GALGOTIAS UNIVERSITY

School of Computing Science and Engineering

Program: BCA - IOP

Course Code: BCAS3031

Course Name: PL/SQL & Cursors and Triggers

Dr. T. Poongodi

Associate Professor

PL/SQL Records

- A record is a data structure that can hold data items of different kinds.
- Records consist of different fields, similar to a row of a database table.
- To keep track of books in a library and can track the following attributes about each book, such as Title, Author, Subject, Book ID.
- A record containing a field for each of these items allows treating a BOOK as a logical unit and allows to organize and represent its information in a better way.

Types of records –

- Table-based records
- Cursor-based records
- User-defined records

Table-Based Records

- The %ROWTYPE attribute enables a programmer to create table-based and cursor based records.
- The following example illustrates the concept of table-based records. Using the CUSTOMERS table,

```
DECLARE
  customer_rec customers%rowtype;
BEGIN
  SELECT * into customer_rec FROM customers WHERE id = 5;
  dbms_output.put_line('Customer ID: ' || customer_rec.id);
  dbms_output.put_line('Customer Name: ' ||
customer_rec.name);
  dbms_output.put_line('Customer Address: ' ||
customer_rec.address);
  dbms_output.put_line('Customer Salary: ' ||
customer_rec.salary);
END;
```

Customer ID: 5

Customer Name: Hardik

Customer Address: Bhopal

Customer Salary: 9000

PL/SQL procedure successfully completed.

Cursor-Based Records

```
DECLARE
  CURSOR customer_cur is
    SELECT id, name, address
    FROM customers;
  customer_rec customer_cur%rowtype;
BEGIN
  OPEN customer_cur;
  LOOP
    FETCH customer_cur into customer_rec;
    EXIT WHEN customer_cur%notfound;
    DBMS_OUTPUT.put_line(customer_rec.id || ' ' || customer_rec.name);
  END LOOP;
END;
/
```

1 Ramesh

2 Khilan

3 kaushik

4 Chaitali

5 Hardik

6 Komal

PL/SQL procedure successfully completed.

User-Defined Records

- PL/SQL provides a user-defined record type that allows to define the different record structures. These records consist of different fields.
- To keep track of books in a library and to track the following attributes about each book –
- Title
- Author
- Subject
- Book ID

Defining a Record

```
TYPE type_name IS RECORD
  ( field_name1  datatype1  [NOT NULL]  [:= DEFAULT
EXPRESSION],
   field_name2   datatype2   [NOT NULL]  [:= DEFAULT
EXPRESSION],
   ...
   field_nameN  datatypeN  [NOT NULL]  [:= DEFAULT
EXPRESSION);
record-name  type_name;
```

Book record is declared

```
DECLARE
TYPE books IS RECORD
   (title  varchar(50),
   author  varchar(50),
   subject varchar(100),
   book_id   number);
book1 books;
book2 books;
```

Accessing Fields

- To access any field of a record, use the dot (.) operator.
- The member access operator is coded as a period between the record variable name and the field that we wish to access.

```
DECLARE
  type books is record
    (title varchar(50),
    author varchar(50),
    subject varchar(100),
    book_id number);
  book1 books;
  book2 books;
BEGIN
  -- Book 1 specification
  book1.title  := 'C Programming';
  book1.author := 'Nuha Ali ';
  book1.subject := 'C Programming Tutorial';
  book1.book_id := 6495407;

  -- Book 2 specification
  book2.title := 'Telecom Billing';
  book2.author := 'Zara Ali';
  book2.subject := 'Telecom Billing Tutorial';
  book2.book_id := 6495700;
```

```
-- Print book 1 record
   dbms_output.put_line('Book 1 title : '|| book1.title);
   dbms_output.put_line('Book 1 author : '|| book1.author);
   dbms_output.put_line('Book 1 subject : '|| book1.subject);
   dbms_output.put_line('Book 1 book_id : ' || book1.book_id);

   -- Print book 2 record
   dbms_output.put_line('Book 2 title : '|| book2.title);
   dbms_output.put_line('Book 2 author : '|| book2.author);
   dbms_output.put_line('Book 2 subject : '|| book2.subject);
   dbms_output.put_line('Book 2 book_id : '|| book2.book_id);
END;
/
```

Book 1 title : C Programming

Book 1 author : Nuha Ali

Book 1 subject : C Programming Tutorial

Book 1 book_id : 6495407

Book 2 title : Telecom Billing

Book 2 author : Zara Ali

Book 2 subject : Telecom Billing Tutorial

Book 2 book_id : 6495700

PL/SQL procedure successfully completed.

Records as Subprogram Parameters

- Pass a record as a subprogram parameter just as you pass any other variable.
- Can access the record fields in the same way as accessed in the above example –

```
DECLARE
  type books is record
    (title  varchar(50),
    author  varchar(50),
    subject varchar(100),
    book_id   number);
  book1 books;
  book2 books;
```

```
PROCEDURE printbook (book books) IS
BEGIN
   dbms_output.put_line ('Book  title :  ' || book.title);
   dbms_output.put_line('Book  author : ' ||
book.author);
   dbms_output.put_line( 'Book  subject : ' ||
book.subject);
   dbms_output.put_line( 'Book book_id : ' ||
book.book_id);
END;
```

```
BEGIN
  -- Book 1 specification
  book1.title  := 'C Programming';
  book1.author := 'Nuha Ali ';
  book1.subject := 'C Programming Tutorial';
  book1.book_id := 6495407;

  -- Book 2 specification
  book2.title := 'Telecom Billing';
  book2.author := 'Zara Ali';
  book2.subject := 'Telecom Billing Tutorial';
  book2.book_id := 6495700;

  -- Use procedure to print book info
  printbook(book1);
  printbook(book2);
END;
/
```

Book title : C Programming
Book author : Nuha Ali
Book subject : C Programming Tutorial
Book book_id : 6495407
Book title : Telecom Billing
Book author : Zara Ali
Book subject : Telecom Billing Tutorial
Book book_id : 6495700

PL/SQL procedure successfully completed.


Thank You