FACULDADE DE ENGENHARIA DA UNIVERSIDADE DO PORTO

Study of SAP Hana in the in-memory context

Gabriel Braga de Medeiros Mota Borges

U.PORTO
FEUP FACULDADE DE ENGENHARIA UNIVERSIDADE DO PORTO

Mestrado Integrado em Engenharia Informática e Computação

Supervisor: Professor João José Pinto Ferreira

June 27, 2016

Study of SAP Hana in the in-memory context

Gabriel Braga de Medeiros Mota Borges

Mestrado Integrado em Engenharia Informática e Computação

June 27, 2016

Abstract

This research focus on the business value created by an in-memory database, such as *SAP* Hana, by comparing the execution times of transactional and analytical queries with persistent databases, such as *SAP* R/3.

Nowadays, systems architectures are getting more complex and database access times are increasing, through queries so complex, that can take hours to materialize. To answer to some of these problems, *SAP*, a German software company, released Hana in 2010, a software platform, based on a new database based 100% on memory, with a scalable architecture, allowing an improvement on access's performances up to 3600 times faster.

The motto on the design of this database was: "if it's known that this system will have 100% real-time for all queries, how will you design it?".This resulted in reduction of architecture and transaction complexity, leading to simplification of data models and applications, allowing on-the-fly aggregation and the elimination of updates to predefined aggregates.

Driven by this motivation, this research was developed to verify if business value creation with Hana is, in fact, possible. The chosen method was to compare how the same set of queries, both analytical and transactional, perform on the two systems with different data loads, through a comparison of execution times. The conclusion reached was that analytical queries performed, on average, 4000 times better on Hana and transactional queries performed between 100 to 1000 times better.

With the mobile trend hitting the most diverse company sectors, this means that ERPs can now be integrated with mobile solutions, allowing decisions to be taken outside of the office and can be made anywhere, anytime.

Speed alone wasn't enough to justify an investment in such an expensive technology. In the course of this dissertation, the author verified that this database can create, in fact, concrete business value and true competitive advantage to an organization.

This research, however, is not enough to give a full advice about adopting in-memory, since these tests aren't enough to fully benchmark a database. Nevertheless, this is a useful addition to the knowledge base about this new paradigm that is in-memory. Companies that only focus on the present needs of their businesses are likely to fail in the future. This research supports the idea that this kind of technology is so disruptive, that the faster companies adopt it, the more will be the advantage over their competition.

Resumo

O foco desta dissertação centra-se no valor para o negócio de uma base de dados em memória, como o *SAP* Hana, através da comparação dos tempos de execução de *queries* analíticas e transaccionais com bases de dados em memória persistente, como o *SAP* R/3.

Nos dias de hoje, a arquitectura dos sistemas são cada vez mais complexas ao mesmo tempo que o tempo de acesso à base de dados aumenta, chegando ao ponto destes acessos demorarem horas até obter resultados. Para responder a estes problemas, a *SAP*, uma companhia de software alemã, lançou em 2010 o Hana, uma plataforma de software, onde se inclui uma base de dados que executa 100% em memória RAM. Esta base de dados foi desenhada de modo ao sistema ser escalável, permitindo um aumento da performance de acesso milhares de vezes.

O Hana foi desenhado com a pressuposto que todas as *queries* conseguiriam ser executadas em *real-time*. Isto resultou numa redução da complexidade da arquitectura e das transacções, levando à simplificação dos modelos de dados e das aplicações, permitindo agregados *on-the-fly* a e eliminação de actualizações a agregados já predefinidos.

Motivado por estes factos, este trabalho foi desenvolvido com o intuito de verificar se a criação de valor para o negócio com o Hana é, de facto, possível. O método escolhido foi através da comparação dos tempos de execução do mesmo conjunto de *queries*, tanto analíticas como transaccionais, nos dois tipos de sistemas. A conclusão a que se chegou foi que as *queries* analíticas têm uma performance, em média, 4000 vezes maior no Hana e as *queries* transaccionais executam 100 a 1000 vezes mais rápido.

Com a tendência dos dispositivos móveis a atingir os mais diversos sectores do negócio, isto significa que agora os ERPs podem ser integrados com soluções móveis, permitindo que as decisões possam ser agora tomadas fora do escritório e que podem ser tomadas a qualquer hora e em qualquer lugar,

A velocidade do Hana sozinha não é suficiente para justificar um investimento numa tecnologia tão dispendiosa. No decorrer desta dissertação, o autor verificou que este tipo de base de dados pode criar, de fato, valor para o negócio de uma maneira concreta e criar vantagens competitivas para uma empresa.

Esta pesquisa sozinha, no entanto, não é suficiente para dar um parecer completo sobre este tipo de bases de dados, uma vez que estes testes não são suficientes para comparar totalmente bases de dados. Porém, esta é uma adição muito útil à base de conhecimento actual sobre este novo paradigma que é o *in-memory*. Um dos comportamentos que se verificou ao longo da história foram que as organizações que só se concentram nas necessidades actuais de seus negócios tendem a falhar no futuro. Esta tese apoia a ideia de que este tipo de tecnologia é tão disruptiva, que as empresas que a adoptarem mais rapidamente, serão aquelas que terão a maior vantagem sobre a sua competição.

Acknowledgements

These last five years were an unimaginable experience. Being from Açores, I arrived here knowing nothing and leave with a feeling of accomplishment, thanks to Universidade do Porto and Faculdade de Engenharia. My warmest thanks to this great school, and I hope you keep graduating great engineers and great persons.

I would like to thank my whole family, who supported this journey and helped me since day one. I would like address specially to my parents, *Mário* and *Bernardete*, who made this journey possible, to my uncles, who made this journey succeed and all my friends, who made this journey enjoyable. A special thanks to *Maria*, who helped to keep me focus during this work.

This investigation wasn't realizable without the help of *AMT Consulting*, specially the *AMT Labs* team. Their experience is invaluable and their peart personality made this whole research easier, where I address a special thanks to *Fábio Silva* and *Filipa Galiza*.

Last but not least, A big thanks to my FEUP's counselor, *João José Pinto Ferreira*, for all the support and guidance on the course of this dissertation.

Gabriel Borges

*"If there's one thing this last week has taught me,
it's better to have a gun and not need it than to need a gun and not have it."*

True Romance (1993)

Contents

List of Figures

LIST OF FIGURES

List of Tables

LIST OF TABLES

Abbreviations

ACID	Atomicity, Consistency, Isolation, Durability
BW	Business Warehouse
CPU	Central Processing Unit
CSV	Comma separated file
DB	Database
DBaaS	Database as a Service
DRAM	Dynamic RAM
DW	Data Warehouse
ERP	Enterprise Resource Planning
GDP	Gross Domestic Product
GUI	Graphical User Interface
HR	Human Resources
HTML	Hypertext Markup Language
IDE	Integrated Development Environment
IS	Information System
KPI	Key Performance Indicator
MDC	Multitenant Database container
MDX	Multidimensional Expressions
MVCC	Multiversion Concurrency Control
OLAP	Online Analytical Processing
OLTP	Online Transactional Processing
RAM	Random Access Memory
RDBMS	Relational Database Management System
SITC	Standard International Trade Classification
SMP	Symmetric Multiprocessing
SQL	Structured Query Language
TCO	Total Cost of Ownership
TSV	Tab separated file
VPN	Virtual Private Network

Chapter 1

Introduction

With the occurrence of the Digital Revolution on the middle of the XX century, we officially entered the Age of Information. With the possibility to computerize and analyze large amounts of data, a new world of possibilities arrived.

Wide, Vast and Fast information now is so intrinsic in a business's strategies that isn't even noticeable to the common person. Organizations use it to plan ahead, deliver faster and work on even more strict margins, allowing bigger profit margins on competitive markets. [BM13]

To collect and analyze this huge flow of information that we now have access, a set of technologies and systems were developed, such as ERPs - Enterprise Resource Planning, DW - Data Warehouses and RDBMS - Relational Database Systems, making possible to take more effective decisions. These systems are critical players on a wide range of fields inside an organization, from improving the management of several areas, from Finance and Marketing to HR (Human Resources) and operations management. [Moh15] Now, unproductive work can be avoided and huge, complex problems seem relatively easy compared to 20 years ago.

Although we can retrieve information from data since some time ago, this process suffered some changes through the years. Hardware suffered big changes in recent years, with RAM and CPU gaining a huge boost in capacity and processing speed. This exponential growth occurring in hardware is explained by Moore's observations, where he stated that the number of transistors on a integrated circuit would double every year.[1] [Sch97]

This evolution in hardware triggered a number of improvements in a wide range of technology components, including databases. With the retrieval of information being increasingly faster, it allowed to turn into reality things that weren't possible before, with a more in-depth analysis, using data that was useless until then to make predictions, design strategies and in the end, gain competitive advantage. Information Systems depend heavily on databases, and their capabilities are limited by how they can stretch and use its resources.

[1] The pace has, however, slowed down, and nowadays the number of transistors doubles every two and half years

In spite of this evolution, many companies feel insecure in moving towards new technologies for a many number of reasons. Their claims are that if they're actually meeting its customer needs at a given time, there's no reason for change. Although this is a good premise to work on, it's equally important to have an eye on the future too. This is explained by the Innovator's Dilemma, where it states that if organizations fail to adapt new technologies and only focus on the current needs, they will likely fail. It explains how disruptive changes on a product or service are the most likely to give competitive edge against rivals. [Chr97]

This was one of the motives why *SAP*, the german multinational company, decided to develop a concept that was thought to remain on paper for some time. Hana was born in 2010, but since then suffered a considerable amount of improvements, mainly to respond to client needs and to integrate with other SAP tools. This is a system that runs 100% in RAM, against the most common disk based databases operated by millions of companies through all the world. With Hana, supposedly it's possible to have "real" real-time analytics and on-the-fly aggregations, saving weeks in obtaining results and have a completely elastic approach to business's analysis. [Dah15]

This dissertation addresses this particular problem, analyzing in which degree Hana can leverage a business, what it's possible to dramatically improve and what it's possible to do now that wasn't possible before. Although each industry/company need a specific plan to address their specifications, the objective of this research is to simulate in magnitude and scale the database load of a real company, and analyze how analytic and transactional queries behave, to have a more concrete view on Hana's business value.

1.1 Environment

This dissertation was developed in a partnership with *AMT Consulting*, an information technologies consulting firm focused on human capital management. This research serves as basis for the shift that will occur in the following years, that it's possible to verify already in worldwide companies, such as Mercedes and John Deere, but in Portugal this fast growth isn't possible to verify yet.The interest is to have a more clear analysis in what SAP Hana can bring, in order to start implementing in-memory solutions in clients.

1.2 Motivation and Objectives

Decision management, with their critical character on businesses strategies and planning, has always been a topic of study on Software Engineering, since it is one of the factors that allows differentiation and improvement of through innovation and applications.

Hana will supposedly, have a big impact on how these decisions are made and strategies are designed, since it's being defined as the foundation for the next generation of application platforms, such as RDBMS, allowing to build large business applications. These applications will lead to three main improvements: flexible and efficient development, low TCO operations and robust and non-disruptive deployment . [SFGL13]

Most organizations already have their own data platforms and IS, with which they have built or adapted their companies since several years ago. A paradigm change, involving significant investments, in a environment where cost cutting is always a priority, usually is not well taken. But like the Innovator's Dilemma states, if you fail to adapt the new technologies and just focus on the present, your business will likely fail.

One of the objectives of this dissertation is to verify if Hana is one of these innovations which enterprises should adopt and if the in-memory technology works as well as *SAP* advertise. In order to verify this, a set of comparisons between execution times of analytical and transactional queries to an in-memory database, *SAP* Hana, and to a persistent memory, *SAP* R/3 database will be made. This will follow a different set of constraints to simulate in the better way possible a real environment, with real data. The queries will be designed in a way where wide set of parameters are verified, to quantify the average improvement on performance is similar to what *SAP* advertises.

In the end, the objective is to answer the investigation question and to have a more clear and independent analysis of the pros and cons of changing from conventional databases to the new Hana solution.

1.3 Dissertation's Structure

After this brief introduction. this dissertation will follow several topics about Hana and about the test suite that will be made in the following four chapters.

In chapter 2 the importance of decision making and the mobile trend for ERP will be approached, while in chapter 3 will revise the literature about in-memory platform and about *SAP* Hana, to have a more clear view on in-memory computing.

In chapter 4 the investigation question will be formulated, as well as a description of the evaluation model and the prediction of results. The data source, suitability and validity will also be explained.

Chapter 5 will explain the implementation and experimentation process in both systems and minutely describe the purpose of each query. discuss the suitability of the data chosen to test and the designed experiments. The results of this implementation will be analyzed and compared in chapter 6 and the investigation question will be answered. The general conclusions of this research is on the final chapter 7, were the limitations, recommendations to practitioners and the path for future work on this subject will be set.

Chapter 2

Decision Making on Business Environments

Decisions are critical processes on running and growing any medium or large business. They're made from the lowest rank of employers to the CEOs and CTOs. It's crucial that these decision makers have the most accurate information and the tools to make decisions on the most efficiently way possible. Delayed decisions, most times, lead to a reduction on the competitive edge of companies, reduction on satisfaction of customers, slowing down of processes, etc, so it's almost mandatory that these decisions are taken fast and correctly.

2.1 Delays in decision making and the mobile shift

According to Aberdeen studies, one of the most important pressures over business management is the delay on decision making due to the lack of timely information. An important factor for managers is to have access to information anywhere, on real-time and accurately. This allows a fast answer to adverse events and take advantages of sudden opportunities. For example, knowing the inventory distribution of a company on real-time allows to answer to shortages on a certain location quickly, reducing eventual losses. On the other hand, a fast growth on the information available to companies is taking place. The ERPs are not as capable to answer efficiently to this increasing amount of data, with more and more complex analysis taking place. A new way to answer to this meaningful data is imperative. [CP12]

One of the differences between the best and the worst performing companies is exactly the time to react and to make decisions. The ability to access data anywhere is a key factor, since employees and managers are often dislocated from the business unit that needs support. With the spread of mobile solutions in the world, it was just a matter of time before ERPs would go mobile too. The study from Aberdeen Group showed that it's three times more likely for a good performing company to implement a mobility strategy compared to under performing companies. These "best-in-class" companies are also 68% more likely to notify decision makers on real-time.

On present age quick decisions are necessary in order to not lose competitive advantages and to reduce losses. Aberdeen recommends companies, in order to increase their overall performance, to integrate a mobility strategy with the ERP. They also recommend to implement KPI (Key Performance Indicators) alerts to decision-makers as part of this strategy. [PC11, CP12]

The mobile shift can also help companies regarding its efficiency. Through modular applications suited for every kind of employees, it can work as data entry at the point of activity, meaning that inventory, sales, etc can be inserted in the ERP on real-time. Or, in case of technicians, can provide useful information on fixing components and can act as an instruction manual.

Improving internal processes of the company is not the only improvement of a mobile shift. It can in fact, improve the customer satisfaction, with 82% of customers claiming better than average satisfaction. [Cas14]

As the mobile shift happens, and decisions can be made anywhere and at anytime, the bottleneck will shift to the ERP. With the increase of data sources and quantity, the decisions won't be able to be taken in real-time, since the ERPs can't handle these amounts of data.

2.2 Time-critical decisions

The new mobile tendency brought new challenges as well. While on mobile phones, decision makers expect to take actions much faster, since they're not willing to wait for applications responses taking longer than any other regular applications.

Some years ago, 100 milliseconds of response time was the average time to users felt like they're using a real-time system. With this time increasing to one second, the flow of work can still stay uninterrupted, but if delays exceeds ten seconds, then users feel doing other tasks while are waiting for the current task to finish. Gradually, these response times have a tendency to became shorter, and nowadays it's already two seconds before the user loose focus on their current task.These delays have a tendency to reduce even more with time. [Nah04]

If users are only willing to wait relatively short time for a web-page to load, it's also expected that they're willing to wait a similar time for mobile apps. Users are not available to wait large amount of time for a response to an action in a small device like a smart-phone or a tablet. Besides the time users are willing to wait, decision delays can affect the competitive edge of businesses, as it was mentioned before.

