FAKULTÄT FÜR !NFORMATIK
Faculty of Informatics

Covering Performance Issues in a Conceptual Framework for Database Migration

DIPLOMARBEIT

zur Erlangung des akademischen Grades

Diplom-Ingenieur

im Rahmen des Masterstudiums

Wirtschaftsinformatik
eingereicht von

Rainer Mücke, Bakk
Matrikelnummer 0207270

an der
Fakultät für Informatik der Technischen Universität Wien

Betreuung: Univ.-Prof. Dipl.-Ing. DDr. Quirchmayr Gerald
Mitwirkung: Mag. Koegler Michael

Wien, 10.10.2018 ____________________ ____________________
(Unterschrift Verfasser) (Unterschrift Betreuung)

Erklärung zur Verfassung der Arbeit

Rainer Mücke, Bakk.
Mohngasse 11, A-3100 Sankt Pölten

Hiermit erkläre ich, dass ich diese Arbeit selbstständig verfasst habe, dass ich die verwendeten Quellen und Hilfsmittel vollständig angegeben habe und dass ich die Stellen der Arbeit (einschließlich Tabellen und Abbildungen), die anderen Werke oder dem Internet im Wortlaut oder dem Sinn nach entnommen sind, auf jeden Fall unter Angabe der Quelle als Entlehnung kenntlich gemacht habe.

Wien, 10.10.2018 ____________________________________

(Unterschrift Verfasser)

Abstract

Objective of this master thesis is the creation of a framework to support a complete database migration. As part of the work to next to the creation of the framework, the major topic of the master thesis shows the possibilities of virtualization and performance analysis in the field of database server.

The trigger for the project described in this paper is the increasing demand for virtualization of IT services, service-orientation of architecture, increased reliability and finally yet importantly cost reduction.

The Migration Framework supports the database migration system and product independently in the phases Requirements Modelling, design migration, and implementation. To develop therefore model, method and tool support. By industry partner predetermined was used standard UML as a modeling language.

Database Migration means the process to lead over structured data to a new database system. In this master thesis it will be shown what is needed for a complete database migration in an ITIL environment. The developed, evaluated, tested and improved framework will support every database migration independent of the infrastructure system and database system and application.

Secondary analysis shows the need for virtualization and the importance of performance analysis. Application requirements, which in turn are subject to business processes, put restrictions for system environments and databases. A detailed technical evaluation, in conjunction with in-depth knowledge about system and databases is necessary. To a relational database, system in the field to optimize performance and semantics requires cooperation with application programmers. Communication plays a major role in the creation of the concept for data migration to be sure to have treated each request. The implementation of the database migration takes place first in the test system. Upon completion and documentation of all tests, the database migration has green light to perform under time and content compliance of the project plan. Self-developed database tools are the main tool support. By integrating the new database in the escalation chain of the monitoring system, the database becomes the productive status.

Kurzfassung

Aufgabenstellung dieser Master Thesis ist die Erstellung eines Frameworks zur Unterstützung der Datenbankmigration. Im Rahmen der Arbeit sollen neben der Erstellung des Frameworks hauptsächlich die Möglichkeiten der Virtualisierung und Performanceanalyse im Bereich der Datenbank-Server untersucht werden.

Auslöser für das in dieser Arbeit beschriebene Projekt ist der zunehmende Bedarf an Virtualisierung von IT-Services, Service-Orientierung der Architektur, erhöhte Ausfallsicherheit und nicht zuletzt auch Kostensenkung.

Das zu erstellende Migrationsframework setzt sich zum Ziel, die Datenbankmigration system- und produktunabhängig in den Phasen Requirements Modeling, Konzeption der Migration, sowie Umsetzung zu unterstützen. Zu entwickeln sind daher im Rahmen des vorgesehenen Frameworks Modell, Methode (Vorgehensweise), und Tool-Unterstützung. Als Modellierungssprache wird vom Industriepartner vorgegebene Standard UML verwendet.

Applikationsanforderungen welche wiederum Geschäftsprozessen unterliegen setzen Rahmenbedingungen und Einschränkungen für Systemumgebungen und Datenbanken. Eine ausführliche technische Evaluierung ermöglicht in Verbindung mit fundiertem fachlichem Detailwissen eine optimale Datenhaltung. Um ein relationales Datenbanksystem im Bereich Performance und Semantik zu optimieren bedarf es der Zusammenarbeit mit Applikationsprogrammierern. Kommunikation spielt eine große Rolle bei der Erstellung des Konzepts zur Datenmigration um sicher zu gehen jede Anforderung behandelt zu haben. Die Umsetzung der Datenbankmigration erfolgt zuerst im Testsystem. Nach Abschluss und Dokumentation aller Tests wird die Datenbankmigration unter zeitlicher und inhaltlicher Einhaltung des Projektplans durchgeführt. Als Toolunterstützung kommen selbst entwickelte für die jeweilige Datenbank angepasste Werkzeuge zum Einsatz. Durch Integration der neuen Systeme in die Eskalationskette des Alarmsystems erreichen alle Datenbanken den produktiven Status.

Contents

1 Motivation and problem definition

Database migration is a very complex topic and places high demands on the database administrator. Each database migration is implemented differently but the approach for realization seems to be almost the same. There are still a lot of tools on the market which supports some database migration scenarios, but there is no tool that covers all possible database migration scenarios. In literature and on research there were found very less approaches for a database migration framework.

So the idea was born to formulate a conceptual framework which supports each direction of database migration, or the complete database migration so to speak. The developed and described conceptual framework in this thesis covers each direction of database migration scenario in an ITIL world independent from the IT infrastructure systems and database system.
The goal of the conceptual framework is to guarantee a complete database migration from the idea or the trigger for migration to the running state of the database in the production environment.

Albeit the process to lead over structured data from one database system to another database system sounds identical in every case, but the practical implementation is always different. Every database and its environment must be treated differently. This master thesis attempts to answer the question what is needed for a complete database migration.

Database Migration means the process to lead over structured data to a new database system. This master thesis shows what course of action is necessary for a complete database migration in an ITIL environment. Done by a developed, evaluated, tested and improved framework with the goal to support every database migration independent of the infrastructure system and database system as well as application.

