A BETTER MANAGEMENT OF DATA FOR FLEXIBLE ECONOMIC APPLICATIONS

Danut-Octavian SIMION, PhD Lecturer
„Athenaeum" University of Bucharest, Romania
E-mail: danut_so@yahoo.com

Abstract:

The paper presents the usage of data stored in different type of collections, like in memory databases. These types of databases offer faster responses, a better usage of memory and permits to the applications to be flexible and robust. For large volume of data, in memory databases permits fast responses in queries and for complex applications the aggregation of different information's required in reports and data analyses. A major benefit is that queries can be made to the necessary columns, analytical queries usually refers to a limited number of columns in a table. Using in Memory Database System has some advantage such as query only the necessary columns, sharing values intervals, compressing repetitive information, processing single instruction for a more values, conditions converted into filters, speed processing algorithms, and makes the applications more flexible and robust.

Keywords: *Framework, In Memory Database System, queries, open-source components, SQL, UML diagram, Web Applications.*

1. Introduction

Systems Database In-Memory are recommended when using complex applications, for which the large volumes of data in a relatively short period of time, the information will be aggregated by running reports of various types. IM purpose of the database is to meet the time requirements of the business, helping managers to make quick decisions in the context of any economic changes occur in the ear their field.

In theory, the choice of a particular system of in-memory databases should be made taking into account many factors, and not a decision that can only take one person. The manager should consult with IT specialists in order to find the best variant for their business purposes. The applications form in memory database already used by a particular company, whether

this company is unique or part of a group. Changing the core business application used is not very often, she often being required by the parent companies, for ease of data integration [2], [5].

2. The characteristics of data used in applications

A database column for each attribute of the transaction keeps a separate structure columns. This database is ideal for testing to quickly extract the necessary information that, when selected few columns, but poor when DML operations. If inserting or deleting a transaction, the entire structure columnar is changed. Until now, users were forced to choose between the format or the other, either send online processing efficiency, either analytical performance.
In Memory Database combines the best features of both formats, allowing information to be populated simultaneously in both types of memory. The new architecture dual-format memory not duplicates requirements. In-Memory format have adjusted the size of objects to be stored in the memory. The cache has been optimized to operate with a smaller capacity than the size of the database. Basically this dual-format architecture waits to increase by more than 20% memory requirements as a whole. It believes it is still a small price to performance that will be obtained [1], [4].

With this approach on the new type of memory, we found one copy of the table into memory, so no additional costs or timing problems. The database will still keep consistency transactions in respect of specific row or column format, as will retain the consistency between tables and indexes. The optimization process will direct the format column analytical applications and related operations to be processed transactions online to another format. This provides outstanding performance and data consistency for all workload without any modification of applications. In-Memory Database (IM) architecture uses the column. Aria IM is a static area in the SGA (System Global Area), whose size is controlled by parameter INMEMORY_SIZE. IM is the current size of the area visible in V $ SGA. Being a static area, any modification of this parameter will not take effect unless the court restarts. Nor it is controlled by AMM (Automatic Memory Management). The minimum size should be 100MB IM. IM area is divided into two areas:

- 1MB area used for storing the current column populated formatted data in memory;
- A 64K area used to store metadata about objects that are populated in IM database;

IM must be populated with data columns that are most important in terms of performance. If required lower performance, this translates into lower costs with components storage or flash disk. However, if a database is sufficiently low, all of its table's columns may be incorporated into the IM. This can be done by adding INMEMORY attribute tables and materialized view sites. However, if this attribute is available at 'tablespace', all new tables and materialized view sites will be available for storage in the columns default IM [3], [4].

Memory IM columns are populated by a set of processes running "in the background" (background), called "Worker Processes". The database is fully accessible during these processes are running. However, on the basis of pure IM date, it can't be accessed until populating all data in the memory, leading to significant availability problems. As the area called 'tablespace' disk is made of several extensions, IM and columns are made of several units IMCU (In-Memory Compression Units). Each "worker process' allocate its units and populates the corresponding data subset.

