

Oracle数据库10g性能概述

Oracle 白皮书
2004 年 7 月

- 引言..... 3
- 网格计算数据库..... 3
- 优化资源用量..... 3
 - 自我调整功能..... 3
 - 自我调整 SGA..... 3
 - 自我调整检查点..... 4
 - 优化 PL/SQL..... 5
- 丰富的查询处理技术..... 6
 - 基于成本的查询优化..... 6
 - SQL 语句转换..... 6
 - 执行计划选择..... 6
 - 成本评估..... 7
 - 动态运行时优化..... 8
 - 全表扫描..... 9
 - 索引编排表的列表分区选项..... 9
 - 概要管理..... 9
 - 实体化视图的扩展支持..... 9
 - 增强型查询重写..... 10
- 大量数据的管理..... 10
 - 大量分区..... 11
 - 散列分区全局索引..... 12
 - 并行单独游标..... 12
 - 高速数据移动..... 12
- 基于 Microsoft WINDOWS 的系统..... 14
- 性能方面新特性的总结..... 15
- 附录 1:例子的详细信息..... 16
 - 查询优化程序：“CPU+IO”模型..... 16
 - 例 1..... 16
 - 例 2..... 17
 - 全表扫描..... 17
 - 大量分区..... 18

引言

Oracle 数据库支持世界上的许多大型信息系统,例如 UK Inland Revenue 的国家税务管理系统, France Telecom 的 10 TB 数据仓库以及 ABB 的巨型 SAP R/3 安装。以 Oracle6 中多个版本的阅读一致性开始,可以发现它的每个版本都为改进数据库性能和可伸缩性引入了具有创新性的特性。而且每个版本都可以在当时可用的所有主要平台上基本不需更改地运行。此外, Oracle 数据库 10g 通过新的性能特性以及数据库优化保持了它的数据库性能领先的记录,同时扩展了 Oracle 数据库的平台覆盖面,包含了 64 位 Windows 和 Linux 的版本。本书着重介绍 Oracle 数据库 10g 与性能和可伸缩性相关的新特性。

网格计算数据库

网格计算将显著改变计算的经济性。就最高层次而言,网格计算的基本思想是用户不需关心数据的位置,也不需考虑哪台计算机处理他的请求。计算应该是一种日常用品,就像电力网或电话网络一样。换言之,网格计算主要是供应硬件和软件资源。从严格的技术观点来看,供应并不一定获得性能上的提高。然而,从业务的角度而言,供应将给予用户更好的性能。因为资源可以根据业务优先级或需求供应给适当的应用程序,所以利用相同的资源,用户能够获得更高的性能。

网格计算的思想与 Oracle 已经开发多年的功能和技术相结合,而且 Oracle 数据库 10g 具有适合于发布未来网格计算技术的体系结构。

优化资源用量

为了进一步保证资源用量的优化, Oracle 数据库 10g 包含了强大的内置工具和自我管理的功能,它可以自动并动态地管理大部分的配置并调整可用资源。

自我调整功能

Oracle 数据库 10g 引入了许多自我调整的功能,这些功能动态调整数据库参数,以便利用系统资源的消耗而出现的变化。

自我调整 SGA

系统全局区 (SGA) 是存储区域,由所有用户共享,它包含某个特定 Oracle 数据库实例的数据和控制信息。SGA 内部划分为几个存储组件,分别代表为满足每类内存分配需求而使用的内存池,例如存储最近使用的数据块,或记录数据库的变化等需求。

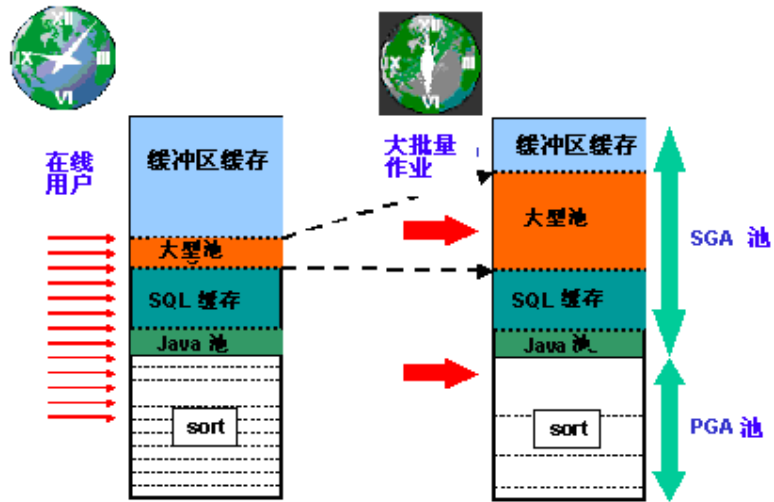


图 1: 自动共享内存调整

即使在有咨询机构帮助的情况下，通过调整这些缓存的大小而优化性能也并不简单。总会有风险，或者是为组件分配的缓存大小过小而导致内存分配失败，或者是为组件分配的缓存大小过大，而浪费内存，这些内存本可用于其他缓存。

