1 PLSQL概述

PL/SQL程序的基本单元是块（BLOCK），块就是实现一定功能的逻辑模块。一个PL/SQL程序由一个或多个块组成，也可以嵌套。一个块可以包括三部分，每个部分由一个关键字标识。

语法结构：
[declare]
--声明部分，可选
begin
--执行部分，必须
[exception]
--异常处理部分，可选
end

块各部分的作用解释：
DECLARE：声明部分标志
程序的声明部分（本部分可省略）用于定义常量、变量、游标和用户自定义异常除了程序中隐含定义的变量以外，所有在程序中用到的变量均应在该部分定义。
BEGIN：可执行部分标志
程序的可执行部分（本部分不可省略）用于实现程序的主要功能，可以书写控制结构，也可以插入SQL语句进行数据库的访问与操作。
EXCEPTION：异常处理部分标志
程序的异常部分（包含在可执行部分中）用于书写程序发生错误时的处理动作代码，如果没有对相应的错误进行处理，会显示系统定义错误信息。
END;：程序结束标志

2 数据类型

2.1 基本数据类型

2.1.1数值类型

数值类型主要包括number、pls_integer和binary_integer三种基本类型，其中，number类型的变量可存储整数或浮点数，而pls_integer和binary_integer类型的变量只存储整数。

number类型还可以通过number（p，s）的形式格式化数字，其中p表示精度，s表示刻度范围。精度是指数值中所有有效数字的个数，而刻度表示小数点右边小数位的个数。

2.1.2 字符类型

字符类型主要包括varchar2、char、long、nchar和nvarchar2，这些类型的变量主要用来存储字符串或字符数据。下面分别介绍：

- varchar2：变长字符，最大可以是32767字节，数据库类型的最大程度是4000字节；
- char：定长字符，最大可以是32767字节，数据库类型的最大程度是2000字节；
- long：变长字符，最大可以是32767字节，数据库类型的最大程度是2G；
- nchar和nvarchar2：这两种数据类型的长度要根据各国字符集确定，只能具体情况具体分析；

2.1.3 日期类型

日期类型只有一种，即date类型。

2.1.4 布尔类型

布尔类型也只有一种，即boolean类型，主要用于程序的流程控制和业务逻辑判断。

2.2 特殊数据类型

为了提高用户的编程效率和解决复杂的业务逻辑需求，PL/SQL语言除了可以使用Oracle规定的基本数据类型外，还提供了3中特殊的数据类型，但这3中类型仍然是建立在基本数据类型基础之上的。

2.2.1 %type类型

使用%type类型关键字可以声明一个与指定列名称相同的数据类型，通常紧跟在指定列名后面；

2.2.2 record类型

记录类型，可以存储由多个列值组成的一行数据，在声明记录类型变量之前，首先需要定义记录类型，然后才可以声明记录类型的变量，记录类型是一种结构化的数据类型，使用type语句进行定义。

2.2.3 %rowtype类型

%rowtype类型结合了以上两种类型的优点，它可以根据数据表中行的结构定义一种特殊的数据类型，用来存储从数据表中检索到的一行数据。

3 定义变量和常量

3.1 定义变量

变量是指其值在程序运行过程中可以改变的数据存储结构，定义变量必须的元素就是变量名和数据类型，另外还有可选择的初始值，语法格式如下：
<变量名> <数据类型> [(长度)：=<初始值>];

例如：v_name varchar2(100):='China';

3.2 定义常量

常量是其值在程序运行过程中不可改变的数据存储结构，定义常量必须的元素包括常量名、数据类型、常量值和constant关键字，语法格式如下：
<常量名> constant <数据类型> :=<常量值>;

例如：c_name constant varchar2(100)：=’ China ‘;

4 PL/SQL程序块举例

4.1 使用常量、变量和%type

```
DECLARE
  C_COURSE VARCHAR2(10) := 'OCP';
  V_ENAME  VARCHAR2(20);
  V_JOB    scott.emp.job%TYPE;

BEGIN
  SELECT T.ENAME, T.JOB
    INTO V_ENAME, V_JOB
    FROM SCOTT.EMP T
   WHERE T.EMPNO = 7369;

   DBMS_OUTPUT.PUT_LINE('Cource is '||C_COURSE);
   DBMS_OUTPUT.PUT_LINE('Name of 7369 is '||V_ENAME);
   DBMS_OUTPUT.PUT_LINE('Job of 7369 is '||V_JOB);

EXCEPTION
  WHEN OTHERS THEN
    DBMS_OUTPUT.PUT_LINE('Error ! Please check .');
END;
```

4.2 使用record类型的数据类型

```
DECLARE
  TYPE REC_EMP IS RECORD(
    ENAME SCOTT.EMP.ENAME%TYPE,
    JOB   VARCHAR2(9));

  V_EMP REC_EMP;

BEGIN
```

```
  SELECT T.ENAME, T.JOB INTO V_EMP FROM SCOTT.EMP T WHERE T.EMPNO =
7369;

  DBMS_OUTPUT.PUT_LINE('Name of 7369 is ' || V_EMP.ENAME);
  DBMS_OUTPUT.PUT_LINE('Job of 7369 is ' || V_EMP.JOB);

EXCEPTION
  WHEN OTHERS THEN
    DBMS_OUTPUT.PUT_LINE('Error ! Please check .');
END;
```

4.3 使用%rowtype类型

```
DECLARE
  V_EMP scott.emp%Rowtype;

BEGIN
  SELECT T.* INTO V_EMP FROM SCOTT.EMP T WHERE T.EMPNO = 7369;

  DBMS_OUTPUT.PUT_LINE('Name of 7369 is ' || V_EMP.ENAME);
  DBMS_OUTPUT.PUT_LINE('Job of 7369 is ' || V_EMP.JOB);

EXCEPTION
  WHEN OTHERS THEN
    DBMS_OUTPUT.PUT_LINE('Error ! Please check .');
END;
```