Some PL/SQL example

Implicit Cursor

```
declare

     SupplierName   s.sname%type;   -- %type: go to DB and
     SupplierStatus s.status%type;  -- find out exact type

begin
     SELECT sname,status
     INTO SupplierName, SupplierStatus
     FROM s
     WHERE snum = 'S4';

     dbms_output.put_line('Supplier Name is '||
          SupplierName || '; status is '|| SupplierStatus);
          -- the || symbol stands for 'concatenate'. Used
          -- with strings the way other languages use
          -- '+' (javascript) or '&' (VB) or '.' (perl)
end;
```

Explicit Cursor in a While Loop

- -- Think of a PL/SQL explicit cursor as being a -- SEQUENTIAL FILE -- when accessed by a loop
- -- -- An explicit cursor declaration contains three components
- -- i) Identifier for the cursor
- -- ii) specifications of the fields of -- the cursor (a schema - optional)
- -- iii) The select statement used to populate the cursor.
- -- cursor <cursor name> [return <return specification>] is <select statement>;

Example

```
declare

        --declare CURSOR C1
        cursor C1 is
                SELECT snum, sname, city
                FROM S;

        SupNum S.snum%type;
        SupName S.sname%type;
        SupCity S.city%type;

begin
        -- open CURSOR like a sequential file
        open C1;

        -- Retrieve current tuple from the
        -- CURSOR to the PL/SQL variables
        fetch C1 into SupNum, SupName, SupCity;

        -- Repeat so long as a tuple is successfully
        -- fetched from the CURSOR
        while C1%found loop
                dbms_output.put_line('Row Number '||
                        C1%rowcount || ' is '|| SupNum||' '||
                        SupName||' '||SupCity);
                fetch C1 into SupNum, SupName, SupCity;
        end loop;
        -- note LOOP/END LOOP pair

        -- close the CURSOR
        close C1;
end;
```

Note

- Observe above we fetch into a set of scalar variables (not a record) .
- In general it is best to declare a record and fetch into it.
- The example is meant to demonstrate the variations possible.

Resourses

- http://webspace.cs.odu.edu/~ibl/450/plsql2/intro.html
- http://webspace.cs.odu.edu/~ibl/450/plsql2/expcursor2.html

PL/SQL Block

```
-- available online in file 'sample1'
DECLARE
   x NUMBER := 100;
BEGIN
   FOR i IN 1..10 LOOP
      IF MOD(i,2) = 0 THEN     -- i is even
         INSERT INTO temp VALUES (i, x, 'i is even');
      ELSE
         INSERT INTO temp VALUES (i, x, 'i is odd');
      END IF;
      x := x + 100;
   END LOOP;
   COMMIT;
END;
```

Output Table

```
• SQL> SELECT * FROM temp ORDER BY col1;

• NUM_COL1 NUM_COL2  CHAR_COL
• -------- --------  ---------
•        1      100  i is odd
•        2      200  i is even
•        3      300  i is odd
•        4      400  i is even
•        5      500  i is odd
•        6      600  i is even
•        7      700  i is odd
•        8      800  i is even
•        9      900  i is odd
•       10     1000  i is even
```

Ex2: Input Table

```
•   SQL> SELECT ename, empno, sal FROM emp ORDER BY sal DESC;

•   ENAME           EMPNO      SAL
•   ---------- --------- --------
•   KING             7839     5000
•   SCOTT            7788     3000
•   FORD             7902     3000
•   JONES            7566     2975
•   BLAKE            7698     2850
•   CLARK            7782     2450
•   ALLEN            7499     1600
•   TURNER           7844     1500
•   MILLER           7934     1300
•   WARD             7521     1250
•   MARTIN           7654     1250
•   ADAMS            7876     1100
•   JAMES            7900      950
•   SMITH            7369      800
```

PL/SQL Block

```
-- available online in file 'sample2'
DECLARE
   CURSOR c1 is
      SELECT ename, empno, sal FROM emp
         ORDER BY sal DESC;   -- start with highest paid employee
   my_ename VARCHAR2(10);
   my_empno NUMBER(4);
   my_sal   NUMBER(7,2);
BEGIN
   OPEN c1;
   FOR i IN 1..5 LOOP
      FETCH c1 INTO my_ename, my_empno, my_sal;
      EXIT WHEN c1%NOTFOUND;  /* in case the number requested */
                              /* is more than the total       */
                              /* number of employees          */
      INSERT INTO temp VALUES (my_sal, my_empno, my_ename);
      COMMIT;
   END LOOP;
   CLOSE c1;
END;
```

Output Table

```
• SQL> SELECT * FROM temp ORDER BY col1 DESC;

• NUM_COL1 NUM_COL2  CHAR_COL
• -------- --------  --------
•     5000     7839  KING
•     3000     7902  FORD
•     3000     7788  SCOTT
•     2975     7566  JONES
•     2850     7698  BLAKE
```

Resourses

- For more examples see:
- http://download.oracle.com/docs/cd/B10500_01/appdev.920/a96624/a_samps.htm#2644