- Data is not sorted or ordered in a certain way while populating - is the order in which they appear in another format type.

- IM objects are populated in columns or in a prioritized list immediately after opening the database or after they are scanned for the first time. The order in which they are populated is controlled by key command PRIORITY.

Order prioritization has 7 levels, the default is NONE, and an object is populated only after it is scanned first. All items related to a certain level of prioritization must be populated before you start populating the objects with a lower level. Stocking order can be replaced if an object is scanned without priority, triggering its popular IM columns. Although almost all items are eligible to be populated in IM columns, there are a few restrictions:

- a user-owned objects that are stored in SYS 'SYSTEM or SYSAUX tablespace' sites;
- an indexed tables;
- a clustered tables ('clustered');
- a time LOB types ('large object bytes').

Also, smaller items are not populated 64KB memory, as they are considered lost in space unnecessarily. It is envisaged that the blocks IM space is partitioned into subspaces 1MB.

In general, compression is considered a mechanism to save space available. IM populated memory data are compressed using a new set of algorithms, which not only saves space but also bring improved performance. The mechanism allows queries to be executed directly on columns tablets. This means that all scanning and filtering operations will

be performed on smaller data volumes. Information is decompressed only when you need it in this way [2], [6].

- By default, the data is compressed using QUERY FOR HIGH option, targeting the best performing queries.
- Option FOR CAPACITY applies additional compression technique over QUERY FOR compression. This can have a significant impact on performance, as each entry must be decompressed before they can be applied to the WHERE clause.
- Option FOR CAPACITY apply a compression algorithm and sophisticated, but with some drawbacks to decompression.

Changing compression clause of a column with the ALTER TABLE command will determine restocking column with any other available data. Compression ratio between 2X and 20X may vary, depending on the option chosen, the type of date and contents of the table. Technical compression used may be different, even partitions in a single table or in different columns. For example, some columns can be optimized for faster scanning and others to save space. It can also be used to estimate a utility compression rate. The estimate is based on a sample of table data.

An analytical query usually refers only to some columns in a table. When data is populated in IM columns, they are automatically compressed, which means that data scanning and filtering is done on a smaller volume of data. In addition, database scanning IM is only necessary columns interrogation covered WHERE clause. By indexing IM is possible and the greatest reduction of data accessed. Indexing are automatically created and maintained in all columns IM base. The information is later found by the filter in SQL statements. History indexing procedure takes minimum and maximum values for each column in a memory IMCU (In-Memory Compression Units). Index column referenced in the WHERE clause is sought IMCU, being compared with the minimum and maximum values stored in the repository indexes. If it is found, scanning IMCU avoided that area.

For data that are not required to be scanned in columns IM procedure is used SIMD (Single Instruction Processing Multiple Data values) → instead of being evaluated data for each entry in the column, the procedure allows a data set to a column to be evaluated with a single CPU instruction. IM columnar storage format used was specially designed to maximize the number of entries that can be uploaded and evaluated in a single statement. The procedure SIMD lets you scan a billion times per second [1], [5].

For example, considering ORDERS_SALES table, we want to find total sales for the value cda_id 1000 column. The table was fully populated in a column IM. Polling begins scanning only column. The first 8 values are

loaded into the SIMD register and compared with the 1000, a single CPU instruction. The number of values to be coated varies depending on the type of data and the compressed memory used. Execution plan shows a new set of keys 'In Memory', indicating that the table displayed in the order line has been marked IN MEMORY. To confirm whether the column was used IM, use session-level statistics using performance view sites such as V $ MYSTAT V and V $ SESSTAT $ STATNAME. All statistics related to IM columns beginning with "IM". For example, "IM scan rows" → counting rows processed by scanning in-memory.

SQL statements linking multiple tables can be processed very efficiently IM architecture, using the advantage Bloom filters. A Bloom filter is a link between the tables converts a filter which can be applied as part of a larger scanning a table. It is effective to apply IM columnar format using SIMD processing. When two tables are linked, making it less usually is scanned and rows that match the WHERE clause are used to create an in-memory table stored in the PGA (Process Global Area). During creation table, a Bloom filter is also created. After applying the WHERE clause and the second table, the rows results will be compared with those before. If no match is found, then the rows will be downloaded.

