2000) INDEX BY BINARY_INTEGER;
  3  EI EMP_TAB;
  4  CNTR BINARY_INTEGER:=0;
  5  BEGIN
  6  ----------------FETCHING FROM DATABASE----------------
  7  FOR I IN (SELECT * FROM EMP) LOOP
  8  CNTR := CNTR + 1;
  9  EI(CNTR):=I.EMPNO||' '||I.ENAME||' '||I.JOB||' '||I.SAL||'
'||I.DEPTNO||CHR(10);
 10  END LOOP;
 11  ------------------------DISPLAY-----------------------
 12  FOR J  IN 1..COUNT.EI   LOOP
 13  &D(EI(J));
 14  END LOOP;
 15* END;
SQL> /
FOR J  IN 1..COUNT.EI   LOOP
                *
ERROR at line 12:
ORA-06550: line 12, column 19:
PLS-00103: Encountered the symbol "." when expecting one of the following:
( * & - + / at loop mod remainder rem <an exponent (**)> ||
multiset


SQL>
SQL>
SQL> ED
Wrote file afiedt.buf

  1  DECLARE
  2  TYPE EMP_TAB IS TABLE OF VARCHAR2(2000) INDEX BY BINARY_INTEGER;
  3  EI EMP_TAB;
  4  CNTR BINARY_INTEGER:=0;
  5  BEGIN
  6  ----------------FETCHING FROM DATABASE----------------
  7  FOR I IN (SELECT * FROM EMP) LOOP
  8  CNTR := CNTR + 1;
  9  EI(CNTR):=I.EMPNO||' '||I.ENAME||' '||I.JOB||' '||I.SAL||'
'||I.DEPTNO||CHR(10);
 10  END LOOP;
 11  ------------------------DISPLAY-----------------------
 12  FOR J  IN 1..LAST.EI   LOOP
 13  &D(EI(J));
 14  END LOOP;
 15* END;
SQL> /
FOR J  IN 1..LAST.EI   LOOP
            *
ERROR at line 12:
ORA-06550: line 12, column 14:
PLS-00201: identifier 'LAST.EI' must be declared
ORA-06550: line 12, column 1:
PL/SQL: Statement ignored
```

```
SQL>
SQL>
SQL>
SQL> ED
Wrote file afiedt.buf

  1  DECLARE
  2  TYPE EMP_TAB IS TABLE OF VARCHAR2(2000) INDEX BY BINARY_INTEGER;
  3  EI EMP_TAB;
  4  CNTR BINARY_INTEGER:=0;
  5  BEGIN
  6  ----------------FETCHING FROM DATABASE----------------
  7  FOR I IN (SELECT * FROM EMP) LOOP
  8  CNTR := CNTR + 1;
  9  EI(CNTR):=I.EMPNO||' '||I.ENAME||' '||I.JOB||' '||I.SAL||'
'||I.DEPTNO||CHR(10);
 10  END LOOP;
 11  ------------------------DISPLAY-----------------------
 12  FOR J  IN 1..LAST.EI   LOOP
 13  &D(EI(J));
 14  END LOOP;
 15* END;
SQL> /
FOR J  IN 1..LAST.EI   LOOP
           *
ERROR at line 12:
ORA-06550: line 12, column 14:
PLS-00201: identifier 'LAST.EI' must be declared
ORA-06550: line 12, column 1:
PL/SQL: Statement ignored


SQL> ED
Wrote file afiedt.buf

  1  DECLARE
  2  TYPE EMP_TAB IS TABLE OF VARCHAR2(2000) INDEX BY BINARY_INTEGER;
  3  EI EMP_TAB;
  4  CNTR BINARY_INTEGER:=0;
  5  BEGIN
  6  ----------------FETCHING FROM DATABASE----------------
  7  FOR I IN (SELECT * FROM EMP) LOOP
  8  CNTR := CNTR + 1;
  9  EI(CNTR):=I.EMPNO||' '||I.ENAME||' '||I.JOB||' '||I.SAL||'
'||I.DEPTNO||CHR(10);
 10  END LOOP;
 11  ------------------------DISPLAY-----------------------
 12  FOR J  IN 1..EI.LAST   LOOP
 13  &D(EI(J));
 14  END LOOP;
 15* END;
SQL> /
7945 ALI SALESMAN 1000 30

7947 ALI SALESMAN 1205 30

7949 KAMRAN SALESMAN 1000 30

5454 SMITH CLERK 900 20

7499 ALLEN SALESMAN 1600 30
```

```
7521 WARD SALESMAN 1250 30

7566 JONES MANAGER 2975 20

7654 MARTIN SALESMAN 1250 30

7698 BLAKE MANAGER 2850 30

7782 CLARK MANAGER 2450 10

7788 SCOTT ANALYST 3000 20

7839 KING PRESIDENT 5000 10

7844 TURNER SALESMAN 1500 30

7876 ADAMS CLERK 1100 20

7900 JAMES CLERK 1000 30

7902 FORD ANALYST 45666 20

7934 MILLER CLERK 1300 10


PL/SQL procedure successfully completed.

SQL>
SQL> ED
Wrote file afiedt.buf

  1  DECLARE
  2  TYPE EMP_TAB IS TABLE OF VARCHAR2(2000) INDEX BY BINARY_INTEGER;
  3  EI EMP_TAB;
  4  CNTR BINARY_INTEGER:=0;
  5  BEGIN
  6  ----------------FETCHING FROM DATABASE----------------
  7  FOR I IN (SELECT * FROM EMP) LOOP
  8  CNTR := CNTR + 1;
  9  EI(CNTR):=I.EMPNO||' '||I.ENAME||' '||I.JOB||' '||I.SAL||'
'||I.DEPTNO||CHR(10);
 10  END LOOP;
 11  ------------------------DISPLAY-----------------------
 12  FOR J  IN 1..EI.COUNT   LOOP
 13  &D(EI(J));
 14  END LOOP;
 15* END;
SQL> /
7945 ALI SALESMAN 1000 30

7947 ALI SALESMAN 1205 30

7949 KAMRAN SALESMAN 1000 30

5454 SMITH CLERK 900 20

7499 ALLEN SALESMAN 1600 30

7521 WARD SALESMAN 1250 30

7566 JONES MANAGER 2975 20
```

```
7654 MARTIN SALESMAN 1250 30

7698 BLAKE MANAGER 2850 30

7782 CLARK MANAGER 2450 10

7788 SCOTT ANALYST 3000 20

7839 KING PRESIDENT 5000 10

7844 TURNER SALESMAN 1500 30

7876 ADAMS CLERK 1100 20

7900 JAMES CLERK 1000 30

7902 FORD ANALYST 45666 20

7934 MILLER CLERK 1300 10


PL/SQL procedure successfully completed.

SQL> EED
SP2-0042: unknown command "EED" - rest of line ignored.
SQL> ED
Wrote file afiedt.buf

  1  DECLARE
  2  TYPE EMP_TAB IS TABLE OF VARCHAR2(2000) INDEX BY BINARY_INTEGER;
  3  EI EMP_TAB;
  4  CNTR BINARY_INTEGER:=0;
  5  BEGIN
  6  ----------------FETCHING FROM DATABASE----------------
  7  FOR I IN (SELECT * FROM EMP) LOOP
  8  CNTR := CNTR + 1;
  9  EI(CNTR):=I.EMPNO||' '||I.ENAME||' '||I.JOB||' '||I.SAL||'
'||I.DEPTNO||CHR(10);
 10  END LOOP;
 11  ------------------------DISPLAY-----------------------
 12  &D('TOTAL RECORDS FETCHED ..'||EI.COUNT);
 13  FOR J  IN 1..EI.COUNT   LOOP
 14  &D(EI(J));
 15  END LOOP;
 16* END;
 17  /
TOTAL RECORDS FETCHED ..17

7945 ALI SALESMAN 1000 30

7947 ALI SALESMAN 1205 30

7949 KAMRAN SALESMAN 1000 30

5454 SMITH CLERK 900 20

7499 ALLEN SALESMAN 1600 30

7521 WARD SALESMAN 1250 30

7566 JONES MANAGER 2975 20

7654 MARTIN SALESMAN 1250 30
```

```
7698 BLAKE MANAGER 2850 30

7782 CLARK MANAGER 2450 10

7788 SCOTT ANALYST 3000 20

7839 KING PRESIDENT 5000 10

7844 TURNER SALESMAN 1500 30

7876 ADAMS CLERK 1100 20

7900 JAMES CLERK 1000 30

7902 FORD ANALYST 45666 20

7934 MILLER CLERK 1300 10


PL/SQL procedure successfully completed.

SQL> ED
Wrote file afiedt.buf

  1  DECLARE
  2  TYPE EMP_TAB IS TABLE OF VARCHAR2(2000) INDEX BY BINARY_INTEGER;
  3  EI EMP_TAB;
  4  CNTR BINARY_INTEGER:=0;
  5  BEGIN
  6  ----------------FETCHING FROM DATABASE----------------
  7  FOR I IN (SELECT * FROM EMP) LOOP
  8  CNTR := CNTR + 1;
  9  EI(CNTR):=I.EMPNO||' '||I.ENAME||' '||I.JOB||' '||I.SAL||'
'||I.DEPTNO||CHR(10);
 10  END LOOP;
 11  ------------------------DISPLAY-----------------------
 12  &D('TOTAL RECORDS FETCHED ..'||EI.COUNT);
 13  FOR J  IN 1..EI.LAST   LOOP
 14  &D(EI(J));
 15  END LOOP;
 16* END;
SQL> /
TOTAL RECORDS FETCHED ..17

7945 ALI SALESMAN 1000 30

7947 ALI SALESMAN 1205 30

7949 KAMRAN SALESMAN 1000 30

5454 SMITH CLERK 900 20

7499 ALLEN SALESMAN 1600 30

7521 WARD SALESMAN 1250 30

7566 JONES MANAGER 2975 20

7654 MARTIN SALESMAN 1250 30

7698 BLAKE MANAGER 2850 30
```

```
7782 CLARK MANAGER 2450 10

7788 SCOTT ANALYST 3000 20

7839 KING PRESIDENT 5000 10

7844 TURNER SALESMAN 1500 30

7876 ADAMS CLERK 1100 20

7900 JAMES CLERK 1000 30

7902 FORD ANALYST 45666 20

7934 MILLER CLERK 1300 10


PL/SQL procedure successfully completed.

SQL> ED
Wrote file afiedt.buf

  1  DECLARE
  2  TYPE EMP_TAB IS TABLE OF VARCHAR2(2000) INDEX BY BINARY_INTEGER;
  3  EI EMP_TAB;
  4  CNTR BINARY_INTEGER:=0;
  5  BEGIN
  6  ----------------FETCHING FROM DATABASE----------------
  7  FOR I IN (SELECT * FROM EMP) LOOP
  8  CNTR := CNTR + 1;
  9  EI(CNTR):=I.EMPNO||' '||I.ENAME||' '||I.JOB||' '||I.SAL||'
'||I.DEPTNO||CHR(10);
 10  END LOOP;
 11  ------------------------DISPLAY-----------------------
 12  &D('TOTAL RECORDS FETCHED ..'||EI.COUNT);
 13  FOR J  IN EI.FIRST..EI.LAST   LOOP
 14  &D(EI(J));
 15  END LOOP;
 16* END;
SQL> /
TOTAL RECORDS FETCHED ..17

7945 ALI SALESMAN 1000 30

7947 ALI SALESMAN 1205 30

7949 KAMRAN SALESMAN 1000 30

5454 SMITH CLERK 900 20

7499 ALLEN SALESMAN 1600 30

7521 WARD SALESMAN 1250 30

7566 JONES MANAGER 2975 20

7654 MARTIN SALESMAN 1250 30

7698 BLAKE MANAGER 2850 30

7782 CLARK MANAGER 2450 10

7788 SCOTT ANALYST 3000 20
```

```
7839 KING PRESIDENT 5000 10

7844 TURNER SALESMAN 1500 30

7876 ADAMS CLERK 1100 20

7900 JAMES CLERK 1000 30

7902 FORD ANALYST 45666 20

7934 MILLER CLERK 1300 10


PL/SQL procedure successfully completed.

SQL>
SQL>
SQL>
SQL> ED
Wrote file afiedt.buf

  1  BEGIN
  2  &D('ORACLE...');
  3* END;
SQL> /
ORACLE...


PL/SQL procedure successfully completed.

SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL> ED
Wrote file afiedt.buf

  1  CREATE  PROCEDURE FIRST_PRO IS
  2  BEGIN
  3  &D('ORACLE...');
  4* END;
SQL> /

Procedure created.

SQL>
SQL>
SQL>
SQL> ED
Wrote file afiedt.buf

  1  CREATE  PROCEDURE FIRST_PRO IS
  2  BEGIN
  3  &D('ORACLE...');
  4* END;
SQL>
SQL> /
```

```
CREATE  PROCEDURE FIRST_PRO IS
                  *
ERROR at line 1:
ORA-00955: name is already used by an existing object


SQL> ED
Wrote file afiedt.buf

  1  CREATE OR REPLACE  PROCEDURE FIRST_PRO IS
  2  BEGIN
  3  &D('ORACLE...');
  4* END;
SQL> /

Procedure created.

SQL> ED
Wrote file afiedt.buf

  1  CREATE OR REPLACE  PROCEDURE FIRST_PRO IS
  2  BEGIN
  3  &D('ORACLE...')
  4* END;
SQL> /

Warning: Procedure created with compilation errors.

SQL>
SQL>
SQL>
SQL>
SQL>
SQL> SHOW ERR
Errors for PROCEDURE FIRST_PRO:

LINE/COL ERROR

-------- -----------------------------------------------------------------

4/1      PLS-00103: Encountered the symbol "END" when expecting one of the

         following:

         := . ( % ;

         The symbol ";" was substituted for "END" to continue.


SQL> SHOW ERROR
Errors for PROCEDURE FIRST_PRO:

LINE/COL ERROR

-------- -----------------------------------------------------------------

4/1      PLS-00103: Encountered the symbol "END" when expecting one of the

         following:

         := . ( % ;
```

```
The symbol ";" was substituted for "END" to continue.


SQL> ED
Wrote file afiedt.buf

  1  CREATE OR REPLACE  PROCEDURE FIRST_PRO IS
  2  BEGIN
  3  &D('ORACLE...');
  4* END;
SQL> /

Procedure created.

SQL> EXECUTE FIRST_PRO;
ORACLE...


PL/SQL procedure successfully completed.

SQL>
SQL>
SQL>
SQL>
SQL> EXEC FIRST_PRO
ORACLE...


PL/SQL procedure successfully completed.

SQL>
SQL>
SQL>
SQL>
SQL> CL SCR
SQL> ED
Wrote file afiedt.buf

  1  CREATE OR REPLACE  PROCEDURE FIRST_PRO IS
  2  BEGIN
  3  &D('ORACLE...');
  4* END;
SQL>
SQL> .
SQL> DESC USER_OBJECTS
 Name                                                      Null?    Type
 --------------------------------------------------------- -------- ------------------------------------
 OBJECT_NAME                                                        VARCHAR2(128)
 SUBOBJECT_NAME                                                     VARCHAR2(30)
 OBJECT_ID                                                          NUMBER
 DATA_OBJECT_ID                                                     NUMBER
 OBJECT_TYPE                                                        VARCHAR2(19)
 CREATED                                                            DATE
 LAST_DDL_TIME                                                      DATE
 TIMESTAMP                                                          VARCHAR2(19)
 STATUS                                                             VARCHAR2(7)
 TEMPORARY                                                          VARCHAR2(1)
 GENERATED                                                          VARCHAR2(1)
 SECONDARY                                                          VARCHAR2(1)

SQL> SELECT OBJECT_NAME,OBJECT_TYPE FROM USER_OBJECTS;
```

```
OBJECT_NAME
--------------------------------------------------------------------------------
---------------
OBJECT_TYPE
-------------------

PK_DEPT
INDEX


DEPT
TABLE


EMP
TABLE


PK_EMP
INDEX


BONUS
TABLE


SALGRADE
TABLE


GET_ORD
FUNCTION


INS_REC
FUNCTION


ADD_EMP
PROCEDURE
```

GET_MGR

FUNCTION

GET_JOB

FUNCTION

EMP_TEST

TABLE

EMP_HIST

TABLE

ADD_NEW_EMP

PROCEDURE

EMP_POSTING

PROCEDURE

DEL_REC

PROCEDURE

P1

PACKAGE

P1

PACKAGE BODY

FORWARD_DEC

PACKAGE

FORWARD_DEC

PACKAGE BODY

OVERPACK

PACKAGE

OVERPACK

PACKAGE BODY

BODYLESS_PACK

PACKAGE

SHOW_TXT

PROCEDURE

WRITE_TO_FILE

PROCEDURE

GET_FILE_TXT

PROCEDURE

TEST_JOB

PROCEDURE

DO_EXE_IMM

PROCEDURE

T1

PROCEDURE

CREATE_TABLE

PROCEDURE

```
TEST

TABLE


SHOW_REC

PROCEDURE

OBJECT_NAME

--------------------------------------------------------------------------------
----------------
OBJECT_TYPE

-------------------


LOG_EMP_HIST

TABLE


EMP_VIEW

VIEW


ADD_DEPT

PROCEDURE


EMP_TEMP

TABLE


VU_SAL

VIEW


VU_MGR

VIEW


GET_ID

FUNCTION
```

ADD_R

PROCEDURE

EIMAGE

TABLE

SET_VDO

PROCEDURE

GET_EMP_VDO_LEN

PROCEDURE

EMP_RESUME

TABLE

SYS_LOB0000052750C00002$$

LOB

LOAD_TXT_DATA

PROCEDURE

EMP_INFO

VIEW

CHK_SAL

PROCEDURE

EMP_AUDIT

TABLE

GET_WORDS

FUNCTION

S1

SEQUENCE

GET_TAX

FUNCTION

EMP_COPY

TABLE

JOB_IDS

TABLE

STD

TABLE

TEST1

TABLE

EMP_EXCEPTION

TABLE

EMP_BACKUP

TABLE

EMP_ATTEND

TABLE

T

TABLE

```
BIN$5kHFFkB9QSidzi0ne+Ldyw==$0

TABLE


MY_CODE

PROCEDURE


SQ1

SEQUENCE


FIRST_PRO


OBJECT_NAME

------------------------------------------------------------------------------
----------------
OBJECT_TYPE

-------------------

PROCEDURE



64 rows selected.

SQL> FORMAT OBJECT_NAME A20
SP2-0734: unknown command beginning "FORMAT OBJ..." - rest of line ignored.
SQL> COLUMN OBJECT_NAME FORMAT A20
SQL>
SQL>
SQL> /

OBJECT_NAME          OBJECT_TYPE

-------------------- -------------------

PK_DEPT              INDEX

DEPT                 TABLE

EMP                  TABLE

PK_EMP               INDEX

BONUS                TABLE

SALGRADE             TABLE

GET_ORD              FUNCTION

INS_REC              FUNCTION
```

ADD_EMP	PROCEDURE
GET_MGR	FUNCTION
GET_JOB	FUNCTION
EMP_TEST	TABLE
EMP_HIST	TABLE
ADD_NEW_EMP	PROCEDURE
EMP_POSTING	PROCEDURE
DEL_REC	PROCEDURE
P1	PACKAGE
P1	PACKAGE BODY
FORWARD_DEC	PACKAGE
FORWARD_DEC	PACKAGE BODY
OVERPACK	PACKAGE
OVERPACK	PACKAGE BODY
BODYLESS_PACK	PACKAGE
SHOW_TXT	PROCEDURE
WRITE_TO_FILE	PROCEDURE
GET_FILE_TXT	PROCEDURE
TEST_JOB	PROCEDURE
DO_EXE_IMM	PROCEDURE
T1	PROCEDURE
CREATE_TABLE	PROCEDURE
TEST	TABLE
SHOW_REC	PROCEDURE
LOG_EMP_HIST	TABLE
EMP_VIEW	VIEW
ADD_DEPT	PROCEDURE
EMP_TEMP	TABLE
VU_SAL	VIEW
VU_MGR	VIEW
GET_ID	FUNCTION
ADD_R	PROCEDURE

```
EIMAGE                TABLE
SET_VDO               PROCEDURE
GET_EMP_VDO_LEN       PROCEDURE
EMP_RESUME            TABLE
SYS_LOB0000052750C00  LOB
002$$

LOAD_TXT_DATA         PROCEDURE
EMP_INFO              VIEW
CHK_SAL               PROCEDURE
EMP_AUDIT             TABLE
GET_WORDS             FUNCTION
S1                    SEQUENCE
GET_TAX               FUNCTION
EMP_COPY              TABLE
JOB_IDS               TABLE
STD                   TABLE
TEST1                 TABLE
EMP_EXCEPTION         TABLE
EMP_BACKUP            TABLE
EMP_ATTEND            TABLE
T                     TABLE
BIN$5kHFFkB9QSidzi0n  TABLE
e+Ldyw==$0

MY_CODE               PROCEDURE
SQ1                   SEQUENCE
FIRST_PRO             PROCEDURE

64 rows selected.

SQL> ED
Wrote file afiedt.buf
```

```
  1  SELECT OBJECT_NAME,OBJECT_TYPE FROM USER_OBJECTS
  2* WHERE OBJECT_TYPE='PROCEDURE'
SQL> /

OBJECT_NAME          OBJECT_TYPE

-------------------- -------------------

ADD_EMP              PROCEDURE

ADD_NEW_EMP          PROCEDURE

EMP_POSTING          PROCEDURE

DEL_REC              PROCEDURE

SHOW_TXT             PROCEDURE

WRITE_TO_FILE        PROCEDURE

GET_FILE_TXT         PROCEDURE

TEST_JOB             PROCEDURE

DO_EXE_IMM           PROCEDURE

T1                   PROCEDURE

CREATE_TABLE         PROCEDURE

SHOW_REC             PROCEDURE

ADD_DEPT             PROCEDURE

ADD_R                PROCEDURE

SET_VDO              PROCEDURE

GET_EMP_VDO_LEN      PROCEDURE

LOAD_TXT_DATA        PROCEDURE

CHK_SAL              PROCEDURE

MY_CODE              PROCEDURE

FIRST_PRO            PROCEDURE


20 rows selected.

SQL> DESC USER_SOURCE
 Name                                                      Null?    Type
 --------------------------------------------------------- --------
-----------------------------------
 NAME                                                               VARCHAR2(30)
 TYPE                                                               VARCHAR2(12)
 LINE                                                               NUMBER
 TEXT                                                               VARCHAR2(4000)

SQL>
SQL>
SQL>
```

```
SQL>
SQL>
SQL> SELECT TEXT FROM USER_SOURCE
  2  WHERE NAME='FIRST_PRO';

TEXT

--------------------------------------------------------------------------------
----------------
PROCEDURE FIRST_PRO IS

BEGIN

DBMS_OUTPUT.PUT_LINE('ORACLE...');

END;


SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL> PROCEDURE FIRST_PRO IS
  2  BEGIN
  3  DBMS_OUTPUT.PUT_LINE('ORACLE...');
  4  END;
  5  .
SQL> ED
Wrote file afiedt.buf

  1  CREATE OR REPLACE PROCEDURE FIRST_PRO IS
  2  BEGIN
  3  DBMS_OUTPUT.PUT_LINE('ORACLE...'||'PLSQL');
  4* END;
SQL> /

Procedure created.

SQL> EXEC FIRST_PRO
ORACLE...PLSQL


PL/SQL procedure successfully completed.

SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL> ED
Wrote file afiedt.buf

  1  /*
```

```
  2               PROCEDURE.
  3                    (I) NONE PARAMETRIZED .
  4                    (II) PARAMETERIZED.(IN MODE ,OUT MODE,INOUT MODE)
  5* */
  6  .
SQL>
SQL>
SQL>
SQL> ED
Wrote file afiedt.buf

  1  DECLARE
  2  V_EMPNO EMP.EMPNO%TYPE:=&EMPNO;
  3  EMP_REC EMP.%ROWTYPE;
  4  BEGIN
  5  SELECT * INTO EMP_REC FROM SCOTT.EMP
  6  WHERE EMPNO=V_EMPNO;
  7  &D(EMP_REC.ENAME||'  ||EMP_REC.JOB||'  '||EMP_REC.SAL||'  '||EMP_REC.DEPTNO);
  8* END;
  9  /
Enter value for empno:
ERROR:
ORA-01756: quoted string not properly terminated


SQL>
SQL> ED
Wrote file afiedt.buf

  1  DECLARE
  2  V_EMPNO EMP.EMPNO%TYPE:=&EMPNO;
  3  EMP_REC EMP.%ROWTYPE;
  4  BEGIN
  5  SELECT * INTO EMP_REC FROM SCOTT.EMP
  6  WHERE EMPNO=V_EMPNO;
  7  &D(EMP_REC.ENAME||'  '||EMP_REC.JOB||'  '||EMP_REC.SAL||'  '||EMP_REC.DEPTNO);
  8* END;
SQL> /
Enter value for empno: 7788
EMP_REC EMP.%ROWTYPE;
          *
ERROR at line 3:
ORA-06550: line 3, column 13:
PLS-00103: Encountered the symbol "%" when expecting one of the following:
<an identifier> <a double-quoted delimited-identifier>
The symbol "<an identifier>" was substituted for "%" to continue.


SQL>
SQL>
SQL>
SQL> ED
Wrote file afiedt.buf

  1  DECLARE
  2  V_EMPNO EMP.EMPNO%TYPE:=&EMPNO;
  3  EMP_REC EMP%ROWTYPE;
  4  BEGIN
  5  SELECT * INTO EMP_REC FROM SCOTT.EMP
  6  WHERE EMPNO=V_EMPNO;
  7  &D(EMP_REC.ENAME||'  '||EMP_REC.JOB||'  '||EMP_REC.SAL||'  '||EMP_REC.DEPTNO);
  8* END;
SQL> /
```

```
Enter value for empno: 7788
SCOTT  ANALYST  3000  20


PL/SQL procedure successfully completed.

SQL>
SQL>
SQL>
SQL>
SQL> /
Enter value for empno: 457
DECLARE
*
ERROR at line 1:
ORA-01403: no data found
ORA-06512: at line 5


SQL>
SQL>
SQL>
SQL> ED
Wrote file afiedt.buf

  1  DECLARE
  2  V_EMPNO EMP.EMPNO%TYPE:=&EMPNO;
  3  EMP_REC EMP%ROWTYPE;
  4  BEGIN
  5  SELECT * INTO EMP_REC FROM SCOTT.EMP
  6  WHERE EMPNO=V_EMPNO;
  7  &D(EMP_REC.ENAME||'  '||EMP_REC.JOB||'  '||EMP_REC.SAL||'  '||EMP_REC.DEPTNO);
  8  EXCEPTION
  9  WHEN NO_DATA_FOUND THEN
 10  &D('RECORD NOT FOUND...');
 11* END;
 12  /
Enter value for empno: 458
RECORD NOT FOUND...


PL/SQL procedure successfully completed.

SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL> ED
Wrote file afiedt.buf

  1  CREATE OR REPLACE PROCEDURE SHOW_REC IS
  2  V_EMPNO EMP.EMPNO%TYPE:=&EMPNO;
  3  EMP_REC EMP%ROWTYPE;
  4  BEGIN
  5  SELECT * INTO EMP_REC FROM SCOTT.EMP
  6  WHERE EMPNO=V_EMPNO;
  7  &D(EMP_REC.ENAME||'  '||EMP_REC.JOB||'  '||EMP_REC.SAL||'  '||EMP_REC.DEPTNO);
  8  EXCEPTION
  9  WHEN NO_DATA_FOUND THEN
 10  &D('RECORD NOT FOUND...');
 11* END;
```

```
SQL> /
Enter value for empno: 7788

Procedure created.

SQL> EXEC SHOW_REC;
SCOTT  ANALYST  3000  20


PL/SQL procedure successfully completed.

SQL>  EXEC SHOW_REC;
SCOTT  ANALYST  3000  20


PL/SQL procedure successfully completed.

SQL>  CREATE OR REPLACE PROCEDURE SHOW_REC IS
  2   V_EMPNO EMP.EMPNO%TYPE:=&EMPNO;
  3   EMP_REC EMP%ROWTYPE;
  4   BEGIN
  5   SELECT * INTO EMP_REC FROM SCOTT.EMP
  6   WHERE EMPNO=V_EMPNO;
  7   &D(EMP_REC.ENAME||'  '||EMP_REC.JOB||'  '||EMP_REC.SAL||'  '||EMP_REC.DEPTNO);
  8   EXCEPTION
  9   WHEN NO_DATA_FOUND THEN
 10   &D('RECORD NOT FOUND...');
 11   END;
 12  /
Enter value for empno: 7839

Procedure created.

SQL>  EXEC SHOW_REC;
KING  PRESIDENT  5000  10


PL/SQL procedure successfully completed.

SQL> EXEC SHOW_REC;
KING  PRESIDENT  5000  10


PL/SQL procedure successfully completed.

SQL> CREATE OR REPLACE PROCEDURE SHOW_REC IS
  2  V_EMPNO EMP.EMPNO%TYPE:=&EMPNO;
  3  EMP_REC EMP%ROWTYPE;
  4  BEGIN
  5  SELECT * INTO EMP_REC FROM SCOTT.EMP
  6  WHERE EMPNO=V_EMPNO;
  7  &D(EMP_REC.ENAME||'  '||EMP_REC.JOB||'  '||EMP_REC.SAL||'  '||EMP_REC.DEPTNO);
  8  EXCEPTION
  9  WHEN NO_DATA_FOUND THEN
 10  &D('RECORD NOT FOUND...');
 11  END;
 12  .
SQL> ED
Wrote file afiedt.buf

  1  CREATE OR REPLACE PROCEDURE SHOW_REC(V_EMPNO IN EMP.EMPNO%TYPE) IS
  2  EMP_REC EMP%ROWTYPE;
  3  BEGIN
```

```
  4  SELECT * INTO EMP_REC FROM SCOTT.EMP
  5  WHERE EMPNO=V_EMPNO;
  6  &D(EMP_REC.ENAME||'  '||EMP_REC.JOB||'  '||EMP_REC.SAL||'  '||EMP_REC.DEPTNO);
  7  EXCEPTION
  8  WHEN NO_DATA_FOUND THEN
  9  &D('RECORD NOT FOUND...');
 10* END;
 11  /

Procedure created.

SQL> EXEC SHOW_REC(7788);
SCOTT  ANALYST  3000  20


PL/SQL procedure successfully completed.

SQL> EXEC SHOW_REC(7839);
KING  PRESIDENT  5000  10


PL/SQL procedure successfully completed.

SQL> ED
Wrote file afiedt.buf

  1  CREATE OR REPLACE PROCEDURE SHOW_REC(V_EMPNO EMP.EMPNO%TYPE) IS
  2  EMP_REC EMP%ROWTYPE;
  3  BEGIN
  4  SELECT * INTO EMP_REC FROM SCOTT.EMP
  5  WHERE EMPNO=V_EMPNO;
  6  &D(EMP_REC.ENAME||'  '||EMP_REC.JOB||'  '||EMP_REC.SAL||'  '||EMP_REC.DEPTNO);
  7  EXCEPTION
  8  WHEN NO_DATA_FOUND THEN
  9  &D('RECORD NOT FOUND...');
 10* END;
SQL> /

Procedure created.

SQL> EXEC SHOW_REC(7839);
KING  PRESIDENT  5000  10


PL/SQL procedure successfully completed.

SQL> DESC SHOW_REC
PROCEDURE SHOW_REC
 Argument Name                  Type                    In/Out Default?
 ------------------------------ ----------------------- ------ --------
 V_EMPNO                        NUMBER(4)               IN

SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
```

```
SQL> ED
Wrote file afiedt.buf

  1  CREATE OR REPLACE PROCEDURE SHOW_REC(V_EMPNO EMP.EMPNO%TYPE DEFAULT 7788) IS
  2  EMP_REC EMP%ROWTYPE;
  3  BEGIN
  4  SELECT * INTO EMP_REC FROM SCOTT.EMP
  5  WHERE EMPNO=V_EMPNO;
  6  &D(EMP_REC.ENAME||'  '||EMP_REC.JOB||'  '||EMP_REC.SAL||'  '||EMP_REC.DEPTNO);
  7  EXCEPTION
  8  WHEN NO_DATA_FOUND THEN
  9  &D('RECORD NOT FOUND...');
 10* END;
SQL> /

Procedure created.

SQL>
SQL>
SQL>
SQL> EXEC SHOW_REC;
SCOTT  ANALYST  3000  20


PL/SQL procedure successfully completed.

SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL> DESC SHOW_REC
PROCEDURE SHOW_REC
 Argument Name                  Type                    In/Out Default?
 ------------------------------ ----------------------- ------ --------
 V_EMPNO                        NUMBER(4)               IN     DEFAULT

SQL>
SQL>
SQL>
SQL>
SQL>
SQL> EXEC SHOW_REC(7839);
KING  PRESIDENT  5000  10


PL/SQL procedure successfully completed.

SQL> EXEC SHOW_REC;
SCOTT  ANALYST  3000  20


PL/SQL procedure successfully completed.

SQL> EXEC SHOW_REC(78);
RECORD NOT FOUND...


PL/SQL procedure successfully completed.

SQL>
SQL>
```

```
SQL>
SQL>
SQL> ED
Wrote file afiedt.buf

  1  DECLARE
  2  V_TAB NUMBER :=&TABLE_NO;
  3  CNTR NUMBER :=0;
  4  BEGIN
  5  FOR I  IN 1..10 LOOP
  6  CNTR := I * V_TAB ;
  7  &D(V_TAB||' X '||I||' = '||CNTR);
  8  END LOOP;
  9* END;
 10  /
Enter value for table_no: 5
5 X 1 = 5

5 X 2 = 10

5 X 3 = 15

5 X 4 = 20

5 X 5 = 25

5 X 6 = 30

5 X 7 = 35

5 X 8 = 40

5 X 9 = 45

5 X 10 = 50


PL/SQL procedure successfully completed.

SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL> ED
Wrote file afiedt.buf

  1  CREATE OR REPLACE PROCEDURE TAB_NO
  2  (
  3  V_TAB NUMBER
  4  )
  5  IS
  6  CNTR NUMBER :=0;
  7  BEGIN
  8  FOR I  IN 1..10 LOOP
  9  CNTR := I * V_TAB ;
 10  &D(V_TAB||' X '||I||' = '||CNTR);
 11  END LOOP;
 12* END;
 13  /

Procedure created.
```

```
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL> EXEC TAB_NO(45);
45 X 1 = 45

45 X 2 = 90

45 X 3 = 135

45 X 4 = 180

45 X 5 = 225

45 X 6 = 270

45 X 7 = 315

45 X 8 = 360

45 X 9 = 405

45 X 10 = 450


PL/SQL procedure successfully completed.

SQL>
SQL>
SQL>
SQL>
SQL>
SQL> ED
Wrote file afiedt.buf

  1  DECLARE
  2  V_T NUMBER :=&TABLE_NO;
  3  BEGIN
  4  TAB_NO(V_T);     ----CALLING PROCEDURE
  5* END;
  6  /
Enter value for table_no: 6
6 X 1 = 6

6 X 2 = 12

6 X 3 = 18

6 X 4 = 24

6 X 5 = 30

6 X 6 = 36

6 X 7 = 42
```

```
6 X 8 = 48

6 X 9 = 54

6 X 10 = 60


PL/SQL procedure successfully completed.

SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL> ED
Wrote file afiedt.buf

  1  DECLARE
  2  V_T NUMBER :=&TABLE_NO;
  3  BEGIN
  4  CALL TAB_NO(V_T);     ----CALLING PROCEDURE
  5* END;
SQL> /
Enter value for table_no:
V_T NUMBER :=;
            *
ERROR at line 2:
ORA-06550: line 2, column 14:
PLS-00103: Encountered the symbol ";" when expecting one of the following:
( - + case mod new not null <an identifier>
<a double-quoted delimited-identifier> <a bind variable> avg
count current exists max min prior sql stddev sum variance
execute forall merge time timestamp interval date
<a string literal with character set specification>
<a number> <a single-quoted SQL string> pipe
<an alternatively-quoted string literal with character set specification>
<an alternatively-quoted S


SQL> /
Enter value for table_no: 45
CALL TAB_NO(V_T);     ----CALLING PROCEDURE
     *
ERROR at line 4:
ORA-06550: line 4, column 6:
PLS-00103: Encountered the symbol "TAB_NO" when expecting one of the following:
:= . ( @ % ;
The symbol ":=" was substituted for "TAB_NO" to continue.


SQL>
SQL>
SQL>
SQL>
SQL>
SQL> ED
Wrote file afiedt.buf

  1  CREATE OR REPLACE PROCEDURE TEST
  2  (
  3  V_T NUMBER
```

```
  4  )IS
  5  BEGIN
  6  &D('I M BEFORE CALLING PROCEDURE');
  7  TAB_NO(V_T);     ----CALLING PROCEDURE
  8  &D('I M BEFORE CALLING PROCEDURE');
  9* END;
SQL> /
CREATE OR REPLACE PROCEDURE TEST
*
ERROR at line 1:
ORA-00955: name is already used by an existing object


SQL> ED
Wrote file afiedt.buf

  1  CREATE OR REPLACE PROCEDURE TEST
  2  (
  3  V_T NUMBER
  4  )IS
  5  BEGIN
  6  &D('I M BEFORE CALLING PROCEDURE');
  7  TAB_NO(V_T);     ----CALLING PROCEDURE
  8  &D('I M BEFORE CALLING PROCEDURE');
  9* END;
 10  /
CREATE OR REPLACE PROCEDURE TEST
*
ERROR at line 1:
ORA-00955: name is already used by an existing object


SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL> ED
Wrote file afiedt.buf

  1  CREATE OR REPLACE PROCEDURE TEST1
  2  (
  3  V_T NUMBER
  4  )IS
  5  BEGIN
  6  &D('I M BEFORE CALLING PROCEDURE');
  7  TAB_NO(V_T);     ----CALLING PROCEDURE
  8  &D('I M BEFORE CALLING PROCEDURE');
  9* END;
SQL> /
CREATE OR REPLACE PROCEDURE TEST1
*
ERROR at line 1:
ORA-00955: name is already used by an existing object


SQL>
SQL>
SQL> ED
Wrote file afiedt.buf

  1  CREATE OR REPLACE PROCEDURE TEST2
```

```
  2  (
  3  V_T NUMBER
  4  )IS
  5  BEGIN
  6  &D('I M BEFORE CALLING PROCEDURE');
  7  TAB_NO(V_T);     ----CALLING PROCEDURE
  8  &D('I M BEFORE CALLING PROCEDURE');
  9* END;
SQL> /

Procedure created.

SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL> EXEC TEST2(5);
I M BEFORE CALLING PROCEDURE

5 X 1 = 5

5 X 2 = 10

5 X 3 = 15

5 X 4 = 20

5 X 5 = 25

5 X 6 = 30

5 X 7 = 35

5 X 8 = 40

5 X 9 = 45

5 X 10 = 50

I M BEFORE CALLING PROCEDURE


PL/SQL procedure successfully completed.

SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL> GRANT EXECUTE ON TEST2 TO HR;

Grant succeeded.

SQL>
SQL>
SQL>
SQL>
```

```
SQL>
SQL>
SQL>
SQL> CONN HR/HR
Connected.
USER is "HR"
linesize 100
pagesize 100
long 80
SQL> EXEC SCOTT.TEST2(90);
I M BEFORE CALLING PROCEDURE

90 X 1 = 90

90 X 2 = 180

90 X 3 = 270

90 X 4 = 360

90 X 5 = 450

90 X 6 = 540

90 X 7 = 630

90 X 8 = 720

90 X 9 = 810

90 X 10 = 900

I M BEFORE CALLING PROCEDURE


PL/SQL procedure successfully completed.

SQL>
SQL>
SQL>
SQL>
SQL>
SQL> SHOW USER
USER is "HR"
SQL>
SQL>
SQL>
SQL>
SQL> CONN SYS/ORACLE AS SYSDBA
Connected.
USER is "SYS"
linesize 100
pagesize 100
long 80
SQL>
SQL>
SQL> CREATE USER B63 IDENTIFIED BY B63;

User created.

SQL> GRANT CONNECT,RESOURCE TO B63;

Grant succeeded.
```

```
SQL> CONN B63/B63
Connected.
USER is "B63"
linesize 100
pagesize 100
long 80
SQL>
SQL>
SQL> SHOW USER
USER is "B63"
SQL>
SQL>
SQL> EXEC SCOTT.TEST2(2);
BEGIN SCOTT.TEST2(2); END;

      *
ERROR at line 1:
ORA-06550: line 1, column 7:
PLS-00201: identifier 'SCOTT.TEST2' must be declared
ORA-06550: line 1, column 7:
PL/SQL: Statement ignored


SQL>
SQL>
SQL>
SQL>
SQL> CONN HR/HR
Connected.
USER is "HR"
linesize 100
pagesize 100
long 80
SQL>
SQL>
SQL>
SQL>
SQL>
SQL> ED
Wrote file afiedt.buf

  1  CREATE OR REPLACE PRO_FWD(T NUMBER)IS
  2  BEGIN
  3  SCOTT.TEST2(T); ----CALLING SCOTT PROCEDURE FROM HR SCHMA
  4* END;
  5  /
CREATE OR REPLACE PRO_FWD(T NUMBER)IS
                  *
ERROR at line 1:
ORA-00922: missing or invalid option


SQL>
SQL>
SQL>
SQL> ED
Wrote file afiedt.buf

  1  CREATE OR REPLACE  PROCEDURE PRO_FWD(T NUMBER)IS
  2  BEGIN
  3  SCOTT.TEST2(T); ----CALLING SCOTT PROCEDURE FROM HR SCHMA
  4* END;
```

```
SQL> /

Procedure created.

SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL> SHOW USER
USER is "HR"
SQL> EXEC PRO_FWD(4);
I M BEFORE CALLING PROCEDURE

4 X 1 = 4

4 X 2 = 8

4 X 3 = 12

4 X 4 = 16

4 X 5 = 20

4 X 6 = 24

4 X 7 = 28

4 X 8 = 32

4 X 9 = 36

4 X 10 = 40

I M BEFORE CALLING PROCEDURE


PL/SQL procedure successfully completed.

SQL>
SQL>
SQL> GRANT EXECUTE ON PRO_FWD TO B63;

Grant succeeded.

SQL> CONN B63/B63
Connected.
USER is "B63"
linesize 100
pagesize 100
long 80
SQL>
SQL>
SQL>
SQL> EXEC HR.PRO_FWD(10);
I M BEFORE CALLING PROCEDURE

10 X 1 = 10
```

```
10 X 2 = 20

10 X 3 = 30

10 X 4 = 40

10 X 5 = 50

10 X 6 = 60

10 X 7 = 70

10 X 8 = 80

10 X 9 = 90

10 X 10 = 100

I M BEFORE CALLING PROCEDURE


PL/SQL procedure successfully completed.

SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL> SHOW USER
USER is "B63"
SQL> CONN SYS/ORACLE AS SYSDBA
Connected.
USER is "SYS"
linesize 100
pagesize 100
long 80
SQL>
SQL>
SQL> DROP USER B63 CASCADE;

User dropped.

SQL>
SQL> SHOW USER
USER is "SYS"
SQL> CONN SCOTT/TIGER
Connected.
USER is "SCOTT"
linesize 100
pagesize 100
long 80
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL> SPOOL OFF
```

```
SQL>
SQL> SELECT OBJECT_NAME FROM USER_OBJECTS
  2  WHERE OBJECT_TYPE='PROCEDURE';

OBJECT_NAME

--------------------------------------------------------------------------------
----------------
ADD_EMP

ADD_NEW_EMP

EMP_POSTING

DEL_REC

SHOW_TXT

WRITE_TO_FILE

GET_FILE_TXT

TEST_JOB

DO_EXE_IMM

T1

CREATE_TABLE

SHOW_REC

ADD_DEPT

ADD_R

SET_VDO

GET_EMP_VDO_LEN

LOAD_TXT_DATA

CHK_SAL

TAB_NO

MY_CODE

FIRST_PRO

TEST2


22 rows selected.

SQL>
SQL>
SQL> SELECT TEXT FROM USER_SOURCE
  2  WHERE NAME='TAB_NO';

TEXT

--------------------------------------------------------------------------------
```

```
----------------
PROCEDURE TAB_NO

(

V_TAB NUMBER

)

IS

CNTR NUMBER :=0;

BEGIN

FOR I  IN 1..10 LOOP

CNTR := I * V_TAB ;

DBMS_OUTPUT.PUT_LINE(V_TAB||' X '||I||' = '||CNTR);

END LOOP;

END;


12 rows selected.

SQL>
SQL>
SQL>
SQL>
SQL>
SQL> PROCEDURE TAB_NO
  2  (
  3  V_TAB NUMBER
  4  )
  5  IS
  6  CNTR NUMBER :=0;
  7  BEGIN
  8  FOR I  IN 1..10 LOOP
  9  CNTR := I * V_TAB ;
 10  DBMS_OUTPUT.PUT_LINE(V_TAB||' X '||I||' = '||CNTR);
 11  END LOOP;
 12  END;
 13  .
SQL> ED
Wrote file afiedt.buf

  1  CREATE OR REPLACE PROCEDURE TAB_NO
  2  (
  3  V_TAB IN  NUMBER
  4  )
  5  IS
  6  CNTR NUMBER :=0;
  7  BEGIN
  8  FOR I  IN 1..10 LOOP
  9  CNTR := I * V_TAB ;
 10  DBMS_OUTPUT.PUT_LINE(V_TAB||' X '||I||' = '||CNTR);
 11  END LOOP;
 12* END;
SQL> /
```

```
Procedure created.

SQL> DESC TAB_NO
PROCEDURE TAB_NO
 Argument Name                  Type                    In/Out Default?
 ------------------------------ ----------------------- ------ --------
 V_TAB                          NUMBER                  IN

SQL>
SQL>
SQL>
SQL> ED
Wrote file afiedt.buf

  1  CREATE OR REPLACE PROCEDURE TAB_NO
  2  (
  3  V_TAB IN  NUMBER,RESULT OUT VARCHAR2
  4  )
  5  IS
  6  CNTR NUMBER :=0;
  7  RES VARCHAR2(200);
  8  BEGIN
  9  FOR I  IN 1..10 LOOP
 10  CNTR := I * V_TAB ;
 11  RES := RES ||V_TAB||' X '||I||' = '||CNTR||CHR(10);
 12  END LOOP;
 13  RESULT := RES;
 14* END;
 15  /

Procedure created.

SQL>  DESC TAB_NO
PROCEDURE TAB_NO
 Argument Name                  Type                    In/Out Default?
 ------------------------------ ----------------------- ------ --------
 V_TAB                          NUMBER                  IN
 RESULT                         VARCHAR2                OUT

SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL> VAR R VARCHAR2(4000);
SQL>
SQL>
SQL>
SQL>
SQL>
SQL> EXEC TAB_NO(5,:R)

PL/SQL procedure successfully completed.

SQL>
SQL>
SQL>
SQL>
SQL>
SQL> PRINT
```

```
R

--------------------------------------------------------------------------------
----------------
5 X 1 = 5

5 X 2 = 10

5 X 3 = 15

5 X 4 = 20

5 X 5 = 25

5 X 6 = 30

5 X 7 = 35

5 X 8 = 40

5 X 9 = 45

5 X 10 = 50



SQL>
SQL>
SQL>
SQL>
SQL> PRINT

R

--------------------------------------------------------------------------------
----------------
5 X 1 = 5

5 X 2 = 10

5 X 3 = 15

5 X 4 = 20

5 X 5 = 25

5 X 6 = 30

5 X 7 = 35

5 X 8 = 40

5 X 9 = 45

5 X 10 = 50



SQL> EXEC TAB_NO(15,:R);
```

PL/SQL procedure successfully completed.

SQL> PRINT

R

----
15 X 1 = 15

15 X 2 = 30

15 X 3 = 45

15 X 4 = 60

15 X 5 = 75

15 X 6 = 90

15 X 7 = 105

15 X 8 = 120

15 X 9 = 135

15 X 10 = 150

SQL>
SQL>
SQL>
SQL> CONN HR/HR
Connected.
USER is "HR"
linesize 100
pagesize 100
long 80
SQL> PRINT

R

----
15 X 1 = 15

15 X 2 = 30

15 X 3 = 45

15 X 4 = 60

15 X 5 = 75

15 X 6 = 90

15 X 7 = 105

15 X 8 = 120

15 X 9 = 135

```
15 X 10 = 150



SQL>
SQL>
SQL>
SQL>
SQL> CONN SCOTT/TIGER
Connected.
USER is "SCOTT"
linesize 100
pagesize 100
long 80
SQL> SET AUTOPRINT ON
SQL>
SQL>
SQL> EXEC TAB_NO(15,:R);

PL/SQL procedure successfully completed.


R

--------------------------------------------------------------------------------
----------------
15 X 1 = 15

15 X 2 = 30

15 X 3 = 45

15 X 4 = 60

15 X 5 = 75

15 X 6 = 90

15 X 7 = 105

15 X 8 = 120

15 X 9 = 135

15 X 10 = 150



SQL>
SQL>
SQL> VAR R2 VARCHAR2(4000);
SQL>
SQL>
SQL> EXEC TAB_NO(10,:R2);

PL/SQL procedure successfully completed.