Oracle 数据库 10g 引入了自我调整 SGA，它允许管理员只指定 SGA 的总大小，而由 Oracle 数据库负责在整个 SGA 池中内部决定优化内存分配的责任。有了这项新功能，分配给不同 SGA 缓存的内存将随着时间动态更改，以适应工作量的变化。

与 Oracle9i 中引入的参数 PGA_AGGREGATE_TARGET 一起，这项功能使 Oracle 数据库可以自动和动态地调整内存消耗，以便更改工作量分配，保证优化内存用量。

自我调整检查点

检查点是将内存中修改的数据与数据库中的数据文件同步的手段。Oracle 定期将检查点之间修改过的数据写入数据文件，这种做法的要求之一就是需要足够的可用内存，以提高为将要进行的操作寻找空闲内存的执行性能。

在 Oracle 数据库 10g 之前，管理员可以通过设置检查点相关初始化参数 (FAST_START_MTTR_TARGET) 来指定预期崩溃恢复时间 (MTTR)。他们可以通过使用 MTTR 顾问来完成以上操作。MTTR 顾问有助于预报由不同 MTTR 目标值引发的物理写入数量。从 Oracle 数据库 10g 开始，数据库可以自我调整检查点以获得对于常规吞吐量影响较低的良好恢复时间。使用自动检查点调整，Oracle 数据库利用 I/O 用量较低的时期将内存中修改

的数据写入到数据文件中，从而对吞吐量不会造成负面影响。因而，即使管理员未设置任何检查点相关参数，或者该参数设置为一个非常大的值，数据库依然可以获得一个合理的崩溃恢复时间。

优化 PL/SQL

PL/SQL 是 Oracle 的过程语言，由 SQL 扩充而来。它将 SQL 的简单和灵活与结构化程序语言的过程功能相结合。PL/SQL 代码可以集中存储在数据库中。

使用 PL/SQL 存储过程可以改善性能并优化内存用量，这是因为：

- 应用程序和数据库之间的网络流量减少了。
- 过程的编译形式在数据库中已经存在，所以在执行过程中不再需要编译过程。
- 多位用户可以共享内存中的一份过程。

通过使用 PL/SQL，Oracle 数据库 10g 在性能方面有了重大改进。

PL/SQL 编译器经过重新编写，它为高效而持续地优化计算密集型 PL/SQL 程序提供了框架。新的编译器包括一个更加成熟的代码生成器和一个可以充分改善大部分程序的性能的全局代码优化器。其结果是提升了性能，尤其是计算密集型 PL/SQL 程序，对于一个单纯的 PL/SQL 程序来说，其性能比 Oracle9i Database Release 2 超出大约 2 倍。

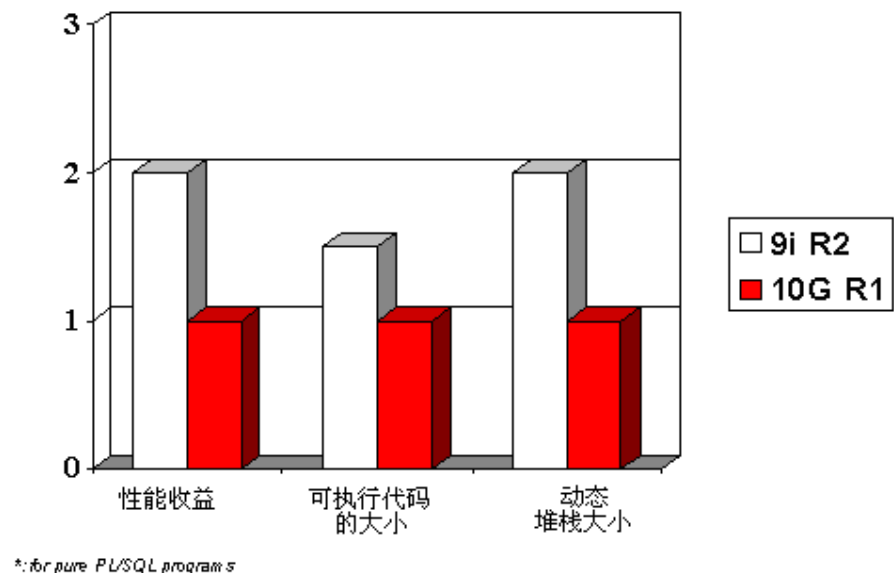


图 2: PL/SQL 改进因子, Oracle 内部测试

而且，PL/SQL 可执行代码的大小缩减到 30%，而动态堆栈大小缩减 2 倍。这些大小的缩减全面改进了性能、可伸缩性以及可靠性，因为 PL/SQL 程序的执行减小了内存的压力，从而改善了整个 Oracle 系统的性能。

