“ 优秀示例 ”:Oracle代码规程 PDF转换可能丢失图片或格式，建议阅读原文 https://www.100test.com/kao_ti2020/253/2021_2022__E2_80_9C_E4_BC_98_E7_A7_80_E7_c102_253964.htm 且不说Oracle的第三代语言(3GLs)，如果对所有SQL和PL/SQL的优秀示例做出完整的论述，大概需要写成一本书事实上，市面上已经有好几本这样的书出版了，在这里我只会列举一些比较“ 重量级 ”的Oracle优秀示例。 在PL/SQL中使用阵列处理是一个很好的做法(如，使用bulk collect和forall)。批量处理能够大大减少PL/SQL语句执行引擎的环境切换次数，从而提高其性能。另一个优秀示例是把存储过程中的所有代码放入锁定的软件包中，这样可以生成模块单元。把存储过程放入软件包里可以实现相关程序和功能的分组。当单个包被使用，整个软件包都会载入内存中(内存会启动整个软件包?)，把磁碟存取时间减到最少。通过这个方法我们同样可以把整个应用程序包载进内存中，防止发生重新载入和代码解析，从而减少严重影响性能的代码递归。 PL/SQL(和SQL)的另一个优秀示例是使用适当的变量类型(当你需要NUMBER时不要使用VARCHAR2，反之亦然)。使用不适当的变量(用character跟number进行比较)会导致无用索引。保证变量类型正确的一种方法就是使用%TYPE 和%ROWTYPE。 还有就是永远使用DBMS_PROFILER或使用像Quest Software的Quest Code Tester工具来验证循环逻辑。DBMS_PROFILER是Oracle提供的一个软件包，能够使你的代码生成对每行执行时间及所需时间的追踪。你可以验证循环执行次数应为最少。 你同样应该验证适当的IF-THEN-ELSE结构。我的意思是你应该把最常用

的选项放在前面(比如exit test)。这个方法同样使用于CASE结构。 PL/SQL(以及Java、C、C 和其他所有Oracle相关的3GLs)中，另一个更为重要的优秀示例是首先调优SQL。即使是世界上设计最精密的程序，如果其中含有的SQL很差的话，运行起来性能也会不好。Quest的SQL Optimizer、Performance Analyzer、TOAD和SQL Navigator都能够优化SQL。 在测试PL/SQL-SQL代码的时候，应利用匿名PL/SQL块来保证处理环境的相似性。如果你在一个标准的SQL环境里用文字代替绑定变量来进行测试，你所得到的执行计划会不同于当你使用匿名PL/SQL块和绑定变量时得到的执行计划，因此你的调整很可能不能获得很好的结果。使用Quest Code Tester能够保证你能获得你想要的结果。 还有一个良好的编程做法是很好地利用临时表和PL/SQL索引表。不正确地使用 ”normal ”表进行临时储存会降低性能，因此应使用临时表。而当内存空间足够运行PL/SQL表时使用临时表也会大大降低性能。你也可以利用视图来创建中间结果的缓存表。 还有一个很简单易行的优秀示例是在IN OUT和OUT变量上利用NOCOPY。在一个使用IN OUT或OUT的PL/SQL程序中，如果NOCOPY关键字没有包含在其页首的变量声明中，所有的变量都会通过copyout程序传值(传值而非传引用)。而使用NONCOPY，所有通过引用传值，这样做会大大提高运行性能。 最后，还要求所有SQL开发者生成解释执行计划，然后再经由同行评审。在较大的开发计划里，数据库管理员不可能检查所有的代码看是否获得最佳的性能，因此开发者首先对SQL代码性能进行调优是很必要的。只要开发者把SQL的索引功能调优，调整为正确的逻辑和技术，数据库管理员便能够对难以处

理的问题代码进行更高级的调优。 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