Oracle 数据库 11g 中的分区

Oracle 白皮书
2007 年 6 月

注意：

以下内容旨在概述产品的总体发展方向。此信息仅供参考，不可纳入任何合同。其中的信息不承诺提供任何资料、代码或功能，并且不能作为购买的依据。所描述的有关 Oracle 产品的任何特性或功能的开发、发行和时间安排均由 Oracle 自行决定。

分区 — 概念

简介

Oracle 分区在 1997 年的 Oracle 8.0 中首次引入，它是 Oracle 数据库最重要、最成功的功能之一，可以提高数以万计的应用程序的性能、可管理性和可用性。Oracle 数据库 11g 引入了第 8 代分区，继续提供突破性的新增强功能；新的分区技术使客户能够针对更多的业务案例进行建模，此外，全新的分区建议和自动化框架使所有人都能使用 Oracle 分区。自从首次引入分区以来，Oracle 数据库 11g 被视为分区的最重大新版本，它可以在今后十年内继续保护客户在分区上的投资。

分区的优势

分区可以通过提高可管理性、性能和可用性，为各种应用程序带来极大的好处。通常，分区可以使某些查询以及维护操作的性能大大提高。此外，分区可以大大降低数据的总拥有成本，它使用“分层存档”方法在低成本存储设备上使较旧的相关信息保持联机状态。如果要在大型环境中进行信息生命周期管理，Oracle 分区可以提供高效、简单但十分强大的方法。

通过分区，数据库设计人员和管理员能够解决前沿应用程序带来的一些难题。分区是构建千兆字节数据系统或超高可用性系统的关键工具。

分区的基本知识

分区能够将表、索引或按索引组织的表进一步细分为小块。这些数据库对象的小块称为分区。每个分区都有自己的名称，还可以选择自己的存储特性。从数据库管理员的角度来看，分区后的对象具有多个小块，这些小块既可以集中管理，也可以单独管理。这就使管理员在管理分区后的对象时具有相当大的灵活性。

但是，从应用程序的角度来看，分区后的表与未分区的表完全相同；在使用 SQL DML 命令访问分区后的表时，无需任何修改。

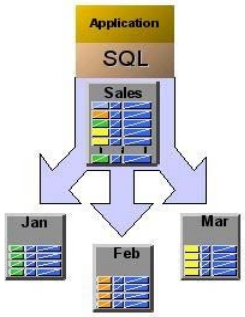

图 1：从应用程序角度和 DBA 角度看待分区后的表

数据库对象（表、索引和按索引组织的表）是使用“分区键”进行分区的，分区键是决定指定行将位于哪个分区的一个列集。例如，图 1 中所示的 sales 表是按销售日期范围分区的（使用按月分区的策略）；表将作为单一的“普通”表显示给所有应用程序。然而，DBA 可以单独管理和存储每个月度分区，可以使用不同的存储层，对较旧的数据应用表压缩，或者在只读表空间中存储旧数据的完整范围。

索引可以与基础表的基本分区策略耦合，也可以不与之耦合，与选定的索引分区策略无关。应该根据业务需求选择相应的索引分区策略，从而实现最合适的分区，以支持任何类型的应用程序。Oracle 数据库 11g 可区分三种类型的分区索引。

- 局部索引：局部索引是针对分区表的索引，该索引可以与基本分区表耦合，并“继承”该表的分区策略。因此，局部索引的每个分区仅对应于基础表的一个分区。耦合可实现优化的分区维护；例如，在删除表分区之后，Oracle 只需删除相应的索引分区。不需要进行代价高昂的索引维护。局部索引在数据仓库环境中是最常见的。
- 全局分区索引：全局分区索引是使用不同于其所在表的分区键或分区策略进行分区的索引，其所在表可以是分区表，也可以是非分区表。全局分区索引可以使用范围分区或散列分区进行分区，还可以解除与基础表的耦合。例如，某个表可以按月份进行范围分区，因此具有十二个分区，而该表上的索引则可以使用不同的分区键进行范围分区，从而具有不同的分区数量。全局分区索引在 OLTP 中比在数据仓库环境中更为常见。
- 全局非分区索引：全局非分区索引实质上与非分区表的索引一样。索引结构未分区并且未与基础表耦合。在数据仓库环境中，全局非分区索引的最常见用法是实现主键约束。另一方面，OLTP 环境在很大程度上依赖于全局非分区索引。