PL/SQL 编译时警告自动识别 PL/SQL 结构的合法类也是 Oracle 数据库 10g 有助于管理性能的新功能，但它可能导致运行时性能下降。

Oracle 数据库 10g 还除去了一些存在于 Oracle9i 数据库中的 PL/SQL 本地执行的限制。PL/SQL 程序的本地执行提供了将 PL/SQL 模块编译到本地代码的功能，并提供了几项性能优势：首先，它消除了与解释字节代码相关的成本；其次，本地代码中的控制流程和异常处理要比在解释的代码中快得多。结果 PL/SQL 程序的执行速度大大提升了。

丰富的查询处理技术

Oracle 数据库提供了多种可以满足非常复杂环境的要求的查询处理技术。这些成熟的技术包括：

- 基于成本的查询优化，以获得有效的数据访问
- 为各类的应用程序定制的索引技术和模式对象。
- 摘要管理功能

Oracle 数据库 10g 引入了关于这些功能的几项改进和扩展。

基于成本的查询优化

查询优化对于关系数据库的性能，特别是对于执行复杂 SQL 语句的性能而言至关重要。Oracle 数据库 10g 只使用基于成本的优化策略。使用基于成本的优化，对于一个给定查询生成多个执行计划，并对每个计划计算估计成本。然后查询优化程序选择最佳计划，即估计成本最低的计划。此查询优化的过程对于应用程序和最终用户是完全透明的。

由于应用程序可能生成非常复杂的 SQL 语句，查询优化程序必须精心构建、功能强大，以保障良好的执行性能。由于 Oracle 数据库的查询优化程序的成本模型精确完整，以及它用来决定为特定查询访问指定数据的最有效手段的技术和方法，Oracle 数据库的查询优化程序可以产生卓越的执行计划。

SQL 语句转换

优化程序将最初的 SQL 语句转换成可以返回相同结果、但可以更有效地处理的 SQL 语句。可以随时应用试探查询转换，例如视图合并或谓词下推，因为它们总是通过大大减少需要扫描、连接或聚合的数据量来提高查询性能。在基于优化程序的成本估算的，使用多种技术（例如实体化视图重写或星型转换）的转换查询决定中，Oracle 数据库也会应用基于成本的查询转换。

执行计划选择

执行计划描述了 SQL 处理的所有执行步骤，如访问表的顺序；如何将这些表连接在一起；以及是否通过索引来访问这些表。

Oracle 数据库提供了非常丰富的数据库结构、分区和索引技术，以及连接方法的选择。其并行执行体系结构事实上允许任何 SQL 语句以任何等级的并行度执行。而且，查询优化程序考虑提示，它是用户驱动的建议，并作为注释写在 SQL 语句中。

结果对于给定 SQL 语句，优化程序可以生成许多不同的计划，因为有多种不同的访问路径、连接方法和连接顺序的组合方式，可以用不同方法访问和处理数据，并产生相同的结果。

成本评估

要评估这些执行计划的成本，并选择其中成本最低的计划，优化程序依赖于组成 SQL 语句执行的单独操作的成本评估。这些评估要尽可能精确，而且 Oracle 数据库集成了一个非常成熟的成本模型，包含关于 Oracle 数据库数据结构的详细内容、对象级和系统统计数据访问方法，以及性能信息：

- 对象级统计数据收集关于数据库中对象（表、索引以及实体化视图）的信息，例如 B 树索引的级数或表的列中不同值的数目（基数）。数据值直方图也可用于得到准确的列数据分布的估计值。
- 系统统计数据描述在典型工作量情况下的活动中收集到的硬件组件（处理器、内存、存储器和网络）的性能特性。

在 Oracle 数据库 10g 中，默认的成本模型是“CPU+IO”。为了估算给定查询的执行时间，查询优化程序估算 IO 操作的数目和类型，以及在查询执行过程中数据库可能执行的 CPU 的周期数。然后它用系统统计数据将这些 CPU 周期数和 IO 操作转换为执行时间。使用改善的成本模型将获得更好的执行计划，从而改进了一些类型查询的性能。在这些情况下，与“IO”模型相比改进因子可达到 77%（消耗时间）。

