Implementing Oracle database12c’s Heat Map and Automatic Data Optimization to Optimize the Database Storage Cost and Performance

NZOUG OTN Day
DECEMBER 4 • FRIDAY

Kai Yu
Oracle Solutions Engineering
Dell Inc
Agenda

• Database Storage Challenges for IT Organizations
• Oracle 12c Information Lifecycle Management (ILM)
  – Heat Map
  – Automatic Data Optimization
• Examples of using Oracle ILM
• Questions
About Author

Kai Yu, Senior Principal Architect, Dell Database Engineering

- 20 years Oracle DBA/Apps DBAS and Solutions Engineering
- Specializing in Oracle RAC/DB, Oracle VM and Oracle EBS
- Oracle ACE Director, Oracle papers author/presenter
- 2011 OAUG Innovator of Year, 2012 Oracle Excellence Award: Technologist of the Year: Cloud Architect by Oracle Magazine
- Co-author Apress Book “Expert Oracle RAC 12c”
My Work: Dell Oracle Solutions Engineering

- Flash SSD based high performance storage solution for database
  - Remove storage bottlenecks for Oracle Databases.
  - Shared storage for Oracle RAC database

**Performance of DAAD over 96-Disk Baseline**

- 1915% increase in Max IOPS
- 272% increase in Max MB/s
- 10% decrease in Latency

![Diagram of Dell Acceleration Appliance for Database (DAAD)]
My Work: Dell Oracle Solutions Engineering

- Preconfigured Ready Infrastructure with all flash storage for Oracle Database and DBaaS.
Database Storage Challenges for IT Organizations
Database Storage Challenges

- Exponential data growth posts another Challenge to IT organization.
  - Exposition in online access and contents such as pictures and videos.
    Mobile, Big Data, Social Media, etc
  - Compliance with government regulation on data retention
  - Database performance impacted by volumes of data
  - Even the configuration event is related IO as is was the free buffer wait caused by the slow writing dirty data from the buffer cache to disks

- IT Budgets Challenge
  - More Storage capacity is very costly
  - Faster storage is even costly even the price is coming down
  - IT budgets are not much increasing in most organization
  - How to improve the performance, how to increase capacity without much growing cost
  - Adapting storage tiering is the way to improve the storage efficiency
Database Storage Challenges

• Storage IOs presents major bottlenecks to database performance
  – As shown the following example, user I/O is the major bottleneck

  – Even the configuration event is related IO as is was the free buffer wait caused by the slow writing dirty data from the buffer cache to disks
Storage tiering: performance vs cost

• Pricing of high performance storage such as SSD is still higher.
• Only small percentage data is frequently accessed transactional data, and majority of data is less frequently accessed.
• An example of the tiered storage design:
  – Tier 1: small set of hot data (frequently accessed read/write intensive), using SLC SSD
  – Tier 2: less frequently accessed or Read Intensive workloads: using MLC SSDs or 15k SAS for sequential workloads
  – Tier 3: large volume of archival data :7.2k SATA/NL-SAS for low performance
Implementation of Storage tiering

- **Methods to implement storage tiering**
  - Manual method: manually assign or move the objects to storage tier.
  - Build-in storage tiering in the storage product
  - By Information management feature of database product.

- **Manual method by Storage admin/DBAs**
  - Manage the tired storage based on the performance, access pattern, sizing and storage cost.
  - Identify IO performance bottleneck and move the objects to different storage tier.
  - Example of two tiers storage with manual method:
    Tier 1: 4 PCI-SSD, Tier 2: MD3220 SAS with 24 X 15k rpms HDs
Implementation of Storage tiering

- Performance of four configurations
  - Config1: all objects stored in Tier 2 (HDs)
  - Config2: all indexes stored in Tier the rest stored in Tier 2
  - Config3: all indexes + one active table stored in Tier 1, the rest in Tier 2
  - Config4: all indexes + four active tables stored in Tier 1 the rest stored in Tier 2

![TPS Increase Percentage Comparison Graph](chart)

![Response times (ms) vs User load Graph](chart)
Implementation of Storage tiering