Secondary, but major analysis shows the need for virtualization and the importance of **performance analysis** in the field of database systems and database engines.

2 State of the art

2.1 Literature review method and literature search

For the literature research the following keywords were searched:

- Database migration framework
- Data migration framework
- Database migration
- Data migration

There exist several migration approaches and methods most bonded on one platform or bonded on a single database. There are many tools out there restricted in terms of functionality and covering problems.

ITIL driven IT service management environment which is a living environment, is a good place for a database migration conceptual framework.

IT infrastructure is not a static structure. IT-Systems itself have a life cycle and in addition technology changes. In addition, the business process is a technology driver. Every application underlies a business process. Moreover, every database underlies an application. Therefore, there is a hierarchy representing the importance of every application and its database.

The research topic of database migration has received a lot of literature across database migration and data migration. A database and its interaction with the associated application can be very complex. With increased complexity in concern of the application or the system environment, the whole migration process will become more expensive. Because of the fact that a database can look very different compared to another database, it seems to be impossible that there exist one tool covering every way of database migration. A database can use a bunch of functions. A framework for database migration understood as a structured guide with a complete collection of tools to cover all variants of requirements fits all needs to ensure a complete and successful database migration were not found.

2.2 Migration concepts

Like mentioned in section 1 database migration can vary in several ways. In literature a lot of migration approaches and a lot of database deployment approaches are found but all are very special to a specific topic or task. In application development there are two well-known approaches.

The state-based method will synchronize two database states in one step. Every table and every script is versioned and allows the application developer to deploy a modification directly in the database. Tools used to migrate a database state from one to another auto-generate a script after comparing the source to the target and synchronize the two states.

State: Your source of truth is how the database should ***be*** [15]

The migration-based method aims to capture lots of small and individual changes to generate multiple small scripts for an iterative database migration. Tools that migrate the database only need to apply each script in the correct designated order. A kind of tool can be a list of alter statements in the correct order processed by the database administrator.

Migrations: Your source of truth is how the database should ***change*** [15]

Redgate, a vendor of serious and useful tools is well known for their industry standard products used by 71% of known companies. A muck of tools like **SQL Compare, SQL Source Control, SQL Data Compare and SQL Toolbelt** are very useful and accepted.

Claudera, Inc. Cloudera Data Warehouse or also called Claudera Enterprise is a new type of data platform based on Apache Hadoop which holds the data and provides access to data with multiple platforms. With hybrid cloud solution designed for self-service analytics it meets the increased scale and analytics demand that growing data warehouses are experiencing. Claudera offers a set of tools to migrate data.

MongoDB, is a document based database. Other than in relational database engines, data is stored in flexible JSON-like documents. The meaning of fields can vary from document to document and data structure can be changed over time. Document objects are mapped to application objects. Classic stored procedures, like in relational databases, with thousand lines of code holding specific intelligence and logic of application makes it near impossible to maintenance. MongoDB do not store application code in the database. Developers are forced to keep the logic in the application, which makes applications more flexible. In terms of big data, a MongoDB will support agile application developing and promotes its document based data holding as one of the fastest ways to store and provide huge data lakes.

2.3 ITIL

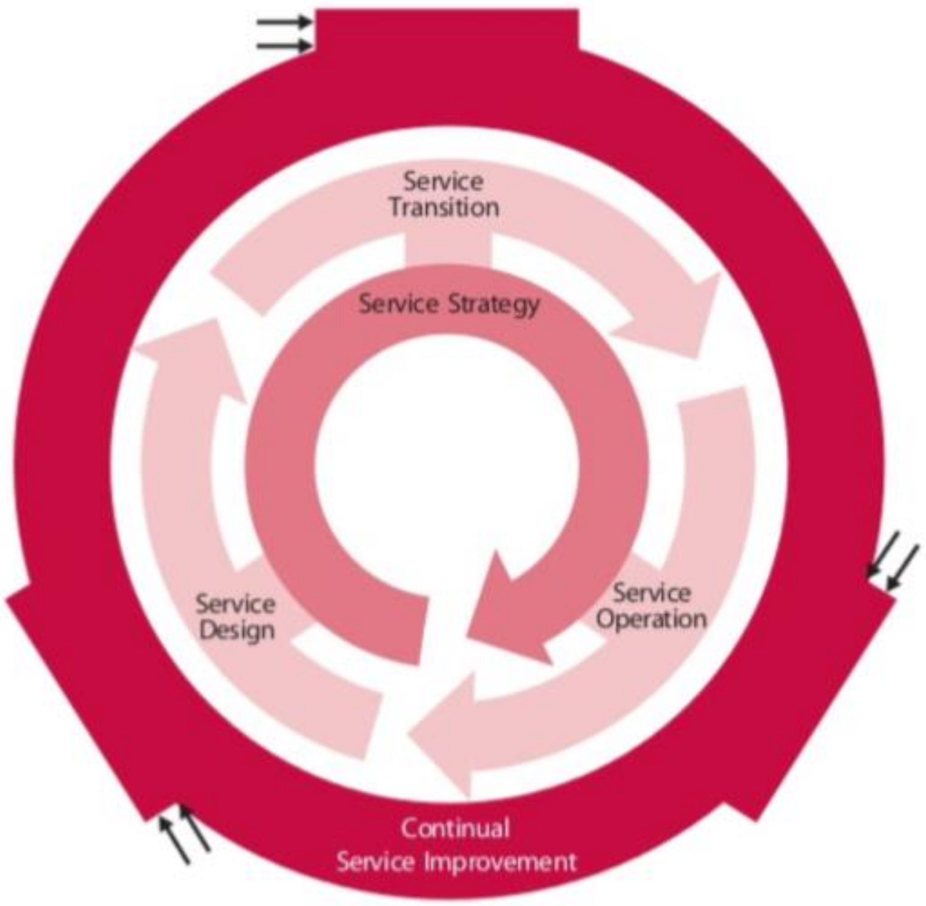


***Figure 1**: ITIL life cycle [14]*

In this thesis, the presented framework developed in a big IT department over a couple of years. This IT department is the digital groundwork for a corporation with more than 7000 employees. ITIL (Information Technology Infrastructure Library) is a set of practices for ITSM (IT Service Management) that focuses on aligning IT services with the needs of business [14]. Business processes formulate the need for IT services. An SLA (Service Level Agreement) specifies the importance of availability and the disaster recovery scenarios. The availability of the support personnel during and beyond the business hours mostly defined from eight to 16 o'clock. The database support in general is always 24/7 guaranteed. Therefore, a team of database administrators is working on standby in shift work. Such a big IT department, separated into divisions, underlies an evolving change process.