In order to operate the mobile revolution on ERPs, the ERP's themselves have to suffer modifications to allow decision makers answer effectively to changing conditions and develop different strategies without having to wait for overnight jobs. These decisions are time-critical and deeply influence the future of a business.

2.3 Conclusion

In the last 6 years, a new technology on databases have been gaining relevance. Instead of having your data in disks, it's stored in-memory (RAM), presumably increasing the access speed. This evolution combined with mobile solutions should now effectively bring the real-time feeling to mobile answers and revolutionize how businesses are run. In the next chapter, it will be possible to see how this system was designed and created a new type of databases for large companies.

Chapter 3

In-memory Computing

The birth of in-memory computing

RDBMS have been around for more than twenty years and since then it has been a core component in enterprises and applications. These systems were developed aiming at aiding these companies in a wide range of fields, from finance and sales to order fulfillment and manufacturing. However, year by year, the analytics involved in each area became more and more complex and a separation between transactional and analytical data started to happening,in order to increase performance and flexibility. With these separations, the data warehouses started to appear, so these analysis's performance could be increased, at the cost of additional management of extracting and loading data to the DW, as well as managing redundancy problems and developing predefined aggregates. [Pla09]

In 2003 when the CPU clock speed was about 3GHz, it seemed that a change of paradigm in data storage still had a long way to go. Surprisingly, two things happened since then: a fast growth in main-memory and the emergence of multi-core CPUs. While main memory was rapidly adapted for caching, transactional databases were not suited for parallelism, and stayed on the servers used until then, SMP servers (symmetric multi processing). For example, R/3 ran update transaction on one single thread and relied on the fast communication between parallel database process on SMP servers. So, although some changes occurred on the hardware world, the separation of transactional and analytical systems stayed the same. [Pla09]

The first tests of RDBMS with in-memory technology, with a row storage organization didn't show advantages compared to the RDBMS's with the same amount of memory for caching. And from these tests, came the idea of using in-memory using a columnar organization and the kick-off for the future of in-memory database was given. This would allow to transactional and analytical capabilities coexist in the same system, eliminating the cumbersome processes of duplicating the data. [Pla09]

3.1 In-memory computing - How it works

Real-time analytics were always an important requisite in applications since the beginning. However, it's possible to observe an huge increment of data available for analysis. By 2020, it's expected to have 5247 GB of data per person, and with traditional RDBMS it's becoming more and more challenging to keep the real-time response times. [KVV13]

Other challenge that businesses face is the separation between analytical and transactional queries. With in-memory, it's possible to have both type of queries on the same system and have increased performance in response times. But firstly, we have to differentiate well what are analytical queries from transactional queries.

3.1.1 Types of Queries

One of the most important concepts used on IT are certainly OLAP - Online Analytical Processing and OLTP - Online Transactional Processing, the two types of queries used on a business's daily basis. OLAP queries are focused on analyzing data in a specific database. It's characterized by being complex queries, which use historical data and are used to make predictions and plan strategies. OLAP queries are often used on business intelligence applications, data writing and report and data mining processes [OLT]. They're used mainly to give insights and to help decisions from managers, like expected sales and deliveries reports, completion date of products, comparing demand and inventory, profitability and costs forecasts, key factors for strategic planning, among many other factors [FML$^+$].

On the other hand in OLTP transactional processes are prioritized. It's characterized by a lot of simple queries, and is mainly used to store input information by the user in the database [OLT]. In a sales process, orders and processes are modified, created, deleted and are the key for production planning and delivery, simple and direct queries, that were the base of disk and row oriented DB. [FML$^+$]

Category	OLTP	OLAP
Applications	Manage usual business, operational and management applications	Management Systems, Reporting and decision
Effectiveness	Number of transactions per second	Response time per query
Volume	Large number of short online transactions (INSERT, UPDATE, DELETE)	Low volume of complex analytical transactions (SELECT)
Space	Operational data stored, typically small	Large data sets of historical information
Data Source	Operational information of the application	Historical and archive data
Focus	Updating data	Retrieval of information
Users	Common staff	Managers and Executives

Table 3.1: OLAP vs OLTP [OLT]

Ideally, databases should be able to process both OLAP and OLTP queries with good performance, but that isn't what happens on most systems. As we have seen before, or these systems are disjoint, to disallow analytical queries from slowing down transactional queries, or otherwise one type of queries is prioritized. This is on the base of in-memory databases, and boosted by the columnar storage, allows a fast OLAP queries processing and the high performance in-memory row store designed to address the OLTP workload.

To achieve and rely on this mixed workload, it's applied an reduced locking with MVCC - Multiversion Concurrency Control and logging schemes with optimization in the two phase commit [SFL+12]. This will allow all reads from the DB to have a consistent snapshot of it, guaranteed by transaction tokens which contain all the information to construct a consistent view of a transaction. To optimize the two phase commit, they commit the disk log to the disk after the first commit phase and then the second commit is done asynchronously and logs for input and output are eliminated by skipping preparation of the commit log entries. Then they grouped all commits and prepare commits requests as much as possible. [PZ12].

3.1.2 Changes in Hardware

This revolution on database system is only possible because hardware evolved to a point where now is affordable to have enough DRAM - Dynamic RAM to house an entire database. In the early 2000's multi-core architecture were introduced, but by now it's possible to have 128 core in just one server and they can have up to 6TB of memory, all of this a relatively low prices (compared to the prices practiced a few years ago) . To have a more clear notion of transfer speed, nowadays it's possible to have maximum speed of transferring data between CPU and DRAM of 85GB per second. With this amount of speed, it was possible to change the way systems were organized, which led to a new world of possibilities, like in-memory databases. [PZ12].

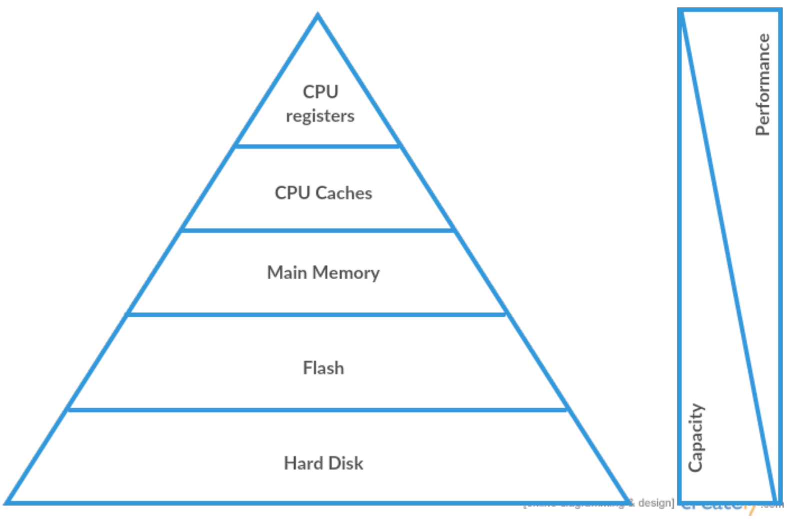


Figure 3.1: Capacity vs Performance - Hardware Memory [PZ12]

One of the bottlenecks was disk writings and readings, necessary to store data, since the evolution of CPU speed and DRAM capacity was not the same as disks. The low prices of memory now allow relying solely on it, while disks are still used, but only for persistence and backup. [PZ12].

3.1.3 Encoding

3.1.3.1 Dictionary Encoding

With disk accesses out of the equation, an increased performance working with data stored in the DB is experienced. Even so, memory access still is the bottleneck in a DB system. So the solution was to minimize data access time. This was possible through compression of data, allowing better transfer times and less memory consumption, by operating only with compressed data.

One of the solutions was to use dictionary encoding. This consists on compressing the tables, using a dictionary to store repeated fields of the DB. For every table, an attribute vector is created with a *ValueID* and this value is mapped to a specific entry, similar to a hash table. The encoding is applied to each column individually, so there's one dictionary for each column. This is particularly efficient for tables with many repeated values. The process of a search operation would occur like this:

1. Search in the dictionary for the requested value, and find the respective *ValueID*
2. Scan the attribute vector with the *ValueID*
3. In the search result, replace the *ValueID* with the corresponding dictionary value ...

Most of an organization's data have low entropy, so dictionary encoding can achieve significant reductions on a table size. After encoding the data, it's still possible to improve access times, by sorting the dictionary. Sorted dictionaries can have look up speeds of *O(log(n))*, with a binary search, compared to the *O(n)* of unsorted dictionaries. However, every time a new value is added, if the value is not already present in the dictionary, the dictionary has to be re-sorted and the attribute vector updated. [PZ12].

3.1.3.2 Compression

Even if memory capacity is growing through the years, is still expensive to process huge amounts of data in main memory. To decrease this cost, dictionary encoding is used, as well as compression techniques, to decrease the memory resources usage and improve performance of queries, reducing the amount of data between the CPU and main memory. The techniques used on enterprise applications have to be lightweight, otherwise encoding and decoding information would be extremely expensive. The most common compression techniques used on in-memory systems are:

- **Prefix Encoding** - Used when a specific column have long sequences of the same value
- **Run-length Encoding** - Used when there's low cardinality with large number of occurrences

- **Cluster Encoding** - The attribute vector is divided in several clusters of fixed size, and if the cluster only contains only one value, it's replaced by one occurrence of this value, otherwise it remains uncompressed.

- **Sparse Encoding** - Used when there are a lot of empty and non-existing values. Uses a bit vector, "1" for existing values and "0" for non-existing values

- **Indirect Encoding** - Similar to the Cluster Encoding and is used when there's a sorted table by a specific column and there's some sort of correlation between columns. Uses small dictionaries for each cluster.

Besides these techniques to reduce the overall size of the attribute vector, there are also compression techniques for the dictionary.**Delta Encoding** works better when the data is sorted and we encounter large number of values with the same prefixes. Nevertheless, compression techniques reach their full potential in sorted columns, and it's only possible sorting a table by one column. Some of them only allow indirect access, which has to be taken in account when we have specific response time requirements. [PZ12]

3.1.4 Data Layout

While relational tables are two dimensional, with in-memory all information is stored in one dimensional way. The DB storage layer has to decide in which way maps the two dimensional information to a one dimensional space, using Column, Row or a Hybrid layout.

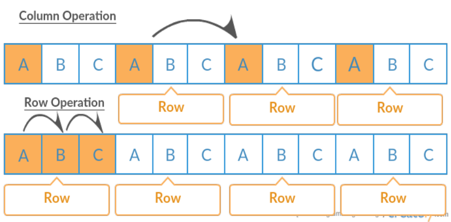
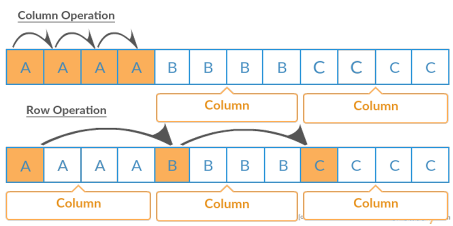
Row Data Layout	Column Data Layout
Data stored in tuples	Data stored attribute-wise
Low cost for reconstruction, but higher cost for scanning a single attribute	High cost for reconstruction
Lower compression techniques	Higher compression techniques
Fast scans of a single complete tuple or joint of complete tuples	Fast column scans or joint of columns
	

Table 3.2: Row VS Column Layout [PZ12]

However, enterprise workloads can be very different, and it's possible to benefit in having a row or a column layout. Hybrid layouts combine the best of both. Some attributes from a table can be on a row based layout, while others are in a column based layout. Nevertheless, finding an optimal layout is crucial for this kind of approach to work. [PZ12]

3.1.5 Parallelism

In order to speed up query processing, we need to parallelize our data and processes. Nowadays, parallelism is inherent in today's computer architecture and in the future it will only increase, mainly due to the growth of the number of CPU's cores in a system.

In-memory uses two kinds of parallelism: pipeline parallelism and data parallelism.

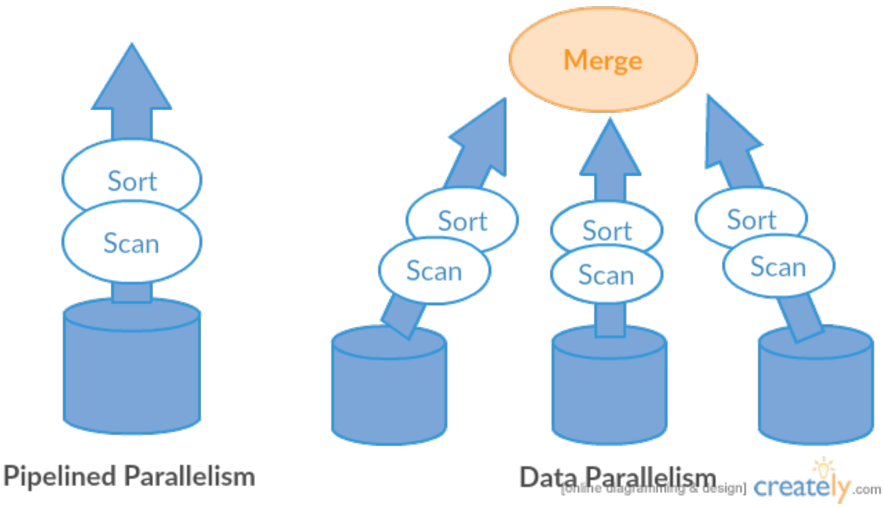


Figure 3.2: The two kinds of in-memory systems of data parallelism [PZ12]

In the pipelined parallelism, the following operators can start before the previous ones finished, even if just with a portion of the data, as it's possible to see in figure 3.2. With data parallelism, the partitioned data is processed individually and then merged in end. These are simple concepts, making use of the multiple CPU's servers, with multiple cores, to enhance even more the system. [PZ12]

3.1.6 Data Aging and Archiving

Not all data in your system is relevant. For example, invoices from 10 years ago have much less relevance than actual data. However, this data can still be relevant for analytical queries, so delete this kind of documents isn't the best option. To improve a system's performance, we can use data aging, where hot (relevant) data is treated differently from cold (irrelevant) data. The data schema is not changed, and the cold data is still accessible, being the opposite of archiving methods.

The solution is to pinpoint the hot data in the main memory, avoiding accessing cold partitions, to improve memory utilization and query performance. They trace and predict workloads and data usage, analyze and rank statistics, to derive rules for aging. [PZ12]

3.1.7 Database Operations

3.1.7.1 Insert, Delete and Update

Database operations, such as Insert, Delete and Update statements also have changed, due to columnar layout and the dictionary encoding. In this section, we will approach how each one of these operations work on in-memory systems.

- **Insert** - Inserts work differently if you a row or columnar organization. For row organizations, the tuple is appended at the end of the table, while in a columnar layout, a new entry in every column is needed. In in-memory systems like SAP Hana (which is mainly column oriented), where dictionary encoding is present, if the new database entry doesn't need a new dictionary value, then the *ValueID* is appended to the attribute vector. Otherwise, if you have to have to add a new entry in the dictionary, you have to resort it and reorganize the attribute vector, which is more expensive.

- **Update** - Update works the same as in relational databases. The problem is the cost to update a single value. The sequence of steps is: update the dictionary, reorganize the dictionary, reorganize the attribute vector and update all the old values in the attribute vector with the new ones. So to avoid this cumbersome process, *SAP* used in Hana an insert-only approach, which is covered ahead.

- **Delete** - Stops the validity of a given tuple. The nature of these deletes can be two: Physical, where a tuple is completely removed from the database. This can occur mainly due to legal reasons, where an organization has to delete a record permanently from its database.The other one is a logical delete, where the validity of a tuple is terminated. This is the most common type of deletes, when the data is not removed from the DB, but set a flag in the record indicating a tuple as non-valid.