R2
```

```
--------------------------------------------------------------------------------
----------------
10 X 1 = 10

10 X 2 = 20

10 X 3 = 30

10 X 4 = 40

10 X 5 = 50

10 X 6 = 60

10 X 7 = 70

10 X 8 = 80

10 X 9 = 90

10 X 10 = 100



SQL>
SQL>
SQL>
SQL>
SQL> PRINT

R

--------------------------------------------------------------------------------
----------------
15 X 1 = 15

15 X 2 = 30

15 X 3 = 45

15 X 4 = 60

15 X 5 = 75

15 X 6 = 90

15 X 7 = 105

15 X 8 = 120

15 X 9 = 135

15 X 10 = 150



R2

--------------------------------------------------------------------------------
----------------
```

```
10 X 1 = 10

10 X 2 = 20

10 X 3 = 30

10 X 4 = 40

10 X 5 = 50

10 X 6 = 60

10 X 7 = 70

10 X 8 = 80

10 X 9 = 90

10 X 10 = 100



SQL> PRINT R

R

--------------------------------------------------------------------------------
----------------
15 X 1 = 15

15 X 2 = 30

15 X 3 = 45

15 X 4 = 60

15 X 5 = 75

15 X 6 = 90

15 X 7 = 105

15 X 8 = 120

15 X 9 = 135

15 X 10 = 150



SQL> ED
Wrote file afiedt.buf

  1  DECLARE
  2  T_NO NUMBER :=&TABLE_NO;
  3  MY_RES VARCHAR2(2000);
  4  BEGIN
  5  TAB_NO(T_NO,MY_RES);  ----CALLING PROCEDURE
  6  &D(MY_RES);
  7* END;
  8  ?
```

```
  9  .
SQL> ED
Wrote file afiedt.buf

  1  DECLARE
  2  T_NO NUMBER :=&TABLE_NO;
  3  MY_RES VARCHAR2(2000);
  4  BEGIN
  5  TAB_NO(T_NO,MY_RES);  ----CALLING PROCEDURE
  6  &D(MY_RES);
  7* END;
  8  /
Enter value for table_no: 45
45 X 1 = 45
45 X 2 = 90
45 X 3 = 135
45 X 4 = 180
45 X 5 = 225
45 X 6 = 270
45 X 7 = 315
45 X 8 =
360
45 X 9 = 405
45 X 10 = 450


PL/SQL procedure successfully completed.

SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>  CREATE OR REPLACE PROCEDURE TAB_NO
  2   (
  3   V_TAB IN  NUMBER,RESULT OUT VARCHAR2
  4   )
  5   IS
  6   CNTR NUMBER :=0;
  7   RES VARCHAR2(200);
  8   BEGIN
  9   FOR I  IN 1..10 LOOP
 10   CNTR := I * V_TAB ;
 11   RES := RES ||V_TAB||' X '||I||' = '||CNTR||CHR(10);
 12   END LOOP;
 13   RESULT := RES;
 14   END;
 15
 16  .ED
 17  .
SQL> ED
Wrote file afiedt.buf

  1   CREATE OR REPLACE PROCEDURE TAB_NO
  2   (
  3   V_TAB IN OUT VARCHAR2
  4   )
  5   IS
  6   CNTR NUMBER :=0;
  7   RES VARCHAR2(200);
  8   BEGIN
  9   FOR I  IN 1..10 LOOP
```

```
 10   CNTR := I * V_TAB ;
 11   RES := RES ||V_TAB||' X '||I||' = '||CNTR||CHR(10);
 12   END LOOP;
 13   V_TAB := RES;
 14*  END;
 15  /

Procedure created.

SQL>
SQL>
SQL>
SQL> DESC TAB_NO
PROCEDURE TAB_NO
 Argument Name                  Type                    In/Out Default?
 ------------------------------ ----------------------- ------ --------
 V_TAB                          VARCHAR2                IN/OUT

SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>  DECLARE
  2   T_NO NUMBER :=&TABLE_NO;
  3   MY_RES VARCHAR2(2000);
  4   BEGIN
  5   TAB_NO(T_NO,MY_RES);  ----CALLING PROCEDURE
  6   &D(MY_RES);
  7   END;
  8  .
SQL> ED
Wrote file afiedt.buf

  1   DECLARE
  2   MY_RES VARCHAR2(2000):='&TABLE_NO;
  3   BEGIN
  4   TAB_NO(MY_RES);  ----CALLING PROCEDURE
  5   &D(MY_RES);
  6*  END;
  7  /
Enter value for table_no: 45
ERROR:
ORA-01756: quoted string not properly terminated


SQL>
SQL>
SQL> ED
Wrote file afiedt.buf

  1   DECLARE
  2   MY_RES VARCHAR2(2000):='&TABLE_NO';
  3   BEGIN
  4   TAB_NO(MY_RES);  ----CALLING PROCEDURE
  5   &D(MY_RES);
  6*  END;
SQL> /
Enter value for table_no: 45
45 X 1 = 45
```

```
45 X 2 = 90
45 X 3 = 135
45 X 4 = 180
45 X 5 = 225
45 X 6 = 270
45 X 7 = 315
45 X 8 =
360
45 X 9 = 405
45 X 10 = 450


PL/SQL procedure successfully completed.

SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL> ED
Wrote file afiedt.buf

  1  DECLARE
  2  V_TAB NUMBER :=&TABLE_NO;
  3  CNTR NUMBER :=0;
  4  BEGIN
  5  FOR I IN 1..10 LOOP
  6  CNTR := V_TAB * I ;
  7  &D(V_TAB||' X '||I||' = '||CNTR);
  8  END LOOP;
  9* END;
 10  /
Enter value for table_no: 5
5 X 1 = 5

5 X 2 = 10

5 X 3 = 15

5 X 4 = 20

5 X 5 = 25

5 X 6 = 30

5 X 7 = 35

5 X 8 = 40

5 X 9 = 45

5 X 10 = 50


PL/SQL procedure successfully completed.

SQL>
SQL>
```

```
SQL>
SQL>
SQL>
SQL> ED
Wrote file afiedt.buf

  1  CREATE OR REPLACE FUNCTION GET_TABLE
  2  (
  3  V_TAB NUMBER
  4  )RETURN VARCHAR2 IS
  5  CNTR NUMBER :=0;
  6  RES VARCHAR2(2000);
  7  BEGIN
  8  FOR I IN 1..10 LOOP
  9  CNTR := V_TAB * I ;
 10  RES := RES ||V_TAB||' X '||I||' = '||CNTR ||CHR(10);
 11  END LOOP;
 12  RETURN(RES);
 13* END;
 14  /

Function created.

SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL> SELECT OBJECT_NAME FROM USER_OBJECTS
  2  WHERE OBJECT_TYPE='FUNCTION';

OBJECT_NAME

--------------------------------------------------------------------------------
----------------
GET_ORD

INS_REC

GET_MGR

GET_JOB

GET_ID

GET_WORDS

GET_TAX

GET_TABLE


8 rows selected.

SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
```

```
SQL>
SQL>
SQL> SELECT GET_TABLE(78) FROM DUAL;

GET_TABLE(78)

--------------------------------------------------------------------------------
----------------
78 X 1 = 78

78 X 2 = 156

78 X 3 = 234

78 X 4 = 312

78 X 5 = 390

78 X 6 = 468

78 X 7 = 546

78 X 8 = 624

78 X 9 = 702

78 X 10 = 780



SQL>
SQL>
SQL>
SQL>
SQL>
SQL> EXECUTE :R := GET_TABLE(90);

PL/SQL procedure successfully completed.


R

--------------------------------------------------------------------------------
----------------
90 X 1 = 90

90 X 2 = 180

90 X 3 = 270

90 X 4 = 360

90 X 5 = 450

90 X 6 = 540

90 X 7 = 630

90 X 8 = 720

90 X 9 = 810
```

```
90 X 10 = 900



SQL>
SQL>
SQL>
SQL>
SQL> ED
Wrote file afiedt.buf

  1     OBJ1(RES);        ----PROCEDURE
  2*  RES := OBJ2;  ----FUNCTION
  3  .
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL> ED
Wrote file afiedt.buf

  1  DECLARE
  2  V_TAB NUMBER :=&TABLE_NO;
  3  RESULT VARCHAR2(200);
  4  BEGIN
  5  RESULT := GET_TABLE(V_TAB); ---CALLING FUNCTION;
  6  &d(RESULT);
  7* END;
  8  /
Enter value for table_no: 65
65 X 1 = 65
65 X 2 = 130
65 X 3 = 195
65 X 4 = 260
65 X 5 = 325
65 X 6 = 390
65 X 7 = 455
65 X 8 =
520
65 X 9 = 585
65 X 10 = 650


PL/SQL procedure successfully completed.

SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL> ED
```

```
Wrote file afiedt.buf

  1  CREATE OR REPLACE FUNCTION GET_ID RETURN NUMBER IS
  2  ID NUMBER :=0;
  3  BEGIN
  4  SELECT MAX(EMPNO)+1 INTO ID FROM SCOTT.EMP;
  5  RETURN(ID);
  6* END;
  7  /

Function created.

SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL> ED
Wrote file afiedt.buf

  1  CREATE OR REPLACE FUNCTION GET_ID  IS
  2  ID NUMBER :=0;
  3  BEGIN
  4  SELECT MAX(EMPNO)+1 INTO ID FROM SCOTT.EMP;
  5  RETURN(ID);
  6* END;
SQL> /

Warning: Function created with compilation errors.

SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL> SHOW ERR
Errors for FUNCTION GET_ID:

LINE/COL ERROR

-------- -----------------------------------------------------------------

1/18     PLS-00103: Encountered the symbol "IS" when expecting one of the
         following:
         ( return compress compiled wrapped


3/1      PLS-00103: Encountered the symbol "BEGIN" when expecting one of
         the following:
         end function package pragma private procedure subtype type
         use <an identifier> <a double-quoted delimited-identifier>
         form current cursor
```

```
6/4       PLS-00103: Encountered the symbol "end-of-file" when expecting
          one of the following:
          end not pragma final instantiable order overriding static
          member constructor map


SQL>
SQL>
SQL>
SQL>
SQL> ED
Wrote file afiedt.buf

  1  CREATE OR REPLACE FUNCTION GET_ID RETURN NUMBER IS
  2  ID NUMBER :=0;
  3  BEGIN
  4  SELECT MAX(EMPNO)+1 INTO ID FROM SCOTT.EMP;
  5  RETURN(ID);
  6* END;
SQL> /

Function created.

SQL>
SQL>
SQL>
SQL> ED
Wrote file afiedt.buf

  1  CREATE OR REPLACE FUNCTION GET_ID RETURN NUMBER IS
  2  ID NUMBER :=0;
  3  BEGIN
  4  SELECT MAX(EMPNO)+1 INTO ID FROM SCOTT.EMP;
  5* END;
  6
  7  /

Function created.

SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL> SELECT GET_ID FROM DUAL;
SELECT GET_ID FROM DUAL
       *
ERROR at line 1:
ORA-06503: PL/SQL: Function returned without value
ORA-06512: at "SCOTT.GET_ID", line 5


SQL>
SQL>
SQL>
SQL>
```

```
SQL>
SQL>
SQL> ED
Wrote file afiedt.buf

  1* SELECT GET_ID FROM DUAL
SQL>
SQL> CREATE OR REPLACE FUNCTION GET_ID RETURN NUMBER IS
  2  ID NUMBER :=0;
  3  BEGIN
  4  SELECT MAX(EMPNO)+1 INTO ID FROM SCOTT.EMP;
  5  END;
  6  .
SQL> ED
Wrote file afiedt.buf

  1  CREATE OR REPLACE FUNCTION GET_ID RETURN NUMBER IS
  2  ID NUMBER :=0;
  3  BEGIN
  4  SELECT MAX(EMPNO)+1 INTO ID FROM SCOTT.EMP;
  5  RETURN(ID);
  6* END;
  7  /

Function created.

SQL>
SQL>
SQL>
SQL> SELECT  GET_ID FROM DUAL;

    GET_ID

----------

      7950


SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL> CREATE OR REPLACE FUNCTION GET_ID RETURN NUMBER IS
  2  ID NUMBER :=0;
  3  BEGIN
  4  SELECT MAX(EMPNO)+1 INTO ID FROM SCOTT.EMP;
  5  RETURN(ID);
  6  END;
  7  .
SQL> ED
Wrote file afiedt.buf

  1  CREATE OR REPLACE FUNCTION GET_ID(T_NAME VARCHAR2) RETURN NUMBER IS
  2  ID NUMBER :=0;
  3  BEGIN
  4  IF T_NAME='EMP' THEN
  5  SELECT MAX(EMPNO)+1 INTO ID FROM SCOTT.EMP;
```

```
  6  ELSIF T_NAME='DEPT' THEN
  7  SELECT MAX(DEPTNO)+10 INTO ID FROM SCOTT.DEPT;
  8  ELSIF T_NAME ='SALRGADE' THEN
  9  SELECT MAX(GRADE)+1 INTO ID FROM SCOTT.SALGRADE;
 10  END IF;
 11  RETURN(ID);
 12* END;
 13  /

Function created.

SQL> SELECT GET_ID('EMP') FROM DUAL;

GET_ID('EMP')

-------------

         7950


SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL> SELECT GET_ID('DEPT') FROM DUAL;

GET_ID('DEPT')

--------------

           109


SQL>
SQL>
SQL>
SQL>
SQL>
SQL> SELECT GET_ID('SALGRADE') FROM DUAL;

GET_ID('SALGRADE')

------------------

                 0


SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL> ED
Wrote file afiedt.buf

  1* SELECT GET_ID('SALRGADE') FROM DUAL
SQL> /

GET_ID('SALRGADE')
```

```
-----------------

                6


SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL> ED
Wrote file afiedt.buf

  1  CREATE OR REPLACE PROCEDURE ADD_EMP
  2  (
  3      V_EMPNO EMP.EMPNO%TYPE,
  4      V_ENAME EMP.ENAME%TYPE,
  5      V_JOB   EMP.JOB%TYPE,
  6      V_SAL   EMP.SAL%TYPE,
  7      V_DEPTNO EMP.DEPTNO%TYPE
  8  ) IS
  9     BEGIN
 10      INSERT INTO EMP(EMPNO,ENAME,JOB,SAL,DEPTNO)
 11      VALUES(V_EMPNO,V_ENAME,V_JOB,V_SAL,V_DEPTNO);
 12     COMMIT;
 13     &D('RECORD CREATED WITH EMPNO '||V_EMPNO);
 14     EXCEPTION
 15     WHEN DUP_VAL_ON_INDEX  THEN
 16     ------------------------NESTED  BLOCK-----------------
 17     DECLARE
 18     ID NUMBER :=0;
 19     BEGIN
 20     SELECT MAX(EMPNO)+1 INTO ID FROM SCOTT.EMP;
 21     INSERT INTO EMP(EMPNO,ENAME,JOB,SAL,DEPTNO)
 22     VALUES(ID,V_ENAME,V_JOB,V_SAL,V_DEPTNO);
 23     COMMIT;
 24     &D('RECORD CREATED WITH EMPNO '||ID ||' INSTEAD OF '||V_EMPNO);
 25     END;
 26     ----------------------------END OF NESTED BLOCK-------------
 27     WHEN OTHERS THEN
 28     &D(SQLCODE||'   '||SQLERRM);
 29*   END;
 30  /

Procedure created.

SQL> DESC ADD_EMP
PROCEDURE ADD_EMP
 Argument Name                  Type                    In/Out Default?
 ------------------------------ ----------------------- ------ --------
 V_EMPNO                        NUMBER(4)               IN
 V_ENAME                        VARCHAR2(10)            IN
 V_JOB                          VARCHAR2(9)             IN
 V_SAL                          NUMBER(7,2)             IN
 V_DEPTNO                       NUMBER(2)               IN

SQL>
SQL>
SQL>
SQL>
SQL>
```

```
SQL> EXEC ADD_EMP(7788,USER,'SALESMAN',1000,30);
RECORD CREATED WITH EMPNO 7950 INSTEAD OF 7788


PL/SQL procedure successfully completed.

SQL> EXEC
ADD_EMP(V_DEPTNO=>30,V_ENAME=>'ALI',V_JOB=>'SALESMAN',V_SAL=>1000,V_EMPNO=>7788);
RECORD CREATED WITH EMPNO 7951 INSTEAD OF 7788


PL/SQL procedure successfully completed.

SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL> ED
Wrote file afiedt.buf

  1  CREATE OR REPLACE PROCEDURE ADD_EMP
  2  (
  3      V_EMPNO EMP.EMPNO%TYPE,
  4      V_ENAME EMP.ENAME%TYPE,
  5      V_JOB   EMP.JOB%TYPE,
  6      V_SAL   EMP.SAL%TYPE,
  7      V_DEPTNO EMP.DEPTNO%TYPE
  8  ) IS
  9     BEGIN
 10      INSERT INTO EMP(EMPNO,ENAME,JOB,SAL,DEPTNO)
 11      VALUES(V_EMPNO,V_ENAME,V_JOB,V_SAL,V_DEPTNO);
 12     COMMIT;
 13     &D('RECORD CREATED WITH EMPNO '||V_EMPNO);
 14     EXCEPTION
 15     WHEN DUP_VAL_ON_INDEX  THEN
 16     ------------------------NESTED  BLOCK-----------------
 17     DECLARE
 18     ID NUMBER :=0;
 19     BEGIN
 20    ----- SELECT MAX(EMPNO)+1 INTO ID FROM SCOTT.EMP;
 21  ID := GET_ID('EMP'); -----CALLING PUBLIC FUNCTION
 22     INSERT INTO EMP(EMPNO,ENAME,JOB,SAL,DEPTNO)
 23     VALUES(ID,V_ENAME,V_JOB,V_SAL,V_DEPTNO);
 24     COMMIT;
 25     &D('RECORD CREATED WITH EMPNO '||ID ||' INSTEAD OF '||V_EMPNO);
 26     END;
 27     ----------------------------END OF NESTED BLOCK-------------
 28     WHEN OTHERS THEN
 29     &D(SQLCODE||'   '||SQLERRM);
 30*   END;
 31  /

Procedure created.

SQL> EXEC ADD_EMP(7788,USER,'SALESMAN',1000,30);
```

```
RECORD CREATED WITH EMPNO 7952 INSTEAD OF 7788


PL/SQL procedure successfully completed.

SQL>
SQL>
SQL> ED
Wrote file afiedt.buf

  1  CREATE OR REPLACE PROCEDURE NEW_ID(ID OUT NUMBER) IS
  2  BEGIN
  3  SELECT MAX(EMPNO)+1 INTO ID FROM SCOTT.EMP;
  4* END;
  5  /

Procedure created.

SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL> ED
Wrote file afiedt.buf

  1  CREATE OR REPLACE PROCEDURE ADD_EMP
  2  (
  3      V_EMPNO EMP.EMPNO%TYPE,
  4      V_ENAME EMP.ENAME%TYPE,
  5      V_JOB   EMP.JOB%TYPE,
  6      V_SAL   EMP.SAL%TYPE,
  7      V_DEPTNO EMP.DEPTNO%TYPE
  8  ) IS
  9     BEGIN
 10      INSERT INTO EMP(EMPNO,ENAME,JOB,SAL,DEPTNO)
 11      VALUES(V_EMPNO,V_ENAME,V_JOB,V_SAL,V_DEPTNO);
 12     COMMIT;
 13     &D('RECORD CREATED WITH EMPNO '||V_EMPNO);
 14     EXCEPTION
 15     WHEN DUP_VAL_ON_INDEX  THEN
 16     ------------------------NESTED  BLOCK-----------------
 17     DECLARE
 18     ID NUMBER :=0;
 19     BEGIN
 20  ----   SELECT MAX(EMPNO)+1 INTO ID FROM SCOTT.EMP;
 21  -----   ID := GET_ID('EMP') ;-------------CALLING FUNCTION
 22      NEW_ID(ID);   --------PROCEDURE
 23     INSERT INTO EMP(EMPNO,ENAME,JOB,SAL,DEPTNO)
 24     VALUES(ID,V_ENAME,V_JOB,V_SAL,V_DEPTNO);
 25     COMMIT;
 26     &D('RECORD CREATED WITH EMPNO '||ID ||' INSTEAD OF '||V_EMPNO);
 27     END;
 28     ----------------------------END OF NESTED BLOCK-------------
 29     WHEN OTHERS THEN
 30     &D(SQLCODE||'   '||SQLERRM);
 31*   END;
 32  /

Procedure created.

SQL>
```

```
SQL>
SQL>
SQL>  EXEC ADD_EMP(7788,USER,'SALESMAN',1000,30);
RECORD CREATED WITH EMPNO 7953 INSTEAD OF 7788


PL/SQL procedure successfully completed.

SQL>
SQL>
SQL>
SQL>
SQL> ED
Wrote file afiedt.buf

  1  CREATE OR REPLACE PROCEDURE ADD_EMP
  2  (
  3      V_EMPNO EMP.EMPNO%TYPE,
  4      V_ENAME EMP.ENAME%TYPE,
  5      V_JOB   EMP.JOB%TYPE,
  6      V_SAL   EMP.SAL%TYPE,
  7      V_DEPTNO EMP.DEPTNO%TYPE
  8  ) IS
  9  ----------------------------PVT PROCEDURE--------------------
 10  PROCEDURE NEW_ID(ID OUT NUMBER) IS
 11  BEGIN
 12  SELECT MAX(EMPNO)+1 INTO ID FROM SCOTT.EMP;
 13  END;
 14  ----------------------------END OF PVT PROCEDURE-------------------
 15     BEGIN
 16      INSERT INTO EMP(EMPNO,ENAME,JOB,SAL,DEPTNO)
 17      VALUES(V_EMPNO,V_ENAME,V_JOB,V_SAL,V_DEPTNO);
 18     COMMIT;
 19     &D('RECORD CREATED WITH EMPNO '||V_EMPNO);
 20     EXCEPTION
 21     WHEN DUP_VAL_ON_INDEX  THEN
 22     ------------------------NESTED  BLOCK-----------------
 23     DECLARE
 24     ID NUMBER :=0;
 25     BEGIN
 26  ----   SELECT MAX(EMPNO)+1 INTO ID FROM SCOTT.EMP;
 27  -----   ID := GET_ID('EMP') ;-------------CALLING PUBLIC FUNCTION
 28  ----    NEW_ID(ID);   --------CALLING PUBLIC PROCEDURE
 29      NEW_ID(ID);   --------CALLING PVT PROCEDURE
 30     INSERT INTO EMP(EMPNO,ENAME,JOB,SAL,DEPTNO)
 31     VALUES(ID,V_ENAME,V_JOB,V_SAL,V_DEPTNO);
 32     COMMIT;
 33     &D('RECORD CREATED WITH EMPNO '||ID ||' INSTEAD OF '||V_EMPNO);
 34     END;
 35     ----------------------------END OF NESTED BLOCK-------------
 36     WHEN OTHERS THEN
 37     &D(SQLCODE||'   '||SQLERRM);
 38*   END;
 39  /

Procedure created.

SQL>
SQL>
SQL>
SQL>
SQL>
SQL>  EXEC ADD_EMP(7788,USER,'SALESMAN',1000,30);
```

```
RECORD CREATED WITH EMPNO 7954 INSTEAD OF 7788


PL/SQL procedure successfully completed.

SQL>
SQL>
SQL>
SQL> DROP PROCEDURE NEW_ID;

Procedure dropped.

SQL>  EXEC ADD_EMP(7788,USER,'SALESMAN',1000,30);
RECORD CREATED WITH EMPNO 7955 INSTEAD OF 7788


PL/SQL procedure successfully completed.

SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL> SELECT * FROM EMP
  2  WHERE EMPNO=7955;

     EMPNO ENAME      JOB               MGR HIREDATE          SAL       COMM
DEPTNO
---------- ---------- --------- ---------- --------- ---------- ----------
----------
      7955 SCOTT      SALESMAN                               1000
30

SQL>
SQL>
SQL>
SQL>
SQL>
SQL> SELECT 'A' FROM DUAL;

'

-

A


SQL>
SQL>
SQL>
SQL>
SQL>
SQL> ED
Wrote file afiedt.buf

  1* SELECT '&ENTER_VAL' FROM DUAL
SQL> /
Enter value for enter_val: A

'
```

```
-

A


SQL>
SQL>
SQL>
SQL> ED
Wrote file afiedt.buf

  1* SELECT ASCII('&ENTER_VAL') FROM DUAL
SQL> /
Enter value for enter_val: A

ASCII('A')

----------

        65


SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL> /
Enter value for enter_val: A

ASCII('A')

----------

        65


SQL>
SQL> /
Enter value for enter_val: a

ASCII('A')

----------

        97


SQL>
SQL>
SQL>
SQL>
SQL> SELECT 97-65 FROM DUAL;

     97-65

----------

        32
```

```
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>  SELECT ASCII('&ENTER_VAL') FROM DUAL
  2
SQL> ED
Wrote file afiedt.buf

  1*  SELECT ASCII('&ENTER_VAL')+32 FROM DUAL
SQL> /
Enter value for enter_val: A

ASCII('A')+32

-------------

           97


SQL>
SQL>
SQL>
SQL> ED
Wrote file afiedt.buf

  1* SELECT CHR(ASCII('&ENTER_VAL')+32) FROM DUAL
  2  /
Enter value for enter_val: A

C

-

a


SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL> /
Enter value for enter_val: G

C

-

g


SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
```

```
SQL>
SQL>
SQL>
SQL>
SQL> /
Enter value for enter_val: T

C

-

t


SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL> ED
Wrote file afiedt.buf

  1* SELECT CHR(ASCII('&ENTER_VAL')-32) FROM DUAL
SQL> /
Enter value for enter_val: a

C

-

A


SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL> SPOOL OFF
```

```
SQL>
SQL>
SQL> SELECT TEXT FROM USER_SOURCE
  2  WHERE NAME='ADD_EMP';

TEXT

--------------------------------------------------------------------------------
----------------
PROCEDURE ADD_EMP

(

    V_EMPNO EMP.EMPNO%TYPE,

    V_ENAME EMP.ENAME%TYPE,

    V_JOB   EMP.JOB%TYPE,

    V_SAL   EMP.SAL%TYPE,

    V_DEPTNO EMP.DEPTNO%TYPE

) IS

----------------------------PVT PROCEDURE--------------------

PROCEDURE NEW_ID(ID OUT NUMBER) IS

BEGIN

SELECT MAX(EMPNO)+1 INTO ID FROM SCOTT.EMP;

END;

----------------------------END OF PVT PROCEDURE-------------------

   BEGIN

    INSERT INTO EMP(EMPNO,ENAME,JOB,SAL,DEPTNO)

    VALUES(V_EMPNO,V_ENAME,V_JOB,V_SAL,V_DEPTNO);

   COMMIT;

   DBMS_OUTPUT.PUT_LINE('RECORD CREATED WITH EMPNO '||V_EMPNO);

   EXCEPTION

   WHEN DUP_VAL_ON_INDEX  THEN

   ------------------------NESTED  BLOCK-----------------

   DECLARE

   ID NUMBER :=0;

   BEGIN

----   SELECT MAX(EMPNO)+1 INTO ID FROM SCOTT.EMP;

-----   ID := GET_ID('EMP') ;-------------CALLING PUBLIC FUNCTION
```

```
----    NEW_ID(ID);   --------CALLING PUBLIC PROCEDURE

    NEW_ID(ID);   --------CALLING PVT PROCEDURE

   INSERT INTO EMP(EMPNO,ENAME,JOB,SAL,DEPTNO)

   VALUES(ID,V_ENAME,V_JOB,V_SAL,V_DEPTNO);

   COMMIT;

   DBMS_OUTPUT.PUT_LINE('RECORD CREATED WITH EMPNO '||ID ||' INSTEAD OF '||V_EMPNO);

   END;

   ----------------------------END OF NESTED BLOCK-------------

   WHEN OTHERS THEN

   DBMS_OUTPUT.PUT_LINE(SQLCODE||'   '||SQLERRM);

  END;


38 rows selected.

SQL> DESC ADD_EMP
PROCEDURE ADD_EMP
 Argument Name                  Type                    In/Out Default?
 ------------------------------ ----------------------- ------ --------
 V_EMPNO                        NUMBER(4)               IN
 V_ENAME                        VARCHAR2(10)            IN
 V_JOB                          VARCHAR2(9)             IN
 V_SAL                          NUMBER(7,2)             IN
 V_DEPTNO                       NUMBER(2)               IN

SQL>
SQL>
SQL>
SQL>
SQL>
SQL> EXEC ADD_EMP(7788,USER,'SALESMAN',1000,30);
RECORD CREATED WITH EMPNO 7935 INSTEAD OF 7788


PL/SQL procedure successfully completed.

SQL>
SQL>
SQL>
SQL> EXEC ADD_EMP(7788,USER,'SALESMAN',1,30);
RECORD CREATED WITH EMPNO 7936 INSTEAD OF 7788


PL/SQL procedure successfully completed.

SQL>
SQL>
SQL>
SQL> ED
Wrote file afiedt.buf

  1  CREATE OR REPLACE FUNCTION VALID_SAL(V_SAL NUMBER) RETURN BOOLEAN
  2  IS
```

```
  3  V_GRADE NUMBER :=0;
  4  BEGIN
  5  SELECT GRADE INTO V_GRADE FROM SCOTT.SALGRADE
  6  WHERE V_SAL BETWEEN LOSAL AND HISAL;
  7  RETURN(TRUE);
  8  EXCEPTION
  9  WHEN NO_DATA_FOUND THEN
 10  RETURN(FALSE);
 11* END;
 12  /

Function created.

SQL>
SQL>
SQL> SELECT VALID_SAL(100) FROM DUAL;
SELECT VALID_SAL(100) FROM DUAL
       *
ERROR at line 1:
ORA-06552: PL/SQL: Statement ignored
ORA-06553: PLS-382: expression is of wrong type


SQL>
SQL>
SQL>
SQL>
SQL>
SQL> PROCEDURE ADD_EMP
  2  (
  3      V_EMPNO EMP.EMPNO%TYPE,
  4      V_ENAME EMP.ENAME%TYPE,
  5      V_JOB   EMP.JOB%TYPE,
  6      V_SAL   EMP.SAL%TYPE,
  7      V_DEPTNO EMP.DEPTNO%TYPE
  8  ) IS
  9  ------------------------------PVT PROCEDURE--------------------
 10  PROCEDURE NEW_ID(ID OUT NUMBER) IS
 11  BEGIN
 12  SELECT MAX(EMPNO)+1 INTO ID FROM SCOTT.EMP;
 13  END;
 14  ------------------------------END OF PVT PROCEDURE-------------------
 15     BEGIN
 16      INSERT INTO EMP(EMPNO,ENAME,JOB,SAL,DEPTNO)
 17      VALUES(V_EMPNO,V_ENAME,V_JOB,V_SAL,V_DEPTNO);
 18     COMMIT;
 19     DBMS_OUTPUT.PUT_LINE('RECORD CREATED WITH EMPNO '||V_EMPNO);
 20     EXCEPTION
 21     WHEN DUP_VAL_ON_INDEX  THEN
 22     ------------------------NESTED  BLOCK-----------------
 23     DECLARE
 24     ID NUMBER :=0;
 25     BEGIN
 26  ----   SELECT MAX(EMPNO)+1 INTO ID FROM SCOTT.EMP;
 27  -----    ID := GET_ID('EMP') ;-------------CALLING PUBLIC FUNCTION
 28  ----     NEW_ID(ID);   --------CALLING PUBLIC PROCEDURE
 29       NEW_ID(ID);   --------CALLING PVT PROCEDURE
 30     INSERT INTO EMP(EMPNO,ENAME,JOB,SAL,DEPTNO)
 31     VALUES(ID,V_ENAME,V_JOB,V_SAL,V_DEPTNO);
 32     COMMIT;
 33     DBMS_OUTPUT.PUT_LINE('RECORD CREATED WITH EMPNO '||ID ||' INSTEAD OF
'||V_EMPNO);
 34     END;
```

```
 35     ----------------------------END OF NESTED BLOCK-------------
 36     WHEN OTHERS THEN
 37     DBMS_OUTPUT.PUT_LINE(SQLCODE||'   '||SQLERRM);
 38    END;
 39
 40  .
SQL> ED
Wrote file afiedt.buf

  1  CREATE OR REPLACE PROCEDURE ADD_EMP
  2  (
  3      V_EMPNO EMP.EMPNO%TYPE,
  4      V_ENAME EMP.ENAME%TYPE,
  5      V_JOB   EMP.JOB%TYPE,
  6      V_SAL   EMP.SAL%TYPE,
  7      V_DEPTNO EMP.DEPTNO%TYPE
  8  ) IS
  9  ----------------------------PVT PROCEDURE--------------------
 10  PROCEDURE NEW_ID(ID OUT NUMBER) IS
 11  BEGIN
 12  SELECT MAX(EMPNO)+1 INTO ID FROM SCOTT.EMP;
 13  END;
 14  ----------------------------END OF PVT PROCEDURE-------------------
 15  VALID_SALARY BOOLEAN;
 16     BEGIN
 17  VALID_SALARY := VALID_SAL(V_SAL);  ----CALLING FUNCTION
 18  IF VALID_SALARY=TRUE THEN
 19     INSERT INTO EMP(EMPNO,ENAME,JOB,SAL,DEPTNO)
 20      VALUES(V_EMPNO,V_ENAME,V_JOB,V_SAL,V_DEPTNO);
 21     COMMIT;
 22     DBMS_OUTPUT.PUT_LINE('RECORD CREATED WITH EMPNO '||V_EMPNO);
 23  ELSE
 24  &D('INVALID SALARY.....');
 25  END IF;
 26     EXCEPTION
 27     WHEN DUP_VAL_ON_INDEX  THEN
 28     ------------------------NESTED  BLOCK-----------------
 29     DECLARE
 30     ID NUMBER :=0;
 31     BEGIN
 32  ----   SELECT MAX(EMPNO)+1 INTO ID FROM SCOTT.EMP;
 33  -----   ID := GET_ID('EMP') ;-------------CALLING PUBLIC FUNCTION
 34  ----    NEW_ID(ID);   --------CALLING PUBLIC PROCEDURE
 35      NEW_ID(ID);   --------CALLING PVT PROCEDURE
 36  IF VALID_SALARY=TRUE THEN
 37  INSERT INTO EMP(EMPNO,ENAME,JOB,SAL,DEPTNO)
 38     VALUES(ID,V_ENAME,V_JOB,V_SAL,V_DEPTNO);
 39     COMMIT;
 40     DBMS_OUTPUT.PUT_LINE('RECORD CREATED WITH EMPNO '||ID ||' INSTEAD OF
'||V_EMPNO);
 41  ELSE
 42  &D('INVALID SALARY..(EXCEPTION)...');
 43  END IF;
 44     END;
 45     ----------------------------END OF NESTED BLOCK-------------
 46     WHEN OTHERS THEN
 47     DBMS_OUTPUT.PUT_LINE(SQLCODE||'   '||SQLERRM);
 48*   END;
 49  /

Warning: Procedure created with compilation errors.

SQL> SHOW ERR
```

```
Errors for PROCEDURE ADD_EMP:

LINE/COL ERROR
-------- -----------------------------------------------------------------
15/1     PLS-00103: Encountered the symbol "VALID_SALARY" when expecting
         one of the following:
         begin function package pragma procedure form


48/6     PLS-00103: Encountered the symbol "end-of-file" when expecting
         one of the following:
         end not pragma final instantiable order overriding static
         member constructor map


SQL> ED
Wrote file afiedt.buf

  1  CREATE OR REPLACE PROCEDURE ADD_EMP
  2  (
  3      V_EMPNO EMP.EMPNO%TYPE,
  4      V_ENAME EMP.ENAME%TYPE,
  5      V_JOB   EMP.JOB%TYPE,
  6      V_SAL   EMP.SAL%TYPE,
  7      V_DEPTNO EMP.DEPTNO%TYPE
  8  ) IS
  9  VALID_SALARY BOOLEAN;
 10  ----------------------------PVT PROCEDURE--------------------
 11  PROCEDURE NEW_ID(ID OUT NUMBER) IS
 12  BEGIN
 13  SELECT MAX(EMPNO)+1 INTO ID FROM SCOTT.EMP;
 14  END;
 15  ----------------------------END OF PVT PROCEDURE-------------------
 16     BEGIN
 17  VALID_SALARY := VALID_SAL(V_SAL);  ----CALLING FUNCTION
 18  IF VALID_SALARY=TRUE THEN
 19     INSERT INTO EMP(EMPNO,ENAME,JOB,SAL,DEPTNO)
 20      VALUES(V_EMPNO,V_ENAME,V_JOB,V_SAL,V_DEPTNO);
 21     COMMIT;
 22     DBMS_OUTPUT.PUT_LINE('RECORD CREATED WITH EMPNO '||V_EMPNO);
 23  ELSE
 24  &D('INVALID SALARY.....');
 25  END IF;
 26     EXCEPTION
 27     WHEN DUP_VAL_ON_INDEX  THEN
 28     ------------------------NESTED  BLOCK-----------------
 29     DECLARE
 30     ID NUMBER :=0;
 31     BEGIN
 32  ----   SELECT MAX(EMPNO)+1 INTO ID FROM SCOTT.EMP;
 33  -----   ID := GET_ID('EMP') ;-------------CALLING PUBLIC FUNCTION
 34  ----    NEW_ID(ID);   --------CALLING PUBLIC PROCEDURE
 35      NEW_ID(ID);   --------CALLING PVT PROCEDURE
 36  IF VALID_SALARY=TRUE THEN
```

```
 37  INSERT INTO EMP(EMPNO,ENAME,JOB,SAL,DEPTNO)
 38     VALUES(ID,V_ENAME,V_JOB,V_SAL,V_DEPTNO);
 39     COMMIT;
 40     DBMS_OUTPUT.PUT_LINE('RECORD CREATED WITH EMPNO '||ID ||' INSTEAD OF
'||V_EMPNO);
 41  ELSE
 42  &D('INVALID SALARY..(EXCEPTION)...');
 43  END IF;
 44     END;
 45     ------------------------------END OF NESTED BLOCK-------------
 46     WHEN OTHERS THEN
 47     DBMS_OUTPUT.PUT_LINE(SQLCODE||'   '||SQLERRM);
 48*   END;
 49  /

Procedure created.

SQL> EXEC ADD_EMP(7788,USER,'SALESMAN',1,30);
INVALID SALARY.....


PL/SQL procedure successfully completed.

SQL> ED
Wrote file afiedt.buf

  1  CREATE OR REPLACE PROCEDURE ADD_EMP
  2  (
  3      V_EMPNO EMP.EMPNO%TYPE,
  4      V_ENAME EMP.ENAME%TYPE,
  5      V_JOB   EMP.JOB%TYPE,
  6      V_SAL   EMP.SAL%TYPE,
  7      V_DEPTNO EMP.DEPTNO%TYPE
  8  ) IS
  9  VALID_SALARY BOOLEAN;
 10  ------------------------------PVT PROCEDURE--------------------
 11  PROCEDURE NEW_ID(ID OUT NUMBER) IS
 12  BEGIN
 13  SELECT MAX(EMPNO)+1 INTO ID FROM SCOTT.EMP;
 14  END;
 15  ------------------------------END OF PVT PROCEDURE-------------------
 16     BEGIN
 17  VALID_SALARY := VALID_SAL(V_SAL);  ----CALLING FUNCTION
 18  IF VALID_SALARY=FALSE THEN
 19  RAISE_APPLICATION_ERROR(-20100,'INVALID SALARY....' );
 20  END IF;
 21     INSERT INTO EMP(EMPNO,ENAME,JOB,SAL,DEPTNO)
 22      VALUES(V_EMPNO,V_ENAME,V_JOB,V_SAL,V_DEPTNO);
 23     COMMIT;
 24     DBMS_OUTPUT.PUT_LINE('RECORD CREATED WITH EMPNO '||V_EMPNO);
 25     EXCEPTION
 26     WHEN DUP_VAL_ON_INDEX  THEN
 27     ------------------------NESTED  BLOCK-----------------
 28     DECLARE
 29     ID NUMBER :=0;
 30     BEGIN
 31  ----   SELECT MAX(EMPNO)+1 INTO ID FROM SCOTT.EMP;
 32  -----    ID := GET_ID('EMP') ;-------------CALLING PUBLIC FUNCTION
 33  ----     NEW_ID(ID);   --------CALLING PUBLIC PROCEDURE
 34       NEW_ID(ID);   --------CALLING PVT PROCEDURE
 35  IF VALID_SALARY=TRUE THEN
 36  INSERT INTO EMP(EMPNO,ENAME,JOB,SAL,DEPTNO)
 37     VALUES(ID,V_ENAME,V_JOB,V_SAL,V_DEPTNO);
```

```
 38     COMMIT;
 39     DBMS_OUTPUT.PUT_LINE('RECORD CREATED WITH EMPNO '||ID ||' INSTEAD OF
'||V_EMPNO);
 40   ELSE
 41   &D('INVALID SALARY..(EXCEPTION)...');
 42   END IF;
 43      END;
 44      ------------------------------END OF NESTED BLOCK-------------
 45      WHEN OTHERS THEN
 46      DBMS_OUTPUT.PUT_LINE(SQLCODE||'   '||SQLERRM);
 47*    END;
 48   /

Procedure created.

SQL>
SQL>
SQL> EXEC ADD_EMP(7788,USER,'SALESMAN',1,30);
-20100   ORA-20100: INVALID SALARY....


PL/SQL procedure successfully completed.

SQL>
SQL>
SQL>
SQL> ED
Wrote file afiedt.buf

  1  CREATE OR REPLACE PROCEDURE ADD_EMP
  2  (
  3      V_EMPNO EMP.EMPNO%TYPE,
  4      V_ENAME EMP.ENAME%TYPE,
  5      V_JOB   EMP.JOB%TYPE,
  6      V_SAL   EMP.SAL%TYPE,
  7      V_DEPTNO EMP.DEPTNO%TYPE
  8  ) IS
  9  VALID_SALARY BOOLEAN;
 10  ----------------------------PVT PROCEDURE--------------------
 11  PROCEDURE NEW_ID(ID OUT NUMBER) IS
 12  BEGIN
 13  SELECT MAX(EMPNO)+1 INTO ID FROM SCOTT.EMP;
 14  END;
 15  ------------------------------END OF PVT PROCEDURE-------------------
 16     BEGIN
 17  VALID_SALARY := VALID_SAL(V_SAL);  ----CALLING FUNCTION
 18  IF VALID_SALARY=FALSE THEN
 19  RAISE_APPLICATION_ERROR(-20100,'INVALID SALARY....' );
 20  END IF;
 21     INSERT INTO EMP(EMPNO,ENAME,JOB,SAL,DEPTNO)
 22      VALUES(V_EMPNO,V_ENAME,V_JOB,V_SAL,V_DEPTNO);
 23     COMMIT;
 24     DBMS_OUTPUT.PUT_LINE('RECORD CREATED WITH EMPNO '||V_EMPNO);
 25     EXCEPTION
 26     WHEN DUP_VAL_ON_INDEX  THEN
 27     ------------------------NESTED  BLOCK-----------------
 28     DECLARE
 29     ID NUMBER :=0;
 30     BEGIN
 31  ----   SELECT MAX(EMPNO)+1 INTO ID FROM SCOTT.EMP;
 32  -----    ID := GET_ID('EMP') ;-------------CALLING PUBLIC FUNCTION
 33  ----     NEW_ID(ID);   --------CALLING PUBLIC PROCEDURE
 34      NEW_ID(ID);   --------CALLING PVT PROCEDURE
```

```
 35  IF VALID_SALARY=TRUE THEN
 36  INSERT INTO EMP(EMPNO,ENAME,JOB,SAL,DEPTNO)
 37     VALUES(ID,V_ENAME,V_JOB,V_SAL,V_DEPTNO);
 38     COMMIT;
 39     DBMS_OUTPUT.PUT_LINE('RECORD CREATED WITH EMPNO '||ID ||' INSTEAD OF
'||V_EMPNO);
 40  ELSE
 41  &D('INVALID SALARY..(EXCEPTION)...');
 42  END IF;
 43     END;
 44     ------------------------------END OF NESTED BLOCK-------------
 45     ----WHEN OTHERS THEN
 46     ----DBMS_OUTPUT.PUT_LINE(SQLCODE||'   '||SQLERRM);
 47*   END;
SQL> /

Procedure created.

SQL>  EXEC ADD_EMP(7788,USER,'SALESMAN',1,30);
BEGIN ADD_EMP(7788,USER,'SALESMAN',1,30); END;

*
ERROR at line 1:
ORA-20100: INVALID SALARY....
ORA-06512: at "SCOTT.ADD_EMP", line 19
ORA-06512: at line 1


SQL>
SQL>
SQL>
SQL> EXEC ADD_EMP(7788,USER,'SALESMAN',1000,30);
RECORD CREATED WITH EMPNO 7937 INSTEAD OF 7788


PL/SQL procedure successfully completed.

SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL> ED
Wrote file afiedt.buf

  1  CREATE OR REPLACE PROCEDURE ADD_EMP
  2  (
  3      V_EMPNO EMP.EMPNO%TYPE,
  4      V_ENAME EMP.ENAME%TYPE,
  5      V_JOB   EMP.JOB%TYPE,
  6      V_SAL   EMP.SAL%TYPE,
  7      V_DEPTNO EMP.DEPTNO%TYPE,
  8      V_MGR    EMP.MGR%TYPE
  9  ) IS
 10  VALID_SALARY BOOLEAN;
 11  ------------------------------PVT PROCEDURE--------------------
 12  PROCEDURE NEW_ID(ID OUT NUMBER) IS
```

```
 13  BEGIN
 14  SELECT MAX(EMPNO)+1 INTO ID FROM SCOTT.EMP;
 15  END;
 16  ----------------------------END OF PVT PROCEDURE------------------
 17     BEGIN
 18  VALID_SALARY := VALID_SAL(V_SAL);  ----CALLING FUNCTION
 19  IF VALID_SALARY=FALSE THEN
 20  RAISE_APPLICATION_ERROR(-20100,'INVALID SALARY....' );
 21  END IF;
 22     INSERT INTO EMP(EMPNO,ENAME,JOB,SAL,DEPTNO,MGR)
 23      VALUES(V_EMPNO,V_ENAME,V_JOB,V_SAL,V_DEPTNO,V_MGR);
 24     COMMIT;
 25     DBMS_OUTPUT.PUT_LINE('RECORD CREATED WITH EMPNO '||V_EMPNO);
 26     EXCEPTION
 27     WHEN DUP_VAL_ON_INDEX  THEN
 28     ----------------------NESTED  BLOCK----------------
 29     DECLARE
 30     ID NUMBER :=0;
 31     BEGIN
 32  ----   SELECT MAX(EMPNO)+1 INTO ID FROM SCOTT.EMP;
 33  -----    ID := GET_ID('EMP') ;-------------CALLING PUBLIC FUNCTION
 34  ----    NEW_ID(ID);   --------CALLING PUBLIC PROCEDURE
 35      NEW_ID(ID);   --------CALLING PVT PROCEDURE
 36  IF VALID_SALARY=TRUE THEN
 37  INSERT INTO EMP(EMPNO,ENAME,JOB,SAL,DEPTNO,MGR)
 38     VALUES(ID,V_ENAME,V_JOB,V_SAL,V_DEPTNO,V_MGR);
 39     COMMIT;
 40     DBMS_OUTPUT.PUT_LINE('RECORD CREATED WITH EMPNO '||ID ||' INSTEAD OF
'||V_EMPNO);
 41  ELSE
 42  &D('INVALID SALARY..(EXCEPTION)...');
 43  END IF;
 44     END;
 45     ----------------------------END OF NESTED BLOCK-------------
 46     ----WHEN OTHERS THEN
 47     ----DBMS_OUTPUT.PUT_LINE(SQLCODE||'   '||SQLERRM);
 48*   END;
SQL> /

Procedure created.

SQL> DESC ADD_EMP;
PROCEDURE ADD_EMP
 Argument Name                  Type                    In/Out Default?
 ------------------------------ ----------------------- ------ --------
 V_EMPNO                        NUMBER(4)               IN
 V_ENAME                        VARCHAR2(10)            IN
 V_JOB                          VARCHAR2(9)             IN
 V_SAL                          NUMBER(7,2)             IN
 V_DEPTNO                       NUMBER(2)               IN
 V_MGR                          NUMBER(5)               IN

SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL> EXEC ADD_EMP(7788,USER,'SALESMAN',1000,30,01);
RECORD CREATED WITH EMPNO 7938 INSTEAD OF 7788


PL/SQL procedure successfully completed.
```

```
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL> SELECT * FROM EMP
  2  WHERE EMPNO=01;

no rows selected

SQL>
SQL>
SQL>
SQL>
SQL> ED
Wrote file afiedt.buf

  1  CREATE OR REPLACE GET_MGR(V_MGR NUMBER) RETURN BOOLEAN
  2  IS
  3  ID NUMBER :=0;
  4  BEGIN
  5  SELECT EMPNO INTO ID FROM SCOTT.EMP
  6  WHERE EMPNO=V_MGR;
  7  RETURN(TRUE);
  8  EXCEPTION
  9  WHEN NO_DATA_FOUND THEN
 10  RETURN(FALSE);
 11* END;
 12  /
CREATE OR REPLACE GET_MGR(V_MGR NUMBER) RETURN BOOLEAN
                  *
ERROR at line 1:
ORA-00922: missing or invalid option


SQL>
SQL>
SQL>
SQL> ED
Wrote file afiedt.buf

  1  CREATE OR REPLACE FUNCTION GET_MGR(V_MGR NUMBER) RETURN BOOLEAN
  2  IS
  3  ID NUMBER :=0;
  4  BEGIN
  5  SELECT EMPNO INTO ID FROM SCOTT.EMP
  6  WHERE EMPNO=V_MGR;
  7  RETURN(TRUE);
  8  EXCEPTION
  9  WHEN NO_DATA_FOUND THEN
 10  RETURN(FALSE);
 11* END;
SQL> /

Function created.

