分析Oracle_OracleForms中多用途代码 PDF转换可能丢失图片或格式，建议阅读原文 https://www.100test.com/kao_ti2020/207/2021_2022__E5_88_86_E6_9E_90Orac_c102_207494.htm 几年前，当Oracle放弃客户端的Forms的时候，随之消失的那些内置的函数中有一项是关于向警告消息函数(alert message function)传递参数。如果你处理错误或者缺失的输入参数(你应该这么做)，Forms迁移过程的一部分是将这个内置的函数从6i版本改成9i版本。复杂的应用可能包含有上千条警告消息，并且一个主要的应用(即Forms)的变化会导致上千条改变。做这样的改变的确是一件讨人嫌的行为。 另一方面，作为一个使用PL/SQL的DBA和程序员，你到底写过多少次DBMS_OUTPUT.PUT_LINE()?必须写的或者敲入的DBMS_OUTPUT.PUT_LINE变得非常无聊，使用方便的、内置的短小的代码不是更好吗?可能并不是经过深思熟虑，但是更多的归咎于好运或者意识到同样的东西必须敲一遍又一遍的现实，机灵的Forms程序员们创建了自己的内置函数，采用了过程的方式来产生警告消息。相同的原理可以被用在你日常的PL/SQL代码中。事实上，你可以创建一个小的消息库管理很多类型的输出消息。让我们看看一些这样的可能性。 一个简单的警告消息过程 就像在这一章节标题中表示的那样，第一个方法是非常简单的。假设你有一个常见的需求要输出某个过程、函数或者代码块更新的记录个数。让我们假定被更新行的个数是46。使用下面的过程之后，一个简单的“ am(46). ”语句就可以你需要的输出： CREATE OR REPLACE procedure am (msg number) as begin dbms_output.put_line(Records 0updated: ||msg). end. / 另一个版

本可以处理字符串类型，因此对“ ams(your message here). ”的调用显著的降低了你敲入的次数。当调试或者解决问题的代码中，有这样一个简单的内置函数对输出“ where am I in the code ”的语句是非常宝贵的。位置报告可以确认，比如，你进入了IF-THEN-ELSE语句中哪个分支。假如你的问题代码调用了很多次其他的对象(过程、函数等等)，输出像“ calling function X ”或者“ returned from function X ”这样的状态信息可以确认过程流。最终，另外一种使用情况是报告数值。你可以报告或者跟踪一个变量的值是如何被改变的。 建立一个警告消息库 当然，你的消息库的复杂性和灵活性完全取决于你。假如你的(输出)消息是简单的，那么保持函数过程简单。更准确的讲，保持函数过程的个数是最少的。只要两个简单的过程，ams和amn，就可以用来输出基于字符串和数值的消息了。 假如你需要让输出的文字内容根据运算的输出有所变化，比如DML语句的输出，那么你可能需要三个新的内置过程(插入、更新和删除运算各一个)。可能你想说明删除的类型或者原因。比如一个批处理作业的某一步是计算重复记录的个数。那么像“ Records counted: 46 ”这样的输出是足够有用的，但是在这种情况下，“ Duplicates counted: 46 ”会显得更有效。因此，我们增加了2个新的内置过程。 这样，我们现在有了至少6个不同的过程。现在，管理性的问题应该比较明显了。我们寻找一些简单的，但是同时又是健壮的过程。至少有两种方式可以用于重新简化需要的功能。一种方法是让警告消息过程能够接收两个输入参数。另外一种方法，正是我准备介绍的，是把这些过程打包。 增加输入参数的个数 再说一遍，假如前面的简单方法可以满足了你的要求，那

么就没必要继续深入了。创建有两个输入参数的过程，第一个参数是消息文字或者说基础，第二个参数可以是输出、位置、状态或者数值。如果你关注数据类型的转化，那么这两个输入参数的组合text/text和text/number都可以统一成text/text类型。你的确必须做这样的转换吗?不，但是为了和你已有的保持一致，如果你在别的地方做了类型映射，那么这里也进行类型映射。不管这些，下面的例子显示了第一种方法的灵活性。 CREATE OR REPLACE procedure am (msg1 varchar2, msg2 varchar2) as begin dbms_output.put_line(msg1||msg2). end. / 编译之后，下面是使用的例子。 SQL> set serveroutput on SQL> exec am(Here I am,46). Here I am46 PL/SQL procedure successfully completed. 好了，这次输出本可以看起来更好一些(注意输出中msg1和msg2中没有空格)。我们到这里就必须格式化一个或者全部两个消息输入让输出好看一些。然而，假如美观不需要关心，那么创建基于像 (dupes ,46)这样的输入的消息，这样做也是非常简单的，虽然你需要处理空格或者格式化。那么这种方法是否有缺陷呢?这就看情况了。假如你需要的仅仅是msg1，而不需要msg2，怎么办?当创建这个过程，需要允许msg2是null值。显然，msg1不需要这样，对吧? CREATE OR REPLACE procedure am (msg1 varchar2, msg2 varchar2 default null) as begin dbms_output.put_line(msg1||msg2). end. / Procedure created. SQL> exec am(Where am I?). Where am I? PL/SQL procedure successfully completed. 100Test 下载频道开通，各类考试题目直接下载。详细请访问 www.100test.com