Data Management for in-memory databases

DISSERTATION

Submitted in partial fulfillment of the requirements of the

MTech Software Engineering Degree programme

By

Nikitha Vuppalanchi
2015SP93008

Under the supervision of

Madan Mohan
Project Manager

Dissertation work carried out at

SAP Labs, Bangalore

BIRLA INSTITUTE OF TECHNOLOGY AND SCIENCE
PILANI (RAJASTHAN)

April, 2017

BIRLA INSTITUTE OF TECHNOLOGY AND SCIENCE, PILANI

CERTIFICATE

This is to certify that the Dissertation entitled "**Data Management for in-memory databases**" and submitted **Ms. Nikitha Vuppalanchi** ID. No. **2015SP93008** in partial fulfillment of the requirements of SESAP ZG629T Dissertation, embodies the work done by him/her under my supervision.

Nikitha 12/04/2017

Place: Bangalore

Date: April, 2016

(Signature of the Supervisor)

Madan Mohan

Project Manager

BIRLA INSTITUTE OF TECHNOLOGY AND SCIENCE, PILANI
I SEMESTER 2016-17
SESAP ZG629T DISSERTATION
Dissertation Outline

BITS ID No. 2015sp93024 **Name of Student: Nishant Kumar**

Name of Supervisor	:	Madan Mohan
Designation of Supervisor	:	Project Manager
Qualification and Experience:		MCA 18+ Years of Industry Experience
E- mail ID of Supervisor	:	madan.mohan@sap.com
Topic of Dissertation	:	Data Management for in-memory databases

Name of First Examiner: Prof. Ramakanth

Designation of First Examiner: _____

Qualification and Experience: _____

E- mail ID of First Examiner:

Name of Second Examiner: _____

Designation of Second Examiner: _____

Qualification and Experience: _____

E- mail ID of Second Examiner: _____

(Signature of Student)

Date: 12/04/2017

(Signature of Supervisor)

Date: ----------

BIRLA INSTITUTE OF TECHNOLOGY & SCIENCE, PILANI
Work Integrated Learning Programmes Division
I SEMESTER 2016-17

SESAP ZG629T DISSERTATION

(EC-2 Mid-Semester Progress Evaluation Sheet)

Scheduled Month April:

NAME OF THE STUDENT: **Nikitha Vuppalanchi**

ID NO. : **2015SP93008**

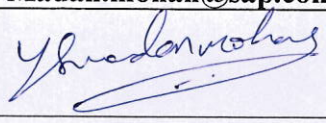
Email Address : **vnikitha@live.in**

NAME OF SUPERVISOR : **Madan Mohan**

PROJECT TITLE : **Data Management for in-memory databases**

EVALUATION DETAILS

EC No.	Component	Weightage	Comments (Technical Quality, Originality, Approach, Progress, Business value)	Marks Awarded
1	Dissertation Outline	10%		
2.	Mid-Sem Progress Seminar Viva Work Progress	 10% 5% 15%		

	Supervisor	Additional Examiner
Name	Madan Mohan	Prof. Ramakanth
Qualification	**MCA**	
Designation & Address	**Project Manager, SAP Labs Pvt. Ltd, Bangalore**	
Email Address	**Madan.mohan@sap.com**	
Signature		
Date		

BIRLA INSTITUTE OF TECHNOLOGY AND SCIENCE, PILANI
I SEMESTER 2016-17
SESAP ZG629T DISSERTATION

Supervisor's Evaluation Form

Supervisor's Rating of the Technical Quality of this Dissertation Outline

EXCELLENT / GOOD / FAIR/ POOR (Please specify): Excellent, keep up the spirit

Supervisor's suggestions and remarks about the outline (if applicable).