SQL>
SQL>
SQL>
SQL>
SQL>  CREATE OR REPLACE PROCEDURE ADD_EMP
```

```
  2   (
  3       V_EMPNO EMP.EMPNO%TYPE,
  4       V_ENAME EMP.ENAME%TYPE,
  5       V_JOB   EMP.JOB%TYPE,
  6       V_SAL   EMP.SAL%TYPE,
  7       V_DEPTNO EMP.DEPTNO%TYPE,
  8       V_MGR    EMP.MGR%TYPE
  9   ) IS
 10   VALID_SALARY BOOLEAN;
 11   ----------------------------PVT PROCEDURE--------------------
 12   PROCEDURE NEW_ID(ID OUT NUMBER) IS
 13   BEGIN
 14   SELECT MAX(EMPNO)+1 INTO ID FROM SCOTT.EMP;
 15   END;
 16   ----------------------------END OF PVT PROCEDURE-------------------
 17      BEGIN
 18   VALID_SALARY := VALID_SAL(V_SAL);  ----CALLING FUNCTION
 19   IF VALID_SALARY=FALSE THEN
 20   RAISE_APPLICATION_ERROR(-20100,'INVALID SALARY....' );
 21   END IF;
 22      INSERT INTO EMP(EMPNO,ENAME,JOB,SAL,DEPTNO,MGR)
 23       VALUES(V_EMPNO,V_ENAME,V_JOB,V_SAL,V_DEPTNO,V_MGR);
 24      COMMIT;
 25      DBMS_OUTPUT.PUT_LINE('RECORD CREATED WITH EMPNO '||V_EMPNO);
 26      EXCEPTION
 27      WHEN DUP_VAL_ON_INDEX  THEN
 28      ------------------------NESTED  BLOCK-----------------
 29      DECLARE
 30      ID NUMBER :=0;
 31      BEGIN
 32   ----   SELECT MAX(EMPNO)+1 INTO ID FROM SCOTT.EMP;
 33   -----   ID := GET_ID('EMP') ;-------------CALLING PUBLIC FUNCTION
 34   ----    NEW_ID(ID);   --------CALLING PUBLIC PROCEDURE
 35       NEW_ID(ID);   --------CALLING PVT PROCEDURE
 36   IF VALID_SALARY=TRUE THEN
 37   INSERT INTO EMP(EMPNO,ENAME,JOB,SAL,DEPTNO,MGR)
 38      VALUES(ID,V_ENAME,V_JOB,V_SAL,V_DEPTNO,V_MGR);
 39      COMMIT;
 40      DBMS_OUTPUT.PUT_LINE('RECORD CREATED WITH EMPNO '||ID ||' INSTEAD OF
'||V_EMPNO);
 41   ELSE
 42   &D('INVALID SALARY..(EXCEPTION)...');
 43   END IF;
 44      END;
 45      ----------------------------END OF NESTED BLOCK-------------
 46      ----WHEN OTHERS THEN
 47      ----DBMS_OUTPUT.PUT_LINE(SQLCODE||'   '||SQLERRM);
 48     END;
 49  .
SQL> ED
Wrote file afiedt.buf

  1   CREATE OR REPLACE PROCEDURE ADD_EMP
  2   (
  3       V_EMPNO EMP.EMPNO%TYPE,
  4       V_ENAME EMP.ENAME%TYPE,
  5       V_JOB   EMP.JOB%TYPE,
  6       V_SAL   EMP.SAL%TYPE,
  7       V_DEPTNO EMP.DEPTNO%TYPE,
  8       V_MGR    EMP.MGR%TYPE
  9   ) IS
 10   VALID_SALARY BOOLEAN;
 11   VALID_MGR BOOLEAN;
```

```
 12   ----------------------------PVT PROCEDURE--------------------
 13   PROCEDURE NEW_ID(ID OUT NUMBER) IS
 14   BEGIN
 15   SELECT MAX(EMPNO)+1 INTO ID FROM SCOTT.EMP;
 16   END;
 17   ----------------------------END OF PVT PROCEDURE------------------
 18      BEGIN
 19   VALID_SALARY := VALID_SAL(V_SAL);  ----CALLING FUNCTION
 20   VALID_MGR  := GET_MGR(V_MGR);   -----CALLING FUNCTION
 21   IF VALID_SALARY=FALSE AND VALID_MGR=FALSE THEN
 22   RAISE_APPLICATION_ERROR(-20100,'INVALID SALARY OR INVALID MANAGER....' );
 23   END IF;
 24      INSERT INTO EMP(EMPNO,ENAME,JOB,SAL,DEPTNO,MGR)
 25       VALUES(V_EMPNO,V_ENAME,V_JOB,V_SAL,V_DEPTNO,V_MGR);
 26      COMMIT;
 27      DBMS_OUTPUT.PUT_LINE('RECORD CREATED WITH EMPNO '||V_EMPNO);
 28      EXCEPTION
 29      WHEN DUP_VAL_ON_INDEX  THEN
 30      ------------------------NESTED  BLOCK-----------------
 31      DECLARE
 32      ID NUMBER :=0;
 33      BEGIN
 34   ----   SELECT MAX(EMPNO)+1 INTO ID FROM SCOTT.EMP;
 35   -----   ID := GET_ID('EMP') ;-------------CALLING PUBLIC FUNCTION
 36   ----    NEW_ID(ID);   --------CALLING PUBLIC PROCEDURE
 37       NEW_ID(ID);   --------CALLING PVT PROCEDURE
 38   INSERT INTO EMP(EMPNO,ENAME,JOB,SAL,DEPTNO,MGR)
 39      VALUES(ID,V_ENAME,V_JOB,V_SAL,V_DEPTNO,V_MGR);
 40      COMMIT;
 41      DBMS_OUTPUT.PUT_LINE('RECORD CREATED WITH EMPNO '||ID ||' INSTEAD OF
'||V_EMPNO);
 42      END;
 43      ----------------------------END OF NESTED BLOCK-------------
 44      ----WHEN OTHERS THEN
 45      ----DBMS_OUTPUT.PUT_LINE(SQLCODE||'   '||SQLERRM);
 46*    END;
 47  /

Procedure created.

SQL>
SQL>
SQL>
SQL>
SQL> EXEC ADD_EMP(7788,USER,'SALESMAN',1000,30,01);
RECORD CREATED WITH EMPNO 7939 INSTEAD OF 7788


PL/SQL procedure successfully completed.

SQL>
SQL>
SQL>
SQL>
SQL>
SQL>    CREATE OR REPLACE PROCEDURE ADD_EMP
  2     (
  3         V_EMPNO EMP.EMPNO%TYPE,
  4         V_ENAME EMP.ENAME%TYPE,
  5         V_JOB   EMP.JOB%TYPE,
  6         V_SAL   EMP.SAL%TYPE,
  7         V_DEPTNO EMP.DEPTNO%TYPE,
  8         V_MGR    EMP.MGR%TYPE
```

```
  9     ) IS
 10     VALID_SALARY BOOLEAN;
 11     VALID_MGR BOOLEAN;
 12     ----------------------------PVT PROCEDURE-------------------
 13     PROCEDURE NEW_ID(ID OUT NUMBER) IS
 14     BEGIN
 15     SELECT MAX(EMPNO)+1 INTO ID FROM SCOTT.EMP;
 16     END;
 17     ----------------------------END OF PVT PROCEDURE------------------
 18        BEGIN
 19     VALID_SALARY := VALID_SAL(V_SAL);  ----CALLING FUNCTION
 20     VALID_MGR  := GET_MGR(V_MGR);   -----CALLING FUNCTION
 21     IF VALID_SALARY=FALSE AND VALID_MGR=FALSE THEN
 22     RAISE_APPLICATION_ERROR(-20100,'INVALID SALARY OR INVALID MANAGER....' );
 23     END IF;
 24        INSERT INTO EMP(EMPNO,ENAME,JOB,SAL,DEPTNO,MGR)
 25         VALUES(V_EMPNO,V_ENAME,V_JOB,V_SAL,V_DEPTNO,V_MGR);
 26        COMMIT;
 27        DBMS_OUTPUT.PUT_LINE('RECORD CREATED WITH EMPNO '||V_EMPNO);
 28        EXCEPTION
 29        WHEN DUP_VAL_ON_INDEX  THEN
 30        ----------------------NESTED  BLOCK----------------
 31        DECLARE
 32        ID NUMBER :=0;
 33        BEGIN
 34     ----   SELECT MAX(EMPNO)+1 INTO ID FROM SCOTT.EMP;
 35     -----   ID := GET_ID('EMP') ;-------------CALLING PUBLIC FUNCTION
 36     ----    NEW_ID(ID);   --------CALLING PUBLIC PROCEDURE
 37         NEW_ID(ID);   --------CALLING PVT PROCEDURE
 38     INSERT INTO EMP(EMPNO,ENAME,JOB,SAL,DEPTNO,MGR)
 39        VALUES(ID,V_ENAME,V_JOB,V_SAL,V_DEPTNO,V_MGR);
 40        COMMIT;
 41        DBMS_OUTPUT.PUT_LINE('RECORD CREATED WITH EMPNO '||ID ||' INSTEAD OF
'||V_EMPNO);
 42        END;
 43        ----------------------------END OF NESTED BLOCK-------------
 44        ----WHEN OTHERS THEN
 45        ----DBMS_OUTPUT.PUT_LINE(SQLCODE||'   '||SQLERRM);
 46  *    END;
 47
 48  .ED
 49  .
SQL> ED
Wrote file afiedt.buf

  1     CREATE OR REPLACE PROCEDURE ADD_EMP
  2     (
  3         V_EMPNO EMP.EMPNO%TYPE,
  4         V_ENAME EMP.ENAME%TYPE,
  5         V_JOB   EMP.JOB%TYPE,
  6         V_SAL   EMP.SAL%TYPE,
  7         V_DEPTNO EMP.DEPTNO%TYPE,
  8         V_MGR    EMP.MGR%TYPE
  9     ) IS
 10     VALID_SALARY BOOLEAN;
 11     VALID_MGR BOOLEAN;
 12     ----------------------------PVT PROCEDURE-------------------
 13     PROCEDURE NEW_ID(ID OUT NUMBER) IS
 14     BEGIN
 15     SELECT MAX(EMPNO)+1 INTO ID FROM SCOTT.EMP;
 16     END;
 17     ----------------------------END OF PVT PROCEDURE------------------
 18        BEGIN
```

```
 19     VALID_SALARY := VALID_SAL(V_SAL);  ----CALLING FUNCTION
 20     VALID_MGR  := GET_MGR(V_MGR);   -----CALLING FUNCTION
 21     IF VALID_SALARY=FALSE OR VALID_MGR=FALSE THEN
 22     RAISE_APPLICATION_ERROR(-20100,'INVALID SALARY OR INVALID MANAGER....' );
 23     END IF;
 24        INSERT INTO EMP(EMPNO,ENAME,JOB,SAL,DEPTNO,MGR)
 25         VALUES(V_EMPNO,V_ENAME,V_JOB,V_SAL,V_DEPTNO,V_MGR);
 26        COMMIT;
 27        DBMS_OUTPUT.PUT_LINE('RECORD CREATED WITH EMPNO '||V_EMPNO);
 28        EXCEPTION
 29        WHEN DUP_VAL_ON_INDEX  THEN
 30        ------------------------NESTED  BLOCK-----------------
 31        DECLARE
 32        ID NUMBER :=0;
 33        BEGIN
 34     ----   SELECT MAX(EMPNO)+1 INTO ID FROM SCOTT.EMP;
 35     -----   ID := GET_ID('EMP') ;-------------CALLING PUBLIC FUNCTION
 36     ----    NEW_ID(ID);   --------CALLING PUBLIC PROCEDURE
 37         NEW_ID(ID);   --------CALLING PVT PROCEDURE
 38     INSERT INTO EMP(EMPNO,ENAME,JOB,SAL,DEPTNO,MGR)
 39        VALUES(ID,V_ENAME,V_JOB,V_SAL,V_DEPTNO,V_MGR);
 40        COMMIT;
 41        DBMS_OUTPUT.PUT_LINE('RECORD CREATED WITH EMPNO '||ID ||' INSTEAD OF
'||V_EMPNO);
 42        END;
 43        ----------------------------END OF NESTED BLOCK-------------
 44        ----WHEN OTHERS THEN
 45        ----DBMS_OUTPUT.PUT_LINE(SQLCODE||'   '||SQLERRM);
 46*     END;
 47  /

Procedure created.

SQL> EXEC ADD_EMP(7788,USER,'SALESMAN',1000,30,01);
BEGIN ADD_EMP(7788,USER,'SALESMAN',1000,30,01); END;

*
ERROR at line 1:
ORA-20100: INVALID SALARY OR INVALID MANAGER....
ORA-06512: at "SCOTT.ADD_EMP", line 22
ORA-06512: at line 1


SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL> SELECT * FROM EMP;

     EMPNO ENAME      JOB              MGR HIREDATE          SAL        COMM
DEPTNO
---------- ---------- --------- ---------- --------- ---------- ----------
----------
      5454 SMITH      CLERK            7902 17-DEC-80        900
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       1000
30
      7902 FORD       ANALYST         7566 03-DEC-81      45666
20
      7934 MILLER     CLERK           7782 23-JAN-85       1300
10
      7935 SCOTT      SALESMAN                             1000
30
      7936 SCOTT      SALESMAN                                1
30
      7937 SCOTT      SALESMAN                             1000
30
      7938 SCOTT      SALESMAN           1                 1000
30
      7939 SCOTT      SALESMAN           1                 1000
30

19 rows selected.

SQL>
SQL>
SQL>
SQL> SELECT * FROM EMP
  2  WHERE MGR=&EMPNO;
Enter value for empno: 7939

no rows selected

SQL> /
Enter value for empno: 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

SQL>
SQL>
SQL>
SQL>
SQL>
SQL> ED
```

```
Wrote file afiedt.buf

SP2-0223: No lines in SQL buffer.
SQL> ..
SP2-0042: unknown command ".." - rest of line ignored.
SQL>
SQL>
SQL>
SQL> ED
SP2-0107: Nothing to save.
SQL> ED
SP2-0107: Nothing to save.
SQL> DECLARE
  2  .
SQL>
SQL> ED
Wrote file afiedt.buf

  1  CREATE OR REPLACE FUNCTION GET_TAX(V_SAL NUMBER)
  2  RETURN NUMBER IS
  3  V_TAX NUMBER :=0;
  4  ANN_SAL NUMBER :=0;
  5  BEGIN
  6  ANN_SAL := V_SAL * 12;
  7  IF ANN_SAL BETWEEN 15000 AND 20000 THEN
  8  V_TAX := V_SAL * 5/100;
  9  ELSIF ANN_SAL BETWEEN 20001 AND 30000 THEN
 10  V_TAX := V_SAL * 7/100;
 11  ELSIF ANN_SAL BETWEEN 30001 AND 40000 THEN
 12  V_TAX := V_SAL * 9/100;
 13  ELSIF ANN_SAL >40000 THEN
 14  V_TAX := V_SAL * 10/100;
 15  END IF;
 16  RETURN(V_TAX);
 17* END;
 18  /

Function created.

SQL>
SQL>
SQL>
SQL>
SQL> SELECT GET_TAX(2000) FROM DUAL;

GET_TAX(2000)

-------------

          140


SQL>
SQL>
SQL>
SQL>
SQL> SELECT EMPNO,ENAME,JOB,SAL,GET_TAX(SAL) FROM EMP;

     EMPNO ENAME      JOB               SAL GET_TAX(SAL)

---------- ---------- --------- ---------- ------------

      5454 SMITH      CLERK             900            0
```

```
      7499 ALLEN      SALESMAN        1600           80
      7521 WARD       SALESMAN        1250         62.5
      7566 JONES      MANAGER         2975       267.75
      7654 MARTIN     SALESMAN        1250         62.5
      7698 BLAKE      MANAGER         2850        256.5
      7782 CLARK      MANAGER         2450        171.5
      7788 SCOTT      ANALYST         3000          270
      7839 KING       PRESIDENT       5000          500
      7844 TURNER     SALESMAN        1500           75
      7876 ADAMS      CLERK           1100            0
      7900 JAMES      CLERK           1000            0
      7902 FORD       ANALYST        45666       4566.6
      7934 MILLER     CLERK           1300           65
      7935 SCOTT      SALESMAN        1000            0
      7936 SCOTT      SALESMAN           1            0
      7937 SCOTT      SALESMAN        1000            0
      7938 SCOTT      SALESMAN        1000            0
      7939 SCOTT      SALESMAN        1000            0

19 rows selected.

SQL>
SQL>
SQL> ED
Wrote file afiedt.buf

  1* SELECT EMPNO,ENAME,JOB,SAL,GET_TAX(SAL) FROM EMP
SQL> .
SQL> ED
Wrote file afiedt.buf

  1  CREATE OR REPLACE FUNCTION NEW_REC
  2  (
  3  V_EMPNO NUMBER,
  4  V_ENAME VARCHAR2,
  5  V_JOB   VARCHAR2,
  6  V_SAL    NUMBER ,
  7  V_DEPTNO  NUMBER ,
  8  V_MGR NUMBER
  9  )
 10  RETURN VARCHAR2
 11  IS
 12  BEGIN
 13  INSERT INTO EMP(EMPNO,ENAME,JOB,SAL,DEPTNO,MGR)
```

```
 14     VALUES(V_EMPNO,V_ENAME,V_JOB,V_SAL,V_DEPTNO,V_MGR);
 15  COMMIT;
 16  RETURN('RECORD CREATED WITH EMPNO '||V_EMPNO);
 17* END;
 18  /

Function created.

SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL> VAR RES VARCHAR2(200);
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL> EXECUTE :RES :=NEW_REC(8000,USER,'SALESMAN',1000,30,102);

PL/SQL procedure successfully completed.

SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL> PRINT

RES

--------------------------------------------------------------------------------
----------------
RECORD CREATED WITH EMPNO 8000


SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL> SELECT NEW_REC(8000,USER,'SALESMAN',1000,30,102) FROM DUAL;
SELECT NEW_REC(8000,USER,'SALESMAN',1000,30,102) FROM DUAL
       *
ERROR at line 1:
ORA-14551: cannot perform a DML operation inside a query
ORA-06512: at "SCOTT.NEW_REC", line 13


SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
```

```
SQL> ------------------PACKAGES--------------
SQL>
SQL>
SQL> SELECT OBJECT_NAME FROM USER_OBJECTS
  2  WHERE OBJECT_TYPE IN ('PROCEDURE','FUNCTION');

OBJECT_NAME

--------------------------------------------------------------------------------
----------------
GET_ORD

INS_REC

ADD_EMP

GET_MGR

GET_JOB

ADD_NEW_EMP

EMP_POSTING

DEL_REC

SHOW_TXT

WRITE_TO_FILE

GET_FILE_TXT

TEST_JOB

DO_EXE_IMM

T1

CREATE_TABLE

SHOW_REC

ADD_DEPT

GET_ID

ADD_R

SET_VDO

GET_EMP_VDO_LEN

LOAD_TXT_DATA

CHK_SAL

GET_WORDS

GET_TAX

TAB_NO

MY_CODE
```

```
FIRST_PRO

TEST2

GET_TABLE

VALID_SAL

NEW_REC


32 rows selected.

SQL> ED
Wrote file afiedt.buf

  1  SELECT OBJECT_NAME NAMES,OBJECT_TYPE FROM USER_OBJECTS
  2* WHERE OBJECT_TYPE IN ('PROCEDURE','FUNCTION')
SQL> /

NAMES

--------------------------------------------------------------------------------
----------------
OBJECT_TYPE

------------------

GET_ORD

FUNCTION


INS_REC

FUNCTION


ADD_EMP

PROCEDURE


GET_MGR

FUNCTION


GET_JOB

FUNCTION


ADD_NEW_EMP

PROCEDURE
```

EMP_POSTING

PROCEDURE

DEL_REC

PROCEDURE

SHOW_TXT

PROCEDURE

WRITE_TO_FILE

PROCEDURE

GET_FILE_TXT

PROCEDURE

TEST_JOB

PROCEDURE

DO_EXE_IMM

PROCEDURE

T1

PROCEDURE

CREATE_TABLE

PROCEDURE

SHOW_REC

PROCEDURE

ADD_DEPT

PROCEDURE

GET_ID

FUNCTION

ADD_R

PROCEDURE

SET_VDO

PROCEDURE

GET_EMP_VDO_LEN

PROCEDURE

LOAD_TXT_DATA

PROCEDURE

CHK_SAL

PROCEDURE

GET_WORDS

FUNCTION

GET_TAX

FUNCTION

TAB_NO

PROCEDURE

MY_CODE

PROCEDURE

```
FIRST_PRO

PROCEDURE


TEST2

PROCEDURE


GET_TABLE

FUNCTION


VALID_SAL

FUNCTION


NEW_REC

FUNCTION

NAMES

--------------------------------------------------------------------------------
----------------
OBJECT_TYPE

-------------------



32 rows selected.

SQL> COL NAMES FORMAT A20
SQL> /

NAMES                OBJECT_TYPE

-------------------- -------------------

GET_ORD              FUNCTION

INS_REC              FUNCTION

ADD_EMP              PROCEDURE

GET_MGR              FUNCTION

GET_JOB              FUNCTION

ADD_NEW_EMP          PROCEDURE
```

```
EMP_POSTING             PROCEDURE

DEL_REC                 PROCEDURE

SHOW_TXT                PROCEDURE

WRITE_TO_FILE           PROCEDURE

GET_FILE_TXT            PROCEDURE

TEST_JOB                PROCEDURE

DO_EXE_IMM              PROCEDURE

T1                      PROCEDURE

CREATE_TABLE            PROCEDURE

SHOW_REC                PROCEDURE

ADD_DEPT                PROCEDURE

GET_ID                  FUNCTION

ADD_R                   PROCEDURE

SET_VDO                 PROCEDURE

GET_EMP_VDO_LEN         PROCEDURE

LOAD_TXT_DATA           PROCEDURE

CHK_SAL                 PROCEDURE

GET_WORDS               FUNCTION

GET_TAX                 FUNCTION

TAB_NO                  PROCEDURE

MY_CODE                 PROCEDURE

FIRST_PRO               PROCEDURE

TEST2                   PROCEDURE

GET_TABLE               FUNCTION

VALID_SAL               FUNCTION

NEW_REC                 FUNCTION


32 rows selected.

SQL> ED
Wrote file afiedt.buf

  1  SELECT OBJECT_NAME NAMES,OBJECT_TYPE FROM USER_OBJECTS
  2* WHERE OBJECT_TYPE IN ('PROCEDURE')
SQL> /

NAMES                   OBJECT_TYPE
```

```
-------------------- --------------------

ADD_EMP              PROCEDURE

ADD_NEW_EMP          PROCEDURE

EMP_POSTING          PROCEDURE

DEL_REC              PROCEDURE

SHOW_TXT             PROCEDURE

WRITE_TO_FILE        PROCEDURE

GET_FILE_TXT         PROCEDURE

TEST_JOB             PROCEDURE

DO_EXE_IMM           PROCEDURE

T1                   PROCEDURE

CREATE_TABLE         PROCEDURE

SHOW_REC             PROCEDURE

ADD_DEPT             PROCEDURE

ADD_R                PROCEDURE

SET_VDO              PROCEDURE

GET_EMP_VDO_LEN      PROCEDURE

LOAD_TXT_DATA        PROCEDURE

CHK_SAL              PROCEDURE

TAB_NO               PROCEDURE

MY_CODE              PROCEDURE

FIRST_PRO            PROCEDURE

TEST2                PROCEDURE


22 rows selected.

SQL> SELECT TEXT FROM USER_SOURCE
  2  WHERE NAME='SHOW_REC';

TEXT

--------------------------------------------------------------------------------
----------------
PROCEDURE SHOW_REC(V_EMPNO EMP.EMPNO%TYPE DEFAULT 7788) IS

EMP_REC EMP%ROWTYPE;

BEGIN
```

```
SELECT * INTO EMP_REC FROM SCOTT.EMP

WHERE EMPNO=V_EMPNO;

DBMS_OUTPUT.PUT_LINE(EMP_REC.ENAME||'  '||EMP_REC.JOB||'  '||EMP_REC.SAL||'
'||EMP_REC.DEPTNO);
EXCEPTION

WHEN NO_DATA_FOUND THEN

DBMS_OUTPUT.PUT_LINE('RECORD NOT FOUND...');

END;


10 rows selected.

SQL>
SQL>
SQL>
SQL>
SQL>
SQL> ED
Wrote file afiedt.buf

  1  SELECT TEXT FROM USER_SOURCE
  2* WHERE NAME='FIRST_PRO'
SQL> /

TEXT

--------------------------------------------------------------------------------
----------------
PROCEDURE FIRST_PRO IS

BEGIN

DBMS_OUTPUT.PUT_LINE('ORACLE...'||'PLSQL');

END;


SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL> ED
Wrote file afiedt.buf

  1  SELECT TEXT FROM USER_SOURCE
  2* WHERE NAME='ADD_NEW_EMP'
SQL> /

TEXT

--------------------------------------------------------------------------------
----------------
PROCEDURE ADD_NEW_EMP
```

```
        (
         V_EMPNO NUMBER,
         V_ENAME EMP.ENAME%TYPE,
         V_JOB EMP.JOB%TYPE,
         V_SAL EMP.SAL%TYPE,
         V_DEPTNO EMP.DEPTNO%TYPE
         )IS
         BEGIN
       INSERT INTO EMP_TEST(EMPNO,ENAME,JOB,SAL,DEPTNO,REV_DATE)
       VALUES(V_EMPNO,V_ENAME,V_JOB,V_SAL,V_DEPTNO,SYSDATE);
        COMMIT;
       DBMS_OUTPUT.PUT_LINE('RECORD CREATED WITH EMPNO ...'||V_EMPNO);
END;

14 rows selected.

SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>  SELECT OBJECT_NAME NAMES,OBJECT_TYPE FROM USER_OBJECTS
  2   WHERE OBJECT_TYPE IN ('FUNCTION');

NAMES                OBJECT_TYPE
-------------------- -------------------
GET_ORD              FUNCTION
INS_REC              FUNCTION
GET_MGR              FUNCTION
GET_JOB              FUNCTION
GET_ID               FUNCTION
GET_WORDS            FUNCTION
GET_TAX              FUNCTION
GET_TABLE            FUNCTION
VALID_SAL            FUNCTION
NEW_REC              FUNCTION
```

```
10 rows selected.

SQL>
SQL>
SQL>
SQL>  SELECT TEXT FROM USER_SOURCE
  2   WHERE NAME='GET_TAX';

TEXT

--------------------------------------------------------------------------------
---------------
FUNCTION GET_TAX(V_SAL NUMBER)

RETURN NUMBER IS

V_TAX NUMBER :=0;

ANN_SAL NUMBER :=0;

BEGIN

ANN_SAL := V_SAL * 12;

IF ANN_SAL BETWEEN 15000 AND 20000 THEN

V_TAX := V_SAL * 5/100;

ELSIF ANN_SAL BETWEEN 20001 AND 30000 THEN

V_TAX := V_SAL * 7/100;

ELSIF ANN_SAL BETWEEN 30001 AND 40000 THEN

V_TAX := V_SAL * 9/100;

ELSIF ANN_SAL >40000 THEN

V_TAX := V_SAL * 10/100;

END IF;

RETURN(V_TAX);

END;


17 rows selected.

SQL> ED
Wrote file afiedt.buf

  1   SELECT TEXT FROM USER_SOURCE
  2*  WHERE NAME='VALID_SAL'
SQL> /

TEXT

--------------------------------------------------------------------------------
---------------
FUNCTION VALID_SAL(V_SAL NUMBER) RETURN BOOLEAN

IS
```

```
V_GRADE NUMBER :=0;

BEGIN

SELECT GRADE INTO V_GRADE FROM SCOTT.SALGRADE

WHERE V_SAL BETWEEN LOSAL AND HISAL;

RETURN(TRUE);

EXCEPTION

WHEN NO_DATA_FOUND THEN

RETURN(FALSE);

END;


11 rows selected.

SQL> ED
Wrote file afiedt.buf

  1   SELECT TEXT FROM USER_SOURCE
  2*  WHERE NAME='GET_MGR'
SQL> /

TEXT

--------------------------------------------------------------------------------
----------------
FUNCTION GET_MGR(V_MGR NUMBER) RETURN BOOLEAN

IS

ID NUMBER :=0;

BEGIN

SELECT EMPNO INTO ID FROM SCOTT.EMP

WHERE EMPNO=V_MGR;

RETURN(TRUE);

EXCEPTION

WHEN NO_DATA_FOUND THEN

RETURN(FALSE);

END;


11 rows selected.

SQL> ED
Wrote file afiedt.buf

  1  CREATE OR REPLACE PACKAGE P1 IS
  2  ------------STARTING PACKAGE SPECIFICATION-----------------
```

```
  3  -------------------PROCEDURE--------------------
  4  PROCEDURE SHOW_REC(V_EMPNO EMP.EMPNO%TYPE DEFAULT 7788);
  5  PROCEDURE FIRST_PRO;
  6  PROCEDURE ADD_NEW_EMP
  7          (
  8           V_EMPNO NUMBER,
  9           V_ENAME EMP.ENAME%TYPE,
 10           V_JOB EMP.JOB%TYPE,
 11           V_SAL EMP.SAL%TYPE,
 12           V_DEPTNO EMP.DEPTNO%TYPE
 13           );
 14  ----------------------FUNCTIONS--------------------
 15  FUNCTION GET_TAX(V_SAL NUMBER)RETURN NUMBER;
 16  FUNCTION VALID_SAL(V_SAL NUMBER) RETURN BOOLEAN;
 17  FUNCTION GET_MGR(V_MGR NUMBER) RETURN BOOLEAN;
 18  ------------END OF  PACKAGE SPECIFICATION-----------------
 19* END;
 20  /

Package created.

SQL>
SQL>
SQL>
SQL>
SQL>
SQL> ED
Wrote file afiedt.buf

  1  CREATE OR REPLACE PACKAGE P1 IS
  2  ------------STARTING PACKAGE SPECIFICATION-----------------
  3  -------------------PROCEDURE--------------------
  4  PROCEDURE SHOW_REC(V_EMPNO EMP.EMPNO%TYPE DEFAULT 7788);
  5  PROCEDURE FIRST_PRO;
  6  PROCEDURE ADD_NEW_EMP
  7          (
  8           V_EMPNO NUMBER,
  9           V_ENAME EMP.ENAME%TYPE,
 10           V_JOB EMP.JOB%TYPE,
 11           V_SAL EMP.SAL%TYPE,
 12           V_DEPTNO EMP.DEPTNO%TYPE
 13           );
 14  ----------------------FUNCTIONS--------------------
 15  FUNCTION GET_TAX(V_SAL NUMBER)RETURN NUMBER;
 16  FUNCTION VALID_SAL(V_SAL NUMBER) RETURN BOOLEAN;
 17  FUNCTION GET_MGR(V_MGR NUMBER) RETURN BOOLEAN;
 18  ------------END OF  PACKAGE SPECIFICATION-----------------
 19* END;
SQL> /

Package created.

SQL> ED
Wrote file afiedt.buf

  1  CREATE OR REPLACE PACKAGE P1 IS
  2  ------------STARTING PACKAGE SPECIFICATION-----------------
  3  V_SALARY NUMBER :=0;
  4  -------------------PROCEDURE--------------------
  5  PROCEDURE SHOW_REC(V_EMPNO EMP.EMPNO%TYPE DEFAULT 7788);
  6  PROCEDURE FIRST_PRO;
  7  PROCEDURE ADD_NEW_EMP
  8          (
```

```
  9           V_EMPNO NUMBER,
 10           V_ENAME EMP.ENAME%TYPE,
 11           V_JOB EMP.JOB%TYPE,
 12           V_SAL EMP.SAL%TYPE,
 13           V_DEPTNO EMP.DEPTNO%TYPE
 14           );
 15  ----------------------FUNCTIONS--------------------
 16  FUNCTION GET_TAX(V_SAL NUMBER)RETURN NUMBER;
 17  FUNCTION VALID_SAL(V_SAL NUMBER) RETURN BOOLEAN;
 18  FUNCTION GET_MGR(V_MGR NUMBER) RETURN BOOLEAN;
 19  ------------END OF  PACKAGE SPECIFICATION-----------------
 20* END;
SQL> /

Package created.

SQL>
SQL>
SQL>
SQL>
SQL> ED
Wrote file afiedt.buf

  1  CREATE OR REPLACE PACKAGE BODY  P1;
  2  ------------------------------PACKGAR
  3  NEW_VAR NUMBER:=0; --------------PVT VARIABLE--------------
  4  ----------------------PROCEDURE 1---------------
  5  PROCEDURE SHOW_REC(V_EMPNO EMP.EMPNO%TYPE DEFAULT 7788) IS
  6  EMP_REC EMP%ROWTYPE;
  7  BEGIN
  8  SELECT * INTO EMP_REC FROM SCOTT.EMP
  9  WHERE EMPNO=V_EMPNO;
 10  DBMS_OUTPUT.PUT_LINE(EMP_REC.ENAME||'  '||EMP_REC.JOB||'  '||EMP_REC.SAL||'
'||EMP_REC.DEPTNO);
 11  EXCEPTION
 12  WHEN NO_DATA_FOUND THEN
 13  DBMS_OUTPUT.PUT_LINE('RECORD NOT FOUND...');
 14  END;
 15  ---------------------------PROCEDURE 2-------------
 16  PROCEDURE FIRST_PRO IS
 17  BEGIN
 18  DBMS_OUTPUT.PUT_LINE('ORACLE...'||'PLSQL');
 19  END;
 20  ----------------------PROCEDURE 3--------------------
 21  PROCEDURE ADD_NEW_EMP
 22          (
 23           V_EMPNO NUMBER,
 24           V_ENAME EMP.ENAME%TYPE,
 25           V_JOB EMP.JOB%TYPE,
 26           V_SAL EMP.SAL%TYPE,
 27           V_DEPTNO EMP.DEPTNO%TYPE
 28           )IS
 29           BEGIN
 30        INSERT INTO EMP_TEST(EMPNO,ENAME,JOB,SAL,DEPTNO,REV_DATE)
 31        VALUES(V_EMPNO,V_ENAME,V_JOB,V_SAL,V_DEPTNO,SYSDATE);
 32         COMMIT;
 33        DBMS_OUTPUT.PUT_LINE('RECORD CREATED WITH EMPNO ...'||V_EMPNO);
 34  END;
 35  ----------------------------FUNCTION 1-------------------------
 36  FUNCTION GET_TAX(V_SAL NUMBER)
 37  RETURN NUMBER IS
 38  V_TAX NUMBER :=0;
 39  ANN_SAL NUMBER :=0;
```

```
 40  BEGIN
 41  ANN_SAL := V_SAL * 12;
 42  IF ANN_SAL BETWEEN 15000 AND 20000 THEN
 43  V_TAX := V_SAL * 5/100;
 44  ELSIF ANN_SAL BETWEEN 20001 AND 30000 THEN
 45  V_TAX := V_SAL * 7/100;
 46  ELSIF ANN_SAL BETWEEN 30001 AND 40000 THEN
 47  V_TAX := V_SAL * 9/100;
 48  ELSIF ANN_SAL >40000 THEN
 49  V_TAX := V_SAL * 10/100;
 50  END IF;
 51  RETURN(V_TAX);
 52  END;
 53  -------------------------------FUNCTION 2----------------------------
 54  FUNCTION VALID_SAL(V_SAL NUMBER) RETURN BOOLEAN
 55  IS
 56  V_GRADE NUMBER :=0;
 57  BEGIN
 58  SELECT GRADE INTO V_GRADE FROM SCOTT.SALGRADE
 59  WHERE V_SAL BETWEEN LOSAL AND HISAL;
 60  RETURN(TRUE);
 61  EXCEPTION
 62  WHEN NO_DATA_FOUND THEN
 63  RETURN(FALSE);
 64  END;
 65  -------------------------------FUNCTION 3 --------------------
 66  FUNCTION GET_MGR(V_MGR NUMBER) RETURN BOOLEAN
 67  IS
 68  ID NUMBER :=0;
 69  BEGIN
 70  SELECT EMPNO INTO ID FROM SCOTT.EMP
 71  WHERE EMPNO=V_MGR;
 72  RETURN(TRUE);
 73  EXCEPTION
 74  WHEN NO_DATA_FOUND THEN
 75  RETURN(FALSE);
 76  END;
 77  -------------------------------END OF PACKAGE BODY-----------
 78* END;
 79  /
CREATE OR REPLACE PACKAGE BODY  P1;
                                  *
ERROR at line 1:
ORA-00911: invalid character


SQL>
SQL>
SQL>
SQL>
SQL> ED
Wrote file afiedt.buf

  1  CREATE OR REPLACE PACKAGE BODY  P1 IS
  2  -------------------------------PACKGAR
  3  NEW_VAR NUMBER:=0; --------------PVT VARIABLE--------------
  4  ----------------------PROCEDURE 1---------------
  5  PROCEDURE SHOW_REC(V_EMPNO EMP.EMPNO%TYPE DEFAULT 7788) IS
  6  EMP_REC EMP%ROWTYPE;
  7  BEGIN
  8  SELECT * INTO EMP_REC FROM SCOTT.EMP
  9  WHERE EMPNO=V_EMPNO;
 10  DBMS_OUTPUT.PUT_LINE(EMP_REC.ENAME||'  '||EMP_REC.JOB||'  '||EMP_REC.SAL||'
```

```
'||EMP_REC.DEPTNO);
 11  EXCEPTION
 12  WHEN NO_DATA_FOUND THEN
 13  DBMS_OUTPUT.PUT_LINE('RECORD NOT FOUND...');
 14  END;
 15  --------------------------PROCEDURE 2------------
 16  PROCEDURE FIRST_PRO IS
 17  BEGIN
 18  DBMS_OUTPUT.PUT_LINE('ORACLE...'||'PLSQL');
 19  END;
 20  ----------------------PROCEDURE 3--------------------
 21  PROCEDURE ADD_NEW_EMP
 22          (
 23           V_EMPNO NUMBER,
 24           V_ENAME EMP.ENAME%TYPE,
 25           V_JOB EMP.JOB%TYPE,
 26           V_SAL EMP.SAL%TYPE,
 27           V_DEPTNO EMP.DEPTNO%TYPE
 28           )IS
 29           BEGIN
 30        INSERT INTO EMP_TEST(EMPNO,ENAME,JOB,SAL,DEPTNO,REV_DATE)
 31        VALUES(V_EMPNO,V_ENAME,V_JOB,V_SAL,V_DEPTNO,SYSDATE);
 32         COMMIT;
 33        DBMS_OUTPUT.PUT_LINE('RECORD CREATED WITH EMPNO ...'||V_EMPNO);
 34  END;
 35  ----------------------------FUNCTION 1-----------------------
 36  FUNCTION GET_TAX(V_SAL NUMBER)
 37  RETURN NUMBER IS
 38  V_TAX NUMBER :=0;
 39  ANN_SAL NUMBER :=0;
 40  BEGIN
 41  ANN_SAL := V_SAL * 12;
 42  IF ANN_SAL BETWEEN 15000 AND 20000 THEN
 43  V_TAX := V_SAL * 5/100;
 44  ELSIF ANN_SAL BETWEEN 20001 AND 30000 THEN
 45  V_TAX := V_SAL * 7/100;
 46  ELSIF ANN_SAL BETWEEN 30001 AND 40000 THEN
 47  V_TAX := V_SAL * 9/100;
 48  ELSIF ANN_SAL >40000 THEN
 49  V_TAX := V_SAL * 10/100;
 50  END IF;
 51  RETURN(V_TAX);
 52  END;
 53  ------------------------------FUNCTION 2---------------------------
 54  FUNCTION VALID_SAL(V_SAL NUMBER) RETURN BOOLEAN
 55  IS
 56  V_GRADE NUMBER :=0;
 57  BEGIN
 58  SELECT GRADE INTO V_GRADE FROM SCOTT.SALGRADE
 59  WHERE V_SAL BETWEEN LOSAL AND HISAL;
 60  RETURN(TRUE);
 61  EXCEPTION
 62  WHEN NO_DATA_FOUND THEN
 63  RETURN(FALSE);
 64  END;
 65  ------------------------------FUNCTION 3 --------------------
 66  FUNCTION GET_MGR(V_MGR NUMBER) RETURN BOOLEAN
 67  IS
 68  ID NUMBER :=0;
 69  BEGIN
 70  SELECT EMPNO INTO ID FROM SCOTT.EMP
 71  WHERE EMPNO=V_MGR;
 72  RETURN(TRUE);
```

```
 73  EXCEPTION
 74  WHEN NO_DATA_FOUND THEN
 75  RETURN(FALSE);
 76  END;
 77  ------------------------------END OF PACKAGE BODY-----------
 78* END;
SQL> /

Package body created.

SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL> SELECT TEXT FROM USER_SOURCE
  2  WHERE NAME='P1';

TEXT

--------------------------------------------------------------------------------
---------------
PACKAGE P1 IS

------------STARTING PACKAGE SPECIFICATION-----------------

V_SALARY NUMBER :=0;

-------------------PROCEDURE--------------------

PROCEDURE SHOW_REC(V_EMPNO EMP.EMPNO%TYPE DEFAULT 7788);

PROCEDURE FIRST_PRO;

PROCEDURE ADD_NEW_EMP

        (

         V_EMPNO NUMBER,

         V_ENAME EMP.ENAME%TYPE,

         V_JOB EMP.JOB%TYPE,

         V_SAL EMP.SAL%TYPE,

         V_DEPTNO EMP.DEPTNO%TYPE

         );

----------------------FUNCTIONS--------------------

FUNCTION GET_TAX(V_SAL NUMBER)RETURN NUMBER;

FUNCTION VALID_SAL(V_SAL NUMBER) RETURN BOOLEAN;

FUNCTION GET_MGR(V_MGR NUMBER) RETURN BOOLEAN;

------------END OF  PACKAGE SPECIFICATION-----------------

END;
```

```
PACKAGE BODY  P1 IS

------------------------------PACKGAR

NEW_VAR NUMBER:=0; --------------PVT VARIABLE--------------

----------------------PROCEDURE 1---------------

PROCEDURE SHOW_REC(V_EMPNO EMP.EMPNO%TYPE DEFAULT 7788) IS

EMP_REC EMP%ROWTYPE;

BEGIN

SELECT * INTO EMP_REC FROM SCOTT.EMP

WHERE EMPNO=V_EMPNO;

DBMS_OUTPUT.PUT_LINE(EMP_REC.ENAME||'  '||EMP_REC.JOB||'  '||EMP_REC.SAL||'
'||EMP_REC.DEPTNO);
EXCEPTION

WHEN NO_DATA_FOUND THEN

DBMS_OUTPUT.PUT_LINE('RECORD NOT FOUND...');

END;

---------------------------PROCEDURE 2-------------

PROCEDURE FIRST_PRO IS

BEGIN

DBMS_OUTPUT.PUT_LINE('ORACLE...'||'PLSQL');

END;

-----------------------PROCEDURE 3---------------------

PROCEDURE ADD_NEW_EMP

        (

         V_EMPNO NUMBER,

         V_ENAME EMP.ENAME%TYPE,

         V_JOB EMP.JOB%TYPE,

         V_SAL EMP.SAL%TYPE,

         V_DEPTNO EMP.DEPTNO%TYPE

         )IS

         BEGIN

       INSERT INTO EMP_TEST(EMPNO,ENAME,JOB,SAL,DEPTNO,REV_DATE)

       VALUES(V_EMPNO,V_ENAME,V_JOB,V_SAL,V_DEPTNO,SYSDATE);

        COMMIT;
```

```
      DBMS_OUTPUT.PUT_LINE('RECORD CREATED WITH EMPNO ...'||V_EMPNO);
END;
----------------------------FUNCTION 1------------------------
FUNCTION GET_TAX(V_SAL NUMBER)
RETURN NUMBER IS
V_TAX NUMBER :=0;
ANN_SAL NUMBER :=0;
BEGIN
ANN_SAL := V_SAL * 12;
IF ANN_SAL BETWEEN 15000 AND 20000 THEN
V_TAX := V_SAL * 5/100;
ELSIF ANN_SAL BETWEEN 20001 AND 30000 THEN
V_TAX := V_SAL * 7/100;
ELSIF ANN_SAL BETWEEN 30001 AND 40000 THEN
V_TAX := V_SAL * 9/100;
ELSIF ANN_SAL >40000 THEN
V_TAX := V_SAL * 10/100;
END IF;
RETURN(V_TAX);
END;
-------------------------------FUNCTION 2----------------------------
FUNCTION VALID_SAL(V_SAL NUMBER) RETURN BOOLEAN
IS
V_GRADE NUMBER :=0;
BEGIN
SELECT GRADE INTO V_GRADE FROM SCOTT.SALGRADE
WHERE V_SAL BETWEEN LOSAL AND HISAL;
RETURN(TRUE);
EXCEPTION
WHEN NO_DATA_FOUND THEN
RETURN(FALSE);
```

```
END;

------------------------------FUNCTION 3 --------------------

FUNCTION GET_MGR(V_MGR NUMBER) RETURN BOOLEAN

IS

ID NUMBER :=0;

BEGIN

SELECT EMPNO INTO ID FROM SCOTT.EMP

WHERE EMPNO=V_MGR;

RETURN(TRUE);

EXCEPTION

WHEN NO_DATA_FOUND THEN

RETURN(FALSE);

END;

------------------------------END OF PACKAGE BODY-----------


TEXT

--------------------------------------------------------------------------------
----------------
END;


98 rows selected.

SQL> DESC P1
PROCEDURE ADD_NEW_EMP
 Argument Name                  Type                    In/Out Default?
 ------------------------------ ----------------------- ------ --------
 V_EMPNO                        NUMBER                  IN
 V_ENAME                        VARCHAR2(10)            IN
 V_JOB                          VARCHAR2(9)             IN
 V_SAL                          NUMBER(7,2)             IN
 V_DEPTNO                       NUMBER(2)               IN
PROCEDURE FIRST_PRO
FUNCTION GET_MGR RETURNS BOOLEAN
 Argument Name                  Type                    In/Out Default?
 ------------------------------ ----------------------- ------ --------
 V_MGR                          NUMBER                  IN
FUNCTION GET_TAX RETURNS NUMBER
 Argument Name                  Type                    In/Out Default?
 ------------------------------ ----------------------- ------ --------
 V_SAL                          NUMBER                  IN
PROCEDURE SHOW_REC
 Argument Name                  Type                    In/Out Default?
 ------------------------------ ----------------------- ------ --------
 V_EMPNO                        NUMBER(4)               IN     DEFAULT
FUNCTION VALID_SAL RETURNS BOOLEAN
 Argument Name                  Type                    In/Out Default?
 ------------------------------ ----------------------- ------ --------
```

```
 V_SAL                          NUMBER                  IN

SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL> DESC DBMS_OUTPUT
PROCEDURE DISABLE
PROCEDURE ENABLE
 Argument Name                  Type                    In/Out Default?
 ------------------------------ ----------------------- ------ --------
 BUFFER_SIZE                    NUMBER(38)              IN     DEFAULT
PROCEDURE GET_LINE
 Argument Name                  Type                    In/Out Default?
 ------------------------------ ----------------------- ------ --------
 LINE                           VARCHAR2                OUT
 STATUS                         NUMBER(38)              OUT
PROCEDURE GET_LINES
 Argument Name                  Type                    In/Out Default?
 ------------------------------ ----------------------- ------ --------
 LINES                          TABLE OF VARCHAR2(32767) OUT
 NUMLINES                       NUMBER(38)              IN/OUT
PROCEDURE GET_LINES
 Argument Name                  Type                    In/Out Default?
 ------------------------------ ----------------------- ------ --------
 LINES                          DBMSOUTPUT_LINESARRAY   OUT
 NUMLINES                       NUMBER(38)              IN/OUT
PROCEDURE NEW_LINE
PROCEDURE PUT
 Argument Name                  Type                    In/Out Default?
 ------------------------------ ----------------------- ------ --------
 A                              VARCHAR2                IN
PROCEDURE PUT_LINE
 Argument Name                  Type                    In/Out Default?
 ------------------------------ ----------------------- ------ --------
 A                              VARCHAR2                IN

SQL> EXEC P1.FIRST_PRO;
ORACLE...PLSQL


PL/SQL procedure successfully completed.

SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL> EXEC FIRST_PRO;
ORACLE...PLSQL


PL/SQL procedure successfully completed.

SQL>
SQL>
```

```
SQL>
SQL>
SQL> DROP PROCEDURE FIRST_PRO;

Procedure dropped.

SQL> EXEC FIRST_PRO;
BEGIN FIRST_PRO; END;

      *
ERROR at line 1:
ORA-06550: line 1, column 7:
PLS-00201: identifier 'FIRST_PRO' must be declared
ORA-06550: line 1, column 7:
PL/SQL: Statement ignored


SQL> EXEC P1.FIRST_PRO;
ORACLE...PLSQL


PL/SQL procedure successfully completed.