All the data stored in databases have value for enterprises in building their models, so having the most historical data possible is a must. With the *Insert-Only* approach, there's no need for updates or deletes, instead you just invalidate tuples. With this method, it's possible to verify how certain data as changed with time, there's no necessity to clean dictionaries, at the cost of a raise in memory consumption.

The *Insert-Only* approach use depends in how your DB is used. It will lower your percentage of updates, since it will only invalidate an old tuple. Another benefit besides the maintenance of historical data. However, it's need to have in account the type of updates, if they are aggregate,status or value updates. [PZ12]

3.1.7.2 Select

For the retrieval of data, it's needed to realize the fundamental operation of Codds, in relational Algebra, in one table. [Voo] To retrieve the data, there are several ways to do it, with different performances. To have an effective plan, the most selective queries should come first, so the smallest set possible is obtained first. For example, if there's a column with low cardinality, this is the one that should be selected first. The look-up into the dictionary is only executed once, and all comparisons are based on the encoded values from the dictionary. As stated before, if the select query objective is to get complete tuples, than a row oriented schema is recommended, otherwise if you want to select several rows with the same attribute, column-oriented is recommended. [PZ12]

	Row Store	Column Store
Simple SELECT query	~46ms	~93ms
Complex SELECT query	1.257 seconds	~127ms

Table 3.3: Row vs Column Store access times [Mun15]

By analyzing the table 3.3, it's possible to conclude that row or column organization should be used in different situations. If aggregation and analytical queries are the focus, we should opt for a column organization, otherwise if the focus is simple and transactional queries, a row organization is recommended. This is a crucial factor in order to improve the overall performance of your platform. [Mun15]

3.1.8 Workload Management and Scheduling

With millions of queries being made to a single system everyday, some are more time sensitive than others. For example, transactional applications have critical business processes interactions with customers, and it's needed to give a fast response. On the other hand, OLAP queries normally are not that time sensitive, and a fast response is not always needed.

If there's no prioritization, the system will delivery slower responses in general. So, to solve this, the queries have to be scheduled and assign specific database resources, to meet the service level objective. This scheduling has to be dynamic, simply because it's difficult to predict and to know which will be a system's workload.

In spite scheduling being a straightforward concept, applied in many areas of business, in the in-memory system it can be expensive, in the presence of many queries, especially because of the dynamic feature. Heuristic strategies can play a big role in managing this problem. The main objectives are to guarantee that all transactional queries are committed, maximize the number of queries processed and to minimize the response time of analytical queries, without compromising the transactional queries. [PZ12]

One thing important to have in account in analytical queries, are once they enter the database, it's impossible to control them anymore, to stop or pause. This could halt an entire system during peak hours and nothing would solve it.

A hypothesis to ensure the system's stability and the delivery of the most important queries first, is to assign as many resources to OLTP queries, to ensure throughput goals. This is done because predict long queries like OLAP queries implies a high overhead. After the OLTP queries have enough resources, the remaining resources are allocated to analytical queries, in the most efficient way, always adjusting the concurrency and parallelism of depending on the transactional load. [PZ12]

3.1.9 In-memory capabilities

Most important than the evolution of the hardware through the years, is how it's used. If a system has more memory, it will be faster than before, but if data is treated before being stored, it will leverage this hardware improvement, through compression techniques, with a reorganized, simpler architecture, while guaranteeing 100% ACID (Atomicity, Consistency, Isolation, Durability) properties. To compare the performance of in-memory systems with persistent memory ones, the newest system from *SAP*, Hana, will be used to compare with R/3, their persistent based database and most widely sold system. Hana's particularities and it's rival products on the market will be approached in the next section.

To sum it up, there are five main characteristics to sum the in-memory technology: [vBDMR14]

- Data stored in memory – decreased access time
- Multiple CPU can process multiple requests at the same time, taking full advantage of computer resources – parallelism.
- Handling mixed workloads, OLAP and OLTP, on the same platform
- High compression techniques, through the column oriented organization
- Insert-only approach

3.2 Commercial Appliances

In the last five years, the first commercial appliances of in-memory started to emerge in the market. Th biggest two database companies (SAP and Oracle), as soon technology allowed, initiated the development of in-memory systems to be used on a large scale.

The analysis of this dissertation will be focused on SAP Hana, since *AMT Consulting* works mainly with *SAP* software. Furthermore, they have already a *SAP* R/3 server, which would be the control group for the tests.

Adding to that, obtaining a license to work with these commercial appliances is only at the reach of big companies. For these reasons, it was chosen to approach only the other in-memory solutions in the market instead of deepen the knowledge base about these other systems.

3.2.1 Hana

SAP, as a tech company, knew that they had to keep up with new technologies. They knew the in-memory concept since it was born, but until now, the concept was unfeasible. With the new advances on hardware, more specifically DRAM, the in-memory could finally pass from paper to a real system. So in 2010 Hana was born, but since then it has suffered a lot of modifications and updates, and it will continually to suffer, as it's usual on systems which adopt new technologies.

Hana incorporates the benefits from the in-memory technology, with on-the-fly data modeling, providing non-materialized views on top of actual and detailed information, without having to wait for model changes and database operations, thereby reducing the total cost of ownership. [R14]

3.2.1.1 Architecture

By adopting a columnar organization, it's possible to eliminate the index structures necessary in the row organization, reducing this way the system's overhead, complexity and eliminating the effort of maintaining the metadata. OLAP queries are affected by the row organization, since the same field is distributed across the memory, resulting in cache misses, slowing down the CPU.

The main management component of Hana is the index server, which contains the data stores and the processing engines, such as SQL and MDX (Multidimensional Expressions) statements. It also uses the preprocessor server for extracting information from text analysis. The name server knows the topology of the Hana system, which components are running and which data is located in which server. SQLScript embeds data-intensive application logic into the database, through parallelization of SQL queries through multiple processors. The database persistence layer is responsible for the durability and atomicity of transactions, by ensuring the restore of the database in case of failure. [R14] The overview of the architecture can be seen in figure 3.3.

The high availability is maintained by eliminating single points of failure and through fast recovery in case of system failure. It's achieved mainly through redundancy, in hardware, network and data center. A system like Hana is supposed to maintain the reliability in case of failure and recovering as quickly as possible. To recover the systems, regular backups are made and a continuous replication is used.

With Hana, it's possible to have all in the same system, without depending on Data Warehouses. From the customer & supplier relationship, Financial management, supply chain and project management to Human Resources.

To sum it up, Hana can empower companies to be even more competitive in the market. With Hana, it's possible:

- Greatly increase computation power
- Faster aggregation, using the column store.
- Analysis of large data sets and complex computation
- Flexible modeling and the absence of data duplication
- Fast data loads

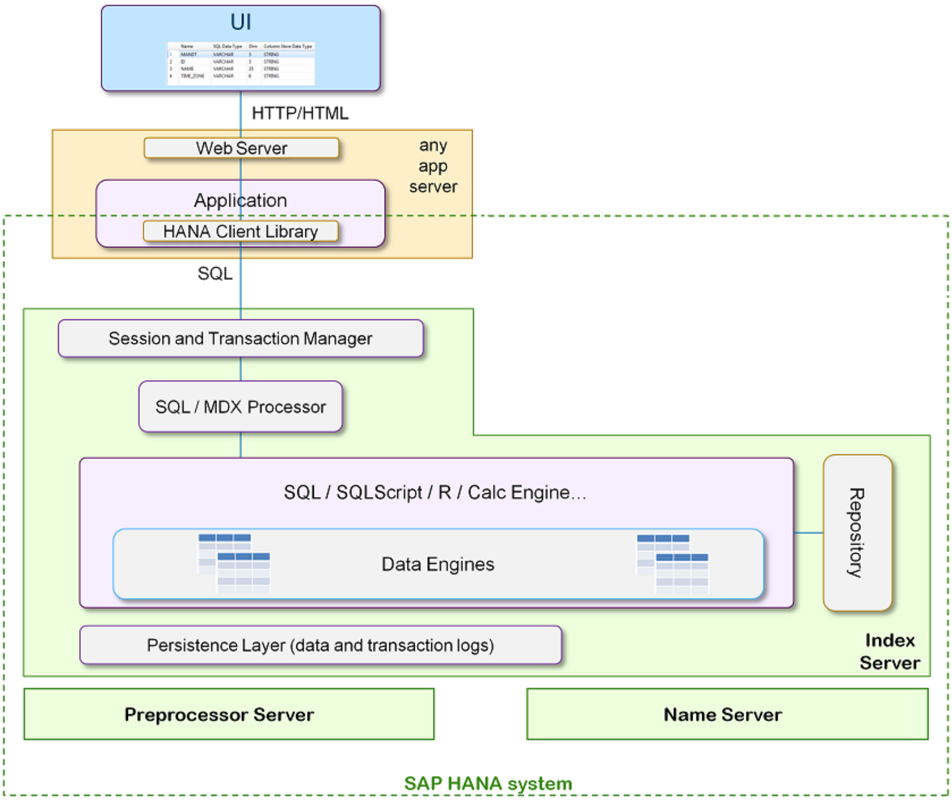


Figure 3.3: SAP HANA Database High-Level Architecture [R14]

3.2.1.2 Real-time Analytics

SAP's big slogan around Hana is real-time analytics. Real-time, as it's possible to see in many definitions across the internet, e.g *"of or relating to applications in which the computer must respond as rapidly as required by the user or necessitated by the process being controlled." [Dic].* During the launch of Hana, Hasso Plattner, one of the main founders of SAP, said *"Speed is important because of mobile – we want information but we don't want to wait for more than 3 seconds – nobody starts an app on their phone and waits 30 secs to load".* [Sul13]

This real-time speed will allow vast improvements on many areas, bu there are three categories which Hana specializes:

- **Operational Reporting** - Transactional systems with SAP Business suite or other SAP ERP, enhancing: [Hen]
  - Sales reporting
  - Financial reporting
  - Shipping reporting
  - Purchasing reporting
  - Master data reporting

- **Data Warehousing** - Run BW (Business Warehouse) applications in Hana, with increased performance, efficiency, in order to make faster decisions. The migration of DW (Data Warehouse) to Hana is simple, because the DW layer and the Hana layer are separated [Dah15]

- **Text Analysis on Big Data** - Insights on customer, employees and suppliers to anticipate behaviors and take anticipated measures, it's possible to use large amounts of data to make predictions on real-time, even using unstructured data [Dah15]

3.2.2 SAP R/3

SAP R/3 is based on a three tier client-server model. It was developed using the ABAP/4 programming language and aimed to integrate business processes in one system. It was first released in 1992 but in the following years a lot of releases were launched in order to improve the system.

The three layer architecture is divided in presentation server, application server and the RDBMS. The presentation server is the GUI of the system, while the application server interprets the developed ABAP/4 programs. The database server stores all the data, including the programs.

In *SAP* R/3 there's only one database, and all components are attached to it. It was designed to be a scalable, robust and portable system. Due to this possibilities back then, many companies all over the world adapted R/3 as their main system.

Through the time however, several issues were identified in R/3. It was designed with a rather complicated architecture. Since the technology is changing so fast, coding in the system became cumbersome and complex. Another issue was the nonexistent support for other languages being limited (Java was introduced afterwards), such as C/C++. [Sap, Kos, Soh]

3.2.3 Other In-memory appliances

In-memory is not a new concept. As it was stated before, the concept evolved faster than the hardware, and only nowadays it was possible to have full in-memory databases.

However, since 1995 that in-memory databases exist. Not in a full scale like Hana, but as a hybrid layout. *WebDNA* was the first of its kind, and was designed as a database driven dynamic web page applications. It was originally used to migrate traditional catalogs to online catalogs. [Rev]

The range of products go from Open Source companies to big multinationals like *SAP* or *Oracle*, from relational SQL to NoSQL databases, but every single system goal is to deliver real-time responses.

For example, *Aerospike NoSQL* was launched in 2012. Instead of being directed to big appliances, like Hana, it focused on web and mobile applications combined with big data, but also in simplifying the development of applications. It contains a database distributed server with failover and uses DRAM or SSD's (Solid State Drivers), a Smart Clients, which remaps requests in case of a change in the number of nodes and a management console.[Mul15a] Besides *Aerospike NoSQL*, there are other *NoSQL* systems, such as *Apache Cassandra*, *MarkLogic Server* and *MongoDB NoSQL*, but they won't be approached in this dissertation, since its focus are relational tables and databases. There are also *No SQL DBaaS (Database as a Service)* such as *Amazon Simple DB* which allows to store and store and query structured data through web services requests. However, it only performs well for simple storage and retrieval, data comparison and aggregation are not suited for this kind of database. [Mul15b]

In the relational field, like Hana, there's also some other solutions in the market, from big companies like IBM, Oracle and Microsoft. *IBM DB2* was released a little after Hana, in 2013, as a relational DBMS with integrated support for some *NoSQL* capabilities. Its more interesting particularity is the ability to skip data which isn't necessary for a query, through metadata records, to determine if a particular set of data has interest. [Mul15c]

Microsoft tried also to enter in the in-memory market, but has only accomplished partly. *Microsoft's SQL Server* only has in-memory optimized for OLTP, which is very limited, since the applications running on it have to be transaction oriented. [Fra15]

The in-memory which looks more like Hana is *Oracle Database 12c*, by running both OLTP and OLAP queries on the same database. Their benefits are very similar to each other, but Oracle offers plans with unlimited processing power and unlimited memory,if customers are willing to pay. There's claims from both SAP and Oracle for which database is better.

3.3 Value proposition of SAP Hana for enterprises

Like any big investment in new technology and systems for companies, it's never an easy decision to make. There are several things to take in account, projections, if there's value being created, etc. Adding to that, companies always favor small changes over big, radical changes, but as it was mentioned, the lack of anticipation will likely end in failure in the future.

Information Technology can bias the organizational performance in several areas, from productivity enhancements, profitability improvements, cost reductions, gain competitive advantage or have a better inventory management [vBDMR14].

As it's possible to see in figure 3.4, what Hana does is leverage the in-memory technologies, such as DRAM, multi-core architecture, columnar organization and compression techniques through processes. This processes allows OLAP and OLTP data in the same system, large volume of database processing and the reduction of the latency times, as well as graph and text processing. With this capabilities, it's possible to create business value, and verify the increments in both productivity and profitability and reduce costs. This is what can truly lead to competitive advantage.

This analysis has, however, some limitations. The comparison is made only with another SAP software, *R/3*, and not other appliances, such as Oracle databases. However, performance comparisons versus other disk-based databases in the market do not fit in the scope of this dissertation, since the author didn't had access to them. The performance evaluation is also limited by the data used. The data used is depth oriented, since the tables have a low number of columns and a high number of records. The performance evaluation is also influenced by the number of records.

In the next chapter, a conceptualization of the investigation approach and the development of a model to analyze and compare the data, as well answering the dissertation question.

