DATA WAREHOUSING
A PERSPECTIVE

Ian Abramson
Thoughtcorp
Director, Enterprise Data Group
June 2009

Thoughtcorp
CLEAR THOUGHT. CLEAR VALUE.

About Me

- Director of Enterprise Data at Thoughtcorp
- Specializing in Data Warehousing and Databases
- Over 20 years of Oracle Experience
- Co-authored 9 Oracle technology books
- IOUG President (www.ioug.org)
- BIWA Vice President
- ian.abramson@thoughtcorp.com

IOUG
independent oracle users group
For the Complete Technology & Database Professional

Today's Agenda

- What do you think?
- What do I think?
- Where is DW'ing today?
- How to respond with Oracle
  - Partitioning
  - Compression
  - Materialized Views
  - Data Transportation
  - Analytical functions

What do CIO's think

CIOs continue to invest in core technologies that can drive distinctive solutions

To what extent will each of the following technologies be a top 5 priority for you in 2008?

	2008	2007	2006	2008 Unweighted budget change
Business intelligence	1	1	1	11.2%
Enterprise applications (ERP, SCM, CRM, etc.)	2	2	**	8.0%
Servers and storage technologies	3	5	9	8.5%
Legacy modernization, upgrade or replacement	4	3	10	5.8%
Security technologies	5	6	2	8.5%
Technical infrastructure	6	8	12	4.7%
Networking, voice and data	7	4	8	6.8%
Collaboration technologies	8	10	4	7.8%
Document management	9	9	**	7.9%
Service-oriented architecture (SOA, SOBA)	10	7	6	6.7%

** New question for 2007

Gartner EXP "Gartner: Making the Difference - The 2008 CIO Agenda", Mark P. McDonald Tina Nunno Dave Aron, January 2008

Thoughtcorp
CLEAR THOUGHT. CLEAR VALUE.

How are data warehouses to be used?

- 4 Types of Data Warehouses:
  - Decision factory
    - This is today, reporting and dashboards
  - Information buffet
    - Adhoc query analysis
  - Hypothesis explored
    - Modeling
  - Discovery
    - Data mining

- Warehouses must make information open and available
- DW's are now mission critical
  - especially prevalent in global organization

Where are we Going?

- According to Gartner:
  - By 2009, 90% of organizations will have an OLTP system dependant on a data warehouse
  - By 2010, data warehouses will hold only 20% of the required strategic data

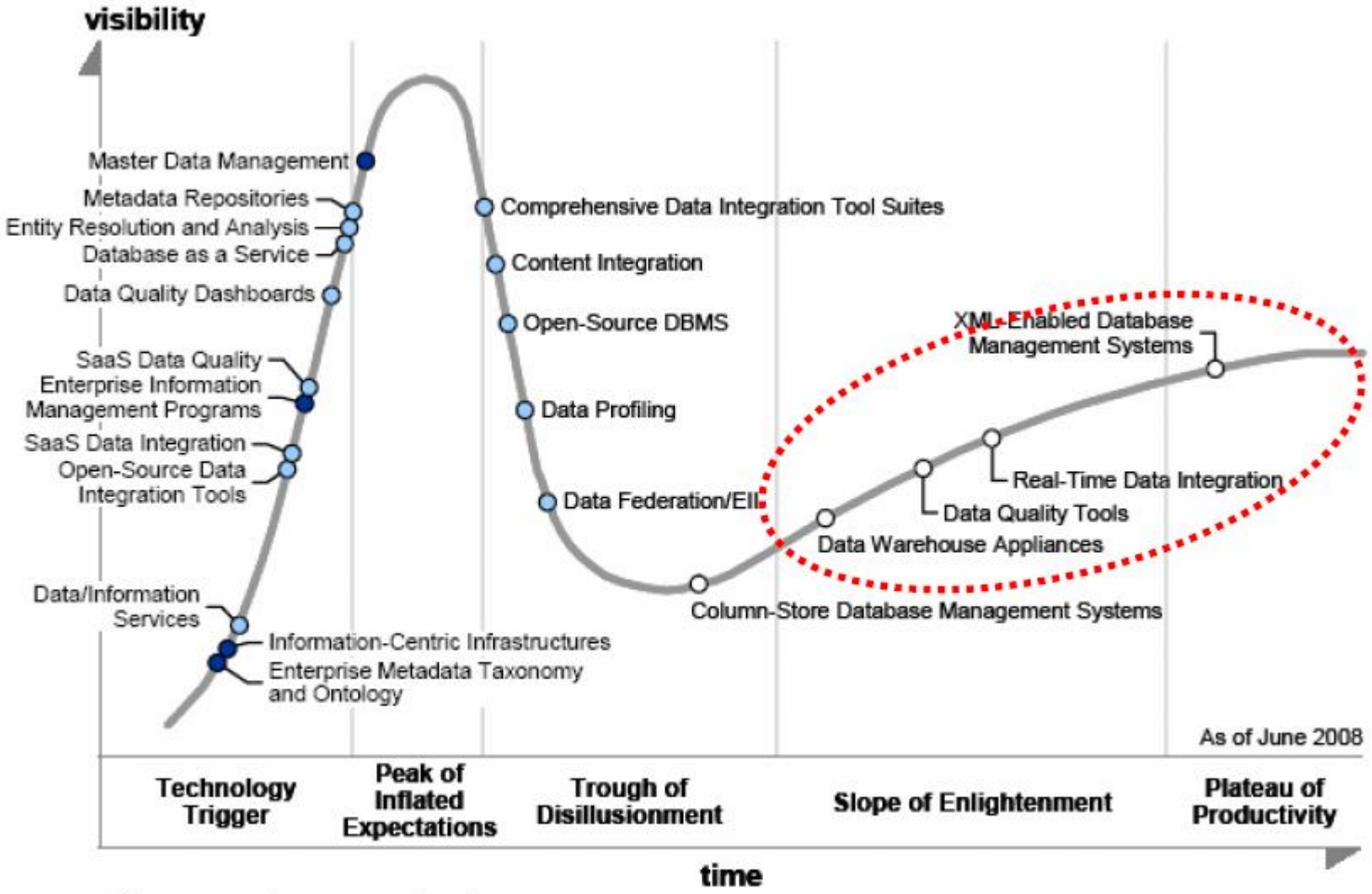
- Data is the primary assets of companies
- **Data Management** becomes the key
  - Governance
  - Integration
  - Quality
- Information will cross architectural layers
- In the coming years all organization MUST have an Master Data Management (MDM) strategy to be successful.

What has Caused the Data Explosion?

Regulatory Compliance Landscape	Government regulations like SOX, HIPAA government regulations that mandate storing historical data for a certain number of years
Web 2.0	A new kind of data source – Web 2.0 such as social networks, blogs leading to various forms of semi-structured and unstructured data. Some of these data is being stored in the database, some in ECM
Multi media content	Bandwidth has become cheap and increasing amounts of multimedia content is being generated and stored
Migration of Legacy Applications	As legacy applications from main frames and other files based databases is being migrated to RDBMS, increasing volumes of data is being stored inside the database
Click-stream	Click-stream and personalization data continues to explore for online sites

Thoughtcorp
CLEAR THOUGHT. CLEAR VALUE.

Gartner 2008 Data Management Hype Cycle



Thoughtcorp
CLEAR THOUGHT. CLEAR VALUE.

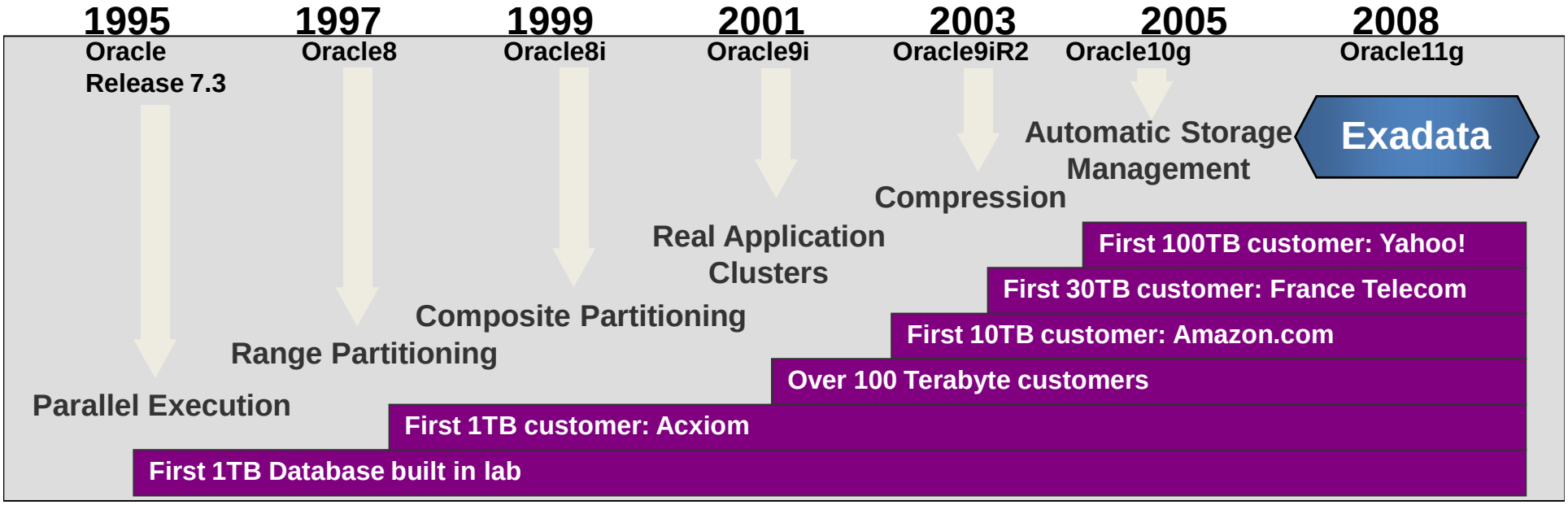
Gartner Priority Matrix for Data Management

