COLLECTIONS

COLLECTION IS A GROUP OF ELEMENTS OF PARTICULAR DATA TYPE AND ELEMENTS ARE ACCESSED BY USING INDEX.PL/SQL SUPPORTING THREE TYPES OF COLLECTIONS THOSE ARE,

1. PL/SQL TABLE (OR) ASSOCIATED ARRAY (OR) INDEX BY TABLE

2. VARRAY

3. NESTED TABLE

1.PL/SQL TABLE:

- PL/SQL TABLE IS A USER DEFINED TYPE WHICH IS USED TO STORE NUMBER OF DATA ITEMS EITHER INTEGERS OR CHARACTERS.

WHEN WE USE PL/SQL TABLE THEN WE FOLLOW THE FOLLOWING TWO STEPS MECHANISM.

STEP1: DECLARE TYPE:

SYNTAX:

```
TYPE TYPENAME IS TABLE OF DATATYPE(SIZE) INDEX BY
BINARY_INTEGER;
```

STEP2: DECLARE VARIABLE:

SYNTAX:

```
VARIABLENAME TYPENAME;
```

EX1: WRITE A PL/SQL PROGRAM TO PRINT INTERGER ELEMENTS BY USING COLLECTION?

```
DECLARE

TYPE NUM_ARRAY IS TABLE OF NUMBER (4) INDEX BY
BINARY_INTEGER;

X NUM_ARRAY;

BEGIN

FOR I IN 1..10
```

```
LOOP
X(I): =I*10;
DBMS_OUTPUT.PUT_LINE(X(I));
END LOOP;
END;
/
```

```
OUTPUT:
======
10
20
30
40
50
60
70
80
90
100
```

EX2: WRITE A PL/SQL PROGRAM TO PRINT ALL DEPARTMENTS NAMES BY USING COLLECTION?

```
DECLARE
TYPE DNAME_ARRAY IS TABLE OF VARCHAR2(10) INDEX BY
BINARY_INTEGER;
D DNAME_ARRAY;
BEGIN
FOR I IN 1..4
LOOP
SELECT DNAME INTO D(I) FROM DEPT WHERE DEPTNO=I*10;
DBMS_OUTPUT.PUT_LINE(D(I));
END LOOP;
```

```
END;
/
```

OUTPUT:

```
ACCOUNTING
RESEARCH
SALES
OPERATIONS
```

NOTE:

- IN THE ABOVE EXAMPLE SELECT STATEMENT IS INSIDE A LOOP SO THAT NO. OF REQUEST IS GOING TO INCRESE

BURDON ON DATABASE AND REDUCE PERFORMANCE.TO OVERCOME THIS PROBLEM WE USE "BULK COLLECT" CLAUSE.

BULK COLLECT:

- BY USING BULK COLLECT IN A SINGLE REQUEST WE CAN GET ALL ELEMENTS FROM DATABASE SERVER AND STORE THOSE ELEMENTS IN A COLLECTION.SO THAT BULK COLLECT REDUCE NUMBER OF TRIPS TO DATABASE SERVER AND IMPROVES PERFORMANCE.

EX:

```
DECLARE
TYPE DNAME_ARRAY IS TABLE OF VARCHAR2(10) INDEX BY BINARY_INTEGER;
D DNAME_ARRAY;
BEGIN
SELECT DNAME BULK COLLECT INTO D FROM DEPT;
FOR I IN 1..4
LOOP
DBMS_OUTPUT.PUT_LINE(D(I));
END LOOP;
```

```
END;
/
```

OUTPUT:

```
ACCOUNTING
RESEARCH
SALES
OPERATIONS
```

COLLECTION METHODS:

FIRST : RETURN INDEX VALUE OF THE FIRST ELEMENT.

LAST : RETURN INDEX VALUE OF THE LAST ELEMENT.

NEXT : RETURN INDEX VALUE OF THE NEXT ELEMENT.

PRIOR : RETURN INDEX VALUE OF PREVIOUS ELEMENT.

NOTE: ALL THESE METHODS ARE USED BY THE COLLECTION NAME.