- Build-in storage tiering feature in the storage product
  - The storage appliance moves blocks or pages of data from one type of disk to another type of disk based on rules or usage.
  - Examples of storage Products:
    - Dell Compellent
    - Dell EqualLogic
    - EMC FAST
    - NetApp

- Advantages:
  - Fully automated process.
  - The simplest way to implement the storage tiering

- Disadvantages:
  - Not application aware.
  - Can’t combine different vendors.
  - Can’t work if the tiered storage on multiple vendors storage
Implementation of Storage tiering

- By Information management feature of applications.
  - Applications keep track of the usage of the data
  - System admin can define the rules for data moving based on the cost
  - Applications move data between these storage tiers based on the data usages and performance
  - Oracle Information Lifecycle Management provides this feature.

- Advantages:
  - The data moving process can be automated by setting the rules.
  - Work well storage tiering among multiple storage products
Oracle Information Lifecycle Management (ILM)
Information Lifecycle Management (ILM)

• ILM is the practice of applying specific policies for effective information management.
• With respect to databases, ILM refers to the processes and practices of aligning digital records—table “rows”—with the most appropriate and cost effective IT infrastructure at each phase of its useful life.
• Examples:
  – Moving transaction data to a data warehouse for improved analytics performance
  – Compressing tables or partitions as data ages to reduce storage capacity
  – Moving data for completed transactions to read-only storage
New features in Oracle Database 12c

- While ILM has been practiced for decades the explosion of new data and the proliferation of storage technologies has lead to innovations in improving management of automated ILM.

- Heat Map
  - Stores system-generated data usage statistics at the block and segment levels
  - Segment level Heat Map tracks the time of last modification and access of tables and partitions
  - Row level Heat Map tracks modification times for individual rows (aggregated to the block level)
  - Heat Map skips internal access done for system tasks—automatically excluding Stats Gathering, DDLs or Table Redefinitions

- Automatic Data Optimization
  - Allows you to create policies for data compression (Smart Compression) and data movement, to implement storage and compression tiering
  - Smart Compression refers to the ability to utilize Heat Map information to associate compression policies, and compression levels, with actual data usage
Automatic Data Optimization (ADO)

- Compression types:

<table>
<thead>
<tr>
<th>Compression algorithm</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>COMPRESS;</td>
<td>Basic table compression. The same as “ROW STORE COMRESS BASIC”</td>
</tr>
<tr>
<td>ROW STORE COMPRESS BASIC:</td>
<td>The standard option (see above)</td>
</tr>
<tr>
<td>ROW STORE COMPRESS ADVANCED:</td>
<td>Oracle DB compresses the data during Data Manipulation Language (DML) operations. This is recommended for Online Transaction Processing Applications (OLTP)</td>
</tr>
<tr>
<td>COLUMNSTORE COMPRESS FOR QUERY:</td>
<td>Enables hybrid columnar compression in which the data is transformed into a column-oriented format then compressed. This is useful for Data Warehouses (OLAP)</td>
</tr>
<tr>
<td>COLUMNSTORE COMPRESS FOR ARCHIVE:</td>
<td>Same as above but compressed at a higher level. This is useful for archiving as it has a negative impact on performance.</td>
</tr>
<tr>
<td>NOCOMPRESS:</td>
<td>Disables table compression. The default.</td>
</tr>
</tbody>
</table>

- Data can be compressed at the table, partition/segment, or row level (rows are aggregated to the block level)
- Data can be tiered at the table or partition level (there is no row-based tiering)
Oracle ILM - Administration

- **DBMS_ILM_ADMIN**
  - `customize_ilm` Provides an interface to customized Automatic Data Optimization (ADO) policy execution.
  - `disable_ilm` Allows you to set parameters for ILM.
  - `enable_ilm` Procedure to turn off background ILM
  - `clear_heat_map_all` Procedure to turn on background ILM
  - `set_heat_map_all` Procedure to delete all rows except dummy row
  - `set_heat_map_table` Procedure to update/insert a row for all tables
  - `clear_heat_map_table` Procedure to update/insert a row for this table/segment
  - `set_heat_map_start` Procedure to clear all/some statistics for table
  - `set_heat_map_start` Procedure to set start date of heat map data
Heap Map and ADO

- Heat Map tracks usage information at the row and segment level:
  
  Views: V$/All/DBA/USER_HEAT_MAP_SEGMENT

- Add ADO policy: three parts
  - Ado action: data compression or data movement
  - Level: segment or row level for table or partition
  - Condition: what condition will initiate an ADO action.

