VLDB

Partitioning
Compression
Oracle Partitioning in Oracle Database 11g
## Oracle Partitioning
### Ten Years of Development

<table>
<thead>
<tr>
<th>Version</th>
<th>Core functionality</th>
<th>Performance</th>
<th>Manageability</th>
</tr>
</thead>
<tbody>
<tr>
<td>Oracle8</td>
<td>Range partitioning</td>
<td>“Static” partition pruning</td>
<td>Basic maintenance operations: add, drop, exchange</td>
</tr>
<tr>
<td></td>
<td>Global range indexes</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Oracle8i R2</td>
<td>Composite range-list partitioning</td>
<td>Fast partition split</td>
<td></td>
</tr>
<tr>
<td>Oracle9i R2</td>
<td>List partitioning</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Oracle9i R2</td>
<td>Global hash indexes</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Oracle10g</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Oracle10g R2</td>
<td>1M partitions per table</td>
<td>“Multi-dimensional” pruning</td>
<td>Fast drop table</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Core functionality:**
- Range partitioning
- Global range indexes
- Hash and composite range-hash partitioning
- List partitioning
- Composite range-list partitioning
- Global hash indexes
- 1M partitions per table

**Performance:**
- “Static” partition pruning
- Partition-wise joins
- “Dynamic” pruning
- Fast partition split
- “Multi-dimensional” pruning

**Manageability:**
- Basic maintenance operations: add, drop, exchange
- Merge operation
- Global index maintenance
- Local Index maintenance
- Fast drop table
<table>
<thead>
<tr>
<th></th>
<th>Core functionality</th>
<th>Performance</th>
<th>Manageability</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Oracle8</strong></td>
<td>Range partitioning</td>
<td>“Static” partition pruning</td>
<td>Basic maintenance operations: add, drop, exchange</td>
</tr>
<tr>
<td></td>
<td>Global range indexes</td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Oracle8i</strong></td>
<td>Hash and composite range-hash partitioning</td>
<td>Partition-wise joins “Dynamic” pruning</td>
<td>Merge operation</td>
</tr>
<tr>
<td><strong>Oracle9i</strong></td>
<td>List partitioning</td>
<td></td>
<td>Global index maintenance</td>
</tr>
<tr>
<td><strong>Oracle9i R2</strong></td>
<td>Composite range-list partitioning</td>
<td>Fast partition split</td>
<td></td>
</tr>
<tr>
<td><strong>Oracle10g</strong></td>
<td>Global hash indexes</td>
<td></td>
<td>Local Index maintenance</td>
</tr>
<tr>
<td><strong>Oracle10g R2</strong></td>
<td>1M partitions per table</td>
<td>“Multi-dimensional” pruning</td>
<td>Fast drop table</td>
</tr>
<tr>
<td><strong>Oracle Database 11g</strong></td>
<td>More composite choices, REF Partitioning, Virtual Column</td>
<td>Interval Partitioning: Partition Advisor</td>
<td></td>
</tr>
</tbody>
</table>
Oracle Partitioning
Fundamental Concepts for Success

• While performance seems to be the most visible one, don't forget about the rest
  • partitioning must address all business-relevant areas of Performance, Manageability, and Availability

• Partition autonomy is crucial
  • fundamental requirement for any partition maintenance operations
  • acknowledge partitions as metadata in the data dictionary
Oracle Partitioning Enhancements

• Complete the basic partitioning strategies defines HOW data is going to be partitioned
  • new composite partitioning methods

• Introduce partitioning extensions defines WHAT controls the data placement
  • enhance the manageability and automation
  • virtual column based partitioning
  • REF partitioning
  • interval partitioning
  • partition advisor
Composite Partitioning in Oracle Database 11g
Extended Composite Partitioning Strategies

• Concept of composite partitioning
  • Data is partitioned along two dimensions (A,B)
  • A distinct value pair for the two dimensions uniquely determines the target partitioning

• Composite partitioning is complementary to multi-column range partitioning

• Extensions in Oracle Database 11g

<table>
<thead>
<tr>
<th>New 11g</th>
<th>Use Case</th>
</tr>
</thead>
<tbody>
<tr>
<td>List - Range</td>
<td>Geography</td>
</tr>
<tr>
<td>Range - Range</td>
<td>ShipDate</td>
</tr>
<tr>
<td>List - Hash</td>
<td>Time</td>
</tr>
<tr>
<td>List - List</td>
<td>OrderDate</td>
</tr>
<tr>
<td></td>
<td>Geography</td>
</tr>
<tr>
<td></td>
<td>OrderID</td>
</tr>
<tr>
<td></td>
<td>Product</td>
</tr>
</tbody>
</table>
Composite Partitioning - Concept