Oracle 还提供了一套全面的 SQL 命令来管理分区表，其中包括添加新分区、删除分区、分解分区、移动分区、合并分区、截断分区以及有选择地压缩分区的命令。

使用分区提高可管理性

通过 Oracle 分区，可将表和索引分成更多、更小的可管理单元，从而使数据库管理员能以“分而治之”的方式管理数据。

使用分区，维护操作可集中在表的特定部分。例如，数据库管理员可以压缩表中包含（例如）2006 年数据的单个分区，而不是压缩整个表。对于整个数据库对象的维护操作，可以在每个分区的基础上进行，从而将维护工作分解成更容易管理的小块。

利用分区提高可管理性的一个典型用法是支持数据仓库中的“滚动视窗”加载进程。假设数据库管理员每周都要向表中加载新数据，则可以对该表进行范围分区，使每个分区包含一周的数据。这样，加载进程就只需添加新的分区。添加一个分区的操作比修改整个表的效率高很多，因为数据库管理员不需要修改任何其他分区。

使用分区的另一个优势是，在移除数据时可以删除整个分区，与单独删除每一行相比，这个操作非常高效、快速。

使用分区提高性能

通过限制要检查或要操作的数据量，分区提供了许多性能优势。这些特性包括：

- 分区修整：分区修整（又称为分区清除）是使用分区提高性能的最简单、最有价值的手段。分区修整通常能够将查询性能提高几个数量级。例如，假设某个应用程序包含一个存储订单历史记录的 ORDERS 表，并且该表已按周进行了分区。查询一周的订单只需访问 ORDERS 表的一个分区。如果该表包含两年的历史记录，则这个查询只需要访问一个分区而不是 104 个分区。仅仅由于分区修整的作用，该查询的执行速度有可能快 100 倍。分区修整能与所有其他 Oracle 性能特性协作。Oracle 公司将把分区修整技术与索引技术、连结技术或并行访问方法结合使用。

- 分区智能联接：分区可以通过称为分区智能联接的技术提高多表联接的性能。如果两个表要联接在一起，并且至少有一个表要用联接键来分区，则可以使用分区智能联接。分区智能联接可将联接表的大型联接分解成“相同”数据集的较小联接。这里，“相同”的定义是在联接两端包含完全相同的分区键值集，从而确保只有这些“相同”数据集的联接才会产生结果，并且无需考虑其他数据集。Oracle 可以使用已经进行（物理）同类分区的表进行联接，或者在运行时透明地重新分配（相当于“重新分区”）一个表来创建与其他表的分区相匹配的分区类型相同的数据集，从而在较短的时间内完成整个联接。这就为串行和并行执行带来了显著的性能改善。

使用分区提高可用性

分区的数据库对象具有分区独立性。该分区独立性特点是高可用性策略的一个重要部分。例如，如果分区表的一个分区不可用，但该表的所有其他分区仍然保持联机并可用。那么，应用程序可以继续对该分区表执行查询和事务处理，只要不访问那个不可用的分区，这些数据库操作都能够成功运行。

数据库管理员可以指定各分区存放在不同的表空间里，从而允许管理员在每个单独分区上执行备份和恢复操作（独立于表的其他分区）。因此，在发生灾难的情况下，可以只为数据库恢复包含活动数据的分区，之后在方便的时间内再恢复其他分区中的非活动数据。这样，就减少了系统停机时间。

此外，分区还可以减少计划停机时间。由于分区改善了性能，这使数据库管理员能够在相对较小的批处理窗口中完成大型数据库对象的维护操作。

分区 — 为业务建模

Oracle 数据库 11g 提供了一组最全面的分区策略，允许客户根据实际业务需要以最佳方式调整数据细分。所提供的所有分区策略都依赖于可用于单一（一级）或组合分区表的基本数据分配方法。此外，Oracle 还提供了多种分区扩展，可以提高分区键选择的灵活性，根据需要提供自动化分区创建，以及为非分区对象建议分区策略。

基本分区策略

Oracle 分区提供了三种基本数据分配方法，用于控制如何将数据实际放入各种单独的分区中，如：

- 范围：数据基于分区键值的范围分配（如果使用日期列作为分区键，则“January-2007”分区将包含分区键值在“01-JAN-2007”到“31-JAN-2007”之间的行）。数据分配是一个没有漏洞的连续统一体，范围的较低边界由前一范围的较高边界自动定义。

- 列表：数据分配由分区键的值列表定义（如果使用地区列作为分区键，则“North America”分区可能包含值“Canada”、“USA”和“Mexico”）。可以定义一个特殊的“DEFAULT”分区，用于捕获未由任何列表显式定义的所有分区键值。