2.3.1 IT department / divisions

- Infrastructure WINTEL (Microsoft)
- Infrastructure UNIX (Linux)
- Database services
- Network services
- Storage systems
- Automation services
- End user computing
- Service management / Helpdesk

Every division is part of the whole IT department and has its own professional core competence managed and controlled by a team leader.

The division **service management** operates the helpdesk and the monitoring system. Additional acts the division service management as a supervisor and mediator between all the other divisions in the IT department and the corporate department. The divisions of the IT operations offering services to all other departments and are so called infrastructure.

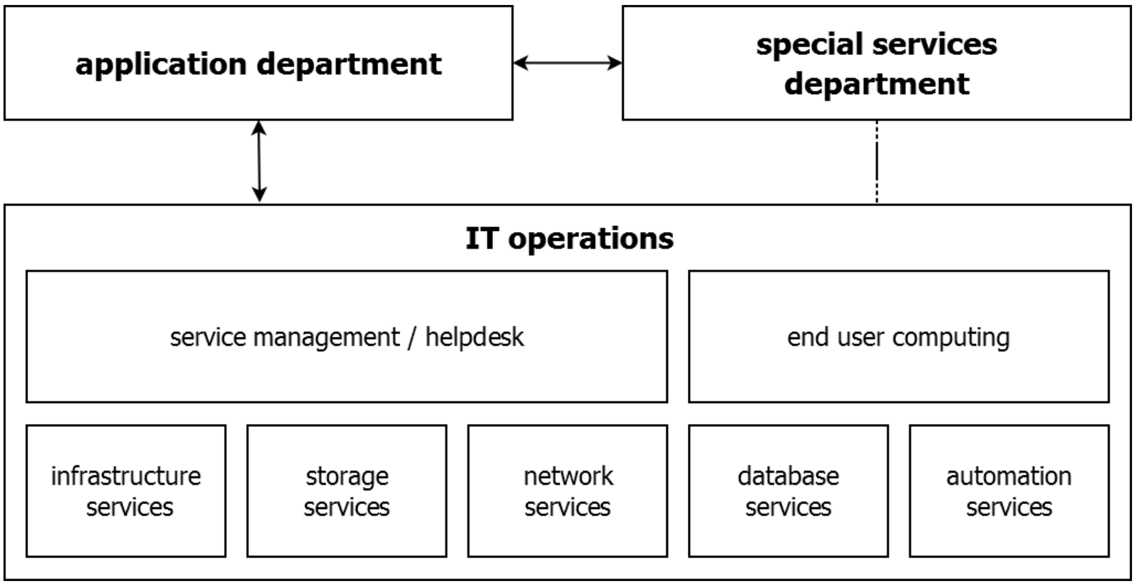


***Figure 2**: ITIL organization structure daily in touch*

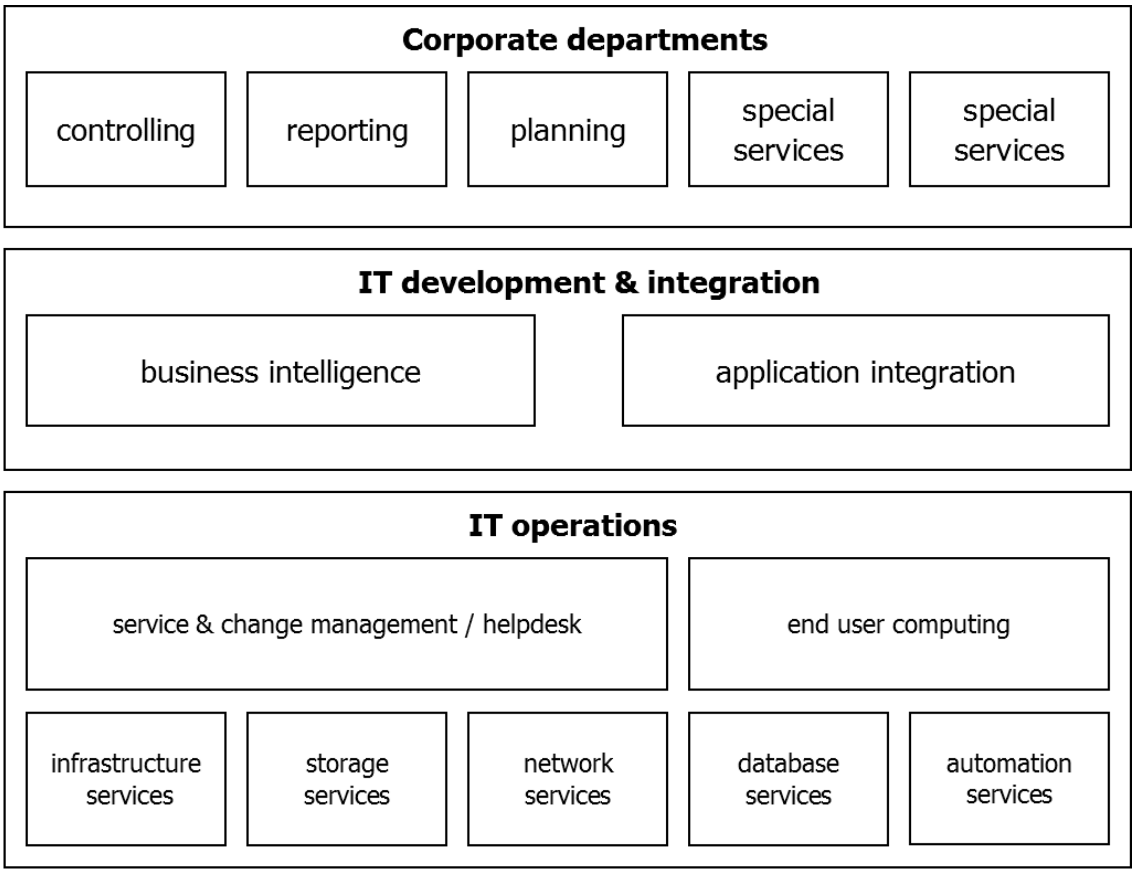


***Figure 3**: ITIL organization structure corporate level*

In the context of security, only every team member of a division has full rights for data access and other services. This makes a working relationship between team members across departmental boundaries necessary.