SQL>
SQL>
SQL>
SQL>
SQL> SELECT SAL,P1.GET_TAX(SAL),GET_TAX(SAL) FROM EMP;

       SAL P1.GET_TAX(SAL) GET_TAX(SAL)

---------- --------------- ------------

       900               0            0

      1600              80           80

      1250            62.5         62.5

      2975          267.75       267.75

      1250            62.5         62.5

      2850           256.5        256.5

      2450           171.5        171.5

      3000             270          270

      5000             500          500

      1500              75           75

      1100               0            0

      1000               0            0

     45666          4566.6       4566.6

      1300              65           65

      1000               0            0
```

```
         1                0             0

      1000                0             0

      1000                0             0

      1000                0             0

      1000                0             0

20 rows selected.

SQL>
SQL>
SQL> SPOOL OFF
```

```
SQL>
SQL>
SQL> DECLARE
  2  .
SQL> ED
Wrote file afiedt.buf

  1  CREATE OR REPLACE PACKAGE BODYLESS_PACK IS
  2  ----------------------BODYLESS PACKAGE----------------
  3  CURSOR C1 IS SELECT EMPNO,ENAME,JOB,D.DEPTNO DEPT_ID,SAL,DNAME
  4            FROM EMP E JOIN DEPT D
  5            ON E.DEPTNO=D.DEPTNO;
  6  CNTR NUMBER :=0;
  7  MULTI_ROWS EXCEPTION;
  8  PRAGMA EXCETPION_INIT(MULTI_ROWS,-1422);
  9  RECORD_NOT_FOUND EXCEPTION;
 10  PRAGMA EXCETPION_INIT(RECORD_NOT_FOUND,100);
 11* END;
 12  /

Warning: Package created with compilation errors.

SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL> SHOW ERR
No errors.
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL> ALTER PACKAGE BODYLESS_PACK COMPILE;

Warning: Package altered with compilation errors.

SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL> SHOW ERR
Errors for PACKAGE BODYLESS_PACK:

LINE/COL ERROR

-------- -----------------------------------------------------------------
```

```
8/8      PLS-00127: Pragma EXCETPION_INIT is not a supported pragma

10/8     PLS-00127: Pragma EXCETPION_INIT is not a supported pragma

SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL> ED
Wrote file afiedt.buf

  1* ALTER PACKAGE BODYLESS_PACK COMPILE
SQL>
SQL>  CREATE OR REPLACE PACKAGE BODYLESS_PACK IS
  2   ----------------------BODYLESS PACKAGE----------------
  3   CURSOR C1 IS SELECT EMPNO,ENAME,JOB,D.DEPTNO DEPT_ID,SAL,DNAME
  4            FROM EMP E JOIN DEPT D
  5            ON E.DEPTNO=D.DEPTNO;
  6   CNTR NUMBER :=0;
  7   MULTI_ROWS EXCEPTION;
  8   PRAGMA EXCETPION_INIT(MULTI_ROWS,-1422);
  9   RECORD_NOT_FOUND EXCEPTION;
 10   PRAGMA EXCETPION_INIT(RECORD_NOT_FOUND,100);
 11   END;
 12  .
SQL> ED
Wrote file afiedt.buf

  1   CREATE OR REPLACE PACKAGE BODYLESS_PACK IS
  2   ----------------------BODYLESS PACKAGE----------------
  3   CURSOR C1 IS SELECT EMPNO,ENAME,JOB,D.DEPTNO DEPT_ID,SAL,DNAME
  4            FROM EMP E JOIN DEPT D
  5            ON E.DEPTNO=D.DEPTNO;
  6   CNTR NUMBER :=0;
  7   MULTI_ROWS EXCEPTION;
  8   PRAGMA EXCEPTION_INIT(MULTI_ROWS,-1422);
  9   RECORD_NOT_FOUND EXCEPTION;
 10   PRAGMA EXCEPTION_INIT(RECORD_NOT_FOUND,100);
 11*  END;
SQL> /

Package created.

SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL> //..
SQL>
SQL>
SQL>
SQL> ED
```

```
Wrote file afiedt.buf

  1  DECLARE
  2  V_JOB VARCHAR2(20):='&JOB';
  3  EMP_REC EMP%ROWTYPE;
  4  BEGIN
  5  SELECT * INTO EMP_REC FROM SCOTT.EMP
  6  WHERE JOB=V_JOB;
  7  &D(EMP_REC.EMPNO||' '||EMP_REC.ENAME||'  '||EMP_REC.JOB);
  8* END;
  9  /
Enter value for job: PRESIDENT
7839 KING  PRESIDENT


PL/SQL procedure successfully completed.

SQL>
SQL>
SQL>
SQL> /
Enter value for job: PRES
DECLARE
*
ERROR at line 1:
ORA-01403: no data found
ORA-06512: at line 5


SQL>
SQL>
SQL>
SQL> ED
Wrote file afiedt.buf

  1  DECLARE
  2  V_JOB VARCHAR2(20):='&JOB';
  3  EMP_REC EMP%ROWTYPE;
  4  BEGIN
  5  SELECT * INTO EMP_REC FROM SCOTT.EMP
  6  WHERE JOB=V_JOB;
  7  &D(EMP_REC.EMPNO||' '||EMP_REC.ENAME||'  '||EMP_REC.JOB);
  8  EXCEPTION
  9  WHEN BODYLESS_PACK.RECORD_NOT_FOUND THEN
 10  &D('RECORD NOT FOUND IN TABLE...');
 11* END;
 12  /
Enter value for job: SAL
RECORD NOT FOUND IN TABLE...


PL/SQL procedure successfully completed.

SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL> /
Enter value for job: SALESMAN
```

```
DECLARE
*
ERROR at line 1:
ORA-01422: exact fetch returns more than requested number of rows
ORA-06512: at line 5


SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL> ED
Wrote file afiedt.buf

  1  DECLARE
  2  V_JOB VARCHAR2(20):='&JOB';
  3  EMP_REC EMP%ROWTYPE;
  4  BEGIN
  5  SELECT * INTO EMP_REC FROM SCOTT.EMP
  6  WHERE JOB=V_JOB;
  7  &D(EMP_REC.EMPNO||' '||EMP_REC.ENAME||'  '||EMP_REC.JOB);
  8  EXCEPTION
  9  WHEN BODYLESS_PACK.RECORD_NOT_FOUND THEN
 10  &D('RECORD NOT FOUND IN TABLE...');
 11  WHEN BODYLESS_PACK.MULTI_ROWS THEN
 12  &D('MULTIPLE RECORDS FOUND....');
 13* END;
 14  /
Enter value for job: SALESMAN
MULTIPLE RECORDS FOUND....


PL/SQL procedure successfully completed.

SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL> ED
Wrote file afiedt.buf

  1  DECLARE
  2  V_JOB VARCHAR2(20):='&JOB';
  3  EMP_REC EMP%ROWTYPE;
  4  BEGIN
  5  SELECT * INTO EMP_REC FROM SCOTT.EMP
  6  WHERE JOB=V_JOB;
  7  &D(EMP_REC.EMPNO||' '||EMP_REC.ENAME||'  '||EMP_REC.JOB);
  8  EXCEPTION
  9  WHEN BODYLESS_PACK.RECORD_NOT_FOUND THEN
 10  &D('RECORD NOT FOUND IN TABLE...');
 11  WHEN BODYLESS_PACK.MULTI_ROWS THEN
 12  ------------------------------NESTED BLOCK-----------------------
 13  BEGIN
 14  FOR I IN BODYLESS_PACK.C1 LOOP
 15  &D( I.EMPNO||'  '||I.ENAME||'  '||I.JOB||' '||I.DNAME);
```

```
 16  END LOOP;
 17  END;
 18  ------------------------END OF NESTED BLOCK-----------------------
 19* END;
 20  /
Enter value for job: SALESMAN
5454  SMITH  CLERK RESEARCH

7499  ALLEN  SALESMAN SALES

7521  WARD  SALESMAN SALES

7566  JONES  MANAGER RESEARCH

7654  MARTIN  SALESMAN SALES

7698  BLAKE  MANAGER SALES

7782  CLARK  MANAGER ACCOUNTING

7788  SCOTT  ANALYST RESEARCH

7839  KING  PRESIDENT ACCOUNTING

7844  TURNER  SALESMAN SALES

7876  ADAMS  CLERK RESEARCH

7900  JAMES  CLERK SALES

7902  FORD  ANALYST RESEARCH

7934  MILLER  CLERK ACCOUNTING

7935  SCOTT  SALESMAN SALES

7936  SCOTT  SALESMAN SALES

7937  SCOTT  SALESMAN SALES

7938  SCOTT  SALESMAN SALES

7939  SCOTT  SALESMAN SALES

8000  SCOTT  SALESMAN SALES


PL/SQL procedure successfully completed.

SQL>
SQL>
SQL> ED
Wrote file afiedt.buf

  1  DECLARE
  2  V_JOB VARCHAR2(20):='&JOB';
  3  EMP_REC EMP%ROWTYPE;
  4  BEGIN
  5  SELECT * INTO EMP_REC FROM SCOTT.EMP
  6  WHERE JOB=V_JOB;
  7  &D(EMP_REC.EMPNO||' '||EMP_REC.ENAME||'  '||EMP_REC.JOB);
  8  EXCEPTION
  9  WHEN BODYLESS_PACK.RECORD_NOT_FOUND THEN
```

```
 10  &D('RECORD NOT FOUND IN TABLE...');
 11  WHEN BODYLESS_PACK.MULTI_ROWS T/
HEN
 12  -------------------------------NESTED BLOCK-----------------------
 13  BEGIN
 14  FOR I IN BODYLESS_PACK.C1 LOOP
 15  BODYLESS_PACK.CNTR := BODYLESS_PACK.CNTR + 1;
 16  &D(BODYLESS_PACK.CNTR||'  '|| I.EMPNO||'  '||I.ENAME||'  '||I.JOB||'
'||I.DNAME);
 17  END LOOP;
 18  END;
 19  ------------------------END OF NESTED BLOCK-----------------------
 20* END;
SQL> /
Enter value for job: SALESMAN
1  5454  SMITH  CLERK RESEARCH

2  7499  ALLEN  SALESMAN SALES

3  7521  WARD  SALESMAN SALES

4  7566  JONES  MANAGER RESEARCH

5  7654  MARTIN  SALESMAN SALES

6  7698  BLAKE  MANAGER SALES

7  7782  CLARK  MANAGER ACCOUNTING

8  7788  SCOTT  ANALYST RESEARCH

9  7839  KING  PRESIDENT ACCOUNTING

10  7844  TURNER  SALESMAN SALES

11  7876  ADAMS  CLERK RESEARCH

12  7900  JAMES  CLERK SALES

13  7902  FORD  ANALYST RESEARCH

14  7934  MILLER  CLERK ACCOUNTING

15  7935  SCOTT  SALESMAN SALES

16  7936  SCOTT  SALESMAN SALES

17  7937  SCOTT  SALESMAN SALES

18  7938  SCOTT  SALESMAN SALES

19  7939  SCOTT  SALESMAN SALES

20  8000  SCOTT  SALESMAN SALES


PL/SQL procedure successfully completed.

SQL>
SQL>
SQL> ED
Wrote file afiedt.buf
```

```
  1  DECLARE
  2  V_JOB VARCHAR2(20):='&JOB';
  3  EMP_REC EMP%ROWTYPE;
  4  BEGIN
  5  SELECT * INTO EMP_REC FROM SCOTT.EMP
  6  WHERE JOB=V_JOB;
  7  &D(EMP_REC.EMPNO||' '||EMP_REC.ENAME||'  '||EMP_REC.JOB);
  8  EXCEPTION
  9  WHEN BODYLESS_PACK.RECORD_NOT_FOUND THEN
 10  &D('RECORD NOT FOUND IN TABLE...');
 11  WHEN BODYLESS_PACK.MULTI_ROWS THEN
 12  ------------------------------NESTED BLOCK----------------------
 13  BEGIN
 14  FOR I IN BODYLESS_PACK.C1 LOOP
 15  IF I.JOB=V_JOB THEN
 16  BODYLESS_PACK.CNTR := BODYLESS_PACK.CNTR + 1;
 17  &D(BODYLESS_PACK.CNTR||'  '|| I.EMPNO||'  '||I.ENAME||'  '||I.JOB||'
'||I.DNAME);
 18  END IF;
 19  END LOOP;
 20  END;
 21  ------------------------END OF NESTED BLOCK-----------------------
 22* END;
 23  /
Enter value for job: SALESMAN
21  7499  ALLEN  SALESMAN SALES

22  7521  WARD  SALESMAN SALES

23  7654  MARTIN  SALESMAN SALES

24  7844  TURNER  SALESMAN SALES

25  7935  SCOTT  SALESMAN SALES

26  7936  SCOTT  SALESMAN SALES

27  7937  SCOTT  SALESMAN SALES

28  7938  SCOTT  SALESMAN SALES

29  7939  SCOTT  SALESMAN SALES

30  8000  SCOTT  SALESMAN SALES


PL/SQL procedure successfully completed.

SQL> /
Enter value for job: MANAGER
31  7566  JONES  MANAGER RESEARCH

32  7698  BLAKE  MANAGER SALES

33  7782  CLARK  MANAGER ACCOUNTING


PL/SQL procedure successfully completed.

SQL>
SQL>
SQL>
SQL>
```

```
SQL> ED
Wrote file afiedt.buf

  1  DECLARE
  2  V_JOB VARCHAR2(20):='&JOB';
  3  EMP_REC EMP%ROWTYPE;
  4  BEGIN
  5  SELECT * INTO EMP_REC FROM SCOTT.EMP
  6  WHERE JOB=V_JOB;
  7  &D(EMP_REC.EMPNO||' '||EMP_REC.ENAME||'  '||EMP_REC.JOB);
  8  EXCEPTION
  9  WHEN BODYLESS_PACK.RECORD_NOT_FOUND THEN
 10  &D('RECORD NOT FOUND IN TABLE...');
 11  WHEN BODYLESS_PACK.MULTI_ROWS THEN
 12  -------------------------------NESTED BLOCK----------------------
 13  BEGIN
 14  FOR I IN BODYLESS_PACK.C1 LOOP
 15  IF I.JOB=V_JOB THEN
 16  BODYLESS_PACK.CNTR := BODYLESS_PACK.CNTR + 1;
 17  &D(BODYLESS_PACK.CNTR||'  '|| I.EMPNO||'  '||I.ENAME||'  '||I.JOB||'
'||I.DNAME);
 18  END IF;
 19  END LOOP;
 20  END;
 21  ------------------------END OF NESTED BLOCK-----------------------
 22* END;
SQL> .
SQL> SELECT OBJECT_NAME FROM USER_OBJECTS
  2  WHERE OBJECT_TYPE='PROCEDURE';

OBJECT_NAME

--------------------------------------------------------------------------------
----------------
ADD_EMP

ADD_NEW_EMP

EMP_POSTING

DEL_REC

SHOW_TXT

WRITE_TO_FILE

GET_FILE_TXT

TEST_JOB

DO_EXE_IMM

T1

CREATE_TABLE

SHOW_REC

ADD_DEPT

ADD_R

SET_VDO
```

```
GET_EMP_VDO_LEN

LOAD_TXT_DATA

CHK_SAL

TAB_NO

MY_CODE

TEST2


21 rows selected.

SQL>
SQL> SELECT TEXT FROM USER_SOURCE
  2  WHERE NAME='SHOW_REC';

TEXT

--------------------------------------------------------------------------------
----------------
PROCEDURE SHOW_REC(V_EMPNO EMP.EMPNO%TYPE DEFAULT 7788) IS

EMP_REC EMP%ROWTYPE;

BEGIN

SELECT * INTO EMP_REC FROM SCOTT.EMP

WHERE EMPNO=V_EMPNO;

DBMS_OUTPUT.PUT_LINE(EMP_REC.ENAME||'  '||EMP_REC.JOB||'  '||EMP_REC.SAL||'
'||EMP_REC.DEPTNO);
EXCEPTION

WHEN NO_DATA_FOUND THEN

DBMS_OUTPUT.PUT_LINE('RECORD NOT FOUND...');

END;


10 rows selected.

SQL>
SQL>
SQL>
SQL>
SQL>
SQL> ED
Wrote file afiedt.buf

  1  SELECT TEXT FROM USER_SOURCE
  2* WHERE NAME='ADD_EMP'
SQL> /

TEXT

--------------------------------------------------------------------------------
----------------
```

```
PROCEDURE ADD_EMP
   (
       V_EMPNO EMP.EMPNO%TYPE,
       V_ENAME EMP.ENAME%TYPE,
       V_JOB   EMP.JOB%TYPE,
       V_SAL   EMP.SAL%TYPE,
       V_DEPTNO EMP.DEPTNO%TYPE,
       V_MGR    EMP.MGR%TYPE
   ) IS
   VALID_SALARY BOOLEAN;
   VALID_MGR BOOLEAN;
   ----------------------------PVT PROCEDURE--------------------
   PROCEDURE NEW_ID(ID OUT NUMBER) IS
   BEGIN
   SELECT MAX(EMPNO)+1 INTO ID FROM SCOTT.EMP;
   END;
   ----------------------------END OF PVT PROCEDURE-------------------
      BEGIN
   VALID_SALARY := VALID_SAL(V_SAL);  ----CALLING FUNCTION
   VALID_MGR  := GET_MGR(V_MGR);   -----CALLING FUNCTION
   IF VALID_SALARY=FALSE OR VALID_MGR=FALSE THEN
   RAISE_APPLICATION_ERROR(-20100,'INVALID SALARY OR INVALID MANAGER....' );
   END IF;
      INSERT INTO EMP(EMPNO,ENAME,JOB,SAL,DEPTNO,MGR)
       VALUES(V_EMPNO,V_ENAME,V_JOB,V_SAL,V_DEPTNO,V_MGR);
      COMMIT;
      DBMS_OUTPUT.PUT_LINE('RECORD CREATED WITH EMPNO '||V_EMPNO);
      EXCEPTION
      WHEN DUP_VAL_ON_INDEX  THEN
      ------------------------NESTED  BLOCK-----------------
      DECLARE
      ID NUMBER :=0;
```

```
      BEGIN

   ----   SELECT MAX(EMPNO)+1 INTO ID FROM SCOTT.EMP;

   -----   ID := GET_ID('EMP') ;-------------CALLING PUBLIC FUNCTION

   ----    NEW_ID(ID);   --------CALLING PUBLIC PROCEDURE

       NEW_ID(ID);   --------CALLING PVT PROCEDURE

   INSERT INTO EMP(EMPNO,ENAME,JOB,SAL,DEPTNO,MGR)

      VALUES(ID,V_ENAME,V_JOB,V_SAL,V_DEPTNO,V_MGR);

      COMMIT;

      DBMS_OUTPUT.PUT_LINE('RECORD CREATED WITH EMPNO '||ID ||' INSTEAD OF
'||V_EMPNO);
      END;

      ------------------------------END OF NESTED BLOCK-------------

      ----WHEN OTHERS THEN

      ----DBMS_OUTPUT.PUT_LINE(SQLCODE||'   '||SQLERRM);

    END;


46 rows selected.

SQL> ED
Wrote file afiedt.buf

  1  SELECT TEXT FROM USER_SOURCE
  2* WHERE NAME='GET_TAX'
SQL> /

TEXT

--------------------------------------------------------------------------------
----------------
FUNCTION GET_TAX(V_SAL NUMBER)

RETURN NUMBER IS

V_TAX NUMBER :=0;

ANN_SAL NUMBER :=0;

BEGIN

ANN_SAL := V_SAL * 12;

IF ANN_SAL BETWEEN 15000 AND 20000 THEN

V_TAX := V_SAL * 5/100;

ELSIF ANN_SAL BETWEEN 20001 AND 30000 THEN

V_TAX := V_SAL * 7/100;
```

```
ELSIF ANN_SAL BETWEEN 30001 AND 40000 THEN

V_TAX := V_SAL * 9/100;

ELSIF ANN_SAL >40000 THEN

V_TAX := V_SAL * 10/100;

END IF;

RETURN(V_TAX);

END;


17 rows selected.

SQL>
SQL>
SQL>
SQL>
SQL> ED
Wrote file afiedt.buf

  1  SELECT TEXT FROM USER_SOURCE
  2* WHERE NAME='VALID_SAL'
SQL> /

TEXT

--------------------------------------------------------------------------------
----------------
FUNCTION VALID_SAL(V_SAL NUMBER) RETURN BOOLEAN

IS

V_GRADE NUMBER :=0;

BEGIN

SELECT GRADE INTO V_GRADE FROM SCOTT.SALGRADE

WHERE V_SAL BETWEEN LOSAL AND HISAL;

RETURN(TRUE);

EXCEPTION

WHEN NO_DATA_FOUND THEN

RETURN(FALSE);

END;


11 rows selected.

SQL>
SQL>
SQL>
SQL> ED
Wrote file afiedt.buf
```

```
  1  CREATE OR REPLACE PACKAGE MY_PACK IS
  2  --------------------PROCEDURE 1 ADD_EMP-------------
  3  PROCEDURE ADD_EMP
  4     (
  5         V_EMPNO EMP.EMPNO%TYPE,
  6         V_ENAME EMP.ENAME%TYPE,
  7         V_JOB   EMP.JOB%TYPE,
  8         V_SAL   EMP.SAL%TYPE,
  9         V_DEPTNO EMP.DEPTNO%TYPE,
 10         V_MGR    EMP.MGR%TYPE
 11     ) ;
 12  -------------------------PROCEDURE 2 SHOW_REC--------------
 13  PROCEDURE SHOW_REC(V_EMPNO EMP.EMPNO%TYPE DEFAULT 7788);
 14  ------------------------------FUNCTION 1 GET_TAX---------
 15  FUNCTION GET_TAX(V_SAL NUMBER) RETURN NUMBER;
 16  --------------------------FUNCTION 2 VALID_SAL------------
 17  FUNCTION VALID_SAL(V_SAL NUMBER) RETURN BOOLEAN;
 18* END;
 19  /

Package created.

SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL> ED
Wrote file afiedt.buf

  1  CREATE OR REPLACE PACKAGE BODY MY_PACK IS
  2  --------------------STARTING PACKAGE BODY------------
  3  --------------------PROCEDURE 1 -------------------
  4  PROCEDURE ADD_EMP
  5     (
  6         V_EMPNO EMP.EMPNO%TYPE,
  7         V_ENAME EMP.ENAME%TYPE,
  8         V_JOB   EMP.JOB%TYPE,
  9         V_SAL   EMP.SAL%TYPE,
 10         V_DEPTNO EMP.DEPTNO%TYPE,
 11         V_MGR    EMP.MGR%TYPE
 12     ) IS
 13     VALID_SALARY BOOLEAN;
 14     VALID_MGR BOOLEAN;
 15     ----------------------------PVT PROCEDURE--------------------
 16     PROCEDURE NEW_ID(ID OUT NUMBER) IS
 17     BEGIN
 18     SELECT MAX(EMPNO)+1 INTO ID FROM SCOTT.EMP;
 19     END;
 20     ----------------------------END OF PVT PROCEDURE-------------------
 21        BEGIN
 22     VALID_SALARY := VALID_SAL(V_SAL);  ----CALLING FUNCTION
 23     VALID_MGR  := GET_MGR(V_MGR);   -----CALLING FUNCTION
 24     IF VALID_SALARY=FALSE OR VALID_MGR=FALSE THEN
 25     RAISE_APPLICATION_ERROR(-20100,'INVALID SALARY OR INVALID MANAGER....' );
 26     END IF;
 27        INSERT INTO EMP(EMPNO,ENAME,JOB,SAL,DEPTNO,MGR)
 28         VALUES(V_EMPNO,V_ENAME,V_JOB,V_SAL,V_DEPTNO,V_MGR);
 29        COMMIT;
 30        DBMS_OUTPUT.PUT_LINE('RECORD CREATED WITH EMPNO '||V_EMPNO);
 31        EXCEPTION
```

```
32        WHEN DUP_VAL_ON_INDEX  THEN
33        ------------------------NESTED  BLOCK-----------------
34        DECLARE
35        ID NUMBER :=0;
36        BEGIN
37     ----   SELECT MAX(EMPNO)+1 INTO ID FROM SCOTT.EMP;
38     -----   ID := GET_ID('EMP') ;-------------CALLING PUBLIC FUNCTION
39     ----    NEW_ID(ID);   --------CALLING PUBLIC PROCEDURE
40         NEW_ID(ID);   --------CALLING PVT PROCEDURE
41     INSERT INTO EMP(EMPNO,ENAME,JOB,SAL,DEPTNO,MGR)
42        VALUES(ID,V_ENAME,V_JOB,V_SAL,V_DEPTNO,V_MGR);
43        COMMIT;
44        DBMS_OUTPUT.PUT_LINE('RECORD CREATED WITH EMPNO '||ID ||' INSTEAD OF
'||V_EMPNO);
45        END;
46        ----------------------------END OF NESTED BLOCK-------------
47        ----WHEN OTHERS THEN
48        ----DBMS_OUTPUT.PUT_LINE(SQLCODE||'   '||SQLERRM);
49      END;
50  ------------------------PROCEDURE 2---------------------
51  PROCEDURE SHOW_REC(V_EMPNO EMP.EMPNO%TYPE DEFAULT 7788) IS
52  EMP_REC EMP%ROWTYPE;
53  BEGIN
54  SELECT * INTO EMP_REC FROM SCOTT.EMP
55  WHERE EMPNO=V_EMPNO;
56  DBMS_OUTPUT.PUT_LINE(EMP_REC.ENAME||'  '||EMP_REC.JOB||'  '||EMP_REC.SAL||'
'||EMP_REC.DEPTNO);
57  EXCEPTION
58  WHEN NO_DATA_FOUND THEN
59  DBMS_OUTPUT.PUT_LINE('RECORD NOT FOUND...');
60  END;
61  ----------------------------FUNCTION 1----------------------
62  FUNCTION GET_TAX(V_SAL NUMBER)
63  RETURN NUMBER IS
64  V_TAX NUMBER :=0;
65  ANN_SAL NUMBER :=0;
66  BEGIN
67  ANN_SAL := V_SAL * 12;
68  IF ANN_SAL BETWEEN 15000 AND 20000 THEN
69  V_TAX := V_SAL * 5/100;
70  ELSIF ANN_SAL BETWEEN 20001 AND 30000 THEN
71  V_TAX := V_SAL * 7/100;
72  ELSIF ANN_SAL BETWEEN 30001 AND 40000 THEN
73  V_TAX := V_SAL * 9/100;
74  ELSIF ANN_SAL >40000 THEN
75  V_TAX := V_SAL * 10/100;
76  END IF;
77  RETURN(V_TAX);
78  END;
79  --------------------------------FUNCTION 2--------------------
80  FUNCTION VALID_SAL(V_SAL NUMBER) RETURN BOOLEAN
81  IS
82  V_GRADE NUMBER :=0;
83  BEGIN
84  SELECT GRADE INTO V_GRADE FROM SCOTT.SALGRADE
85  WHERE V_SAL BETWEEN LOSAL AND HISAL;
86  RETURN(TRUE);
87  EXCEPTION
88  WHEN NO_DATA_FOUND THEN
89  RETURN(FALSE);
90  END;
91  ----------------------------END OF PACKAGE BODY------------
92* END;
```

```
 93  /

Package body created.

SQL>
SQL>
SQL>
SQL>
SQL> SELECT TEXT FROM USER_SOURCE WHERE NAME='MY_PACK';

TEXT
--------------------------------------------------------------------------------
----------------
PACKAGE MY_PACK IS

----------------------PROCEDURE 1 ADD_EMP-------------

PROCEDURE ADD_EMP
   (
       V_EMPNO EMP.EMPNO%TYPE,
       V_ENAME EMP.ENAME%TYPE,
       V_JOB   EMP.JOB%TYPE,
       V_SAL   EMP.SAL%TYPE,
       V_DEPTNO EMP.DEPTNO%TYPE,
       V_MGR    EMP.MGR%TYPE
   ) ;
--------------------------PROCEDURE 2 SHOW_REC--------------
PROCEDURE SHOW_REC(V_EMPNO EMP.EMPNO%TYPE DEFAULT 7788);
------------------------------FUNCTION 1 GET_TAX---------
FUNCTION GET_TAX(V_SAL NUMBER) RETURN NUMBER;
---------------------------FUNCTION 2 VALID_SAL------------
FUNCTION VALID_SAL(V_SAL NUMBER) RETURN BOOLEAN;
END;
PACKAGE BODY MY_PACK IS
---------------------STARTING PACKAGE BODY------------
--------------------PROCEDURE 1 ------------------
PROCEDURE ADD_EMP
   (
       V_EMPNO EMP.EMPNO%TYPE,
       V_ENAME EMP.ENAME%TYPE,
```

```
    V_JOB   EMP.JOB%TYPE,
    V_SAL   EMP.SAL%TYPE,
    V_DEPTNO EMP.DEPTNO%TYPE,
    V_MGR    EMP.MGR%TYPE
) IS
VALID_SALARY BOOLEAN;
VALID_MGR BOOLEAN;
----------------------------PVT PROCEDURE--------------------
PROCEDURE NEW_ID(ID OUT NUMBER) IS
BEGIN
SELECT MAX(EMPNO)+1 INTO ID FROM SCOTT.EMP;
END;
----------------------------END OF PVT PROCEDURE-------------------
   BEGIN
VALID_SALARY := VALID_SAL(V_SAL);  ----CALLING FUNCTION
VALID_MGR  := GET_MGR(V_MGR);   -----CALLING FUNCTION
IF VALID_SALARY=FALSE OR VALID_MGR=FALSE THEN
RAISE_APPLICATION_ERROR(-20100,'INVALID SALARY OR INVALID MANAGER....' );
END IF;
   INSERT INTO EMP(EMPNO,ENAME,JOB,SAL,DEPTNO,MGR)
    VALUES(V_EMPNO,V_ENAME,V_JOB,V_SAL,V_DEPTNO,V_MGR);
   COMMIT;
   DBMS_OUTPUT.PUT_LINE('RECORD CREATED WITH EMPNO '||V_EMPNO);
   EXCEPTION
   WHEN DUP_VAL_ON_INDEX  THEN
   ------------------------NESTED  BLOCK-----------------
   DECLARE
   ID NUMBER :=0;
   BEGIN
----   SELECT MAX(EMPNO)+1 INTO ID FROM SCOTT.EMP;
-----   ID := GET_ID('EMP') ;-------------CALLING PUBLIC FUNCTION
```

```
   ----    NEW_ID(ID);   --------CALLING PUBLIC PROCEDURE

       NEW_ID(ID);   --------CALLING PVT PROCEDURE

   INSERT INTO EMP(EMPNO,ENAME,JOB,SAL,DEPTNO,MGR)

      VALUES(ID,V_ENAME,V_JOB,V_SAL,V_DEPTNO,V_MGR);

      COMMIT;

      DBMS_OUTPUT.PUT_LINE('RECORD CREATED WITH EMPNO '||ID ||' INSTEAD OF
'||V_EMPNO);
      END;

      ----------------------------END OF NESTED BLOCK-------------

      ----WHEN OTHERS THEN

      ----DBMS_OUTPUT.PUT_LINE(SQLCODE||'   '||SQLERRM);

    END;

------------------------PROCEDURE 2---------------------

PROCEDURE SHOW_REC(V_EMPNO EMP.EMPNO%TYPE DEFAULT 7788) IS

EMP_REC EMP%ROWTYPE;

BEGIN

SELECT * INTO EMP_REC FROM SCOTT.EMP

WHERE EMPNO=V_EMPNO;

DBMS_OUTPUT.PUT_LINE(EMP_REC.ENAME||'  '||EMP_REC.JOB||'  '||EMP_REC.SAL||'
'||EMP_REC.DEPTNO);
EXCEPTION

WHEN NO_DATA_FOUND THEN

DBMS_OUTPUT.PUT_LINE('RECORD NOT FOUND...');

END;

----------------------------FUNCTION 1----------------------

FUNCTION GET_TAX(V_SAL NUMBER)

RETURN NUMBER IS

V_TAX NUMBER :=0;

ANN_SAL NUMBER :=0;

BEGIN

ANN_SAL := V_SAL * 12;

IF ANN_SAL BETWEEN 15000 AND 20000 THEN

V_TAX := V_SAL * 5/100;

ELSIF ANN_SAL BETWEEN 20001 AND 30000 THEN
```

```
V_TAX := V_SAL * 7/100;

ELSIF ANN_SAL BETWEEN 30001 AND 40000 THEN

V_TAX := V_SAL * 9/100;

ELSIF ANN_SAL >40000 THEN

V_TAX := V_SAL * 10/100;

END IF;

RETURN(V_TAX);

END;

---------------------------------FUNCTION 2--------------------


TEXT

--------------------------------------------------------------------------------
----------------
FUNCTION VALID_SAL(V_SAL NUMBER) RETURN BOOLEAN

IS

V_GRADE NUMBER :=0;

BEGIN

SELECT GRADE INTO V_GRADE FROM SCOTT.SALGRADE

WHERE V_SAL BETWEEN LOSAL AND HISAL;

RETURN(TRUE);

EXCEPTION

WHEN NO_DATA_FOUND THEN

RETURN(FALSE);

END;

------------------------------END OF PACKAGE BODY------------

END;


110 rows selected.

SQL> DESC MY_PACK
PROCEDURE ADD_EMP
 Argument Name                  Type                    In/Out Default?
 ------------------------------ ----------------------- ------ --------
 V_EMPNO                        NUMBER(4)               IN
 V_ENAME                        VARCHAR2(10)            IN
 V_JOB                          VARCHAR2(9)             IN
 V_SAL                          NUMBER(7,2)             IN
 V_DEPTNO                       NUMBER(2)               IN
 V_MGR                          NUMBER(5)               IN
```

```
FUNCTION GET_TAX RETURNS NUMBER
 Argument Name                  Type                      In/Out Default?
 ------------------------------ ------------------------- ------ --------
 V_SAL                          NUMBER                    IN
PROCEDURE SHOW_REC
 Argument Name                  Type                      In/Out Default?
 ------------------------------ ------------------------- ------ --------
 V_EMPNO                        NUMBER(4)                 IN     DEFAULT
FUNCTION VALID_SAL RETURNS BOOLEAN
 Argument Name                  Type                      In/Out Default?
 ------------------------------ ------------------------- ------ --------
 V_SAL                          NUMBER                    IN

SQL>
SQL>
SQL>
SQL> SELECT SAL,GET_TAX(SAL) FROM EMP;

       SAL GET_TAX(SAL)

---------- ------------

       900            0

      1600           80

      1250         62.5

      2975       267.75

      1250         62.5

      2850        256.5

      2450        171.5

      3000          270

      5000          500

      1500           75

      1100            0

      1000            0

     45666       4566.6

      1300           65

      1000            0

         1            0

      1000            0

      1000            0

      1000            0

      1000            0
```

```
20 rows selected.

SQL> ED
Wrote file afiedt.buf

  1* SELECT SAL,MY_PACK.GET_TAX(SAL) FROM EMP
SQL> /

       SAL MY_PACK.GET_TAX(SAL)
---------- --------------------
       900                    0
      1600                   80
      1250                 62.5
      2975               267.75
      1250                 62.5
      2850                256.5
      2450                171.5
      3000                  270
      5000                  500
      1500                   75
      1100                    0
      1000                    0
     45666               4566.6
      1300                   65
      1000                    0
         1                    0
      1000                    0
      1000                    0
      1000                    0
      1000                    0

20 rows selected.

SQL> ED
Wrote file afiedt.buf

  1  CREATE OR REPLACE PACKAGE BODY MY_PACK IS
  2  ---------------------STARTING PACKAGE BODY------------
  3  FUNCTION GET_ID RETURN NUMBER IS
  4  EMPID NUMBER :=0;
  5  BEGIN
```

```
 6  SELECT MAX(EMPNO)+1 INTO EMPID FROM SCOTT.EMP;
 7  RETURN(EMPID);
 8  END;
 9  ---------------------PROCEDURE 1 -------------------
10  PROCEDURE ADD_EMP
11     (
12         V_EMPNO EMP.EMPNO%TYPE,
13         V_ENAME EMP.ENAME%TYPE,
14         V_JOB   EMP.JOB%TYPE,
15         V_SAL   EMP.SAL%TYPE,
16         V_DEPTNO EMP.DEPTNO%TYPE,
17         V_MGR    EMP.MGR%TYPE
18     ) IS
19     VALID_SALARY BOOLEAN;
20     VALID_MGR BOOLEAN;
21     ----------------------------PVT PROCEDURE-------------------
22     PROCEDURE NEW_ID(ID OUT NUMBER) IS
23     BEGIN
24     SELECT MAX(EMPNO)+1 INTO ID FROM SCOTT.EMP;
25     END;
26     ----------------------------END OF PVT PROCEDURE-------------------
27        BEGIN
28     VALID_SALARY := VALID_SAL(V_SAL);  ----CALLING FUNCTION
29     VALID_MGR  := GET_MGR(V_MGR);   -----CALLING FUNCTION
30     IF VALID_SALARY=FALSE OR VALID_MGR=FALSE THEN
31     RAISE_APPLICATION_ERROR(-20100,'INVALID SALARY OR INVALID MANAGER....' );
32     END IF;
33        INSERT INTO EMP(EMPNO,ENAME,JOB,SAL,DEPTNO,MGR)
34         VALUES(V_EMPNO,V_ENAME,V_JOB,V_SAL,V_DEPTNO,V_MGR);
35        COMMIT;
36        DBMS_OUTPUT.PUT_LINE('RECORD CREATED WITH EMPNO '||V_EMPNO);
37        EXCEPTION
38        WHEN DUP_VAL_ON_INDEX  THEN
39        ------------------------NESTED  BLOCK-----------------
40        DECLARE
41        ID NUMBER :=0;
42        BEGIN
43     ----   SELECT MAX(EMPNO)+1 INTO ID FROM SCOTT.EMP;
44     -----   ID := GET_ID('EMP') ;-------------CALLING PUBLIC FUNCTION
45     ----    NEW_ID(ID);   --------CALLING PUBLIC PROCEDURE
46  ---       NEW_ID(ID);   --------CALLING PVT PROCEDURE
47      ID:= GET_ID; -----NEW PVT FUNCTION
48     INSERT INTO EMP(EMPNO,ENAME,JOB,SAL,DEPTNO,MGR)
49        VALUES(ID,V_ENAME,V_JOB,V_SAL,V_DEPTNO,V_MGR);
50        COMMIT;
51        DBMS_OUTPUT.PUT_LINE('RECORD CREATED WITH EMPNO '||ID ||' INSTEAD OF
'||V_EMPNO);
52        END;
53        ----------------------------END OF NESTED BLOCK-------------
54        ----WHEN OTHERS THEN
55        ----DBMS_OUTPUT.PUT_LINE(SQLCODE||'   '||SQLERRM);
56      END;
57  ------------------------PROCEDURE 2---------------------
58  PROCEDURE SHOW_REC(V_EMPNO EMP.EMPNO%TYPE DEFAULT 7788) IS
59  EMP_REC EMP%ROWTYPE;
60  BEGIN
61  SELECT * INTO EMP_REC FROM SCOTT.EMP
62  WHERE EMPNO=V_EMPNO;
63  DBMS_OUTPUT.PUT_LINE(EMP_REC.ENAME||'  '||EMP_REC.JOB||'  '||EMP_REC.SAL||'
'||EMP_REC.DEPTNO);
64  EXCEPTION
65  WHEN NO_DATA_FOUND THEN
66  DBMS_OUTPUT.PUT_LINE('RECORD NOT FOUND...');
```

```
 67  END;
 68  ----------------------------FUNCTION 1----------------------
 69  FUNCTION GET_TAX(V_SAL NUMBER)
 70  RETURN NUMBER IS
 71  V_TAX NUMBER :=0;
 72  ANN_SAL NUMBER :=0;
 73  BEGIN
 74  ANN_SAL := V_SAL * 12;
 75  IF ANN_SAL BETWEEN 15000 AND 20000 THEN
 76  V_TAX := V_SAL * 5/100;
 77  ELSIF ANN_SAL BETWEEN 20001 AND 30000 THEN
 78  V_TAX := V_SAL * 7/100;
 79  ELSIF ANN_SAL BETWEEN 30001 AND 40000 THEN
 80  V_TAX := V_SAL * 9/100;
 81  ELSIF ANN_SAL >40000 THEN
 82  V_TAX := V_SAL * 10/100;
 83  END IF;
 84  RETURN(V_TAX);
 85  END;
 86  --------------------------------FUNCTION 2---------------------
 87  FUNCTION VALID_SAL(V_SAL NUMBER) RETURN BOOLEAN
 88  IS
 89  V_GRADE NUMBER :=0;
 90  BEGIN
 91  SELECT GRADE INTO V_GRADE FROM SCOTT.SALGRADE
 92  WHERE V_SAL BETWEEN LOSAL AND HISAL;
 93  RETURN(TRUE);
 94  EXCEPTION
 95  WHEN NO_DATA_FOUND THEN
 96  RETURN(FALSE);
 97  END;
 98  ------------------------------END OF PACKAGE BODY------------
 99* END;
100  /

Package body created.

SQL>
SQL>
SQL>
SQL>
SQL>
SQL> DESC MY_PACK
PROCEDURE ADD_EMP
 Argument Name                  Type                    In/Out Default?
 ------------------------------ ----------------------- ------ --------
 V_EMPNO                        NUMBER(4)               IN
 V_ENAME                        VARCHAR2(10)            IN
 V_JOB                          VARCHAR2(9)             IN
 V_SAL                          NUMBER(7,2)             IN
 V_DEPTNO                       NUMBER(2)               IN
 V_MGR                          NUMBER(5)               IN
FUNCTION GET_TAX RETURNS NUMBER
 Argument Name                  Type                    In/Out Default?
 ------------------------------ ----------------------- ------ --------
 V_SAL                          NUMBER                  IN
PROCEDURE SHOW_REC
 Argument Name                  Type                    In/Out Default?
 ------------------------------ ----------------------- ------ --------
 V_EMPNO                        NUMBER(4)               IN     DEFAULT
FUNCTION VALID_SAL RETURNS BOOLEAN
 Argument Name                  Type                    In/Out Default?
 ------------------------------ ----------------------- ------ --------
```

```
 V_SAL                          NUMBER                  IN

SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL> EXEC MY_PACK.ADD_EMP(7788,USER,'SALESMAN',1000,30,7788);
RECORD CREATED WITH EMPNO 8001 INSTEAD OF 7788


PL/SQL procedure successfully completed.

SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL> ED
Wrote file afiedt.buf

  1  CREATE OR REPLACE PACKAGE BODY MY_PACK IS
  2  ---------------------STARTING PACKAGE BODY------------
  3  ---------------------PROCEDURE 1 -------------------
  4  PROCEDURE ADD_EMP
  5     (
  6         V_EMPNO EMP.EMPNO%TYPE,
  7         V_ENAME EMP.ENAME%TYPE,
  8         V_JOB   EMP.JOB%TYPE,
  9         V_SAL   EMP.SAL%TYPE,
 10         V_DEPTNO EMP.DEPTNO%TYPE,
 11         V_MGR    EMP.MGR%TYPE
 12     ) IS
 13     VALID_SALARY BOOLEAN;
 14     VALID_MGR BOOLEAN;
 15     ----------------------------PVT PROCEDURE--------------------
 16     PROCEDURE NEW_ID(ID OUT NUMBER) IS
 17     BEGIN
 18     SELECT MAX(EMPNO)+1 INTO ID FROM SCOTT.EMP;
 19     END;
 20     ----------------------------END OF PVT PROCEDURE-------------------
 21        BEGIN
 22     VALID_SALARY := VALID_SAL(V_SAL);  ----CALLING FUNCTION
 23     VALID_MGR  := GET_MGR(V_MGR);   -----CALLING FUNCTION
 24     IF VALID_SALARY=FALSE OR VALID_MGR=FALSE THEN
 25     RAISE_APPLICATION_ERROR(-20100,'INVALID SALARY OR INVALID MANAGER....' );
 26     END IF;
 27        INSERT INTO EMP(EMPNO,ENAME,JOB,SAL,DEPTNO,MGR)
 28         VALUES(V_EMPNO,V_ENAME,V_JOB,V_SAL,V_DEPTNO,V_MGR);
 29        COMMIT;
 30        DBMS_OUTPUT.PUT_LINE('RECORD CREATED WITH EMPNO '||V_EMPNO);
 31        EXCEPTION
 32        WHEN DUP_VAL_ON_INDEX  THEN
 33        ----------------------NESTED  BLOCK----------------
 34        DECLARE
```

```
 35        ID NUMBER :=0;
 36        BEGIN
 37     ----   SELECT MAX(EMPNO)+1 INTO ID FROM SCOTT.EMP;
 38     -----   ID := GET_ID('EMP') ;-------------CALLING PUBLIC FUNCTION
 39     ----    NEW_ID(ID);   --------CALLING PUBLIC PROCEDURE
 40  ---       NEW_ID(ID);   --------CALLING PVT PROCEDURE
 41      ID:= GET_ID; -----NEW PVT FUNCTION
 42     INSERT INTO EMP(EMPNO,ENAME,JOB,SAL,DEPTNO,MGR)
 43        VALUES(ID,V_ENAME,V_JOB,V_SAL,V_DEPTNO,V_MGR);
 44        COMMIT;
 45        DBMS_OUTPUT.PUT_LINE('RECORD CREATED WITH EMPNO '||ID ||' INSTEAD OF
'||V_EMPNO);
 46        END;
 47        ------------------------------END OF NESTED BLOCK-------------
 48        ----WHEN OTHERS THEN
 49        ----DBMS_OUTPUT.PUT_LINE(SQLCODE||'   '||SQLERRM);
 50      END;
 51  ------------------------PROCEDURE 2---------------------
 52  PROCEDURE SHOW_REC(V_EMPNO EMP.EMPNO%TYPE DEFAULT 7788) IS
 53  EMP_REC EMP%ROWTYPE;
 54  BEGIN
 55  SELECT * INTO EMP_REC FROM SCOTT.EMP
 56  WHERE EMPNO=V_EMPNO;
 57  DBMS_OUTPUT.PUT_LINE(EMP_REC.ENAME||'  '||EMP_REC.JOB||'  '||EMP_REC.SAL||'
'||EMP_REC.DEPTNO);
 58  EXCEPTION
 59  WHEN NO_DATA_FOUND THEN
 60  DBMS_OUTPUT.PUT_LINE('RECORD NOT FOUND...');
 61  END;
 62  ------------------------------FUNCTION 1-----------------------
 63  FUNCTION GET_TAX(V_SAL NUMBER)
 64  RETURN NUMBER IS
 65  V_TAX NUMBER :=0;
 66  ANN_SAL NUMBER :=0;
 67  BEGIN
 68  ANN_SAL := V_SAL * 12;
 69  IF ANN_SAL BETWEEN 15000 AND 20000 THEN
 70  V_TAX := V_SAL * 5/100;
 71  ELSIF ANN_SAL BETWEEN 20001 AND 30000 THEN
 72  V_TAX := V_SAL * 7/100;
 73  ELSIF ANN_SAL BETWEEN 30001 AND 40000 THEN
 74  V_TAX := V_SAL * 9/100;
 75  ELSIF ANN_SAL >40000 THEN
 76  V_TAX := V_SAL * 10/100;
 77  END IF;
 78  RETURN(V_TAX);
 79  END;
 80  ---------------------------------FUNCTION 2---------------------
 81  FUNCTION VALID_SAL(V_SAL NUMBER) RETURN BOOLEAN
 82  IS
 83  V_GRADE NUMBER :=0;
 84  BEGIN
 85  SELECT GRADE INTO V_GRADE FROM SCOTT.SALGRADE
 86  WHERE V_SAL BETWEEN LOSAL AND HISAL;
 87  RETURN(TRUE);
 88  EXCEPTION
 89  WHEN NO_DATA_FOUND THEN
 90  RETURN(FALSE);
 91  END;
 92  ---------------PVT FUNCTION --------------
 93  FUNCTION GET_ID RETURN NUMBER IS
 94  EMPID NUMBER :=0;
 95  BEGIN
```

```
 96  SELECT MAX(EMPNO)+1 INTO EMPID FROM SCOTT.EMP;
 97  RETURN(EMPID);
 98  END;
 99  ------------------------------END OF PACKAGE BODY------------
100* END;
101  /

Warning: Package Body created with compilation errors.