- 散列：将散列算法用于分区键以确定给定行的分区。与其他两个数据分配方法不同，散列不提供数据与分区之间的任何逻辑映射。

使用上述数据分配方法，可以将表分成单一表或组合分区表：

- 单一（一级）分区：通过指定其中一个数据分配方法来定义表，并使用一个或多个列作为分区键。例如，某个表使用数字列作为分区键，并具有两个分区“less_than_five_hundred”和 less_than_thousand”，“less_than_thousand”分区包含符合以下条件的行：500 <= 分区键 <1000。

您可以指定范围、列表和散列分区表。

- 组合分区：使用两个数据分配方法的组合来定义组合分区表。首先，使用一个数据分配方法将表分区，然后使用第二个数据分配方法为每个分区进一步划分子分区。给定分区的所有子分区一起代表数据的逻辑子集。例如，首先对范围-散列组合分区表进行范围分区，然后使用散列分区技术为每个单独的范围分区进一步划分子分区。

  所提供的组合分区技术包括范围-散列、范围-列表、范围-范围、列表-范围、列表-列表和列表-散列。

- 可以使用范围、散列和列表分区对按索引组织的表 (IOT) 进行分区。组合分区不支持 IOT。

分区扩展

除了基本分区策略以外，Oracle 还提供了分区扩展。Oracle 数据库 11g 中的扩展主要关注两个目标：

(a) 显著增强分区表的可管理性。

(b) 扩展分区键定义的灵活性。

扩展包括：

间隔分区：间隔分区是 Oracle 数据库 11g 中的一个新的分区策略，它扩展了范围方法的功能，可以使用间隔定义来定义同类分区范围。当分区的数据首次插入时，Oracle 将根据需要自动创建分区，而非显式指定单独的范围。间隔分区可以大大提高分区表的可管理性。例如，可以定义一个间隔分区表，以便 Oracle 为日历年的每个月份创建一个新分区；随后，当“September 2007”的第一条记录插入数据库时，就会为该月份自动创建一个分区。

可用于间隔分区表的技术包括间隔、间隔-列表、间隔-散列和间隔-范围。

引用分区：Oracle 数据库 11g 允许利用现有的父子关系对表进行分区。子表可以继承父表的分区策略，而无需在子表中存储父表的分区键列。在没有引用分区的情况下，如果您想要利用相同的分区策略，就必须将父表的所有分区键列复制到子表；此外，引用分区还允许您根据逻辑数据模型自然地对表进行分区，而无需存储分区键列，从而减少了非规范化的手动开销并节省了空间。引用分区还可以透明地继承所有分区维护操作，从而将表的逻辑形式从父表更改为子表。此外，引用分区还可以为父表和子表的同类分区自动进行分区智能联接，从而提高该操作的性能。

例如，父表 ORDERS 根据 ORDER_DATE 列进行了范围分区；其子表 ORDER ITEMS 没有包含 ORDER_DATE 列，但可以根据对 ORDERS 表的引用进行分区。如果 ORDERS 表按月份分区，则“Jan-2007”订单的所有订单项将存储在 ORDER ITEMS 表的单个分区中，该表与父表 ORDERS 具有相同类型的分区。如果将“Feb-2007”分区添加到 ORDERS 表，Oracle 将透明地将同类分区添加到 ORDER ITEMS 表。所有基本分区策略都可用于引用分区。

基于虚拟列的分区：在以前的 Oracle 版本中，只有当分区键以物理方式存在于表中，才可以对表进行分区。虚拟列是 Oracle 数据库 11g 中的一个新功能，该功能移除了这个限制，并允许使用表达式定义分区键，以便使用表的一个或多个现有列，并将表达式仅作为元数据存储。

分区已经得到增强，以允许在虚拟列上定义分区策略，从而实现更全面地满足业务需求。您通常会看到信息过载的列；例如，一个 10 位的帐户 ID 可以在前三位中包含帐户分支信息。利用基于虚拟列的分区扩展，可以通过虚拟（派生）列 ACCOUNT_BRANCH 来扩展包含 ACCOUNT_ID 列的 ACCOUNTS 表，ACCOUNT_BRANCH 列派生自 ACCOUNT_ID 列（该表的分区键）的前三位。