The **infrastructure WINTEL (Microsoft)** division operates all the Microsoft Windows servers, VMWare ESX Hosts servers, file transfer protocol servers and the backup servers.

The **storage services** division guarantees the availability of all persistent data stored in big storage systems. In case of high availability depending of the service-level agreement, some data have to be stored location independent and mirrored.

2.3.2 Monitoring system

A service helpdesk well established in many corporations over decades, is the first reception center that recording every customer request and try to solve the problem immediately or assign the problem to the responsible standby on duty. If computer systems and services available 24/7, it is necessary that the helpdesk is also available round the clock.

Modern helpdesks have a complex monitoring system, which makes the standby of the helpdesk obsolete, and save therefore money. Every authorized user has the possibility to determine a disturbance with status critical for business causing an alarm. The monitoring system collecting data from important systems and observe important systems with end-to-end monitoring nonstop. A sensitive alarm escalation chain will take care about the assignment of an alarm to the right person on standby. This person gets a call on his mobile phone. The person must authenticate and assign the alarm. So the escalation process is stopped and the monitoring system knows that a person takes charge of the problem. Additional the person has to acknowledge the incident in the monitoring system. If the problem could be resolved a short description and problem solving information is required to close the incident and stop the escalation process. If there is a bigger problem, which cannot be resolved immediately and without help of other experts, a short description added to the journal for status documentation. External help, ordered immediately in the form of external support or help from another division standby to solve the problem as fast as possible.

A real world example of an excerpt of a document from the change management division in which the responsible persons, defined and bound on weekly reports.

Funktion	Detailverantwortungen	Name	Organisationseinheit	Telefon
Projektmanager	Projektplanung (Projektplan-Erstellung und Projektsteuerung), Rekrutierung von Personal, Monatsberichte, Ansprechpartner, ...		Infrastruktur und Datenbankbetrieb	
Betriebssystem & Software	Betriebssystem, Clustering und Backups		Infrastruktur WINTEL, Infrastruktur Unix/Linux Oracle	
Hardware & System	Bestellung von HW und SW, Bladeverwaltung und Firmwareservices			
Applikations-verantwortliche	Applikationstests ...		Fachdienst, Entwicklung und Datawarehouse	

Figure 4*: ITIL change management responsible persons*

2.4 Virtualization

Virtualization is an abstraction layer that decollates the physical hardware from the operating system with the major aim to provide more computer resources more flexible and efficiently.

2.4.1 Virtual machines

Server hardware environments consists of several server computer and each server has an operating system and need to be explicit managed in terms of redundant power supply, redundant network link-up, operating system, its security updates and firmware updates and so on.
Virtual infrastructure environments replacing common cluster based server infrastructure systems. A powerful computer also called host computer is providing multiple software server instances managed by the hypervisor software. Virtual machines, short VMs, are an abstraction of physical hardware. A virtual machine is the representation of physical hardware by software. Virtual server environments are cheaper than physical server environments. Virtual environments offer many advantages like cost reduction and better scaling. Processing and computing power can be provided and used more efficiently. System enlargements or other changes can be done using a hotfix without any downtime instead of time intensive defective structural alternation works. Operating costs, hardware costs and licensing costs can be reduced because of the better prospects of hardware consolidation.

2.4.2 Containers

Instead of virtual machines, containers are an abstraction at the application layer combining the code and dependencies. Multiple containers share the operation system kernel and running as isolated processes. Containers used together in virtual machines provide maximal flexibility and portability in managing and deploying an application. The big advantage is the small operating system kernel instead of the whole operating system which is necessary in a virtual machine.

2.5 Related work

In several works, data migration treated similar, but the core is always to lead over data from a source system to a destination system. Many works explain the advantages and necessity of migrating databases from legacy systems to relational open source systems. The data quality is a big issue. Some heterogeneous data on legacy systems distributed, duplicative or multiple present. Tools are able to lead over the historical grown and not documented legacy data to a relational data set. Duplicate data records merged to one record. In all cases of database migration different tools needed to relocate the data. A tool can be standard software, a special written software tool or as in most cases a bunch of scripts like T-SQL scripts or for instance Microsoft SSIS Integration Services Packages.

In literature exists frameworks well defined for special purposes. A complete database migration neither in an ITIL environment nor in another complex environment was not found. Any tool or framework found focused on a specific topic. Nevertheless, no tool can handle every sort of migration scenario. In addition, no tool or guidance is able to formulate all tasks and processes what were necessary for a complete data migration.

It exists a schema evolution workbench called PRISM discussed in [6]. This workbench assists the DBA in the design process of evolution steps with the concise SMO language used to formulate schema changes.

An automated database migration evaluation, called MEET DB2, will be discussed and tested in [3]. MEET DB2 consists of the following relatively independent components like preprocessor, parser, analyzer, knowledge base, effort estimator and report generator linked together by step outputs in XML format. Goal of MEET DB2 is dramatically reducing the time required for a database migration evaluation analysis.
Technical sales stuff with only reasonable knowledge should be able to use MEET DB2 and interpret the output. The results will be compared in terms of accuracy, with the results of an expert.

Evaluation tests have shown that MEET DB2 has shown 97.6% reduction in analysis time and 48% improvement in problem identification compared to a human expert.

In the following, all known and found migration methods described. This knowledge is the fundament for a framework.

2.6 Migration methods overview

ETL versus ELT. In most cases of data migration, it is necessary to transform the data before loading it into the new system. **ETL** means extract, transform and load. The data will be extracted then it will be transformed to meet the new schema description in the new database system and then it will be loaded into the new database system. This process is a traditionally one-way action and can be understood like a pipeline. Data from the source flow through a transformation engine to the target. **ELT** means extract, load and transform. Data will be loaded to the target and then transformed in place. This is a newer approach, needs high-end database server, and engines with much more performance.

