Title

P14. Initialize an associative array with the collection constructor in the PL/SQL block.

Description

An associate array is a collection of key-value pairs that stores a set of elements of the same data type.

An associative array consists of two parts: the key and the value.

The key data type can be integer or varchar2; the value can be a scalar or record data type.

For example, we can create an associative array for a set of employees from a department. The key is the employee ID, and the value is the employee name or the employee record.

To create an associative array, we must define the type of the key and the value first. Then, we can declare an associative array variable of the type and initialize it with the collection constructor (function).

The collection constructor is a function that creates a collection and initializes it with the specified values. Oracle creates the constructor function implicitly for each defined collection type. That is, the name of the constructor function is the same as the name of the collection type.

Let's see an example.

Example

Assume we have a table `emp` with three fields: `emp_id`, `fname`, and `salary`:

```
create table emp (
    emp_id number,
    fname varchar2(50),
    salary number
);
```

Create an associative array

The following code creates an associative array to store rows from the `emp` table.

The key data type is a number, and the value data type is a record with the same structure as the `emp` table.

You need two steps to create an associative array: 1. Define the record type that has the same structure as the emp table. 2. Declare an associative array variable of the record type.

```
set serveroutput on
declare
    type emp_aa_type is table of emp%rowtype index by pls_integer;
    emp_aa emp_aa_type;
begin
    null;
end;
/
```

So far, the associative array emp_aa is empty because we have not initialized it with any values.

Initialize the associative array

There are two ways to initialize an associative array: using assignment statements or the collection constructor.

Before Oracle 18c, we must use assignment statements to initialize an associative array.

Using assignment statements For example, we can initialize the emp_aa associative array with two elements:

```
set serveroutput on
declare
    -- #1 Define the record type and create the record constructor
    type emp_rec is record (
        emp_id emp.emp_id%type,
        fname emp.fname%type,
        salary emp.salary%type
    );

    type emp_aa_type is table of emp%rowtype index by pls_integer;
    emp_aa emp_aa_type;
begin
    -- use assignment statements to initialize the associative array
    emp_aa(1) = emp_rec(100, 'John', 1000);
    emp_aa(1).fname := 'John';
    emp_aa(1).salary := 1000;

    emp_aa(2).emp_id := 200;
    emp_aa(2).fname := 'Doe';
    emp_aa(2).salary := 2000;

    dbms_output.put_line('Employee ID: ' || emp_aa(1).emp_id);
    dbms_output.put_line('Employee Name: ' || emp_aa(1).fname);
```

```
    dbms_output.put_line('Salary: ' || emp_aa(1).salary);

    dbms_output.put_line('Employee ID: ' || emp_aa(2).emp_id);
    dbms_output.put_line('Employee Name: ' || emp_aa(2).fname);
    dbms_output.put_line('Salary: ' || emp_aa(2).salary);
end;
/
```

Using the collection constructor After Oracle 18c (including Oracle 18c), we can use the record constructor and collection constructor to initialize an associative array.

The following code initializes the `emp_aa` associative array with two elements using the collection constructor:

```
set serveroutput on
declare
    -- #1 Define the record type and create the record constructor
    type emp_rec is record (
        emp_id emp.emp_id%type,
        fname emp.fname%type,
        salary emp.salary%type
    );
    -- #2 Define the associative array type and create the collection constructor
    type emp_aa_type is table of emp%rowtype index by pls_integer;

    -- #3 Use the collection constructor to initialize the associative array
    -- #4 Named notation: key => value
    emp_aa emp_aa_type := emp_aa_type(
        1 => emp_rec(100, 'John', 1000),
        2 => emp_rec(200, 'Doe', 2000)
    );
begin
    dbms_output.put_line('Employee ID: ' || emp_aa(1).emp_id);
    dbms_output.put_line('Employee Name: ' || emp_aa(1).fname);
    dbms_output.put_line('Salary: ' || emp_aa(1).salary);

    dbms_output.put_line('Employee ID: ' || emp_aa(2).emp_id);
    dbms_output.put_line('Employee Name: ' || emp_aa(2).fname);
    dbms_output.put_line('Salary: ' || emp_aa(2).salary);
end;
/
```

We use the named notation in the collection constructor. The notation is “key => value”.

For example, to initialize the first element of the `emp_aa` associative array, we use the named notation `1 => emp_rec(100, 'John', 1000)`.

The named notation makes the code more readable and maintainable.

If you want to see more details, please refer to [1].

References

[1] Feuerstein, S. , 2019. Easy Initializing for Records and Arrays, Oracle Connect.