benefit	years to mainstream adoption			
	less than 2 years	2 to 5 years	5 to 10 years	more than 10 years
transformational		**Data Service Architectures**	Information-Centric Infrastructure	
high	**Data Quality Tools** **Data Warehouse Appliances** **Linux as a Mission-Critical DBMS Platform** **XML-Enabled Database Management Systems**	Comprehensive Data Integration Tool Suites Content Integration Entity Resolution and Analysis Real-Time Data Integration	Enterprise Information Management Master Data Management	
moderate	Data Quality Dashboards OSS DBMS for Non-Mission-Critical Applications	Data Federation/EII Data Profiling OSS DBMS for Mission-Critical Applications SaaS Data Integration and Data Quality XQuery	Metadata Ontology Management	
low		Open-Source Data Integration Tools		

As of June 2007

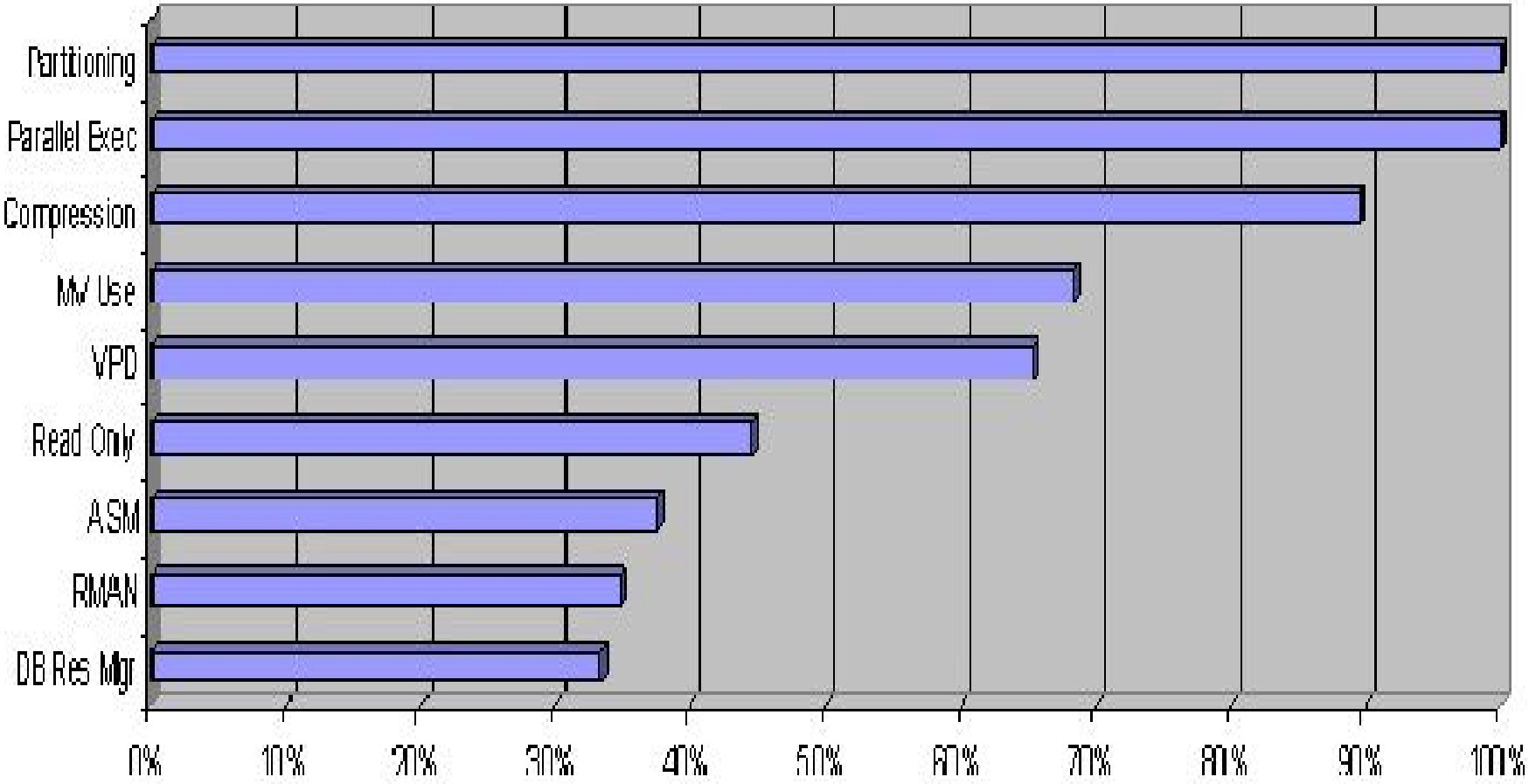
Source: Gartner (June 2007)

Oracle & Data Warehousing



Thoughtcorp
CLEAR THOUGHT. CLEAR VALUE.

Oracle DW Features Usage



Source: TB Club Report: A survey of 30+ multi-TB Oracle DW's

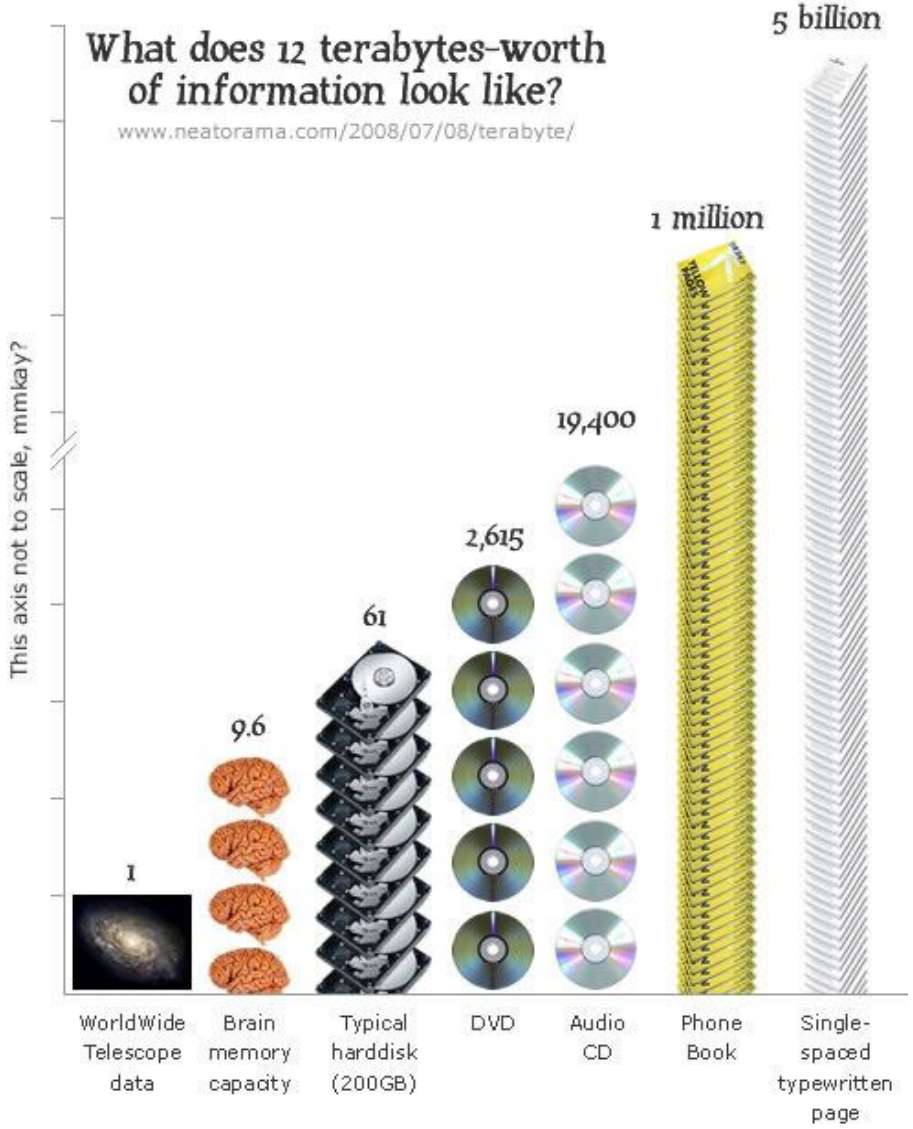
Thoughtcorp
CLEAR THOUGHT. CLEAR VALUE.