SQL> SHOW ERR
Errors for PACKAGE BODY MY_PACK:

LINE/COL ERROR

-------- -----------------------------------------------------------------

41/5     PL/SQL: Statement ignored

41/10    PLS-00313: 'GET_ID' not declared in this scope

SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL> ED
Wrote file afiedt.buf

  1  CREATE OR REPLACE PACKAGE BODY MY_PACK IS
  2  ----STARTING PACKAGE BODY-(EXAMPLE OF FORWARD AND WITHOUT FORWARD
DECLARATION-----------
  3  ----------------PVT FUNCTION --------------
  4  FUNCTION GET_ID RETURN NUMBER IS
  5  EMPID NUMBER :=0;
  6  BEGIN
  7  SELECT MAX(EMPNO)+1 INTO EMPID FROM SCOTT.EMP;
  8  RETURN(EMPID);
  9  END;
 10  ---------------------PROCEDURE 1 -------------------
 11  PROCEDURE ADD_EMP
 12     (
 13         V_EMPNO EMP.EMPNO%TYPE,
 14         V_ENAME EMP.ENAME%TYPE,
 15         V_JOB   EMP.JOB%TYPE,
 16         V_SAL   EMP.SAL%TYPE,
 17         V_DEPTNO EMP.DEPTNO%TYPE,
 18         V_MGR    EMP.MGR%TYPE
 19     ) IS
 20     VALID_SALARY BOOLEAN;
 21     VALID_MGR BOOLEAN;
 22     ----------------------------PVT PROCEDURE--------------------
 23     PROCEDURE NEW_ID(ID OUT NUMBER) IS
 24     BEGIN
 25     SELECT MAX(EMPNO)+1 INTO ID FROM SCOTT.EMP;
 26     END;
 27     ------------------------------END OF PVT PROCEDURE-------------------
 28        BEGIN
 29     VALID_SALARY := VALID_SAL(V_SAL);  ----CALLING FUNCTION
```

```
 30     VALID_MGR  := GET_MGR(V_MGR);   -----CALLING FUNCTION
 31     IF VALID_SALARY=FALSE OR VALID_MGR=FALSE THEN
 32     RAISE_APPLICATION_ERROR(-20100,'INVALID SALARY OR INVALID MANAGER....' );
 33     END IF;
 34        INSERT INTO EMP(EMPNO,ENAME,JOB,SAL,DEPTNO,MGR)
 35         VALUES(V_EMPNO,V_ENAME,V_JOB,V_SAL,V_DEPTNO,V_MGR);
 36        COMMIT;
 37        DBMS_OUTPUT.PUT_LINE('RECORD CREATED WITH EMPNO '||V_EMPNO);
 38        EXCEPTION
 39        WHEN DUP_VAL_ON_INDEX  THEN
 40        ------------------------NESTED  BLOCK-----------------
 41        DECLARE
 42        ID NUMBER :=0;
 43        BEGIN
 44     ----   SELECT MAX(EMPNO)+1 INTO ID FROM SCOTT.EMP;
 45     -----   ID := GET_ID('EMP') ;-------------CALLING PUBLIC FUNCTION
 46     ----    NEW_ID(ID);   --------CALLING PUBLIC PROCEDURE
 47  ---        NEW_ID(ID);   --------CALLING PVT PROCEDURE
 48      ID:= GET_ID; -----NEW PVT FUNCTION
 49     INSERT INTO EMP(EMPNO,ENAME,JOB,SAL,DEPTNO,MGR)
 50        VALUES(ID,V_ENAME,V_JOB,V_SAL,V_DEPTNO,V_MGR);
 51        COMMIT;
 52        DBMS_OUTPUT.PUT_LINE('RECORD CREATED WITH EMPNO '||ID ||' INSTEAD OF
'||V_EMPNO);
 53        END;
 54        ----------------------------END OF NESTED BLOCK-------------
 55        ----WHEN OTHERS THEN
 56        ----DBMS_OUTPUT.PUT_LINE(SQLCODE||'   '||SQLERRM);
 57      END;
 58  ------------------------PROCEDURE 2---------------------
 59  PROCEDURE SHOW_REC(V_EMPNO EMP.EMPNO%TYPE DEFAULT 7788) IS
 60  EMP_REC EMP%ROWTYPE;
 61  BEGIN
 62  SELECT * INTO EMP_REC FROM SCOTT.EMP
 63  WHERE EMPNO=V_EMPNO;
 64  DBMS_OUTPUT.PUT_LINE(EMP_REC.ENAME||'  '||EMP_REC.JOB||'  '||EMP_REC.SAL||'
'||EMP_REC.DEPTNO);
 65  EXCEPTION
 66  WHEN NO_DATA_FOUND THEN
 67  DBMS_OUTPUT.PUT_LINE('RECORD NOT FOUND...');
 68  END;
 69  ----------------------------FUNCTION 1----------------------
 70  FUNCTION GET_TAX(V_SAL NUMBER)
 71  RETURN NUMBER IS
 72  V_TAX NUMBER :=0;
 73  ANN_SAL NUMBER :=0;
 74  BEGIN
 75  ANN_SAL := V_SAL * 12;
 76  IF ANN_SAL BETWEEN 15000 AND 20000 THEN
 77  V_TAX := V_SAL * 5/100;
 78  ELSIF ANN_SAL BETWEEN 20001 AND 30000 THEN
 79  V_TAX := V_SAL * 7/100;
 80  ELSIF ANN_SAL BETWEEN 30001 AND 40000 THEN
 81  V_TAX := V_SAL * 9/100;
 82  ELSIF ANN_SAL >40000 THEN
 83  V_TAX := V_SAL * 10/100;
 84  END IF;
 85  RETURN(V_TAX);
 86  END;
 87  --------------------------------FUNCTION 2--------------------
 88  FUNCTION VALID_SAL(V_SAL NUMBER) RETURN BOOLEAN
 89  IS
 90  V_GRADE NUMBER :=0;
```

```
 91  BEGIN
 92  SELECT GRADE INTO V_GRADE FROM SCOTT.SALGRADE
 93  WHERE V_SAL BETWEEN LOSAL AND HISAL;
 94  RETURN(TRUE);
 95  EXCEPTION
 96  WHEN NO_DATA_FOUND THEN
 97  RETURN(FALSE);
 98  END;
 99  ------------------------------END OF PACKAGE BODY------------
100* END;
101  /

Package body created.

SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL> ED
Wrote file afiedt.buf

  1  CREATE OR REPLACE PACKAGE BODY MY_PACK IS
  2  ----STARTING PACKAGE BODY-(EXAMPLE OF FORWARD AND WITHOUT FORWARD
DECLARATION-----------
  3  FUNCTION GET_ID RETURN NUMBER;
  4  ---------------------PROCEDURE 1 -------------------
  5  PROCEDURE ADD_EMP
  6     (
  7         V_EMPNO EMP.EMPNO%TYPE,
  8         V_ENAME EMP.ENAME%TYPE,
  9         V_JOB   EMP.JOB%TYPE,
 10         V_SAL   EMP.SAL%TYPE,
 11         V_DEPTNO EMP.DEPTNO%TYPE,
 12         V_MGR    EMP.MGR%TYPE
 13     ) IS
 14     VALID_SALARY BOOLEAN;
 15     VALID_MGR BOOLEAN;
 16     ----------------------------PVT PROCEDURE--------------------
 17     PROCEDURE NEW_ID(ID OUT NUMBER) IS
 18     BEGIN
 19     SELECT MAX(EMPNO)+1 INTO ID FROM SCOTT.EMP;
 20     END;
 21     ----------------------------END OF PVT PROCEDURE-------------------
 22        BEGIN
 23     VALID_SALARY := VALID_SAL(V_SAL);  ----CALLING FUNCTION
 24     VALID_MGR  := GET_MGR(V_MGR);   -----CALLING FUNCTION
 25     IF VALID_SALARY=FALSE OR VALID_MGR=FALSE THEN
 26     RAISE_APPLICATION_ERROR(-20100,'INVALID SALARY OR INVALID MANAGER....' );
 27     END IF;
 28        INSERT INTO EMP(EMPNO,ENAME,JOB,SAL,DEPTNO,MGR)
 29         VALUES(V_EMPNO,V_ENAME,V_JOB,V_SAL,V_DEPTNO,V_MGR);
 30        COMMIT;
 31        DBMS_OUTPUT.PUT_LINE('RECORD CREATED WITH EMPNO '||V_EMPNO);
 32        EXCEPTION
 33        WHEN DUP_VAL_ON_INDEX  THEN
 34        ------------------------NESTED  BLOCK-----------------
 35        DECLARE
 36        ID NUMBER :=0;
 37        BEGIN
 38     ----   SELECT MAX(EMPNO)+1 INTO ID FROM SCOTT.EMP;
 39     -----   ID := GET_ID('EMP') ;-------------CALLING PUBLIC FUNCTION
```

```
 40     ----    NEW_ID(ID);   --------CALLING PUBLIC PROCEDURE
 41  ---       NEW_ID(ID);   --------CALLING PVT PROCEDURE
 42      ID:= GET_ID; -----NEW PVT FUNCTION
 43     INSERT INTO EMP(EMPNO,ENAME,JOB,SAL,DEPTNO,MGR)
 44        VALUES(ID,V_ENAME,V_JOB,V_SAL,V_DEPTNO,V_MGR);
 45        COMMIT;
 46        DBMS_OUTPUT.PUT_LINE('RECORD CREATED WITH EMPNO '||ID ||' INSTEAD OF
'||V_EMPNO);
 47        END;
 48        ------------------------------END OF NESTED BLOCK-------------
 49        ----WHEN OTHERS THEN
 50        ----DBMS_OUTPUT.PUT_LINE(SQLCODE||'   '||SQLERRM);
 51      END;
 52  ------------------------PROCEDURE 2---------------------
 53  PROCEDURE SHOW_REC(V_EMPNO EMP.EMPNO%TYPE DEFAULT 7788) IS
 54  EMP_REC EMP%ROWTYPE;
 55  BEGIN
 56  SELECT * INTO EMP_REC FROM SCOTT.EMP
 57  WHERE EMPNO=V_EMPNO;
 58  DBMS_OUTPUT.PUT_LINE(EMP_REC.ENAME||'  '||EMP_REC.JOB||'  '||EMP_REC.SAL||'
'||EMP_REC.DEPTNO);
 59  EXCEPTION
 60  WHEN NO_DATA_FOUND THEN
 61  DBMS_OUTPUT.PUT_LINE('RECORD NOT FOUND...');
 62  END;
 63  ------------------------------FUNCTION 1----------------------
 64  FUNCTION GET_TAX(V_SAL NUMBER)
 65  RETURN NUMBER IS
 66  V_TAX NUMBER :=0;
 67  ANN_SAL NUMBER :=0;
 68  BEGIN
 69  ANN_SAL := V_SAL * 12;
 70  IF ANN_SAL BETWEEN 15000 AND 20000 THEN
 71  V_TAX := V_SAL * 5/100;
 72  ELSIF ANN_SAL BETWEEN 20001 AND 30000 THEN
 73  V_TAX := V_SAL * 7/100;
 74  ELSIF ANN_SAL BETWEEN 30001 AND 40000 THEN
 75  V_TAX := V_SAL * 9/100;
 76  ELSIF ANN_SAL >40000 THEN
 77  V_TAX := V_SAL * 10/100;
 78  END IF;
 79  RETURN(V_TAX);
 80  END;
 81  --------------------------------FUNCTION 2---------------------
 82  FUNCTION VALID_SAL(V_SAL NUMBER) RETURN BOOLEAN
 83  IS
 84  V_GRADE NUMBER :=0;
 85  BEGIN
 86  SELECT GRADE INTO V_GRADE FROM SCOTT.SALGRADE
 87  WHERE V_SAL BETWEEN LOSAL AND HISAL;
 88  RETURN(TRUE);
 89  EXCEPTION
 90  WHEN NO_DATA_FOUND THEN
 91  RETURN(FALSE);
 92  END;
 93  ---------------PVT FUNCTION --------------
 94  FUNCTION GET_ID RETURN NUMBER IS
 95  EMPID NUMBER :=0;
 96  BEGIN
 97  SELECT MAX(EMPNO)+1 INTO EMPID FROM SCOTT.EMP;
 98  RETURN(EMPID);
 99  END;
100  ------------------------------END OF PACKAGE BODY------------
```

```
101* END;
102  /

Package body created.

SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL> CREATE OR REPLACE PROCEDURE GET_TAX IS
  2  BEGIN
  3  NULL;
  4  END;
  5  /
CREATE OR REPLACE PROCEDURE GET_TAX IS
*
ERROR at line 1:
ORA-00955: name is already used by an existing object


SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL> ED
Wrote file afiedt.buf

  1    CREATE OR REPLACE PACKAGE OVERPACK IS
  2      ------------------PROCEDURE-----------------
  3      PROCEDURE ADD_REC(PLOC DEPT.LOC%TYPE,PDNAME DEPT.DNAME%TYPE,
  4                        PDEPTNO DEPT.DEPTNO%TYPE);
  5      PROCEDURE ADD_REC(PEMPNO EMP.EMPNO%TYPE,PENAME EMP.ENAME%TYPE,
  6                      PJOB EMP.JOB%TYPE,PDEPTNO EMP.DEPTNO%TYPE);
  7      NN VARCHAR2(200);      ------------PUBLIC VARIABLE
  8              ----------------FUNCTIONS----------------
  9       FUNCTION TO_CHAR(P1 DATE) RETURN VARCHAR2;
 10       FUNCTION TO_CHAR(P1 NUMBER) RETURN VARCHAR2;
 11       FUNCTION TO_CHAR(P1 DATE,FORMAT VARCHAR2) RETURN VARCHAR2;
 12      FUNCTION TO_CHAR(P1 NUMBER,FORMAT VARCHAR2)RETURN VARCHAR2;
 13*  END OVERPACK;
 14  /

Package created.

SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL> ED
Wrote file afiedt.buf
```

```
SP2-0223: No lines in SQL buffer.
SQL> ED
SP2-0107: Nothing to save.
SQL> DECLARE
  2  .
SQL> ED
Wrote file afiedt.buf

  1      CREATE OR REPLACE PACKAGE BODY MY_NEW_PACK IS
  2              MAX_ID NUMBER:=0;               -----PVT VARIABLE
  3              ------------------PVT PROCEDURE FUNCTION
  4              PROCEDURE RETU_MSG IS                ------PVT PROCEDURE
  5              BEGIN
  6              COMMIT;
  7              &D('RECORD CREATED SUCCESS FULLY.....');
  8              END;
  9              ------------------------------------PVT FUNCTION
 10           FUNCTION GET_UK_ID(TAB_NAME IN CHAR) RETURN NUMBER IS
 11            BEGIN
 12            IF TAB_NAME='D' THEN
 13            SELECT MAX(DEPTNO)+10 INTO MAX_ID FROM DEPT;
 14            ELSIF TAB_NAME='E' THEN
 15            SELECT MAX(EMPNO)+1 INTO MAX_ID FROM EMP;
 16            END IF;
 17            RETURN(MAX_ID);
 18           END GET_UK_ID;
 19    ---------------------FIRST PROCEDURE-----------------------------------
 20      PROCEDURE ADD_REC(PLOC DEPT.LOC%TYPE,PDNAME DEPT.DNAME%TYPE,PDEPTNO
DEPT.DEPTNO%TYPE) IS
 21           BEGIN
 22          IF PDEPTNO IS NULL THEN
 23           MAX_ID:=GET_UK_ID('D');       --------------CALLING PVT FUNCTION
 24           END IF;
 25            INSERT INTO DEPT
 26            VALUES(NVL(PDEPTNO,MAX_ID),PDNAME,PLOC);
 27             RETU_MSG;               ------CALLING PVT PROCEDURE
 28            EXCEPTION
 29            WHEN OTHERS THEN
 30            RAISE_APPLICATION_ERROR(-20901,SQLERRM);
 31            END ADD_REC;
 32    ------------------------------SECOND PCEDURE-------------------
 33            PROCEDURE ADD_REC(PEMPNO EMP.EMPNO%TYPE,PENAME EMP.ENAME%TYPE,
 34           PJOB EMP.JOB%TYPE,PDEPTNO EMP.DEPTNO%TYPE)IS
 35        PSAL EMP.SAL%TYPE :=1000;
 36          ECOMM NUMBER :=0;
 37           PDEPT NUMBER:=PDEPTNO;
 38            BEGIN
 39            IF PJOB='SALESMAN' THEN
 40               ECOMM:=TRUNC((PSAL*30)/100,0) ;
 41           PDEPT:= 30;
 42            ELSE
 43            ECOMM:=NULL;
 44           END IF;
 45            IF PEMPNO IS NULL THEN
 46            MAX_ID:=GET_UK_ID('E');   ------CALLING PVT FUNCTION-----
 47            ELSE
 48           MAX_ID:=PEMPNO;
 49            END IF;
 50            INSERT INTO EMP(EMPNO,ENAME,JOB,DEPTNO,SAL,COMM)
 51            VALUES(MAX_ID,PENAME,PJOB,PDEPTNO,PSAL,ECOMM);
 52            RETU_MSG;   ----CALLING PVT PROCEDURE
 53            EXCEPTION
```

```
 54           WHEN DUP_VAL_ON_INDEX THEN
 55           MAX_ID:=GET_UK_ID('E');      ----CALLING PVT FUNCTION-----------
 56           INSERT INTO EMP(EMPNO,ENAME,JOB,DEPTNO,SAL,COMM)
 57           VALUES(MAX_ID,PENAME,PJOB,PDEPTNO,PSAL,ECOMM);
 58           &D('RECORD CREATED WITH '||MAX_ID||'INSTEAD OF '||PEMPNO);
 59          RETU_MSG;
 60            END ADD_REC;
 61   -----------FUCTION 1
 62           FUNCTION TO_CHAR(P1 DATE) RETURN VARCHAR2 IS
 63           BEGIN
 64           SELECT TO_CHAR(P1,'DD.MM.RRRR DDD') INTO NN FROM DUAL;
 65          RETURN(NN);
 66           END TO_CHAR;
 67           ---------------------------------FUNCTION 2
 68           FUNCTION TO_CHAR(P1 NUMBER)RETURN VARCHAR2 IS
 69           BEGIN
 70           SELECT TO_CHAR(TO_DATE(P1,'J'),'JSP') INTO NN FROM DUAL;
 71           RETURN(NN);
 72           END TO_CHAR;
 73           ------------------------------FUNCTION 3
 74            FUNCTION TO_CHAR(P1 DATE,FORMAT VARCHAR2)RETURN VARCHAR2 IS
 75           BEGIN
 76           SELECT TO_CHAR(P1,FORMAT) INTO NN FROM DUAL;
 77           RETURN(NN);
 78           END TO_CHAR;
 79           ------------------------------FUNCTION 4
 80          FUNCTION TO_CHAR(P1 NUMBER,FORMAT VARCHAR2) RETURN VARCHAR2 IS
 81          BEGIN
 82          SELECT TO_CHAR(P1,FORMAT) INTO NN FROM DUAL;
 83          RETURN(NN);
 84          END TO_CHAR;
 85*       END;
 86  /

Warning: Package Body created with compilation errors.

SQL>
SQL>
SQL>
SQL> SHOW ERR
Errors for PACKAGE BODY MY_NEW_PACK:

LINE/COL ERROR

-------- -----------------------------------------------------------------

0/0      PL/SQL: Compilation unit analysis terminated

1/14     PLS-00201: identifier 'MY_NEW_PACK' must be declared

1/14     PLS-00304: cannot compile body of 'MY_NEW_PACK' without its

         specification



SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL> ED
Wrote file afiedt.buf
```

```
  1      CREATE OR REPLACE PACKAGE BODY OVERPACK IS
  2              MAX_ID NUMBER:=0;               -----PVT VARIABLE
  3              ------------------PVT PROCEDURE FUNCTION
  4              PROCEDURE RETU_MSG IS                ------PVT PROCEDURE
  5              BEGIN
  6              COMMIT;
  7              &D('RECORD CREATED SUCCESS FULLY.....');
  8              END;
  9              -------------------------------------PVT FUNCTION
 10           FUNCTION GET_UK_ID(TAB_NAME IN CHAR) RETURN NUMBER IS
 11            BEGIN
 12            IF TAB_NAME='D' THEN
 13            SELECT MAX(DEPTNO)+10 INTO MAX_ID FROM DEPT;
 14            ELSIF TAB_NAME='E' THEN
 15            SELECT MAX(EMPNO)+1 INTO MAX_ID FROM EMP;
 16            END IF;
 17            RETURN(MAX_ID);
 18           END GET_UK_ID;
 19    ---------------------FIRST PROCEDURE-------------------------------------
 20      PROCEDURE ADD_REC(PLOC DEPT.LOC%TYPE,PDNAME DEPT.DNAME%TYPE,PDEPTNO
DEPT.DEPTNO%TYPE) IS
 21            BEGIN
 22           IF PDEPTNO IS NULL THEN
 23            MAX_ID:=GET_UK_ID('D');       --------------CALLING PVT FUNCTION
 24            END IF;
 25             INSERT INTO DEPT
 26             VALUES(NVL(PDEPTNO,MAX_ID),PDNAME,PLOC);
 27              RETU_MSG;              ------CALLING PVT PROCEDURE
 28             EXCEPTION
 29             WHEN OTHERS THEN
 30             RAISE_APPLICATION_ERROR(-20901,SQLERRM);
 31             END ADD_REC;
 32    -------------------------------SECOND PCEDURE-------------------
 33             PROCEDURE ADD_REC(PEMPNO EMP.EMPNO%TYPE,PENAME EMP.ENAME%TYPE,
 34           PJOB EMP.JOB%TYPE,PDEPTNO EMP.DEPTNO%TYPE)IS
 35        PSAL EMP.SAL%TYPE :=1000;
 36          ECOMM NUMBER :=0;
 37           PDEPT NUMBER:=PDEPTNO;
 38            BEGIN
 39            IF PJOB='SALESMAN' THEN
 40               ECOMM:=TRUNC((PSAL*30)/100,0) ;
 41           PDEPT:= 30;
 42            ELSE
 43            ECOMM:=NULL;
 44           END IF;
 45            IF PEMPNO IS NULL THEN
 46            MAX_ID:=GET_UK_ID('E');   ------CALLING PVT FUNCTION-----
 47            ELSE
 48           MAX_ID:=PEMPNO;
 49            END IF;
 50            INSERT INTO EMP(EMPNO,ENAME,JOB,DEPTNO,SAL,COMM)
 51            VALUES(MAX_ID,PENAME,PJOB,PDEPTNO,PSAL,ECOMM);
 52            RETU_MSG;   ----CALLING PVT PROCEDURE
 53            EXCEPTION
 54            WHEN DUP_VAL_ON_INDEX THEN
 55            MAX_ID:=GET_UK_ID('E');      ----CALLING PVT FUNCTION-----------
 56            INSERT INTO EMP(EMPNO,ENAME,JOB,DEPTNO,SAL,COMM)
 57            VALUES(MAX_ID,PENAME,PJOB,PDEPTNO,PSAL,ECOMM);
 58            &D('RECORD CREATED WITH '||MAX_ID||'INSTEAD OF '||PEMPNO);
 59           RETU_MSG;
 60             END ADD_REC;
 61    -----------FUCTION 1
```

```
 62           FUNCTION TO_CHAR(P1 DATE) RETURN VARCHAR2 IS
 63           BEGIN
 64           SELECT TO_CHAR(P1,'DD.MM.RRRR DDD') INTO NN FROM DUAL;
 65          RETURN(NN);
 66           END TO_CHAR;
 67           --------------------------------FUNCTION 2
 68           FUNCTION TO_CHAR(P1 NUMBER)RETURN VARCHAR2 IS
 69           BEGIN
 70           SELECT TO_CHAR(TO_DATE(P1,'J'),'JSP') INTO NN FROM DUAL;
 71           RETURN(NN);
 72           END TO_CHAR;
 73           ------------------------------FUNCTION 3
 74            FUNCTION TO_CHAR(P1 DATE,FORMAT VARCHAR2)RETURN VARCHAR2 IS
 75           BEGIN
 76           SELECT TO_CHAR(P1,FORMAT) INTO NN FROM DUAL;
 77           RETURN(NN);
 78           END TO_CHAR;
 79           ----------------------------FUNCTION 4
 80          FUNCTION TO_CHAR(P1 NUMBER,FORMAT VARCHAR2) RETURN VARCHAR2 IS
 81          BEGIN
 82          SELECT TO_CHAR(P1,FORMAT) INTO NN FROM DUAL;
 83          RETURN(NN);
 84          END TO_CHAR;
 85*       END;
SQL> /

Package body created.

SQL> DESC OVERPACK
PROCEDURE ADD_REC
 Argument Name                  Type                    In/Out Default?
 ------------------------------ ----------------------- ------ --------
 PLOC                           VARCHAR2(13)            IN
 PDNAME                         VARCHAR2(14)            IN
 PDEPTNO                        NUMBER(2)               IN
PROCEDURE ADD_REC
 Argument Name                  Type                    In/Out Default?
 ------------------------------ ----------------------- ------ --------
 PEMPNO                         NUMBER(4)               IN
 PENAME                         VARCHAR2(10)            IN
 PJOB                           VARCHAR2(9)             IN
 PDEPTNO                        NUMBER(2)               IN
FUNCTION TO_CHAR RETURNS VARCHAR2
 Argument Name                  Type                    In/Out Default?
 ------------------------------ ----------------------- ------ --------
 P1                             DATE                    IN
FUNCTION TO_CHAR RETURNS VARCHAR2
 Argument Name                  Type                    In/Out Default?
 ------------------------------ ----------------------- ------ --------
 P1                             NUMBER                  IN
FUNCTION TO_CHAR RETURNS VARCHAR2
 Argument Name                  Type                    In/Out Default?
 ------------------------------ ----------------------- ------ --------
 P1                             DATE                    IN
 FORMAT                         VARCHAR2                IN
FUNCTION TO_CHAR RETURNS VARCHAR2
 Argument Name                  Type                    In/Out Default?
 ------------------------------ ----------------------- ------ --------
 P1                             NUMBER                  IN
 FORMAT                         VARCHAR2                IN

SQL>
SQL>
```

```
SQL>
SQL> SELECT
OVERPACK.TO_CHAR(SYSDATE),OVERPACK.TO_CHAR(1234),OVERPACK(SYSDATE,'DD-MON-YYYY'
  2  .
SQL> ED
Wrote file afiedt.buf

  1  SELECT OVERPACK.TO_CHAR(SYSDATE),OVERPACK.TO_CHAR(1234),
  2* OVERPACK.TO_CHAR(SYSDATE,'DD-MON-YYYY') FROM DUAL
SQL> /

OVERPACK.TO_CHAR(SYSDATE)

--------------------------------------------------------------------------------
---------------
OVERPACK.TO_CHAR(1234)

--------------------------------------------------------------------------------
---------------
OVERPACK.TO_CHAR(SYSDATE,'DD-MON-YYYY')

--------------------------------------------------------------------------------
---------------
02.03.2013 061

ONE THOUSAND TWO HUNDRED THIRTY-FOUR

02-MAR-2013



SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL> SPOOL OFF
```

```
SQL>
SQL> .
SQL> ---------------UTL_FILE_PRACTICAL----------------
SQL>
SQL> CONN SYS/ORACLE AS SYSDBA
Connected.
USER is "SYS"
linesize 100
pagesize 100
long 80
SQL>
SQL>
SQL> SHOW PARAMETER UTL_FILE

NAME                                 TYPE        VALUE

------------------------------------ ----------- ------------------------------

utl_file_dir                         string      C:\PL_PRAC

SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL> SHUTDOWN IMMEDIATE
Database closed.
Database dismounted.
ORACLE instance shut down.
SQL>
SQL>
SQL> startup force
ORACLE instance started.

Total System Global Area  612368384 bytes

Fixed Size                  1250428 bytes

Variable Size             167775108 bytes

Database Buffers          436207616 bytes

Redo Buffers                7135232 bytes

Database mounted.
Database opened.
SQL> SHOW PARAMETER UTL_FILE

NAME                                 TYPE        VALUE

------------------------------------ ----------- ------------------------------

utl_file_dir                         string      D:\PL_PRAC

SQL>
SQL>
SQL>
SQL>
```

```
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL> grant CREATE ANY DIRECTORY,DROP ANY DIRECTORY TO SCOTT;

Grant succeeded.

SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL> DESC DBA_DIRECTORIES
 Name                                                      Null?    Type
 ----------------------------------------------------------------- --------
-----------------------------------
 OWNER                                                     NOT NULL VARCHAR2(30)
 DIRECTORY_NAME                                            NOT NULL VARCHAR2(30)
 DIRECTORY_PATH                                                     VARCHAR2(4000)

SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL> SELECT * FROM DBA_DIRECTORIES;

OWNER                          DIRECTORY_NAME

------------------------------ ------------------------------

DIRECTORY_PATH

--------------------------------------------------------------------------------
----------------
SYS                            ADMIN_DIR

C:\ADE\aime_10.2_nt_push\oracle/md/admin



SYS                            DATA_PUMP_DIR

D:\oracle\product\10.2.0\admin\orcl\dpdump\



SYS                            DATA_FILE_DIR
```

```
D:\oracle\product\10.2.0\db_1\demo\schema\sales_history\


SYS                            WORK_DIR

C:\ADE\aime_10.2_nt_push\oracle/work


SYS                            LOG_FILE_DIR

D:\oracle\product\10.2.0\db_1\demo\schema\log\


SYS                            MEDIA_DIR

D:\oracle\product\10.2.0\db_1\demo\schema\product_media\


SYS                            XMLDIR

D:\oracle\product\10.2.0\db_1\demo\schema\order_entry\


SYS                            SUBDIR

D:\oracle\product\10.2.0\db_1\demo\schema\order_entry\/2002/Sep


SYS                            UTL_DIR

C:\PL_PRAC



9 rows selected.

SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL> GRANT SELECT ON DBA_DIRECTORIES TO SCOTT;

Grant succeeded.

SQL>
SQL>
SQL>
SQL>
SQL> CONN SCOT/TTIGER
ERROR:
ORA-01017: invalid username/password; logon denied
```

```
Warning: You are no longer connected to ORACLE.
SQL> CONN SCOTT/TIGER
Connected.
USER is "SCOTT"
linesize 100
pagesize 100
long 80
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL> SELECT * FROM SYS.DBA_DIRECTORIES;

OWNER                          DIRECTORY_NAME

------------------------------ ------------------------------

DIRECTORY_PATH

--------------------------------------------------------------------------------
----------------
SYS                            ADMIN_DIR

C:\ADE\aime_10.2_nt_push\oracle/md/admin


SYS                            DATA_PUMP_DIR

D:\oracle\product\10.2.0\admin\orcl\dpdump\


SYS                            DATA_FILE_DIR

D:\oracle\product\10.2.0\db_1\demo\schema\sales_history\


SYS                            WORK_DIR

C:\ADE\aime_10.2_nt_push\oracle/work


SYS                            LOG_FILE_DIR

D:\oracle\product\10.2.0\db_1\demo\schema\log\


SYS                            MEDIA_DIR

D:\oracle\product\10.2.0\db_1\demo\schema\product_media\


SYS                            XMLDIR

D:\oracle\product\10.2.0\db_1\demo\schema\order_entry\
```

```
SYS                            SUBDIR

D:\oracle\product\10.2.0\db_1\demo\schema\order_entry\/2002/Sep


SYS                            UTL_DIR

C:\PL_PRAC



9 rows selected.

SQL>
SQL>
SQL> ED
Wrote file afiedt.buf

  1  CREATE OR REPLACE VIEW V1 AS
  2*   SELECT EMPNO,ENAME,SAL FROM EMP
SQL> /

View created.

SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL> ED
Wrote file afiedt.buf

  1  CREATE OR REPLACE VIEW V1 AS
  2*   SELECT EMPNO,ENAME,SAL FROM EMP
SQL>
SQL>
SQL>
SQL> CREATE DIRECTORY UTL_FILE AS 'D:\PL_PRAC';

Directory created.

SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL> CREATE DIRECTORY UTL_FILE AS 'D:\PL_PRAC';
CREATE DIRECTORY UTL_FILE AS 'D:\PL_PRAC'
```

```
                *
ERROR at line 1:
ORA-00955: name is already used by an existing object


SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL> DROP DIRECTORY UTL_FILE;

Directory dropped.

SQL> CREATE DIRECTORY UTL_FILE AS 'D:\PL_PRAC';

Directory created.

SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL> ED
Wrote file afiedt.buf

  1      CREATE OR REPLACE PROCEDURE
  2      WRITE_TO_FILE
  3      (
  4       DIR_NAME IN VARCHAR2 DEFAULT 'UTL_FILE',
  5        FILE_NAME IN VARCHAR2 DEFAULT 'EINFO.TXT')IS
  6        F1 UTL_FILE.FILE_TYPE;
  7          PRESENT BOOLEAN;
  8          FLENGTH NUMBER;
  9          BSIZE PLS_INTEGER;
 10          CURSOR C1 IS SELECT * FROM V1;
 11         CNTR NUMBER:=0;
 12         BEGIN
 13        UTL_FILE.FGETATTR(LOCATION=>DIR_NAME,FILENAME=>FILE_NAME,
 14        FEXISTS=>PRESENT,FILE_LENGTH=>FLENGTH,
 15        BLOCK_SIZE=>BSIZE);
 16        IF PRESENT THEN
 17        F1:=UTL_FILE.FOPEN(DIR_NAME,FILE_NAME,'a');     ------APPEND MODE----
 18        UTL_FILE.PUT_LINE(F1,RPAD('*',LENGTH(CURRENT_TIMESTAMP),'*'));
 19        ELSE
 20        F1:=UTL_FILE.FOPEN(DIR_NAME,FILE_NAME,'W');   -----WRITE MODE-------
 21        END IF;
 22        FOR I IN C1 LOOP
 23        CNTR := CNTR +1;
 24     UTL_FILE.PUT_LINE(F1,RPAD(CNTR,10,' ')||RPAD(I.EMPNO,10,' ')
 25     ||RPAD(I.ENAME,10,' ')||RPAD(I.SAL,10,' '));
 26        END LOOP;
 27    DBMS_OUTPUT.PUT_LINE('FILE CREATED ...'||FILE_NAME||'...AS ON
...'||CURRENT_TIMESTAMP);
 28        UTL_FILE.FCLOSE(F1);
 29        EXCEPTION
 30        WHEN UTL_FILE.INVALID_FILEHANDLE THEN
 31        RAISE_APPLICATION_ERROR(-20001,'UNABLE TO WRITE DATA...');
```

```
 32        WHEN OTHERS THEN
 33    &D('MESSAGE FROM ORACLE SERVER....'||SQLERRM);
 34*    END WRITE_TO_FILE;
 35  /

Procedure created.

SQL>
SQL>
SQL>
SQL>
SQL>
SQL> desc WRITE_TO_FILE
PROCEDURE WRITE_TO_FILE
 Argument Name                  Type                    In/Out Default?
 ------------------------------ ----------------------- ------ --------
 DIR_NAME                       VARCHAR2                IN     DEFAULT
 FILE_NAME                      VARCHAR2                IN     DEFAULT

SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL> EXEC WRITE_TO_FILE;
FILE CREATED ...EINFO.TXT...AS ON ...05-MAR-13 08.41.42.828000000 PM +05:00


PL/SQL procedure successfully completed.

SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL> EXEC WRITE_TO_FILE;
FILE CREATED ...EINFO.TXT...AS ON ...05-MAR-13 08.42.58.796000000 PM +05:00


PL/SQL procedure successfully completed.

SQL> EXEC WRITE_TO_FILE('UTL_FILE','MY_NEW_FILE.TXT');
FILE CREATED ...MY_NEW_FILE.TXT...AS ON ...05-MAR-13 08.44.02.718000000 PM +05:00


PL/SQL procedure successfully completed.

SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
```

```
SQL>
SQL> ED
Wrote file afiedt.buf

  1  CREATE OR REPLACE PROCEDURE SHOW_TEXT IS
  2  BEGIN
  3  DBMS_OUTPUT.PUT_LINE('PAKISTAN....');
  4* END;
SQL> /

Procedure created.

SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL> EXEC SHOW_TEXT
PAKISTAN....


PL/SQL procedure successfully completed.

SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL> SAVE C:\TEMP\MY_PROG.SQL
Created file C:\TEMP\MY_PROG.SQL
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL> CL SCR
SQL> CLEAR BUFFER
buffer cleared
SQL> /
SP2-0103: Nothing in SQL buffer to run.
SQL> L
SP2-0223: No lines in SQL buffer.
SQL> SELECT TEXT FROM USER_SOURCE
  2  WHERE NAME='SHOW_TEXT';

TEXT

--------------------------------------------------------------------------------
----------------
PROCEDURE SHOW_TEXT IS

BEGIN
```

```
DBMS_OUTPUT.PUT_LINE('PAKISTAN....');

END;


SQL> @ C:\TEMP\MY_NEW_PROG.SQL

Procedure created.

SQL> SELECT TEXT FROM USER_SOURCE
  2  WHERE NAME='SHOW_TEXT';

TEXT

--------------------------------------------------------------------------------
----------------
PROCEDURE SHOW_TEXT wrapped

a000000

b2

abcd

abcd

abcd

abcd

abcd

abcd

abcd

abcd

abcd

abcd

abcd

abcd

abcd

abcd

abcd

abcd

7

48 85

J+ltHlN9XCwz70X4Lee1o6a1Y7Awg5nnm7+fMr2ywFznM6Uo0ChS8OfMpXSLwMAy/tKGwFKb

skr+KLK957KzHQYwLK4k6rKBpoAgco8vgCQAymmxoIvAgcctyaamGY228w==
```

```
SQL> EXEC SHOW_TEXT;
PAKISTAN....


PL/SQL procedure successfully completed.

SQL> \\
SP2-0042: unknown command "\\" - rest of line ignored.
SQL> \
SP2-0042: unknown command "\" - rest of line ignored.
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL> ED
Wrote file afiedt.buf

  1  CREATE OR REPLACE PROCEDURE SHOW_TEXT IS
  2  BEGIN
  3  DBMS_OUTPUT.PUT_LINE('PAKISTAN....');
  4* END;
SQL> /

Procedure created.

SQL>
SQL>
SQL>
SQL>
SQL>
SQL> SELECT TEXT FROM USER_SOURCE
  2  WHERE NAME='SHOW_TEXT';

TEXT

--------------------------------------------------------------------------------
----------------
PROCEDURE SHOW_TEXT IS

BEGIN

DBMS_OUTPUT.PUT_LINE('PAKISTAN....');

END;


SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL> ED
```

```
Wrote file afiedt.buf

  1  SELECT TEXT FROM USER_SOURCE
  2* WHERE NAME='SHOW_TEXT'
SQL>
SQL> PROCEDURE SHOW_TEXT IS
  2  BEGIN
  3  DBMS_OUTPUT.PUT_LINE('PAKISTAN....');
  4  END;
  5  .
SQL> ED
Wrote file afiedt.buf

  1  PROCEDURE SHOW_TEXT IS
  2  BEGIN
  3  DBMS_OUTPUT.PUT_LINE('PAKISTAN....');
  4* END;
SQL> ED
Wrote file afiedt.buf

  1  BEGIN
  2      DBMS_DDL.CREATE_WRAPPED
  3      ('
  4      CREATE OR REPLACE PROCEDURE SHOW_TEXT IS
  5      BEGIN
  6      DBMS_OUTPUT.PUT_LINE(''PAKISTAN ..........'');
  7      END;
  8     ');
  9*   END;
SQL> /

PL/SQL procedure successfully completed.

SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL> SELECT TEXT FROM USER_SOURCE
  2  WHERE NAME='SHOW_TEXT';

TEXT

--------------------------------------------------------------------------------
---------------
PROCEDURE SHOW_TEXT wrapped

a000000

b2

abcd

abcd

abcd

abcd

abcd

abcd
```

abcd

abcd

abcd

abcd

abcd

abcd

abcd

abcd

abcd

7

60 8d

YaPDzKe3HE/FNs2VAGmqbU+w4gMwg5nnm7+fMr2ywFznM6Uo0ChS8OfMpXSLCabhSeq/riR8

2ZX6eFcZJCEUyiGiKOOGEHpzcSp3HXbtANaEdnOO0i9nTOur2Go9ckRHcKamQW1vTA==

```
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL> SPOOL OFF
```

```
SQL>
SQL>
SQL>
SQL> --------------------job
SQL>
SQL>
SQL> decalre
SP2-0042: unknown command "decalre" - rest of line ignored.
SQL> DECLARE
  2  .
SQL> ED
Wrote file afiedt.buf

  1  CREATE OR REPLACE PROCEDURE ADD_REC
  2  IS
  3  ID NUMBER :=0;
  4  BEGIN
  5  SELECT MAX(EMPNO)+1 INTO ID FROM SCOTT.EMP;
  6  INSERT INTO EMP(EMPNO,ENAME,JOB,SAL,DEPTNO)
  7  VALUES(ID,USER,'SALESMAN',1000,30);
  8  COMMIT;
  9* END;
 10  /

Procedure created.

SQL> SELECT * FROM EMP;

     EMPNO ENAME      JOB              MGR HIREDATE          SAL        COMM
DEPTNO
---------- ---------- --------- ---------- --------- ---------- ----------
----------
      8001 SCOTT      SALESMAN         7788                 1000
30
      5454 SMITH      CLERK            7902 17-DEC-80        900
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       1000
30
      7902 FORD       ANALYST          7566 03-DEC-81      45666
20
      7934 MILLER     CLERK            7782 23-JAN-85       1300
10
      7935 SCOTT      SALESMAN                              1000
```

```
30
      7936 SCOTT      SALESMAN                                  1
30
      7937 SCOTT      SALESMAN                               1000
30
      7938 SCOTT      SALESMAN            1                  1000
30
      7939 SCOTT      SALESMAN            1                  1000
30
      8000 SCOTT      SALESMAN          102                  1000
30

21 rows selected.

SQL> DELETE FROM EMP
  2  WHERE HIREDATE IS NULL;
DELETE FROM EMP
*
ERROR at line 1:
ORA-02292: integrity constraint (SCOTT.SYS_C005474) violated - child record found


SQL> DROP TABLE EMP_IMAGE;

Table dropped.

SQL> DELETE FROM EMP
  2  WHERE HIREDATE IS NULL;

7 rows deleted.

SQL> COMMIT;

Commit complete.

SQL> SELECT * FROM EMP;

     EMPNO ENAME      JOB               MGR HIREDATE          SAL        COMM
DEPTNO
---------- ---------- --------- ---------- --------- ---------- ----------
----------
      5454 SMITH      CLERK           7902 17-DEC-80         900
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       1000
30
      7902 FORD       ANALYST         7566 03-DEC-81      45666
20
      7934 MILLER     CLERK           7782 23-JAN-85       1300
10

14 rows selected.

SQL>
SQL>
SQL> EXEC ADD_REC;

PL/SQL procedure successfully completed.

SQL> SELECT * FROM EMP;

     EMPNO ENAME      JOB              MGR HIREDATE          SAL        COMM
DEPTNO
---------- ---------- --------- ---------- --------- ---------- ----------
----------
      7935 SCOTT      SALESMAN                             1000
30
      5454 SMITH      CLERK           7902 17-DEC-80        900
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       1000
30
      7902 FORD       ANALYST         7566 03-DEC-81      45666
20
      7934 MILLER     CLERK           7782 23-JAN-85       1300
10

15 rows selected.

SQL>
SQL> SELECT SYSDATE FROM DUAL;

SYSDATE

---------

07-MAR-13
```

```
SQL>
SQL>
SQL> ALTER SESSION SET NLS_DATE_FORMAT='DD-MON-YYY HH12:MI:SS AM';

Session altered.

SQL> SELECT SYSDATE FROM DUAL;

SYSDATE

----------------------

07-MAR-013 07:29:31 PM


SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL> /

SYSDATE

----------------------

07-MAR-013 07:29:51 PM


SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL> ED
Wrote file afiedt.buf

  1* SELECT SYSDATE,SYSDATE+1 FROM DUAL
SQL> /

SYSDATE                SYSDATE+1

---------------------- ----------------------

07-MAR-013 07:30:08 PM 08-MAR-013 07:30:08 PM


SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL> ED
Wrote file afiedt.buf
```

```
  1* SELECT SYSDATE,SYSDATE+1/(24*60) FROM DUAL
SQL> /

SYSDATE                 SYSDATE+1/(24*60)

---------------------- ----------------------

07-MAR-013 07:30:56 PM 07-MAR-013 07:31:56 PM


SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL> ED
Wrote file afiedt.buf

  1* SELECT SYSDATE,SYSDATE+1/1440 FROM DUAL
SQL> /

SYSDATE                 SYSDATE+1/1440

---------------------- ----------------------

07-MAR-013 07:31:46 PM 07-MAR-013 07:32:46 PM


SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL> ED
Wrote file afiedt.buf

  1* SELECT SYSDATE,SYSDATE+30/1440 FROM DUAL
SQL> /

SYSDATE                 SYSDATE+30/1440

---------------------- ----------------------

07-MAR-013 07:32:02 PM 07-MAR-013 08:02:02 PM


SQL>
```

```
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL> ED
Wrote file afiedt.buf

  1* SELECT SYSDATE,SYSDATE+240/1440 FROM DUAL
SQL> /

SYSDATE                SYSDATE+240/1440

---------------------- ----------------------

07-MAR-013 07:34:38 PM 07-MAR-013 11:34:38 PM


SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL> ED
Wrote file afiedt.buf

  1   BEGIN
  2             DBMS_SCHEDULER.CREATE_JOB(
  3             JOB_NAME=>'MY_JOB',
  4              JOB_TYPE=>'PLSQL_BLOCK',
  5              JOB_ACTION=>'BEGIN ADD_REC;END;',
  6              START_DATE=>SYSDATE,
  7              REPEAT_INTERVAL=>'SYSDATE+1/1440',
  8              ENABLED=>TRUE,
  9              COMMENTS=>'DEMO FOR JOB SCHEDULE'
 10          );
 11*  END;
 12  /

PL/SQL procedure successfully completed.

SQL>
SQL>
SQL>
SQL> SELECT COUNT(*) FROM EMP;

  COUNT(*)
```

```
----------
         16

SQL> /

  COUNT(*)

----------
         16

SQL> /

  COUNT(*)

----------
         16

SQL> /

  COUNT(*)

----------
         16

SQL> /

  COUNT(*)

----------
         17

SQL>
SQL>
SQL> /

  COUNT(*)

----------
         18

SQL> /

  COUNT(*)

----------
         19

SQL> USER USER_JOBS
SP2-0734: unknown command beginning "USER USER_..." - rest of line ignored.
```

```
SQL>
SQL>
SQL> DESC USER_JOBS
 Name                                                      Null?    Type
 --------------------------------------------------------- --------
-----------------------------------
 JOB                                                       NOT NULL NUMBER
 LOG_USER                                                  NOT NULL VARCHAR2(30)
 PRIV_USER                                                 NOT NULL VARCHAR2(30)
 SCHEMA_USER                                               NOT NULL VARCHAR2(30)
 LAST_DATE                                                          DATE
 LAST_SEC                                                           VARCHAR2(8)
 THIS_DATE                                                          DATE
 THIS_SEC                                                           VARCHAR2(8)
 NEXT_DATE                                                 NOT NULL DATE
 NEXT_SEC                                                           VARCHAR2(8)
 TOTAL_TIME                                                         NUMBER
 BROKEN                                                             VARCHAR2(1)
 INTERVAL                                                  NOT NULL VARCHAR2(200)
 FAILURES                                                           NUMBER
 WHAT                                                               VARCHAR2(4000)
 NLS_ENV                                                            VARCHAR2(4000)
 MISC_ENV                                                           RAW(32)
 INSTANCE                                                           NUMBER

SQL>
SQL>
SQL>
SQL> SELECT JOB FROM USER_JOBS;

no rows selected

SQL> DESC DBA_JOBS
ERROR:
ORA-04043: object "SYS"."DBA_JOBS" does not exist


SQL>
SQL>
SQL> DESC ALL_JOBS
 Name                                                      Null?    Type
 --------------------------------------------------------- --------
-----------------------------------
 JOB                                                       NOT NULL NUMBER
 LOG_USER                                                  NOT NULL VARCHAR2(30)
 PRIV_USER                                                 NOT NULL VARCHAR2(30)
 SCHEMA_USER                                               NOT NULL VARCHAR2(30)
 LAST_DATE                                                          DATE
 LAST_SEC                                                           VARCHAR2(8)
 THIS_DATE                                                          DATE
 THIS_SEC                                                           VARCHAR2(8)
 NEXT_DATE                                                 NOT NULL DATE
 NEXT_SEC                                                           VARCHAR2(8)
 TOTAL_TIME                                                         NUMBER
 BROKEN                                                             VARCHAR2(1)
 INTERVAL                                                  NOT NULL VARCHAR2(200)
 FAILURES                                                           NUMBER
 WHAT                                                               VARCHAR2(4000)
 NLS_ENV                                                            VARCHAR2(4000)
 MISC_ENV                                                           RAW(32)
 INSTANCE                                                           NUMBER

SQL> SELECT JOB FROM ALL_JOBS;
```

```
no rows selected

SQL>
SQL>
SQL>
SQL> SELECT JOB FROM USER_JOBS;

no rows selected

SQL> EXEC DBMS_SCHEDULER.STOP_JOB('MY_JOB');
BEGIN DBMS_SCHEDULER.STOP_JOB('MY_JOB'); END;

*
ERROR at line 1:
ORA-27366: job "SCOTT.MY_JOB" is not running
ORA-06512: at "SYS.DBMS_ISCHED", line 164
ORA-06512: at "SYS.DBMS_SCHEDULER", line 483
ORA-06512: at line 1


SQL>
SQL>
SQL>
SQL>
SQL> EXEC DBMS_SCHEDULER.DROP_JOB('MY_JOB');

PL/SQL procedure successfully completed.