基于虚拟列的分区支持所有基本分区策略。

Partition advisor

Oracle 数据库 11g 中的 SQL Access Advisor 已经得到增强，除了已经为索引、物化视图和物化视图日志提供的建议之外，还可以生成分区建议。SQL Access Advisor 生成的建议（仅针对分区或针对历史）将显示在其实现后将获得的预期性能提高。生成的脚本可以手动实现，也可以提交到 Oracle 企业管理器中的队列。

利用分区建议扩展，客户不仅能够获得专门针对分区的建议，还可以获得更全面的整体 SQL Access Advisor 建议，从而全面提高 SQL 语句的整体性能。

Partition Advisor 已集成到 SQL Access Advisor 中，它是 Oracle Tuning Pack（一个额外许可选件）的一部分。它可以从企业管理器中使用，或者通过命令行接口使用。

分区策略和扩展概述

下表列出了 Oracle 数据库 11g 中提供的所有基本分区策略的概念性概述：

分区策略	数据分配	示例业务案例
范围分区	基于值的连续范围。	Orders 表根据 order_date 进行范围分区
列表分区	基于值的无序列表。	Orders 表根据 country 进行列表分区
散列分区	基于散列算法。	Orders 表根据 customer_id 进行散列分区
组合分区 范围-范围 范围-列表 范围-散列 列表-列表 列表-范围 列表-散列	基于上面提到的两个基本技术（范围、列表、散列和间隔分区）的组合	Orders 表先根据 order_date 进行范围分区，然后再根据 customer_id 进行散列分区 Orders 表先根据 order_date 进行范围分区，然后再根据 shipment_date 进行范围分区

除了所提供的分区策略以外，Oracle 数据库 11g 还提供了以下分区扩展：

分区扩展	分区键	示例业务案例
间隔分区 间隔 间隔-范围 间隔-列表 间隔-散列	对范围分区的扩展。由间隔时间定义，提供等宽范围。除了第一个分区，在匹配数据插入时，其他所有分区都将按需自动创建。	Orders 表根据 order_date 进行分区，order_date 是预先定义的每日间隔，自“01-Jan-2007”开始。
引用分区	子表的分区通过主键-外键关系从父表继承。分区键不存储在子表的实际列中。	(父) Orders 表根据 order_date 进行范围分区，并将分区技术遗传给 (子) order lines 表。order_date 列仅在 orders 父表中出现。
基于虚拟列的分区	由上面提到的其中一个分区技术定义，并且分区键基于虚拟列。虚拟列不存储在磁盘上，仅以元数据的形式存在。	Orders 表具有一个虚拟列，可基于客户帐号的前三个数字得出销售区域。然后，根据销售区域对 orders 表进行列表分区。

利用分区进行信息生命周期管理

利用 Oracle 分区，可以最好地解决当今面临的以尽可能低的成本存储大量数据的挑战。单个分区的独立性是解决“分层存档”策略联机部分的关键点。特别是在包含历史数据的表中，数据的重要性（以及访问模式）在很大程度上依赖于数据的年龄；分区使您能够将单个分区（或分区组）存储在不同的存储层，从而提供不同的物理属性和价值点。例如，一个包含 2 年数据的 Orders 表可以仅将最近一个季度的数据存储在昂贵的高端存储层上，并将表的其余数据（几乎 90% 的数据）存放在廉价的低成本存储层上。通过 Oracle 分区，可以大大降低存储成本（通常可以节省 50% 或更多的成本），并且不会影响最终用户访问，从而最佳化存储信息的拥有成本。

Oracle ILM Assistant 是一个可从 OTN 上下载的免费工具，它可以说明这些成本节省，演示如何对表进行分区，并建议何时将分区移到其他存储层。

总结

考虑到 Oracle 分区的新功能和改进的功能，我们认为 Oracle 数据库 11g 是自 1997 年引入 Oracle 分区以来最重要的版本。在每一个重要版本中，Oracle 都增强了分区的功能（通过添加新的分区技术、增强可伸缩性，或者扩展可管理性和维护功能）。Oracle 公司计划继续增加新的分区技术，确保提供一个面向所有业务需求的最佳分区技术。

分区将面向所有人。Oracle 分区可以大大增强几乎所有数据库应用程序的可管理性、性能和可用性。分区可用于前沿应用程序，并且是确保这些应用程序成功的关键技术因素。此外，分区还可用于较为普通的数据库应用程序，以便简化这些应用程序的管理工作，并降低管理成本。

由于分区对应用程序是透明的，因此可以轻松实现，因为这不需要进行高成本且耗时的应用程序更改。

Oracle 数据库 11g 中的分区
2007 年 6 月
作者：Hermann Baer
合作者：

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
oracle.com