Main Memory database to support database migration is a strategy for fast high speed migration scenarios. It is necessary to hold actual backups at hand. In case of crash the backup, undo log files and redo log files needed to restore the database. A main memory database for supporting database migration need special recovery methods described in [1]. One physical limitation can be the main memory limit of a server hardware. In case of migration of one table of a big data warehouse with one terabyte of data there will be other migration methods necessary than the main memory method.

Data migrated beforehand. In some cases, particularly data in form of one or some tables will be migrated by tools beforehand and on the other hand it is only possible to migrate such particularly data manually beforehand.

Data generated by new system afterwards. The schema of the new database will be modified and does not match with the schema of the source database. The migration of the data can be realized with an ETL process or a transformation script.

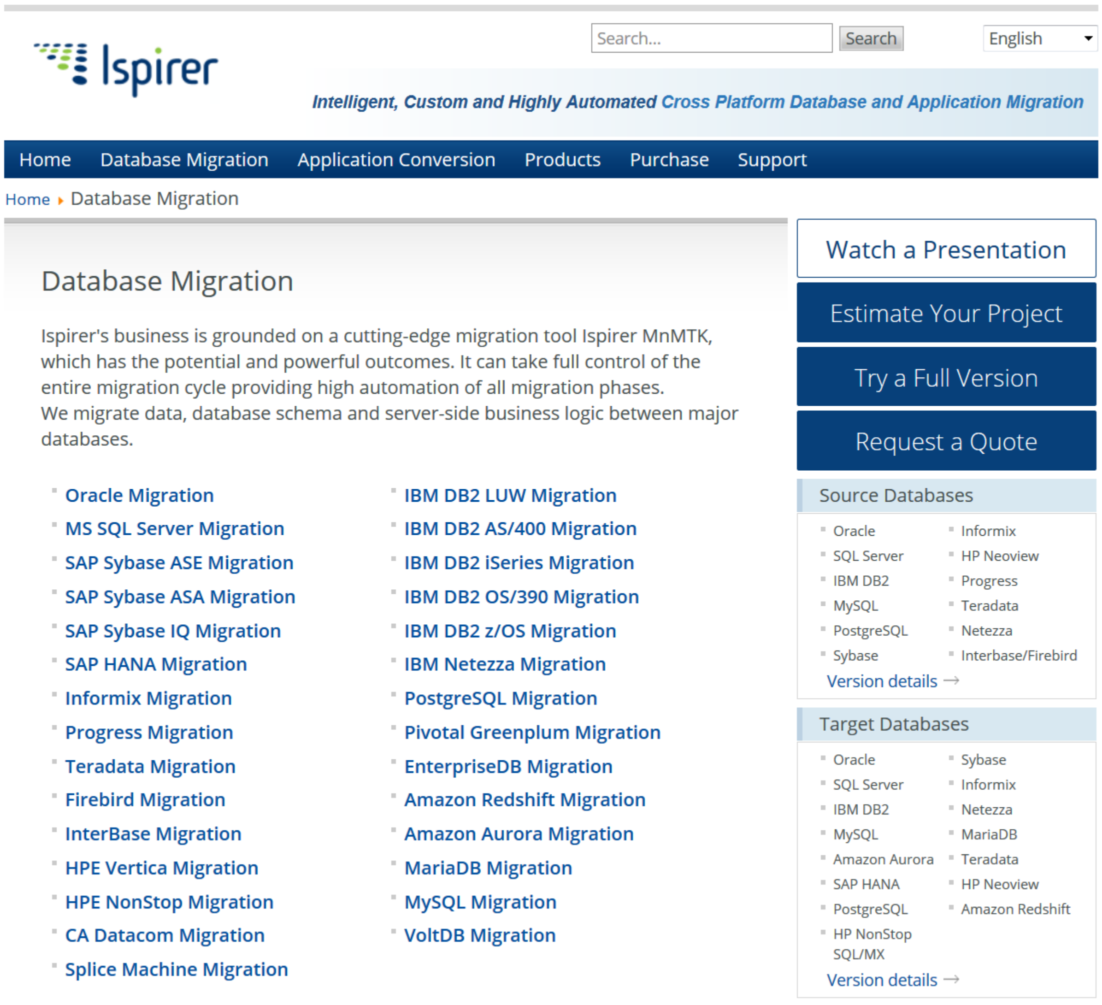
2.7 Software Tools

The technical implementation of a database system also the data transfer requires the use of software tools. This can be a single script, a bunch of multiple scripts or a series of programs and tools. While research a lot of tools were found but to care about not to go beyond the scope only one software tool will be specified.

Ispirer.com offers a so called cutting-edge migration tool Ispirer MnMTK, which can take full control of the entire migration cycle providing high automation of all migration phases. They migrate data, database schema and server-side business logic between major databases.

Ispirer SQLWays Database Migration Software.

The software offers many detailed descriptions to match data types, functions and expressions, reserved words and version differences independent of which database engine.



Database Migration

Ispirer's business is grounded on a cutting-edge migration tool Ispirer MnMTK, which has the potential and powerful outcomes. It can take full control of the entire migration cycle providing high automation of all migration phases.
We migrate data, database schema and server-side business logic between major databases.

- Oracle Migration
- MS SQL Server Migration
- SAP Sybase ASE Migration
- SAP Sybase ASA Migration
- SAP Sybase IQ Migration
- SAP HANA Migration
- Informix Migration
- Progress Migration
- Teradata Migration
- Firebird Migration
- InterBase Migration
- HPE Vertica Migration
- HPE NonStop Migration
- CA Datacom Migration
- Splice Machine Migration
- IBM DB2 LUW Migration
- IBM DB2 AS/400 Migration
- IBM DB2 iSeries Migration
- IBM DB2 OS/390 Migration
- IBM DB2 z/OS Migration
- IBM Netezza Migration
- PostgreSQL Migration
- Pivotal Greenplum Migration
- EnterpriseDB Migration
- Amazon Redshift Migration
- Amazon Aurora Migration
- MariaDB Migration
- MySQL Migration
- VoltDB Migration

Source Databases

- Oracle
- SQL Server
- IBM DB2
- MySQL
- PostgreSQL
- Sybase
- Informix
- HP Neoview
- Progress
- Teradata
- Netezza
- Interbase/Firebird

Version details →

Target Databases

- Oracle
- SQL Server
- IBM DB2
- MySQL
- Amazon Aurora
- SAP HANA
- PostgreSQL
- HP NonStop SQL/MX
- Sybase
- Informix
- Netezza
- MariaDB
- Teradata
- HP Neoview
- Amazon Redshift

Version details →

***Figure** 5: Ispirer Database Migration Tools*

This product volume is a serious and extensive offer and in most cases very helpful.
Most real life scenarios can bear up problems that need tools they are more specific.
Such tools generated manually are in depth T-SQL scripts and SSIS Packages only for a specific task.

