Tech Mahindra

Collections

Collection

- A collection is a group of similar items
  - VARYING Arrays (VARRAY)
  - Nested Tables

Either of these collection types can be treated as object type.
The Collection can consist of

- ✓ Built in data types
- ✓ User Defined data types

What is VARRAY

- VARRAY is basically an array.
- An array is a collection of elements that are of same type
- From 8.0, Oracle supports VARRAYS (Varying length arrays).
- A varying length contains zero or more elements.I.e the array can vary between 0 and maximum size mentioned at the time of creating VARRAY.

What is VARRAY

- We can define a column of a relational table as a VARRAY. From Oracle 8.0, Oracle allows arrays to be stored in tables.
- We can create VARRAY using CREATE TYPE SQL Command
- VARRAY is stored in-line. That is the data of a column of VARRAY type is stored along with the remaining data of the row.

How to Define VARRAY

- Syntax for creating VARRAY is

CREATE TYPE <array-name> AS VARRAY (<limit>)
OF <data-type>

Array-name Name of VARRAY

Limit Max no of elements that an Array contains

Data-type Data type of each element of the array

Example

- The following command create a varying array which can contain upto 20 numbers where each number can contain upto 10 digits.
- Create type phones_array as varray(20) of number(10);
- create table customer(cno number(3),cname varchar2(20),phone phones_array);
- insert into customer values (100,'kris',phones_array(55274020, 9885366959))

Example

```
declare
    ph phones_array;
begin
    select phone into ph from customer where cno=101;
   for  i in 1 .. ph.count
     loop
      dbms_output.put_line(ph(i));
   end loop;
end;
/
```

Limitations

1) The data type of varray can not be
   VARRAY data type
   Other Object Type

2) We can not specify any constraints such as NOT NULL, PRIMARY KEY on VARRAY

3) We can not create index on VARRAY data type column

DML statements on VARRAY

- update customer set phone = phones_array(27676786,null);

EXAMPLE

```
declare
    ph phones_array;
   X NUMBER;
begin
    select phone into ph from customer WHERE CNO=101;
X := PH(PH.PRIOR(PH.LAST));
DBMS_OUTPUT.PUT_LINE(X);
end;
```

EXAMPLE

```
declare
    ph phones_array;
   X NUMBER;
begin
    select phone into ph from customer WHERE CNO=101;
X := PH(PH.NEXT(PH.FIRST));
DBMS_OUTPUT.PUT_LINE(X);
end;
```

EXAMPLE

- declare
- ph phones_array;
- X NUMBER;
- begin
- select phone into ph from customer WHERE CNO=101;
- X := PH(PH.PRIOR(PH.NEXT(PH.FIRST)));
- DBMS_OUTPUT.PUT_LINE(X);
- end;

