Note: This document is for informational purposes. It is not a commitment to deliver any material, code, or functionality, and should not be relied upon in making purchasing decisions. The development, release, and timing of any features or functionality described in this document remains at the sole discretion of Oracle.

ORACLE PARTITIONING

ORACLE PARTITIONING

- Proven functionality in its 8th generation
- Broadest and most comprehensive offering on the market
- Partitioning improves the Performance, Availability, and Manageability for a wide variety of applications
- Key enabler for Information Lifecycle Management inside the database: online "tiered archiving' dramatically reduces the TCO
- Implementation without application changes

Oracle Partitioning, an option of Oracle Database 11g Enterprise Edition, enhances the manageability, performance, and availability of a wide variety of applications. Partitioning allows tables, indexes, and index-organized tables to be subdivided into smaller pieces, enabling these database objects to be managed and accessed at a finer level of granularity. Oracle provides a comprehensive range of partitioning schemes to address every business requirement. Moreover, since it is entirely transparent in SQL statements, partitioning can be applied to any application, from OLTP to Data Warehousing.

Benefits of Oracle Partitioning

Partitioning can provide tremendous benefits to a wide variety of applications by improving manageability, performance, and availability. In its 8th generation, Oracle Partitioning is proven key functionality for building multi-terabyte systems or systems with extremely high availability requirements.

For example, it is not unusual for partitioning to improve the performance of certain queries or maintenance operations by an order of magnitude. Moreover, partitioning can greatly reduce the total cost of ownership, using a "tiered archiving" approach of keeping older relevant information still online on low cost storage devices. Thus Oracle Partitioning enables an efficient and simple, yet very powerful approach to the Information Lifecycle Management for large environments.

Basics of Oracle Partitioning

Partitioning allows a table, index or index-organized table to be subdivided into smaller pieces. Each of these smaller pieces of a database object is called a partition (or subpartition for composite partitioned objects). Each partition has its own name, and may optionally have its own storage characteristics, such as having table compression enabled or having partitions being stored in different tablespaces, potentially on different ASM diskgroups. From the perspective of a database administrator, a partitioned object has multiple pieces which can be managed either collectively or individually. This gives the administrator considerably flexibility in managing partitioned objects. However, from the perspective of the application, a partitioned table is identical to a non-partitioned table; no modifications are necessary

when accessing a partitioned table using SQL DML commands.

Tables are partitioned using a 'partitioning key', a set of columns which determine in which partition a given row will reside. Oracle Database 11g provides the most comprehensive Partitioning offering in the market, with a broad variety of partitioning techniques and advanced mechanisms,. A Partition Advisor is also available which will recommend how to partition a table based on how the data will be accessed.

The following table gives an overview of all available basic partitioning strategies in Oracle Database 11g.

Partitioning Strategy	Data Distribution	Sample Business Case
Range Partitioning	Based on consecutive ranges of values.	• Orders table range partitioned by order_date
List Partitioning	Based on unordered lists of values.	• Orders table list partitioned by country
Hash Partitioning	Based on a hash algorithm.	• Orders table hash partitioned by customer_id
Composite Partitioning • Range-Range • Range-List • Range-Hash • List-List • List-Range • List-Hash • Interval-Range * • Interval-List * • Interval-Hash *	Based on a combination of two of the above-mentioned basic techniques of Range, List, Hash, and Interval Partitioning	• Orders table is range partitioned by order_date and sub-partitioned by hash on customer_id • Orders table is range partitioned by order_date and sub-partitioned by range on shipment_date

In addition to the available partitioning strategies, Oracle Database 11g provides the following partitioning extensions::

Partitioning Strategy	Data Distribution	Sample Business Case
Interval Partitioning	Defined by an interval, providing equi-width ranges. With the exception of the first partition all partitions are automatically created as-needed when matching data arrives. An extension to Range Partitioning.	• Orders table partitioned by order_date with a predefined daily interval, starting with '01-Jan-2007'
REF Partitioning	Partitioning for a child table is inherited from the parent table through a primary key – foreign key relationship. The partitioning keys are not stored in actual columns in the child table.	• (Parent) Orders table range partitioned by order_date and inherits the partitioning technique to (child) order lines table. Column order_date is only present in the parent orders table

Virtual column based Partitioning	Partitioning is defined by one of the above-mentioned partition techniques and the partitioning key is based on a virtual column. Virtual columns are not stored on disk and only exist as metadata.	• Orders table has a virtual column that derives the sales region based on the customer account number. The orders table is then list partitioned by sales region.

Index-organized tables can be range-partitioned, list-partitioned, or hash-partitioned. Oracle Database 11g also provides three types of partitioned indexes:

Local Indexes: A local index is an index on a partitioned table which is partitioned in the exact same manner as the underlying partitioned table. Each partition of a local index corresponds to one and only one partition of the underlying table.

Global Partitioned Indexes: A global partitioned index is an index on a partitioned or non-partitioned table which is partitioned using a different partitioning-key from the table. Global-partitioned indexes can only be partitioned using range partitioning.