2.8 The need for change

First, it is necessary to evaluate all sort of trigger for a database migration. What driving factors leads to a database migration. There a two main driver from the technical aspect in an ITIL environment called application and infrastructure system environment.

2.8.1 Applications

Small changes in applications such as a bug-fix, hotfix or new features in application take change over after all tests and is realized in a short time.

Big changes in applications such as extensions will also be deployed in the database after implementation in the application and after a successful test phase.
Some changes do not have any risky impact such as increasing a database column named address from size 20 to size 50 because one address in a data row exceeded the size 20. In most databases this will be implemented with a registered hotfix in only a few minutes with no impact to the availability of the database. On the other hand, if the same change is needed in a very big data warehouse this could lead to a big challenge. In a big data warehouse database administrator often have to deal with tables holding a few billion rows. Such a big table could not be handled like a small or normal table. Even though table partitioning is in use for better handling it will be the fastest way to delete all data in the table. Then one can change the table structure and the table can be filled with the saved data. After that, index rebuilding will be necessary, and take also a lot of time. One can see that the same change on several occasions can have completely different influence to daily business in database operations. Therefore, it is always mandatory to give the testing phase a very high priority.

Application changes are triggered by functional needs, optimizations or error correction or by upgrading to a higher version of the database engine. Usually in case of upgrading the database engine there is no need to do any changes by the application code. Only if it brings an advantage to use new features of the database engine offered by new transactional SQL code features. If there are changes in SQL, code execution planning this could lead to optimize some queries for the application. However, a change in the application not always includes a change in the database at all.

2.8.2 Infrastructure systems

Trigger for infrastructure changes are support life cycle, server consolidation and the need for virtualization.

2.8.2.1 Support life-cycle

Support life cycle is the period in which the hardware vendor or the software vendor guarantees support. Every operating system and every database system underlies a support life cycle. A support life cycle subdivided in three phases.

End of Mainstream Support. The product has lost the option for request to change product design and features, hardware warranty claims, non-security update support and complimentary support like phone support and online support. For some products, limited complimentary support is available.

End of Extended Support. The product security updates, paid-support and product-specific information that is available online and in knowledge bases.

End of Life means that the vendor supports its own product never more. The product becomes outdated and get vulnerable in terms of security because there are no more security updates coming in future. The product does not fit the requirements of the application vendor and have to be updated. Technical is it possible to use the product but it is not allowed in production systems where full vendor support is mandatory.

2.8.2.2 Server consolidation

Continuous process evaluation and optimization is one of the major challenges in an ITIL infrastructure. Cost reduction is one of the driving factors leading to server consolidation. Another driving factor for server consolidation is the increasing server performance due to improvement of technology.

CPU's will become faster and more efficiently. The number of cores per CPU is rising continuous because of technology improvement and they become more energy-efficient. Increasing amount of CPU level cache improves flexibility and performance.

Cluster organizations like classic failover-cluster become smaller and faster.

Following figure 4 illustrate a failover cluster combination for Microsoft SQL Server version 2000 and version 2005 often used by companies between 2004 and 2010. For cost saving reasons, the cluster for testing purposes had lesser performance than the cluster for production purposes. To spend costs and anyway use the maximum of performance every SQL Server service was running with the maximum maintainable amount of working memory. When a failover occurs, it leads often to heavy swapping. That means that the availability of the SQL Server service was massive delayed because of too less working memory for the SQL server service. In worst cases, the SQL Server service or the whole server is unattainable and need a reboot.