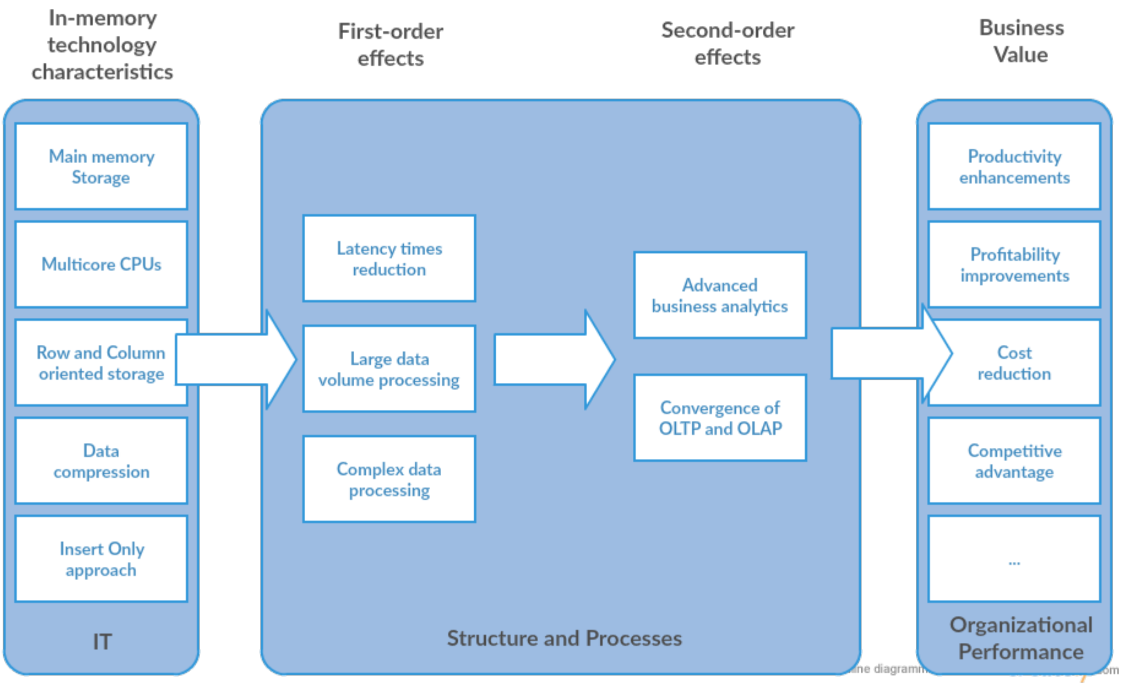


Figure 3.4: Business Value creation through Hana [vBDMR14]

Chapter 4

Approach to Research

The newer and emerging technologies are one of the important factors to determine the current and future strategies of a company. The most advanced IS and Databases, like Hana in-memory system, allow enterprises to engage and develop new business strategies and processes. [Hev04]

In this chapter the research methodology used in this dissertation is presented, as well the design of the research question. A model to evaluate and compare Hana with R/3 will also be presented.

4.1 Research Question

Changing to an in-memory solution raises concerns to enterprises, since it's a substantial investment. Although this evaluation's objective is to augment the knowledge base on this subject, it aims to help enterprises by recommending when, if and why change to this kind of solution and how it can create business value.

"Since when opting for an in-memory solution, such as Hana, can create business value?"

To answer the dissertation question, this investigation alone isn't sufficient to give a full answer, but it is useful to hereafter constitute a meaningful addition to the knowledge base that exist right now. The purpose is to know when is valuable for a company to operate the switch and how much better is Hana compared to most appliances in the market right now.

The prospect of this analysis is to confirm what *SAP* claims about Hana and in-memory systems and recommend to enterprises if and when they should start to implement in-memory. The value proposition is justified by how faster is Hana than compared to their older solution.

4.2 Research Approach

The end of Design Science is to create and evaluate IT artifacts, intended to solve organizational problems through innovations, allowing the analysis, design, implementation and use of IS to be accomplished. It's also possible to create and evaluate these IT artifacts and to solve organizational problems. This will be accomplished through analytical simulation and quantitative comparisons of queries to both databases, persistent and in-memory databases. [Hev04] With these tests, the assessment of the utility of Hana and in-memory to businesses can be accomplished.

Designing useful artifacts can sometimes be complex, due to the need of creative advances where theory shows to be insufficient.Theses artifacts aim to extend the capabilities of problem solving and organizational capabilities. [Hev04] Using clever design processes, it's attainable to produce innovative products. In this case, the purpose of the design process is to evaluate Hana as an innovative product in a quantitative manner. This means that, although it's known that Hana is better, it's important to know in a quantitative way and how innovative it is and since when, from a technical point of view, Hana can justify an investment for a business.

The business's needs can be framed by the goals, problems and opportunities perceived by the people within the organization. These needs are evaluated in the context of strategies, structure and existing processes, positioned with their current technologies and architectures. [Hev04]

The experiment to be conducted is a white box structural test, with the objective of testing a metric, the execution times of analytical queries. Most claims about IT artifacts are dependent on performance metrics, to determine how well an artifact works. In this problem, the value proposition of *SAP* Hana, is a rather complex problem and too big for a single person to answer, so there's a need to sub-divide in smaller problems and add all the conclusion to the knowledge base [Hev04]. So the objective is to extend the current knowledge base with these evaluation results, to allow conclusions became more accurate.

The model that we expect to reach is similar to claims byx *SAP* , which is the execution times being 3600 faster than a persistent database, for analytical queries [SAP11]. The expected results is something like figure 4.1, with both execution times increasing with the increment of the data load, but Hana scaling better than R/3.

Figure 4.1: Expected results for execution times

An effective problem representation is essential to find an effective solution. In the next section, the problem, solution and tools will be framed in order to allow an answer to the investigation question.

4.2.1 Information Systems Research Framework

To define the design for this dissertation, it was decided to use the IS Research Framework (figure 4.2), which can explain in a simple manner the research that was conducted. The Environment in which this research is based are organizations that have non optimized processes on their systems, such as complex architectures and data that remains untouched and there's no use for it. Usually, these companies have no competitive edge and are under performing on a multitude of fields, regarding their databases and ERP implementations. They use outdated technologies and systems, not keeping up with the technological development and the mobile migration. It's clear that they need to update to a more efficient technology, but since such change is actually expensive and linger, an IS research is needed to evaluate if Hana is a worthy for them and if their needs complements with what Hana as to offer.

To develop this research we need a strong knowledge base. It's important to know why some companies are under performing due to their ERP implementations, how in-memory and persistent memory systems work and how this migration can create business value now and in the future. Since this dissertation will mainly focus in the differences between analytical and transactional queries on R/3 databases and Hana databases. Analyze the full spectrum of benefits of a Hana solution does not fit on the scope of this dissertation, but instead the methodology will focus on analyzing on a quantitative manner, how much an in-memory database like Hana will perform better than the R/3 database. The database was replicated in size and the test set will involve several number of queries, and real data was used, although the data differ in quality relatively to the purpose of the some of organizations.

After this research is conducted, it's expected to justify when change of paradigm in databases and ERPS is recommendable. As stated before, this research will consist in analyzing the execution times of several queries, involving all kinds of query's predicates and statements. Then it will be possible to have a quantitative evaluation comparing both SAP databases and give a more accurate response to the investigation question. [Hev04]

4.3 Data

Usually, companies that would adopt a solution like Hana, have huge amounts of data dispersed across a high number of tables. Since this data is very specific and vary from company to company, it's difficult to predict how the system will behave. To try to estimate how OLTP and OLAP queries will perform, the tests should be on data that is similar in size and in type.

However, gaining access to the kind of data enterprises use is a difficult process. These companies don't allow external access to their data unless under strict confidentiality agreements, so competition don't gain any kind of advantage. Without the possibility to replicate the data on type, the replication was size oriented.

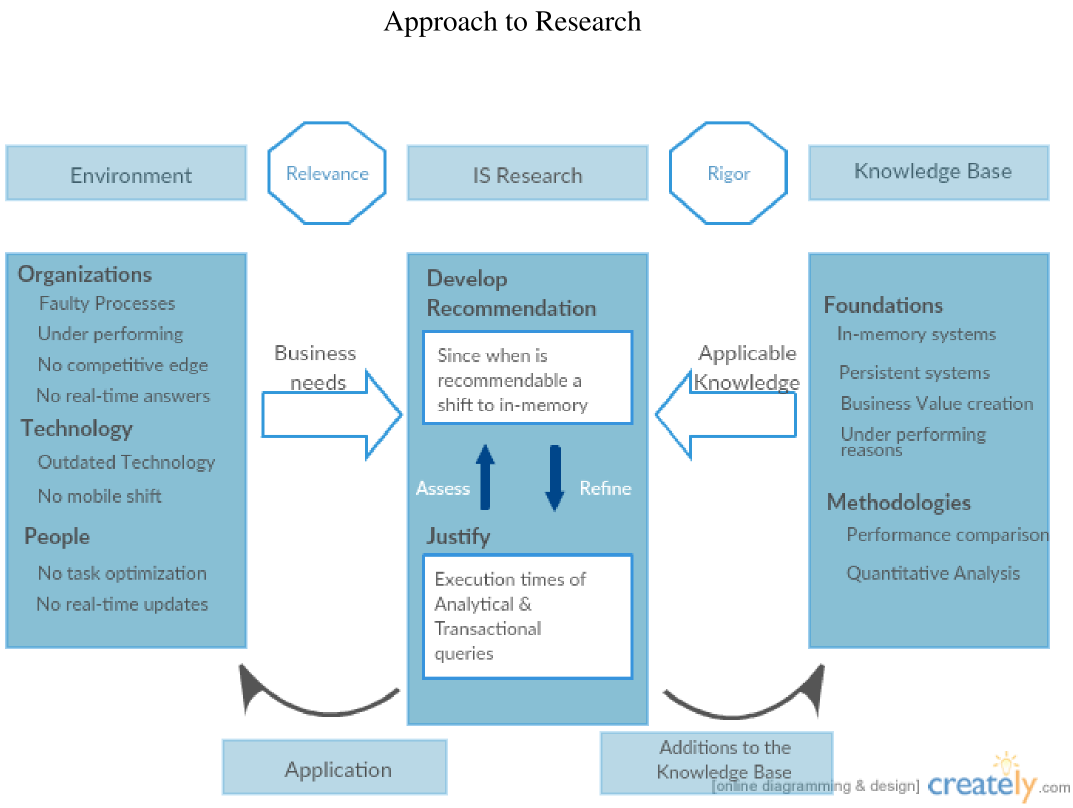


Figure 4.2: Information Systems Research Framework [Hev04]

Without this kind of data, it was necessary to find other solutions. It was important though, that the arranged solution would use real data, so the tests would be somewhat similar to what companies use.

So, we started to gather weather data, which produces large amounts of data, and combine with data from flights, takeoffs and passengers. However, even with this data isn't possible to gauge relevant results. In-memory is directed to big data, so it was mandatory to get larger data sets. After some weeks of searching, we found a large enough data set which would allow meaningful conclusions, as it's possible to see in the next chapter.

4.3.1 Source

Since it wasn't possible to get real data from a company, it was chosen to use real, meaningful data, that would relate until a certain point with data used on businesses.

The main table, the Exportation table (see figure 4.3) which is considered this way because it was the only data set with a large cardinality of entries, and the SITC description table (which will be explained in the next paragraphs) were obtained from the Observatory of Economic Complexity [MIT10], which was initially a Master Thesis at the MIT Media Lab. They gathered the data from *The Center for International Data from Robert Feenstra* and from the *UN COMTRADE*, as it's possible to see in their data source page.

The data from the other tables, such as Country, GDP, Passengers transported by country and number of takeoffs by country, were gathered from the World Bank web-repository, which contains a varied number of indicators, divided by country. [Wor]

After everything was put together, the organization of data was like in figure 4.3 in the Hana test app.[1]

The main table, Exportation, is the one which the test will focus more, since is the one containing more records. It shows the product trade on a bilateral way between countries, in the period 1964 - 2014. It contains the code of the origin and destination country, the export and import value and a SITC code. SITC stands for Standard International Trade Classification, which translates a set or type of products into a four character code.

The data from World bank is similar to each other. It contained, for each entry, the value of the respective field (GDP, number of passengers and number of takeoffs) in each year. The data had to be re-arranged, as it will be seen in the next section.

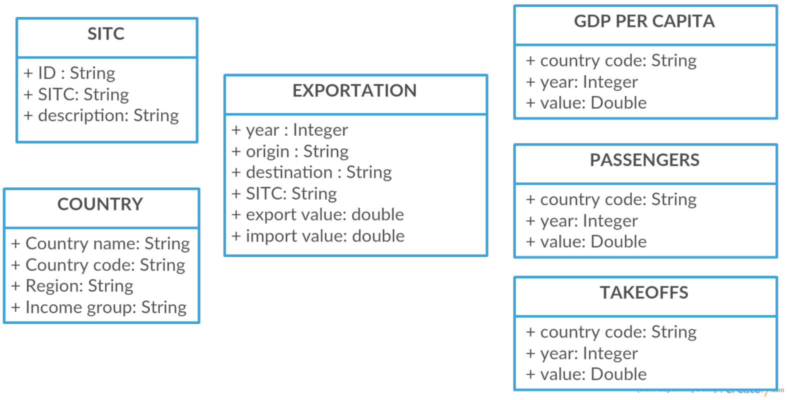


Figure 4.3: Organization of test data - Hana

4.3.2 Suitability

In spite this data is not the most common for analytical queries, such as sales,finance and shipping reports (wider tables), the structure of this data is also similar to some of the narrow tables that companies use, such as wage processing. Through this, it will be possible to assess with certainty how much better the queries will perform in this kind of data. The dimension of the data used is also comparable in scale to what medium and large companies use.

The data files came in CSV and TSV types. When the data comes from different places, it's likely that the data has to suffer modifications to suit the database tables.

The SITC table codes were inconsistent with the exportation table ones, with some codes missing the first digit, if it was a zero. This was corrected manually. Some text entries came with quotation marks, namely in the country table, which would pass to the database as well. To remove them was used the *Regex* functionality of the *Sublime Text 2* program. This functionality was also use to remove the decimal part of the number of takeoffs and passengers of the respective file.

But the main difference was between the organization of data from the World Bank (GDP, Passengers and Takeoffs) and the OEC (Observatory of Economic Complexity). Each year from

[1] In R/3 the data types are somewhat different due to syntax reasons

the World Bank data was a column, while the OEC data had a column named *year*. The queries would be cumbersome if an uniformed way wasn't found. To overcome this problem, since Hana don't have a transpose functionality yet, a Java program was developed to transpose the file before it was inserted in the database.

In the end , there was some problems with file's sizes. The exportation data file was too big to be inserted at once, both on Hana and on R/3, and it was needed to divide in smaller chunks. It was developed a Java program, which divided the file in smaller chunks. The size chosen was 600MB for Hana and a smaller size of 60MB for R/3.

4.3.3 Validity

Choosing to use test data different from what companies use raised some concerns, mainly because the results also depend on the type of data used. If, for example, the enterprise tables are more commonly wider than deeper, the results will be different.[2] But in the most important characteristic, which is size, was replicated.

Nonetheless, the scope of this dissertation is focused on narrower tables with a large number of records. To enterprises with data similar to this one, this study can be helpful to make a decision to change to a system like Hana. This kind of data can and is also used by companies. Smaller reports and wage reports are, in fact, narrow tables. Also, the data generated by devices integrated with Internet of things usually have a small number of fields, producing narrower tables.

Other possibility was to generate data to fill the database. This route was not taken by taking in consideration two reasons: First, to generate data for completely different tables would be a cumbersome process and would need to follow a set of constraints and rules on the generation phase. Secondly, it doesn't allow to have a real grasp on the system performance, since the data as no meaning. For example, in our tests, it's expected that the exportations and importations suffer a breakdown in most countries between 2007 - 2010, because of the financial crisis that struck mainly Europe and the US.

4.3.4 Size

Every company has different data loads which vary while they develop. The objective was to address large amounts of data, since in-memory was made to reduce complexity and execution times of database operations with large amounts of data. The tables vary in size and complexity, ranging from a couple hundreds of records to more than one hundred million records. In the table 4.1, it's possible to see the maximum number of entries for each table.

To have some understanding on the size of data, a medium size air carrier has tables with about twenty million records. These tables usually are more wide than deep, with around forty to fifty columns. Depth oriented tables on the other hand are also use, for example, in transactional data from machines or companies with a high level of automation and control. This is an example of a

[2] Batch inserts, for example, perform worst on tables wider than 80-100 columns [App13]

SITC	988
Country	247
Exportation	103 152 822
GDP per capita	12 152
Passengers	12 103
Takeoffs	12 152

Table 4.1: Number of records in each table

single table, and it's important to note that it can take many forms and complexity, according to a company's necessities.