ADO policy examples:

```
alter table CUSTOMER
  ILM ADD POLICY
row store compress advanced segment
  after 10 days of no modification;
```

```
SQL> alter table ORDER MODIFY PARTITION ORD1
  ILM ADD POLICY compress segment after 5 days of no modification;
```
Automatic Data Optimization (ADO)

- Ado policy for data movement

```
SQL> ALTER TABLE ORDER ilm ADD POLICY TIER TO T2DATA;
Table altered.

SQL> alter table ORDER MODIFY PARTITION ORD2
ILM ADD POLICY tier to T2DATA;
```

- Ado policy for data movement by condition:

```
SQL> CREATE OR REPLACE FUNCTION MOVE3D
( OBJ_ID IN NUMBER )
RETURN BOOLEAN AS
  w_time date;
  now date;
BEGIN
  select SEGMENT_WRITE_TIME into w_time
  from user_objects u join user_heat_map_segment h
  on u.object_name=h.object_name and u.subobject_name=h.subobject_name
  where u.object_name=OBJ_ID;
  now := systimestamp;
  if (trunc(now - w_time) > 2) then RETURN TRUE; else RETURN FALSE; end if;
END MOVE3D;

SQL> ALTER TABLE orders ILM ADD POLICY TIER TO tier_loc ON move3d;
```
Automatic Data Optimization (ADO)

- Thresholds at which ADO data movement policy will be started/stopped
- Tablespace percent used
- Tablespace percent free

Data Moving conditions:
- %used > 85% → start
- Move condition to meet
- Data moving stop:
- %free < 30% → stop

- Tablespaces on tier storage (in this example)
  - Tier 1 on 12 x SLA SSDs in Raid 10:
    - T1DATA tablespace
  - Tier 2 on 12x 1TB 7.2k rpm Hard disks in RAID10
    - T2DATA tablespace
Automatic Data Optimization

Examples

This is a subtitle or bulleted list
Example 1: ADO policy to compress tables

- **Example 1:** Create segment-level ADO policy to automatically compress table after 10 days no modification
- Create compression policy

```sql
SQL> alter table CUSTOMER
2    ILM ADD POLICY
3   ROW STORE COMPRESS ADVANCED SEGMENT
4   AFTER 10 DAYS OF NO MODIFICATION;
Table altered.
```

Check the policy via USER_ILMDataMovementPolicies

```sql
SQL> select POLICY_NAME, ACTION_TYPE, SCOPE, COMPRESSION_LEVEL, CONDITION_TYPE, CONDITION_DAYS
2   from user_ilmdatamovementpolicies where object_name = 'CUSTOMER';

<table>
<thead>
<tr>
<th>POLICY_NAME</th>
<th>ACTION_TYPE</th>
<th>SCOPE</th>
<th>COMPRESSION_LEVEL</th>
<th>CONDITION_TYPE</th>
<th>CONDITION_DAYS</th>
</tr>
</thead>
<tbody>
<tr>
<td>P102</td>
<td>COMPRESSION SEGMENT ADVANCED</td>
<td>LAST MODIFICATION TIME</td>
<td>10</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
```

Display Heat map tracking data via user_heat_map_segment

```sql
SQL> SELECT OBJECT_NAME, SEGMENT_WRITE_TIME
2   from user_Heat_Map_Segment
3   where OBJECT_NAME = 'CUSTOMER';

<table>
<thead>
<tr>
<th>OBJECT_NAME</th>
<th>SEGMENT_WRITE_TIME</th>
</tr>
</thead>
<tbody>
<tr>
<td>CUSTOMER</td>
<td>1-SEP-14</td>
</tr>
</tbody>
</table>

SQL> !date
Fri Sep 19 03:29:50 CDT 2014
Example 1: ADO policy to compress tables

- Execute the ADO policy immediately