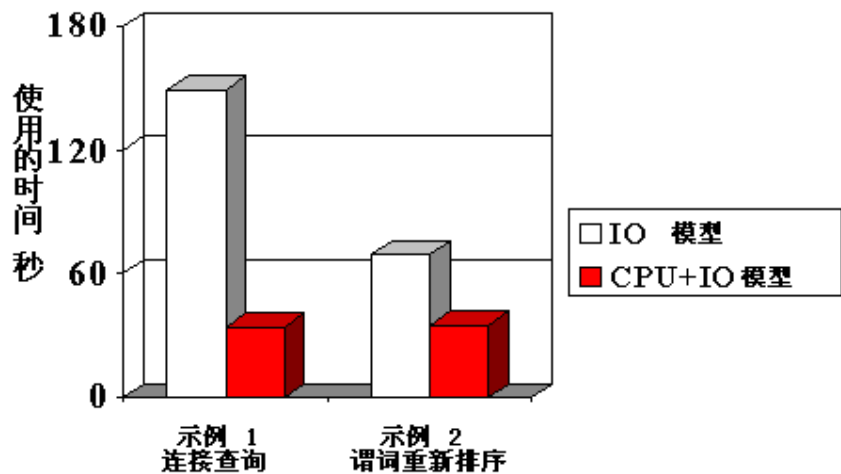


图 3: 使用“CPU+IO”成本模型使性能改进的例子

图 3 所阐述的第一个例子是在 Oracle 实例计划中的销售和产品表中执行连接查询。查询检查销售和产品表间的参考完整性限制是否加强了。使用 IO 成本模型时，查询优化程序只会考虑扫描销售表的成本，并选择嵌套循环作为执行计划。而使用 CPU+IO 成本模型时，内存中执行的操作的成本也会被考虑，因而，优化程序会选择散列连接。结果可获得更好的执行计划，以及更好的性能。

“CPU+IO”成本模型的另一个潜在好处是在查询时可以重新排序谓词。谓词重排只有在每个谓词的成本都可以估算时才会进行，而这只有在

“CPU+IO”成本模型中才可能，因为在大多数情况下，谓词的成本只包含 CPU 周期。第二个例子中使用的查询有两个谓词，其中执行第一个谓词的成本要高得多。使用 IO 成本模型时，谓词按照原始查询顺序执行。而使用 CPU+IO 成本模型时，谓词被重排，以便成本较低的谓词首先执行。采用这种重排，第二个谓词的执行忽略了很多不符合第一个条件的行，从而获得了更好的性能¹。

收集统计数据和执行信息的过程应该既高效又高度自动化，所以 Oracle 使用了很多可以使这个过程自动化并快速的功能。

Oracle 数据库 10g 引入了自动统计数据收集。包含失效的统计数据或没有统计数据对象被自动分析，管理员从跟踪哪些需要分析、哪些不需要分析的任务中解放出来，而只按照需要进行分析。全自动统计数据收集随之改进了 SQL 的执行性能。

Oracle 数据库通过检查相关的数据样本的取样方法来收集统计数据。取样可能是静态的或动态的，与查询在同一个事务处理中发生，可以与并行运行结合使用。Oracle 数据库的统计数据收集程序自动决定恰当的抽样百分比和适当程度的并行运行，根据基础表的数据特性确定。Oracle 数据库也可以隐式决定哪些列需要直方图（用于获得列数据分布的准确估算）。

用户可以通过设置优化程序的方法和目标来影响优化程序的选择。Oracle 数据库提供两种优化程序模式。第一种模式将返回查询前 n 行的时间最小化。这种模式与应用程序（例如操作系统）相关，它的目标是获得返回最先几行的最佳反应时间。第二种模式将返回查询所有行的时间最小化，它的目标是获得最佳吞吐量。

动态运行时优化

由于 SQL 执行的所有方面并非都可以在事先进行最佳计划，Oracle 数据库根据业务优先权和当前数据库工作量和硬件性能对其查询过程策略进行运行时动态调整。该动态优化的目标是获得优化的执行性能，即使每个查询可能不能够获得理想的 CPU 或内存资源。

Oracle 数据库自动调整查询的并行度，动态为每个查询分配恰当数量的内存，并使用资源管理器根据资源计划中的指示在查询之中分配资源。

结果是使查询优化程序获得更为精确的成本和大小模型。这有助于优化程序产生更好的执行计划，改进查询性能。

¹ 关于例子的更多详细信息请参阅附录 1

全表扫描

在 Oracle 数据库 10g 中，全表扫描的性能显著改进了。使用简单谓词或者没有谓词的情况下对单个表进行扫描的消耗时间和 CPU 利用率的改进因子在 40% 和 60% 之间。

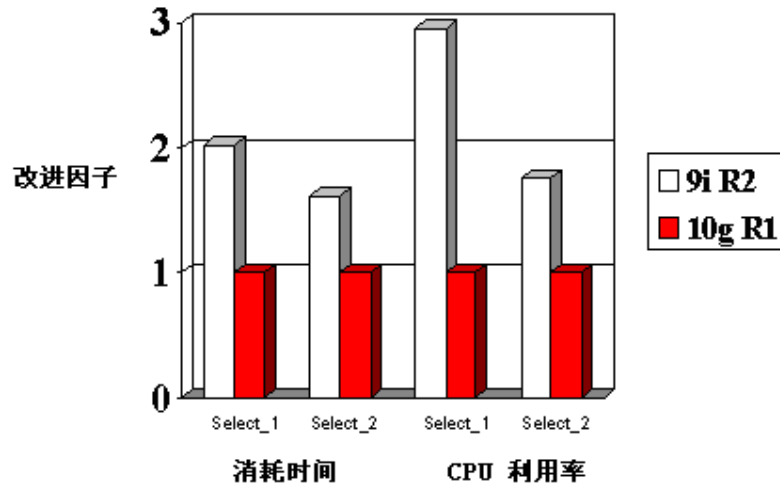


图 4: 全表扫描

图 4 阐述了对一个压缩表的全表扫描的性能改进²。

索引编排表的列表分区选项

由于索引信息和数据存储在一起，索引编排表可以为包含主关键字的精确匹配和/或范围搜索的查询提供表数据的快速访问。由于关键字列并不会在表和主关键字索引中重复存储，所以使用索引编排表可以降低存储要求。它消除了常规索引中为将索引值和行数据相连而使用的行存储位置的额外存储需要。结果由于检索数据所需的访问时间缩短了，性能得到了提高。