SYNTAX:

```
        <COLLECTION NAME>. <METHOD>
```

EX:

```
DECLARE
TYPE DNAME_ARRAY IS TABLE OF VARCHAR2(10) INDEX BY
BINARY_INTEGER;
D DNAME_ARRAY;
BEGIN
SELECT DNAME BULK COLLECT INTO D FROM DEPT;
FOR I IN D. FIRST..D.LAST
LOOP
DBMS_OUTPUT.PUT_LINE(D(I));
END LOOP;
END;
/
```

OUTPUT:

```
ACCOUNTING
RESEARCH
SALES
OPERATIONS
```

EX ON BULK COLLECTING WITH FORWARD NAVIGATION USING "FOR"LOOP WITH RECORDS TYPE:

```
DECLARE
TYPE DNAME_ARRAY IS TABLE OF DEPT%ROWTYPE INDEX BY BINARY_INTEGER;
D DNAME_ARRAY;
BEGIN
SELECT * BULK COLLECT INTO D FROM DEPT;
FOR I IN D. FIRST..D. LAST
LOOP
DBMS_OUTPUT.PUT_LINE(D(I). DEPTNO||','||D(I). DNAME||','||D(I). LOC);
END LOOP;
END;
/
```

EX ON BULK COLLECTING WITH BACKWARD NAVIGATION USING "FOR" LOOP WITH RECORD TYPE:

```
DECLARE
TYPE DNAME_ARRAY IS TABLE OF DEPT%ROWTYPE INDEX BY BINARY_INTEGER;
D DNAME_ARRAY;
BEGIN
SELECT * BULK COLLECT INTO D FROM DEPT;
FOR I IN REVERSE D.FIRST..D.LAST
LOOP
DBMS_OUTPUT.PUT_LINE(D(I).DEPTNO||','||D(I).DNAME||','||D(I).LOC);
```

```
END LOOP;
END;
/
```

EX ON BULK COLLECTING WITH FORWARD NAVIGATION USING "WHILE LOOP" WITH RECORD TYPE:

```
DECLARE
TYPE DNAME_ARRAY IS TABLE OF DEPT%ROWTYPE INDEX BY BINARY_INTEGER;
D DNAME_ARRAY;
X NUMBER (10);
BEGIN
SELECT * BULK COLLECT INTO D FROM DEPT;
X:= D.FIRST;
WHILE (X<=D.LAST)
LOOP
DBMS_OUTPUT.PUT_LINE(D(X).DEPTNO||','||D(X).DNAME||','||D(X).LOC);
X:=D.NEXT(X);
END LOOP;
END;
/
```

EX ON BULK COLLECTING WITH BACKWARD NAVIGATION USING "WHILE LOOP" WITH RECORD TYPE:

```
DECLARE
TYPE DNAME_ARRAY IS TABLE OF DEPT%ROWTYPE INDEX BY BINARY_INTEGER;
D DNAME_ARRAY;
X NUMBER(10);
BEGIN
SELECT * BULK COLLECT INTO D FROM DEPT;
X := D.LAST;
WHILE (X>=D.FIRST)
```

```
LOOP
DBMS_OUTPUT.PUT_LINE(D(X).DEPTNO||','||D(X).DNAME||','||D(X).LOC);
X:=D.PRIOR(X);
END LOOP;
END;
/
```

VARRAY:

- VARRAY ALSO USER DEFINED TYPES WHICH IS USED TO STORE NUMBER OF DATA ITEMS IN A SINGLE UNIT AND

DECLARE WITH SIZE.HERE NO. OF ELEMENTS ARE LIMITED AS PER ARRAY SIZE.

EX:

```
DECLARE
TYPE T1 IS VARRAY (10) OF VARCHAR2(10);
V_T T1;
BEGIN
SELECT ENAME BULK COLLECT INTO V_T FROM EMP WHERE ROWNUM<=10;
FOR I IN V_T. FIRST..V_T.LAST
LOOP
DBMS_OUTPUT.PUT_LINE(V_T(I));
END LOOP;
END;
/
```

NESTED TABLE:

- NESTED TABLE ALSO USER DEFINED TYPES WHICH IS USED TO STORE NUMBER OF DATA ITEMS IN A SINGLE UNIT AND NOT DECLARE WITH SIZE.HERE NO. OF ELEMENTS ARE UNLIMITED.

EX:

```
DECLARE

TYPE T1 IS TABLE OF VARCHAR2(10);

V_T T1;

BEGIN

SELECT ENAME BULK COLLECT INTO V_T FROM EMP WHERE
ROWNUM<=5;

FOR I IN V_T.FIRST..V_T.LAST

LOOP

DBMS_OUTPUT.PUT_LINE(V_T(I));

END LOOP;

END;

/
```