Date 12/04/2017

(Signature of Supervisor)

Name of the supervisor: Madan Mohan

Email Id of Supervisor: madan.mohan@sap.com

Mob # of supervisor: +91 9886306807

BIRLA INSTITUTE OF TECHNOLOGY & SCIENCE, PILANI
Work Integrated Learning Programmes Division
I SEMESTER 2015-16

SESAP ZG629T DISSERTATION

(Final Evaluation Sheet)
Scheduled Date April 15th 2017

NAME OF THE STUDENT: **Nikitha Vuppalanchi**

ID NO. : **2015SP93008**

Email Address : **vnikitha@live.in**

NAME OF THE SUPERVISOR: **Madan Mohan**

PROJECT TITLE : **Data Management for in-memory databases**

(Please put a tick (✔) mark in the appropriate box)

S.No.	Criteria	Excellent	Good	Fair	Poor
1	Work Progress and Achievements	✓			
2	Technical/Professional Competence	✓			
3	Documentation and expression	✓			
4	Initiative and originality	✓			
5	Punctuality	✓			
6	Reliability	✓			
	Recommended Final Grade	✓			

EVALUATION DETAILS

EC No.	Component	Weightage	Marks Awarded
1	Dissertation Outline	10%	
2	Mid-Sem Progress Seminar Viva Work Progress	 10% 5% 15%	
3	Final Seminar/Viva	20%	
4	Final Report	40%	
	Total out of	100%	

BIRLA INSTITUTE OF TECHNOLOGY & SCIENCE, PILANI
FIRST SEMESTER 2016-17

SESAP ZG629T DISSERTATION

Dissertation Title	:	Data Management for in-memory databases
Name of Supervisor	:	Madan Mohan
Name of Student	:	Nikitha Vuppalanchi
ID No. of Student	:	2015SP93008

Abstract

An in-memory database stores all of the data in the main memory or RAM. The data access is faster than other databases as the disk access is slower than memory access. The efficiency of in-memory databases is much higher when compared to normal database systems. In normal database the data is fetched to main memory and then the output is processed and hence there will be a seek time. As the data is stored in the memory for in-memory databases there will be no seek time to access the data. Implementing an in-memory database is costlier.

Data is growing exponentially. And keeping all the irrelevant data in the memory will be a costlier affair. So this data which is not potentially useful should be moved to a secondary storage. There are two reasons where we need to move our data from the in-memory storage to other storage.

1. When the data is business complete
2. Data privacy laws. Older data should be removed from the active database according to data privacy laws.

Understanding the rate at which each table grows and then which data to move and at what rate there should be aging runs scheduled. Managing all these processes is difficult. So with the help of this solution managing data aging process becomes easier.

(Signature of the Student)

Nikitha Vuppalanchi

2015SP93008

(Signature of the Supervisor)

Madan Mohan

Project Manager

Tables

Contents

Chapters

1 Introduction & Background

An in-memory database stores all of its data in the main memory. SAP HANA is one of the in-memory databases in today's market and also the best in-memory database till date as it provides additional features like data modeling, analytics etc.

SAP HANA combines an ACID-compliant database with high-speed analytics, application services, and flexible data acquisition in a single platform—without data duplication.

SAP HANA processes both transactions and analytics on a single data copy, in-memory delivering up-to-date insights. It eliminates indices, aggregate tables, materialized views, and data redundancy, to achieve dramatic savings with reduced data footprint, hardware, and operations.

SAP HANA is a SQL compliant database. SAP HANA enables a new class of business applications. Leverage in-memory, data processing capabilities – text, spatial, predictive, streaming, and graph – to build intelligent applications that provide deeper insight at unprecedented speed.

SAP HANA stores data in memory in columnar format, and uses advanced compression for 3-5x data footprint reduction. If the data size is more than memory size, SAP HANA automatically moves less frequently used data to disk – providing unlimited scale at the right price performance.

SAP HANA delivers enterprise-grade, high availability and disaster recovery capabilities to ensure business continuity. And because it runs on a variety of servers delivered by 12 partners and 1000+ configurations, you can start small and grow as you need without hardware lock-in.

Data is really growing faster. And keeping all the data in memory is not an efficient method as it reduces the processing speed. More data slow processing. And also we might need additional space and we will have to buy memory space instead if we will be able to manage the existing memory capacity well we can enable cost savings for the organization.

So the data that has to be moved from main memory to another database storage is

1. If the data is business complete. It means the data is no more useful for transactional purpose and might be used only for analytics or auditing purpose.
2. If the data is older. Older data should not be stored in main memory. The user might not require the data currently and keeping that sort of data in the main memory is not suggestable.

