Functions

By Rahul Barve

Functions

- A function is a subroutine which is similar to a procedure.
- The major difference between a procedure and a function is, a function must always return a value, but a procedure may or may not return a value.

Functions

- Syntax:

```
CREATE [OR REPLACE] FUNCTION function_name
[parameters]
RETURN return_datatype AS
Declaration_section
BEGIN
    Execution_section
    Return return_variable;
EXCEPTION
    exception section
    Return return_variable;
END;
```

Package

By Rahul Barve

Package

- A package is a schema object that groups logically related PL/SQL types, variables, and subprograms.
- Allows to isolate PL/SQL libraries from each other.
- Used to reduce naming conflicts in case of sub-routines like procedures or functions.

Package

- A PL/SQL package is divided into 2 sections:
  - Package Specification
  - Package Body

Package Specification

By Rahul Barve

Package Specification

- A specification is an interface to the package.
- It declares the types, variables, constants, exceptions, cursors that can be referenced from outside the package.
- Holds public declarations, which are visible to the stored procedures and other code outside the package.

Package Body

Package Body

- A package body provides an actual implementation of the subroutines declared in the package specification.
- The implementation is hidden from the code outside the package, enabling encapsulation.

Package Specification Syntax

```
CREATE OR REPLACE PACKAGE
<PACKAGE-NAME> AS
      [PROCEDURE DECLARATIONS]
      [FUNCTION DECLARATIONS]
      [CURSOR DECLARATIONS]
     …..
END <PACKAGE-NAME>;
/
```

Package Body Syntax

```
CREATE OR REPLACE PACKAGE BODY
<PACKAGE-NAME> AS
     [PROCEDURE IMPLEMENTATIONS]
     [FUNCTION IMPLEMENTATIONS]
     [CURSOR PROCESSING]
     …..
END <PACKAGE-NAME>;
/
```

Exception Handling

By Rahul Barve

Exception Handling

- Exception is an error that occurs due to some abnormal situation during the execution of a PL/SQL block.
-

Exception Handling

- When an exception is raised, an error message is generated by oracle engine.
- Catching the exception and taking some action is known as exception handling.

Exception Handling

- PL/SQL exception consists of 3 parts:
  - Type of Exception
  - An Error Code
  - A message

Type of Exception

Type of Exception

- There might be different abnormal situations occurring during the execution of PL/SQL block and hence there are different types of exceptions.

Error Code

By Rahul Barve

Error Code

- Every exception is associated with a unique identification number known as an error code.

Message

By Rahul Barve

Message

- Whenever an exception is raised, it is required to display the error message which will be useful for debugging.
- Hence, for every exception there is an error message associated.

Exception Handling Syntax

```
EXCEPTION
    WHEN <exception-name> THEN
        statements
    WHEN <exception-name> THEN
        statements
    WHEN <exception-name> THEN
        statements
```

Exception Categories

Exception Handling

- In PL/SQL exceptions are categorized as:
  - Named System Exceptions
  - Unnamed System Exceptions
  - User-defined Exceptions

Named System Exceptions

By Rahul Barve

Named System Exceptions

- System exceptions are raised by Oracle, when a program violates any RDBMS rule.
- System exceptions have predefined names given by oracle.

Named System Exceptions

- Not declared explicitly.
- Raised implicitly when predefined oracle error occurs.
- Caught by referencing the standard name.

Named System Exceptions

- CURSOR_ALREADY_OPEN
- INVALID_CURSOR
- NO_DATA_FOUND
- TOO_MANY_ROWS
- ZERO_DIVIDE

Unnamed System Exceptions

By Rahul Barve

Unnamed System Exceptions

- The exception for which oracle does not provide any name is known as unnamed system exception.
- These Exceptions have an error code and an associated error message.

Unnamed System Exceptions

- There are two ways to handle unnamed system exceptions:
  - Using `WHEN OTHERS` exception handler.
  - By associating the exception code to a name and using it as a named exception.

Unnamed System Exceptions

- It’s possible to assign a name to an unnamed system exceptions.
- It is done using a Pragma called as `EXCEPTION_INIT`.

Unnamed System Exceptions

- Syntax:

```
PRAGMA
EXCEPTION_INIT          (<ex-name>,<err-
no>);
```

User Defined Exceptions

By Rahul Barve

User Defined Exceptions

- Sometimes there is a necessity of customizing the exception types based upon the business requirements.
- PL/SQL provides a solution to address this issue by supporting user defined exceptions.

User Defined Exceptions

- Explicitly declared in the declaration section.
- Explicitly raised in the execution section.
- Handled by referencing the user-defined exception name in the exception section.

Exception Handling

- User Defined Exceptions:
  - Declaration:
    - Syntax:

      `DECLARE`

      <exception-name> `EXCEPTION`
  - Raising an exception
    - Syntax:

      `RAISE` <exception-name>

User Defined Exceptions

By Rahul Barve

User Defined Exceptions

- `RAISE_APPLICATION_ERROR` is a built-in procedure in oracle which is used to display the user-defined error messages along with the error number whose range is in between -20000 and -20999.

User Defined Exceptions

- Syntax:

```
RAISE_APPLICATION_ERROR
(<error-no>, <error-message>)
```