Oracle's New 11g BI and DW Features

- VLDB
  - –•Composite Range-Range
  - –•Composite List-Range
  - –•Composite List-List
  - –•Composite List-Hash
  - –REF Partitioning
  - –Virtual Column Partitioning
  - –Compression enhancements
- ••Performance
  - –Query Result Cache
- •Manageability
  - –Partition Advisor
  - –Interval Partitioning
  - –SQL Plan Management
  - –Automatic SQL Tuning with Self-Learning Capabilities
  - –Enhanced Optimizer Statistics Maintenance
  - –Multi-Column Optimizer Statistics
  - –ASM Fast Resync, Fast VLDB Startup and other enhancements
- •SQL
  - –SQL Pivot and Unpivot
  - –Continuous Query Notification
- ••OLAP
  - –Materialized view refresh and SQL rewrite
  - –Continued database integration
    - •Cube metadata in the Data Dictionary
    - •Fine-grained data security on cubes
  - –Simplified application development
    - •Fully declarative cube calculations
    - •Cost-Based Aggregation
    - •Simpler calculation definition
  - –Data Mining
    - •Simplified development and deployment of models
    - •Supermodels: data preparation combined with mining model
    - •Additional packaged predictive analytics
    - •Integration in database dictionary
  - –New algorithms: "General Linear Models"
    - •Encapsulates several widely used analytic methods
    - •Multivariate linear regression; logistic regression
- •ETL
  - –OWB Repository installed with Database by default
  - –Seibel connector
  - –Graphical creation of views, materialized views

Thoughtcorp
CLEAR THOUGHT. CLEAR VALUE.

Storing Data



Thoughtcorp
CLEAR THOUGHT. CLEAR VALUE.

Partitioning: The Evolution

Oracle Version	Functionality	Performance Impact	Maintenance
Oracle8	Range partitioning Global range indexes	"Static" partition pruning	Basic maintenance operations: add, drop, exchange
Oracle8i	Hash and composite range-hash partitioning	Partition-wise joins "Dynamic" pruning	Merge operation
Oracle9i	List partitioning		Global index maintenance
Oracle9i R2	Composite range-List partitioning	Fast partition split	
Oracle10g	Global hash indexes		Local Index maintenance
Oracle10g R2	1M partitions per table	"Multi-dimensional pruning	Fast drop table

Thoughtcorp
CLEAR THOUGHT. CLEAR VALUE.

Partitioning Methods

- Range:

```
CREATE TABLE quarterly_sales
(txn_date         DATE,
 deptno           NUMBER,
 item_no          VARCHAR2(20),
 txn_amount       NUMBER,
 state            VARCHAR2(2))
PARTITION BY RANGE (txn_date)
(PARTITION q1_2007 VALUES
 LESS THAN(TO_DATE('1-APR-2007','DD-MON-YYYY')),
 PARTITION q2_2007 VALUES
 LESS THAN(TO_DATE('1-JUL-2007','DD-MON-YYYY')),
 PARTITION q3_2007 VALUES
 LESS THAN(TO_DATE('1-OCT-2007','DD-MON-YYYY')),
 PARTITION q4_2007 VALUES
 LESS THAN(TO_DATE('1-JAN-2008','DD-MON-YYYY')),
 PARTITION out_of_range VALUES (DEFAULT));
```

Partitioning Methods

- Hash:

```
CREATE TABLE CUSTOMERS
  (CUSTOMER_ID            NUMBER,
   CREATE_DATE            DATE,
   NAME                   VARCHAR2(80),
   ADDRESS                VARCHAR2(80),
   CITY                   VARCHAR2(40),
   PROV_CD                VARCHAR2(2),
   POSTAL_CODE            VARCHAR2(10),
   CUSTOMER_TYP_KY        NUMBER(6),
   CUSTOMER_TYP_DESC      VARCHAR2(80))
 PARTITION BY HASH (CUSTOMER_ID)
 PARTITIONS 4
 STORE IN (TBS_01, TBS_02, TBS_03, TBS_04);
```

Partitioning Methods

- List:

```
CREATE TABLE BRANCHES
  (BRANCH_REGION          VARCHAR2(6),
   BRANCH_ID              NUMBER,
   CREATE_DATE            DATE,
   NAME                   VARCHAR2(80),
   ADDRESS                VARCHAR2(80),
   CITY                   VARCHAR2(40),
   PROV_CD                VARCHAR2(2),
   POSTAL_CODE            VARCHAR2(10),
   BRANCH_TYP_KY          NUMBER(6))
PARTITION BY LIST (BRANCH_REGION)
(PARTITION BRANCH_REGION1 VALUES('EAST', 'NORTH'),
 PARTITION BRANCH_REGION2 VALUES ('WEST'),
 PARTITION BRANCH_REGION3 VALUES ('SOUTH'),
 PARTITION BRANCH_REGION4 VALUES ('VIP'),
 PARTITION BRANCH_other VALUES(DEFAULT));
```

Composite Partitioning

- Two-dimensional partitioning schemes
- Extensions in Oracle Database 11g:
- Examples:
  - List-range:
    - Partition by country, then by week
    - Partition by line-of-business, then by week
    - Partition by Line of Business and month

	Range	List	Hash
Range	**11g**	**9iR2**	**8i**
List	**11g**	**11g**	**11g**

Composite (Range & Hash)

```
CREATE TABLE sales
( prod_id NUMBER(6)
, cust_id NUMBER
, time_id DATE
, channel_id CHAR(1)
, promo_id NUMBER(6)
, quantity_sold NUMBER(3)
, amount_sold NUMBER(10,2)
)
PARTITION BY RANGE (time_id) SUBPARTITION BY HASH (cust_id)
SUBPARTITIONS 8 STORE IN (ts1, ts2, ts3, ts4)
( PARTITION sales_q1_2007 VALUES LESS THAN (TO_DATE('01-APR-
   2007','dd-MON-yyyy'))
, PARTITION sales_q2_2007 VALUES LESS THAN (TO_DATE('01-JUL-
   2007','dd-MON-yyyy'))
, PARTITION sales_q3_2007 VALUES LESS THAN (TO_DATE('01-OCT-
   2007','dd-MON-yyyy'))
, PARTITION sales_q4_2007 VALUES LESS THAN (TO_DATE('01-JAN-
   2008','dd-MON-yyyy'))
);
```

Composite (List & Range)

```
CREATE TABLE quarterly_regional_sales
(deptno number,
 item_no varchar2(20),
 txn_date date,
 txn_amount number,
 prov_state varchar2(2))
TABLESPACE ts4
  PARTITION BY RANGE (txn_date)
 SUBPARTITION BY LIST (state)
(PARTITION q1_2007 VALUES LESS THAN (TO_DATE('1-APR-2007','DD-MON-YYYY'))
(SUBPARTITION q1_2007_north VALUES ('ON', 'QC', 'NS',BC),
SUBPARTITION q1_2007_southwest VALUES ('AZ', 'UT', 'NM'),
SUBPARTITION q1_2007_northeast VALUES ('NY', 'VM', 'NJ'),
SUBPARTITION q1_2007_southeast VALUES ('FL', 'GA'),
SUBPARTITION q1_2007_northcentral VALUES ('SD', 'WI'),
SUBPARTITION q1_2007_southcentral VALUES ('OK', 'TX')
),
PARTITION q2_2007 VALUES LESS THAN ( TO_DATE('1-JUL-2007','DD-MON-YYYY'))
(SUBPARTITION q1_2007_north VALUES ('ON', 'QC', 'NS',BC),
SUBPARTITION q2_2007_southwest VALUES ('AZ', 'UT', 'NM'),
SUBPARTITION q2_2007_northeast VALUES ('NY', 'VM', 'NJ'),
SUBPARTITION q2_2007_southeast VALUES ('FL', 'GA'),
SUBPARTITION q2_2007_northcentral VALUES ('SD', 'WI'),
SUBPARTITION q2_2007_southcentral VALUES ('OK', 'TX')
),
PARTITION q3_2007 VALUES LESS THAN (TO_DATE('1-OCT-2007','DD-MON-YYYY'))
(SUBPARTITION q1_2007_north VALUES ('ON', 'QC', 'NS',BC),
SUBPARTITION q3_2007_southwest VALUES ('AZ', 'UT', 'NM'),
SUBPARTITION q3_2007_northeast VALUES ('NY', 'VM', 'NJ'),
SUBPARTITION q3_2007_southeast VALUES ('FL', 'GA'),
SUBPARTITION q3_2007_northcentral VALUES ('SD', 'WI'),
SUBPARTITION q3_2007_southcentral VALUES ('OK', 'TX')
),
PARTITION q4_2007 VALUES LESS THAN ( TO_DATE('1-JAN-2008','DD-MON-YYYY'))
(SUBPARTITION q1_2007_north VALUES ('ON', 'QC', 'NS',BC),
SUBPARTITION q4_2007_southwest VALUES ('AZ', 'UT', 'NM'),
SUBPARTITION q4_2007_northeast VALUES ('NY', 'VM', 'NJ'),
SUBPARTITION q4_2007_southeast VALUES ('FL', 'GA'),
SUBPARTITION q4_2007_northcentral VALUES ('SD', 'WI'),
SUBPARTITION q4_2007_southcentral VALUES ('OK', 'TX')));
```


Thoughtcorp ©2009

Thoughtcorp
CLEAR THOUGHT. CLEAR VALUE.