Table SALES
RANGE(order_date)-RANGE(ship_date)

ship_date

Jan 2006
Feb 2006
May 2006


order_date
Composite Partitioning - Concept

Table SALES
RANGE(order_date)-RANGE(ship_date)

- All records with order_date in March 2006
Composite Partitioning - Concept

Table SALES
RANGE(order_date)-RANGE(ship_date)

ship_date

<table>
<thead>
<tr>
<th>Jan 2006</th>
<th>Feb 2006</th>
<th>May 2006</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

order_date

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

• All records with ship_date in May 2006
Table SALES
RANGE(order_date)-RANGE(ship_date)

- All records with order_date in March 2006 AND ship_date in May 2006
Virtual Column based Partitioning
Virtual Columns

Business Problem
• Extended Schema attributes are fully derived and dependent on existing common data
• Redundant storage or extended view definitions are solving this problem today
  • requires additional maintenance and creates overhead

Solution
• Oracle Database 11g introduces virtual columns
  • purely virtual, meta-data only
• Treated as real columns except no DML
  • can have statistics
  • eligible as partitioning key
• Enhanced performance and manageability
Virtual Columns - Example

- Base table with all attributes ...

```sql
CREATE TABLE accounts
(acc_no number(10) not null,
 acc_name varchar2(50) not null,
...
```

<table>
<thead>
<tr>
<th>acc_no</th>
<th>Name</th>
</tr>
</thead>
<tbody>
<tr>
<td>12500</td>
<td>Adams</td>
</tr>
<tr>
<td>12507</td>
<td>Blake</td>
</tr>
<tr>
<td>12666</td>
<td>King</td>
</tr>
<tr>
<td>12875</td>
<td>Smith</td>
</tr>
</tbody>
</table>
Virtual Columns - Example

• Base table with all attributes ...
  • ... is extended with the virtual (derived) column

```
CREATE TABLE accounts
(acc_no     number(10)   not null,
 acc_name   varchar2(50) not null, ...
 acc_branch number(2)    generated always as
 (to_number(substr(to_char(acc_no),1,2)))
```

<table>
<thead>
<tr>
<th>acc_no</th>
<th>acc_name</th>
<th>acc_branch</th>
</tr>
</thead>
<tbody>
<tr>
<td>12500</td>
<td>Adams</td>
<td>12</td>
</tr>
<tr>
<td>12507</td>
<td>Blake</td>
<td>12</td>
</tr>
<tr>
<td>12666</td>
<td>King</td>
<td>12</td>
</tr>
<tr>
<td>12875</td>
<td>Smith</td>
<td>12</td>
</tr>
</tbody>
</table>
Virtual Columns - Example

- Base table with all attributes ...
  - ... is extended with the virtual (derived) column
  - ... and the virtual column is used as partitioning key

```
CREATE TABLE accounts
(acc_no number(10) not null,
 acc_name varchar2(50) not null, ...
 acc_branch number(2) generated always as
                  (to_number(substr(to_char(acc_no),1,2)))
partition by list (acc_branch) ...
```

<table>
<thead>
<tr>
<th>acc_no</th>
<th>Name</th>
<th>Branch</th>
</tr>
</thead>
<tbody>
<tr>
<td>12500</td>
<td>Adams</td>
<td>12</td>
</tr>
<tr>
<td>12507</td>
<td>Blake</td>
<td>12</td>
</tr>
<tr>
<td>12666</td>
<td>King</td>
<td>12</td>
</tr>
<tr>
<td>12875</td>
<td>Smith</td>
<td>12</td>
</tr>
<tr>
<td>32320</td>
<td>Jones</td>
<td>32</td>
</tr>
<tr>
<td>32407</td>
<td>Clark</td>
<td>32</td>
</tr>
<tr>
<td>32758</td>
<td>Huru</td>
<td>32</td>
</tr>
<tr>
<td>32980</td>
<td>Phillips</td>
<td>32</td>
</tr>
</tbody>
</table>
Interval Partitioning
Interval Partitioning

- Partitioning is key-enabling functionality for managing large volumes of data
  - one logical object for application transparency
  - multiple physical segments for administration

but

- Physical segmentation requires additional data management overhead
  - new partitions must be created on-time for new data