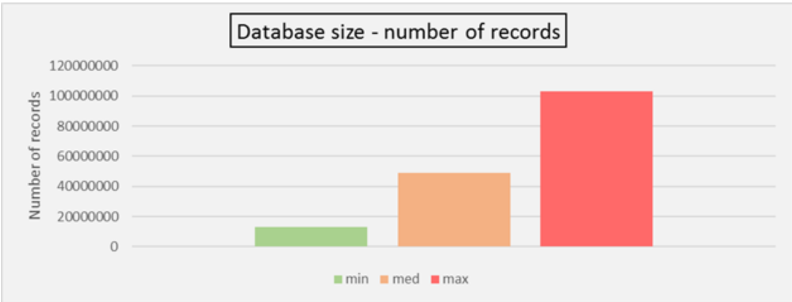


Figure 4.4: How the data load increased on the three iterations

Since data can take many forms and complexity, the objective was to slowly increase progressively the data, so it was possible to have a wide analysis how queries behave, but unfortunately, the availability of the servers was limited. So, to realize the tests, it was decided to divide the load of the exportation tables in three, as it's possible to see in figure 4.4. The data was divided in this particular way because since it wasn't possible to verify the performance of the tests progressively, the objective was to verify the different loads in the minimum number of iterations possible.

4.4 Evaluation method

As was stated before, the goal of this dissertation is to present a concrete value proposition by *SAP* Hana. Ideally, on the analysis of query's performance it should use similar queries as enterprises, such as yearly and quarterly reports, calculate margins, losses, receivables, among others. However, such data was not available. So, in spite it was not possible to replicate the data in terms of quality, the data was replicated both in size as it was in complexity.

The analysis that will be made will not take in account the number of requests to the database. This means that when taking note of execution and response times, there will be only one request at the time to the database. This approach was taken because there is no way to replicate the number of requests in a way that would be similar to a real organization without generating fake queries. Also the availability of the servers and being a developer edition limited the possibilities.

For this reason, if the tests are run several times, it's likely that both execution and response times are similar, because there isn't other external factors that will influence the results besides the idleness of the database. So, to have a concise analysis on the the tests, it's enough to follow a slight adaptation of the the statistical analysis of discarding the outlying results [Qua], without further analysis on the outlier, since it will occur due to idleness of the database or high latency of the connection.[3]

To find if one result is an outlier, we have to find the IQR (Interquartile Range). The IQR is calculated based on the difference between the third quartile (bigger than 75% of the data) and the first quartile (bigger than 25% of the data). The Q1 and Q2 are found using the second quartile (or median), which splits 50% of the data. [Qua]

$$IQR = Q3 - Q1 \tag{4.1}$$

The outliers are found using the the 1.5 x IQR rule. They are outliers if:

$$< Q1 - 1.5 * IQR \tag{4.2}$$

or:

$$> Q3 + 1.5 * IQR \tag{4.3}$$

In the analytical tests to be run, it isn't expected significant variation on the results, so each query will be made five times and the result will be the average of the execution times without the outlier values, if they exist. If there's a lower outlier, it won't be discarded. This approach was taken because if there's an outlier outside the lower bound, it means that it had lowest connection interference and idleness of the database. On the persistent database it's expected an high variation on the results, since the server is often being used for AMT's overnight jobs, which will bias the execution time.

In the transactional tests, such as queries involving INSERTS, UPDATES and DELETES, the tests will be run a single time. The main reason for this approach is due to the low execution times associated to these types of queries. Adding to this, in Hana, UPDATES and DELETES are not used as often as they are in persistent memory RDBMS.

4.5 What is the value proposition of SAP Hana?

Enterprises are entities oriented to maximize profit and success, following specific goals and objectives, in order to thrive. Several factors combined together are the reason why businesses thrive or fail. One of this factors, as it was noted before, is Information Systems artifacts, like databases. [Hev04]

In order to implement newer and modern artifacts, it's imperative to demonstrate, through evaluation methods designed purposely for it, to verify it's utility, quality and efficacy of the artifact. In this dissertation, the verification will be made on the handling of analytical and transactional

[3] there isn't enough information about both server's connection's latency, so these values can be discarded.

queries through measurement of execution times. Nonetheless, the analysis that will be made it's not sufficient to prove the utility, quality and efficacy of SAP Hana, requiring other types of analysis to have a fully perception of it's value.

Summing it up, The answer to this question will focus on the performance of Hana with analytical and transactional queries, instead a fully analysis to the Hana platform. The objective is to answer the if and when a change to in-memory appliances like Hana can bring valuable competitive advantage and business value to companies.

These methodologies will be applied in a solution to test the value and performance of Hana. In the next chapter a description of the solution experimented and the respective results.

Chapter 5

Implementation & Experimentation

To answer the investigation question, the if and when should be considered implementing an in-memory solution such as Hana, it's crucial to know in what degree Hana performs better than their previous and most widely distributed solution.

To analyze if Hana can really create value and business competitiveness, part of the job is to test the performance of the database. To verify if a database performs better than other, a large and specific set of tests have to be run, only at the range of specific companies to do this job. This kind of approach would not fit in the scope of this dissertation. The focus of this dissertation is analyze how transactional and analytical SQL queries perform on both in-memory and disk-based databases in order to answer the investigation question. The performance depends also on the kind of data used. Some queries may perform differently in row or column oriented table.

5.1 Experimentation

To answer to the investigation question, it's necessary to build a scenario to test both databases and afterwards, make a comparison. The scenario will consist in two applications that will be able to run the same set of tests. Both applications are different, since the languages to build applications in Hana and R/3 are different, but their outcome is the same, the execution time of queries. These two apps are relatively simple, but their purpose is important.

The metrics used to compare the performance of queries in the two systems is the execution time of queries. The execution time is defined by the "the time spent by the system executing that task, including the time spent executing run-time or system services on its behalf" [Ada]. It's the time since the query is sent to the database until an answer is received.

The queries to be run are exactly the same on Hana and R/3, without any kind of performance improvements in mind while developing it. The syntax is different on queries for Hana and R/3, due to the different names on tables and fields and minor differences on syntax. The analytical queries were developed with relevance and analytical objective in mind, such as analyze patterns and the behavior of exportations, passengers and takeoffs during specific time periods. The transactional queries don't have any particularity beside the fact that in Hana tests the performance with and without dictionary re-writing.

The organization of the data is slightly different in both systems. While in Hana it's used a columnar organization, since in-memory is leveraged by the columnar organization (as it was approached before in literature review), in R/3 there's not a specific architecture defined. With this columnar organization Hana, it's to expected worst results in transactional queries. Since the tests contain both analytical and transactional queries, the chosen R/3 the architecture wasn't defined.

The two systems, as represented in figure 5.1, were connected by a remote connection. Hana was stored in one of SAP servers, in the cloud, while R/3 was stored in AMT facilities, and the connection was through a VPN connection

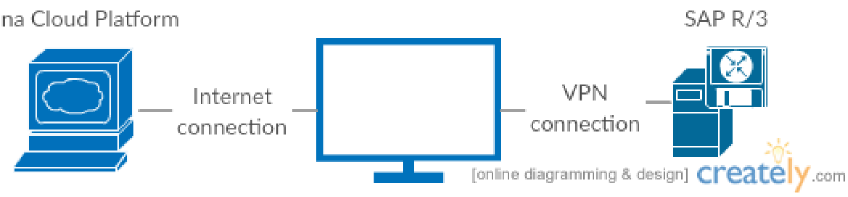


Figure 5.1: The two kinds of in-memory systems of data parallelism

In the next two sections (5.1.1, 5.1.2) both scenarios will be described more deeply, to better understand how the tests platforms were designed.

5.1.1 Hana Test App

As it was mentioned before, the Hana test app was built using the MDC (Multitenant Database Containers) database, since the slimmer 1GB version didn't had enough memory to run the amount of data to test. This was the first step taken to start the development of the app.

The first step was to create an user with the necessary roles to built the app. The privileges are given through the web IDE provided by *SAP* to create apps for Hana. The web IDE is divided in 4 sections: Editor, Catalog, Security and Traces. The Security tab was the one used to give the privileges to the new user, the Editor tab is used to develop and test the app, the Catalog tab allows to manually make SQL queries to the database and the trace tab stores the error log.

After the User has the privileges to create and test apps, the creation of the contexts (tables) to store the data was the following step. The data was imported in two ways: Through the web IDE if the data sets were small and through Eclipse (with Hana integration) for the large data sets.

Then, the objective was to run the tests on our data, stored in the database. In order to achieve this, it was necessary to write the server code in order to send the queries to the database,using Hana's server-side language, *XSJS* (javascript). The client-side allows the user to visualize which tests will be run, the execution times of each test and the respective result-set was developed using HTML and the *SAP* UI library.

The application behavior, as illustrated in figure 5.2, is simple. The queries to be executed are read from a file stored in the server. Whenever the user wants to execute the queries, they're sent to the database, one by one, and the result come back to the server, which saves the result and the execution time. When all the queries finish, the client receive the results and the execution times. The client also take note of the time to receive the results, the display time. The user can download the .csv file with all the queries and their respective execution and display time and can see the result set of each query online. To change the queries to be tested, the file in the server has to be updated.

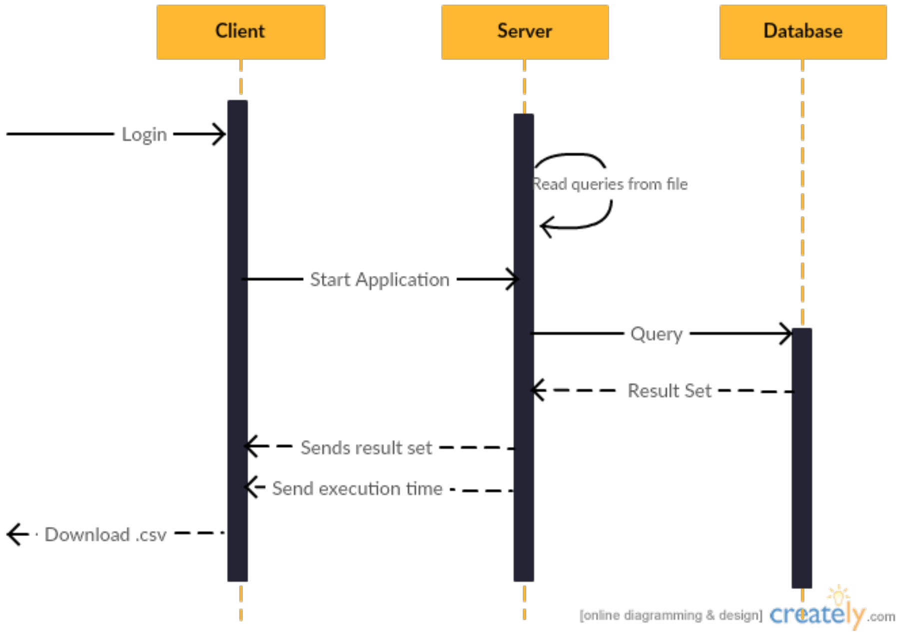


Figure 5.2: Sequence Diagram - Hana Test app

5.1.2 R/3 Test app

To run the tests in the R/3 database, it was through SAP Netweaver, the GUI for Windows, version 740.

Through the Netweaver, the same tables as Hana were created. To load the data into the tables, it was through a program developed to load the .csv files into the the ERP and then another program was written to load the folder with all the files into the tables. All the programs used in the R/3 database tests were developed using ABAP.

Another program was developed to run the SQL queries, since the usual way to to select data from the database isn't through direct SQL and would involve a more cumbersome process. The queries are hard coded and it allows to select which queries to run. The program usually run in background mode, to prevent the idleness of the computer. The result set and the execution time of the query/queries are displayed on the screen. The execution times are taken manually.

5.1.3 Experiments

The tests will be transactional and analytical SQL statements, such as SELECTS, UPDATES, INSERTS and DELETES. It will focus in the execution times on Hana and on R/3, using tables with a huge number of rows and a low number of columns. The tables use a columnar organization on Hana and and on R/3 and the number of records will vary between 3 states, so we have a comprehensive analysis on the behavior with different amounts of data.

The analytical queries also comprehend different keywords, such as sums, averages, inner joins, etc to assert if some queries with specific keywords influences the performance in a particular system. The queries also involve a different number of tables, so it's possible to analyze if the query execution time also change depending on the number of tables.

To evaluate the performance of both databases, the test set needed to be the same.[1] The four types of statements (Select, Insert,Update and Delete) are included in the test-set. The complexity of analytical queries vary, ranging from simple selects to complex inner joins with a various number of tables involved. The Select statements (table 5.1) are expected to have a more variation than the Insert/Update/Delete queries - Appendix B B since transactional queries are usually faster than analytical queries.

1	Exportations made from Thailand to Sweden of Milled Rice on 1962
2	Volume of exportations & importations made from Portugal to Spain between 1963 and 2010
3	All types of exportations made from Portugal in 1999
4	Which type of exportation was the most profitable one in the US in each year
5	Variation (%) in the total value of importations in Portugal between 2000 and 2010
6	Destinations China exports and the respective % is from all exportations in 2010
7	How the exportation of Medicaments by France evolved through the years (1962-2010)
8	Relationship between the Importation of Aircraft related components and the number of takeoffs in Europe and Central Asia
9	What each country exported the most and in which year
10	Which exportation grow more than 100% between 1979 to 2010 in Russia
11	How the GDP per capita and the exportations grew between 2009 and 2010
12	How the Average growth in importations correlates with the Average growth in passengers and takeoffs in Portugal
13	Ordered list of which type of exportation grew the most between 1989 and 2010 in Portugal
14	Average growth in importations vs Average growth in passengers and takeoffs in the American Continent
15	Average growth in importations vs Average growth in passengers and takeoffs in Europe

Table 5.1: Analytical statements to be tested

The objective with these analytical queries is to evaluate thoughtfully the system, so these queries vary in terms of complexity and tables involved. The reasons why these queries were developed are:

[1] There's some slight changes on the queries when the number of records is changed, since there's not enough data on the minimum capacity to take conclusions

1. The first query is to test how the system behave handling a simple select statement, retrieving a single record from a table.

2. This query demands a little computation, summing the exportations between two countries.It's rather simple, but involves two tables

3. This simple query fetches data from three different tables, by crossing the data from the country table and the SITC table with the exportations from one country.

4. This already involves a significant amount of computation, by comparing the volume of each type of exportation by one country for each year and crossing with the country and SITC table.

5. This query only fetches data from one single table, but involves some computation, since it will compare all the importations made by a country in two different years.

6. This query was oriented to see how math keywords, such as an average of values, is handled.

7. This particular tries to emulate a observation on a evolution of a particular item over a couple of years.

8. This particular query is a bit more complex, since it will find any related aircraft components using the "LIKE" keyword, involving 4 tables to fetch the final result.

9. This one involves uses a large amount of computational resources, since it will travel through all exportations by every country and see in which year each country exported the most.

10. This query is rather simple, but already involves the inner join of three different tables

11. This is a more complex query, involving already several inner joins (joining several tables through joins consumes a considerable amount of resources)

12. This was a query developed with the aim of being complex query, but involving several inner joins, as well as different keywords, such as averages of importations and exportations.

13. This query was designed with what would companies use in mind, since it compares the growth of exportations between two years.

14. This query is similar to query nº 12, but from a group of countries , provided by the user.

15. This query is similar to query nº 12, but from a different group of countries , provided by the user.

5.2 Server's characteristics

5.2.1 Hana Cloud Platform

Without the possibility to make tests on a full Hana instance, since they're expensive systems, it was opted to use the *HCP Developer Edition*, that allows to use Hana for evaluation and experiments. This enables hosting multiple *SAP* HANA databases on a single system, sharing the same resources.

Even though the server is hosted in the cloud, this actually can be an advantage for companies. They can pay less for a cloud-based server than a dedicated server to run Hana, and remove all the problems of an installation and can scale much more easily.

The details of the server are not totally known, since is a trial instance. The normal HCP trial instance only have 1 GB of memory and one CPU core. Using a MDC database, the available RAM is 242GB. The number of CPUs and cores is unknown, but the percentage of usage is usually very low (under 10%). The disk capacity and usage is unknown.

5.2.2 R/3

R/3 runs on Windows Server 2008 R2 Enterprise Edition 64-bit. The server's CPU is a Intel Xeon E5335 with 2 GHz and has 12 GB of RAM installed.