Composite (List & Range)

```
CREATE TABLE quarterly_regional_sales
(deptno number,
  item_no varchar2(20),
  txn_date date,
  txn_amount number,
  prov_state varchar2(2))
TABLESPACE ts4
   PARTITION BY RANGE (txn_date)
  SUBPARTITION BY LIST (state)
(PARTITION q1_2007 VALUES LESS THAN (TO_DATE('1-APR-2007','DD-MON-YYYY'))
(SUBPARTITION q1_2007_north VALUES (‘ON', ‘QC‘, ‘NS’,BC),
SUBPARTITION q1_2007_southwest VALUES ('AZ', 'UT', 'NM'),
SUBPARTITION q1_2007_northeast VALUES ('NY', 'VM', 'NJ'),
SUBPARTITION q1_2007_southeast VALUES ('FL', 'GA'),
SUBPARTITION q1_2007_northcentral VALUES ('SD', 'WI'),
SUBPARTITION q1_2007_southcentral VALUES ('OK', 'TX')
),
SUBPARTITION q3_2007_northcentral VALUES ('SD', 'WI'),
SUBPARTITION q3_2007_southcentral VALUES ('OK', 'TX')
),
PARTITION q4_2007 VALUES LESS THAN ( TO_DATE('1-JAN-2008','DD-MON-YYYY'))
(SUBPARTITION q1_2007_north VALUES (‘ON', ‘QC‘, ‘NS’,BC),
SUBPARTITION q4_2007_southwest VALUES ('AZ', 'UT', 'NM'),
SUBPARTITION q4_2007_northeast VALUES ('NY', 'VM', 'NJ'),
SUBPARTITION q4_2007_southeast VALUES ('FL', 'GA'),
SUBPARTITION q4_2007_northcentral VALUES ('SD', 'WI'),
SUBPARTITION q4_2007_southcentral VALUES ('OK', 'TX')));
```


Thoughtcorp
CLEAR THOUGHT. CLEAR VALUE.

Partition Maintenance

Maintenance Operation	Range Composite Range-*	Interval Composite Interval-*	Hash	List Composite List-*	Reference
Adding Partitions	ADD PARTITION	ADD PARTITION	ADD PARTITION	ADD PARTITION	N/A[1]
Coalescing Partitions	N/A	N/A	COALESCE PARTITION	N/A	N/A[1]
Dropping Partitions	DROP PARTITION	DROP PARTITION	N/A	DROP PARTITION	N/A[1]
Exchanging Partitions	EXCHANGE PARTITION	EXCHANGE PARTITION	EXCHANGE PARTITION	EXCHANGE PARTITION	EXCHANGE PARTITION
Merging Partitions	MERGE PARTITIONS	MERGE PARTITIONS	N/A	MERGE PARTITIONS	N/A[1]
Modifying Default Attributes	MODIFY DEFAULT ATTRIBUTES	MODIFY DEFAULT ATTRIBUTES	MODIFY DEFAULT ATTRIBUTES	MODIFY DEFAULT ATTRIBUTES	MODIFY DEFAULT ATTRIBUTES
Modifying Real Attributes of Partitions	MODIFY PARTITION	MODIFY PARTITION	MODIFY PARTITION	MODIFY PARTITION	MODIFY PARTITION
Modifying List Partitions: Adding Values	N/A	N/A	N/A	MODIFY PARTITION ... ADD VALUES	N/A
Modifying List Partitions: Dropping Values	N/A	N/A	N/A	MODIFY PARTITION ... DROP VALUES	N/A
Moving Partitions	MOVE PARTITION	MOVE PARTITION	MOVE PARTITION	MOVE PARTITION	MOVE PARTITION
Renaming Partitions	RENAME PARTITION	RENAME PARTITION	RENAME PARTITION	RENAME PARTITION	RENAME PARTITION
Splitting Partitions	SPLIT PARTITION	SPLIT PARTITION	N/A	SPLIT PARTITION	N/A[1]
Truncating Partitions	TRUNCATE PARTITION	TRUNCATE PARTITION	TRUNCATE PARTITION	TRUNCATE PARTITION	TRUNCATE PARTITION

Thoughtcorp ©2009

Thoughtcorp
CLEAR THOUGHT. CLEAR VALUE.

Sub-Partition Maintenance

Maintenance Operation	Composite *-Range	Composite *-Hash	Composite *-List
Adding Partitions	MODIFY PARTITION ... ADD SUBPARTITION	MODIFY PARTITION ... ADD SUBPARTITION	MODIFY PARTITION ... ADD SUBPARTITION
Coalescing Partitions	N/A	MODIFY PARTITION ... COALESCE SUBPARTITION	N/A
Dropping Partitions	DROP SUBPARTITION	N/A	DROP SUBPARTITION
Exchanging Partitions	EXCHANGE SUBPARTITION	N/A	EXCHANGE SUBPARTITION
Merging Partitions	MERGE SUBPARTITIONS	N/A	MERGE SUBPARTITIONS
Modifying Default Attributes	MODIFY DEFAULT ATTRIBUTES FOR PARTITION	MODIFY DEFAULT ATTRIBUTES FOR PARTITION	MODIFY DEFAULT ATTRIBUTES FOR PARTITION
Modifying Real Attributes of Partitions	MODIFY SUBPARTITION	MODIFY SUBPARTITION	MODIFY SUBPARTITION
Modifying List Partitions: Adding Values	N/A	N/A	MODIFY SUBPARTITION ... ADD VALUES
Modifying List Partitions: Dropping Values	N/A	N/A	MODIFY SUBPARTITION ... DROP VALUES
Modifying a Subpartition Template	SET SUBPARTITION TEMPLATE	SET SUBPARTITION TEMPLATE	SET SUBPARTITION TEMPLATE
Moving Partitions	MOVE SUBPARTITION	MOVE SUBPARTITION	MOVE SUBPARTITION
Renaming Partitions	RENAME SUBPARTITION	RENAME SUBPARTITION	RENAME SUBPARTITION
Splitting Partitions	SPLIT SUBPARTITION	N/A	SPLIT SUBPARTITION
Truncating Partitions	TRUNCATE SUBPARTITION	TRUNCATE SUBPARTITION	TRUNCATE SUBPARTITION

Thoughtcorp ©2009

Thoughtcorp
CLEAR THOUGHT. CLEAR VALUE.

Maintenance Commands (Add & Drop)

- Add range-list partition:

```
ALTER TABLE quarterly_regional_sales
ADD PARTITION q1_2000 VALUES LESS THAN (TO_DATE('1-APR-2008','DD-MON-YYYY'))
STORAGE (INITIAL 20K NEXT 20K) TABLESPACE ts3 NOLOGGING
(
SUBPARTITION q1_2008_north VALUES (‘ON', ‘QC'),
SUBPARTITION q1_2008_southwest VALUES ('AZ', 'UT', 'NM'),
SUBPARTITION q1_2008_northeast VALUES ('NY', 'VM', 'NJ'),
SUBPARTITION q1_2008_southeast VALUES ('FL', 'GA'),
SUBPARTITION q1_2008_northcentral VALUES ('SD', 'WI'),
SUBPARTITION q1_2008_southcentral VALUES ('OK', 'TX')
);
```

- Reduce Hash table partitions:
  - `ALTER TABLE ouu1 COALESCE PARTITION;`
- Dropping Partitions:
  - `ALTER TABLE sales DROP PARTITION dec2005;`
  - `ALTER INDEX sales_area_ix REBUILD;`
  - `ALTER TABLE sales DROP PARTITION dec2005 UPDATE INDEXES;`

More Maintenance (Exchange)

```
ALTER TABLE stocks
EXCHANGE PARTITION p3 WITH TABLE stock_table_3;

ALTER TABLE vet_cats
SPLIT PARTITION
fee_katy at (100) INTO ( PARTITION
fee_katy1, PARTITION fee_katy2);

ALTER INDEX JAF1 REBUILD PARTITION fee_katy1;
ALTER INDEX JAF1 REBUILD PARTITION fee_katy2;
ALTER INDEX VET REBUILD PARTITION vet_parta;
ALTER INDEX VET REBUILD PARTITION vet_partb;

ALTER TABLE sales
DROP PARTITION sales_2000
UPDATE GLOBAL INDEXES PARALLEL;
```

Virtual Columns

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

Using Virtual Columns

- 1) The way we usually create tables:

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

- 2)Now add a calculated column within the object:

```
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)))
```