3 Objective of the project

Work out and establish a S/4-tailored concept for reducing HANA memory footprint as new data volume management capability that

- Makes sure only operationally relevant (hot/current) data is loaded into HANA main memory
- Keeps the other (cold/historical) data on "unlimited" (less expensive but usually slower) storage, stabilizing hot data performance while cold data remains accessible via SQL on request
- Has a lower TCD (easier implementation) and TCO (less admin effort) than archiving and current archive method
- Covers the top high volume aging objects

Scope:

- Enhance required HANA capabilities (like paging, partitioning)
- Develop ABAP Aging Framework
- Enable Cloud Qualities for Aging
- Cover top high volume objects from both Basis and Application by Aging

4 Uniqueness of the project

Basically only two temperatures, HOT and COLD, are relevant for Data Aging. The HOT zone comprises all of the data that is loaded into main memory, which is needed to process the data.4 The most essential difference between HOT and COLD is that the visibility of COLD data is not expected in operational business processes. Most queries must meet the expectation of the user to return only HOT data. Including COLD data in a result set is not impossible but should be the exception for certain scenarios. The WARM zone symbolizes (limited) storage optimizations within HANA that are transparent for application queries.5 The ARCHIVED zone is currently available in the form of ADK-Based Data Archiving and ILM Retention Management incl. immutable storage and destruction for legal reasons. It is supposed to coexist with Data Aging. The FROZEN zone is a future option for compliant archiving (structured read-only storage within HANA), which is currently out of scope.

For the HOT/COLD separation, we have evaluated different applications (see chapter 8) and agreed that the split between HOT and COLD data can typically be bound to the current archivability criteria. This is because the records selected must be aged together (as an object), and this data belongs to closed business processes if it passes the archivability check. Moreover, the access logic that "switches" to read archived data is already implemented in the Suite and can be adapted to SQL requests against COLD data. Even today, customers control the "archiving threshold" by setting the most appropriate residence time. For example, customers already archive CO line items after a few months if their data volume is growing too fast. However, the customer can then no longer work on these line items and, finally, CO documents containing archived lines items can no longer be cancelled. When data is aged instead of archived, easier access to COLD data can be implemented: It is now SQL instead of ABAP function module calls. Moreover, the aged data can be consumed by non ABAP (HANA based analytics) applications, provided that they explicitly request to cover COLD data as well. Finally, we have the potential to provide more aggressive aging compared with archiving. We could even go a little further by not disabling SQL UPDATEs to COLD data. Note that HANA supports updates anyway – also for COLD data as we are proposing it, and preventing updates is more effort than allowing them. This is because there is no way around (HANA-internal) updates in COLD when data is deleted (which is required by data privacy regardless of where the data resides). By preventing updates, we can neither implement a warm-up operation nor get rid of the current workarounds that have been developed for the exceptional updates to archived data when necessary from a scenario perspective. As in classical archiving, we only gain from moving data from HOT to COLD when we accept different qualities after the move. Besides visibility and lower access performance (when explicitly requested), it must be acceptable that no primary key uniqueness is enforced by the database – like applications are used to from an archive, see chapter 5.2.2 for details. To sum it up, the COLD zone consists of passive database content - in the sense that this data no longer belongs to operationally relevant business processes. Users accept that COLD data is hidden by default and that the system does not provide the same access quality as for HOT data - for the sake of a lower TCO and stable or even increased performance for HOT data.

5 Benefit to the organization

Implementing in-memory databases is really costlier. To enable cost savings we need to manage the data that exists in our database very well. And also keeping all the data in-memory makes it slower. More data less computation speed and throughput.

Data aging helps to manage this data efficiently by storing cold data in secondary storage. The retrieval of the data is also easier. Within no time we can get back our data on the secondary store to the main memory.

By enabling data aging this can be managed. Data aging is a huge process and managing this process is really difficult. The rate at which the tables grow and then when and how to schedule the aging runs and how to manage different data aging groups efficiently is a tedious job.

