PL/ SQL ARRAYS

Arrays, mainly called Varrays (Variable Arrays), are dynamic information systems that permit the storage of multiple values of the same information type underneath a single variable name.

Syntax for Creating a Varray Type:

```
CREATE TYPE num_array AS VARRAY(10) OF INT;
```

1. PL/ SQL programme for creating and using a Varray of Integers

```
DECLARE
   numbers num_array;
BEGIN
   numbers := num_array(1, 2, 3, 4, 5);
   -- Perform operations on 'numbers'
   FOR i IN 1..numbers.COUNT LOOP
      DBMS_OUTPUT.PUT_LINE('Element ' || i || ': ' || numbers(i));
   END LOOP;

   -- Update an element
   numbers(3) := 30;

   -- Display the updated elements
   FOR i IN 1..numbers.COUNT LOOP
      DBMS_OUTPUT.PUT_LINE('Updated Element ' || i || ': ' || numbers(i));
   END LOOP;
END;
```

Output:

```
Element 1: 1
Element 2: 2
Element 3:3
Element 4: 4
Element 5: 5
```

Updated Output:

Update the detail at index three (value three) to 30.
Use a loop to display the updated elements.

```
Updated Element 1: 1
Updated Element 2: 2
Updated Element 3: 30
Updated Element 4: 4
Updated Element 5: 5
```

Varray of characters:

```
CREATE TYPE char_array AS VARRAY(5) OF CHAR(10);
```

2. PL/ SQL programme for creating a Varray of Strings

```
DECLARE
   -- Declare a variable of the VARRAY type
   my_array char_array := char_array('Apple', 'Banana', 'Cherry');

BEGIN
   -- Display elements
   FOR i IN 1..my_array.COUNT LOOP
      DBMS_OUTPUT.PUT_LINE('Element ' || i || ': ' || my_array(i));
   END LOOP;

   -- Insert a new element
   my_array.EXTEND;
   my_array(4) := 'Date';

   -- Display elements after insertion
   FOR i IN 1..my_array.COUNT LOOP
      DBMS_OUTPUT.PUT_LINE('Element ' || i || ': ' || my_array(i));
   END LOOP;

   -- Update an element
   my_array(2) := 'Blueberry';

   -- Display elements after update
   FOR i IN 1..my_array.COUNT LOOP
      DBMS_OUTPUT.PUT_LINE('Updated Element ' || i || ': ' || my_array(i));
   END LOOP;
END;
```

Output:

```
Element 1: Apple
Element 2: Banana
Element 3: Orange
```

Insert '**Arabian Date**' at index four and display updated elements.

Output:

```
 Element 1: Apple
 Element 2: Banana
 Element 3: Cherry
 Element 4: Date
```

Update index element 2 to 'Strawberry' and Display final elements after the update.

Output:

```
Updated Element 1: Apple
Updated Element 2: Strawberry
Updated Element 3: Cherry
Updated Element 4: Date
```

3. PL/ SQL programme for creating a marks sheet using multiple types of Varrays.

```
DECLARE
   type namesarray IS VARRAY(5) OF VARCHAR2(10);
   type grades IS VARRAY(5) OF INTEGER;
   names namesarray;
   marks grades;
   total integer;
BEGIN
   names := namesarray('Kavya', 'Samiksha', 'Shruthi', 'Sanjay', 'Hemanth');
   marks:= grades(89, 97, 98, 95, 82);
   total := names.count;
   dbms_output.put_line('Total '|| total || ' Students');
   FOR i in 1 .. total LOOP
      dbms_output.put_line('Student: ' || names(i) || '
      Marks: ' || marks(i));
   END LOOP;
END;
/
```

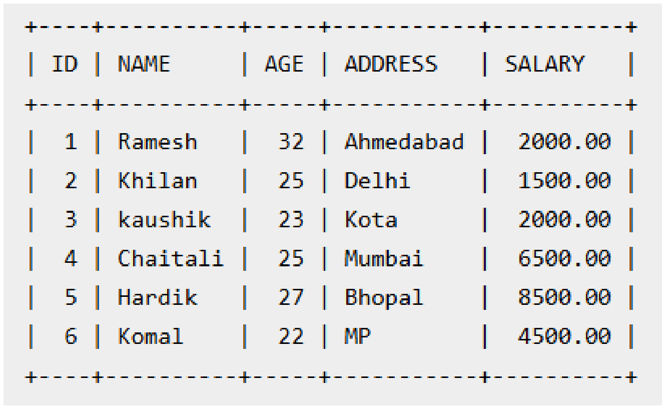
Output:

```
Total 5 Students
Student: Kavya  Marks: 89
Student: Samiksha  Marks: 97
Student: Shruthi  Marks: 98
Student: Sanjay  Marks: 95
```

Student: Hemanth Marks: 82

4. PL/ SQL programme for Using varray as %ROWTYPE.

   Create the following table with the below details and make use of cursors to access each data separately.



```
+----+----------+-----+-----------+----------+
| ID | NAME     | AGE | ADDRESS   | SALARY   |
+----+----------+-----+-----------+----------+
|  1 | Ramesh   |  32 | Ahmedabad |  2000.00 |
|  2 | Khilan   |  25 | Delhi     |  1500.00 |
|  3 | kaushik  |  23 | Kota      |  2000.00 |
|  4 | Chaitali |  25 | Mumbai    |  6500.00 |
|  5 | Hardik   |  27 | Bhopal    |  8500.00 |
|  6 | Komal    |  22 | MP        |  4500.00 |
+----+----------+-----+-----------+----------+
```

```
DECLARE
  CURSOR c_customers is
  SELECT  name FROM customers;
  type c_list is varray (6) of customers.name%type;
  name_list c_list := c_list();
  counter integer :=0;
BEGIN
  FOR n IN c_customers LOOP
    counter := counter + 1;
    name_list.extend;
    name_list(counter)  := n.name;
    dbms_output.put_line('Customer('||counter ||'):'||name_list(counter));
  END LOOP;
END;
/
```

Output:

```
Customer(1): Ramesh
Customer(2): Khilan
Customer(3): kaushik
Customer(4): Chaitali
Customer(5): Hardik
Customer(6): Komal
```

5. PL/ SQL programme for creating a procedure.

Creating a Procedure

A procedure is created with the CREATE OR REPLACE PROCEDURE statement. The simplified syntax for the CREATE OR REPLACE PROCEDURE statement is as follows –

```
CREATE [OR REPLACE] PROCEDURE procedure_name
[(parameter_name [IN | OUT | IN OUT] type [, ...])]
{IS | AS}
BEGIN
  < procedure_body >
END procedure_name;
```

Sample Procedure:

```
CREATE OR REPLACE PROCEDURE greetings
AS
BEGIN
   dbms_output.put_line('Hello World!');
END;
/
```