Bloom filters are easy to find on the drawing: will appear in two places - the creation and application. Also, it can be created more Bloom filters with a single scan.

Example:

```
SELECT SUM(do.price * do.discount) income
FROM detailorders do, calendar c
WHERE do.data_cda = c.data
        AND do.discount BEETWEEN 9 AND 58
        AND c.data >= '01-JAN-2016';
```

The first step is actually running a full scan line 4 of table → Calendar. Bloom Filter (: BF0000) is created immediately after scanning is complete this table (line 3), and is applied as part of the scan IM DETAILORDERS table (lines 5 and 6).

Line 2 (HASH JOIN) occurs because it is possible that the filter applied to return a fake. Order confirms that all rows returned scan fit connecting condition.

The conditions used to create Bloom filter can be viewed with 'SYS_OP_BLOOM_FILTER'.

Analytical queries and filters require more than simple links. They require complex aggregations and summarizing. The process of optimization is divided into two phases.

Step 1:

1. Query tables needed to start scanning.
2. A new structure is created based on the results of these scans;
3. This structure is used to create an additional structure called In-Memory Accumulator. It is a multidimensional matrix created in PGA (Process Global Area) and to aggregate or group data when scanning table with aggregate data, rather than do later.
4. At the end of the first phase, temporary tables are created to keep the columns referenced in the SELECT statement.

Step 2:

5. The second part of the execution plan starts scanning table with aggregated data and application key vectors: for every entry in this table, corresponding connecting condition, the sales value will be added relating nearest cell In-Memory Accumulator. If the value already exists in that cell, the two values will be added together and the result is added to the cell.
6. Finally, the results table large temporary tables are added back (created as part of the scan size tables).

Columns can I improve the performance of all types of queries radical but very few databases are 'read only' - available only for reading. To be truly effective, columns IM should allow both massive data load and processing online transactions.

Bulk upload of data (bulk data) appear in data warehousing environments. Charging them is the following: the information entered is decomposed into smaller parts, each resulting part is converted to type once used, then built a columnar type structure to that date. These structures are used to form blocks of data and built-index keys. The new formatted blocks are written directly into the database [4], [6].

This method involves charging a COMMIT at the end; otherwise the data is not loaded. If something happens during the operation, the whole procedure will be canceled. Charging will be made in the above segment data blocks created with the index marking maximum number of data blocks used to date by an object or segment. Once charging is complete, the index is amended to include new blocks created. The new data will be visible to other SQL operations.

Before this step, IM columns have no reference whatsoever about any change in the database. Once surgery was completed, allowing referral differences IM columns of data, BYTES_NOT_POPULATED column in the view v $ IM_SEGMENTS. If the object has specified priority level, then the columns will be automatically populated in IM. Otherwise, next time

will be questioned, the process running in the background will trigger IM populate columns (the idea that there is free space in these columns).
It is recommended that large data tables to be partitioned. Null partition of the benefits is the ability to upload data quickly and with minimal impact on users. There is a natural move data when the partitions, but is updated a data dictionary.
Steps:

1- Creating an external file tables;
2- Using CTAS order to create a no partitioned tables, temporary;
3- Setting of the INMEMORY;
4- Populate the temporary table column architecture IM;
5- Changing table and changing a partition with temporary table;

Data modification operations (Data Manipulation Language DML operations) are executed in a row "buffer cache" in the same way they would be executed if there were no in-memory database. If the object that runs DML operations is populated in IM columns, the changes are reflected here as they arise. Memory "buffer cache" and the IM maintain consistency by using in-memory transactions Transaction Manager. All the logs are on the base table as before; no need any IM logging and columns.