SQL>
SQL>
SQL>
SQL> SELECT JOB FROM USER_JOBS;

no rows selected

SQL> SELECT COUNT(*) FROM EMP;

  COUNT(*)

----------

        23


SQL> SELECT SYSDATE FROM DUAL
  2  ;

SYSDATE

---------------------

07-MAR-013 08:06:28 PM


SQL>
SQL>
SQL>
SQL> ED
Wrote file afiedt.buf

  1  CREATE OR REPLACE PROCEDURE DO_EXE_IMM
  2   (USER_NAME IN VARCHAR2,TABLE_NAME IN VARCHAR2, COL_NAME IN VARCHAR2,
```

```
  3       DATATYPE IN VARCHAR2 DEFAULT 'VARCHAR2(10)',
  4       WHAT_DO IN CHAR) IS
  5       USER_EXCEP EXCEPTION;
  6       PRAGMA EXCEPTION_INIT(USER_EXCEP,-942);
  7      MSG_TYPE VARCHAR2(10);
  8       TAB_NAME VARCHAR2(50):=USER_NAME||'.'||TABLE_NAME;
  9       BEGIN
 10      IF WHAT_DO='C' THEN
 11      MSG_TYPE:='CREATED.';
 12   EXECUTE IMMEDIATE ' ALTER TABLE '||TAB_NAME||' ADD '||COL_NAME||'
'||DATATYPE;
 13     ELSIF WHAT_DO='M' THEN
 14      MSG_TYPE:='MODIFIED.';
 15   EXECUTE IMMEDIATE ' ALTER TABLE '||TAB_NAME||' MODIFY '||COL_NAME||'
        '||DATATYPE;
 16   ELSIF WHAT_DO='D' THEN
 17     MSG_TYPE:='DROPPED.';
 18    EXECUTE IMMEDIATE ' ALTER TABLE '||TAB_NAME||' DROP ('||COL_NAME|| ')';
 19    ELSE
 20  RAISE_APPLICATION_ERROR(-20101,'SORRY, PASS C FOR CREATE, M FOR MODIFY, D FOR
     DROP..');
 21     END IF;
 22        &D('TASK ACCOMPLISHED.. Column '||COL_NAME||' '||msg_type);
 23        EXCEPTION
 24        WHEN USER_EXCEP THEN
 25        RAISE_APPLICATION_ERROR(-20102,'SORRY, INVALID STATEMENT....');
 26        WHEN OTHERS THEN
 27        RAISE_APPLICATION_ERROR(-20901,SQLERRM);
 28*     END;
 29  /

Procedure created.

SQL>
SQL>
SQL>
SQL>
SQL> DESC DO_EXE_IMM
PROCEDURE DO_EXE_IMM
 Argument Name                  Type                    In/Out Default?
 ------------------------------ ----------------------- ------ --------
 USER_NAME                      VARCHAR2                IN
 TABLE_NAME                     VARCHAR2                IN
 COL_NAME                       VARCHAR2                IN
 DATATYPE                       VARCHAR2                IN     DEFAULT
 WHAT_DO                        CHAR                    IN

SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL> DSEC AB
SP2-0042: unknown command "DSEC AB" - rest of line ignored.
SQL> DESC ABC
ERROR:
ORA-04043: object ABC does not exist


SQL> DESC EMP_COPY
```

```
 Name                                                      Null?    Type
 --------------------------------------------------------- -------- ------------------------------------
 EMPNO                                                              NUMBER(4)
 ENAME                                                              VARCHAR2(10)
 JOB                                                                VARCHAR2(9)
 MGR                                                                NUMBER(5)
 HIREDATE                                                           DATE
 SAL                                                                NUMBER(7,2)
 COMM                                                               NUMBER(7,2)
 DEPTNO                                                             NUMBER(2)
 ADRESS                                                             VARCHAR2(2000)
 OLD_SAL                                                            NUMBER(7,2)
 DIFF                                                               NUMBER(7,2)

SQL>
SQL>
SQL> DESE DO_EXE_IMM
SP2-0734: unknown command beginning "DESE DO_EX..." - rest of line ignored.
SQL>
SQL>
SQL> DESC DO_EXE_IMM
PROCEDURE DO_EXE_IMM
 Argument Name                  Type                    In/Out Default?
 ------------------------------ ----------------------- ------ --------
 USER_NAME                      VARCHAR2                IN
 TABLE_NAME                     VARCHAR2                IN
 COL_NAME                       VARCHAR2                IN
 DATATYPE                       VARCHAR2                IN     DEFAULT
 WHAT_DO                        CHAR                    IN

SQL>
SQL>
SQL>
SQL> EXEC DO_EXE_IMM(USER,'EMP_COPY','ADDRESS','VARCHAR2(20)','C');
TASK ACCOMPLISHED.. Column ADDRESS CREATED.


PL/SQL procedure successfully completed.

SQL>
SQL>
SQL>
SQL> DESC EMP_COPY
 Name                                                      Null?    Type
 --------------------------------------------------------- -------- ------------------------------------
 EMPNO                                                              NUMBER(4)
 ENAME                                                              VARCHAR2(10)
 JOB                                                                VARCHAR2(9)
 MGR                                                                NUMBER(5)
 HIREDATE                                                           DATE
 SAL                                                                NUMBER(7,2)
 COMM                                                               NUMBER(7,2)
 DEPTNO                                                             NUMBER(2)
 ADRESS                                                             VARCHAR2(2000)
 OLD_SAL                                                            NUMBER(7,2)
 DIFF                                                               NUMBER(7,2)
 ADDRESS                                                            VARCHAR2(20)

SQL>
SQL>
SQL> EXEC DO_EXE_IMM(USER,'EMP_COPY','ADDRESS','VARCHAR2(30)','M');
```

```
TASK ACCOMPLISHED.. Column ADDRESS MODIFIED.


PL/SQL procedure successfully completed.

SQL>
SQL>
SQL>
SQL>  DESC EMP_COPY
 Name                                                      Null?    Type
 ----------------------------------------------------------------- --------
-----------------------------------
 EMPNO                                                              NUMBER(4)
 ENAME                                                              VARCHAR2(10)
 JOB                                                                VARCHAR2(9)
 MGR                                                                NUMBER(5)
 HIREDATE                                                           DATE
 SAL                                                                NUMBER(7,2)
 COMM                                                               NUMBER(7,2)
 DEPTNO                                                             NUMBER(2)
 ADRESS                                                             VARCHAR2(2000)
 OLD_SAL                                                            NUMBER(7,2)
 DIFF                                                               NUMBER(7,2)
 ADDRESS                                                            VARCHAR2(30)

SQL>
SQL>
SQL>
SQL>
SQL>  EXEC DO_EXE_IMM(USER,'EMP_COPY','ADDRESS','VARCHAR2(30)','D');
TASK ACCOMPLISHED.. Column ADDRESS DROPPED.


PL/SQL procedure successfully completed.

SQL>
SQL>
SQL> DESC EMP_COPY
 Name                                                      Null?    Type
 ----------------------------------------------------------------- --------
-----------------------------------
 EMPNO                                                              NUMBER(4)
 ENAME                                                              VARCHAR2(10)
 JOB                                                                VARCHAR2(9)
 MGR                                                                NUMBER(5)
 HIREDATE                                                           DATE
 SAL                                                                NUMBER(7,2)
 COMM                                                               NUMBER(7,2)
 DEPTNO                                                             NUMBER(2)
 ADRESS                                                             VARCHAR2(2000)
 OLD_SAL                                                            NUMBER(7,2)
 DIFF                                                               NUMBER(7,2)

SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL> EXEC DO_EXE_IMM(USER,'EMP_COPY','ADDRESS','VARCHAR2(30)','T');
BEGIN DO_EXE_IMM(USER,'EMP_COPY','ADDRESS','VARCHAR2(30)','T'); END;

*
```

```
ERROR at line 1:
ORA-20901: ORA-20101: SORRY, PASS C FOR CREATE, M FOR MODIFY, D FOR         DROP..
ORA-06512: at "SCOTT.DO_EXE_IMM", line 27
ORA-06512: at line 1


SQL>
SQL>
SQL>
SQL>
SQL>
SQL> SPOOL OFF
```

```
SQL>
SQL>
SQL> DECLARE
  2  .
SQL> ED
Wrote file afiedt.buf

  1     DECLARE
  2     TYPE R_CURSOR IS REF CURSOR;
  3     C_EMP R_CURSOR;
  4    TYPE REC_EMP IS RECORD(ENAME VARCHAR2(20),SAL NUMBER(6),JOB VARCHAR2(20));
  5          ER REC_EMP;
  6          BEGIN
  7          FOR I IN (SELECT DEPTNO,DNAME FROM DEPT) LOOP
  8  OPEN C_EMP  FOR SELECT ENAME,SAL,JOB FROM EMP WHERE DEPTNO = I.DEPTNO;
  9          &D(I.DEPTNO||'   '||I.DNAME);
 10       &D('----------------------');
 11    ------------------------LOOP DETAIL TABLE----------------------
 12    LOOP
 13        FETCH C_EMP INTO ER;
 14        EXIT WHEN C_EMP%NOTFOUND;
 15       &D(ER.ENAME||'....'||ER.SAL||'...'||ER.JOB);
 16       END LOOP;
 17    ------------------END OF LOOP DETAIL TABLE----------------------
 18       CLOSE C_EMP;
 19       END LOOP;
 20*   END;
 21  /
       FOR I IN (SELECT DEPTNO,DNAME FROM DEPT) LOOP
                                            *
ERROR at line 7:
ORA-06550: line 7, column 43:
PL/SQL: ORA-00942: table or view does not exist
ORA-06550: line 7, column 18:
PL/SQL: SQL Statement ignored
ORA-06550: line 8, column 43:
PL/SQL: ORA-00942: table or view does not exist
ORA-06550: line 8, column 17:
PL/SQL: SQL Statement ignored
ORA-06550: line 9, column 30:
PLS-00364: loop index variable 'I' use is invalid
ORA-06550: line 9, column 9:
PL/SQL: Statement ignored


SQL> SELECT * FROM DEPT;
SELECT * FROM DEPT
              *
ERROR at line 1:
ORA-00942: table or view does not exist


SQL>
SQL>
SQL> SHOW  USER
USER is "SYS"
SQL> CONN SCOTT/TIGER
Connected.
USER is "SCOTT"
linesize 100
pagesize 100
long 80
SQL>     DECLARE
```

```
  2      TYPE R_CURSOR IS REF CURSOR;
  3      C_EMP R_CURSOR;
  4     TYPE REC_EMP IS RECORD(ENAME VARCHAR2(20),SAL NUMBER(6),JOB VARCHAR2(20));
  5           ER REC_EMP;
  6           BEGIN
  7           FOR I IN (SELECT DEPTNO,DNAME FROM DEPT) LOOP
  8   OPEN C_EMP  FOR SELECT ENAME,SAL,JOB FROM EMP WHERE DEPTNO = I.DEPTNO;
  9           &D(I.DEPTNO||'   '||I.DNAME);
 10        &D('----------------------');
 11     ------------------------LOOP DETAIL TABLE----------------------
 12     LOOP
 13         FETCH C_EMP INTO ER;
 14         EXIT WHEN C_EMP%NOTFOUND;
 15        &D(ER.ENAME||'....'||ER.SAL||'...'||ER.JOB);
 16        END LOOP;
 17     ------------------END OF LOOP DETAIL TABLE----------------------
 18        CLOSE C_EMP;
 19        END LOOP;
 20     END;
 21  /
41   UPDATE_REQ

----------------------

99   OTHERS

----------------------

50   HR

----------------------

60   NEW HR

----------------------

10   ACCOUNTING

----------------------

CLARK....2450...MANAGER

KING....5000...PRESIDENT

MILLER....1300...CLERK

20   RESEARCH

----------------------

SMITH....900...CLERK

JONES....2975...MANAGER

SCOTT....3000...ANALYST

ADAMS....1100...CLERK

FORD....45666...ANALYST

30   SALES

----------------------
```

```
ALLEN....1600...SALESMAN

WARD....1250...SALESMAN

MARTIN....1250...SALESMAN

BLAKE....2850...MANAGER

TURNER....1500...SALESMAN

JAMES....1000...CLERK

40   OPERATIONS

----------------------


PL/SQL procedure successfully completed.

SQL> ED
Wrote file afiedt.buf

  1      DECLARE
  2      TYPE R_CURSOR IS REF CURSOR;
  3      C_EMP R_CURSOR;
  4     TYPE REC_EMP IS RECORD(ENAME VARCHAR2(20),SAL NUMBER(6),JOB VARCHAR2(20));
  5           ER REC_EMP;
  6           BEGIN
  7           FOR I IN (SELECT DEPTNO,DNAME FROM DEPT ORDER BY DEPTNO) LOOP
  8   OPEN C_EMP  FOR SELECT ENAME,SAL,JOB FROM EMP WHERE DEPTNO = I.DEPTNO;
  9           &D(I.DEPTNO||'   '||I.DNAME);
 10        &D('----------------------');
 11     ------------------------LOOP DETAIL TABLE----------------------
 12     LOOP
 13         FETCH C_EMP INTO ER;
 14         EXIT WHEN C_EMP%NOTFOUND;
 15        &D(ER.ENAME||'....'||ER.SAL||'...'||ER.JOB);
 16        END LOOP;
 17     ------------------END OF LOOP DETAIL TABLE----------------------
 18        CLOSE C_EMP;
 19        END LOOP;
 20*    END;
SQL> /
10   ACCOUNTING

----------------------

CLARK....2450...MANAGER

KING....5000...PRESIDENT

MILLER....1300...CLERK

20   RESEARCH

----------------------

SMITH....900...CLERK

JONES....2975...MANAGER

SCOTT....3000...ANALYST
```

```
ADAMS....1100...CLERK

FORD....45666...ANALYST

30   SALES

----------------------

ALLEN....1600...SALESMAN

WARD....1250...SALESMAN

MARTIN....1250...SALESMAN

BLAKE....2850...MANAGER

TURNER....1500...SALESMAN

JAMES....1000...CLERK

40   OPERATIONS

----------------------

41   UPDATE_REQ

----------------------

50   HR

----------------------

60   NEW HR

----------------------

99   OTHERS

----------------------


PL/SQL procedure successfully completed.

SQL> ED
Wrote file afiedt.buf

  1  CREATE OR REPLACE PROCEDURE T1
  2         (
  3         P_DEPTNO IN NUMBER,P_CURSOR OUT SYS_REFCURSOR) IS
  4                 BEGIN
  5              OPEN P_CURSOR FOR SELECT * FROM EMP WHERE DEPTNO=P_DEPTNO;
  6*           END;
  7  /

Procedure created.

SQL>
SQL>
SQL>
SQL>
SQL> DESC T1
PROCEDURE T1
```

```
 Argument Name                  Type                      In/Out Default?
 ------------------------------ ------------------------- ------ --------
 P_DEPTNO                       NUMBER                    IN
 P_CURSOR                       REF CURSOR                OUT

SQL>
SQL>
SQL>
SQL>
SQL>
SQL> VAR EMP_ROW REFCUROSR;
Usage: VAR[IABLE] [ <variable> [ NUMBER | CHAR | CHAR (n [CHAR|BYTE]) |
                     VARCHAR2 (n [CHAR|BYTE]) | NCHAR | NCHAR (n) |
                     NVARCHAR2 (n) | CLOB | NCLOB | REFCURSOR |
                     BINARY_FLOAT | BINARY_DOUBLE ] ]
SQL> VAR EMP_ROW REFCURSOR;
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL> EXEC T1(30,:EMP_ROW);

PL/SQL procedure successfully completed.

SQL>
SQL>
SQL>
SQL>
SQL> PRINT

     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
      7698 BLAKE      MANAGER         7839 01-MAY-81       2850
30
      7844 TURNER     SALESMAN        7698 08-SEP-81       1500          0
30
      7900 JAMES      CLERK           7698 03-DEC-81       1000
30

6 rows selected.

SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL> ED
Wrote file afiedt.buf

  1  DECLARE
  2      JOB_ID VARCHAR2(20) := '&JOB';
  3           TYPE R_CURSOR IS REF CURSOR RETURN EMP%ROWTYPE;
```

```
  4           C_EMP R_CURSOR;
  5           ER C_EMP%ROWTYPE;
  6           BEGIN
  7           OPEN C_EMP FOR SELECT * FROM EMP WHERE JOB=JOB_ID;
  8   LOOP
  9           FETCH C_EMP INTO ER;
 10         EXIT WHEN C_EMP%NOTFOUND;
 11        &D(ER.ENAME||'='||ER.SAL);
 12         END LOOP;
 13        CLOSE C_EMP;
 14*      END;
 15  /
Enter value for job: SALESMAN
ALLEN=1600

WARD=1250

MARTIN=1250

TURNER=1500


PL/SQL procedure successfully completed.

SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL> ED
Wrote file afiedt.buf

  1  CREATE OR REPLACE PROCEDURE
  2         CREATE_TABLE
  3     (TABLE_NAME IN VARCHAR2,MY_COLUMNS IN VARCHAR2)IS
  4          NN NUMBER:=0;
  5          CURSOR_NAME INTEGER;
  6          BEGIN
  7           CURSOR_NAME:=DBMS_SQL.OPEN_CURSOR;
  8  DBMS_SQL.PARSE(CURSOR_NAME,'CREATE TABLE '||TABLE_NAME||'('||MY_COLUMNS||')',
  9           DBMS_SQL.NATIVE);
 10           &D(TABLE_NAME||' CREATED...');
 11         DBMS_SQL.CLOSE_CURSOR(CURSOR_NAME);
 12       EXCEPTION
 13        WHEN OTHERS THEN
 14         RAISE_APPLICATION_ERROR(-20302,SQLERRM);
 15*     END;
SQL> /

Procedure created.

SQL>
SQL>
SQL>
SQL>
```

```
SQL>
SQL> DESC CREATE_TABLE
PROCEDURE CREATE_TABLE
 Argument Name                  Type                    In/Out Default?
 ------------------------------ ----------------------- ------ --------
 TABLE_NAME                     VARCHAR2                IN
 MY_COLUMNS                     VARCHAR2                IN

SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL> EXEC CREATE_TABLE('STUDENT','EMPNO NUMBER(4),ENAME VARCHAR2(20)');
STUDENT CREATED...


PL/SQL procedure successfully completed.

SQL>
SQL>
SQL>
SQL>
SQL> DESC STUDENT
 Name                                                      Null?    Type
 ----------------------------------------------------------------- --------
-------------------------------------
 EMPNO                                                              NUMBER(4)
 ENAME                                                              VARCHAR2(20)

SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL> CL SCR
SQL> CREATE OR REPLACE PROCEDURE.
  2  ED
  3  .
SQL> ED
Wrote file afiedt.buf

  1  CREATE OR REPLACE PROCEDURE SHOW_DATA(V_EMPNO NUMBER) IS
  2  EMP_REC EMP%ROWTYPE;
  3  BEGIN
  4  SELECT * INTO EMP_REC FROM SCOTT.EMP
  5  WHERE EMPNO=V_EMPNO;
  6  &D(EMP_REC.ENAME||'  '||EMP_REC.JOB||'  '||EMP_REC.SAL);
  7  EXCEPTION
  8  WHEN OTHERS THEN
  9  &D(SQLCODE||'      '||SQLERRM);
 10* END;
 11  /

Procedure created.

SQL> EXEC SHOW_DATA(7788);
SCOTT  ANALYST  3000
```

```
PL/SQL procedure successfully completed.

SQL> REVOKE SELECT ON EMP FROM HR;

Revoke succeeded.

SQL> GRANT EXECUTE ON SHOW_DATA TO HR;

Grant succeeded.

SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL> CONN HR/HR
Connected.
USER is "HR"
linesize 100
pagesize 100
long 80
SQL>
SQL>
SQL>
SQL> SELECT * FROM SCOTT.EMP;
SELECT * FROM SCOTT.EMP
                   *
ERROR at line 1:
ORA-01031: insufficient privileges


SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL> EXEC SCOTT.SHOW_DATA(7788);
SCOTT  ANALYST  3000


PL/SQL procedure successfully completed.

SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL> CONN SCOTT/TIGER
Connected.
USER is "SCOTT"
linesize 100
pagesize 100
long 80
SQL>  CREATE OR REPLACE PROCEDURE SHOW_DATA(V_EMPNO NUMBER) IS
  2   EMP_REC EMP%ROWTYPE;
```

```
  3   BEGIN
  4   SELECT * INTO EMP_REC FROM SCOTT.EMP
  5   WHERE EMPNO=V_EMPNO;
  6   &D(EMP_REC.ENAME||'  '||EMP_REC.JOB||'  '||EMP_REC.SAL);
  7   EXCEPTION
  8   WHEN OTHERS THEN
  9   &D(SQLCODE||'      '||SQLERRM);
 10   END;
 11  .
SQL> ED
Wrote file afiedt.buf

  1   CREATE OR REPLACE PROCEDURE SHOW_DATA(V_EMPNO NUMBER)AUTHID DEFINER IS
  2   EMP_REC EMP%ROWTYPE;
  3   BEGIN
  4   SELECT * INTO EMP_REC FROM SCOTT.EMP
  5   WHERE EMPNO=V_EMPNO;
  6   &D(EMP_REC.ENAME||'  '||EMP_REC.JOB||'  '||EMP_REC.SAL);
  7   EXCEPTION
  8   WHEN OTHERS THEN
  9   &D(SQLCODE||'      '||SQLERRM);
 10*  END;
SQL> /

Procedure created.

SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL> ED
Wrote file afiedt.buf

  1   CREATE OR REPLACE PROCEDURE SHOW_DATA(V_EMPNO NUMBER)AUTHID CURRENT_USER IS
  2   EMP_REC EMP%ROWTYPE;
  3   BEGIN
  4   SELECT * INTO EMP_REC FROM SCOTT.EMP
  5   WHERE EMPNO=V_EMPNO;
  6   &D(EMP_REC.ENAME||'  '||EMP_REC.JOB||'  '||EMP_REC.SAL);
  7   EXCEPTION
  8   WHEN OTHERS THEN
  9   &D(SQLCODE||'      '||SQLERRM);
 10*  END;
SQL> /

Procedure created.

SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL> CONN HR/HR
Connected.
USER is "HR"
linesize 100
pagesize 100
```

```
long 80
SQL> EXEC SCOTT.SHOW_DATA(7839);
-1031      ORA-01031: insufficient privileges


PL/SQL procedure successfully completed.

SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL> CONN SCOTT/TIGER
Connected.
USER is "SCOTT"
linesize 100
pagesize 100
long 80
SQL>
SQL>
SQL>
SQL>
SQL> GRANT SELECT ON EMP TO HR;

Grant succeeded.

SQL> CONN HR/HR
Connected.
USER is "HR"
linesize 100
pagesize 100
long 80
SQL> EXEC SCOTT.SHOW_DATA(7839);
KING  PRESIDENT  5000


PL/SQL procedure successfully completed.

SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL> SELECT * FROM EMP;
SELECT * FROM EMP
              *
ERROR at line 1:
ORA-00942: table or view does not exist


SQL> CONN SCOTT/TIGE
ERROR:
```

```
ORA-01017: invalid username/password; logon denied


Warning: You are no longer connected to ORACLE.
SQL> CONN SCOTT/TIGER
Connected.
USER is "SCOTT"
linesize 100
pagesize 100
long 80
SQL> CL SCR
SQL> SELECT * FROM EMP;

     EMPNO ENAME      JOB              MGR HIREDATE          SAL       COMM
DEPTNO
---------- ---------- --------- ---------- --------- ---------- ----------
----------
      5454 SMITH      CLERK            7902 17-DEC-80        900
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       1000
30
      7902 FORD       ANALYST          7566 03-DEC-81      45666
20
      7934 MILLER     CLERK            7782 23-JAN-85       1300
10

14 rows selected.

SQL> DESC EMP
 Name                                                  Null?    Type
 ----------------------------------------------------- -------- ------
----------------------------------
 EMPNO                                                 NOT NULL NUMBER(4)
 ENAME                                                          VARCHAR2(10)
 JOB                                                            VARCHAR2(9)
 MGR                                                            NUMBER(5)
 HIREDATE                                                       DATE
 SAL                                                            NUMBER(7,2)
 COMM                                                           NUMBER(7,2)
 DEPTNO                                                         NUMBER(2)

SQL>
SQL>
```

```
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL> SELECT * FROM V1;

     EMPNO ENAME              SAL

---------- ---------- ----------

      5454 SMITH             900

      7499 ALLEN            1600

      7521 WARD             1250

      7566 JONES            2975

      7654 MARTIN           1250

      7698 BLAKE            2850

      7782 CLARK            2450

      7788 SCOTT            3000

      7839 KING             5000

      7844 TURNER           1500

      7876 ADAMS            1100

      7900 JAMES            1000

      7902 FORD            45666

      7934 MILLER           1300


14 rows selected.

SQL>
SQL>
SQL>
SQL>
SQL>
SQL> DESC USER_VIEWS
 Name                                                      Null?    Type
 --------------------------------------------------------- -------- ------
-----------------------------------
 VIEW_NAME                                                 NOT NULL VARCHAR2(30)
 TEXT_LENGTH                                                        NUMBER
 TEXT                                                               LONG
 TYPE_TEXT_LENGTH                                                   NUMBER
 TYPE_TEXT                                                          VARCHAR2(4000)
 OID_TEXT_LENGTH                                                    NUMBER
 OID_TEXT                                                           VARCHAR2(4000)
 VIEW_TYPE_OWNER                                                    VARCHAR2(30)
 VIEW_TYPE                                                          VARCHAR2(30)
 SUPERVIEW_NAME                                                     VARCHAR2(30)
```

```
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL> SELECT TEXT FROM USER_VIEWS
  2  WHERE VIEW_NAME='V1';

TEXT

--------------------------------------------------------------------------------

SELECT EMPNO,ENAME,SAL FROM EMP


SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL> SELECT DBMS_METADATA.GET_DDL('VIEW','V1',USER) FROM DUAL;

DBMS_METADATA.GET_DDL('VIEW','V1',USER)

--------------------------------------------------------------------------------


  CREATE OR REPLACE FORCE VIEW "SCOTT"."V1" ("EMPNO", "ENAME", "SAL") AS

  SEL



SQL> SET LONG 10000
SQL> /

DBMS_METADATA.GET_DDL('VIEW','V1',USER)

--------------------------------------------------------------------------------


  CREATE OR REPLACE FORCE VIEW "SCOTT"."V1" ("EMPNO", "ENAME", "SAL") AS

  SELECT EMPNO,ENAME,SAL FROM EMP



SQL>
```

```
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL> ED
Wrote file afiedt.buf

  1* SELECT DBMS_METADATA.GET_DDL('TABLE','EMP','SCOTT') FROM DUAL
SQL> /

DBMS_METADATA.GET_DDL('TABLE','EMP','SCOTT')

--------------------------------------------------------------------------------


  CREATE TABLE "SCOTT"."EMP"
   (    "EMPNO" NUMBER(4,0),
        "ENAME" VARCHAR2(10),
        "JOB" VARCHAR2(9),
        "MGR" NUMBER(5,0),
        "HIREDATE" DATE,
        "SAL" NUMBER(7,2),
        "COMM" NUMBER(7,2),
        "DEPTNO" NUMBER(2,0),
         CONSTRAINT "PK_EMP" PRIMARY KEY ("EMPNO")
  USING INDEX PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS
  STORAGE(INITIAL 65536 NEXT 1048576 MINEXTENTS 1 MAXEXTENTS 2147483645
  PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1 BUFFER_POOL DEFAULT)
  TABLESPACE "USERS"  ENABLE,
         CONSTRAINT "FK_DEPTNO" FOREIGN KEY ("DEPTNO")
          REFERENCES "SCOTT"."DEPT" ("DEPTNO") ENABLE
   ) PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255 NOCOMPRESS LOGGING
  STORAGE(INITIAL 65536 NEXT 1048576 MINEXTENTS 1 MAXEXTENTS 2147483645
  PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1 BUFFER_POOL DEFAULT)
  TABLESPACE "USERS"
```

```
SQL>
SQL>
SQL>
SQL>
SQL> DESC DBMS_METADATA
FUNCTION ADD_TRANSFORM RETURNS NUMBER
 Argument Name                  Type                    In/Out Default?
 ------------------------------ ----------------------- ------ --------
 HANDLE                         NUMBER                  IN
 NAME                           VARCHAR2                IN
 ENCODING                       VARCHAR2                IN     DEFAULT
 OBJECT_TYPE                    VARCHAR2                IN     DEFAULT
FUNCTION CHECK_MATCH_TEMPLATE RETURNS NUMBER
 Argument Name                  Type                    In/Out Default?
 ------------------------------ ----------------------- ------ --------
 POBJNO                         NUMBER                  IN
 SPCNT                          NUMBER                  IN
FUNCTION CHECK_MATCH_TEMPLATE_LOB RETURNS NUMBER
 Argument Name                  Type                    In/Out Default?
 ------------------------------ ----------------------- ------ --------
 POBJNO                         NUMBER                  IN
 SPCNT                          NUMBER                  IN
FUNCTION CHECK_MATCH_TEMPLATE_PAR RETURNS NUMBER
 Argument Name                  Type                    In/Out Default?
 ------------------------------ ----------------------- ------ --------
 POBJNO                         NUMBER                  IN
 SPCNT                          NUMBER                  IN
PROCEDURE CHECK_TYPE
 Argument Name                  Type                    In/Out Default?
 ------------------------------ ----------------------- ------ --------
 SCHEMA                         VARCHAR2                IN
 TYPE_NAME                      VARCHAR2                IN
 VERSION                        VARCHAR2                IN
 HASHCODE                       VARCHAR2                IN
 TYPEID                         VARCHAR2                IN
PROCEDURE CLOSE
 Argument Name                  Type                    In/Out Default?
 ------------------------------ ----------------------- ------ --------
 HANDLE                         NUMBER                  IN
FUNCTION CONVERT RETURNS KU$_MULTI_DDLS
 Argument Name                  Type                    In/Out Default?
 ------------------------------ ----------------------- ------ --------
 HANDLE                         NUMBER                  IN
 DOCUMENT                       XMLTYPE                 IN
FUNCTION CONVERT RETURNS KU$_MULTI_DDLS
 Argument Name                  Type                    In/Out Default?
 ------------------------------ ----------------------- ------ --------
 HANDLE                         NUMBER                  IN
 DOCUMENT                       CLOB                    IN
FUNCTION CONVERT RETURNS CLOB
 Argument Name                  Type                    In/Out Default?
 ------------------------------ ----------------------- ------ --------
 HANDLE                         NUMBER                  IN
 DOCUMENT                       CLOB                    IN
 OFFSETS                        TABLE OF TABLE OF       OUT
PROCEDURE CONVERT
 Argument Name                  Type                    In/Out Default?
 ------------------------------ ----------------------- ------ --------
 HANDLE                         NUMBER                  IN
 DOCUMENT                       XMLTYPE                 IN
 RESULT                         CLOB                    IN/OUT
PROCEDURE CONVERT
 Argument Name                  Type                    In/Out Default?
```

```
 ------------------------------ ----------------------- ------ --------
 HANDLE                         NUMBER                  IN
 DOCUMENT                       CLOB                    IN
 RESULT                         CLOB                    IN/OUT
FUNCTION FETCH_CLOB RETURNS CLOB
 Argument Name                  Type                    In/Out Default?
 ------------------------------ ----------------------- ------ --------
 HANDLE                         NUMBER                  IN
 CACHE_LOB                      BOOLEAN                 IN     DEFAULT
 LOB_DURATION                   BINARY_INTEGER          IN     DEFAULT
PROCEDURE FETCH_CLOB
 Argument Name                  Type                    In/Out Default?
 ------------------------------ ----------------------- ------ --------
 HANDLE                         NUMBER                  IN
 XMLDOC                         CLOB                    IN/OUT
FUNCTION FETCH_DDL RETURNS KU$_DDLS
 Argument Name                  Type                    In/Out Default?
 ------------------------------ ----------------------- ------ --------
 HANDLE                         NUMBER                  IN
FUNCTION FETCH_DDL_TEXT RETURNS VARCHAR2
 Argument Name                  Type                    In/Out Default?
 ------------------------------ ----------------------- ------ --------
 HANDLE                         NUMBER                  IN
 PARTIAL                        NUMBER                  OUT
FUNCTION FETCH_OBJNUMS RETURNS KU$_OBJNUMSET
 Argument Name                  Type                    In/Out Default?
 ------------------------------ ----------------------- ------ --------
 HANDLE                         NUMBER                  IN
FUNCTION FETCH_SORTED_OBJNUMS RETURNS KU$_OBJNUMPAIRLIST
 Argument Name                  Type                    In/Out Default?
 ------------------------------ ----------------------- ------ --------
 HANDLE                         NUMBER                  IN
FUNCTION FETCH_XML RETURNS XMLTYPE
 Argument Name                  Type                    In/Out Default?
 ------------------------------ ----------------------- ------ --------
 HANDLE                         NUMBER                  IN
PROCEDURE FETCH_XML_CLOB
 Argument Name                  Type                    In/Out Default?
 ------------------------------ ----------------------- ------ --------
 HANDLE                         NUMBER                  IN
 DOC                            CLOB                    IN/OUT
 PARSED_ITEMS                   KU$_PARSED_ITEMS        IN/OUT
 OBJECT_TYPE_PATH               VARCHAR2                OUT
PROCEDURE FETCH_XML_CLOB
 Argument Name                  Type                    In/Out Default?
 ------------------------------ ----------------------- ------ --------
 HANDLE                         NUMBER                  IN
 DOC                            CLOB                    IN/OUT
 PARSED_ITEMS                   KU$_PARSED_ITEMS        IN/OUT
 OBJECT_TYPE_PATH               VARCHAR2                OUT
 SEQNO                          NUMBER                  OUT
 PROCOBJ_ERRORS                 KU$_VCNT                OUT
PROCEDURE FREE_CONTEXT_ENTRY
 Argument Name                  Type                    In/Out Default?
 ------------------------------ ----------------------- ------ --------
 IND                            NUMBER                  IN
FUNCTION GET_ACTION_INSTANCE RETURNS KU$_PROCOBJ_LINES
 Argument Name                  Type                    In/Out Default?
 ------------------------------ ----------------------- ------ --------
 PACKAGE                        VARCHAR2                IN
 PKG_SCHEMA                     VARCHAR2                IN
 FUNCTION                       VARCHAR2                IN
 NAME                           VARCHAR2                IN
```

```
 SCHEMA                         VARCHAR2                IN
 NAMESPACE                      NUMBER                  IN
 OBJTYPE                        NUMBER                  IN
 PREPOST                        NUMBER                  IN
 ISDBA                          NUMBER                  IN
FUNCTION GET_ACTION_SCHEMA RETURNS KU$_PROCOBJ_LINES
 Argument Name                  Type                    In/Out Default?
 ------------------------------ ----------------------- ------ --------
 PACKAGE                        VARCHAR2                IN
 PKG_SCHEMA                     VARCHAR2                IN
 FUNCTION                       VARCHAR2                IN
 SCHEMA                         VARCHAR2                IN
 PREPOST                        NUMBER                  IN
 ISDBA                          NUMBER                  IN
FUNCTION GET_ACTION_SYS RETURNS KU$_PROCOBJ_LINES
 Argument Name                  Type                    In/Out Default?
 ------------------------------ ----------------------- ------ --------
 PACKAGE                        VARCHAR2                IN
 PKG_SCHEMA                     VARCHAR2                IN
 FUNCTION                       VARCHAR2                IN
 PREPOST                        NUMBER                  IN
FUNCTION GET_CANONICAL_VSN RETURNS VARCHAR2
 Argument Name                  Type                    In/Out Default?
 ------------------------------ ----------------------- ------ --------
 VERSION                        VARCHAR2                IN
FUNCTION GET_DDL RETURNS CLOB
 Argument Name                  Type                    In/Out Default?
 ------------------------------ ----------------------- ------ --------
 OBJECT_TYPE                    VARCHAR2                IN
 NAME                           VARCHAR2                IN
 SCHEMA                         VARCHAR2                IN     DEFAULT
 VERSION                        VARCHAR2                IN     DEFAULT
 MODEL                          VARCHAR2                IN     DEFAULT
 TRANSFORM                      VARCHAR2                IN     DEFAULT
FUNCTION GET_DEPENDENT_DDL RETURNS CLOB
 Argument Name                  Type                    In/Out Default?
 ------------------------------ ----------------------- ------ --------
 OBJECT_TYPE                    VARCHAR2                IN
 BASE_OBJECT_NAME               VARCHAR2                IN
 BASE_OBJECT_SCHEMA             VARCHAR2                IN     DEFAULT
 VERSION                        VARCHAR2                IN     DEFAULT
 MODEL                          VARCHAR2                IN     DEFAULT
 TRANSFORM                      VARCHAR2                IN     DEFAULT
 OBJECT_COUNT                   NUMBER                  IN     DEFAULT
FUNCTION GET_DEPENDENT_XML RETURNS CLOB
 Argument Name                  Type                    In/Out Default?
 ------------------------------ ----------------------- ------ --------
 OBJECT_TYPE                    VARCHAR2                IN
 BASE_OBJECT_NAME               VARCHAR2                IN
 BASE_OBJECT_SCHEMA             VARCHAR2                IN     DEFAULT
 VERSION                        VARCHAR2                IN     DEFAULT
 MODEL                          VARCHAR2                IN     DEFAULT
 TRANSFORM                      VARCHAR2                IN     DEFAULT
 OBJECT_COUNT                   NUMBER                  IN     DEFAULT
FUNCTION GET_DOMIDX_METADATA RETURNS KU$_PROCOBJ_LINES
 Argument Name                  Type                    In/Out Default?
 ------------------------------ ----------------------- ------ --------
 INDEX_NAME                     VARCHAR2                IN
 INDEX_SCHEMA                   VARCHAR2                IN
 TYPE_NAME                      VARCHAR2                IN
 TYPE_SCHEMA                    VARCHAR2                IN
 FLAGS                          NUMBER                  IN
PROCEDURE GET_DPSTRM_MD
```

```
 Argument Name                  Type                    In/Out Default?
 ------------------------------ ----------------------- ------ --------
 SCHEMA                         VARCHAR2                IN
 NAME                           VARCHAR2                IN
 MDVERSION                      VARCHAR2                IN     DEFAULT
 DPAPIVERSION                   NUMBER                  IN     DEFAULT
 DOC                            CLOB                    IN/OUT
 NETWORK_LINK                   VARCHAR2                IN     DEFAULT
 FORCE_LOB_BE                   BOOLEAN                 IN     DEFAULT
 FORCE_NO_ENCRYPT               BOOLEAN                 IN     DEFAULT
FUNCTION GET_GRANTED_DDL RETURNS CLOB
 Argument Name                  Type                    In/Out Default?
 ------------------------------ ----------------------- ------ --------
 OBJECT_TYPE                    VARCHAR2                IN
 GRANTEE                        VARCHAR2                IN     DEFAULT
 VERSION                        VARCHAR2                IN     DEFAULT
 MODEL                          VARCHAR2                IN     DEFAULT
 TRANSFORM                      VARCHAR2                IN     DEFAULT
 OBJECT_COUNT                   NUMBER                  IN     DEFAULT
FUNCTION GET_GRANTED_XML RETURNS CLOB
 Argument Name                  Type                    In/Out Default?
 ------------------------------ ----------------------- ------ --------
 OBJECT_TYPE                    VARCHAR2                IN
 GRANTEE                        VARCHAR2                IN     DEFAULT
 VERSION                        VARCHAR2                IN     DEFAULT
 MODEL                          VARCHAR2                IN     DEFAULT
 TRANSFORM                      VARCHAR2                IN     DEFAULT
 OBJECT_COUNT                   NUMBER                  IN     DEFAULT
FUNCTION GET_JAVA_METADATA RETURNS KU$_JAVA_T
 Argument Name                  Type                    In/Out Default?
 ------------------------------ ----------------------- ------ --------
 JAVA_NAME                      VARCHAR2                IN
 JAVA_SCHEMA                    VARCHAR2                IN
 TYPE_NUM                       NUMBER                  IN
FUNCTION GET_PLUGTS_BLK RETURNS KU$_PROCOBJ_LINES
 Argument Name                  Type                    In/Out Default?
 ------------------------------ ----------------------- ------ --------
 BLOCKID                        NUMBER                  IN
FUNCTION GET_PREPOST_TABLE_ACT RETURNS KU$_TACTION_LIST_T
 Argument Name                  Type                    In/Out Default?
 ------------------------------ ----------------------- ------ --------
 PREPOST                        NUMBER                  IN
 SCHEMA                         VARCHAR2                IN
 TNAME                          VARCHAR2                IN
FUNCTION GET_PROCOBJ RETURNS KU$_PROCOBJ_LINES
 Argument Name                  Type                    In/Out Default?
 ------------------------------ ----------------------- ------ --------
 PACKAGE                        VARCHAR2                IN
 PKG_SCHEMA                     VARCHAR2                IN
 FUNCTION                       VARCHAR2                IN
 OBJID                          NUMBER                  IN
 ISDBA                          BINARY_INTEGER          IN
FUNCTION GET_PROCOBJ_GRANT RETURNS KU$_PROCOBJ_LINES
 Argument Name                  Type                    In/Out Default?
 ------------------------------ ----------------------- ------ --------
 PACKAGE                        VARCHAR2                IN
 PKG_SCHEMA                     VARCHAR2                IN
 FUNCTION                       VARCHAR2                IN
 OBJID                          NUMBER                  IN
 ISDBA                          BINARY_INTEGER          IN
FUNCTION GET_QUERY RETURNS VARCHAR2
 Argument Name                  Type                    In/Out Default?
 ------------------------------ ----------------------- ------ --------
```

```
 HANDLE                         NUMBER                  IN
FUNCTION GET_SYSPRIVS RETURNS KU$_PROCOBJ_LINES
 Argument Name                  Type                    In/Out Default?
 ------------------------------ ----------------------- ------ --------
 PACKAGE                        VARCHAR2                IN
 PKG_SCHEMA                     VARCHAR2                IN
 FUNCTION                       VARCHAR2                IN
FUNCTION GET_XML RETURNS CLOB
 Argument Name                  Type                    In/Out Default?
 ------------------------------ ----------------------- ------ --------
 OBJECT_TYPE                    VARCHAR2                IN
 NAME                           VARCHAR2                IN
 SCHEMA                         VARCHAR2                IN     DEFAULT
 VERSION                        VARCHAR2                IN     DEFAULT
 MODEL                          VARCHAR2                IN     DEFAULT
 TRANSFORM                      VARCHAR2                IN     DEFAULT
PROCEDURE NETWORK_CALLOUTS
 Argument Name                  Type                    In/Out Default?
 ------------------------------ ----------------------- ------ --------
 HANDLE                         NUMBER                  IN
FUNCTION NETWORK_FETCH_CLOB RETURNS VARCHAR2
 Argument Name                  Type                    In/Out Default?
 ------------------------------ ----------------------- ------ --------
 HANDLE                         NUMBER                  IN
 DO_XSL_PARSE                   NUMBER                  IN
 PARTIAL                        NUMBER                  OUT
 PARSE_DELIM                    VARCHAR2                OUT
 DO_CALLOUT                     NUMBER                  OUT
 HAVE_ERRORS                    NUMBER                  OUT
FUNCTION NETWORK_FETCH_ERRORS RETURNS VARCHAR2
 Argument Name                  Type                    In/Out Default?
 ------------------------------ ----------------------- ------ --------
 HANDLE                         NUMBER                  IN
 CNT                            NUMBER                  OUT
 PARTIAL                        NUMBER                  OUT
 SEQNO                          NUMBER                  OUT
 PATH                           VARCHAR2                OUT
FUNCTION NETWORK_FETCH_PARSE RETURNS VARCHAR2
 Argument Name                  Type                    In/Out Default?
 ------------------------------ ----------------------- ------ --------
 HANDLE                         NUMBER                  IN
 CNT                            NUMBER                  OUT
 PARTIAL                        NUMBER                  OUT
 SEQNO                          NUMBER                  OUT
 PATH                           VARCHAR2                OUT
FUNCTION NETWORK_OPEN RETURNS NUMBER
 Argument Name                  Type                    In/Out Default?
 ------------------------------ ----------------------- ------ --------
 OBJECT_TYPE                    VARCHAR2                IN
 VERSION                        VARCHAR2                IN     DEFAULT
 MODEL                          VARCHAR2                IN     DEFAULT
 CLIENT_VERSION                 NUMBER                  IN
 PROTOCOL_VERSION               NUMBER                  OUT
PROCEDURE NET_SET_DEBUG
 Argument Name                  Type                    In/Out Default?
 ------------------------------ ----------------------- ------ --------
 ON_OFF                         NUMBER                  IN
FUNCTION OKTOEXP_2NDARY_TABLE RETURNS BINARY_INTEGER
 Argument Name                  Type                    In/Out Default?
 ------------------------------ ----------------------- ------ --------
 TAB_OBJ_NUM                    NUMBER                  IN
FUNCTION OPEN RETURNS NUMBER
 Argument Name                  Type                    In/Out Default?
```

```
 ------------------------------ ---------------------- ------ --------
 OBJECT_TYPE                    VARCHAR2               IN
 VERSION                        VARCHAR2               IN     DEFAULT
 MODEL                          VARCHAR2               IN     DEFAULT
 NETWORK_LINK                   VARCHAR2               IN     DEFAULT
FUNCTION OPENW RETURNS NUMBER
 Argument Name                  Type                   In/Out Default?
 ------------------------------ ---------------------- ------ --------
 OBJECT_TYPE                    VARCHAR2               IN
 VERSION                        VARCHAR2               IN     DEFAULT
 MODEL                          VARCHAR2               IN     DEFAULT
PROCEDURE PATCH_TYPEID
 Argument Name                  Type                   In/Out Default?
 ------------------------------ ---------------------- ------ --------
 SCHEMA                         VARCHAR2               IN
 NAME                           VARCHAR2               IN
 TYPEID                         VARCHAR2               IN
 HASHCODE                       VARCHAR2               IN
FUNCTION PUT RETURNS BOOLEAN
 Argument Name                  Type                   In/Out Default?
 ------------------------------ ---------------------- ------ --------
 HANDLE                         NUMBER                 IN
 DOCUMENT                       XMLTYPE                IN
 FLAGS                          NUMBER                 IN
 RESULTS                        KU$_SUBMITRESULTS      IN/OUT
FUNCTION PUT RETURNS BOOLEAN
 Argument Name                  Type                   In/Out Default?
 ------------------------------ ---------------------- ------ --------
 HANDLE                         NUMBER                 IN
 DOCUMENT                       CLOB                   IN
 FLAGS                          NUMBER                 IN
 RESULTS                        KU$_SUBMITRESULTS      IN/OUT
PROCEDURE SET_COUNT
 Argument Name                  Type                   In/Out Default?
 ------------------------------ ---------------------- ------ --------
 HANDLE                         NUMBER                 IN
 VALUE                          NUMBER                 IN
 OBJECT_TYPE_PATH               VARCHAR2               IN     DEFAULT
PROCEDURE SET_DEBUG
 Argument Name                  Type                   In/Out Default?
 ------------------------------ ---------------------- ------ --------
 ON_OFF                         BOOLEAN                IN
 IP_ADDR                        VARCHAR2               IN     DEFAULT
PROCEDURE SET_FILTER
 Argument Name                  Type                   In/Out Default?
 ------------------------------ ---------------------- ------ --------
 HANDLE                         NUMBER                 IN
 NAME                           VARCHAR2               IN
 VALUE                          VARCHAR2               IN
 OBJECT_TYPE_PATH               VARCHAR2               IN     DEFAULT
PROCEDURE SET_FILTER
 Argument Name                  Type                   In/Out Default?
 ------------------------------ ---------------------- ------ --------
 HANDLE                         NUMBER                 IN
 NAME                           VARCHAR2               IN
 VALUE                          BOOLEAN                IN     DEFAULT
 OBJECT_TYPE_PATH               VARCHAR2               IN     DEFAULT
PROCEDURE SET_FILTER
 Argument Name                  Type                   In/Out Default?
 ------------------------------ ---------------------- ------ --------
 HANDLE                         NUMBER                 IN
 NAME                           VARCHAR2               IN
 VALUE                          NUMBER                 IN
```

```
 OBJECT_TYPE_PATH               VARCHAR2                IN     DEFAULT
PROCEDURE SET_PARSE_ITEM
 Argument Name                  Type                    In/Out Default?
 ------------------------------ ----------------------- ------ --------
 HANDLE                         NUMBER                  IN
 NAME                           VARCHAR2                IN
 OBJECT_TYPE                    VARCHAR2                IN     DEFAULT
PROCEDURE SET_REMAP_PARAM
 Argument Name                  Type                    In/Out Default?
 ------------------------------ ----------------------- ------ --------
 TRANSFORM_HANDLE               NUMBER                  IN
 NAME                           VARCHAR2                IN
 OLD_VALUE                      VARCHAR2                IN
 NEW_VALUE                      VARCHAR2                IN
 OBJECT_TYPE                    VARCHAR2                IN     DEFAULT
PROCEDURE SET_TRANSFORM_PARAM
 Argument Name                  Type                    In/Out Default?
 ------------------------------ ----------------------- ------ --------
 TRANSFORM_HANDLE               NUMBER                  IN
 NAME                           VARCHAR2                IN
 VALUE                          VARCHAR2                IN
 OBJECT_TYPE                    VARCHAR2                IN     DEFAULT
PROCEDURE SET_TRANSFORM_PARAM
 Argument Name                  Type                    In/Out Default?
 ------------------------------ ----------------------- ------ --------
 TRANSFORM_HANDLE               NUMBER                  IN
 NAME                           VARCHAR2                IN
 VALUE                          BOOLEAN                 IN     DEFAULT
 OBJECT_TYPE                    VARCHAR2                IN     DEFAULT
PROCEDURE SET_TRANSFORM_PARAM
 Argument Name                  Type                    In/Out Default?
 ------------------------------ ----------------------- ------ --------
 TRANSFORM_HANDLE               NUMBER                  IN
 NAME                           VARCHAR2                IN
 VALUE                          NUMBER                  IN
 OBJECT_TYPE                    VARCHAR2                IN     DEFAULT
PROCEDURE SET_XMLFORMAT
 Argument Name                  Type                    In/Out Default?
 ------------------------------ ----------------------- ------ --------
 HANDLE                         NUMBER                  IN
 NAME                           VARCHAR2                IN
 VALUE                          BOOLEAN                 IN     DEFAULT

SQL> SELECT DBMS_METADATA.GET_DEPENDENT_DDL('OBJECT_GRANT','EMP',USER) FROM DUAL;

DBMS_METADATA.GET_DEPENDENT_DDL('OBJECT_GRANT','EMP',USER)