5.3 Measurement and expected errors

To measure the execution time in Hana, a timer is set before the query is sent to the database and after the result came. The execution time is the difference between the after timer and the before timer. Furthermore, in Hana, the display time (the time it takes to the user obtain the results) is also noted.

The tests are made five times on both systems, to reduce the errors. It was taken in account that it may occur variations in the results due to errors, so the tests were made interchangeably and spread in time. As it was stated in chapter 2, the outlier formula was used, to discard abnormal values, if they happen.

The expected errors in measuring the time are abnormal high execution times. This can happen if the connection latency is high or the usage of the server was high, due the usage of *AMT* day-to-day processes. On Hana, while testing the app, sometimes high execution times occurred on the first query, after the database being idle for some time.

While developing the queries and testing both apps, some errors occurred. Since the tests in Hana are running in the developers edition, the memory available to run the queries is limited. The most common error while developing the queries to test was out of memory in the SQL processor. This involved reducing the complexity of some queries.

```
cannot allocate enough memory: Out of memory in SQL processor
```

Another error that happened was the overflow of selected fields, due to sums of large numbers. This error happened in on of Hana's test on the 14th test, with the maximum load.

```
Error: (dberror) 314 - numeric overflow: search table error: [6944] AttributeEngine
    : overflow in numeric calculation
```

In R/3 however, another type of errors appeared while running the first tests. If a query, for example, had sub-selects[2], the type of data would need to be pre-structured, because it couldn't resolve the type of each field.

It's predicted that Hana will have lower execution times than R/3 database. However, it's not known how much faster Hana can really be. In the next chapter, we will verify the performance improvements and take conclusions on the business value of Hana.

[2]Selects inside Selects

Chapter 6

Experimentation results

It was expected that Hana database would be faster than R/3, but now we have a real measure in how both systems scale and how much better Hana is. The tests conducted gave us insights on how the two systems behave with transactional and analytical queries, and confirmed the claims SAP made on Hana's launch.

6.1 Hana

First, the analysis focused on how the system scale and handles the increase of data, without comparisons between the two systems. It will be based on the execution time of transactional queries (Inserts,Updates,Deletes) and analytical queries (Selects) in the three different loads of the database.

6.1.1 Inserts - Table B.1

As it's possible to see in graph B.1, the execution time of a single insert query doesn't change significantly with the load of the database. They are almost instant, with differences ranging between 1 millisecond between the three data loads, which can be considered negligible.

To load the data into the database with batch inserts, Hana made approximately 3174 inserts per second. This means about 3 inserts per millisecond, including the delay from the connection, meaning that the execution time is even less, in spite that's not possible to measure it. Hana have fast insertion times, as it will be possible to compare with R/3 insertion times, further ahead.

6.1.2 Deletes - Table B.3

Deletes in smaller tables are as fast as inserts, however when the exportations table is with the full load, it takes a longer, between 14 to 20 milliseconds (which is still considered real-time) - see graph B.3. This may have happened due to small latency in the connection, and with such small error margins, it's impossible to assert the origin of this delay. At this scale nonetheless, this differences are still negligible.

6.1.3 Updates - Table B.2

Updates on the other side, are a bit more expensive, but such statements still have real-time execution times. By analyzing the graph - B.2, it's possible to assert that even updating a single record, the update execution time grew on an exponential manner. Batch updates also grew as the data load increased. In small tables, such as the Takeoffs table, the updates time remained low.

6.1.4 Selects - Figure B.4

The Analytical queries, which were the ones more in focus on this dissertation, behave differently between them, as it's possible to see on figure B.4 , so a specific analysis has to made, since some queries maintained the executions times with the increment of the data load, while others increased on an exponential manner.

The more simple queries, such as 1,2 and 3 (see figure B.4), have constant and very low execution time through different data loads, so it's possible to assert a good scalabilty of the execution times.

On the other side, queries 4 and 9 are similar and behave differently from the remaining.On the first two iterations, the response is in real-time, however on the third iteration, it took unusual high execution times compared to the others queries. Query 4 selects the top exportation in the U.S in each year. To do this, it will course through every type of exportation, for each country, for each year and check which one is the greatest. Instead of checking for only one country, Query 9 verifies which volume of exportations was the greatest and in which year. This query (9) uses four inner joins. However, no relevant differences were verified in the execution times of both queries. It's not possible to take more conclusions on why it took over two minutes on these two queries, since it should at least have increased on a exponential manner.

On queries 7,8 and 13, a similar behavior as the last ones, but kept the executions times lower than one second, which is good for this kind of analytical queries.

Queries 6, 10, 11 and 12 have very low execution times (under 250 milliseconds) and doesn't increase with the database load. On the other side, 15 increased exponentially, so there's no optimization of the query regarding the increase in data.[1]

6.2 R/3

In R/3, some unexpected problems happened and it wasn't possible to have the full results like Hana. The server had memory issues handling the queries with medium load and couldn't execute at all queries in the maximum load. Due to this reason, the transactional queries are constrained to the first iteration and the analytical queries to the second.

[1]Query 14 was expected to behave similar, but due to an overflow error, it wasn't possible to measure the execution time under maximum load

6.2.1 Inserts - Table C.1

Regarding the insert results, the execution times inserting in the main table - Exportation table - are relatively high and already fall out of the real-time scope. The insertion in a smaller table, like the takeoffs table, has execution times under 50 milliseconds, which are real-time.

In terms of batch inserts, R/3 makes 21 inserts per second, including the connection time.

6.2.2 Deletes - Table C.3

The results of the delete statements were very similar to the insert results. The delete statements on the exportation table (with the lowest load) aren't in real-time, while the queries in smaller tables have real-time execution times.

6.2.3 Updates - Table C.2

The updates expensive where a little more expensive than inserts and deletes, with execution times of 4 seconds if executed in the exportation tables. If these statements were executed in a smaller table, the execution times are also more expensive, around half a second. This was only executed, like inserts and deletes, on the exportation table with the minimum load.

6.2.4 Selects - Figure C.1

Regarding the select statements, not a single query had an execution time lower than two seconds, falling out of the real-time scope. A simple select to the database with the lowest load takes almost five seconds.

The R/3 server had memory issues regarding handling these queries. Due to this reason, the tests in the database with maximum load were unachievable.

The queries 3,4,5 and 11 had incoherent results, since the execution times in the database with minimum load are bigger than the times with medium load.

Queries 9,14 and 15 were not enforceable, since it was not possible to retrieve a result set in a reasonable amount of time .

Queries 1,2,6,7,8,12 and 13 behaved as expected, meaning that with the increment of data, the execution time of the queries grew as well. However, this increment is erratic, since the increment is rather different from query to query.

6.3 Discussion

6.3.1 Transactional Queries

The results obtained on these test confirm that Hana has effectively real-time execution times, for transactional queries.

Based on the results from Insert queries, it's possible to assert with certainty that the increment of data doesn't have influence on the execution time of insertion queries on Hana. It's also possible to assert that a columnar organization doesn't provoke any delay on the insertion of data and that the dictionary re-writing doesn't delay significantly the insertion queries. In R/3, in spite we don't possess more than one iteration of inserts, it's possible to verify through the inserts in two different tables, that the insertion times increase with the amount of data.

Comparing the insertion times of both systems, as seen on table 6.1 ,it is possible to verify that the Hana insertion times are much faster than the R/3 ones. A single insert on Hana is, on average, 938 times faster than an insert on R/3, while batch inserts are 151 times faster. Since insert times on Hana is almost the same for large or small tables, if we compare only the insert times on the exportation table, this difference increases to 1729 times.

	Hana	R/3
Single Insert	1.25 milliseconds	1173 milliseconds
Batch Insert	3174 per second	21 per second

Table 6.1: Average execution times of single and batch inserts

On the other side, it was verified that delete statements behave on a similar way to select statements. While in Hana this kind of statements are a little more expensive than inserts, they're still in real-time, and even if slight variations of the execution times were verified, these are small, which means that delete queries scale well with increasing data loads. In R/3, the average times for deletes in tables with a high number of records is 53 slower than inserts on a smaller table.[2] These delete times on R/3 are double of insertion times (around 4.5 seconds, compared to the 2 seconds of a insert).

On the update statements, it is also possible to take conclusions about the value of dictionary on Hana. By analyzing this with the data from inserts and deletes on Hana, it's possible to assert that an delete followed by an insert can be an alternative to update statements, since the combined execution times of insert and delete is usually lesser than an update request to the database, as it's possible to verify in table 6.2

	Insert + Delete	Update
1	10 ms	38,66667 ms
2	6,333333 ms	107 ms
3	3 ms	2 ms
4	2 ms	3,666667 ms

Table 6.2: Insert + Delete vs Update statements on Hana

While on Hana this insert + delete makes sense, in R/3 it doesn't work in the same way. As it's possible to see in table 6.3, insert + delete are only "cheaper" if the number of records is low. Otherwise, an update to the database is usually cheaper.

Updates statements alone on Hana run, on average, 71 times faster for high loaded tables and 322 times faster for tables with fewer records.

[2] The increment of data in the exportation table is, however, 1083 times more than the takeoffs table

	Insert + Delete	Update
1	7139 ms	4808 ms
2	6713 ms	5572 ms
3	102 ms	855 ms
4	96 ms	972 ms

Table 6.3: Insert + Delete vs Update statements on R/3

Overall, in terms of transactional queries, Hana runs way faster than R/3. Hana was designed to handle both type of queries with high performances. This means that it can be a great alternative if a company wants a real-time feeling even for transactional queries, even if R/3 execution times aren't prohibitively to have a good work flow with transactional data. On analytical queries, it will be possible to verify that this difference is even bigger.

6.3.2 Analytical queries

Hana has incredible performances compared to its ancestor, regarding analytical queries. Combining the performance of transactional queries that were analyzed in the previous sections, Hana can be the game changer regarding databases.

Overall, Hana response times for transactional queries are 100% in real-time. For analytical queries, in-memory behaved in real-time with loads until 50 million records. With more than 100 million, most queries behaved in real-time, while others executed in relatively short times. In R/3 not a single query executed in real-time, independently of the data load. Plus, R/3 couldn't run 3 queries that run perfectly fine on Hana on a reasonable amount of time.

By comparing the average execution times of analytical times with the two loads of data (since it wasn't possible to test R/3 with the full data load, the comparison wasn't possible), in figure 6.1, it's possible to see the abysmal difference between the two databases regarding execution times. Overall, the average of execution times on Hana is **4164 times** faster than R/3.

	Data load increment	Hana execution times Increment	R/3 execution times Increment
2	382,7967 %	374,8448 %	142,8822 %
3	806,5746 %	657,3039 %	[redacted]

Table 6.4: Comparison of data and execution times increment of analytical queries

By comparing how the executions times scale, we see that Hana execution times increase, on average, as the data increases. Surprisingly on R/3, the increment is actually lower than the increment of data.[3](see table 6.4). However, the execution times with the lowest data load are already very high, and this increment is not taking in account that the system couldn't execute three queries.

[3] we only have two iterations of data, so a full comparison wasn't possible

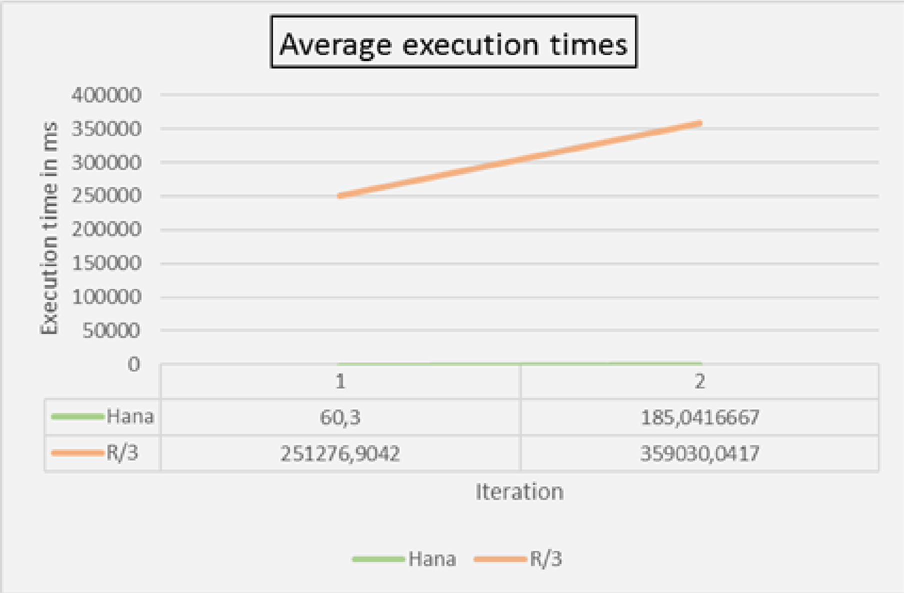


Figure 6.1: Average execution times of all 15 analytical queries on Hana and R/3

Overall, the executions times on R/3 surpass the ones from Hana on a large scale. This will introduce a new paradigm on database and system design, which will only get better in the future, as it will be possible to see in the next sections.

6.4 Validity

These results have some limitations. The server were R/3 is stored maybe wasn't designed to handle this amount of data or complexity of queries. In spite we don't know for certain Hana's server characteristics, this was the only server of this kind to run the tests.

However, regarding the validity of these results, it's possible to verify by comparing the results obtained in this dissertation with the claims from SAP. SAP claimed that their computing technology runs 3600 times faster than old appliances.[SAP11] By comparing our test data with SAP claims, the results are somewhat similar to the ones in SAP. Our tests on Hana run, on average, 4164 times faster than R/3, regarding analytical queries. This slightly difference can be explained by the uneven R/3 server usage, since it wasn't 100% dedicated to the tests. The similarity of this research with *SAP* researches validate the obtained results.

The tests were scheduled to avoid peak server's usage and run multiple times to reduce the error margin. The times were measured with the computer clock, round to the milliseconds. This approach was to reduce the error margin inherent to these tests.

6.5 Generalization

While the tests in R/3 were made on a known server, with known specifications, Hana's on the other hand were run on a cloud based server, dedicated to developers first steps. The specifications are rather unclear, but the results are similar to what SAP claims.

The main limitations of these experience are the data quality, since wide tables are more common when using analytical queries. This data, as it was stated before, is hardly available, so an analysis to two different types of data wasn't possible.

Despite the fact these tables are rather different, some business have similar data to the one tested. For example, solutions on the field of Internet of Things can use similar data to the one used, as well as wage reports from companies. So it's possible to generalize to depth oriented tables the results obtained from this research.

6.6 Value Proposition

Hana truly delivers, in most cases, real-time answers, even with huge database loads. With these capabilities, enterprises are able to transfigure themselves and develop competitive advantage never seen until today.

Through the addition of the knowledge about why businesses are not thriving, such as the slow mobile migration, with the behavior of in-memory and the test results, it's possible to assert with confidence that an in-memory solution will be part of a mobile migration.

The executions time of Hana tests were, as expected, really low and belong to the real-time scope, with the exception of two queries exceeding the 150 seconds. In most cases, these times scaled well with the increasing data load and it's possible to deduce that it will work in this way with slim database tables and with similar amounts of data. **R/3 database is around 4000 times slower compared to Hana in analytical queries and 100 to 1000 times slower regarding transactional queries.** In R/3, not a single analytical query was in real-time, which precludes any attempts of using this kind of ERP with such amounts of data on a mobile solution.

"Since when opting for an in-memory solution, such as Hana, can create business value?"

Catching up with our investigation question, we can certainly say, that with this kind of data, an in-memory solution is countless times better than a persistent database. This will allow do things that were unfeasible before, proven for example, by the three queries that couldn't run on R/3.