Automate the partition management
Interval Partitioning

• Interval Partitioning
  • extension to range partitioning
  • full automation for equi-sized range partitions
• Partitions are created as metadata information only
  • start partition is made persistent
• Segments are allocated as soon as new data arrives
  • no need to create new partitions
  • local indexes are created and maintained as well

No need for any partition management
Interval Partitioning

How it works

CREATE TABLE sales (order_date DATE, ...)
PARTITION BY RANGE (order_date)
INTERVAL (NUMTOYMINTERVAL(1, 'month'))
(PARTITION p_first VALUES LESS THAN ('01-FEB-2006'));

Table SALES

|----------|----------|----------|----------|----------|----------|

First segment is created
### Interval Partitioning

**How it works**

```sql
CREATE TABLE sales (order_date DATE, ...)
PARTITION BY RANGE (order_date)
INTERVAL (NUMTOYMINTERVAL(1, 'month'))
(PARTITION p_first VALUES LESS THAN ('01-FEB-2006'));
```

**Table SALES**

|----------|----------|----------|-----|----------|----------|----------|

Other partitions only exist in metadata
Interval Partitioning

How it works

CREATE TABLE sales (order_date DATE, ...)
PARTITION BY RANGE (order_date)
INTERVAL (NUMTOYMINTERVAL(1,'month'))
(PARTITION p_first VALUES LESS THAN ('01-FEB-2006');

Table SALES

|----------|----------|----------|----------|----------|----------|

New segment is automatically allocated

INSERT INTO sales (order_date DATE, ...)
VALUES ('04-MAR-2006', ...);
**Interval Partitioning**

*How it works*

```sql
CREATE TABLE sales (order_date DATE, ...)
PARTITION BY RANGE (order_date)
INTERVAL (NUMTOYMINTERVAL(1, 'month'))
(PARTITION p_first VALUES LESS THAN ('01-FEB-2006'));
```

Table SALES

|----------|----------|----------|-----|----------|-----|----------|----------|

... whenever data for a new partition arrives

```sql
INSERT INTO sales (order_date DATE, ...) VALUES ('17-OCT-2009', ...);
```
Interval Partitioning

*How it works*

- Interval partitioned table can have classical range and automated interval section
  - Automated new partition management plus full partition maintenance capabilities: *“Best of both worlds”*

Table SALES

|----------|----------|----------|----------|----------|----------|

---

ORACLE
Interval Partitioning

*How it works*

- Interval partitioned table can have classical range and automated interval section
  - Automated new partition management plus full partition maintenance capabilities: "Best of both worlds"

Table SALES

|------|----------|----------|----------|

Range partition section

MERGE and move old partitions for ILM
Interval Partitioning

How it works

- Interval partitioned table can have classical range and automated interval section
  - Automated new partition management plus full partition maintenance capabilities: “Best of both worlds”

Table SALES

Range partition section  
Interval partition section

2006  
Jan 2007  
Oct 2009  
Nov 2009

Insert new data
  - Automatic segment creation

INSERT INTO sales (order_date DATE, ...) VALUES ('13-NOV-2009', ...);
REF Partitioning
REF Partitioning

Business Problem
• Related tables benefit from same partitioning strategy
  • e.g. order – lineitem
• Redundant storage of the same information solves this problem
  • data overhead
  • maintenance overhead

Solution
• Oracle Database 11g introduces REF Partitioning
  • child table inherits the partitioning strategy of parent table through PK-FK relationship
  • intuitive modelling
• Enhanced Performance and Manageability
Before REF Partitioning

Table ORDERS

- RANGE(order_date)
- Primary key order_id

Table LINEITEMS

- Redundant storage of order_date
- Redundant maintenance

- RANGE(order_date)
- Foreign key order_id
REF Partitioning

Table ORDERS

- RANGE(order_date)
- Primary key order_id

PARTITION BY REFERENCE
- Partitioning key inherited through PK-FK relationship

Table LINEITEMS

- RANGE(order_date)
- Foreign key order_id
Partitioning Advisor

- Considers entire query workload to improve query performance
- Advises on partitioning methods
  - Range (equal-interval), range key and interval
  - Hash, hash key
- Integrated, non-conflicting advice with Indexes, MVs