```sql
DECLARE
  v_executionid NUMBER;
BEGIN
  DBMS_ILM.EXECUTE_ILM (
    owner                => 'HAMMERBASE',
    object_name          => 'CUSTOMER',
    task_id              => v_executionid);
END;
/
PL/SQL procedure successfully completed.
```

- Check the task status: policy_name: P102, task_id: 2146

```sql
SELECT task_id, policy_name, object_name, selected_for_execution, job_name
FROM user_ilmvaluationdetails
WHERE policy_name = 'P102';
```

```sql
2146  P102  CUSTOMER  SELECTED FOR EXECUTION  ILMJOB2288
```

- Verify

```sql
SELECT task_id, state, start_time, completion_time
2 FROM user_ilmtasks WHERE task_id = 2146;
```

```sql
<table>
<thead>
<tr>
<th>TASK_ID</th>
<th>STATE</th>
<th>START_TIME</th>
<th>COMPLETION_TIME</th>
</tr>
</thead>
<tbody>
<tr>
<td>2146</td>
<td>ACTIVE</td>
<td>19-SEP-14 03.40.01.399956 AM</td>
<td>19-SEP-14 03.46.56.493519 AM</td>
</tr>
</tbody>
</table>
```

```sql
SELECT compression, compress_for FROM user_tables WHERE table_name = 'CUSTOMER';
```

```sql
<table>
<thead>
<tr>
<th>COMPRESS</th>
<th>COMPRESS_FOR</th>
</tr>
</thead>
<tbody>
<tr>
<td>ENABLED</td>
<td>ADVANCED</td>
</tr>
</tbody>
</table>
```
Example 2: ADO policy to move tables

- Example 2: Create ADO policy to move objects from T1DATA to T2DATA

Current TBS:

```
SQL> select tablespace_name from user_tables where table_name = 'CUSTOMER';

TABLESPACE_NAME
-----------------------
T1DATA
```

- Check Tablespaces space usage: PCT_FREE, PCT_USED

```
SQL> SELECT a.tablespace_name TBS_NAME, ROUND ( (NVL (b.bytes_free, 0) / a.bytes_alloc) * 100, 2) PCT_FREE, 100 - ROUND ( (NVL (b.bytes_free, 0) / a.bytes_alloc) * 100, 2) PCT_USED FROM (SELECT tablespace_name, SUM (BYTES) bytes_alloc FROM dba_data_files GROUP BY tablespace_name) a, (SELECT tablespace_name, SUM (BYTES) bytes_free FROM dba_free_space GROUP BY tablespace_name) b WHERE a.tablespace_name = b.tablespace_name(+);

<table>
<thead>
<tr>
<th>TBS_NAME</th>
<th>PCT_FREE</th>
<th>PCT_USED</th>
</tr>
</thead>
<tbody>
<tr>
<td>T1DATA</td>
<td>8.38</td>
<td>91.62</td>
</tr>
<tr>
<td>T2DATA</td>
<td>85.62</td>
<td>14.38</td>
</tr>
</tbody>
</table>
```
Example 2: ADO policy to move tables

- Check ADO parameters for ADO data migration threshold
  T1DATA: PCT_USED: 91.62% > 85% threshold

```
SQL> select * from dba_ilmparameters;
NAME      VALUE
---------- ----------
ENABLED    1
JOB LIMIT  10
EXECUTION MODE  3
EXECUTION INTERVAL  15
TBS PERCENT USED  85
TBS PERCENT FREE  30
6 rows selected.
```

- Start migration when pct_used > 85%, stop migration when pct_free < 30%
- Create data movement policy:

```
SQL> ALTER TABLE ORDER ilm ADD POLICY TIER TO T2DATA;
Table altered.
```

- Check the policy: policy name: P122

```
SQL> select POLICY_NAME, ACTION_TYPE, SCOPE, TIER_TABLESPACE
           from user_ilmdatamovemovementpolicies where object_name = 'CUSTOMER';

POLICY_NAME ACTION_TYPE SCOPE  TIER_TABLESPACE
---------- --------- ------ ----------
P122        STORAGE SEGMENT T2DATA
```
Example 2: ADO policy to move tables

- Execute the ADO policy immediately