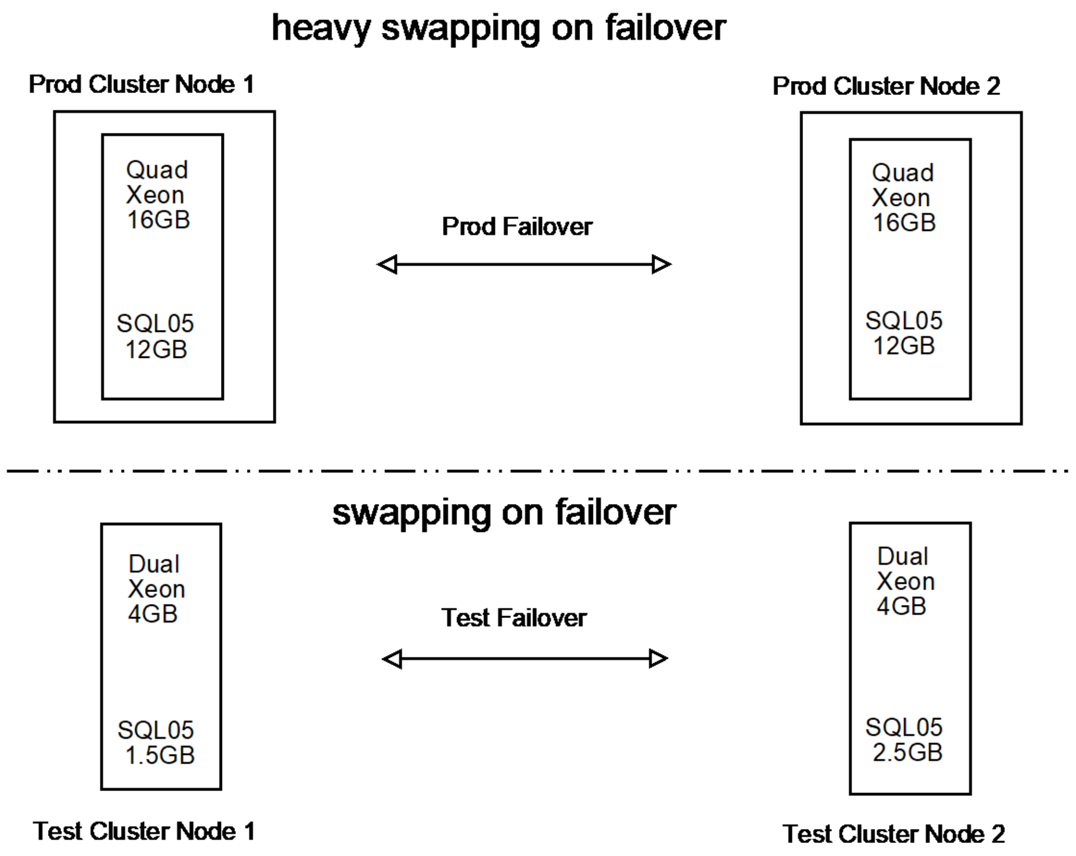


***Figure 6**: Server cluster architecture 2004-2010*

The next figure 5 shows a server cluster architecture for SQL Server 2008/R2. The possibility to operate a four-node cluster will have picked up, against as usual two-node cluster mode. All four nodes have the same performance. Both production server is able to use the maximum memory for the SQL Server service. This is possible when the two server for testing purposes are the failover nodes for the two server for production purposes. To provide the maximum server performance for the production server in the case of a failover a shutdown of the test SQL server service is essential.

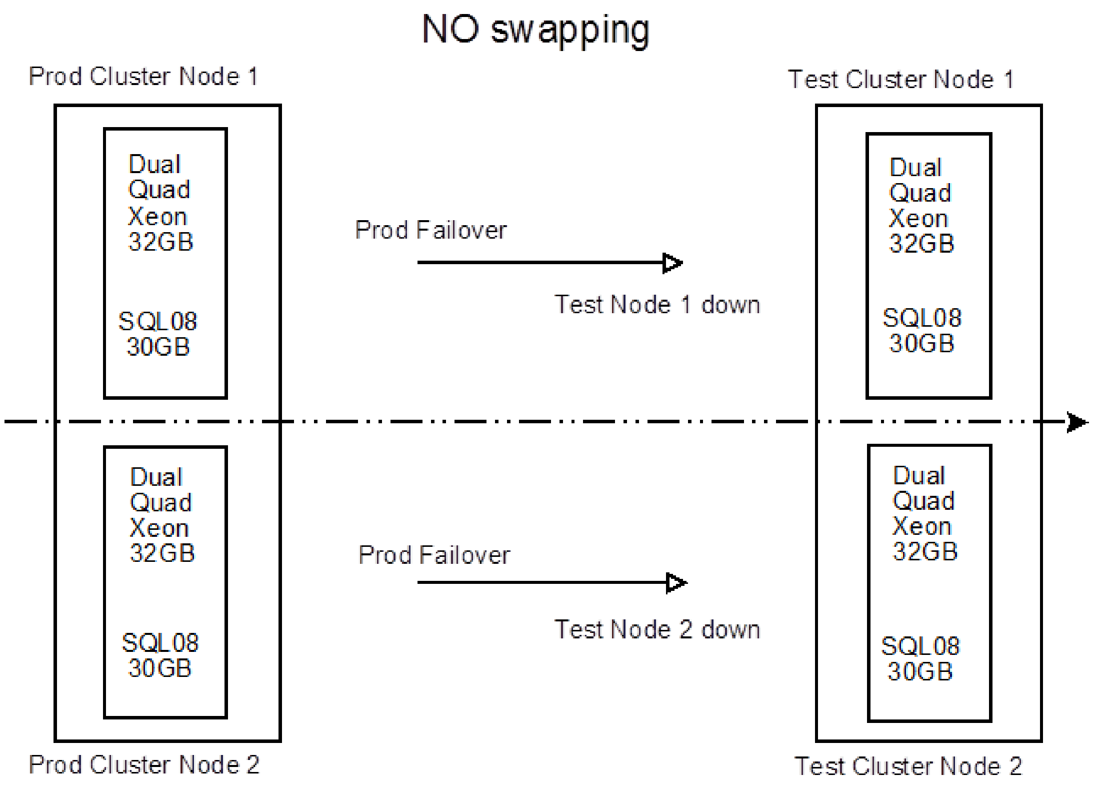


***Figure** 7: Server cluster architecture 2010-2012*

In figure 6 the failover production SQL server service starts on any available test server node. The affected test SQL server service has to failover to the other test server node. Therefore, it is mandatory that both test SQL server services are running always with the half of maximum memory.
This configuration leads additional to a high level of availability because of the location disaster redundancy. Two cluster nodes separated from the other two nodes in different data center.
The right behavior in case of a failover will be guaranteed with cluster scripts. In case of a location disaster like high water, air condition failure or fire the right script brings both production SQL server services online in the neighbor

data center. Therefore, the script takes care for the production memory and brings both test SQL server services down.

