Oracle database 12c In-Memory option: Overview

Introduction

Oracle has introduced industry breakthrough enhancement in patch set 1 of existing database 12c, called In-Memory.

Traditionally, Oracle stores data in tables in form of rows. This new feature will allow to store data in memory in columnar format.

Storing data in columnar format give multiple times performance gain for DSS systems where you tend to retrieve fewer columns with multiple rows and this performance enhancement is further multiplied when you retrieve that data from memory instead of slower disks.

High level steps

It's one of the easiest features to implement. You can implement it in three steps.

1. Define parameter INMEMORY_SIZE in MB/GB.
   a) Edit init.ora file
   b) Include INMEMORY_SIZE=nnn.. GB
   c) Restart database
   d) Validate memory allocated to InMemory:
      Instance startup messages will show you additional line for In-Memory. If you miss that then run below command on sqlplus
      Show parameter inmemory;

2. Set tables attribute to InMemory.
   a) Alter table hr.people inmemory;

3. Access the table or restart database. Tables will be populated in In-Memory column store only when they will be accessed first time or database is restarted.
   a) Select count(*) from hr.people
   b) If you do not want to run count then restart of database will also populate table in In-Memory column store.
   c) Monitor the progress of InMemory population and then it is ready to fly.
      SELECT owner, segment_name, populate_status FROM v$im_segments;
      SELECT owner, segment_name name,populate_status ,bytes_not_populated FROM v$im_segments;

Example for inmemory option

1. Download scripts in zip file from article home page

   1.1 run create_people.sql to create table
   1.2 run dataload_0724.sql to load 1 mil rows
   1.3 If you want to add more rows you can run bulk0724.sql.You can edit this file for data volume you want to load
   1.4 At the end run CR_Index.sql to create index on table

2. Define parameter INMEMORY_SIZE in MB/GB in init.ora file and validate it after instance startup or during startup

3. Check In-Memory attribute of tables

   SQL> column TABLE_NAME format a40
   SQL> column CACHE format a5
   SQL> column INMEMORY_PRIORITY format a25
   SQL> column INMEMORY format a25
   SQL> set linesize120
   SQL> select TABLE_NAME, cache, INMEMORY_PRIORITY, INMEMORY from user_tables where table_name like 'PEO%';

```
TABLE_NAME                               CACHE INMEMORY_PRIORITY        INMEMORY
---------------------------------------- ----- ------------------------ -------------------------
PEOPLE                                   N                              DISABLED
PEOPLE2                                  N                              DISABLED
PEOPLE3                                  N                              DISABLED
```

4. Run a sample query without In-Memory

   SQL> select distinct count (last_name) from anuj.people2 where sal between 100000 and 100100 ;

```
COUNT(LAST_NAME)
----------------
               2
Execution Plan
----------------------------------------------------------
Plan hash value: 3468796632

-------------------------------------------------------------------------------
| Id  | Operation          | Name    | Rows  | Bytes | Cost (%CPU)| Time     |
-------------------------------------------------------------------------------
|   0 | SELECT STATEMENT   |         |     1 |    30 |   409   (0)| 00:00:01 |
|   1 |  SORT AGGREGATE    |         |     1 |    30 |            |          |
|*  2 |   TABLE ACCESS FULL| PEOPLE2 |     2 |    60 |   409   (0)| 00:00:01 |
-------------------------------------------------------------------------------
```

5. Change tables attribute to inmemory and validate

```
SQL> alter table anuj.people2 inmemory;

Table altered.

SQL> select TABLE_NAME,cache,INMEMORY_PRIORITY,INMEMORY from user_tables
where table_name like 'PEO%';

TABLE_NAME                     CACHE INMEMORY_PRIORITY       INMEMORY
------------------------------ ----- ------------------------ ------------------------
PEOPLE2                        N                              NONE                     ENABLED
PEOPLE                         N                                                       DISABLED
PEOPLE3                        N                                                       DISABLED
```

6. Run a sample query to populate table in In-memory column store

```
SQL> select /*+ full(ppl) noparallel (ppl) */ count(*) from anuj.people2 ppl;

  COUNT(*)
----------
    100000
```

7. Run sample query to see In-Memory operation

```
SQL> set autotrace on
SQL> select distinct count(last_name)  from anuj.people2 where sal between 100000
and 100100;

COUNT(LAST_NAME)
----------------
               2

Execution Plan
----------------------------------------------------------
Plan hash value: 3468796632

-------------------------------------------------------------------------------------------
| Id  | Operation                | Name    | Rows  | Bytes | Cost (%CPU)| Time     |
-------------------------------------------------------------------------------------------
|   0 | SELECT STATEMENT         |         |     1 |    30 |   409   (0)| 00:00:01 |
|   1 |  SORT AGGREGATE          |         |     1 |    30 |            |          |
|*  2 |   TABLE ACCESS INMEMORY FULL| PEOPLE2 |     2 |    60 |   409   (0)| 00:00:01 |
-------------------------------------------------------------------------------------------
```

8. Run below command to take table out of In-Memory column store.

```
SQL> set autotrace off
SQL> alter table anuj.people2 no inmemory;
```

9. Run sample query to validate no inmemory operation

```
 SQL> set autotrace on
 SQL> select distinct count(last_name)  from anuj.people2 where sal between 100000
and 100100 ;
```

```
COUNT(LAST_NAME)
----------------
               2

Execution Plan
----------------------------------------------------------
Plan hash value: 3468796632

-------------------------------------------------------------------------------
| Id  | Operation         | Name    | Rows  | Bytes | Cost (%CPU)| Time     |
-------------------------------------------------------------------------------
|   0 | SELECT STATEMENT   |         |     1 |    30 |   409   (0)| 00:00:01 |
|   1 |  SORT AGGREGATE    |         |     1 |    30 |            |          |
|*  2 |   TABLE ACCESS FULL| PEOPLE2 |     2 |    60 |   409   (0)| 00:00:01 |
-------------------------------------------------------------------------------
```

Other In-Memory Parameters:

Inmemory_query You can use this parameter to set in-memory attribute for queries on the tables at session or system level.

Inmemory_clause_default You can use this parameter to to define which table will go in In-memory. If you want to put all the tables to in-memory then set INMEORY_CLAUSE_DEFAULT to 'INMEMORY'. If you want certain tables to be in-memory then you can specify here and they will be populated with instance startup. Alternatively you can set this parameter a blank string that means tables will be populated in-memory by first setting in-memory attribute using alter table and then explicitly accessing it.

Inmemory_force You can set this parameter to either force all tables in in-memory or no tables in in-memory. Possible values for this parameter is "ON" and "OFF". If it is set to "ON" then you do not need to use alter command to change in-memory attribute.

Stay tuned for next article on In-Memory. Thanks for reading!