```
DECLARE
  v_executionid NUMBER;
BEGIN
  DBMS_ILM.EXECUTE_ILM (owner => 'HAMMERBASE',
                         object_name => 'CUSTOMER',
                         task_id    => v_executionid);
END;
/
```

- Check the task status: `policy_name: P122, task_id: 2214`

```
SQL> select task_id, policy_name, object_name, selected_for_execution, job_name
       from user_ilmevaluationdetails
       where policy_name = 'P122';

  TASK_ID | POLICY_NAME | OBJECT_NAME | SELECTED_FOR_EXECUTION | JOB_NAME |
---------|-------------|-------------|------------------------|----------|
    2214   | P122        | CUSTOMER    | SELECTED FOR EXECUTION  | ILMJOB2354 |

SQL> select TASK_ID, STATE, START_TIME, COMPLETION_TIME
       from user_ilmtasks
       where task_id = 2214;

  TASK_ID | STATE | START_TIME          | COMPLETION_TIME          |
---------|-------|---------------------|--------------------------|
    2214   | COMPLETED | 19-SEP-14 05.19.02.744300 PM | 19-SEP-14 05.25.33.355364 PM |
```
Example2: ADO policy to move tables

- Check the result

```sql
SQL> select table_name, tablespace_name from
       user_tables where table_name = 'CUSTOMER';

<table>
<thead>
<tr>
<th>TABLE_NAME</th>
<th>TABLESPACE_NAME</th>
</tr>
</thead>
<tbody>
<tr>
<td>CUSTOMER</td>
<td>T2DATA</td>
</tr>
</tbody>
</table>

@tablespace_pcts.sql

<table>
<thead>
<tr>
<th>TBS_NAME</th>
<th>PCT_FREE</th>
<th>PCT_USED</th>
</tr>
</thead>
<tbody>
<tr>
<td>T1DATA</td>
<td>15.38</td>
<td>84.62</td>
</tr>
<tr>
<td>T2DATA</td>
<td>85.62</td>
<td>14.38</td>
</tr>
</tbody>
</table>
```

Tablespace space usage: free space on T1DATA tablespace:

Before  | 8.38% free
---------|-------------

After   | 15.38% free
---------|-------------
Example 3: ADO policy to compress partitions

- Example 3: Compress partitions
  Check partitions and their write time

- Create policy

```sql
SQL> select object_name, subobject_name, segment_write_time
    2 from user_heat_map_segment where object_name = 'ORDER';

<table>
<thead>
<tr>
<th>OBJECT_NAME</th>
<th>SUBOBJECT_NAME</th>
<th>SEGMENT_WRITE_TIME</th>
</tr>
</thead>
<tbody>
<tr>
<td>ORDER</td>
<td>ORD1</td>
<td>15-SEP-14</td>
</tr>
<tr>
<td>ORDER</td>
<td>ORD2</td>
<td>15-SEP-14</td>
</tr>
<tr>
<td>ORDER</td>
<td>ORD3</td>
<td>15-SEP-14</td>
</tr>
<tr>
<td>ORDER</td>
<td>ORD4</td>
<td>15-SEP-14</td>
</tr>
<tr>
<td>ORDER</td>
<td>ORD5</td>
<td>15-SEP-14</td>
</tr>
<tr>
<td>ORDER</td>
<td>ORD6</td>
<td>15-SEP-14</td>
</tr>
</tbody>
</table>

SQL> select table_name, compression, compress_for
    2 from user_tables where table_name = 'ORDER';

<table>
<thead>
<tr>
<th>TABLE_NAME</th>
<th>COMPRESSION</th>
<th>COMPRESS_FOR</th>
</tr>
</thead>
<tbody>
<tr>
<td>ORDER</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

SQL> alter table ORDER modify partition ORD1
    1 ILM ADD POLICY compress_segment AFTER 10 DAYS OF NO MODIFICATION;
Table altered.
SQL> select policy_name, object_name from user_ilmobjects
    2 where object_name = 'ORDER';