To Answer the investigation question, a reverse approach will be taken, as it will be advised when not to choose an in-memory solution like Hana. A company should not opt for this kind of solution if:

- their business is not time critical, meaning that fast decisions and fast data entry is necessary.
- making decisions is occasional and doesn't require day-to-day supervision
- it's not database dependent or if its usage is low

- a mobile solution integrated with the ERP isn't part of actual and future plans for the company
- a company is still relatively small and doesn't have the amount of data on the database to make it viable

In case these conditions apply, is not recommended to invest in in-memory systems like Hana and keep their current solution. However, if these conditions doesn't apply, it's recommended to opt for an in-memory solution such as Hana, since the improvements in executions times allow fast decision making and to use idle data that was being thrown away. So if a company is relatively big, have a mobile ERP solution and works with larges amount of data, it's recommended to opt for a in-memory solution, since the business value created is huge.

However, there's still a need for more research regarding this subject, since a lot of parameters have to be taken in account to make such a disrupt change. In the next chapter we will conclude the investigation for this MSc dissertation , as well as setting the path for future work on this subject.

Chapter 7

Conclusions and Future Work

Five years ago, the first steps on in-memory solution for enterprises were taken, with the launch of Hana. A solution combining the analytical queries with and on transactional data at the same time, makes the so called on-the-fly aggregations feasible. This kind of technology implemented on big companies is still on the early stages, but in the future all enterprise applications will be built in an aggregation and redundancy free manner.[Pla14]

The resulting reduction in transaction complexity leads to simplification of data models and applications, simplifying and modernizing how enterprise systems are built, mainly through the elimination of updates to transaction maintained aggregates.[Pla14]

This simplified development is also verified by the author, through the development of both apps. While Hana was developed using javascript, a widely spread programming language, to test the queries in R/3 the programs had to be developed with ABAP, a not so known language. The time taken to develop and test the apps for both system isn't comparable.

With an average improvement of about 4000 times for analytical queries and 100-1000 times in transactional queries, Hana is definitely a disrupt technology. Most important than that, was the verification of real-time execution times even with large amounts of data, confirming *SAP* allegations.

The objectives set in this dissertation were accomplished. Hana is effectively an innovative solution in the databases market. It was verified by the tests that Hana can create value through huge performance improvements on execution times on depth oriented tables. These tests were made to complement the base of knowledge of in-memory systems and to verify if Hana indeed creates business value.

7.1 Research Limitations

Along the development of this dissertation, some issues appeared, limiting this research, but new approaches where taken as difficulties appeared.

First of all, it wasn't possible to use a dedicated Hana server to run the tests, since it's a very expensive system. Instead, a system with unclear specifications was used, but the obtained results were similar to SAP's results, validating them.

R/3 server had limited resources and availability, since it was used for day-to-day company work. so the tests were extend for a long number of weeks and scheduled to be done mostly at night or weekends. Regarding the resources problem, the tests shut down the server countless times due to many reasons, including abnormal disk usage. This is also a point in favor of Hana, since none of this problems were encountered while running the tests on Hana.

Another limitation encountered was the source of data. Since it came from different sources, it demanded additional work regarding its organization. External programs had to be developed to organize the data to a point that would be possible to create relation between them and that would fit in the specific database tables.

7.2 Recommendations to practitioners

As we stated in the previous chapter, Hana can bring business value for relatively large companies who have an implemented ERP mobile solution (or are planning to do so) and where decisions are made constantly and anywhere.

For companies where making decisions for managers and technicians is part of the day-to-day, long waiting times reduce their productivity. Furthermore, decisions are moving out of the office and into the field, with the mobile trend on ERPs. Hana brings exactly that, cloud based servers and easy to develop apps, adding to the huge improvements in execution times. Furthermore, it was verified that a new range of possibilities is open, since it's possible to make queries to the database that weren't possible before.

So if there's a necessity to eliminate long waiting times for compiling monthly reports, develop strategies without overnight waiting times, make decisions on the field, an in-memory solution such as Hana is recommended.

7.3 Future Work

In-memory still have to researched more deeply to strength the knowledge base and to remove, step by step, the "cloud of doubt" that still persists around this technology.

This dissertation was focused more in transactional a analytical queries in tables with high amounts of data. Unfortunately, it wasn't possible to assert if the statements made on the course of all the chapters are correct for wider tables,[1] due to the impossibility to get access to this kind of data.

Another step that can be take is to compare Hana with other persistent memory databases, not developed and supported by *SAP*. It's probably that Hana is better than *SAP's* previous system, but it's important to verify if similar results are obtained against other systems from other companies.

[1] With a higher number of columns

Hana, as it was mentioned before, is not the only in-memory in the market, since *Oracle* and other companies are upgrading their disk-based system as well. In fact, there's no bench-marking by *SAP* against their main competitor, *Oracle 12c*. This is one of the most important researches that can be made, a full study on how Hana and 12c behave. This, however, is only at reach of big companies, who have access to these two types of systems, which are rather expensive.

SAP claims also that the current appliances and applications made by companies integrate well with the new Hana database and it's not needed develop and design new applications. Future research on this topic can be valuable, since it was not possible to verify this due to limitations of time and the nature of Hana's server.

References

[Ada] Ada Reference Manual. http://www.adaic.org/resources/add{_}content/standards/05rm/html/RM-D-14.html. Accessed: 2016-06-02.

[App13] John Appleby. Best Practices for SAP HANA Data Loads. https://blogs.saphana.com/2013/04/07/best-practices-for-sap-hana-data-loads/, 2013. Accessed: 2016-05-31.

[BM13] Christof Büchi and Susanne Mathys. Relational Data Access on Big Data. 2013.

[Cas14] Nick Castellina. MOBILE ERP: TAKING ERP ROI INTO YOUR OWN HANDS, 2014.

[Chr97] Clayton M Christensen. *The Innovator's Dilemma When New Technologies Cause Great Firms to Fail*. Harvard Business School Press, Boston, Massachusetts, 1997.

[CP12] Nick Castellina and Kevin Prouty. Aberdeen Research: ERP in Manufacturing 2012, 2012.

[Dah15] Kishor Daharwal. What Is SAP HANA? http://scn.sap.com/docs/DOC-60338, 2015. Accessed: 2015-02-26.

[Dic] Dictionary - Real-time. http://www.dictionary.com/browse/real-time. Accessed: 2016-05-30.

[FML$^+$] Franz Färber, Norman May, Wolfgang Lehner, Philipp Große, Ingo Müller, Hannes Rauhe, and Jonathan Dees. In *29th IEEE International Conference on Data Engineering*, Brisbane, Australia.

[Fra15] Curtis Franklin Jr. SAP HANA: Not The Only In-Memory Game In Town. http://www.informationweek.com/big-data/software-platforms/sap-hana-not-the-only-in-memory-game-in-town/a/d-id/1320595, 2015. Accessed: 2016-05-30.

[Hen] Kristina Henn. Is the SAP HANA Operational Reporting Rapid-deployment Solution Right for You? http://scn.sap.com/community/rapid-deployment/blog/2013/01/23/is-the-sap-hanaoperational-reporting-rapid-deployment-solution-right- Accessed: 2016-02-05.

[Hev04] S. Hevner, A., March, S., Park, J., and Ram. Design Science in Information Systems Research. *MIS Quarterly*, 28(1):74–106, 2004.

[Kos] Donald Kossmann. Architecture and Database Interfaces.

[KVV13] Mohit Kumar Gupta, Vishal Verma, and Megha Singh Verma. In-Memory Database Systems -A Paradigm Shift. *International Journal of Engineering Trends and Technology*, 6(6), 2013.

[MIT10] MIT Media Lab. The Observatory of Economic Complexity. http://atlas.media.mit.edu/, 2010. Accessed: 2016-05-30.

[Moh15] Ping Wang Mohamed Hassan. a Review Paper on the Impact and the Importance of Management. 4:27–30, 2015.

[Mul15a] Craig S. Mullins. Aerospike NoSQL DBMS overview. http://searchdatamanagement.techtarget.com/feature/Aerospike-NoSQL-DBMS-overview, 2015. Accessed: 2016-05-30.

[Mul15b] Craig S Mullins. Amazon Simple DB. http://searchdatamanagement.techtarget.com/feature/Amazon-SimpleDB-overview, 2015. Accessed: 2016-05-30.

[Mul15c] Craig S. Mullins. IBM DB2. http://searchdatamanagement.techtarget.com/feature/IBM-DB2-relational-DBMS-overview, 2015. Accessed: 2016-05-30.

[Mun15] Matthieu Munch. Row Store vs Column Store in SAP HANA. http://blog.dbi-services.com/rowstore-vs-column-store-in-sap-hana, 2015. Accessed: 2016-02-06.

[Nah04] Fiona Fui-Hoon Nah. A study on tolerable waiting time: how long are Web users willing to wait? *Behaviour & Information Technology*, 23(3):153–163, may 2004.

[OLT] "OLTP vs OLAP: Definitions and Comparison. http://www.dataonfocus.com/oltp-vs-olap/. Accessed: 2015-12-27.

[PC11] Kevin Prouty and Nick Castellina. Mobility in ERP 2011, 2011.

[Pla09] Hasso Plattner. A common database approach for OLTP and OLAP using an in-memory column database. In *Proceedings of the 35th SIGMOD international conference on Management of data - SIGMOD '09*, pages 1–2, New York, New York, USA, 2009. ACM Press.

[Pla14] Hasso Plattner. The Impact of Columnar In-Memory Databases on Enterprise Systems. 7:1722, 2014.

[PZ12] Hasso Plattner and Alexander Zeier. *In-Memory Data Management An Inflection Point for Enterprise Applications*. Springer, Berlin:, 2012.

[Qua] Quartiles and the Five Number Summary. http://www.sfu.ca/{~}jackd/Stat203/Wk02{_}1{_}Full.pdf. Accessed: 2016-05-12.

[R14] Radhika K R. Overview of Sap Hana: In-Memory Computing Technology and ITS Applications. *International Journal of Innovative Technology and Exploring Engineering*, (35):2278–3075, 2014.

[Rev] Revolvy -WebDNA. http://www.revolvy.com/main/index.php?s=WebDNA. Accessed: 2016-05-27.

[Sap] Sap R3 Architecture Introduction. http://sapbrainsonline.com/help/sap-r3-architecture-introduction.html. Accessed: 2016-05-31.

[SAP11] SAP. Ten reasons customers choose SAP HANA® to help transform their business. http://www.tyconz.com/more/Top10ReasonsCustomersChooseSAPHANA.pdf, 2011.

[Sch97] R.R. Schaller. Moore's law: past, present and future. *IEEE Spectrum*, 34(6):52–59, 1997.

[SFGL13] Vishal Sikka, Franz Färber, Anil Goel, and Wolfgang Lehner. SAP HANA: the evolution from a modern main-memory data platform to an enterprise application platform. *Proceedings of the VLDB Endowment*, 6(11):1184–1185, 2013.

[SFL+12] Vishal Sikka, Franz Färber, Wolfgang Lehner, Sang Kyun Cha, Thomas Peh, and Christof Bornhövd. Efficient transaction processing in SAP HANA database. In *Proceedings of the 2012 international conference on Management of Data - SIGMOD '12*, page 731, New York, New York, USA, 2012. ACM Press.

[Soh] Manjinder Singh Sohi. SAP R/3 Overview; Basis Technology. www-bcf.usc.edu/{~}anthonyb/itp320/SAP{_}Overview.ppt. Accessed: 2016-06-01.

[Sul13] Khidr Suleman. IT PRO - SAP aims for 3-second response times with HANA platform. http://www.itpro.co.uk/645034/sap-aims-for-3-second-response-times-with-hana-platform, 2013. Accessed: 2016-05-30.

[vBDMR14] Jan vom Brocke, Stefan Debortoli, Oliver Miller, and Nadine Reuter. How in-memory technology can create business value: Insights from the hilti case. *Communications of the Association for Information Systems*, 34(1):151–168, 2014.

[Voo] Dave Voorhis. Codd's Twelve Rules. http://computing.derby.ac.uk/c/codds-twelve-rules/. Accessed: 2016-02-11.

[Wor] World Bank Data. http://data.worldbank.org/indicator/. Accessed: 2016-05-30.

REFERENCES

Appendix A

In-memory - additional concepts

A.1 Partitioning

Partitioning is the process of dividing logical DB into distinct independent data sets. This is made to achieve data-level parallelism to increase performance, managing each of the partitioned sets independently. It's not easy to achieve an optimal partitioning, so there are several techniques to be applied in different situations:

- **Vertical** - Split the data in attribute groups and replicate the primary key. Attributes that are usually accessed together should be in the same set. This kind of partitioning is used more in column oriented databases.

- **Horizontal** - Split the data in groups of tuples, according to some special condition. There's three types of horizontal partitioning:

  - **Range** - Split according a partitioning key, which determines how the data is split between partitions.
  - **Round-Robin** - A fast and reliable way of dividing the tuples, by simply dividing all tuples evenly between partitions.
  - **Hash-Based**- Uses a hash function to assign each row to partitions. Like every other hash algorithms, relies on how good is the hash function.
  - **Semantic** - Separate the tuples according to specific application specifications, like different exclusive processes that have few accesses to other partitions.

To choose the right partitioning technique it's mandatory to have a deep knowledge of the application and how data is organized and if processes have a need to access data across all partitions. [PZ12]

A.2 Architecture's Particularities

A.2.1 Differential Buffer

Inserting, updating and deleting data from the DB is expensive, involving dictionary sorting and attribute vector reordering. To overcome this problem, a differential buffer was introduced.

This concept divided the database into a main store and a buffer, where all DB operations are performed. The main store, where only read operations are performed, is not changed. However, read operations have to be performed in both the buffer and the main store, as all data from the database is a join between the old data and the new inserted data in the buffer. The buffer is much smaller than the main store, so the overhead is smaller and has less impact in performance. After each day, or another amount of time, a merger of the old main store with the differential buffer occurs, into a new main store, while a new buffer receives the new entries.

With this architecture, insertions can be much faster, because there's no need to resort the main dictionary every time a new value is inserted. It consumes more memory, but the benefits outweigh this disadvantage, due to checking multiple positions for validity with SIMD instructions. Another problem we may think is the update of values in the main store. This is solved using bit validators, a vector, that remains uncompressed, with a bit saying if a tuple is still valid or not. After updating values, this bit is changed, and the new entry is now the valid one. [PZ12]

A.2.2 Aggregate Cache

With the arrival of columnar databases, now is possible to get rid of predefined aggregates and is possible to do OLAP queries directly, on-the-fly, on top of OLTP data, and the system can handle mixed workloads. However, doing this aggregates directly on the transactional data, can slow down the system for the overall users. When a lot of analytical queries arrive at the same time, Hana already makes use of its high parallelization and uses the machine at his potential, thus the workload of several analytical queries can be too high. To scale the system, is used an aggregate cache, in the delta-architecture, by applying incremental materialized view maintenance and query compensation techniques. This cache builds its aggregates using only the main store, thus not being invalidated when new data is inserted in the differential buffer. This is compensated with on-the-fly aggregates over the delta partition (differential buffer), leaving the system with much less workload. [Voo]

The predefined aggregates are unique combinations of each table id, grouping attributes and filter the predicates. The differences caused by updates and deletes are also handled using bit vectors. Handling loaded servers with this approach of an aggregate buffer can improve in a factor of ten, for OLAP queries. [PZ12]

A.2.3 Replication

It is known that OLAP queries are much more complex than OLTP queries, so it's natural if they consume more resources, having the majority of the workload. SAP, who developed Hana to hold mixed workload, have come with a solution, where they replicate data in two separate systems, one to hold the read-only queries and other the inserts.