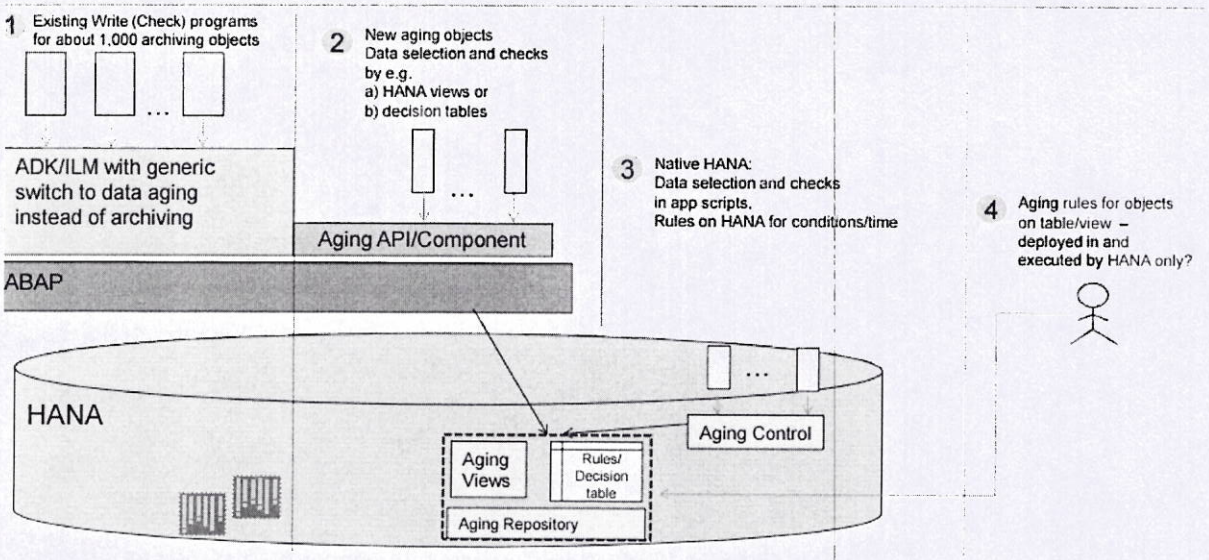
Hence this solution enables to manage the data aging groups easily and reduces the effort. Also the UI is really simple to use and provides a great experience to the user.

6 Scope of work

With the transactional app Manage Data Aging Groups, you can monitor the status of data aging runs that have been executed for your data aging groups. In the app you get detailed information about individual data aging runs and you have the option of analyzing error situations, such as failed runs. For example you can schedule new jobs to resolve error situations.

- Display a list of data aging groups with information about the status of the latest data aging run
- Sort, filter and group data aging groups by various criteria
- Display details about the recent runs for each data aging group
- Display a data aging log with lists of messages and processed data aging objects for each run
- Create, delete, reschedule and start jobs
- Share notes referring to data aging groups using SAP JAM groups to discuss issues and information with other SAP JAM group members
- You can stop a data aging run in process.
- You can create a data aging group and assign multiple data aging objects to the group. A data aging object can be assigned to only one group at a time.
- You can delete a data aging group. The data aging objects associated to this group are now available to be assigned to other data aging groups.
- You can edit a data aging group. You can change the description, add, or remove the associated data aging objects.

7 Solution architecture



1. Range partitioning with the option to mark certain partitions as COLD. Tables with cold partitions shall expose the following behavior (non-cold partitions are named "hot" here):
   - ❖ Unique constraints are enforced on hot partitions only, i.e. drop them on COLD partitions
   - ❖ DDL, i.e. metadata changes, operate on hot and cold partitions (note by TIP: except for unique constraints, and attribute-storage information, metadata should be the same for hot and cold)
   - ❖ Export of tables with cold partitions should consider hot and cold per default

2. Single-level Range partitioning is currently only allowed on key columns. Adding a dummy first-level partition is a workaround. However:
   - ❖ Tests have shown that updates on a range partition in the second level have poor performance. Update calls are ten times slower on such range partitions as compared to an update on a non-partitioned table. It has to be clarified whether the dummy first level influences the update performance.
   - ❖ More intuitive syntax would be nice.
3. Generic enhancement of SQL statements to restrict their scope to HOT (similar to adding the MANDT field in a WHERE clause).
   - ❖ However, it shall be possible to override the generic query pruning and to stick to the original behavior, that is, all data from the specified tables is selected, regardless in which partition (including COLD) it resides.
4. SQL Statement/connection context switch to "INCLUDING COLD"