SQL Workload

SQL Advisor

- SQL Plan Tuning
- SQL Structure Analysis
- Access Analysis

Partition Analysis

- Partition Advice

Well-tuned SQL & Schema

Packaged Apps

Custom Apps
Oracle Partitioning Summary

• Proven functionality in 7th generation
  • experience comes with age and customer usage
• Fundamental concepts for success
  • the most comprehensive and complete offering
• 8th generation introduces ground-breaking new functionality
  • extended COMPOSITE strategies further flexibility and performance
  • INTERVAL for simplicity, ease of management
  • VIRTUAL COLUMNS extend applicability with low maintenance
  • REF enhanced performance for related tables
Advanced Compression Option
Challenges

• Explosion in data volume managed by Enterprises
  • Government regulations (Sarbanes-Oxley, HIPPA, etc)
  • User generated content (Web 2.0)
• IT managers must support larger volumes of data with limited technology budgets
  • Need to optimize storage consumption
  • Also maintain acceptable application performance
• Intelligent and efficient compression technology can help address these challenges
Introducing Advanced Compression Option

• Oracle Database 11g introduces a comprehensive set of compression capabilities
  • Structured/Relational data compression
  • Unstructured data compression
  • Compression for backup data
  • Network transport compression
• Reduces resource requirements and costs
  • Storage System
  • Network Bandwidth
  • Memory Usage
Table Compression

- Introduced in Oracle9i Release 2
  - Supports compression during bulk load operations (Direct Load, CTAS)
  - Data modified using conventional DML not compressed
- Optimized compression algorithm for relational data
- Improved performance for queries accessing large amounts of data
  - Fewer IOs
  - Buffer Cache efficiency
- Data is compressed at the database block level
  - Each block contains own compression metadata – improves IO efficiency
  - Local symbol table dynamically adapts to data changes
- Compression can be specified at either the table or partition levels
- Completely transparent to applications
- Noticeable impact on write performance
OLTP Table Compression

• Oracle Database 11g extends compression for OLTP data
  • Support for conventional DML Operations (INSERT, UPDATE, DELETE)
• New algorithm significantly reduces write overhead
  • Batched compression ensures no impact for most OLTP transactions
• No impact on reads
  • Reads may actually see improved performance due to fewer IOs and enhanced memory efficiency
OLTP Table Compression

- Adaptable, continuous compression
- Compression automatically triggered when block usage reaches PCTFREE
- Compression eliminates holes created due to deletions and maximizes contiguous free space in block
Using OLTP Table Compression

• Requires database compatibility level at 11.1 or greater

• New Syntax extends the ‘COMPRESS’ keyword
  • COMPRESS [FOR {ALL | DIRECT_LOAD} OPERATIONS]
  • DIRECT_LOAD (DEFAULT)
    • Refers to Bulk load operations from 10g and prior releases
  • ALL
    • OLTP + Direct loads

• Enable compression for a new table
  
  CREATE TABLE t1 COMPRESS FOR ALL OPERATIONS

• Enable only direct load compression on existing table
  
  ALTER TABLE t2 COMPRESS
  • only new rows are compressed, existing rows are uncompressed
SecureFiles - Deduplication

- Enables storage of a single physical image for duplicate data
- Significantly reduces space consumption
- Dramatically improves write and copy operations
- No adverse impact on read operations
  - may actually improve read performance for cache data
- Duplicate detection happens within a table, partition or sub-partition
- Specially useful for content management, email applications and data archival applications
SecureFiles - Compression

• Huge storage savings
  • Industry standard compression algorithms
  • 2-3x compression for typical files (doc, pdf, xml)
  • Minimal CPU overhead during compression
• Automatically detects if SecureFile data is compressible
  • Skips compression for already compressed data
  • Auto-turn off compression when space savings are minimal or zero
• Two levels of compression provide different compression ratios
  • Compression Levels: MEDIUM (default), HIGH
  • Higher the degree of compression, higher the latency and CPU overhead incurred
• SecureFiles Compression is independent of table or index compression
• Server-side compression
  • Allows for random reads and writes to SecureFile data
  • Can be specified at a partition level
Data Pump Compression

- Metadata compression available since Oracle Database 10g
- Oracle Database 11g extends compression to table data during exports
  - no need to decompress before import
- Single step compression of both data and metadata
  - compressed data directly hits disk resulting in reduced disk space requirements
  - 75% reduction in dump file size on export of sample OE and SH schemas
- Compression factor comparable to GNU gzip utility
- Application transparent
  - complete Data Pump functionality available on compressed files
Backup data and Network transport Compression

- **Fast RMAN Compression**
  - compresses the backup set contents before writing them to disk or tape
  - no extra decompression steps are required during recovery when you use RMAN compression.
  - high performance, industry standard compression algorithm
    - 40% faster backup compression versus Oracle Database 10g
  - suitable for fast, incremental daily backups
  - reduces network usage

