有关Oracle中虚拟专用数据库的探讨 PDF转换可能丢失图片或格式，建议阅读原文 https://www.100test.com/kao_ti2020/143/2021_2022__E6_9C_89_E5_85_B3Orac_c102_143754.htm 虚拟专用数据库 (VPD) 也称为细粒度访问控制，它提供强大的行级安全功能。它是在 Oracle8i 中推出的，已经受到广泛的欢迎，并且在从教育软件到金融服务等各种应用程序得到采用。 VPD 的工作方法是，通过透明地更改对数据的请求，基于一系列定义的标准向用户提供表的局部视图。在运行时，所有查询都附加了谓词，以便筛选出准许用户看到的行。例如，如果只允许用户查看帐户管理员 SCOTT 的帐户，则 VPD 设置自动地将查询： 0select * from accounts. 重写为： 0select * from accounts. where am_name = SCOTT. DBA 在表 ACCOUNTS 上设置了一项安全策略。该策略具有一个相关函数，称为policy function，它返回一个用作谓词的字符串 where am_name = SCOTT。如果您不熟悉该特性的全部功能，我建议您阅读 Oracle 杂志的文章“ 利用 VPD 保持信息的私密性 ”。 策略类型 生成谓词所需的重复分析是一种在某些情况下可以进行修整的开销。例如，在大部分实际情况中，谓词并不象 am_name = SCOTT 那样是静态的；它基于用户的身份、用户的权限级别、用户向哪个帐户管理员进行报告等情况,可能更具有动态性。由策略函数创建并返回的字符串可能会具有很强的动态性，而为了保证其结果，Oracle 必须每次重新执行策略函数，既浪费资源又降低性能。在这种类型的策略中，谓词每次执行时可能会有很大的差别，该策略称为“ 动态 ”策略，在 Oracle9i 数据库以及以前的版本中已经提供了这种策略。 除了保留动态策

略之外，Oracle 数据库 10g 还基于谓词的构造推出了几种新类型的策略，为提高性能提供了更好的控制：context_sensitive、shared_context_sensitive、shared_static 和 static。现在，让我们来了解每种策略类型的意义以及如何在适当的场合中使用它们。 动态策略 为保持向后兼容性，10g 中的默认策略类型为“ dynamic ” 正如 Oracle9i 中一样。在这种情况下，对于每行以及每位用户，在每次访问表时都对策略函数进行重新求值。让我们来详细分析策略谓词： where am_name = SCOTT 忽略掉 where 子句，谓词就具有两个不同的部分：在等式操作符之前的部分 (am_name) 和等式操作符之后的部分 (SCOTT)。在大多数情况下，后面的部分更象是变量，因为它是由用户的数据提供的（如果用户是 SCOTT，则其值为 SCOTT）。在等号前面的部分是静态的。因此，即使函数不必为生成适当的谓词而对每行求出策略函数的值，由于了解前面部分的静态性以及后面部分的动态性，也可以提高性能。在 10g 中，可以在 dbms_rls.add_policy 调用中使用 "context_sensitive" 类型的策略作为参数来实现这种方法： policy_type => dbms_rls.context_sensitive 在另一个示例中，我们有一个称为 ACCOUNTS 的表，它拥有几列，其中一列是 BALANCE，表示帐户余额。假设允许某个用户查看低于某特定余额的帐户，而该余额由应用程序上下文所决定。我们并不在策略函数中将此余额值固定，而是3是根据应用程序上下文确定，如： create or replace vpd_pol_func (p_schema in varchar2, p_table in varchar2) return varchar2 is begin return balance 应用程序上下文 VPDCTX 的属性 MAXBAL 可以在会话的前期设定，而函数在运行时可以容易地获得该数值。 请

仔细注意该示例。谓词有两部分：小于号之前的部分和之后的部分。之前的部分是“ balance ”一词，它是文字符。后面的部分从某种程度而言是静态的，因为应用程序上下文变量在改变之前一直是常量。如果应用程序上下文属性不变，则整个谓词是常量，因此不需要重新执行函数。如果策略类型定义为对上下文敏感，则 Oracle 数据库 10g 可以识别此情况以用于优化。如果在会话期间没有发生会话上下文的变化，则不重新执行该函数，从而显著提高了性能。 静态策略 有时业务操作可以确保谓词更加静态。例如，在上下文敏感的策略类型示例中，我们将用户所见的最大余额定义为一个变量。当 web 应用程序中的 Oracle userid 由许多 web 用户共享，并且应用程序基于这些用户的权限来设置该变量（应用程序上下文）时，这种方法很有用。因此，web 用户 TAO 和 KARTHIK 都是以用户 APPUSER 连接到数据库的，二者可以在其会话中拥有两个不同的应用程序上下文的值。此时 MAXBAL 的值并不依赖于 Oracle userid，而是依赖 TAO 和 KARTHIK 各自的会话。 在静态策略的情况下，谓词更具有可预测性，其说明如下。 LORA 和 MICHELLE 分别是 Acme Bearings 和 Goldtone Bearings 的帐户管理员。当他们连接数据库时，他们使用自己的 id，并且只应该看到属于他们的那些行。在 Lora 方面，谓词变成 where CUST_NAME = ACME；而对于 Michelle，则是 where CUST_NAME = GOLDTONE。在这里，谓词依赖于他们的 userid，因此他们所创建的任何会话在应用程序上下文中始终具有相同的值。