- This is the current working assumption. Maybe there are strong cases for "ONLY COLD" as well.
- Not a HANA requirement but to be realized by NW layers (probably DBSL)

5. Generic append of new temperature column – optionally visible to applications.
   - ❖ Details to be clarified (e.g. only visible at DB layer, not in DDIC; which process is to issue an ALTER TABLE, datatype, ...)
   - ❖ Also it needs to be decided if this new column is to be appended to all tables generically or which selection mechanism is to be used in which install process (unfortunately, statically registered tables in the archiving framework are not the upper set of all tables that can be archived and could be aged. Also, Customizing tables can probably left out.)
6. It shall be possible to put old partitions on slow disks. Currently one Index Server process has one data volume on disk. It would be required to have multiple volumes per Index Server so that cold partitions may be moved to a second and typically slower disk.
   - ❖ The COLD partition is not expected to reside in main-memory, at least not completely in main-memory and for as short a time as possible (e.g. all attributes could be paged, appropriate caching and eviction is possible but not in the same quality as implemented for hot partitions that typically reside permanently in the main-memory)
   - ❖ This requirement is only considered as a "nice to have" since the aging concept can work without it to a large extent. The main objective is to occupy less main memory. This can be achieved with any kind of disk. The requirement adds value if customers want to store very large amounts of data and when the tradeoff between access times and disk costs is optimized.
7. Support for Warm-up-operation o Details to be clarified: Finding referenced records assumes either foreign key knowledge or the communication of the object construction by applications to the database. Otherwise only a warm-up by record can be supported by HANA.
8. If the logic of archivability checks is implemented on HANA level using views or procedures, an exit approach to bind customers' special procedures has to be provided by the HANA language. The current solution to modify the procedures is not sufficient, a solid extension concept is needed.
9. The proposed concept is not specific for tables in the column store. As SAP HANA currently does not support the partitioning for row store tables but such tables are subject to archiving, row store partitioning becomes a requirement, unless all aging-relevant row store tables are converted into columnar storage.

8 Resources needed for the project

Tools:

ABAP in Eclipse, SAP Netweaver, WebIDE.

Programming Languages:

ABAP, SAP Fiori

9 Project plan & Deliverables

#	Task	Expected date of completion
1.	Understanding the project requirements and specifications	Week 1, 2
2.	Learn about how in-memory Databases works and also come up with possible solutions for scheduling data aging runs and get to know about the status of the runs. Also learn about OData standard and exposing OData services through the SAP system and consuming them in the UI.	Week 3,4,5
3.	Setup the architecture and development framework	Week 6,7
4.	Implementation of the application	Week 8,9,10,11,12
5.	Testing and feedback	Week 13,14
6	Documentation	Week 15,16

Table 1 Project Plan

10 Work accomplished so far

10.1 Understanding the Requirements

Understand how data aging process works and which tasks of this process can be simplified.

Understand how HANA database works and how the aging process is carried out in HANA DB anf how that can be simplified.

How to classify data whether the data is hot or cold and how to identify the relation between different tables.

Understanding how tables are classified into objects and then how objects are classified to groups.

How to manage these groups and simplify this process.

10.2 Architecture and Design

Coming up with solution to break down this tedious task and simplify and develop services and consume them in the UI. Simple UI according to fiori standards and it simplifies the complex process

10.3 Building the API's

The API's are ready which will be called by the services.

11 Key challenges faced during the project

Data aging is a complex process and managing this process is a really tedious task and while trying to implement this we need to have a detailed understanding of this process.

And sometimes this could be different for different in-memory databases. Coming up with a generic solution which is applicable for all the in-memory database is a challenge.

12 Potential Risk and Mitigation Plan

As this is a complex process not everything can be simplified. While trying to make this complex process simpler we might have left out implementing few features if the complex process.

But I have tried to take up a task and fulfil it completely so no potential risks discovered till now.