For each IMCU (In-Memory Compression Units) columns IM, it is created and maintained a log of the transaction. When an order amending of a row in an object that is populated in IM columns then all entries that are marked as old IMCU and a copy of the new version is added to the log. IMCU original entries are not immediately replaced, to provide and maintain consistency in reading data compression. Any transaction executed on this object columns IM, started before the advent of DML changes should look at the original entries. Read columns IM consistency is managed using SCNs (System Change Numbers).
When a query is executed with a new SCN on an object, read all the entries for columns IMCU (In-Memory Compression Units), except old entries. They will be found either in the transaction log or in the table of "buffer cache".

The more entries there IMCU old (In-Memory Compression Units), the smoother will be scanned to. It is therefore recommended when restocking IMCU reach a threshold of aging. This threshold is determined by heuristic aging and takes into account the frequency with which it is accessed IMCU and the number of old ROWS of it. Obviously, more frequent restocking is needed for when IMCU is accessed frequently or has a high percentage of old times. Restocking is an operation executive processes running in the background. Information is available at all times and any changes that occur on occasions during restocking IMCU are automatically recorded [1], [5].

In addition to the standard algorithm restocking there and another algorithm that erases the old entries using a low-priority background process (IMCO In-Memory Coordinator). This process runs very often, minutes, and check if you have made any restocking. For example, attribute INMEMORY just amended clause priority for a new object. IMCO will also check if there are old entries in columns IM. If it finds something will trigger background processes to repopulate. Restocking frequency is limited by the parameter INMEMORY_TRICKLE_REPOPULATE_SERVERS_PERCENT. This parameter controls the maximum percentage of time that the background process may attend such restocking. If too many such processes are running is affected central processing unit (CPU).

Trying to maintain consistency in columns IM transactions varies according to: the rate of change, selected in-memory compression for a table, rows modified location, any type of operation that occurs on the database. Rows modified and are co-locate in the same block will incur a charge lower than the rows that are scattered randomly break. Examples of occasions co-locate in the same block are new rows inserted, the database that will group together or loaded data in a single operation. For tables that have a high rate of change is commended MEMCOMPRESS FOR DML procedure. Where possible it is recommended the use partitioning to locate the changes in a table. For example, partition can be used to locate information in a table by date, so most changes will be limited to information stored in the latest partition.

3. The usage of IM data in applications

Create a table or initialization attribute In Memory IN MEMORY:
a) Create a table in the In-Memory:

```
CREATE TABLE customers
(customer_name VARCHAR2(250),
fiscal_code  NUMBER,
country VARCHAR2(30)
)
INMEMORY;
```

b) Initialize IN MEMORY attribute to the desired tables from database classic:

ALTER TABLE customers INMEMRORY;

In Memory Populating effective zone occurs when the table is accessed first. As a result, initialization, you run a query on the table and then check the view v $ IM_SEGMENTS:

SELECT count(customer_name) FROM client GROUP BY client_id;

SELECT owner, segment, populate_status FROM v$im_segments;

All table columns or its partitions inherit the attributes In Memory.
After a few seconds of creating the table, will start a process that runs in the background, which will create a new record in a table Dictionary.

EXEC for('SELECT

ts#,file#,block#,obj#,dataobj#,ulevel,sublevel,ilevel,flags,best

sortcol,

tinsize,ctinsize,toutsize,cmpsize,uncmpsize,mtime,spare1,spar

e2,spare3,spare4 FROM compression$');

Verify this information refer to the table defined previously by running the command:

EXEC for ('SELECT object_name, object_type, owner from

dba_objects

WHERE data_object_id = 7548');

a) Removing a table in the IM is via the command:
ALTER TABLE customers NO INMMEMORY;

b) If it is desired that a particular column in a table to populate the in-memory, use the following command:

ALTER TABLE customers INMEMORY NO INMEMORY (customer_id);

In Memory Database System benefits:

- Query only the necessary columns → analytical queries usually refers to a limited number of columns in a table. IM databases minimizes labor and increase performance by accessing only those columns needed query and process them without having to uncompressed them;