--------------------------------------------------------------------------------


  GRANT SELECT ON "SCOTT"."EMP" TO "HR"


  GRANT UPDATE ("ENAME") ON "SCOTT"."EMP" TO "HR"



SQL>
```

```
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL> ED
Wrote file afiedt.buf

  1* SELECT DBMS_METADATA.GET_GRANTED_DDL('SYS_GRANT',USER) FROM DUAL
SQL> /
ERROR:
ORA-31600: invalid input value SYS_GRANT for parameter OBJECT_TYPE in function
GET_GRANTED_DDL
ORA-06512: at "SYS.DBMS_METADATA", line 2681
ORA-06512: at "SYS.DBMS_METADATA", line 2732
ORA-06512: at "SYS.DBMS_METADATA", line 4450
ORA-06512: at line 1



no rows selected

SQL>
SQL>
SQL>
SQL>
SQL>
SQL> ED
Wrote file afiedt.buf

  1* SELECT DBMS_METADATA.GET_GRANTED_DDL('SYSTEM_GRANT',USER) FROM DUAL
SQL> /

DBMS_METADATA.GET_GRANTED_DDL('SYSTEM_GRANT',USER)

--------------------------------------------------------------------------------



  GRANT CREATE JOB TO "SCOTT"


  GRANT DROP ANY DIRECTORY TO "SCOTT"


  GRANT CREATE ANY DIRECTORY TO "SCOTT"


  GRANT CREATE VIEW TO "SCOTT"


  GRANT CREATE TABLE TO "SCOTT"


  GRANT UNLIMITED TABLESPACE TO "SCOTT"
```

```
SQL>
SQL>
SQL>
SQL>
SQL> ED
Wrote file afiedt.buf

  1* SELECT DBMS_METADATA.GET_DDL('TABLE','EMP',USER) FROM DUAL
SQL> /

DBMS_METADATA.GET_DDL('TABLE','EMP',USER)
--------------------------------------------------------------------------------


  CREATE TABLE "SCOTT"."EMP"
   (    "EMPNO" NUMBER(4,0),
        "ENAME" VARCHAR2(10),
        "JOB" VARCHAR2(9),
        "MGR" NUMBER(5,0),
        "HIREDATE" DATE,
        "SAL" NUMBER(7,2),
        "COMM" NUMBER(7,2),
        "DEPTNO" NUMBER(2,0),
         CONSTRAINT "PK_EMP" PRIMARY KEY ("EMPNO")
  USING INDEX PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS
  STORAGE(INITIAL 65536 NEXT 1048576 MINEXTENTS 1 MAXEXTENTS 2147483645
  PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1 BUFFER_POOL DEFAULT)
  TABLESPACE "USERS"  ENABLE,
         CONSTRAINT "FK_DEPTNO" FOREIGN KEY ("DEPTNO")
          REFERENCES "SCOTT"."DEPT" ("DEPTNO") ENABLE
   ) PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255 NOCOMPRESS LOGGING
  STORAGE(INITIAL 65536 NEXT 1048576 MINEXTENTS 1 MAXEXTENTS 2147483645
  PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1 BUFFER_POOL DEFAULT)
  TABLESPACE "USERS"
```

```
SQL> ED
Wrote file afiedt.buf

  1* SELECT DBMS_METADATA.GET_XML('TABLE','EMP',USER) FROM DUAL
SQL> /

DBMS_METADATA.GET_XML('TABLE','EMP',USER)

--------------------------------------------------------------------------------

<?xml version="1.0"?><ROWSET><ROW>

  <TABLE_T>

 <VERS_MAJOR>1</VERS_MAJOR>

 <VERS_MINOR>1 </VERS_MINOR>

 <OBJ_NUM>51151</OBJ_NUM>

 <SCHEMA_OBJ>

  <OBJ_NUM>51151</OBJ_NUM>

  <DATAOBJ_NUM>51151</DATAOBJ_NUM>

  <OWNER_NUM>54</OWNER_NUM>

  <OWNER_NAME>SCOTT</OWNER_NAME>

  <NAME>EMP</NAME>

  <NAMESPACE>1</NAMESPACE>

  <TYPE_NUM>2</TYPE_NUM>

  <TYPE_NAME>TABLE</TYPE_NAME>

  <CTIME>2005-08-30 15:06:10</CTIME>

  <MTIME>2013-03-12 20:06:52</MTIME>

  <STIME>2013-02-02 19:11:21</STIME>

  <STATUS>1</STATUS>

  <FLAGS>0</FLAGS>

  <SPARE1>6</SPARE1>

  <SPARE2>9</SPARE2>

 </SCHEMA_OBJ>

 <STORAGE>

  <FILE_NUM>4</FILE_NUM>

  <BLOCK_NUM>27</BLOCK_NUM>
```

```
 <TYPE_NUM>5</TYPE_NUM>
 <TS_NUM>4</TS_NUM>
 <BLOCKS>8</BLOCKS>
 <EXTENTS>1</EXTENTS>
 <INIEXTS>8</INIEXTS>
 <MINEXTS>1</MINEXTS>
 <MAXEXTS>2147483645</MAXEXTS>
 <EXTSIZE>128</EXTSIZE>
 <EXTPCT>0</EXTPCT>
 <USER_NUM>54</USER_NUM>
 <LISTS>1</LISTS>
 <GROUPS>1</GROUPS>
 <BITMAPRANGES>0</BITMAPRANGES>
 <CACHEHINT>0</CACHEHINT>
 <SCANHINT>0</SCANHINT>
 <HWMINCR>51151</HWMINCR>
 <FLAGS>131329</FLAGS>
</STORAGE>
<TS_NAME>USERS</TS_NAME>
<BLOCKSIZE>8192</BLOCKSIZE>
<DATAOBJ_NUM>51151</DATAOBJ_NUM>
<COLS>8</COLS>
<PCT_FREE>10</PCT_FREE>
<PCT_USED>40</PCT_USED>
<INITRANS>1</INITRANS>
<MAXTRANS>255</MAXTRANS>
<FLAGS>1073742353</FLAGS>
<AUDIT_VAL>--------------------------------------</AUDIT_VAL>
<ROWCNT>20</ROWCNT>
<BLKCNT>5</BLKCNT>
<EMPCNT>0</EMPCNT>
<AVGSPC>0</AVGSPC>
```

<CHNCNT>0</CHNCNT>
<AVGRLN>35</AVGRLN>
<AVGSPC_FLB>0</AVGSPC_FLB>
<FLBCNT>0</FLBCNT>
<ANALYZETIME>02-MAR-13</ANALYZETIME>
<SAMPLESIZE>20</SAMPLESIZE>
<INTCOLS>8</INTCOLS>
<KERNELCOLS>8</KERNELCOLS>
<PROPERTY>536870912</PROPERTY>
<XMLSCHEMACOLS>N</XMLSCHEMACOLS>
<TRIGFLAG>0</TRIGFLAG>
<SPARE1>736</SPARE1>
<SPARE6>02-FEB-13</SPARE6>
<COL_LIST>
 <COL_LIST_ITEM>
  <OBJ_NUM>51151</OBJ_NUM>
  <COL_NUM>1</COL_NUM>
  <INTCOL_NUM>1</INTCOL_NUM>
  <SEGCOL_NUM>1</SEGCOL_NUM>
  <PROPERTY>0</PROPERTY>
  <NAME>EMPNO</NAME>
  <TYPE_NUM>2</TYPE_NUM>
  <LENGTH>22</LENGTH>
  <PRECISION_NUM>4</PRECISION_NUM>
  <SCALE>0</SCALE>
  <NOT_NULL>1</NOT_NULL>
  <CHARSETID>0</CHARSETID>
  <CHARSETFORM>0</CHARSETFORM>
  <SPARE1>0</SPARE1>
  <SPARE2>0</SPARE2>
  <SPARE3>0</SPARE3>

```
  </COL_LIST_ITEM>
  <COL_LIST_ITEM>
   <OBJ_NUM>51151</OBJ_NUM>
   <COL_NUM>2</COL_NUM>
   <INTCOL_NUM>2</INTCOL_NUM>
   <SEGCOL_NUM>2</SEGCOL_NUM>
   <PROPERTY>0</PROPERTY>
   <NAME>ENAME</NAME>
   <TYPE_NUM>1</TYPE_NUM>

DBMS_METADATA.GET_XML('TABLE','EMP',USER)
--------------------------------------------------------------------------------
   <LENGTH>10</LENGTH>
   <NOT_NULL>0</NOT_NULL>
   <CHARSETID>178</CHARSETID>
   <CHARSETFORM>1</CHARSETFORM>
   <SPARE1>0</SPARE1>
   <SPARE2>0</SPARE2>
   <SPARE3>10</SPARE3>
  </COL_LIST_ITEM>
  <COL_LIST_ITEM>
   <OBJ_NUM>51151</OBJ_NUM>
   <COL_NUM>3</COL_NUM>
   <INTCOL_NUM>3</INTCOL_NUM>
   <SEGCOL_NUM>3</SEGCOL_NUM>
   <PROPERTY>0</PROPERTY>
   <NAME>JOB</NAME>
   <TYPE_NUM>1</TYPE_NUM>
   <LENGTH>9</LENGTH>
   <NOT_NULL>0</NOT_NULL>
   <CHARSETID>178</CHARSETID>
   <CHARSETFORM>1</CHARSETFORM>
```

```
 <SPARE1>0</SPARE1>
 <SPARE2>0</SPARE2>
 <SPARE3>9</SPARE3>
</COL_LIST_ITEM>
<COL_LIST_ITEM>
 <OBJ_NUM>51151</OBJ_NUM>
 <COL_NUM>4</COL_NUM>
 <INTCOL_NUM>4</INTCOL_NUM>
 <SEGCOL_NUM>4</SEGCOL_NUM>
 <PROPERTY>0</PROPERTY>
 <NAME>MGR</NAME>
 <TYPE_NUM>2</TYPE_NUM>
 <LENGTH>22</LENGTH>
 <PRECISION_NUM>5</PRECISION_NUM>
 <SCALE>0</SCALE>
 <NOT_NULL>0</NOT_NULL>
 <CHARSETID>0</CHARSETID>
 <CHARSETFORM>0</CHARSETFORM>
 <SPARE1>0</SPARE1>
 <SPARE2>0</SPARE2>
 <SPARE3>0</SPARE3>
</COL_LIST_ITEM>
<COL_LIST_ITEM>
 <OBJ_NUM>51151</OBJ_NUM>
 <COL_NUM>5</COL_NUM>
 <INTCOL_NUM>5</INTCOL_NUM>
 <SEGCOL_NUM>5</SEGCOL_NUM>
 <PROPERTY>0</PROPERTY>
 <NAME>HIREDATE</NAME>
 <TYPE_NUM>12</TYPE_NUM>
 <LENGTH>7</LENGTH>
 <NOT_NULL>0</NOT_NULL>
```

```
 <CHARSETID>0</CHARSETID>
 <CHARSETFORM>0</CHARSETFORM>
 <SPARE1>0</SPARE1>
 <SPARE2>0</SPARE2>
 <SPARE3>0</SPARE3>
</COL_LIST_ITEM>
<COL_LIST_ITEM>
 <OBJ_NUM>51151</OBJ_NUM>
 <COL_NUM>6</COL_NUM>
 <INTCOL_NUM>6</INTCOL_NUM>
 <SEGCOL_NUM>6</SEGCOL_NUM>
 <PROPERTY>0</PROPERTY>
 <NAME>SAL</NAME>
 <TYPE_NUM>2</TYPE_NUM>
 <LENGTH>22</LENGTH>
 <PRECISION_NUM>7</PRECISION_NUM>
 <SCALE>2</SCALE>
 <NOT_NULL>0</NOT_NULL>
 <CHARSETID>0</CHARSETID>
 <CHARSETFORM>0</CHARSETFORM>
 <SPARE1>0</SPARE1>
 <SPARE2>0</SPARE2>
 <SPARE3>0</SPARE3>
</COL_LIST_ITEM>
<COL_LIST_ITEM>
 <OBJ_NUM>51151</OBJ_NUM>
 <COL_NUM>7</COL_NUM>
 <INTCOL_NUM>7</INTCOL_NUM>
 <SEGCOL_NUM>7</SEGCOL_NUM>
 <PROPERTY>0</PROPERTY>
 <NAME>COMM</NAME>
```

```
    <TYPE_NUM>2</TYPE_NUM>
    <LENGTH>22</LENGTH>
    <PRECISION_NUM>7</PRECISION_NUM>
    <SCALE>2</SCALE>
    <NOT_NULL>0</NOT_NULL>
    <CHARSETID>0</CHARSETID>
    <CHARSETFORM>0</CHARSETFORM>
    <SPARE1>0</SPARE1>
    <SPARE2>0</SPARE2>
    <SPARE3>0</SPARE3>
  </COL_LIST_ITEM>
  <COL_LIST_ITEM>
    <OBJ_NUM>51151</OBJ_NUM>
    <COL_NUM>8</COL_NUM>

DBMS_METADATA.GET_XML('TABLE','EMP',USER)
--------------------------------------------------------------------------------
    <INTCOL_NUM>8</INTCOL_NUM>
    <SEGCOL_NUM>8</SEGCOL_NUM>
    <PROPERTY>0</PROPERTY>
    <NAME>DEPTNO</NAME>
    <TYPE_NUM>2</TYPE_NUM>
    <LENGTH>22</LENGTH>
    <PRECISION_NUM>2</PRECISION_NUM>
    <SCALE>0</SCALE>
    <NOT_NULL>0</NOT_NULL>
    <CHARSETID>0</CHARSETID>
    <CHARSETFORM>0</CHARSETFORM>
    <SPARE1>0</SPARE1>
    <SPARE2>0</SPARE2>
    <SPARE3>0</SPARE3>
  </COL_LIST_ITEM>
```

```
</COL_LIST>
<CON0_LIST/>
<CON1_LIST>
 <CON1_LIST_ITEM>
  <OWNER_NUM>54</OWNER_NUM>
  <NAME>PK_EMP</NAME>
  <CON_NUM>5141</CON_NUM>
  <OBJ_NUM>51151</OBJ_NUM>
  <PROPERTY>536870912</PROPERTY>
  <NUMCOLS>1</NUMCOLS>
  <CONTYPE>2</CONTYPE>
  <ENABLED>51152</ENABLED>
  <INTCOLS>1</INTCOLS>
  <MTIME>30-AUG-05</MTIME>
  <FLAGS>4</FLAGS>
  <OID_OR_SETID>0</OID_OR_SETID>
  <COL_LIST>
   <COL_LIST_ITEM>
    <CON_NUM>5141</CON_NUM>
    <OBJ_NUM>51151</OBJ_NUM>
    <INTCOL_NUM>1</INTCOL_NUM>
    <POS_NUM>1</POS_NUM>
    <SPARE1>0</SPARE1>
    <OID_OR_SETID>0</OID_OR_SETID>
    <COL>
     <OBJ_NUM>51151</OBJ_NUM>
     <COL_NUM>1</COL_NUM>
     <INTCOL_NUM>1</INTCOL_NUM>
     <SEGCOL_NUM>1</SEGCOL_NUM>
     <PROPERTY>0</PROPERTY>
     <NAME>EMPNO</NAME>
     <TYPE_NUM>2</TYPE_NUM>
```

```
  </COL>
 </COL_LIST_ITEM>
</COL_LIST>
<IND>
 <VERS_MAJOR>1</VERS_MAJOR>
 <VERS_MINOR>2 </VERS_MINOR>
 <OBJ_NUM>51152</OBJ_NUM>
 <SCHEMA_OBJ>
  <OBJ_NUM>51152</OBJ_NUM>
  <DATAOBJ_NUM>51152</DATAOBJ_NUM>
  <OWNER_NUM>54</OWNER_NUM>
  <OWNER_NAME>SCOTT</OWNER_NAME>
  <NAME>PK_EMP</NAME>
  <NAMESPACE>4</NAMESPACE>
  <TYPE_NUM>1</TYPE_NUM>
  <TYPE_NAME>INDEX</TYPE_NAME>
  <CTIME>2005-08-30 15:06:10</CTIME>
  <MTIME>2005-08-30 15:06:10</MTIME>
  <STIME>2005-08-30 15:06:10</STIME>
  <STATUS>1</STATUS>
  <FLAGS>0</FLAGS>
  <SPARE1>0</SPARE1>
  <SPARE2>65535</SPARE2>
 </SCHEMA_OBJ>
 <COL_LIST>
  <COL_LIST_ITEM>
   <OBJ_NUM>51152</OBJ_NUM>
   <BO_NUM>51151</BO_NUM>
   <INTCOL_NUM>1</INTCOL_NUM>
   <COL>
    <OBJ_NUM>51151</OBJ_NUM>
```

```
        <COL_NUM>1</COL_NUM>
        <INTCOL_NUM>1</INTCOL_NUM>
        <SEGCOL_NUM>1</SEGCOL_NUM>
        <PROPERTY>0</PROPERTY>
        <NAME>EMPNO</NAME>
        <TYPE_NUM>2</TYPE_NUM>
       </COL>
       <POS_NUM>1</POS_NUM>
       <SEGCOL_NUM>0</SEGCOL_NUM>
       <SEGCOLLEN>0</SEGCOLLEN>
       <OFFSET>0</OFFSET>
       <FLAGS>0</FLAGS>
       <SPARE2>0</SPARE2>
       <OID_OR_SETID>0</OID_OR_SETID>
      </COL_LIST_ITEM>
     </COL_LIST>
     <TS_NAME>USERS</TS_NAME>
     <BLOCKSIZE>8192</BLOCKSIZE>
     <STORAGE>

DBMS_METADATA.GET_XML('TABLE','EMP',USER)
--------------------------------------------------------------------------------

      <FILE_NUM>4</FILE_NUM>
      <BLOCK_NUM>35</BLOCK_NUM>
      <TYPE_NUM>6</TYPE_NUM>
      <TS_NUM>4</TS_NUM>
      <BLOCKS>8</BLOCKS>
      <EXTENTS>1</EXTENTS>
      <INIEXTS>8</INIEXTS>
      <MINEXTS>1</MINEXTS>
      <MAXEXTS>2147483645</MAXEXTS>
      <EXTSIZE>128</EXTSIZE>
```

```
 <EXTPCT>0</EXTPCT>
 <USER_NUM>54</USER_NUM>
 <LISTS>1</LISTS>
 <GROUPS>1</GROUPS>
 <BITMAPRANGES>0</BITMAPRANGES>
 <CACHEHINT>0</CACHEHINT>
 <SCANHINT>0</SCANHINT>
 <HWMINCR>51152</HWMINCR>
 <FLAGS>131329</FLAGS>
</STORAGE>
<DATAOBJ_NUM>51152</DATAOBJ_NUM>
<BASE_OBJ_NUM>51151</BASE_OBJ_NUM>
<BASE_OBJ>
 <OBJ_NUM>51151</OBJ_NUM>
 <DATAOBJ_NUM>51151</DATAOBJ_NUM>
 <OWNER_NUM>54</OWNER_NUM>
 <OWNER_NAME>SCOTT</OWNER_NAME>
 <NAME>EMP</NAME>
 <NAMESPACE>1</NAMESPACE>
 <TYPE_NUM>2</TYPE_NUM>
 <TYPE_NAME>TABLE</TYPE_NAME>
 <CTIME>2005-08-30 15:06:10</CTIME>
 <MTIME>2013-03-12 20:06:52</MTIME>
 <STIME>2013-02-02 19:11:21</STIME>
 <STATUS>1</STATUS>
 <FLAGS>0</FLAGS>
 <SPARE1>6</SPARE1>
 <SPARE2>9</SPARE2>
</BASE_OBJ>
<INDMETHOD_NUM>0</INDMETHOD_NUM>
<COLS>1</COLS>
<PCT_FREE>10</PCT_FREE>
```

```
    <INITRANS>2</INITRANS>
    <MAXTRANS>255</MAXTRANS>
    <TYPE_NUM>1</TYPE_NUM>
    <FLAGS>2050</FLAGS>
    <PROPERTY>4097</PROPERTY>
    <BLEVEL>0</BLEVEL>
    <LEAFCNT>1</LEAFCNT>
    <DISTKEY>20</DISTKEY>
    <LBLKKEY>1</LBLKKEY>
    <DBLKKEY>1</DBLKKEY>
    <CLUFAC>1</CLUFAC>
    <ANALYZETIME>02-MAR-13</ANALYZETIME>
    <SAMPLESIZE>20</SAMPLESIZE>
    <ROWCNT>20</ROWCNT>
    <INTCOLS>1</INTCOLS>
    <NUMCOLSDEP>1</NUMCOLSDEP>
    <SPARE6>30-AUG-05</SPARE6>
    <FOR_PKOID>0</FOR_PKOID>
    <OID_OR_SETID>0</OID_OR_SETID>
   </IND>
  </CON1_LIST_ITEM>
</CON1_LIST>
<CON2_LIST>
 <CON2_LIST_ITEM>
  <OWNER_NUM>54</OWNER_NUM>
  <NAME>FK_DEPTNO</NAME>
  <CON_NUM>5142</CON_NUM>
  <OBJ_NUM>51151</OBJ_NUM>
  <NUMCOLS>1</NUMCOLS>
  <CONTYPE>4</CONTYPE>
  <ROBJ_NUM>51149</ROBJ_NUM>
```

```
   <RCON_NUM>5140</RCON_NUM>
   <ENABLED>1</ENABLED>
   <INTCOLS>1</INTCOLS>
   <MTIME>30-AUG-05</MTIME>
   <FLAGS>4</FLAGS>
   <SCHEMA_OBJ>
    <OBJ_NUM>51149</OBJ_NUM>
    <DATAOBJ_NUM>51149</DATAOBJ_NUM>
    <OWNER_NUM>54</OWNER_NUM>
    <OWNER_NAME>SCOTT</OWNER_NAME>
    <NAME>DEPT</NAME>
    <NAMESPACE>1</NAMESPACE>
    <TYPE_NUM>2</TYPE_NUM>
    <TYPE_NAME>TABLE</TYPE_NAME>
    <CTIME>2005-08-30 15:06:10</CTIME>
    <MTIME>2005-08-30 15:06:10</MTIME>
    <STIME>2005-08-30 15:06:10</STIME>
    <STATUS>1</STATUS>
    <FLAGS>0</FLAGS>
    <SPARE1>6</SPARE1>
    <SPARE2>1</SPARE2>
   <


SQL> ED
Wrote file afiedt.buf

  1* SELECT DBMS_METADATA.GET_DDL('TABLE','EMP',USER) FROM DUAL
SQL> /

DBMS_METADATA.GET_DDL('TABLE','EMP',USER)

--------------------------------------------------------------------------------


  CREATE TABLE "SCOTT"."EMP"
   (    "EMPNO" NUMBER(4,0),
```

```
        "ENAME" VARCHAR2(10),
        "JOB" VARCHAR2(9),
        "MGR" NUMBER(5,0),
        "HIREDATE" DATE,
        "SAL" NUMBER(7,2),
        "COMM" NUMBER(7,2),
        "DEPTNO" NUMBER(2,0),
         CONSTRAINT "PK_EMP" PRIMARY KEY ("EMPNO")
  USING INDEX PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS
  STORAGE(INITIAL 65536 NEXT 1048576 MINEXTENTS 1 MAXEXTENTS 2147483645
  PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1 BUFFER_POOL DEFAULT)
  TABLESPACE "USERS"  ENABLE,
         CONSTRAINT "FK_DEPTNO" FOREIGN KEY ("DEPTNO")
          REFERENCES "SCOTT"."DEPT" ("DEPTNO") ENABLE
   ) PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255 NOCOMPRESS LOGGING
  STORAGE(INITIAL 65536 NEXT 1048576 MINEXTENTS 1 MAXEXTENTS 2147483645
  PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1 BUFFER_POOL DEFAULT)
  TABLESPACE "USERS"


SQL> SPOOL OFF
```

```
SQL>
SQL>
SQL> STARTUP FORCE
ORACLE instance started.

Total System Global Area  612368384 bytes

Fixed Size                  1250428 bytes

Variable Size             167775108 bytes

Database Buffers          436207616 bytes

Redo Buffers                7135232 bytes

Database mounted.
Database opened.
SQL> SHOW USER
USER is "SYS"
SQL> CONN SCOTT/TIGER
Connected.
USER is "SCOTT"
linesize 100
pagesize 100
long 80
SQL> SELECT OBJECT_NAME FROM USER_OBJECTS
  2  WHERE OBJECT_TYPE='PROCEDURE';

OBJECT_NAME

--------------------------------------------------------------------------------
----------------
ADD_EMP

ADD_NEW_EMP

EMP_POSTING

DEL_REC

SHOW_TXT

WRITE_TO_FILE

GET_FILE_TXT

TEST_JOB

DO_EXE_IMM

T1

CREATE_TABLE

SHOW_REC

ADD_DEPT

ADD_R

SET_VDO

GET_EMP_VDO_LEN
```

```
LOAD_TXT_DATA

CHK_SAL

TAB_NO

MY_CODE

TEST2

SHOW_TEXT

ADD_REC

SHOW_DATA


24 rows selected.

SQL> SELECT * FROM EMP
  2  WHERE EMPNO=8000;

no rows selected

SQL>
SQL>
SQL> /

     EMPNO ENAME      JOB               MGR HIREDATE          SAL        COMM
DEPTNO
---------- ---------- --------- ---------- --------- ---------- ----------
----------
      8000 SCOTT      SALESMAN                               1000
30

SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL> ED
Wrote file afiedt.buf

  1  SELECT * FROM EMP
  2* WHERE EMPNO=8001
SQL> /

     EMPNO ENAME      JOB               MGR HIREDATE          SAL        COMM
DEPTNO
---------- ---------- --------- ---------- --------- ---------- ----------
----------
      8001 SCOTT      SALESMAN                               1000
35

SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
```

```
SQL>
SQL> ED
Wrote file afiedt.buf

  1  CREATE OR REPLACE PROCEDURE ADD_DEPT(DEPT_ID NUMBER) IS
  2  PRAGMA AUTONOMOUS_TRANSACTION;
  3  BEGIN
  4  INSERT INTO DEPT(DEPTNO,DNAME)
  5  VALUES(DEPT_ID,'UPDATE REQ');
  6  COMMIT;
  7*   END;
SQL> /

Procedure created.

SQL>
SQL>
SQL>
SQL>
SQL>
SQL> ED
Wrote file afiedt.buf

  1  CREATE OR REPLACE PROCEDURE ADD_EMP1
  2     (
  3         V_EMPNO EMP.EMPNO%TYPE,
  4         V_ENAME EMP.ENAME%TYPE,
  5         V_JOB   EMP.JOB%TYPE,
  6         V_SAL   EMP.SAL%TYPE,
  7         V_DEPTNO EMP.DEPTNO%TYPE
  8     ) IS
  9    MASTER_NOT_FOUND EXCEPTION;
 10    PRAGMA EXCEPTION_INIT(MASTER_NOT_FOUND,-2291);
 11        BEGIN
 12        INSERT INTO EMP(EMPNO,ENAME,JOB,SAL,DEPTNO)
 13         VALUES(V_EMPNO,V_ENAME,V_JOB,V_SAL,V_DEPTNO);
 14         COMMIT;
 15        EXCEPTION
 16          WHEN MASTER_NOT_FOUND THEN
 17  ADD_DEPT(V_DEPTNO); ----CALLING PROCEDURE----
 18        INSERT INTO EMP(EMPNO,ENAME,JOB,SAL,DEPTNO)
 19         VALUES(V_EMPNO,V_ENAME,V_JOB,V_SAL,V_DEPTNO);
 20         COMMIT;
 21*     END;
 22  /

Procedure created.

SQL>
SQL>
SQL>
SQL>
SQL> ED
Wrote file afiedt.buf

  1  DECLARE
  2      TYPE DEPT_TAB IS TABLE OF SCOTT.DEPT%ROWTYPE;
  3      DEPTS DEPT_TAB;
  4      BEGIN
  5      SELECT * BULK COLLECT INTO DEPTS FROM DEPT;
  6      FOR I IN 1..DEPTS.COUNT LOOP
  7      &D(I||'   '||DEPTS(I).DEPTNO||' '||DEPTS(I).DNAME||'   '||DEPTS(I).LOC);
  8      END LOOP;
```

```
  9*    END;
SQL> /
1   41 UPDATE_REQ

2   99 OTHERS

3   35 UPDATE REQ

4   50 HR   KARACHI

5   60 NEW HR   LHR

6   10 ACCOUNTING   NEW YORK

7   20 RESEARCH   DALLAS

8   30 SALES   CHICAGO

9   40 OPERATIONS   BOSTON


PL/SQL procedure successfully completed.

SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL> ED
Wrote file afiedt.buf

  1  DECLARE
  2      TYPE DEPT_TAB IS TABLE OF SCOTT.DEPT%ROWTYPE;
  3      DEPTS DEPT_TAB;
  4      BEGIN
  5      SELECT * BULK COLLECT INTO DEPTS FROM DEPT;
  6      FOR I IN DEPTS.FIRST..DEPTS.LAST LOOP
  7      &D(I||'   '||DEPTS(I).DEPTNO||' '||DEPTS(I).DNAME||'   '||DEPTS(I).LOC);
  8      END LOOP;
  9*    END;
SQL> /
1   41 UPDATE_REQ

2   99 OTHERS

3   35 UPDATE REQ

4   50 HR   KARACHI

5   60 NEW HR   LHR

6   10 ACCOUNTING   NEW YORK

7   20 RESEARCH   DALLAS

8   30 SALES   CHICAGO

9   40 OPERATIONS   BOSTON


PL/SQL procedure successfully completed.
```

```
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL> ED
Wrote file afiedt.buf

  1  DECLARE
  2      TYPE DEPT_TAB IS TABLE OF SCOTT.DEPT%ROWTYPE;
  3      DEPTS DEPT_TAB;
  4      BEGIN
  5      SELECT * BULK COLLECT INTO DEPTS FROM DEPT;
  6     &D('TOTAL RECORDS FOUND.....'||DEPTS.COUNT);
  7      FOR I IN DEPTS.FIRST..DEPTS.LAST LOOP
  8      &D(I||'   '||DEPTS(I).DEPTNO||' '||DEPTS(I).DNAME||'   '||DEPTS(I).LOC);
  9      END LOOP;
 10*    END;
 11  /
TOTAL RECORDS FOUND.....9

1   41 UPDATE_REQ

2   99 OTHERS

3   35 UPDATE REQ

4   50 HR   KARACHI

5   60 NEW HR   LHR

6   10 ACCOUNTING   NEW YORK

7   20 RESEARCH   DALLAS

8   30 SALES   CHICAGO

9   40 OPERATIONS   BOSTON


PL/SQL procedure successfully completed.

SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL> ED
Wrote file afiedt.buf

  1   DECLARE
  2      CURSOR DEPT_CSR(DID IN NUMBER) IS
  3       SELECT * FROM DEPT NATURAL JOIN EMP WHERE DEPTNO=DID;
```

```
  4      TYPE DE_INFO IS TABLE OF DEPT_CSR%ROWTYPE;
  5      DEI DE_INFO;
  6      BEGIN
  7      OPEN DEPT_CSR(&DEPARMENT_ID);
  8      FETCH DEPT_CSR BULK COLLECT INTO DEI;
  9      CLOSE DEPT_CSR;
 10     FOR I IN 1..DEI.COUNT LOOP
 11      &D(DEI(I).EMPNO||'    '||DEI(I).ENAME);
 12      END LOOP;
 13*   END;
 14   /
Enter value for deparment_id: 30
8000    SCOTT

7499    ALLEN

7521    WARD

7654    MARTIN

7698    BLAKE

7844    TURNER

7900    JAMES


PL/SQL procedure successfully completed.

SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL> /*
SQL>          INSERT INTO EMP
SQL>          SELECT * FORM EMP;
SQL>
SQL> */
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL> SPOOL OFF
```

```
SQL>
SQL>
SQL> SELECT .
  2  ED
  3  .
SQL> ED
Wrote file afiedt.buf

  1* SELECT TO_CHAR(TO_DATE(&ANY_NO,'J'),'JSP') FROM DUAL
SQL> /
Enter value for any_no: 54

TO_CHAR(TO

----------

FIFTY-FOUR


SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL> /
Enter value for any_no: 543

TO_CHAR(TO_DATE(543,'J')

------------------------

FIVE HUNDRED FORTY-THREE


SQL> /
Enter value for any_no: 34345345345435
SELECT TO_CHAR(TO_DATE(34345345345435,'J'),'JSP') FROM DUAL
                       *
ERROR at line 1:
ORA-01830: date format picture ends before converting entire input string


SQL> /
Enter value for any_no: 5.5
SELECT TO_CHAR(TO_DATE(5.5,'J'),'JSP') FROM DUAL
                       *
ERROR at line 1:
ORA-01830: date format picture ends before converting entire input string


SQL> /
Enter value for any_no: 5

TO_C

----

FIVE


SQL> @ D:\BATCH_63\PLSQL\PL_CLASS_20_15062011\DH_UTIL.LIB
```

```
Package created.


Package body created.

SQL>
SQL>
SQL>
SQL> DESC DH_UTIL
FUNCTION CHECK_PROTECT RETURNS VARCHAR2
 Argument Name                  Type                    In/Out Default?
 ------------------------------ ----------------------- ------ --------
 X                              NUMBER                  IN
FUNCTION SPELL RETURNS VARCHAR2
 Argument Name                  Type                    In/Out Default?
 ------------------------------ ----------------------- ------ --------
 X                              NUMBER                  IN

SQL>
SQL>
SQL>
SQL>
SQL>
SQL> SELECT DH_UTIL.SPELL(3345.5) FROM DUAL;

DH_UTIL.SPELL(3345.5)

--------------------------------------------------------------------------------
----------------
Three Thousand Three Hundred Forty-Five and Five / Tenths


SQL>
SQL>
SQL>
SQL>
SQL>
SQL> ED
Wrote file afiedt.buf

  1* SELECT DH_UTIL.SPELL(33453453534534534534534534535345345443) FROM DUAL
SQL> /

DH_UTIL.SPELL(33453453534534534534534534535345345443)

--------------------------------------------------------------------------------
----------------
Three Hundred Thirty-Four Decillion Five Hundred Thirty-Four Nonillion Five Hundred
Thirty-Five Octi
llion Three Hundred Forty-Five Septillion Three Hundred Forty-Five Sextillion Three
Hundred Forty-Fi
ve Quintillion Three Hundred Forty-Five Quadrillion Three Hundred Forty-Five
Trillion Four Hundred T
hirty-Five Billion Three Hundred Forty-Five Million Three Hundred Forty-Five
Thousand Four Hundred F
orty-Three



SQL>
SQL>
SQL>
```

```
SQL>
SQL>
SQL>
SQL>
SQL>
SQL> ED
Wrote file afiedt.buf

  1* SELECT
DH_UTIL.SPELL(33453534534534543534534543453453534534534534534543534534544 3) FROM
DUAL
SQL> /

DH_UTIL.SPELL(33453534534534543534534543453453534534534534534543534534544 3)

--------------------------------------------------------------------------------
----------------
Three Hundred Thirty-Four Octodecillion Five Hundred Thirty-Five Septendecillion
Three Hundred Forty
-Five Sexdecillion Three Hundred Forty-Five Quindecillion Three Hundred Forty-Five
Quattuordecillion
 Four Hundred Thirty-Five Tredecillion Three Hundred Forty-Five Duodecillion Three
Hundred Forty-Fiv
e Undecillion Four Hundred Thirty-Four Decillion Five Hundred Thirty-Four Nonillion
Five Hundred Thi
rty-Five Octillion Three Hundred Forty-Five Septillion Three Hundred Forty-Five
Sextillion Three Hun
dred Quintillion




SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL> ED
Wrote file afiedt.buf

  1  ------------------BULK_COLLECT_FORALL-----------------------
  2      DECLARE
  3           TYPE NUMLIST IS TABLE OF NUMBER;
  4           DEPTS NUMLIST := NUMLIST(10,20,30);
  5          TYPE ENUM_T IS TABLE OF EMP.EMPNO%TYPE;
  6              E_IDS ENUM_T;
  7              TYPE DEPT_T IS TABLE OF EMP.DEPTNO%TYPE;
  8              D_IDS DEPT_T;
  9              BEGIN
 10            FORALL J IN DEPTS.FIRST..DEPTS.LAST
 11          DELETE FROM EMP_TEMP WHERE DEPTNO=DEPTS(J)
 12          RETURNING EMPNO,DEPTNO BULK COLLECT INTO E_IDS,D_IDS;
 13           &D('DELETED #'||SQL%ROWCOUNT||' ROWS :');
 14           FOR I IN E_IDS.FIRST..E_IDS.LAST LOOP
 15           &D('EMPLOYEE #'||E_IDS(I)||' FROM DEPTNO '||D_IDS(I));
 16          END LOOP;
 17*      END;
SQL> /
DELETED #22 ROWS :

EMPLOYEE #7782 FROM DEPTNO 10
```

```
EMPLOYEE #7839 FROM DEPTNO 10

EMPLOYEE #7934 FROM DEPTNO 10

EMPLOYEE #7369 FROM DEPTNO 20

EMPLOYEE #7566 FROM DEPTNO 20

EMPLOYEE #7788 FROM DEPTNO 20

EMPLOYEE #7876 FROM DEPTNO 20

EMPLOYEE #7902 FROM DEPTNO 20

EMPLOYEE #7937 FROM DEPTNO 30

EMPLOYEE #7938 FROM DEPTNO 30

EMPLOYEE #7939 FROM DEPTNO 30

EMPLOYEE #7935 FROM DEPTNO 30

EMPLOYEE #7936 FROM DEPTNO 30

EMPLOYEE #7940 FROM DEPTNO 30

EMPLOYEE #7941 FROM DEPTNO 30

EMPLOYEE #1000 FROM DEPTNO 30

EMPLOYEE #7499 FROM DEPTNO 30

EMPLOYEE #7521 FROM DEPTNO 30

EMPLOYEE #7654 FROM DEPTNO 30

EMPLOYEE #7698 FROM DEPTNO 30

EMPLOYEE #7844 FROM DEPTNO 30

EMPLOYEE #7900 FROM DEPTNO 30


PL/SQL procedure successfully completed.

SQL>
SQL>
SQL> ED
Wrote file afiedt.buf

  1   DECLARE
  2                TYPE EmpTabTyp IS TABLE OF emp%ROWTYPE;
  3                emp_tab EmpTabTyp := EmpTabTyp(NULL);  -- initialize
  4           t1 NUMBER;
  5           t2 NUMBER;
  6           t3 NUMBER;
  7            PROCEDURE get_time (t OUT NUMBER) IS
  8            BEGIN
  9                t := DBMS_UTILITY.get_time;
 10              END;
 11  PROCEDURE do_nothing1
 12  (tab IN OUT EmpTabTyp) IS --RETURN VALUES NOT NAME OF PARAMETER
```

```
 13           BEGIN
 14           NULL;
 15             END;
 16           PROCEDURE do_nothing2
 17  (tab IN OUT NOCOPY EmpTabTyp) IS ----RETURN NAME OF PARAMETER
 18           BEGIN
 19            NULL;
 20              END;
 21          BEGIN
 22             SELECT * INTO emp_tab(1) FROM emp WHERE empno = 7839;
 23              emp_tab.EXTEND(49999, 1);  -- copy element 1 into 2..50000
 24                get_time(t1);
 25            &D('BEFORE CALLING DO_NOTHING1 '||T1);
 26              do_nothing1(emp_tab);  -- pass IN OUT parameter
 27              get_time(t2);
 28              &D('BEFORE CALLING DO_NOTHING2 '||T2);
 29              do_nothing2(emp_tab);  -- pass IN OUT NOCOPY parameter
 30            get_time(t3);
 31           &D('BEFORE FINALIZE THE CALCULATION '||T3);
 32             DBMS_OUTPUT.PUT_LINE('Call Duration (secs)');
 33             DBMS_OUTPUT.PUT_LINE('--------------------');
 34      DBMS_OUTPUT.PUT_LINE('Just IN OUT: ' || TO_CHAR((t2 - t1)/100.0));
 35      DBMS_OUTPUT.PUT_LINE('With NOCOPY: ' || TO_CHAR((t3 - t2))/100.0);
 36*     END;
 37  /
BEFORE CALLING DO_NOTHING1 476167

BEFORE CALLING DO_NOTHING2 476173

BEFORE FINALIZE THE CALCULATION 476173

Call Duration (secs)

--------------------

Just IN OUT: .06

With NOCOPY: 0


PL/SQL procedure successfully completed.

SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL> /
BEFORE CALLING DO_NOTHING1 480685

BEFORE CALLING DO_NOTHING2 480693

BEFORE FINALIZE THE CALCULATION 480693

Call Duration (secs)
```

```
--------------------

Just IN OUT: .08

With NOCOPY: 0


PL/SQL procedure successfully completed.

SQL>
SQL>
SQL>
SQL>
SQL>
SQL> CL SCR
SQL> CREATE OR REPLACE VIEW EMP_INFO AS
  2  SELECT * FROM EMP;

View created.

SQL> SELECT * FROM EMP_INFO;

     EMPNO ENAME      JOB               MGR HIREDATE          SAL        COMM
DEPTNO
---------- ---------- --------- ---------- --------- ---------- ----------
----------
      8000 SCOTT      SALESMAN                               1000
30
      8001 SCOTT      SALESMAN                               1000
35
      5454 SMITH      CLERK             7902 17-DEC-80        900
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       1000
30
      7902 FORD       ANALYST           7566 03-DEC-81      45666
20
      7934 MILLER     CLERK             7782 23-JAN-85       1300
10

16 rows selected.

SQL> CREATE OR REPLACE SAL_INFO AS SELECT * FROM EMP_INFO WHERE SAL>1;
CREATE OR REPLACE SAL_INFO AS SELECT * FROM EMP_INFO WHERE SAL>1
```

```
                *
ERROR at line 1:
ORA-00922: missing or invalid option


SQL> ED
Wrote file afiedt.buf

  1* CREATE OR REPLACE VIEW SAL_INFO AS SELECT * FROM EMP_INFO WHERE SAL>1
SQL> /

View created.

SQL>
SQL>
SQL>
SQL>
SQL> CREATE OR REPLACE VIEW MGR_INFO AS SELECT * FROM
  2  SAL_INFO WHERE JOB='MANAGER';

View created.

SQL> SELECT * FORM MGR_INFO;
SELECT * FORM MGR_INFO
       *
ERROR at line 1:
ORA-00923: FROM keyword not found where expected


SQL>
SQL>
SQL> ED
Wrote file afiedt.buf

  1* SELECT * FROM MGR_INFO
SQL> /

     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

SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL> ED
Wrote file afiedt.buf

  1  CREATE OR REPLACE FUNCTION GET_ID RETURN NUMBER IS
  2  ID NUMBER :=0;
```

```
  3  BEGIN
  4  SELECT MAX(EMPNO)+1 INTO ID FROM SCOTT.EMP_INFO;
  5  RETURN(ID);
  6* END;
  7  /

Function created.

SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL> ED
Wrote file afiedt.buf

  1  CREATE OR REPLACE PROCEDURE A_EMP IS
  2  ID NUMBER :=0;
  3  BEGIN
  4  ID := GET_ID;  ---CALLING FUNCTION
  5  INSERT INTO EMP(EMPNO,ENAME,JOB,SAL,DEPTNO)
  6  VALUES(ID,USER,'SALESMAN',1000,30);
  7  COMMIT;
  8* END;
SQL> /.

Procedure created.

SQL> EXEC A_EMP;

PL/SQL procedure successfully completed.

SQL>
SQL>
SQL> SELECT * FROM EMP;

     EMPNO ENAME      JOB              MGR HIREDATE          SAL        COMM
DEPTNO
---------- ---------- --------- ---------- --------- ---------- ----------
----------
      8000 SCOTT      SALESMAN                           1000
30
      8001 SCOTT      SALESMAN                           1000
35
      8002 SCOTT      SALESMAN                           1000
30
      5454 SMITH      CLERK           7902 17-DEC-80        900
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       1000
30
      7902 FORD       ANALYST         7566 03-DEC-81      45666
20
      7934 MILLER     CLERK           7782 23-JAN-85       1300
10

17 rows selected.

SQL> EXEC A_EMP;

PL/SQL procedure successfully completed.

SQL> SELECT * FROM EMP;

     EMPNO ENAME      JOB              MGR HIREDATE          SAL        COMM
DEPTNO
---------- ---------- --------- ---------- --------- ---------- ----------
----------
      8000 SCOTT      SALESMAN                             1000
30
      8001 SCOTT      SALESMAN                             1000
35
      8002 SCOTT      SALESMAN                             1000
30
      8003 SCOTT      SALESMAN                             1000
30
      5454 SMITH      CLERK           7902 17-DEC-80        900
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       1000
30
      7902 FORD       ANALYST         7566 03-DEC-81      45666
20
      7934 MILLER     CLERK           7782 23-JAN-85       1300
10

18 rows selected.
```

```
SQL> SELECT OBJECT_NAME,OBJECT_TYPE,STATUS FROM USER_OBJECTS
  2  WHERE STATUS='VALID';

OBJECT_NAME

--------------------------------------------------------------------------------
----------------
OBJECT_TYPE         STATUS

------------------- -------

PK_DEPT

INDEX               VALID


DEPT

TABLE               VALID


EMP

TABLE               VALID


PK_EMP

INDEX               VALID


BONUS

TABLE               VALID


SALGRADE

TABLE               VALID


ADD_EMP

PROCEDURE           VALID


GET_MGR

FUNCTION            VALID


EMP_TEST

TABLE               VALID
```

EMP_HIST

TABLE VALID

P1

PACKAGE VALID

P1

PACKAGE BODY VALID

OVERPACK

PACKAGE VALID

OVERPACK

PACKAGE BODY VALID

BODYLESS_PACK

PACKAGE VALID

SHOW_TXT

PROCEDURE VALID

WRITE_TO_FILE

PROCEDURE VALID

GET_FILE_TXT

PROCEDURE VALID

DO_EXE_IMM

PROCEDURE VALID

T1

PROCEDURE VALID

CREATE_TABLE

PROCEDURE VALID

TEST

TABLE VALID

SHOW_REC

PROCEDURE VALID

LOG_EMP_HIST

TABLE VALID

EMP_VIEW

VIEW VALID

ADD_DEPT

PROCEDURE VALID

EMP_TEMP

TABLE VALID

GET_ID

FUNCTION VALID

EIMAGE

TABLE VALID

SET_VDO

PROCEDURE VALID

```
GET_EMP_VDO_LEN
PROCEDURE            VALID


EMP_RESUME
TABLE                VALID

OBJECT_NAME
--------------------------------------------------------------------------------
----------------
OBJECT_TYPE          STATUS
------------------- -------


SYS_LOB0000052750C00002$$
LOB                  VALID

LOAD_TXT_DATA
PROCEDURE            VALID

EMP_INFO
VIEW                 VALID

CHK_SAL
PROCEDURE            VALID

EMP_AUDIT
TABLE                VALID

GET_WORDS
FUNCTION             VALID

S1
SEQUENCE             VALID
```

GET_TAX

FUNCTION VALID

EMP_COPY

TABLE VALID

JOB_IDS

TABLE VALID

STD

TABLE VALID

TEST1

TABLE VALID

EMP_EXCEPTION

TABLE VALID

TAB_NO

PROCEDURE VALID

EMP_BACKUP

TABLE VALID

EMP_ATTEND

TABLE VALID

T

TABLE VALID

MY_CODE

PROCEDURE VALID

SQ1

SEQUENCE VALID

CHECK1

TABLE VALID

GET_TABLE

FUNCTION VALID

VALID_SAL

FUNCTION VALID

NEW_REC

FUNCTION VALID

SYS_LOB0000053362C00002$$

LOB VALID

MY_PACK

PACKAGE BODY VALID

MY_PACK

PACKAGE VALID

V1

VIEW VALID

SHOW_TEXT

PROCEDURE VALID

```
ADD_REC
PROCEDURE           VALID

BIN$sjH4KpabQO2lmlGnRm1aLw==$0
TABLE               VALID

BIN$6BvplJl2TnSiUg1ZurRz2g==$0
INDEX               VALID

SAL_INFO

OBJECT_NAME
--------------------------------------------------------------------------------
----------------
OBJECT_TYPE         STATUS
------------------- -------
VIEW                VALID

MGR_INFO
VIEW                VALID

A_EMP
PROCEDURE           VALID

STUDENT
TABLE               VALID

SHOW_DATA
PROCEDURE           VALID

ADD_EMP1
PROCEDURE           VALID
```

```
HIGH

PROCEDURE            VALID


DH_UTIL

PACKAGE              VALID


DH_UTIL

PACKAGE BODY         VALID



72 rows selected.