索引编排表支持全表功能，包括分区和并行查询。在 Oracle 数据库 10g 中，对索引编排表可用的分区选项的选择扩展成包含列表分区。

概要管理

实体化视图的扩展支持

实体化视图是可以用来汇总、预先计算、复制和分发数据的模式对象。它适用于很多计算环境，例如数据仓库、决策支持，以及分布式或移动计算。例如在数据仓库应用程序中，用户经常进行按常规维度（例如月份、产品或地区）汇总详细数据的查询。实体化视图为存储这些多维和概要计算提

² 关于例子的更多详细信息请参阅附录 1

供机制。查询优化程序使用实体化视图可以显著提高查询性能（请参阅下一节关于查询重写部分）。

实体化视图在其主表中的数据更改时必须刷新。完全刷新将重新执行实体化视图的查询，因而从主表完全重新计算实体化视图的内容。由于完全刷新可能要求时间非常长，所以很多数据仓库环境要求快速、增量的刷新，以满足其操作目标。

快速刷新使用各种更新实体化视图的增量算法来将主表中新的和更新的数据列入考虑。Oracle 数据库提供常规的快速刷新机制，在对主表进行常规 DML 操作（例如 UPDATE 或直接载入操作）时使用；以及分区敏感快速刷新机制，它遵循基表分区的维护操作或 DML 更改。例如，如果基表的分区被截断或删除，实体化视图中受影响的行将被识别并删除。

Oracle 数据库 10g 将快速刷新的支持机制扩展到了更多种实体化视图中。在这个版本中，分区敏感快速刷新将扩展到列表分区或使用 ROWID 作为分区标识的基表的实体化视图中。

Oracle 数据库 10g 也通过使用功能相关性和查询重写扩展快速刷新。当用户根据维度层次定义实体化视图时，Oracle 数据库发现与基表中受影响的分区相关的实体化视图的受影响分区，并根据其他实体化视图或基表生成有效的刷新表达式。

增强型查询重写

查询重写是一项查询优化技术，能够转换根据表和视图来编写的用户查询，从而从实体化视图中攫取数据来提高执行速度。

当查询要求详细记录的概要时，查询优化程序自动识别何时可以并应该使用一个现有实体化视图满足该要求。优化程序直接重写查询，以便使用实体化视图，而不是基础表。这使查询性能显著提高，因为实体化视图在执行和在数据库中存储结果之前在数据库上有预先计算的连接和聚合操作。将查询重写为使用实体化视图可以避免每次使用查询时都进行详细记录的汇总操作。

Oracle 具有一系列强有力的实体化视图重写技术，允许每个实体化视图能被用在尽可能多的查询类型中。在 Oracle 数据库 10g 中，查询重写可以使用多个实体化视图。所以，有更多的查询符合查询重写条件，并且改善查询反应时间。

大量数据的管理

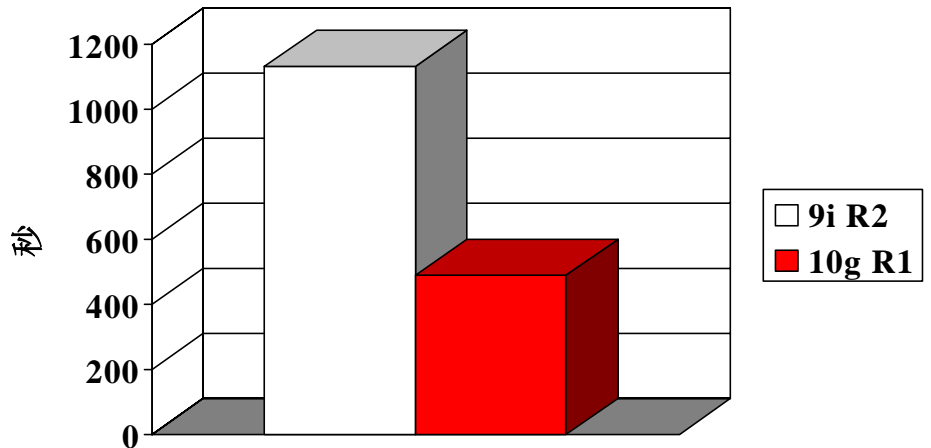
在 Oracle 的客户中数据库安装管理大量数据是非常普遍的。在 Oracle 数据库的数据库大小不是问题：Oracle 数据库 10g 可以支持极大的数据库，达到 8 exabytes（8 百万兆兆字节）数据。