- **Data Guard Network Compression**
  - compression of redo traffic over the network
  - improves redo transport performance
    - gap resolution is up to 2x faster
Summary

- Advanced Compression Option contains comprehensive data compression capabilities for all types of data
  - Structured, Unstructured, Backup, Network Transport
- Reduces storage consumption by 2 to 3 times
- Improves read performance
- Enhances memory, buffer cache utilization
- Complete application transparency
- Benefits diverse application workloads
• Partitioning
• Compression
EXADATA

Extreme Performance
Data Warehousing
• Large data warehouses want to scan dozens, hundreds, or thousands of disks at full disk speed
• Pipes between disks and servers constrain bandwidth by 10x or more
• Result is that warehouses can become slower as they get bigger
Solutions To Data Bandwidth Bottleneck

- Ship less data through pipes
- Add more pipes
- Make pipes bigger
Exadata – A New Architecture
Breaks Data Bandwidth Bottleneck

• Exadata Ships Less Data Through Pipes
  • Query processing is moved into storage
to dramatically reduce data sent to
servers while offloading server CPUs

• Exadata has More Pipes
  • Modular storage “cell” building blocks
organized into Massively Parallel Grid
  • Bandwidth scales with capacity

• Exadata has Bigger Pipes
  • InfiniBand interconnect transfers data 5x
faster than Fibre Channel

Exadata Moves a Lot
Less Data a Lot Faster
Exadata – A New Architecture

- Exadata delivers brawny hardware for use by Oracle’s brainy software
- Hardware by HP
- Software by Oracle
- Sold by Oracle
- Support point of contact is Oracle
HP Oracle Exadata Storage Server

• Storage Server (Cell) is building block of Exadata Storage Grid
• Cells ship with all hardware and software components pre-installed
• Runs Exadata Storage Software, Oracle Infiniband protocol, Oracle Enterprise Linux 5.1, and HP hardware management software
  • Does not support non-database storage
• Hewlett Packard is exclusive hardware provider
  • HP has worldwide presence, great reputation, outstanding x86 platforms, high volume leadership, excellent support organization
  • Co-branded by HP and Oracle
• Absolutely no custom hardware - all parts off the shelf high-volume
HP Oracle Database Machine
Configured for High Performance Data Warehousing

- 8 Oracle Database servers
  - 64 Intel processor cores
  - Oracle Enterprise Linux
  - Oracle Real Application Clusters
- 14 Exadata Storage Servers
  - 50 to 168 TB raw storage
- InfiniBand switches

- Optimized, Certified, and Supported by Oracle
Add Performance and Capacity by Adding More Database Machines
Extreme Performance in Action

Scan TB of User Data In 3.5 sec.
Scalable Pure Storage

Scale to 18 cells in one rack

Each cell connects to 2 InfiniBand switches for Redundancy

Add racks to scale further

SAS raw capacity per rack: 65TB
SATA raw capacity per rack: 216TB
Peak throughput per rack: >18GB/s

InfiniBand links across racks for full connectivity
Exadata Product Capacity

<table>
<thead>
<tr>
<th>Model</th>
<th>Raw Storage</th>
<th>User Data</th>
<th>Data Bandwidth</th>
</tr>
</thead>
<tbody>
<tr>
<td>HP Exadata Storage Server Hardware SAS 12x450</td>
<td>3.6 TB</td>
<td>1 TB</td>
<td>1 GB/s</td>
</tr>
<tr>
<td>HP Exadata Storage Server Hardware SATA 12x1000</td>
<td>12 TB</td>
<td>3.3 TB</td>
<td>0.75 GB/s</td>
</tr>
<tr>
<td>HP Oracle Database Machine Hardware SAS 450</td>
<td>50 TB</td>
<td>14 TB</td>
<td>14 GB/s</td>
</tr>
<tr>
<td>HP Oracle Database Machine Hardware SATA 1000</td>
<td>168 TB</td>
<td>46 TB</td>
<td>10.5 GB/s</td>
</tr>
</tbody>
</table>
Exadata Changes the Game

- Exadata delivers brawny hardware for use by Oracle’s brainy software

- A new approach and architecture
- Delivers database intelligence and massively parallel scaling in the storage tier
  - Using state of the art industry standard hardware
- Eliminates all bottlenecks preventing high performance query processing
- Dramatic performance improvement for data warehousing
- Simplicity of appliance seamlessly integrated with the database