These replicas, which only handle read-only workload, have a low delay of data transfer with the master, so the snapshot of the primary database is consistent. Once they're connected, all logged changes are replicated continuously, where each persistent log in the primary system is sent to the secondary. [PZ12]

Appendix B

Hana

B.1 Insert,Update and Delete Queries

1	Insert Record in the Exportation table, without dictionary re-writing
2	Insert Record in the Exportation table, with dictionary re-writing
3	Insert Record in the takeoffs table, without dictionary re-writing
4	Insert Record in the takeoffs table, with dictionary re-writing

Table B.1: Insert statements to be tested

1	Update one Record in the Exportation table
2	Update batch of Records in the Exportation table
3	Update Record in the takeoffs table, without dictionary re-writing
4	Update Record in the takeoffs table, with dictionary re-writing

Table B.2: Update statements to be tested

1	Delete Record in the Exportation table, without dictionary re-writing
2	Delete Record in the Exportation table, with dictionary re-writing
3	Delete Record in the takeoffs table, without dictionary re-writing
5	Delete Record in the takeoffs table, with dictionary re-writing

Table B.3: Delete statements to be tested

B.2 Test Results

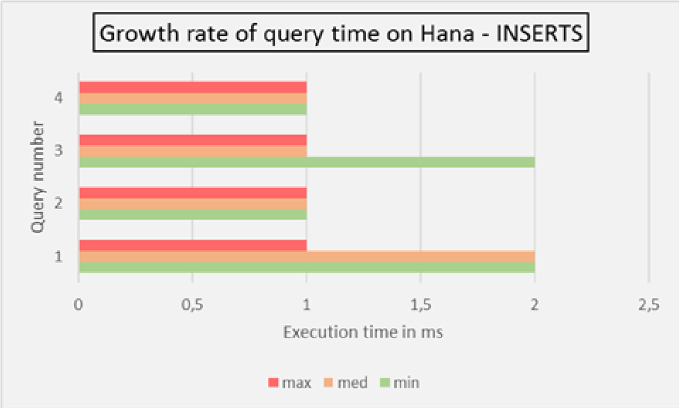


Figure B.1: Hana inserts results

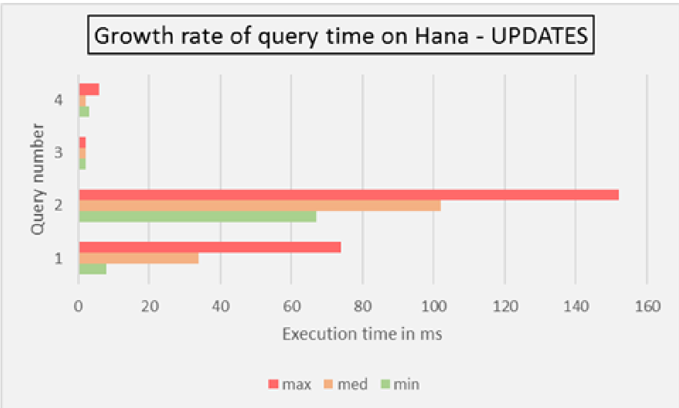


Figure B.2: Hana updates results

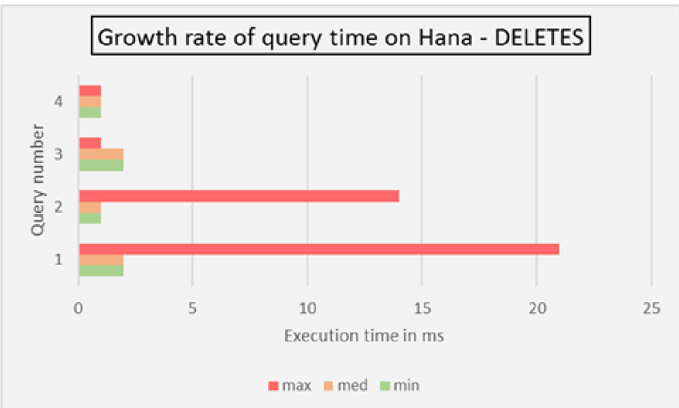


Figure B.3: Hana deletes results

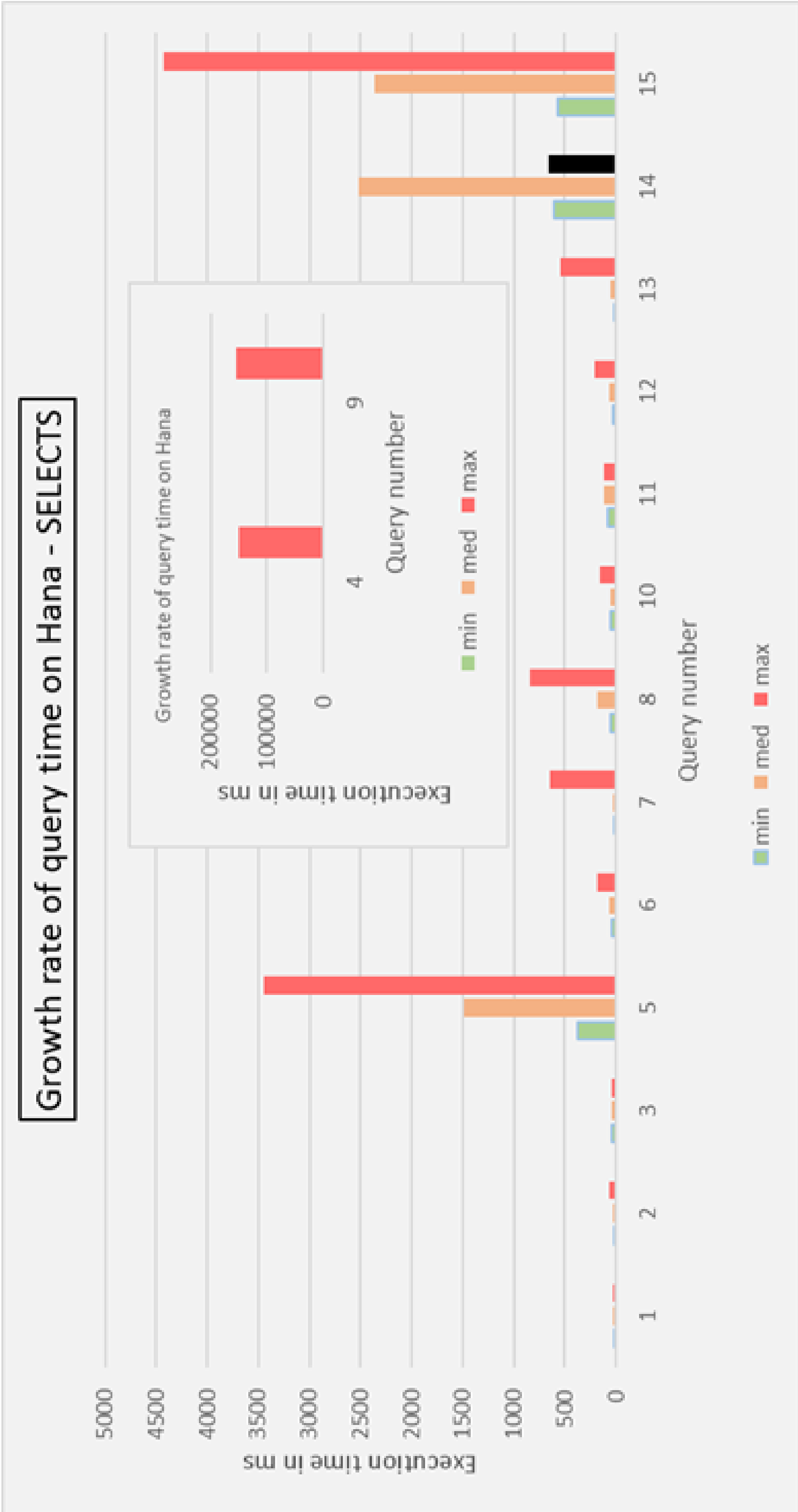


Figure B.4: Hana Selects results

query nº	min. load	med. load	max. load
1	5,2	25,25	28,8
2	7,2	27,25	60,4
3	43,2	42,8	40
4	25	86,5	150854,25
5	377,2	1487,75	3450,8
6	42,75	66	179,5
7	13	27,25	639
8	53	181,5	835,2
9	88,25	353,8	155549,2
10	44,8	55,2	148
11	74,25	115,75	113,8
12	20,6	58	202
13	17,4	47,25	547,4
14	607	2514,2	659,6
15	570	2366,6	4425,25

Table B.4: Execution times of analytical queries with 3 different loads on Hana

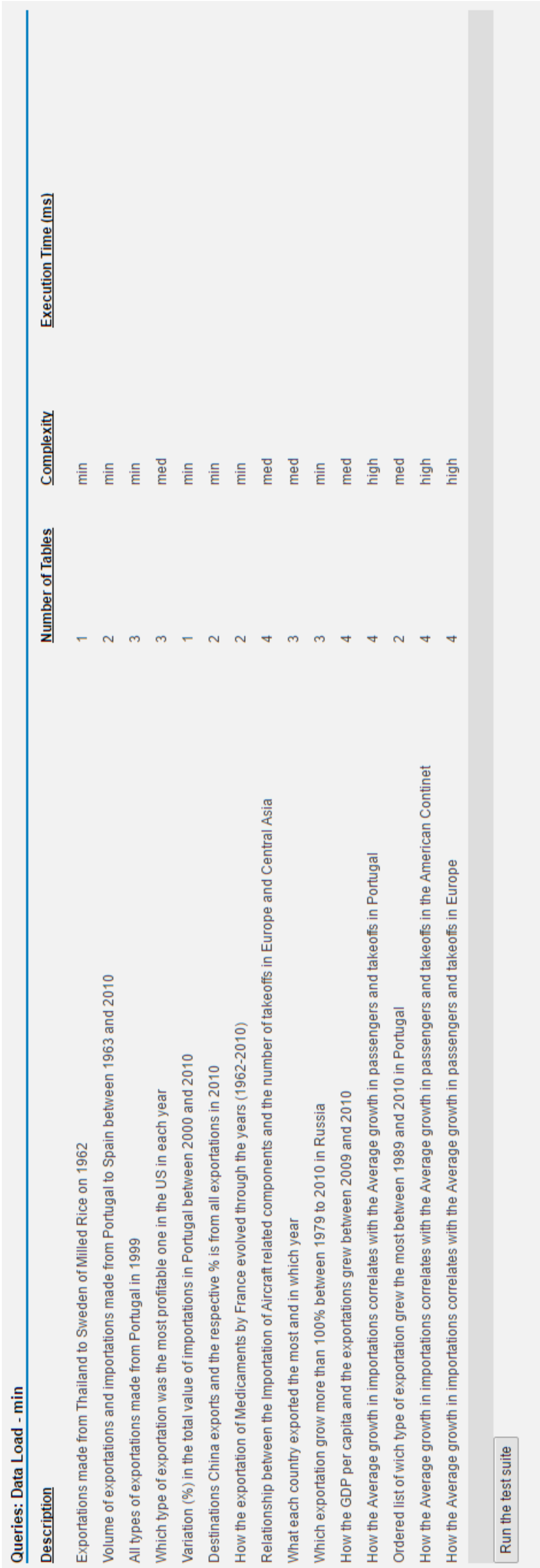


Figure B.5: User Interface Hana test App

Appendix C

2 Result Tables - R/3

1	Insert Record in the Exportation table, without dictionary re-writing	2456
2	Insert,Record in the Exportation table, with dictionary re-writing	2144
3	Insert,Record in the takeoffs table, without dictionary re-writing	45
4	Insert,Record in the takeoffs table, with dictionary re-writing	47

Table C.1: Execution time of Insert queries - minimum capacity

1	Update 1 Record in the Exportation table	4808
2	Update batch of Recors in the Exportation table	5572
3	Update Record in the takeoffs table, without dictionary re-writing	855
4	Update Record in the takeoffs table, with dictionary re-writing	972

Table C.2: Execution time of Update queries - minimum capacity

1	Delete Record in the Exportation table, without dictionary re-writing	4683
2	Delete Record in the Exportation table, with dictionary re-writing	4569
3	Delete Record in the takeoffs table, without dictionary re-writing	57
4	Delete Record in the takeoffs table, with dictionary re-writing	49

Table C.3: Execution time of Delete queries - minimum capacity

Analytical Query number	min. load	med. load	max. load
1	4794	83857	
2	30632,5	141458,6	
3	219614,6	142751,4	
4	1042514,6	795893,6	
5	184598,6	167653,8	
6	18354,75	59878,8	
7	12933,4	55750,6	
8	599949,8	1325376,8	
9			
10	9957	1228451,4	
11	866988	209915	
12	14822,2	60400	
13	10163,4	36973,5	
14			
15			

Table C.4: Execution times of analytical queries with 3 different loads on R/3

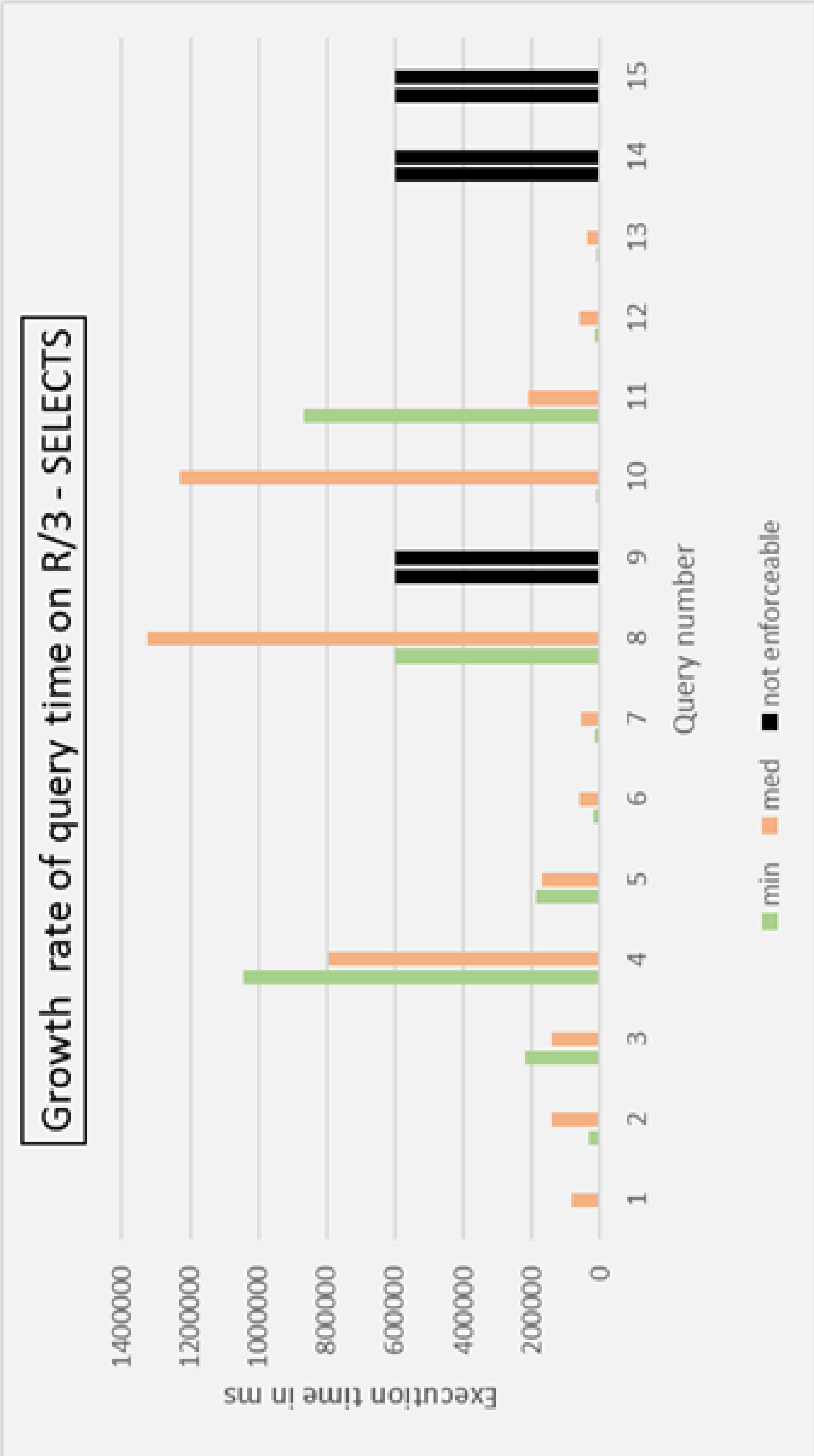


Figure C.1: R/3 Test Results