SQL> COL OBJECT_NAME FORMAT A20
SQL> .
SQL> .
SQL> /

OBJECT_NAME          OBJECT_TYPE          STATUS

-------------------- -------------------- -------

PK_DEPT              INDEX                VALID

DEPT                 TABLE                VALID

EMP                  TABLE                VALID

PK_EMP               INDEX                VALID

BONUS                TABLE                VALID

SALGRADE             TABLE                VALID

ADD_EMP              PROCEDURE            VALID

GET_MGR              FUNCTION             VALID

EMP_TEST             TABLE                VALID

EMP_HIST             TABLE                VALID

P1                   PACKAGE              VALID

P1                   PACKAGE BODY         VALID

OVERPACK             PACKAGE              VALID

OVERPACK             PACKAGE BODY         VALID

BODYLESS_PACK        PACKAGE              VALID
```

SHOW_TXT	PROCEDURE	VALID
WRITE_TO_FILE	PROCEDURE	VALID
GET_FILE_TXT	PROCEDURE	VALID
DO_EXE_IMM	PROCEDURE	VALID
T1	PROCEDURE	VALID
CREATE_TABLE	PROCEDURE	VALID
TEST	TABLE	VALID
SHOW_REC	PROCEDURE	VALID
LOG_EMP_HIST	TABLE	VALID
EMP_VIEW	VIEW	VALID
ADD_DEPT	PROCEDURE	VALID
EMP_TEMP	TABLE	VALID
GET_ID	FUNCTION	VALID
EIMAGE	TABLE	VALID
SET_VDO	PROCEDURE	VALID
GET_EMP_VDO_LEN	PROCEDURE	VALID
EMP_RESUME	TABLE	VALID
SYS_LOB0000052750C00 002$$	LOB	VALID
LOAD_TXT_DATA	PROCEDURE	VALID
EMP_INFO	VIEW	VALID
CHK_SAL	PROCEDURE	VALID
EMP_AUDIT	TABLE	VALID
GET_WORDS	FUNCTION	VALID
S1	SEQUENCE	VALID
GET_TAX	FUNCTION	VALID
EMP_COPY	TABLE	VALID
JOB_IDS	TABLE	VALID
STD	TABLE	VALID
TEST1	TABLE	VALID
EMP_EXCEPTION	TABLE	VALID

TAB_NO	PROCEDURE	VALID
EMP_BACKUP	TABLE	VALID
EMP_ATTEND	TABLE	VALID
T	TABLE	VALID
MY_CODE	PROCEDURE	VALID
SQ1	SEQUENCE	VALID
CHECK1	TABLE	VALID
GET_TABLE	FUNCTION	VALID
VALID_SAL	FUNCTION	VALID
NEW_REC	FUNCTION	VALID
SYS_LOB0000053362C00 002$$	LOB	VALID
MY_PACK	PACKAGE BODY	VALID
MY_PACK	PACKAGE	VALID
V1	VIEW	VALID
SHOW_TEXT	PROCEDURE	VALID
ADD_REC	PROCEDURE	VALID
BIN$sjH4KpabQO2lmlGn Rm1aLw==$0	TABLE	VALID
BIN$6BvplJl2TnSiUg1Z urRz2g==$0	INDEX	VALID
SAL_INFO	VIEW	VALID
MGR_INFO	VIEW	VALID
A_EMP	PROCEDURE	VALID
STUDENT	TABLE	VALID
SHOW_DATA	PROCEDURE	VALID
ADD_EMP1	PROCEDURE	VALID
HIGH	PROCEDURE	VALID

```
DH_UTIL                PACKAGE              VALID

DH_UTIL                PACKAGE BODY         VALID


72 rows selected.

SQL> ED
Wrote file afiedt.buf

  1  SELECT OBJECT_NAME,OBJECT_TYPE,STATUS FROM USER_OBJECTS
  2* WHERE STATUS='INVALID'
SQL> /

OBJECT_NAME            OBJECT_TYPE          STATUS

---------------------- -------------------- -------

GET_ORD                FUNCTION             INVALID

INS_REC                FUNCTION             INVALID

GET_JOB                FUNCTION             INVALID

ADD_NEW_EMP            PROCEDURE            INVALID

EMP_POSTING            PROCEDURE            INVALID

DEL_REC                PROCEDURE            INVALID

FORWARD_DEC            PACKAGE              INVALID

FORWARD_DEC            PACKAGE BODY         INVALID

TEST_JOB               PROCEDURE            INVALID

VU_SAL                 VIEW                 INVALID

VU_MGR                 VIEW                 INVALID

ADD_R                  PROCEDURE            INVALID

TEST2                  PROCEDURE            INVALID

MY_NEW_PACK            PACKAGE BODY         INVALID


14 rows selected.

SQL> DROP VIEW EMP_INFO;

View dropped.

SQL> SELECT OBJECT_NAME,OBJECT_TYPE,STATUS FROM USER_OBJECTS
  2  WHERE STATUS='INVALID';

OBJECT_NAME            OBJECT_TYPE          STATUS

---------------------- -------------------- -------

GET_ORD                FUNCTION             INVALID

INS_REC                FUNCTION             INVALID
```

```
GET_JOB              FUNCTION            INVALID

ADD_NEW_EMP          PROCEDURE           INVALID

EMP_POSTING          PROCEDURE           INVALID

DEL_REC              PROCEDURE           INVALID

FORWARD_DEC          PACKAGE             INVALID

FORWARD_DEC          PACKAGE BODY        INVALID

TEST_JOB             PROCEDURE           INVALID

VU_SAL               VIEW                INVALID

VU_MGR               VIEW                INVALID

GET_ID               FUNCTION            INVALID

ADD_R                PROCEDURE           INVALID

TEST2                PROCEDURE           INVALID

MY_NEW_PACK          PACKAGE BODY        INVALID

SAL_INFO             VIEW                INVALID

MGR_INFO             VIEW                INVALID

A_EMP                PROCEDURE           INVALID

18 rows selected.

SQL> CREATE OR REPLACE VIEW EMP_INFO AS SELECT * FROM EMP;

View created.

SQL> SELECT OBJECT_NAME,OBJECT_TYPE,STATUS FROM USER_OBJECTS
  2  WHERE STATUS='INVALID';

OBJECT_NAME          OBJECT_TYPE         STATUS

-------------------- ------------------- -------

GET_ORD              FUNCTION            INVALID

INS_REC              FUNCTION            INVALID

GET_JOB              FUNCTION            INVALID

ADD_NEW_EMP          PROCEDURE           INVALID

EMP_POSTING          PROCEDURE           INVALID

DEL_REC              PROCEDURE           INVALID

FORWARD_DEC          PACKAGE             INVALID

FORWARD_DEC          PACKAGE BODY        INVALID
```

```
TEST_JOB              PROCEDURE            INVALID

VU_SAL                VIEW                 INVALID

VU_MGR                VIEW                 INVALID

GET_ID                FUNCTION             INVALID

ADD_R                 PROCEDURE            INVALID

TEST2                 PROCEDURE            INVALID

MY_NEW_PACK           PACKAGE BODY         INVALID

SAL_INFO              VIEW                 INVALID

MGR_INFO              VIEW                 INVALID

A_EMP                 PROCEDURE            INVALID

18 rows selected.

SQL> SELECT * FROM MGR_INFO;

     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

SQL> SELECT OBJECT_NAME,OBJECT_TYPE,STATUS FROM USER_OBJECTS
  2  WHERE STATUS='INVALID';

OBJECT_NAME           OBJECT_TYPE          STATUS

-------------------- ------------------- -------

GET_ORD               FUNCTION             INVALID

INS_REC               FUNCTION             INVALID

GET_JOB               FUNCTION             INVALID

ADD_NEW_EMP           PROCEDURE            INVALID

EMP_POSTING           PROCEDURE            INVALID

DEL_REC               PROCEDURE            INVALID

FORWARD_DEC           PACKAGE              INVALID

FORWARD_DEC           PACKAGE BODY         INVALID

TEST_JOB              PROCEDURE            INVALID

VU_SAL                VIEW                 INVALID
```

```
VU_MGR                VIEW                INVALID

GET_ID                FUNCTION            INVALID

ADD_R                 PROCEDURE           INVALID

TEST2                 PROCEDURE           INVALID

MY_NEW_PACK           PACKAGE BODY        INVALID

A_EMP                 PROCEDURE           INVALID


16 rows selected.

SQL> ALTER PROCEDURE A_EMP COMPILE;

Procedure altered.

SQL> SELECT OBJECT_NAME,OBJECT_TYPE,STATUS FROM USER_OBJECTS
  2  WHERE STATUS='INVALID';

OBJECT_NAME           OBJECT_TYPE         STATUS

--------------------- ------------------- -------

GET_ORD               FUNCTION            INVALID

INS_REC               FUNCTION            INVALID

GET_JOB               FUNCTION            INVALID

ADD_NEW_EMP           PROCEDURE           INVALID

EMP_POSTING           PROCEDURE           INVALID

DEL_REC               PROCEDURE           INVALID

FORWARD_DEC           PACKAGE             INVALID

FORWARD_DEC           PACKAGE BODY        INVALID

TEST_JOB              PROCEDURE           INVALID

VU_SAL                VIEW                INVALID

VU_MGR                VIEW                INVALID

ADD_R                 PROCEDURE           INVALID

TEST2                 PROCEDURE           INVALID

MY_NEW_PACK           PACKAGE BODY        INVALID


14 rows selected.

SQL> DESC EMP
 Name                                                  Null?    Type
 ----------------------------------------------------- -------- ------------------------------------
 EMPNO                                                 NOT NULL NUMBER(4)
 ENAME                                                          VARCHAR2(10)
```

```
 JOB                                                       VARCHAR2(9)
 MGR                                                       NUMBER(5)
 HIREDATE                                                  DATE
 SAL                                                       NUMBER(7,2)
 COMM                                                      NUMBER(7,2)
 DEPTNO                                                    NUMBER(2)

SQL>
SQL>
SQL>
SQL> ALTER TABLE EMP MODIFY ENAME VARCHAR2(11);

Table altered.

SQL> DESC EMP
 Name                                                      Null?    Type
 --------------------------------------------------------- -------- ------
-----------------------------------
 EMPNO                                                     NOT NULL NUMBER(4)
 ENAME                                                              VARCHAR2(11)
 JOB                                                                VARCHAR2(9)
 MGR                                                                NUMBER(5)
 HIREDATE                                                           DATE
 SAL                                                                NUMBER(7,2)
 COMM                                                               NUMBER(7,2)
 DEPTNO                                                             NUMBER(2)

SQL> SELECT OBJECT_NAME,OBJECT_TYPE,STATUS FROM USER_OBJECTS
  2  WHERE STATUS='INVALID';

OBJECT_NAME          OBJECT_TYPE         STATUS

-------------------- ------------------- -------

GET_ORD              FUNCTION            INVALID

INS_REC              FUNCTION            INVALID

ADD_EMP              PROCEDURE           INVALID

GET_MGR              FUNCTION            INVALID

GET_JOB              FUNCTION            INVALID

ADD_NEW_EMP          PROCEDURE           INVALID

EMP_POSTING          PROCEDURE           INVALID

DEL_REC              PROCEDURE           INVALID

P1                   PACKAGE             INVALID

P1                   PACKAGE BODY        INVALID

FORWARD_DEC          PACKAGE             INVALID

FORWARD_DEC          PACKAGE BODY        INVALID

OVERPACK             PACKAGE             INVALID

OVERPACK             PACKAGE BODY        INVALID

BODYLESS_PACK        PACKAGE             INVALID
```

```
WRITE_TO_FILE        PROCEDURE           INVALID

TEST_JOB             PROCEDURE           INVALID

T1                   PROCEDURE           INVALID

SHOW_REC             PROCEDURE           INVALID

EMP_VIEW             VIEW                INVALID

VU_SAL               VIEW                INVALID

VU_MGR               VIEW                INVALID

GET_ID               FUNCTION            INVALID

ADD_R                PROCEDURE           INVALID

MY_CODE              PROCEDURE           INVALID

TEST2                PROCEDURE           INVALID

NEW_REC              FUNCTION            INVALID

MY_PACK              PACKAGE BODY        INVALID

MY_PACK              PACKAGE             INVALID

MY_NEW_PACK          PACKAGE BODY        INVALID

V1                   VIEW                INVALID

ADD_REC              PROCEDURE           INVALID

SAL_INFO             VIEW                INVALID

MGR_INFO             VIEW                INVALID

A_EMP                PROCEDURE           INVALID

EMP_INFO             VIEW                INVALID

SHOW_DATA            PROCEDURE           INVALID

ADD_EMP1             PROCEDURE           INVALID


38 rows selected.

SQL> CONN SYS/ORACLE AS SYSDBA
Connected.
USER is "SYS"
linesize 100
pagesize 100
long 80
SQL>
SQL>
SQL>
SQL> @ D:\oracle\product\10.2.0\db_1\RDBMS\ADMIN\UTLDTREE.SQL

Sequence dropped.
```

```
Sequence created.


Table dropped.


Table created.


Procedure created.


View dropped.

SQL>
SQL> REM This view will succeed if current user is sys.  This view shows
SQL> REM which shared cursors depend on the given object.  If the current
SQL> REM user is not sys, then this view get an error either about lack
SQL> REM of privileges or about the non-existence of table x$kglxs.
SQL>
SQL> set echo off

View created.

SQL>
SQL> REM This view will succeed if current user is not sys.  This view
SQL> REM does *not* show which shared cursors depend on the given object.
SQL> REM If the current user is sys then this view will get an error
SQL> REM indicating that the view already exists (since prior view create
SQL> REM will have succeeded).
SQL>
SQL> set echo off
create view deptree
            *
ERROR at line 1:
ORA-00955: name is already used by an existing object



View dropped.


View created.

SQL> EXEC DEPTREE_FILL('TABLE','SCOTT','EMP');

PL/SQL procedure successfully completed.

SQL>
SQL>
SQL> SELECT NESTED_LEVEL,TYPE,NAME FROM DEPTREE ORDER BY SEQ#;

NESTED_LEVEL TYPE

------------ ------------------

NAME

--------------------------------------------------------------------------------
----------------
            0 TABLE
```

EMP

1 PACKAGE BODY

FORWARD_DEC

1 PROCEDURE

ADD_NEW_EMP

1 FUNCTION

GET_JOB

1 FUNCTION

INS_REC

1 PACKAGE

FORWARD_DEC

2 PACKAGE BODY

FORWARD_DEC

1 PROCEDURE

SHOW_REC

1 PROCEDURE

TEST_JOB

1 FUNCTION

NEW_REC

1 PROCEDURE

ADD_R

1 PROCEDURE
ADD_EMP1

1 PROCEDURE
MY_CODE

1 PACKAGE
P1

2 PACKAGE BODY
P1

1 VIEW
EMP_VIEW

1 PROCEDURE
SHOW_REC

1 PROCEDURE
ADD_EMP

1 FUNCTION
GET_MGR

2 PROCEDURE
ADD_EMP

2 PACKAGE BODY
MY_PACK

1 PACKAGE BODY

P1

1 PACKAGE

BODYLESS_PACK

1 PACKAGE

MY_PACK

2 PACKAGE BODY

MY_PACK

1 PACKAGE BODY

MY_PACK

1 PACKAGE BODY

OVERPACK

1 PACKAGE

OVERPACK

2 PACKAGE BODY

OVERPACK

1 VIEW

V1

2 PROCEDURE

WRITE_TO_FILE

1 PROCEDURE

ADD_REC

```
NESTED_LEVEL TYPE
------------ ------------------
NAME
--------------------------------------------------------------------------------
----------------

           1 PROCEDURE
T1


           1 PROCEDURE
SHOW_DATA


           1 VIEW
EMP_INFO


           2 VIEW
SAL_INFO


           3 VIEW
MGR_INFO


           2 FUNCTION
GET_ID


           3 PROCEDURE
ADD_R


           3 PROCEDURE
A_EMP


           1 PROCEDURE
A_EMP
```

```
41 rows selected.

SQL>  COL NAME FORMAT A20
SQL> /

NESTED_LEVEL TYPE                NAME
------------ ------------------- --------------------
           0 TABLE               EMP
           1 PACKAGE BODY        FORWARD_DEC
           1 PROCEDURE           ADD_NEW_EMP
           1 FUNCTION            GET_JOB
           1 FUNCTION            INS_REC
           1 PACKAGE             FORWARD_DEC
           2 PACKAGE BODY        FORWARD_DEC
           1 PROCEDURE           SHOW_REC
           1 PROCEDURE           TEST_JOB
           1 FUNCTION            NEW_REC
           1 PROCEDURE           ADD_R
           1 PROCEDURE           ADD_EMP1
           1 PROCEDURE           MY_CODE
           1 PACKAGE             P1
           2 PACKAGE BODY        P1
           1 VIEW                EMP_VIEW
           1 PROCEDURE           SHOW_REC
           1 PROCEDURE           ADD_EMP
           1 FUNCTION            GET_MGR
           2 PROCEDURE           ADD_EMP
           2 PACKAGE BODY        MY_PACK
           1 PACKAGE BODY        P1
           1 PACKAGE             BODYLESS_PACK
           1 PACKAGE             MY_PACK
           2 PACKAGE BODY        MY_PACK
           1 PACKAGE BODY        MY_PACK
```

```
         1 PACKAGE BODY        OVERPACK

         1 PACKAGE             OVERPACK

         2 PACKAGE BODY        OVERPACK

         1 VIEW                V1

         2 PROCEDURE           WRITE_TO_FILE

         1 PROCEDURE           ADD_REC

         1 PROCEDURE           T1

         1 PROCEDURE           SHOW_DATA

         1 VIEW                EMP_INFO

         2 VIEW                SAL_INFO

         3 VIEW                MGR_INFO

         2 FUNCTION            GET_ID

         3 PROCEDURE           ADD_R

         3 PROCEDURE           A_EMP

         1 PROCEDURE           A_EMP

41 rows selected.

SQL>
SQL>
SQL> SPOOL OFF
```

```
SQL>
SQL>
SQL> DECLARE
  2  .
SQL> ED
Wrote file afiedt.buf

  1  CREATE OR REPLACE TRIGGER CHK_DATE
  2    BEFORE INSERT OR UPDATE OR DELETE ON EMP
  3                   BEGIN
  4          IF SUBSTR(SYSDATE,1,2) NOT BETWEEN '01' AND '15' THEN
  5  RAISE_APPLICATION_ERROR(-20401,'U CAN NOT PERFORM DML AFTER 15TH  OF EVERY
MONTH....');
  6                 END IF;
  7*           END;
SQL> /

Trigger created.

SQL> INSERT INTO EMP(EMPNO)
  2  VALUES(1000);
INSERT INTO EMP(EMPNO)
            *
ERROR at line 1:
ORA-20401: U CAN NOT PERFORM DML AFTER 15TH  OF EVERY MONTH....
ORA-06512: at "SCOTT.CHK_DATE", line 3
ORA-04088: error during execution of trigger 'SCOTT.CHK_DATE'


SQL>
SQL>
SQL>
SQL>
SQL>
SQL> /

1 row created.

SQL> ROLLBACK;

Rollback complete.

SQL> ED
Wrote file afiedt.buf

  1   CREATE TABLE LOG_EMP_HIST
  2          (EMPNO NUMBER(4),OP_DATE VARCHAR2(80),
  3               OP_PERFORM CHAR(1) CHECK(OP_PERFORM IN('I','U','D')),
  4          INCRMENT_BY NUMBER(7,2),DESIGNATION VARCHAR2(10),
  5          SALARY VARCHAR2(10),
  6                  JOIN_DATE DATE,OLD_SALARY NUMBER(7,2),
  7*          DEPTNO NUMBER(02)REFERENCES DEPT(DEPTNO))
SQL>
SQL> /
 CREATE TABLE LOG_EMP_HIST
              *
ERROR at line 1:
ORA-00955: name is already used by an existing object


SQL>
SQL>
SQL>
```

```
SQL> DROP TABLE LOG_EMP_HIST;

Table dropped.

SQL>
SQL>  CREATE TABLE LOG_EMP_HIST
  2          (EMPNO NUMBER(4),OP_DATE VARCHAR2(80),
  3               OP_PERFORM CHAR(1) CHECK(OP_PERFORM IN('I','U','D')),
  4          INCRMENT_BY NUMBER(7,2),DESIGNATION VARCHAR2(10),
  5          SALARY VARCHAR2(10),
  6                  JOIN_DATE DATE,OLD_SALARY NUMBER(7,2),
  7           DEPTNO NUMBER(02)REFERENCES DEPT(DEPTNO));

Table created.

SQL>
SQL>
SQL> DESC LOG_EMP_HIST
 Name                                                  Null?    Type
 ----------------------------------------------------- -------- ------------------------------------
 EMPNO                                                          NUMBER(4)
 OP_DATE                                                        VARCHAR2(80)
 OP_PERFORM                                                     CHAR(1)
 INCRMENT_BY                                                    NUMBER(7,2)
 DESIGNATION                                                    VARCHAR2(10)
 SALARY                                                         VARCHAR2(10)
 JOIN_DATE                                                      DATE
 OLD_SALARY                                                     NUMBER(7,2)
 DEPTNO                                                         NUMBER(2)

SQL>
SQL>
SQL>
SQL>
SQL>
SQL> DROP TRIGGER CHK_DATE;

Trigger dropped.

SQL> ED
Wrote file afiedt.buf

  1  CREATE OR REPLACE TRIGGER DML_ON_EMP
  2           BEFORE INSERT OR UPDATE OR DELETE ON EMP
  3               DECLARE
  4               OP_PER CHAR(1);
  5               BEGIN
  6               IF INSERTING THEN
  7              OP_PER:='I';
  8              ELSIF UPDATING THEN
  9              OP_PER:='U';
 10          ELSIF DELETING THEN
 11            OP_PER:='D';
 12            END IF;
 13          INSERT INTO LOG_EMP_HIST(OP_DATE,OP_PERFORM)
 14         VALUES(CURRENT_TIMESTAMP,OP_PER);
 15*    END;
 16  /

Trigger created.

SQL> SELECT * FROM LOG_EMP_HIST;
```

```
no rows selected

SQL> INSERT INTO EMP(EMPNO)
  2  VALUES(1000);

1 row created.

SQL> SELECT * FROM LOG_EMP_HIST;

     EMPNO OP_DATE
       O
----------
--------------------------------------------------------------------------------- -

INCRMENT_BY DESIGNATIO SALARY     JOIN_DATE OLD_SALARY     DEPTNO

----------- ---------- ---------- --------- ---------- ----------

           21-MAR-13 07.24.39.234000 PM +05:00
       I




SQL> ROLLBACK;

Rollback complete.

SQL> DROP TRIGGER   DML_ON_EMP;

Trigger dropped.

SQL> ED
Wrote file afiedt.buf

  1      CREATE OR REPLACE TRIGGER CHK_HIREDATE
  2         BEFORE INSERT OR UPDATE ON EMP FOR EACH ROW
  3        BEGIN
  4    IF SUBSTR(:NEW.HIREDATE,1,2)>=25 AND SUBSTR(LAST_DAY(:NEW.HIREDATE),1,2)<=31
THEN
  5  RAISE_APPLICATION_ERROR(-20401,'HIREDATE MUST BE LESS THEN 25TH OF EVERY
MONTH...');
  6                  END IF;
  7*            END;
SQL> /

Trigger created.

SQL>
SQL>
SQL> INSERT INTO EMP(EMPNO,HIREDATE)
  2  VALUES(1200,SYSDATE);

1 row created.

SQL> /
INSERT INTO EMP(EMPNO,HIREDATE)
            *
ERROR at line 1:
ORA-20401: HIREDATE MUST BE LESS THEN 25TH OF EVERY MONTH...
ORA-06512: at "SCOTT.CHK_HIREDATE", line 3
```

```
ORA-04088: error during execution of trigger 'SCOTT.CHK_HIREDATE'


SQL> SELECT SYSDATE FROM DUAL;

SYSDATE

---------

29-MAR-13


SQL>
SQL>
SQL>
SQL>       CREATE OR REPLACE TRIGGER CHK_HIREDATE
  2           BEFORE INSERT OR UPDATE ON EMP FOR EACH ROW
  3         BEGIN
  4      IF SUBSTR(:NEW.HIREDATE,1,2)>=25 AND
SUBSTR(LAST_DAY(:NEW.HIREDATE),1,2)<=31 THEN
  5    RAISE_APPLICATION_ERROR(-20401,'HIREDATE MUST BE LESS THEN 25TH OF EVERY
MONTH...');
  6                    END IF;
  7  *            END;
  8  .
SQL> ED
Wrote file afiedt.buf

  1        CREATE OR REPLACE TRIGGER CHK_HIREDATE
  2           BEFORE INSERT OR UPDATE ON EMP FOR EACH ROW
  3         BEGIN
  4      IF SUBSTR(:NEW.HIREDATE,1,2)>=25 AND
SUBSTR(LAST_DAY(:NEW.HIREDATE),1,2)<=31 THEN
  5    &D('HIREDATE MUST BE LESS THEN 25TH OF EVERY MONTH...');
  6                    END IF;
  7*            END;
SQL> /

Trigger created.

SQL> INSERT INTO EMP(EMPNO,HIREDATE)
  2  VALUES(1200,SYSDATE);
HIREDATE MUST BE LESS THEN 25TH OF EVERY MONTH...

INSERT INTO EMP(EMPNO,HIREDATE)
*
ERROR at line 1:
ORA-00001: unique constraint (SCOTT.PK_EMP) violated


SQL>
SQL>
SQL>
SQL>
SQL> ED
Wrote file afiedt.buf

  1  INSERT INTO EMP(EMPNO,HIREDATE)
  2* VALUES(1201,SYSDATE)
SQL> /
HIREDATE MUST BE LESS THEN 25TH OF EVERY MONTH...
```

```
1 row created.

SQL>
SQL>
SQL>
SQL>
SQL>
SQL> ED
Wrote file afiedt.buf

SP2-0223: No lines in SQL buffer.
SQL>     CREATE OR REPLACE TRIGGER CHK_HIREDATE
  2         BEFORE INSERT OR UPDATE ON EMP FOR EACH ROW
  3        BEGIN
  4    IF SUBSTR(:NEW.HIREDATE,1,2)>=25 AND SUBSTR(LAST_DAY(:NEW.HIREDATE),1,2)<=31
THEN
  5  RAISE_APPLICATION_ERROR(-20401,'HIREDATE MUST BE LESS THEN 25TH OF EVERY
MONTH...');
  6                 END IF;
  7             END;
  8  /

Trigger created.

SQL>
SQL> INSERT INTO EMP(EMPNO,HIREDATE)
  2  VALUES(1201,SYSDATE)
  3  ;
INSERT INTO EMP(EMPNO,HIREDATE)
            *
ERROR at line 1:
ORA-20401: HIREDATE MUST BE LESS THEN 25TH OF EVERY MONTH...
ORA-06512: at "SCOTT.CHK_HIREDATE", line 3
ORA-04088: error during execution of trigger 'SCOTT.CHK_HIREDATE'


SQL>
SQL>
SQL>
SQL>
SQL>
SQL> DROP TRIGGER CHK_HIREDATE;

Trigger dropped.

SQL>
SQL>
SQL>
SQL> ED
Wrote file afiedt.buf

  1     CREATE OR REPLACE TRIGGER CHK_TIMING
  2     BEFORE INSERT OR  UPDATE OR DELETE ON EMP
  3          BEGIN
  4          IF TO_CHAR(SYSDATE,'DY') IN('FRI','SAT','SUN')
  5       OR TO_CHAR(SYSDATE,'HH24')NOT BETWEEN '09' AND '17' THEN
  6          IF INSERTING THEN
  7           RAISE_APPLICATION_ERROR
  8            (-20100,'OPERATION INVALID,CAN NOT PERFORM INSERT AFTER 5:00PM OR
FRIDAY TO SUNDAY...');
  9          ELSIF UPDATING THEN
 10         RAISE_APPLICATION_ERROR
 11         (-20101,'OPERATION INVALID, CAN NOT PERFORM UPDATE AFTER 5:00 PM OR
```

```
FRIDDAY TO SUNDAY...');
 12        ELSIF DELETING THEN
 13        RAISE_APPLICATION_ERROR
 14        (-20102,'OPERATION INVALID,CAN NOT PERFORM DELETE AFTER 5:00 OR FRIDAY TO
SUNDAY...');
 15        END IF;
 16        END IF;
 17*    END;
SQL> /

Trigger created.

SQL>
SQL>
SQL>
SQL>
SQL> INSERT INTO EMP(EMPNO)
  2  VALUES(123);
INSERT INTO EMP(EMPNO)
            *
ERROR at line 1:
ORA-20100: OPERATION INVALID,CAN NOT PERFORM INSERT AFTER 5:00PM OR FRIDAY TO
SUNDAY...
ORA-06512: at "SCOTT.CHK_TIMING", line 5
ORA-04088: error during execution of trigger 'SCOTT.CHK_TIMING'


SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL> ALTER TRIGGER CHK_TIMING DISABLE;

Trigger altered.

SQL>
SQL>
SQL>
SQL> INSERT INTO EMP(EMPNO)
  2  VALUES(123);

1 row created.

SQL>
SQL>
SQL>
SQL> ED
Wrote file afiedt.buf

  1     CREATE OR REPLACE TRIGGER DML_ON_EMP
  2         BEFORE INSERT OR UPDATE OR DELETE ON EMP
  3          FOR EACH ROW
  4          BEGIN
-----------------------------------INSERTING------------------------
  5          IF INSERTING THEN
  6          INSERT INTO LOG_EMP_HIST
  7          (EMPNO,OP_DATE,OP_PERFORM,DESIGNATION,SALARY,JOIN_DATE,DEPTNO)
  8          VALUES
  9
(:NEW.EMPNO,CURRENT_TIMESTAMP,'I',:NEW.JOB,:NEW.SAL,:NEW.HIREDATE,:NEW.DEPTNO);
 10        ----------------------------------DELETION------------------------------
```

```
 11       ELSIF DELETING THEN
 12        INSERT INTO LOG_EMP_HIST
 13        (EMPNO,OP_DATE,OP_PERFORM,DESIGNATION,SALARY,JOIN_DATE,DEPTNO)
 14        VALUES
 15
(:OLD.EMPNO,CURRENT_TIMESTAMP,'D',:OLD.JOB,:OLD.SAL,:OLD.HIREDATE,:OLD.DEPTNO);
 16        --------------------------UPDATING-------------------------------------
 17        ELSIF UPDATING THEN
 18         DECLARE
 19            CNTR NUMBER:=0;
 20         BEGIN
 21          CNTR:=:NEW.SAL-:OLD.SAL;
 22       INSERT INTO LOG_EMP_HIST
 23        VALUES
 24
(:OLD.EMPNO,CURRENT_TIMESTAMP,'U',CNTR,:NEW.JOB,:NEW.SAL,:NEW.HIREDATE,:OLD.SAL,:NEW
.DEPTNO);
 25       END;
 26       END IF;
 27*   END;
 28  /

Trigger created.

SQL>
SQL>
SQL>
SQL>
SQL> SELECT * FROM EMP;

     EMPNO ENAME       JOB               MGR HIREDATE          SAL       COMM
DEPTNO
---------- ----------- --------- ---------- --------- ---------- ----------
----------
      8000 SCOTT       SALESMAN                             1000
30
      8001 SCOTT       SALESMAN                             1000
35
      1200                                    21-MAR-13

      1201                                    29-MAR-13

       123

      8002 SCOTT       SALESMAN                             1000
30
      8003 SCOTT       SALESMAN                             1000
30
      5454 SMITH       CLERK             7902 17-DEC-80        900
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       1000
30
      7902 FORD       ANALYST         7566 03-DEC-81      45666
20
      7934 MILLER     CLERK           7782 23-JAN-85       1300
10

21 rows selected.

SQL> UPDATE EMP
  2  SET SAL = SAL + 500
  3  WHERE SAL=1000;

5 rows updated.

SQL> SELECT * FROM EMP;

     EMPNO ENAME      JOB              MGR HIREDATE          SAL       COMM
DEPTNO
---------- ---------- --------- ---------- --------- ---------- ----------
----------
      8000 SCOTT      SALESMAN                             1500
30
      8001 SCOTT      SALESMAN                             1500
35
      1200                                 21-MAR-13

      1201                                 29-MAR-13

       123

      8002 SCOTT      SALESMAN                             1500
30
      8003 SCOTT      SALESMAN                             1500
30
      5454 SMITH      CLERK           7902 17-DEC-80        900
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       1500
30
      7902 FORD        ANALYST         7566 03-DEC-81      45666
20
      7934 MILLER      CLERK           7782 23-JAN-85       1300
10

21 rows selected.

SQL>
SQL>
SQL>
SQL> SELECT * FROM LOG_EMP_HIST;

     EMPNO OP_DATE
       O
----------
------------------------------------------------------------------------------- -

INCRMENT_BY DESIGNATIO SALARY     JOIN_DATE OLD_SALARY     DEPTNO

----------- ---------- ---------- --------- ---------- ----------

      8000 29-MAR-13 07.55.16.937000 PM +05:00
       U
         500 SALESMAN   1500                        1000         30


      8001 29-MAR-13 07.55.16.953000 PM +05:00
       U
         500 SALESMAN   1500                        1000         35


      8002 29-MAR-13 07.55.16.953000 PM +05:00
       U
         500 SALESMAN   1500                        1000         30


      8003 29-MAR-13 07.55.16.953000 PM +05:00
       U
         500 SALESMAN   1500                        1000         30


      7900 29-MAR-13 07.55.16.953000 PM +05:00
       U
         500 CLERK      1500       03-DEC-81        1000         30



SQL>
SQL>
SQL>
SQL> COMMIT
  2  ;

Commit complete.
```

```
SQL> DELETE FROM EMP
  2  WHERE HIREDATE IS NULL;

5 rows deleted.

SQL> COMMIT;

Commit complete.

SQL> SELECT * FROM LOG_EMP_HIST;

     EMPNO OP_DATE
       O
----------
--------------------------------------------------------------------------------

INCRMENT_BY DESIGNATIO SALARY     JOIN_DATE OLD_SALARY     DEPTNO

----------- ---------- ---------- --------- ---------- ----------

      8000 29-MAR-13 07.55.16.937000 PM +05:00
       U
        500 SALESMAN   1500                        1000         30



      8001 29-MAR-13 07.55.16.953000 PM +05:00
       U
        500 SALESMAN   1500                        1000         35



      8002 29-MAR-13 07.55.16.953000 PM +05:00
       U
        500 SALESMAN   1500                        1000         30



      8003 29-MAR-13 07.55.16.953000 PM +05:00
       U
        500 SALESMAN   1500                        1000         30



      7900 29-MAR-13 07.55.16.953000 PM +05:00
       U
        500 CLERK      1500       03-DEC-81       1000         30



      8000 29-MAR-13 07.56.33.375000 PM +05:00
       D
            SALESMAN   1500                                    30



      8001 29-MAR-13 07.56.33.375000 PM +05:00
       D
            SALESMAN   1500                                    35



       123 29-MAR-13 07.56.33.375000 PM +05:00
```

```
       D



      8002 29-MAR-13 07.56.33.375000 PM +05:00
       D
            SALESMAN   1500                                        30


      8003 29-MAR-13 07.56.33.375000 PM +05:00
       D
            SALESMAN   1500                                        30



10 rows selected.

SQL>
SQL>
SQL>
SQL>
SQL> ED
Wrote file afiedt.buf

  1      CREATE OR REPLACE TRIGGER DRIVE_COMM
  2         BEFORE INSERT OR UPDATE OF SAL ON EMP FOR EACH ROW
  3            WHEN (NEW.JOB='SALESMAN')
  4            BEGIN
  5         IF INSERTING AND :NEW.COMM IS NULL THEN
  6             :NEW.COMM:=TRUNC((:NEW.SAL*30)/100);
  7            ELSIF :OLD.COMM IS NULL THEN
  8             :NEW.COMM:=TRUNC((:NEW.SAL*30)/100);
  9            ELSIF :OLD.JOB='SALESMAN' THEN
 10          :NEW.COMM:=NVL(:OLD.COMM,0)+TRUNC(((:NEW.SAL-:OLD.SAL)*30)/100);
 11         ELSE
 12         :NEW.COMM :='';
 13         END IF;
 14*     END DRIVE_COMM;
 15  /

Trigger created.

SQL> INSERT INTO EMP(EMPNO,SAL)
  2  VALUES(1002,1000);

1 row created.

SQL> COMMIT;

Commit complete.

SQL> SELECT * FROM EMP
  2  WHERE EMPNO=1002;

     EMPNO ENAME       JOB              MGR HIREDATE          SAL       COMM
DEPTNO
---------- ----------- --------- ---------- --------- ---------- ----------
----------
      1002                                                  1000
```

```
SQL>
SQL>
SQL>
SQL>  INSERT INTO EMP(EMPNO,SAL)
  2   VALUES(1002,1000)
  3  .
SQL> ED
Wrote file afiedt.buf

  1   INSERT INTO EMP(EMPNO,SAL,JOB)
  2*  VALUES(1003,1000,'SALESMAN')
SQL> /

1 row created.

SQL> SELECT * FROM EMP
  2  WHERE EMPNO=1003;

     EMPNO ENAME       JOB               MGR HIREDATE          SAL        COMM
DEPTNO
---------- ----------- --------- ---------- --------- ---------- ----------
----------
      1003             SALESMAN                              1000         300


SQL>
SQL>
SQL>
SQL>
SQL> COMMIT;

Commit complete.

SQL> UPDATE EMP
  2  SET SAL  =SAL + 500;

18 rows updated.

SQL> ROLLBACKL
SP2-0042: unknown command "ROLLBACKL" - rest of line ignored.
SQL> ROLLBACK;

Rollback complete.

SQL> UPDATE EMP
  2  SET SAL  =SAL + 500 WHERE EMPNO=1003;

1 row updated.

SQL>
SQL>
SQL>
SQL>
SQL>  SELECT * FROM EMP
  2   WHERE EMPNO=1003;

     EMPNO ENAME       JOB               MGR HIREDATE          SAL        COMM
DEPTNO
---------- ----------- --------- ---------- --------- ---------- ----------
----------
      1003             SALESMAN                              1500         450
```

```
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL> ED
Wrote file afiedt.buf

  1  CREATE OR REPLACE PROCEDURE CHK_SAL(SALARY IN NUMBER)IS
  2     AA NUMBER:=0;
  3     MISAL NUMBER:=0;
  4     MXSAL NUMBER:=0;
  5          BEGIN
  6    SELECT MIN(LOSAL),MAX(HISAL) INTO MISAL,MXSAL FROM SCOTT.SALGRADE;
  7     SELECT GRADE INTO AA FROM SCOTT.SALGRADE
  8     WHERE SALARY BETWEEN LOSAL AND HISAL;
  9         EXCEPTION
 10     WHEN NO_DATA_FOUND THEN
 11    RAISE_APPLICATION_ERROR(-20222,'SORRY MUST BE BETWEEN...'||MISAL||
 12                              ' AND '||MXSAL||'..');
 13*     END;
 14  /

Procedure created.

SQL> EXEC CHK_SAL(100);
BEGIN CHK_SAL(100); END;

*
ERROR at line 1:
ORA-20222: SORRY MUST BE BETWEEN...700 AND 9999..
ORA-06512: at "SCOTT.CHK_SAL", line 11
ORA-06512: at line 1


SQL>
SQL>
SQL>
SQL> ED
Wrote file afiedt.buf

  1   CREATE OR REPLACE TRIGGER EMP_CHK
  2        BEFORE INSERT OR UPDATE OF SAL ON EMP
  3          FOR EACH ROW
  4*      CALL CHK_SAL(:NEW.SAL)
  5  /

Trigger created.

SQL> INSERT INTO EMP(EMPNO,SAL) VALUES(1021,100);
INSERT INTO EMP(EMPNO,SAL) VALUES(1021,100)
       *
ERROR at line 1:
ORA-20222: SORRY MUST BE BETWEEN...700 AND 9999..
ORA-06512: at "SCOTT.CHK_SAL", line 11
ORA-06512: at "SCOTT.EMP_CHK", line 1
```

```
ORA-04088: error during execution of trigger 'SCOTT.EMP_CHK'


SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL> INSERT INTO EMP(EMPNO,SAL) VALUES(1021,1000);

1 row created.

SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL> ED
Wrote file afiedt.buf

  1  CREATE TABLE EMP_AUDIT(EMP_AUDIT_ID NUMBER(4),
  2*  UP_DATE DATE,NEW_SAL NUMBER(7,2),OLD_SAL NUMBER(7,2))
SQL> /
CREATE TABLE EMP_AUDIT(EMP_AUDIT_ID NUMBER(4),
             *
ERROR at line 1:
ORA-00955: name is already used by an existing object


SQL>
SQL>
SQL>
SQL> DROP TABLE EMP_AUDIT;

Table dropped.

SQL>  CREATE TABLE EMP_AUDIT(EMP_AUDIT_ID NUMBER(4),
  2    UP_DATE DATE,NEW_SAL NUMBER(7,2),OLD_SAL NUMBER(7,2));

Table created.

SQL>
SQL>
SQL>
SQL> ED
Wrote file afiedt.buf

  1   ----------------TRIGGER----------
  2           CREATE OR REPLACE TRIGGER AUDIT_SAL
  3           AFTER UPDATE OF SAL ON EMP FOR EACH ROW
  4           DECLARE
  5           PRAGMA AUTONOMOUS_TRANSACTION;
  6          BEGIN
  7       INSERT INTO EMP_AUDIT
  8       VALUES (:OLD.EMPNO,SYSDATE,:NEW.SAL,:OLD.SAL);
  9           COMMIT;
 10*    END;
SQL> /

Trigger created.
```

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

     EMPNO ENAME       JOB              MGR HIREDATE          SAL       COMM
DEPTNO
---------- ----------- --------- ---------- --------- ---------- ----------
----------
      1021                                                 1000
      1200                                   21-MAR-13
      1201                                   29-MAR-13
      1002                                                 1000
      1003             SALESMAN                            1500        450
      5454 SMITH       CLERK            7902 17-DEC-80        900
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       1500
30
      7902 FORD        ANALYST          7566 03-DEC-81      45666
20
      7934 MILLER      CLERK            7782 23-JAN-85       1300
10

19 rows selected.

SQL>
SQL>
SQL>
SQL> COMMIT;

Commit complete.

SQL> UPDATE EMP
  2  SET SAL=(SELECT SAL FROM EMP WHERE EMPNO=7566)
```

```
  3  WHERE EMPNO=1021;

1 row updated.

SQL>
SQL>
SQL>
SQL>
SQL>
SQL> SELECT * FROM EMP_AUDIT;

EMP_AUDIT_ID UP_DATE      NEW_SAL    OLD_SAL

------------ --------- ---------- ----------

        1021 29-MAR-13       2975       1000


SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL> ED
Wrote file afiedt.buf

  1    ---------------------------MUTATING TABLE------------
  2         CREATE OR REPLACE TRIGGER CHECK_SALARY
  3          BEFORE INSERT OR UPDATE OF SAL,JOB ON EMP
  4          FOR EACH ROW
  5          WHEN(NEW.JOB<>'PRESIDENT')
  6         DECLARE
  7          MIN_SAL EMP.SAL%TYPE;
  8          MAX_SAL EMP.SAL%TYPE;
  9          BEGIN
 10         SELECT MIN(SAL),MAX(SAL) INTO MIN_SAL,MAX_SAL FROM EMP
 11        WHERE JOB=:NEW.JOB;
 12       IF :NEW.SAL<MIN_SAL OR :NEW.SAL>MAX_SAL THEN
 13        RAISE_APPLICATION_ERROR(-20100,'INVALID SALARY FOR THIS JOB.
 14              VALID SALARY IS..'||MIN_SAL ||' AND '||MAX_SAL);
 15       END IF;
 16*    END;
SQL> /

Trigger created.

SQL> SELECT MIN(SAL),MAX(SAL) FROM EMP
  2  WHERE JOB='&JOB';
Enter value for job: PRESIDENT

  MIN(SAL)   MAX(SAL)

---------- ----------
```

```
      5000       5000


SQL>
SQL> /
Enter value for job: SALESMAN

  MIN(SAL)   MAX(SAL)

---------- ----------

      1250       1600


SQL>
SQL>
SQL>
SQL>
SQL>
SQL> /
Enter value for job: MANAGER

  MIN(SAL)   MAX(SAL)

---------- ----------

      2450       2975


SQL>
SQL>
SQL> INSERT INTO EMP(EMPNO,SAL,JOB) VALUES(232,1000,'SALESMAN');
INSERT INTO EMP(EMPNO,SAL,JOB) VALUES(232,1000,'SALESMAN')
            *
ERROR at line 1:
ORA-20100: INVALID SALARY FOR THIS JOB.
VALID SALARY IS..1250 AND 1600
ORA-06512: at "SCOTT.CHECK_SALARY", line 8
ORA-04088: error during execution of trigger 'SCOTT.CHECK_SALARY'


SQL>
SQL>
SQL> INSERT INTO EMP(EMPNO,SAL,JOB) VALUES(232,1601,'SALESMAN');
INSERT INTO EMP(EMPNO,SAL,JOB) VALUES(232,1601,'SALESMAN')
            *
ERROR at line 1:
ORA-20100: INVALID SALARY FOR THIS JOB.
VALID SALARY IS..1250 AND 1600
ORA-06512: at "SCOTT.CHECK_SALARY", line 8
ORA-04088: error during execution of trigger 'SCOTT.CHECK_SALARY'


SQL>
SQL> INSERT INTO EMP(EMPNO,SAL,JOB) VALUES(232,1600,'SALESMAN');

1 row created.

SQL>
SQL>
SQL>
SQL> UPDATE EMP
```

```
  2  SET SAL=1601
  3  WHERE SAL=1600
  4  AND JOB='SALESMAN';
UPDATE EMP
       *
ERROR at line 1:
ORA-04091: table SCOTT.EMP is mutating, trigger/function may not see it
ORA-06512: at "SCOTT.CHECK_SALARY", line 5
ORA-04088: error during execution of trigger 'SCOTT.CHECK_SALARY'


SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL> DROP TRIGGER CHECK_SALARY;

Trigger dropped.

SQL> CONN SYS/ORACLE AS SYSDBA
Connected.
USER is "SYS"
linesize 100
pagesize 100
long 80
SQL> DROP  TABLE USER_DATABASE_OBJECTS;

Table dropped.

SQL> ED
Wrote file afiedt.buf

  1    CREATE TABLE USER_DATABASE_OBJECTS
  2      (USERID VARCHAR2(30),
  3      TASK_PERFORM VARCHAR2(200),
  4*   WHICH_TIME TIMESTAMP WITH TIME ZONE)
SQL> /

Table created.

SQL>
SQL>
SQL>
SQL> DESC USER_DATABASE_OBJECTS
 Name                                                  Null?    Type
 ----------------------------------------------------- -------- ------------------------------------
 USERID                                                         VARCHAR2(30)
 TASK_PERFORM                                                   VARCHAR2(200)
 WHICH_TIME                                                     TIMESTAMP(6) WITH
TIME ZONE

SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL> ED
Wrote file afiedt.buf
```

```
  1   CREATE OR REPLACE TRIGGER DDL_CHK
  2       AFTER CREATE OR ALTER OR DROP ON DATABASE
  3       DECLARE
  4      TP VARCHAR2(200);
  5      BEGIN
  6      TP:=SYS_CONTEXT('USERENV','TERMINAL')
  7      ||' '||ORA_SYSEVENT||' '||ORA_DICT_OBJ_NAME
  8      ||' '||SYS_CONTEXT('USERENV','OS_USER');
  9      INSERT INTO USER_DATABASE_OBJECTS
 10     VALUES(USER,TP,LOCALTIMESTAMP);
 11*   END;
 12  /

Trigger created.

SQL>
SQL>
SQL>
SQL> SELECT * FROM USER_DATABASE_OBJECTS;

no rows selected

SQL> /

USERID

------------------------------

TASK_PERFORM

--------------------------------------------------------------------------------
----------------
WHICH_TIME

--------------------------------------------------------------------------------

SCOTT

PC-SOHAIL CREATE TESTING PC-SOHAIL\orasoft

29-MAR-13 08.47.28.593000 PM +05:00



SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL> /

USERID
```

```
------------------------------

TASK_PERFORM

--------------------------------------------------------------------------------
----------------
WHICH_TIME

--------------------------------------------------------------------------------

SCOTT

PC-SOHAIL CREATE TESTING PC-SOHAIL\orasoft

29-MAR-13 08.47.28.593000 PM +05:00


SCOTT

PC-SOHAIL ALTER TESTING PC-SOHAIL\orasoft

29-MAR-13 08.48.19.140000 PM +05:00



SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL> /

USERID

------------------------------

TASK_PERFORM

--------------------------------------------------------------------------------
----------------
WHICH_TIME

--------------------------------------------------------------------------------

SCOTT

PC-SOHAIL CREATE TESTING PC-SOHAIL\orasoft

29-MAR-13 08.47.28.593000 PM +05:00


SCOTT
```

PC-SOHAIL ALTER TESTING PC-SOHAIL\orasoft

29-MAR-13 08.48.19.140000 PM +05:00

SCOTT

PC-SOHAIL ALTER TESTING PC-SOHAIL\orasoft

29-MAR-13 08.48.49.812000 PM +05:00

SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL> /

USERID

TASK_PERFORM

WHICH_TIME

SCOTT

PC-SOHAIL CREATE TESTING PC-SOHAIL\orasoft

29-MAR-13 08.47.28.593000 PM +05:00

SCOTT

PC-SOHAIL ALTER TESTING PC-SOHAIL\orasoft

29-MAR-13 08.48.19.140000 PM +05:00

SCOTT

PC-SOHAIL ALTER TESTING PC-SOHAIL\orasoft

29-MAR-13 08.48.49.812000 PM +05:00

```
SCOTT

PC-SOHAIL ALTER TESTING PC-SOHAIL\orasoft

29-MAR-13 08.49.03.515000 PM +05:00


SCOTT

PC-SOHAIL DROP TESTING PC-SOHAIL\orasoft

29-MAR-13 08.49.03.515000 PM +05:00



SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL> DROP TRIGGER DDL_CHK;

Trigger dropped.

SQL> CONN SCOTT/TIGER
Connected.
USER is "SCOTT"
linesize 100
pagesize 100
long 80
SQL>
SQL>
SQL>
SQL> SPOOL OFF
```