- 3) Now Partition on the calculated 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)))
partition by list (acc_branch) ...
```

Interval Partitioning (Pre-11g)

- Partitioning is key-enabling functionality for
- managing large volumes of data
- One logical object for application transparency
- Multiple physical segments for Administration
- Improves Manageability, Availability, and Performance

- **BUT**

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

- Can be a Maintenance Headache!

Interval Partitioning (With 11g)

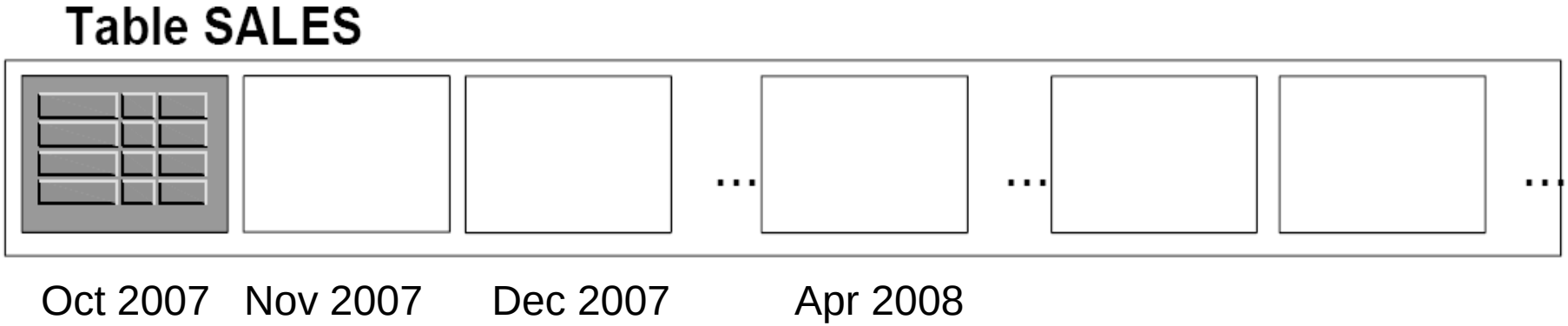
- Interval Partitioning
  - Extension to Range Partitioning
  - Full automation for equi-sized range partitions
- Partitions are created as metadata information only
  - Start Partition is made persistent
  - Segments are allocated as soon as new data arrives
- No need to create new partitions
- Local indexes are created and maintained as well

- No need for managing these partitions

Interval Partitioning the SQL

1) Create table with first partition:

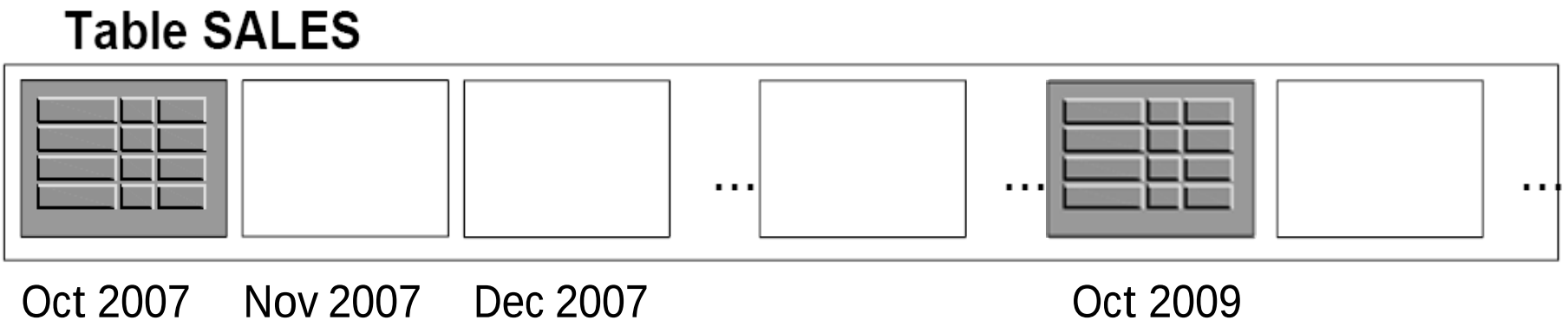
```
CREATE TABLE sales (order_date DATE, ...)
PARTITON BY RANGE (order_date)
INTERVAL(NUMTOYMINTERVAL(1,'month')
(PARTITION p_first VALUES
       LESS THAN ('01-OCT-2007');
```



When Inserting into Interval Tables

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

Insert will then cause the Oct 2009 partition to be physically allocated when we actually need it



Add Intervals to an Existing Table

- Range partitioned tables can be extended into interval partitioned tables
  - Simple metadata command
  - Investment protection
- Does not affect current data that may already be partitioned.

```
ALTER TABLE sales (order_date DATE, ...)
SET INTERVAL(NUMTOYMINTERVAL(1,'month');
```

REF Partitioning

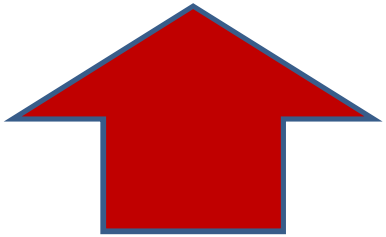
- Oracle Database 11g introduces REF Partitioning
- Child table inherits the partitioning strategy of parent table
- through PK-FK relationship
- Intuitive modeling
- Enhanced Performance and Manageability

Reference Partitions

```
CREATE TABLE orders
(order_id NUMBER(12),
order_date TIMESTAMP WITH LOCAL TIME ZONE,
order_mode VARCHAR2(8),
customer_id NUMBER(6),
order_status NUMBER(2),
order_total NUMBER(8,2),
sales_rep_id NUMBER(6),
promotion_id NUMBER(6),
CONSTRAINT orders_pk PRIMARY KEY(order_id)
)
PARTITION BY RANGE(order_date)
(PARTITION Q1_2007 VALUES LESS THAN (TO_DATE('01-APR-2007','DD-MON-YYYY')),
PARTITION Q2_2007 VALUES LESS THAN (TO_DATE('01-JUL-2007','DD-MON-YYYY')),
PARTITION Q3_2007 VALUES LESS THAN (TO_DATE('01-OCT-2007','DD-MON-YYYY')),
PARTITION Q4_2007 VALUES LESS THAN (TO_DATE('01-JAN-2008','DD-MON-YYYY'))
);
```

Reference Partitions

```
CREATE TABLE order_items
( order_id NUMBER(12) NOT NULL,
line_item_id NUMBER(3) NOT NULL,
product_id NUMBER(6) NOT NULL,
unit_price NUMBER(8,2),
quantity NUMBER(8),
CONSTRAINT order_items_fk
FOREIGN KEY(order_id) REFERENCES orders(order_id)
)
PARTITION BY REFERENCE(order_items_fk);
```



Partition Indexes

- Global indexes
  - Refer to data across partitions
  - Must be range or hash
  - May index any type of partition table
  - Does not have to have same number of partitions as the underlying table
  - Must be a prefixed index
  - Does not currently support non-prefixed global indexes
  - Most partition maintenance operations invalidate global partitioned indexes, forcing an index rebuild.

- Local Indexes
  - Partitions match those of the underlying table
  - If partitions are changed the local index will manage the maintenance to ensure it remains in line with the update partition definitions

Partition Indexes the SQL

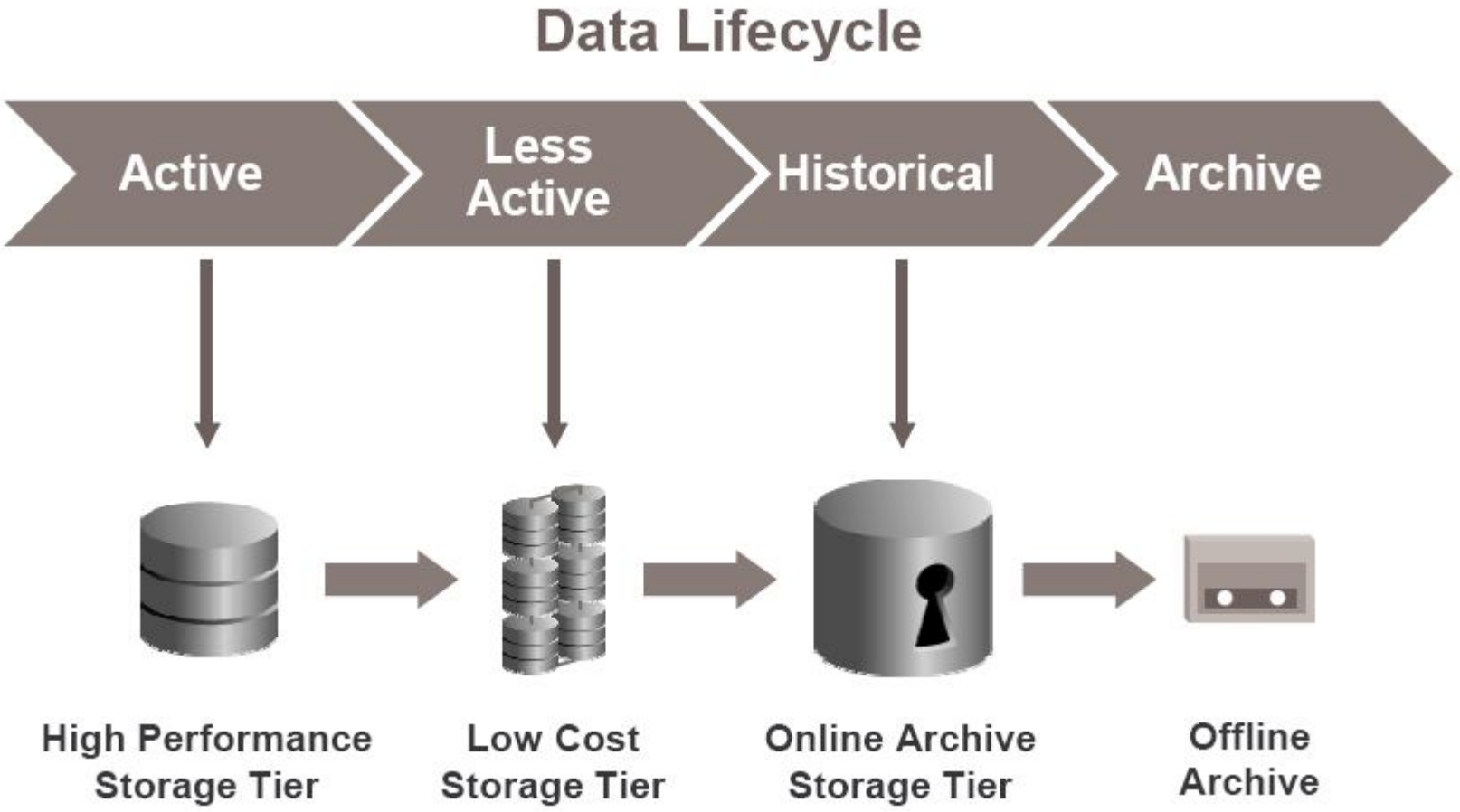
- Local Indexes:

```
CREATE BITMAP INDEX prod_id_ix
ON sales(prod_id) LOCAL
PARALLEL NOLOGGING;
```

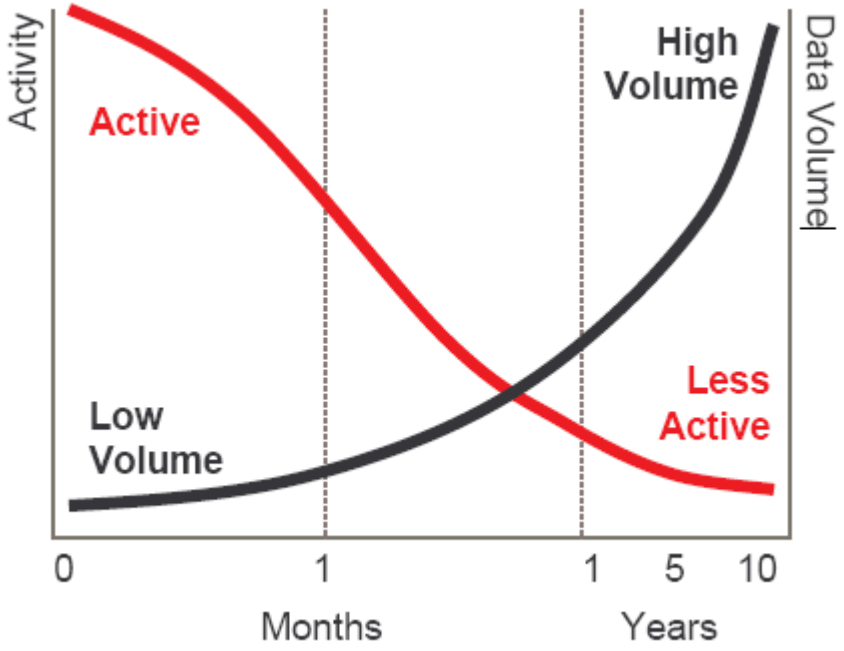
- Global Indexes:

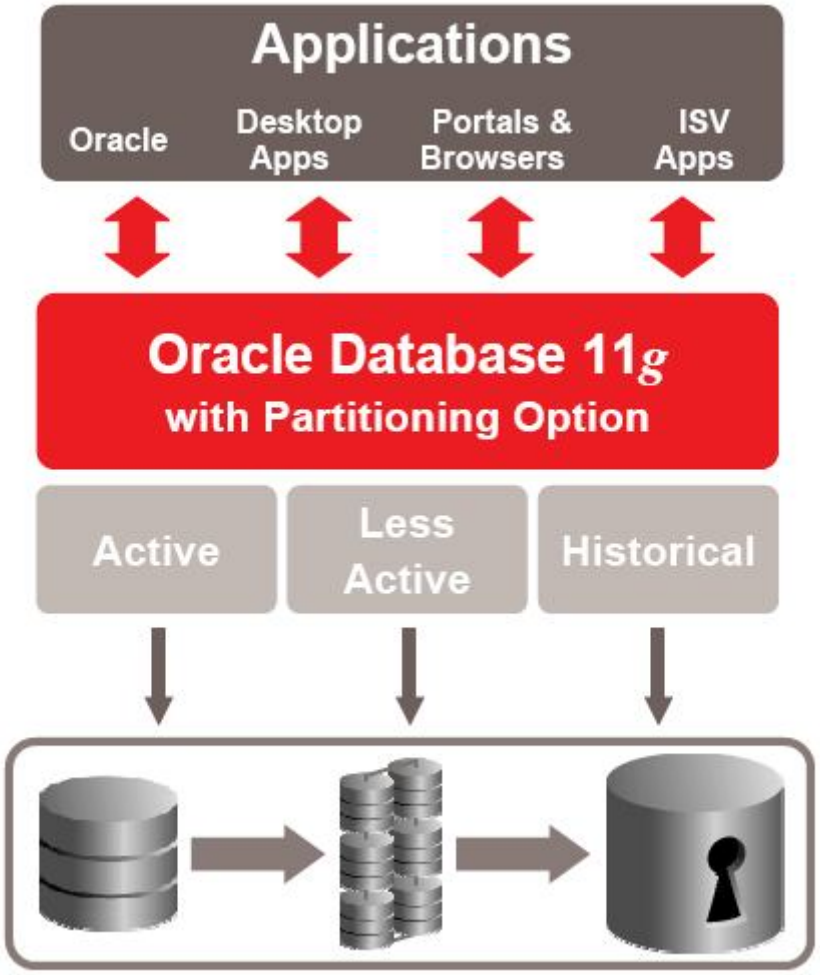
```
CREATE INDEX cust_id_prod_id_global_ix
ON sales(cust_id,prod_id)
GLOBAL PARTITION BY HASH (cust_id)
( PARTITION p1 TABLESPACE tbs1
, PARTITION p2 TABLESPACE tbs2
, PARTITION p3 TABLESPACE tbs3
, PARTITION p4 TABLESPACE tbs4
)
PARALLEL NOLOGGING;
```

Information Lifecycle Management (ILM) – 11g



Information Lifecycle Management (ILM) – 11g





Source: OTN - Oracle Information Lifecycle Management

Setting up ILM

- Define Data Classes
  - By Date
  - By age
  - By Product
  - By privacy level
- Implement Partitions to support Data Classes
- Create Storage Tiers for Data Classes
  - High performance
  - Low cost
  - Online Archive
  - Offline Archive (optional)
- Create Data Access and Migration Policies

Setting up Storage Tiers

- **High Performance Disks:**

```
CREATE TABLESPACE "ILM_HIGH_PERFORMANCE"
DATAFILE '/hdsk1/oradata/orcl/ilm_high_perf' SIZE 10G
AUTOEXTEND ON NEXT 1G MAXSIZE UNLIMITED LOGGING
EXTENT MANAGEMENT LOCAL SEGMENT SPACE MANAGEMENT AUTO ;
```

- **Low Cost Storage:**

```
CREATE SMALLFILE TABLESPACE "ILM_LOW_COST"
DATAFILE '/lcdsk1/oradata/orcl/ilm_low_cost' SIZE 100G
AUTOEXTEND ON NEXT 10G MAXSIZE UNLIMITED LOGGING
EXTENT MANAGEMENT LOCAL SEGMENT SPACE MANAGEMENT AUTO ;
```

- **More Low Cost Storage**