<table>
<thead>
<tr>
<th>POLICY_NAME</th>
<th>OBJECT_NAME</th>
</tr>
</thead>
<tbody>
<tr>
<td>P142</td>
<td>ORDER</td>
</tr>
</tbody>
</table>
```
Example 3: ADO policy to compress partitions

- Check policy
  ```sql
  SQL> select POLICY_NAME, ACTION_TYPE, SCOPE, COMPRESSION_LEVEL, CONDITION_TYPE, CONDITION_DAYS from user_ilm_datamovementpolicies WHERE POLICY_NAME = 'P142';
  POLICY_NAME  ACTION_TYPE  SCOPE  COMPRESSION_LEVEL  CONDITION_TYPE  CONDITION_DAYS
  ---------  --------------  ------  ----------------  -----------------  -----------------------
  P142      COMPRESSION SEGMENT LAST MODIFICATION TIME 10
  ```

- Check execution
  ```sql
  SQL> select task_id, policy_name, object_name, selected_for_execution, job_name from user_ilm_evaluatiomdetails where policy_name = 'P142';
  TASK_ID  POLICY_NAME  OBJECT_NAME  SELECTED_FOR_EXECUTION  JOB_NAME
  --------  ----------  ----------  -----------------  --------
  2258      P142       ORDER       PRECONDITION NOT SATISFIED
  SQL> !date
  Sat Sep 20 21:46:38 CDT 2014
  ```

- Recreate Policy
  ```sql
  SQL> alter table ORDER MODIFY PARTITION ORD1
  ILM ADD POLICY compress segment AFTER 5 DAYS OF NO MODIFICATION;
  ```

  ```sql
  SQL> select policy_name, object_name from user_ilm_objects
  where object_name = 'ORDER';
  POLICY_NAME  OBJECT_NAME
  ----------  ---------
  P145        ORDER
  ```
Example3: ADO policy to compress partitions

• Check policy: **policy_name**: P145, **task_id**: 2358

```sql
SQL> select POLICY_NAME, ACTION_TYPE, SCOPE , COMPRESSION_LEVEL, CONDITION_TYPE, CONDITION_DAYS from user_ilmdatamovementpolicies WHERE POLICY_NAME = 'P145';

POLICY_NAME ACTION_TYPE SCOPE COMPRESSION_LEVEL CONDITION_TYPE CONDITION_DAYS
P145 COMPRESSION SEGMENT LAST MODIFICATION TIME 5
```

• Check execution of task

```sql
SQL> select task_id, policy_name, object_name, selected_for_execution ,job_name from user_ilmevaluationdetails where policy_name = 'P145';

TASK_ID POLICY_NAME OBJECT_NAME SELECTED_FOR_EXECUTION JOB_N
2358 P145 ORDER SELECTED FOR EXECUTION ILMJOB2472
```

• Check the result

```sql
SQL> select TASK_ID, STATE , START_TIME, COMPLETION_TIME 2 from user_ilmtasks where task_id = 2258;

TASK_ID STATE START_TIME  COMPLETION_TIME
2358 COMPLETED 21-SEP-14 02.57.41.890760 AM 21-SEP-14 02.57.43.564471 AM
```

```sql
SQL> select TABLE_NAME, PARTITION_NAME, COMPRESSION, COMPRESS_FOR 2 from user_tab_partitions where PARTITION_NAME = 'ORD1';