- Sharing values intervals → Logic tables is to be divided into sections and minimum and maximum values of each column is maintained for each section of the table. This allows queries to skip those sections that contain values outside the range of the query necessary data;

- Compressing repetitive information → some columns have values that are repeated. For example, a column that stores each geographic region sales transactions will have the same information several times. In-Memory System efficiency is based on this information repetitive compression and query optimization processes so as to execute only once for each unique value in the column IM;

- Processing single instruction for a more values → microprocessors time SIMD processing multiple values in a single CPU cycle;

- Conditions converted into filters → due columnar format; IM links between tables are transformed into filters applied during data stairs;

- Speed processing algorithms used in aggregation IM systems allow high speed running analytical queries and reports that aggregate large volumes of data. Thus reach a rate of one billion transactions per second per CPU core. Analyzes that lasted hours or days before up to complete, now ends in a few seconds.

- Elimination of indexes → eliminating the need for indexes has eliminated the time spent optimization. Users are no longer limited to the number of queries you can launch.

- Isolation of analytical reports made to avoid overloading → OLTP processing analytical reports, by running them on different clusters.

- Data selection → IM system does not require that the entire database to be replicated in the IM. Users can choose only certain tables or partitions to be included in the IM. Not requiring outstanding performance data can be stored on disks, generating lower costs by storing them.

- no restrictions on database size → because queries can be run for any data, regardless of area or storage (memory, flash disk), IM system can be used with databases of any size.

- improving fault tolerance in the IM → data storage is replicated to multiple nodes of a cluster. If a node becomes inoperative, queries can use duplicate data from other nodes.

- improved performance, effortlessly updating databases → classical IM systems can be done without the user losing functionality or application to fear that there will be compatibility between the two databases. It is not necessary migration of applications, rewriting or redrawing.

- Easy deployment and management;

- Lower costs → possibility of reducing the storage space;

- Improved productivity because users do not have to wait long before running a report; database administrators spend less time on maintenance / optimization systems.

3. Conclusions

A good management of data implies the usage of flexible indicators that can be found in a different type of database such as In Database Memory System. This type of database makes a good usage of memory and offers faster responses for queries, but also it is possible to use SQL commands [1], [4]. In Memory Database System have benefits such as query only the necessary columns, sharing values intervals, compressing repetitive information, processing single instruction for a more values, speed processing algorithms used in aggregation IM systems allow high speed running analytical queries and reports that aggregate large volumes of data, elimination of indexes, isolation of analytical reports made to avoid overloading, data selection, no restrictions on database size, improved performance, effortlessly updating databases [2], [3]. Other advantages of using these types of data in enterprise applications are: easy deployment and management, lower costs and the possibility of reducing the storage space, but also improves productivity by reducing the running time of a report or query.

References

[1] Chris Evans, *"In-memory databases - what they do and the storage they need"*, Computer Weekly.com, 2015;
[2], *"In-Memory Databases: Do You Need the Speed?"* Information Week.com, 2014;

[3] Danut-Octavian Simion, *"Using Java in Business Applications"*, WSEAS Conferences in the Universitatea Politehnica, Bucharest, Romania, pp. 218–223, April 20-22, 2010, Conference/Session: ECC_COMPUTING, ISBN: 978-960-474-178-6, ISSN: 1790-5117;
[4] Joab Jackson, *"In-memory technologies move databases to real time"*, pcworld.com, 2014;
[5] Nikita Ivanov, *"In-Memory Database vs. In-Memory Data Grid: Revisited"*, GridGain.com, 2015;
[6] Mohit Kumar Gupta, Vishal Verma, Megha Singh Verma, *"In-Memory Database Systems - A Paradigm Shift"*, International Journal of Engineering Trends and Technology (IJETT) – Volume 6 Number 6- Dec 2013;
URI: https://inmemory.net/
URI: http:// nmemory.codeplex.com/

URI: http:// www.exasol.com/en/
URI: http://www.manageability.org/
URI: http:// www.mcobject.com/
URI: http:// www.memsql.com