```
CREATE SMALLFILE TABLESPACE "ILM_ONLINE_ARCHIVE"
DATAFILE '/lcdisk10/oradata/orcl/ilm_oarchive1' SIZE
  15T AUTOEXTEND ON NEXT 1G MAXSIZE UNLIMITED,
'/lcdisk11/oradata/orcl/ilm_oarchive2' SIZE 5T
AUTOEXTEND ON NEXT 1G MAXSIZE UNLIMITED
LOGGING EXTENT MANAGEMENT LOCAL SEGMENT SPACE
  MANAGEMENT AUTO ;
```

A Table Utilizing ILM

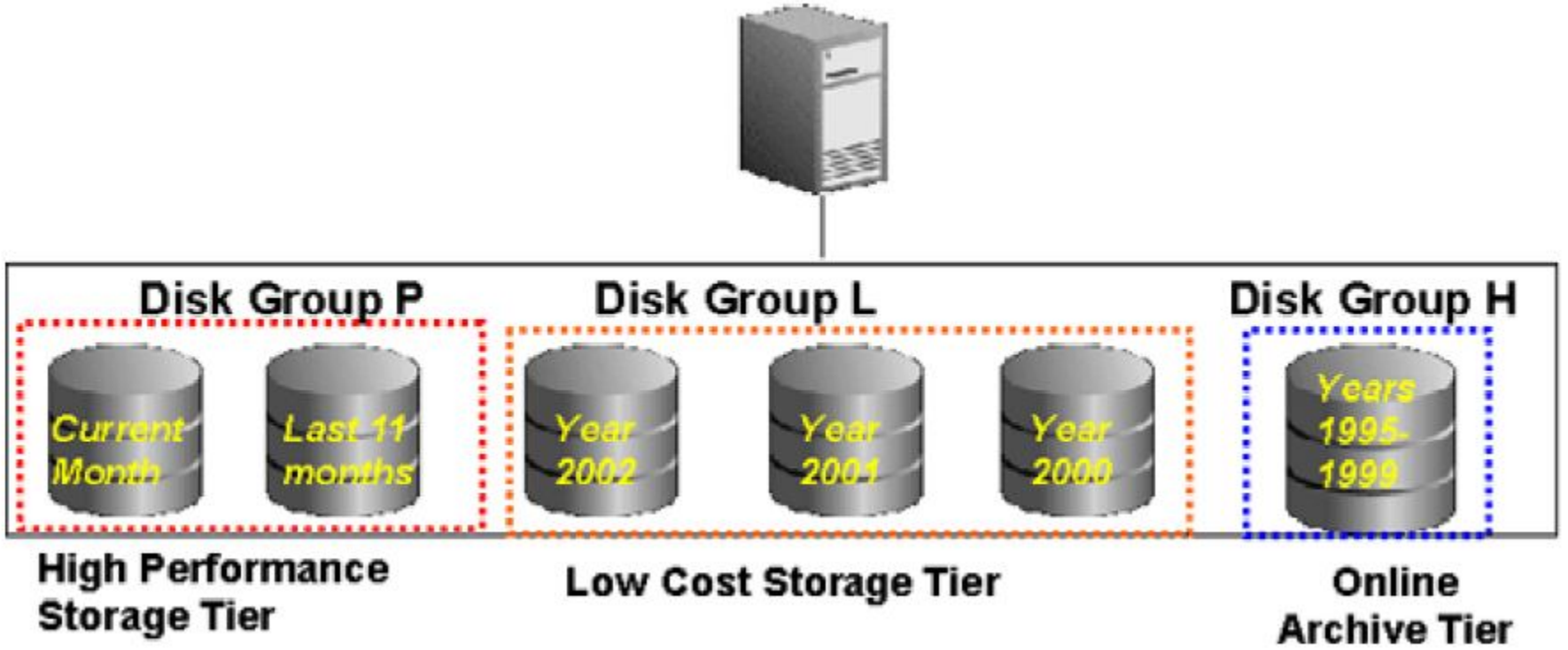
```
CREATE TABLE orders (
prod_id NUMBER NOT NULL,
cust_id NUMBER NOT NULL,
time_id DATE NOT NULL,
channel_id NUMBER NOT NULL,
promo_id NUMBER NOT NULL,
quantity_sold NUMBER(10,2) NOT NULL,
amount_sold NUMBER(10,2) NOT NULL)
--
-- partitions
--
PARTITION BY RANGE (time_id)
( partition orders_pre_2006 VALUES LESS THAN(TO_DATE('1/1/2006','DD/MM/YYYY'))
TABLESPACE ILM_ONLINE_ARCHIVE,
partition orders_2006 VALUES LESS THAN TO_DATE('1/1/2007','DD/MM/YYYY'))
TABLESPACE ILM_ONLINE_ARCHIVE ,
partition orders_janmar_2007 VALUES LESS THAN (TO_DATE('1/4/2007','DD/MM/YYYY'))
   TABLESPACE ILM_LOW_COST ,
partition orders_aprjun_2007 VALUES LESS THAN (TO_DATE('1/7/2007','DD/MM/YYYY'))
   TABLESPACE ILM_LOW_COST ,
partition orders_julsep_2007 VALUES LESS THAN(TO_DATE('1/10/2007','DD/MM/YYYY'))
   TABLESPACE ILM_LOW_COST,
partition orders_octdec_2008 VALUES LESS THAN(TO_DATE('1/1/2009','DD/MM/YYYY'))
TABLESPACE ILM_HIGH_PERFORMANCE,
PARTITION other_orders VALUES(DEFAULT) TABLESPACE ILM_HIGH_PERFORMANCE );
```


Thoughtcorp ©2009

Thoughtcorp
CLEAR THOUGHT. CLEAR VALUE.

ILM and ASM

- Use disk groups for partitions



Read Only Data

- To support “locked” data
- To improve performance

- If you want a Read-only storage tier
  - `ALTER TABLESPACE ilm_online_archive READ ONLY;`
- If you want a read-only table (11g):
  - `ALTER TABLE online_orders READ ONLY;`

Materialized Views

- In data warehouses, you can use materialized views to:
  - Pre-compute and store aggregated data such as the sum of sales. Materialized views in these environments
  - Pre-compute joins with or without aggregations.

- A materialized view eliminates the overhead associated with expensive joins and aggregations for a large or important class of queries.
- You can use materialized views to increase the speed of queries on very large databases.
- The SQL Access Advisor, which recommends materialized views, partitions, and indexes to create.
- TUNE_MVIEW, which shows you how to make your materialized view fast refreshable and use general query rewrite.

Query Rewrite

- The Oracle optimizer automatically transforms SQL statements to retrieve data from materialized views instead of the fact tables
- Transformation is transparent to the end user
- Materialized views can be created or dropped without affecting the application
- Oracle Parameters:
  - QUERY_REWRITE_ENABLED parameter must be set to TRUE.
  - Associated Materialized views must have the ENABLE QUERY REWRITE clause.
  - Cost-based optimization must be used
  - QUERY_REWRITE_INTEGRITY parameter should be set accordingly to the constraints states
    (Enforced, Trusted, Stale_Tolerated)
  - SQL statement can include hints (REWRITE/NOREWRITE)

Thoughtcorp
CLEAR THOUGHT. CLEAR VALUE.

Materialized views in SQL (compress & partitions)

```
CREATE MATERIALIZED VIEW country_sales
PARTITION BY HASH (country_id)
PARTITIONS 16
COMPRESS FOR ALL OPERATIONS
PARALLEL NOLOGGING
ENABLE QUERY REWRITE
AS SELECT co.country_id
   , co.country_name
   , co.country_subregion
   , co.country_region
   , sum(sa.quantity_sold) country_quantity_sold
   , sum(sa.amount_sold) country_amount_sold
FROM sales sa
   , customers cu
   , countries co
WHERE sa.cust_id = cu.cust_id
AND cu.country_id = co.country_id
GROUP BY co.country_id
       , co.country_name
       , co.country_subregion
       , co.country_region;
```

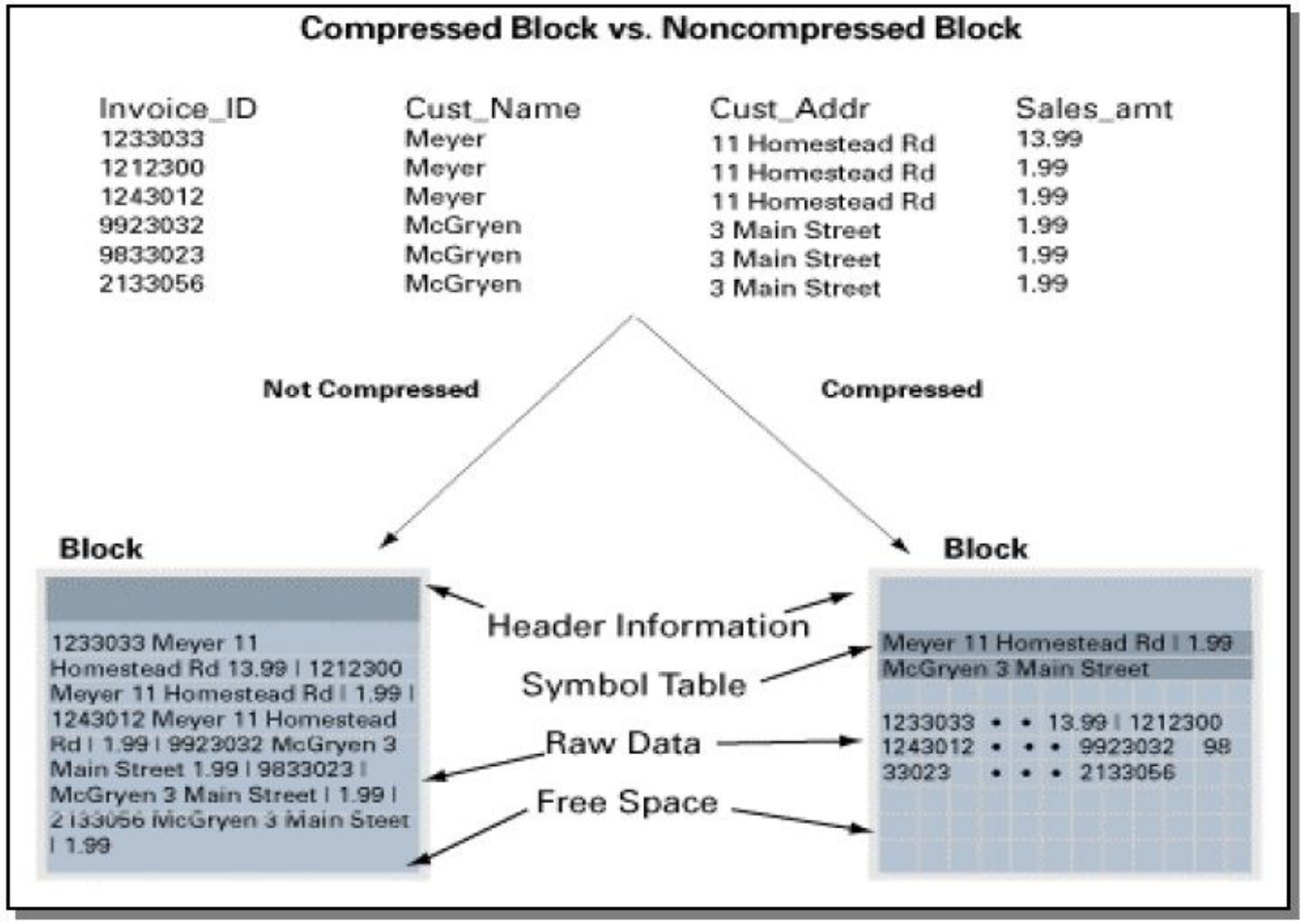
Oracle Advanced Compression

- With 11g now supports, insert, update and delete
- Disk savings up to 3x
- More efficient use of disk
- More effective retrieval of information
- Data stored in two forms:
  - Data
  - Pointers
- Great for facts
- For more efficient storage sort your data
- Stores any data type

Sample Compression Savings

Table Name	Original Size (MB)	Compressed Size (MB)	% of Original	% Compressed
C_District	5	5	100	0
C_Item	11	11	100	0
C_New_Order	624	448	72	28
C_Customer	93,882	78,249	83	17
C_History	7,680	3,327	43	**57**
C_Order_Line	104,528	72,703	70	30
C_Stock	66,304	56,065	85	15

Compression Storage



Source: OTN - Oracle Advanced Compression White Paper (June 07)

Data Compression

- 3 Ways to Compresses repeating data