Oracle 数据库包括强大的机制，有助于创建、部署、管理并使用这些大量数据，同时显著改善所有类型的数据库操作的性能。Oracle 数据库通过支持所有类型操作的并行执行来充分利用并行处理，并提供了分区方法和选项的最广泛的选择，它设计为处理各种应用程序方案。Oracle 数据库 10g 通过提供几项增强来进一步扩展这些机制。

大量分区

分区常用用量的日益增长以及数据仓库大小的不断增长使具有上万个分区的数据库结构的创建成为可能。在 Oracle 数据库 10g 中，我们显著改善了分区对象的可伸缩性和内存用量，以保证这些数目对这些对象上的操作的性能影响有限。

作为一个例子，图 5 显示了进行 DROP TABLE 操作的性能改进（该操作所基于的表有 21,504 个分区）。该测试表示 Oracle 数据库 10g 消耗时间比 Oracle9i 缩减了大约 56%³。



TPC-H 模式的 *Lineitem* 表，在 *L_shipdate* (84 分区) 进行范围分区，在 *L_partkey* (每个分区有 256 子分区) 进行散列分区 -> $84 \times 256 = 21504$ 分区

图 5: 具有大量分区的 DROP TABLE

³ 关于例子的更多详细信息请参阅附录 1

散列分区全局索引全局索引通常用在在线事务处理 (OLTP) 环境, 在该环境中, 能够使用不同的标准有效地访问任何单个记录是基本要求之一。由于这个原因, 大多数 OLTP 系统在大型表上都有很多索引。Oracle 自己的应用程序电子商务套件在其很多大型表中都有十几个甚至更多的索引。全局分区索引更灵活, 因为分区的程度和分区键是独立于表的分区方法的。下图阐述了如何根据 Customer_ID 键对“客户”表进行分区以及如何根据 Customer_Name 键创建全局索引和进行分区。

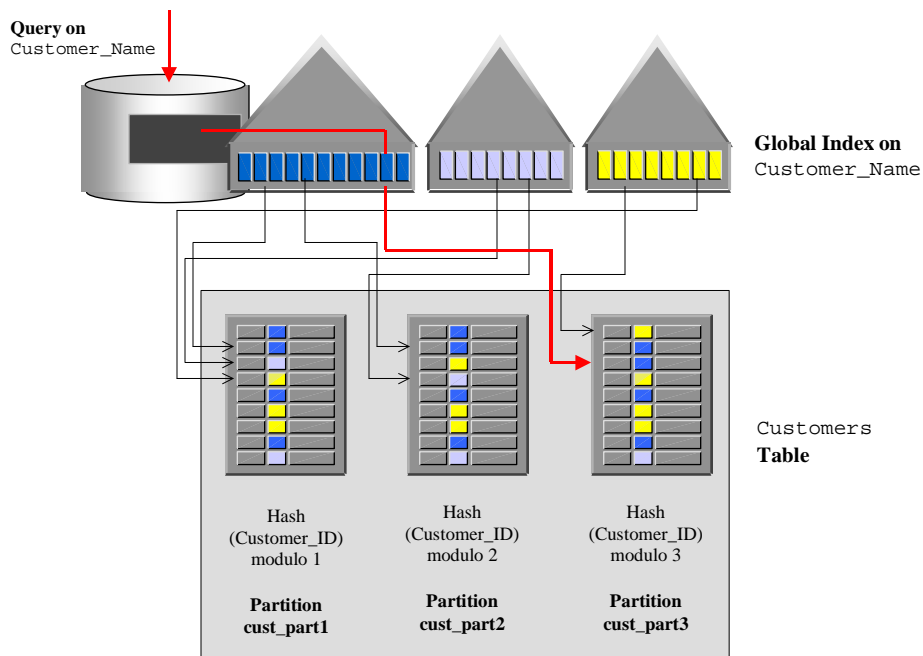


图 6: 全局分区索引的例子

在 Oracle 数据库 10g 中, 用户可以在表、分区表, 以及索引编排表上散列分区索引。这为具有大量并行插入的应用程序改善了吞吐量。

并行单独游标

在 Oracle 数据库 10g 中, 查询并行执行模型从从属 SQL 模型移到并行独立游标模型。一个独立游标包含所有并行执行需要的信息, 并将用在全部并行执行处理中。

因为过去顺序进行的操作变为并行执行, 以及共享内存消耗的减少, 并行独立游标体系结构得到了立竿见影的性能提高。

高速数据移动

Oracle 数据库 10g 为数据的提取、加载以及转换提供了新的功能, 以促进建立和刷新大型数据仓库或多个数据中心的效率。

对于数据的批量移动，Oracle 数据库 10g 提供了可传输的表的跨平台支持，允许大量数据可以从一个平台快速分离，然后与另一个平台上的数据库重新连接。

Oracle 数据库 10g 还引入了新的数据泵工具。Oracle 数据泵是一个高速、并行的基础架构，它实现了数据和元数据从一个数据库到另一个数据库的快速转移。这项技术是 Oracle 新的数据移动工具，数据泵导出和数据泵导入的基础。