Global Non-Partitioned Indexes: A global non-partitioned index is essentially identical to an index on a non-partitioned table. The index structure is not partitioned.

Oracle provides a robust set of techniques for partitioning tables, indexes, and index-organized tables, so that partitioning can be optimally applied to any application in any business environment.

In addition, Oracle provides a comprehensive set of SQL commands for managing partitioning tables, which includes commands for adding new partitions, dropping, truncating, splitting, merging. Moving, and compressing partitions.

Oracle Partitioning for Manageability

The Oracle Partitioning option allows tables and indexes to be partitioned into smaller, more manageable units, providing database administrators with the ability to pursue a "divide and conquer" approach to data management.

With partitioning, maintenance operations can be focused on particular portions of tables. For example, a database administrator could back up a single partition of a table, rather than backing up the entire table. For maintenance operations across an entire database object, it is possible to perform these operations on a per-partition basis, thus dividing the maintenance process into more manageable chunks.

A typical usage of partitioning for manageability is to support a 'rolling window' load process in a data warehouse. Suppose that a DBA loads new data into a table on weekly basis. That table could be range-partitioned so that each partition contains one week of data. The load process is then simply the addition of a new partition. Adding a single partition is much more efficient than modfying the entire table, since the DBA does not need to modify any other partitions. The same is true for purging data from a partitioned table. You simply drop a partition, a very cheap and quick data dictionary operation, rather than issuing a DELETE command, using lots of resources and touching all the data to be deleted.

Oracle Partitioning for Performance

When data volumes increase, a common concern, is that system performance will degrade because of all the extra data that has to be examined. Oracle Partitioning eliminates this problem, by limiting the amount of data to be examined or operated on, thus significantly improving performance beyond what is possible with a non-partitioned table. Oracle Partitioning option provides a number of performance benefits, including the following:

Partitioning Pruning: Partitioning pruning is the simplest and also the most substantial means to improve performance using partitioning. For example, suppose an application contains a Shipment table containing a historical record of shipments, and that this table has been partitioned by day. A query requesting shipments for a single day would only access a single partition of the Shipments table. If the Shipments table had 2 years of historical data, this query would access one partition instead of 730 partitions. This query could potentially execute appr. 700x faster simply because of partition-pruning. Partition pruning works with all of Oracle's other performance features. Oracle will utilize partition pruning in conjunction with any indexing technique, join technique, or parallel access method.

Partition-wise Joins: Partitioning can also improve the performance of multi-table joins, by using a technique known as partition-wise join. Partition-wise joins can be applied with two tables are being joined together, and both of these tables are partitioned on the join key. Partition-wise joins break large joins into smaller joins that occur between each of the partitions, completing the overall join in less time. This offers significant performance benefits both for serial and parallel execution.

Oracle Partitioning for Availability

Partitioned database objects provide partition independence. which is an important part of a high-availability strategy. For example, if one partition in a table is unavailable, all of the other partitions of the table remain online and available; the application can continue to execute queries and transactions against this partitioned table, and these database operations will run successfully if they do not need to access the unavailable partition.

Moreover, partitioning can reduce scheduled downtime. The performance gains provided by partitioning may enable database administrators to complete maintenance operations on large database objects in relatively small batch windows.

Information Lifecycle Management with Oracle Partitioning

Today's challenge of storing vast quantities of data for the lowest possible cost can be optimally addressed using Oracle Partitioning. By understanding of how data is accessed, the independence of individual partitions is the key enabler for addressing the online portion of a “tiered archiving” strategy. Specifically in tables containing historical data, partitioning enables individual partitions (or groups of partitions) to be stored on different storage tiers, providing different physical attributes and price points. For example an Orders table containing 2 years worth of data could have only the most recent quarter being stored on an expensive high-end storage tier and keep

SIDEBAR HEAD

RELATED PRODUCTS AND SERVICES:

- Oracle Partitioning is an option to Oracle Database 11g Enterprise Edition

the rest of the table (almost 90% of the data) on an inexpensive low cost storagetier. Through Oracle Partitioning, the storage costs are reduced by factors (cost savings of 50% or more are not uncommon), without impacting the end user access, thus optimizing the cost of ownership for the stored information.

Oracle Partitioning is for Everybody

Oracle Partitioning can greatly enhancethe manageability, performance, and availability of almost any database application. Partitioning can be applied to cutting-edge applications and indeed partitioning can be a crucial technology ingredient to ensure these applications' success. However, partitioning can also be applied to more commonplace database applications in order to simplify the administration and costs of managing such applications.

Considering the new and improved functionality for partitioning, Oracle Database 11g is the most significant release since the introduction of Oracle Partitioning in 1997. In every major release, Oracle has enhanced the functionality of Partitioning, by either adding new partitioning techniques, enhancing the scalability, or extending the manageability and maintenance capabilities. Oracle plans to continue to add new partitioning techniques to ensure that an optimal partitioning technique is available for every business requirement.

ORACLE