```
CREATE TABLE MARKET_ORDERS
(CUST_KEY  number NOT NULL,
 PROD_KEY  number NOT NULL,
 STORE_KEY number NOT NULL,
 SALE_DATE DATE   NOT NULL,
 QUANTITY NUMBER(10,2) NOT NULL )
 COMPRESS ;


ALTER TABLE MARKET_ORDERS COMPRESS;

CREATE MATERIALIZED VIEW MV_SALES_COMP COMPRESS
AS
SELECT P.PART_NAME, H.STORE_ID, H.SALE_DATE, H.QUANTITY
FROM SALES_HISTORY H, PARTS P
WHERE P.PART_ID = H.PART_ID;
```


Thoughtcorp
CLEAR THOUGHT. CLEAR VALUE.

Compression in Oracle11g

- Oracle Advanced Compression is an option for EE
- Support compression on INSERT, UPDATE and
  - 10g only supported compression for bulk data-loading operations.
- Advanced compression allows for an 2-3 x compression rate of structured and unstructured data.
- Backup files can be compressed when using RMAN or DataPump
- DataGuard redo data may be compressed

External Tables for Loading or Unloading Data

- A type of Oracle database table
- Points to an operating system file
- DDL refers to directory object
- Can INSERT INTO….SELECT …. FROM …..MERGE
- Can perform multi table UPDATES
- Can JOIN with regular database tables
- 9i –Read only, table can only be queried
- 10g –Can also extract to, using ORACLE_DATAPUMP type

External Tables

First we create a “directory”:

```
> create directory dump_dir as '/u01/ext_tab';
```

We define the table where we link the files to be used for external tables:

```
create table appl_ext
organization external
(
   type oracle_datapump
   default directory dump_dir
   location ('appl_1_ext.dmp',
             'appl_2_ext.dmp',
             'appl_3_ext.dmp',
             'appl_4_ext.dmp'))
parallel 4
as
select * from appl;
>
```


Thoughtcorp
CLEAR THOUGHT. CLEAR VALUE.

Another External Table

```
CREATE OR REPLACE DIRECTORY bdump AS
'c:\oracle\product\admin\orcl\bdump\';

CREATE TABLE log_table (TEXT VARCHAR2(400))
ORGANIZATION EXTERNAL (
 TYPE oracle_loader
 DEFAULT DIRECTORY bdump
 ACCESS PARAMETERS (
  RECORDS DELIMITED BY NEWLINE
  NOBADFILE NODISCARDFILE NOLOGFILE
  FIELDS TERMINATED BY '0x0A'
  MISSING FIELD VALUES ARE NULL)
 LOCATION ('alert_orcl.log'))
REJECT LIMIT unlimited;

SELECT * FROM log_table;
```

Other Methods for Data Movement (ETL)

- SQL*Loader
- Export/Import (DataPump)
- External Tables
- OWB/ODI
- Transportable tablespaces

Transportable Tablespaces

- Allow a database to be migrated from one platform to another
- Great ETL option
- Provide an easier and more efficient means for content providers to publish structured data and distribute it to customers running Oracle Database on different platforms
- Simplify the distribution of data from a data warehouse environment to data marts

Steps to Transport a Tablespace

1. For cross-platform transport, check the endian format of both platforms by querying the V$TRANSPORTABLE_PLATFORM view.
   - Ignore this step if you are transporting your tablespace set to the same platform.
2. Pick a self-contained set of tablespaces.
3. Generate a transportable tablespace set.
4. Transport the tablespace set.
5. Import the tablespace set.

Indexes

- Why Bitmap indexes in DWs?
  - Mostly used in Data warehouse environment
  - Efficient for columns with low cardinality
  - Smaller index storage requirement
  - On partitions fact tables must be created as local
  - Star Transformation with bitmap indexes – transform the original query to star queries
  - Should not be used with concurrent inserts
  - Bitmap Join Index – bitmap index generated from join of two or more tables

Thoughtcorp
CLEAR THOUGHT. CLEAR VALUE.

SQL Tips

- SQL for Aggregations
  - ROLLUP Extension to GROUP BY
  - CUBE Extension to GROUP BY
  - GROUPING Functions (Windowing)
  - GROUPING SETS Expression
- SQL for Analysis and Reporting
  - Ranking Functions
  - Windowing Aggregate Functions
  - Reporting Aggregate Functions
  - LAG/LEAD Functions
  - FIRST/LAST Functions
  - Inverse Percentile Functions
  - Hypothetical Rank and Distribution Functions
  - Linear Regression Functions
  - Pivoting Operations
  - CASE Expressions
- SQL Modeling
- OLAP and Data Mining
- SQL Performance Advisor (DBMS_SQLPA Package)

OLAP versus Data Mining

- OLAP provides:
  - Summary data and generates rich calculations. For example,
  - OLAP answers questions like:
    - How do sales of mutual funds in North America for this quarter compare with sales a year ago?
    - What can we predict for sales next quarter?
    - What is the trend as measured by percent change?
- Data mining discovers hidden patterns in data. Data mining operates at a detail level instead of a summary level.
- Data mining answers questions like:
  - Who is likely to buy a mutual fund in the next six moths, and what are the characteristics of these likely buyers?"
- OLAP and data mining can complementary

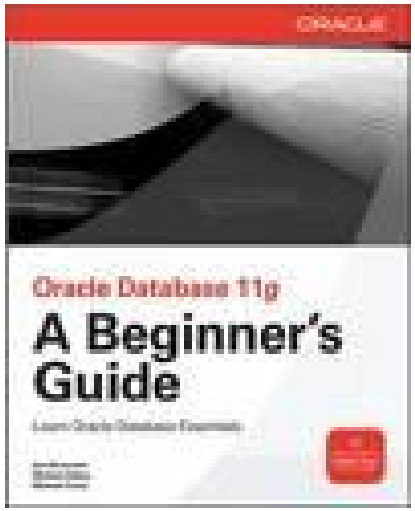
Summary

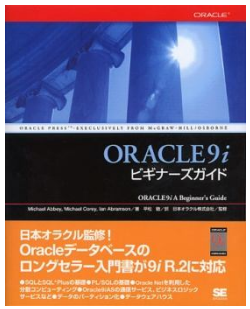
- Many features to use:
  - Partitioning
  - Compression
  - Parallel Query
  - Materialized Views
  - ETL
  - SQL

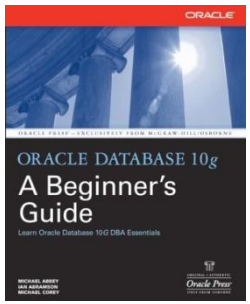
Thanks

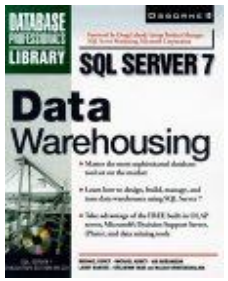
Q&A

Ian Abramson
Director Enterprise Data Group
Thoughtcorp Inc.
ian.abramson@thoughtcorp.com
416-254-9286









Thoughtcorp
CLEAR THOUGHT. CLEAR VALUE.

References:

- Oracle Database VLDB and Partitioning Guide (v11.1)
- Oracle Database Data Warehousing Guide (v11.1)
- Oracle presentation: Partitioning in Database 11g
- http://www.oracle.com/technology/pub/articles/oracle-database-11g-top-features/11g-spa.html