数据泵导出和数据泵导入的设计大大加强了原始导出和导入的性能。以下图表分别比较了使用原始导出和导入工具新的数据泵工具的单流数据移动的损耗时间：

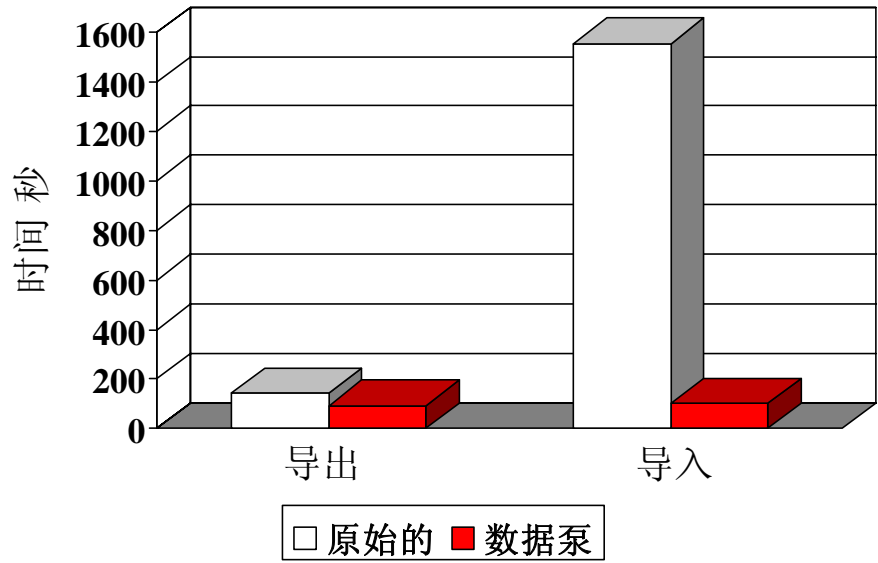


图 7: 性能提高的数据泵例子

1.0 GB 数据 (9.3 M 行)从大型门户公司, 单流。硬件: 1-CPU Ultra 60, 1 GB 内存, 2 磁盘驱动器

而且，通过使用 PARALLEL 参数，可以指定代表导入或导出作业的活动执行服务器操作的最大线程数，从而获得卸载和加载数据的更好的性能。

新的数据泵导出和导入工具在对象选择方面提供更大的灵活性：基于对象和对象类型，有对细粒度对象选择的支持。新的工具还支持“网络”模式，

允许在源和目标数据库之间发生文件较少的、重叠的导出/导入操作。PL/SQL 程序包使用户可以利用公开成文的接口写自己的数据移动工具。

这个版本中导出/导入环境的管理也加强了。不论作业是由用户自愿停止还是发生了无法预见的情况而造成作业中止，数据泵操作都可以完全重新启动。用户可以决定给定作业要消耗多少磁盘空间，而且可以估计完成所需的时间。管理员可以从多个位置通过分离和重新连接到长时间运行的作业来监控作业，而且可以修改作业的特定属性，例如并行性和转储文件可用性。

基于 MICROSOFT WINDOWS 的系统

64 位的 Windows 的出现消除了 32 位平台上存在的一些限制，例如文件大小和内存寻址的限制。Oracle 数据库 10g 在 64 位 Windows 上可以作为本地 64 位应用程序应用，充分利用了基于 Intel Itanium 2 处理器的硬件体系结构的高性能。

性能方面新特性的总结

下表总结了 Oracle 数据库 10g 引入的主要性能特性。

领域	特性
分区	全局散列分区索引 索引编排表支持列表分区
基于 Windows 的系统	支持 64 位 Windows
实体化视图	改进了的分区敏感刷新 查询重写可以使用多个实体化视图
OLAP	并行 SQL 导入 并行聚合
查询优化程序	自动统计数据收集
PL/SQL	新的代码生成器和全局优化程序
自我调整内存	自我调整 SGA 自我调整检查点
ETL	数据泵导出和导入工具 跨平台可传输表空间

附录 1:例子的详细信息

查询优化程序：“CPU+IO”模型

销售表:

```
SQL> desc sales
Name                               Null?Type
-----
COMPANY_ID                          NOT NULL NUMBER(1)
PROD_ID                              NOT NULL NUMBER   CUST ID
NOT NULL NUMBER
TIME_ID                              NOT NULL DATE
CHANNEL_ID                           NOT NULL NUMBER
PROMO_ID                             NOT NULL NUMBER
QUANTITY_SOLD                       NOT NULL NUMBER(10,2)
AMOUNT_SOLD                          NOT NULL NUMBER(10,2)
```