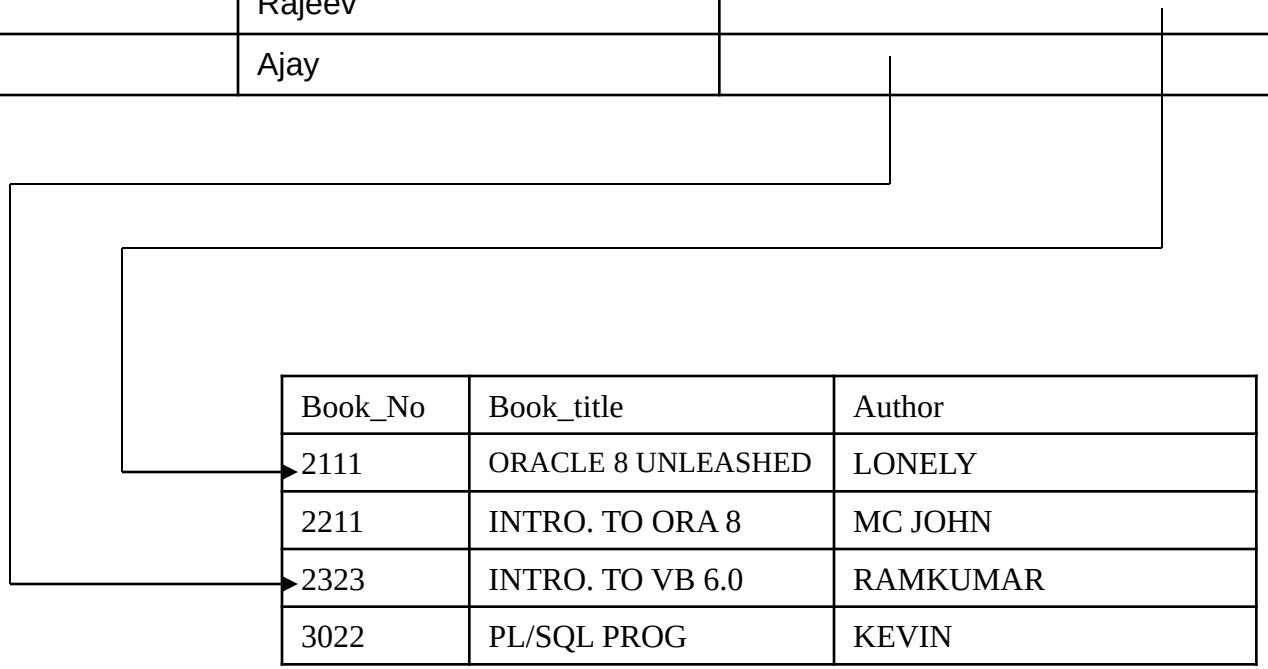
Nested Tables

- It is useful for data models requiring referential integrity and is suitable for master detail and one to many relationship.

- A Nested Table is a database table which stores data in it, that cannot be accessed directly.

Example

Student_No	Student_Name	Books_Issued
01	Rajeev	
02	Ajay	



Book_No	Book_title	Author
2111	ORACLE 8 UNLEASHED	LONELY
2211	INTRO. TO ORA 8	MC JOHN
2323	INTRO. TO VB 6.0	RAMKUMAR
3022	PL/SQL PROG	KEVIN

Nested Table

- A Nested Table can be included in a table definition as one of the columns. That is why they are known as Nested Tables.

- Nested Tables can be manipulated directly using SQL.

Creating nested table

- SQL> CREATE TYPE BOOKS_TYPE AS OBJECT (BOOK_NO NUMBER(4),
  BOOK_TITLE VARCHAR2(20),
  AUTHOR VARCHAR2(20));

SQL> CREATE TYPE BOOKS AS TABLE
OF BOOKS_TYPE;

CREATING NESTED TABLES

- Oracle will implicitly provide a default constructor to create a Nested Table instance with values or an empty Nested Table instance. An empty Nested Table is NOT NULL.
- The Table Type is stored in the data dictionary.

CREATING NESTED TABLE

- SQL> CREATE TABLE STUDENT(
  STUDENT_NO NUMBER(4) NOT NULL,
  STUDENT_NAME VARCHAR2(25),
  BOOKS_ISSUED BOOKS)
  NESTED TABLE BOOKS_ISSUED
  STORE AS BOOK_TABLE;

SQL STATEMENTS ON NESTED TABLE

- SQL> INSERT INTO STUDENT VALUES(1001,’AMIT KUMAR’, BOOKS(BOOKS_TYPE(2111,’ORACLE 8 ‘, ’LONEY’), BOOKS_TYPE(1022,’PL/SQL PROG’,’KEVIN’)));

BOOKS is the constructor method for table type BOOKS and BOOK_TYPE is the constructor method for object type BOOK_TYPE .

In the above example , two rows have been inserted in the Nested Table.

A Pl/SQL program to insert rows into Nested Table

```
DECLARE
    BOOK_VAR BOOKS;
BEGIN
  BOOK_VAR := BOOKS(BOOKS_TYPE(3111,'ORACLE ARCHITECTURE',
'LONEY'));
INSERT INTO STUDENT VALUES(1021, 'ASHOK MARTIN', BOOK_VAR);
END;
```

UPDATION

- DECLARE
- BOOK_VAR BOOKS;
- BEGIN
- BOOK_VAR := BOOKS(BOOK_TYPE(3111, ‘ORACLE ARCHITECTURE’, ‘LONEY’), BOOKS_TYPE(3112, ‘ORACLE IN 21 DAYS’, ‘MOHAN’));
UPDATE STUDENT SET BOOKS_ISSUED = BOOK_VAR WHERE STUDENT_NO = 1021;
END;

DELETE

- SQL> DELETE FROM STUDENT WHERE STUDENT_NO =1021;

example

```
declare
  mbook student.books_issued%type;
  mname student.student_name%type;
cursor c1(sno number) is
  select student_name,books_issued from student where student_no = sno;
begin
    open c1(1001);
    loop
    fetch c1 into mname,mbook;
    exit when c1%notfound;
    dbms_output.put_line(' books issued to ' || mname);
    for i in 1 .. mbook.count
    loop
     dbms_output.put_line('  '||mbook(i).book_title);
    end loop;
end loop;
close c1;
end;
/
```

Updating nested table

- THE operator allows nested tables to be manipulated using DML when it is stored in a table.
- THE table takes sub query as argument and returns the nested table to be used in DML. The sub query must return single nested columns.

example

- update the(select books_issued from student where student_no = 1001) set book_title = 'oracle unleashed' where book_no = 2111;
- Insert into the(select books_issued from student where student_no = 1001) values(books_type(4111,'visual basic','ken fol'))

Example

- delete from the(select books_issued from student where student_no = 1001) where book_no = 2111;