TABLE_NAME PARTITION_NAME COMPRESSION COMPRESS_FOR ORDER ORD1 ENABLED BASIC
```
Example 4: ADO policy to move partitions

- Example 4: Move partition to tier 2 storage T2DATA:

  - Current tablespace:

    ```sql
    SQL> select TABLE_NAME, PARTITION_NAME, TABLESPACE_NAME
           2 from user_TAB_PARTITIONS where PARTITION_NAME = 'ORD1';
    TABLE_NAME   PARTITION_NAME   TABLESPACE_NAME
    --------     --------------   --------------
    ORDER        ORD1             T1DATA
    ```

  - Create policy:

    ```sql
    SQL> alter table ORDER MODIFY PARTITION ORD2
    ILM ADD POLICY tier to T2DATA;
    Table altered.
    SQL> select POLICY_NAME, ACTION_TYPE, SCOPE, TIER_TABLESPACE
           from user_ilmdatamovemaint policies where object_name = 'ORDER';
    POLICY_NAME   ACTION_TYPE   SCOPE     TIER_TABLESPACE
    --------      ----------     -------   --------------
    P143          STORAGE      SEGMENT   T2DATA
    ```

  - Execute the policy:

    ```sql
    SQL> @execute_policy.sql ORDER
    old  5:          object_name => '1',
    new  5:          object_name => 'ORDER',
    PL/SQL procedure successfully completed.
    SQL> select task_id, policy_name, object_name, selected_for_execution, job_name
           from user_ilmevaluation_details where policy_name = 'P143';
    no rows selected
    ```
Example 4: ADO policy to move partitions

- Why? T1DATA’s PCT_USED (84.62%) < 85% threshold

<table>
<thead>
<tr>
<th>TBS_NAME</th>
<th>PCT_FREE</th>
<th>PCT_USED</th>
</tr>
</thead>
<tbody>
<tr>
<td>T1DATA</td>
<td>15.38</td>
<td>84.62</td>
</tr>
<tr>
<td>T2DATA</td>
<td>85.62</td>
<td>14.38</td>
</tr>
</tbody>
</table>

SQL> select * from dba_ILMparameters;

<table>
<thead>
<tr>
<th>NAME</th>
<th>VALUE</th>
</tr>
</thead>
<tbody>
<tr>
<td>ENABLED</td>
<td>1</td>
</tr>
<tr>
<td>JOB LIMIT</td>
<td>10</td>
</tr>
<tr>
<td>EXECUTION MODE</td>
<td>3</td>
</tr>
<tr>
<td>EXECUTION INTERVAL</td>
<td>15</td>
</tr>
<tr>
<td>TBS PERCENT USED</td>
<td>85</td>
</tr>
<tr>
<td>TBS PERCENT FREE</td>
<td>30</td>
</tr>
</tbody>
</table>

- Force the policy to get executed by changing T1DATa PCT_USED parameter

```
SQL> exec dbms_ILM_admin.customize_ILM(dbms_ILM_admin.tbs_percent_used,80)
```

- Execute the policy again:

```
SQL> @execute_policy.sql ORDER
SQL> declare
2 v_executionid number;
3 begin
4 dbms_ILM.execute_ILM (owner => 'HAMMERBASE',
5 object_name => '&1',
6 task_id => v_executionid);
7 end;
8 /
old 5:
new 5:
   object_name => '&1',
   object_name => 'ORDER',
PL/SQL procedure successfully completed.
```
Example 4: ADO policy to move partitions

• Check the task:

```
SQL> select task_id, policy_name, object_name, selected_for_execution, job_name
        from user_ilmevaluationdetails where policy_name = 'P143';
```

```
2264  P143 ORDER SELECTED FOR EXECUTION ILMJOB2396
```

```
SQL> select TASK_ID, STATE , START_TIME, COMPLETION_TIME
        2 from user_ilmtasks where task_id = 2264;
```

```
TASK_ID  STATE    START_TIME            COMPLETION_TIME
         COMPLETED 20-SEP-14 04.03.49.639380 AM  20-SEP-14 04.03.51.528645 AM
```

• Check the result

```
SQL> select TABLE_NAME, PARTITION_NAME, TABLESPACE_NAME
        2 from user_TAB_PARTITIONS where PARTITION_NAME = 'ORD1';
```

```
TABLE_NAME       PARTITION_NAME   TABLESPACE_NAME
ORDER            ORD1            T2DATA
```

• Tablespace before the data movement

```
SQL> select TABLE_NAME, PARTITION_NAME, TABLESPACE_NAME
        2 from user_TAB_PARTITIONS where PARTITION_NAME = 'ORD1';
```

```
TABLE_NAME       PARTITION_NAME   TABLESPACE_NAME
ORDER            ORD1            T1DATA
```
Thank You and QA

Contact me at kai_yu@dell.com or visit my Oracle Blog at http://kyuoracleblog.wordpress.com/