产品表:

```
SQL> desc products;
Name                               Null?Type
-----
PROD_ID                             NOT NULL NUMBER(6)
PROD_NAME                           NOT NULL VARCHAR2(50)
PROD_DESC                            NOT NULL VARCHAR2(4000)
PROD_SUBCATEGORY                    NOT NULL VARCHAR2(50)
PROD_SUBCATEGORY_ID                 NOT NULL NUMBER
PROD_SUBCATEGORY_DESC               NOT NULL VARCHAR2(2000)
PROD_CATEGORY                       NOT NULL VARCHAR2(50)
PROD_CATEGORY_ID                    NOT NULL NUMBER
PROD_CATEGORY_DESC                  NOT NULL VARCHAR2(2000)
PROD_WEIGHT_CLASS                   NOT NULL NUMBER(2)
PROD_UNIT_OF_MEASURE                 VARCHAR2(20)
PROD_PACK_SIZE                       NOT NULL VARCHAR2(30)
SUPPLIER_ID                         NOT NULL NUMBER(6)
PROD_STATUS                          NOT NULL VARCHAR2(20)
PROD_LIST_PRICE                     NOT NULL NUMBER(8,2)
PROD_MIN_PRICE                       NOT NULL NUMBER(8,2)
PROD_TOTAL                           NOT NULL VARCHAR2(13)
PROD_TOTAL_ID                       NOT NULL NUMBER
COMPANY_ID                           NOT NULL NUMBER(1)
```

唯一的索引 `products_pk` 建立在产品表上, `key = PROD_ID`。销售表有 149,960,000 行, 产品表有 10,000 行。

例 1

用于检察限制是否强制执行的查询是:

```
select * from sales where prod_id not in ( select prod_id
from products);
```

例 2

用于测试谓词重排的查询是:

```
select count(*) from sales where to_number(to_char(time_id, 'YYYY')) > 1998 and promo_id = 98;
```

在“CPU+IO”成本模型中，谓词“promo_id = 98”首先执行，从而显著缩短执行时间。

全表扫描

1998 年到 2002 年的销售数据加载到 SALES 表中，且启用压缩。Oracle9i 数据库和 Oracle 数据库 10g 一样，压缩过的 SALES 表大小为 2709.75 MB，一共 112,378,000 行，。SALES 未建立任何索引。

用来测试的 2 个查询是:

```
select_1:
select * from sales where company_id !=2;
```

```
select_2:
select * from sales where amount_sold >14965 and company_id =2;
```

查询 select_1 没有返回行，而 select_2 返回 16 行。两个查询都使用并行度 4 运行。

以下表格阐述了 Oracle9i 和 Oracle 数据库 10g 的性能比较（消耗时间和 CPU 使用量）。

消耗时间:

查询	Oracle9i 数据库	Oracle 数据库 10g	改进
select_1	00:00:39.20	00:00:19.33	51.02%
select_2	00:00:37.49	00:00:23.20	38.12%

CPU 使用量*:

查询	Oracle9i 数据库	Oracle 数据库 10g	改进
select_1	3.61X39.2=141.51	2.48X19.33=47.94	66.12%
select_2	3.84X37.49=143.96	3.55X23.20=82.36	42.78%

*: CPU 使用量 x 消耗时间的平均值

大量分区

SQL 用于创建分区表（TPC-H 模式的 Lineitem 表）：

```
create table l256(
l_shipdate          date ,
l_orderkey          number ,
l_discount          number ,
l_extendedprice    number ,
l_suppkey          number ,
l_quantity         number ,
l_returnflag       char(1) ,
l_partkey          number ,
l_linestatus       char(1) ,
l_tax              number ,
l_commitdate       date ,
l_receiptdate      date ,
l_shipmode         char(10) ,
l_linenum         number ,
l_shipinstruct     char(25) ,
l_comment          varchar(44)
)
pctfree 1
pctused 99
initrans 10
storage (freelists 99)
, parallel
nologging
tablespace ts_1
partition by range (l_shipdate)
subpartition by hash(l_partkey)
subpartitions 256
(
partition item1 values less than
(to_date('1992-01-01','YYYY-MM-DD'))
,
partition item2 values less than
(to_date('1992-02-01','YYYY-MM-DD'))
.....
partition item84 values less than (MAXVALUE)
);
```



Oracle 数据库 10g 性能概述

2004 年 7 月

作者: Vineet Buch、Hervé Lejeune

协作者:

Oracle Corporation

全球总部

500 Oracle Parkway

Redwood Shores, CA 94065

U. S. A.

全球咨询热线:

电话: +1.650.506.7000

传真: +1.650.506.7200

www.oracle.com

版权所有 © 2003, Oracle。保留所有权利。

本文档只用于提供信息,

其中的内容如有更改恕不通知。

不保证本文档中没有错误,也不提供任何其它保证或条件(无论是口头

表达还是法律暗示),包括商用的隐含保证和条件或者对特殊目的的适用性。

我们明确拒绝与本文档有关的任何责任,并且本文档不构成任何直接或间接的
契约义务。未经我们事先的书面许可,不得以任何形式或方法(电子或机械方法)
为任何目的复制或传输本文档。

Oracle 是 Oracle Corporation 和/或其会员的注册商标。其他名称可能是其
各自所有者的商标。