PL/SQL

By Rahul Barve

Objectives

- Introduction to Script Files
- Introduction to PL / SQL
- PL / SQL Sections
- Programming Constructs
- Working with Cursors
- Understanding Procedures & Functions
- Understating Packages
- Exception Handling
- Working with Triggers

Script Files

Script Files

- Script files are `.sql` files.
- Simplify managing the database objects.
- Especially used for DDL statements.
- Provide more flexibility.

Script Files

- Script file for table creation (create.sql)

```
create table <first-table-name> (….)/
create table <second-table-name> (….)/
```

- Script file for table removal (drop.sql)

```
Drop table <first-table-name>/
Drop table <second-table-name>/
```

What is PL/SQL

By Rahul Barve

What is PL/SQL

- PL/SQL stands for Procedural Language extension to SQL.
- PL/SQL is a combination of SQL along with the procedural features of programming languages.
- It was developed by Oracle Corporation in the early 90's to enhance the capabilities of SQL.

PL/SQL

- PL SQL consists of blocks of code, which can be nested within each other.
- Each block forms a unit of a task or a logical module.

PL/SQL

- PL/SQL Blocks can be stored in the database and reused.
- PL SQL consists of procedural language constructs such as conditional statements, loops.

Benifits

- Handles errors or exceptions effectively during the execution of a PL/SQL program.
- Once an exception is caught, specific actions can be taken depending upon the type of the exception.

Sections in PL/SQL

By Rahul Barve

Sections in PL/SQL

- The Declaration section (optional).
- The Execution section (mandatory).
- The Exception (or Error) Handling section (optional).

Basic PL/SQL

By Rahul Barve

Basic PL/SQL

- Syntax:

```
DECLARE
    <var-name> <dimension>;
BEGIN
    ---PL/SQL Code
    --Statements
END;
/
```

Variables

By Rahul Barve

Variables

- In PL/SQL, variables are declared using the declaration section.

Variables

- E.g.

```
DECLARE
    price number(5);
    descr varchar2(20);
```

Operators

Operators

- PL/SQL provides several built-in operators:
  - Assignment Operator: `:=`
  - Arithmetic operators: `+, -, *, /, **`
  - Relational operators:

    `=, !=, < >, <, >, <=, >=`
  - Comparison operators:

    `LIKE, BETWEEN, IN, IS NULL`
  - Logical operators: `AND, OR, NOT`
  - Modulus Operator: `MOD`

Conditional Statements

By Rahul Barve

Conditional Statements

- Used for decision making.
- Syntax (Single condition):

```
IF <condition> THEN
    statements
END IF
```

- Syntax (2 Conditions)

```
IF <condition> THEN
    statements
ELSE
    statements
END IF
```

Conditional Statements

- Syntax (Multiple conditions):

```
IF <condition> THEN
   statements
ELSIF <condition> THEN
   statements
ELSE
   statements
END IF
```

Iterative Statements

By Rahul Barve

Iterative Statements

- Used to execute block of statements multiple no of times depending upon certain condition.
- PL/SQL provides 3 types of loops:
  - Simple Loop
  - `WHILE` Loop
  - `FOR` Loop

Simple Loop

By Rahul Barve

Simple Loop

- Used when a set of statements is to be executed at least once before the loop terminates.
- An EXIT condition must be specified in the loop, otherwise the loop will go into an infinite number of iterations.

Simple Loop

- Syntax:

```
LOOP
    statements
    EXIT;
    (or EXIT WHEN <condition>);
END LOOP;
```

WHILE LOOP

By Rahul Barve

WHILE LOOP

- Used when a set of statements is to be executed as long as a condition is true.
- The condition is evaluated at the beginning of each iteration.

WHILE LOOP

- Syntax:

```
WHILE <condition>
LOOP
    statements
END LOOP;
```

FOR Loop

By Rahul Barve

FOR Loop

- Used to execute a set of statements for a predetermined number of times.
- Iteration occurs between the start and end integer values given.

FOR Loop

- Syntax:

```
FOR <var-name> IN <val1>..<val2>
LOOP
    statements
END LOOP;
```

Adding SQL

By Rahul Barve

Adding SQL

- Since PL/SQL is an extension to SQL, it is very frequently required to work upon SQL statements.
- SQL DML statements like `INSERT`, `UPDATE` and `DELETE` remain unchanged in PL/SQL.

Adding SQL

- PL/SQL uses a different syntax for `SELECT` queries.
- E.g.

```
SELECT <column-name(s)> INTO
<variable-name(s)>
FROM <table-name>
```

Using %TYPE

Using %TYPE

- There is a special attribute provided by PL/SQL known as `%TYPE` that is used while declaring variables.
- Allows to declare the type of the variable that is exactly equal to that of the database column.

Using %TYPE

- Syntax:

```
<VAR-NAME> <TABLE-NAME>.<COL-NAME>%TYPE
```

Cursors

By Rahul Barve

Cursors

- Oracle creates a memory area, known as the context area, for processing a SQL statement.
- A cursor is a pointer to this context area.

Cursors

- PL/SQL controls the context area through a cursor.
- A cursor holds the rows (one or more) returned by a SQL statement.

Cursors

- Cursors are divided into 2 types:
  - Implicit cursors
  - Explicit cursors

Implicit Cursors

- Implicit cursors are automatically created when some SQL statement is executed.
- Whenever a DML statement (`INSERT`, `UPDATE` and `DELETE`) is issued, an implicit cursor is associated with this statement.

Implicit Cursors

- For `INSERT` operations, the cursor holds the data that needs to be inserted.
- For `UPDATE` and `DELETE` operations, the cursor identifies the rows that would be affected.

Cursor Attributes

By Rahul Barve

Cursor Attributes

- There are 4 attributes of a cursor:
  - %FOUND
  - %NOTFOUND
  - %ISOPEN
  - %ROWCOUNT

%FOUND

- Returns TRUE if:
  - The DML statements like `INSERT`, `UPDATE` and `DELETE` affect at least one row
  - The `SELECT ….INTO` statement returns at least one row.

%NOTFOUND

- Behaves exactly opposite to that of %FOUND

%ISOPEN

- Returns TRUE if the cursor is already open.

%ROWCOUNT

- Returns the number of rows affected by the DML operations `INSERT`, `UPDATE` and `DELETE` or the number of rows fetched using `SELECT` statement.