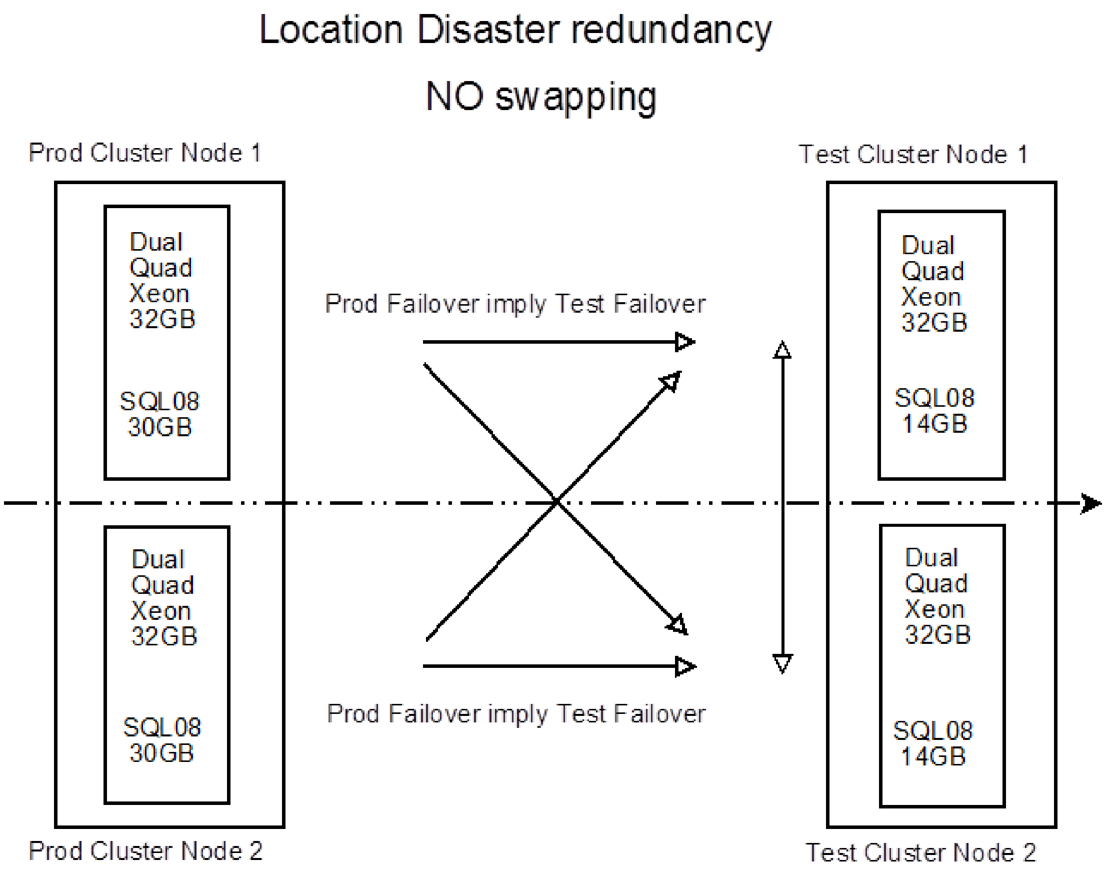


***Figure 8**: Disaster tolerance architecture 2010-2012*

2.8.2.3 Virtualization

A thing called Server related to computer can be hardware or software that offers services. Like a well-known personal computer, a server consists of the same components. The only difference is the focus related to reliability and availability. A server comes with uninterruptible multiple power supplies, multiple disks in an external professional storage system, multiple network adapters which supports load balancing and so on.

A **failover cluster system** is a composite of two or four servers. The idea is to provide important services. In the event of a disruption, the service will start over on another server called failover cluster node immediately.

In a **virtual environment** also called virtual farm a software product providing almost all operating systems on the market runs directly on the server hardware able to host multiple virtual servers. Periodically improved technologies allowing the virtual servers to use hardware more efficiently but there is already a scaling factor, which is very important especially for database servers and for database migrations. More information about this important scaling factor explained in the following chapters.

Benefits. Flexibility and cost reduction are the most benefit of virtual environments. It is possible to install a virtual server with all actual patch levels and pre-installed tools to provide this image for further needed virtual machines only with a copy command. This saves much time and is less error prune. The possibility to move a virtual machine to another server host without interruption makes cluster solutions obsolete. All server components of a virtual machine can be online resized without interruption.

Problems. Scaling is the major problem in virtual environments. Therefore, exists a technical term called overcommitting or overprovisioning. Overcommitting is dangerous for CPUs and also for memory. For example, a physical server has two processors and in daily business, the average usage of both processors in sum is 20%. Leave 10% for the operating system rest 70% processor performance always unused.
When a virtual machine is using 20% processor performance, there is no idle cpu time. The cpu time is administered by the host scheduler. Cost saving leads to overprovisioning. Therefore, it is common that a host with 48 processors supplies 60 virtual machines using in sum 160 CPUs. However, this is false and bring much more problems especially performance problems. In the white papers and best practice guides of any database system are warning line about overcommitting.

2.8.3 Technology

In some cases, it is possible to switch legacy systems or applications and the involved infrastructure to a new technology. Therefore, a detailed evaluation from the business needs down to the infrastructure is mandatory to meet all requirements. A good example is the document based database engine MongoDB. In some cases, relational databases will be replaced by document based databases like MongoDB. If handling of the increasing amount of big data becomes expensive and the application is well documented, it is possible to build the database in a new format like document based database. The most advantage is to extract the application logic from the database and bring it completely to the application because a document based database is not designed to hold and handle application logic.

3 Methodology of the database migration framework

This thesis will examine database migration from all point of views in an ITIL environment such as infrastructure and application. This point of view will already cover all occurrences of database migration scenarios.
Application requirements, which in turn are subject to business processes, put restrictions for system environments and databases. A detailed technical evaluation, in conjunction with in-depth knowledge about system and databases is necessary. To optimize a relational database related to performance issues and semantics, cooperation with application developers is required. Communication plays a major role in the creation of the concept for data migration to be sure to have treated each request. The implementation of the database migration takes place first in the test system if it is possible. In some cases, it is necessary to simulate the production system partial because of a missing test system. Upon completion and documentation of all tests, the database migration happens under time and content compliance of the project plan. Self-developed database tools are the main tool support. By integrating the new database in the escalation chain of the monitoring system, the database became the productive status.

The database migration framework was developed and continuously improved over years in a real world ITIL environment.
It covers all necessary and important items to guarantee the success of a database migration. A database is a collection of information and data. This data managed consistent, persistent and mostly highly available by the database system. Different database systems exist on the market but only a few have the potential to fulfill the high expectations of an enterprise environment. The most common database system for relational data processing and OLAP processing are Oracle, Microsoft SQL Server and Sybase. DB2 is a very professional database legacy system largely replaced by newer and lower priced database systems with lower priced hardware environments.

To keep this thesis manageable further explanations and examples all made about Microsoft SQL Server and Microsoft Windows Server Environments.
A database migration affects primarily only the database and the data but the dependencies in an ITIL environment can be very big and complex. The migration officer has to deal with a lot of addicted components and involved

persons. The database as major component, the application as well as dependencies between multiple applications, the underlying server infrastructure and their sub-components like server, network, storage and operating system. First approach is to consider all components by every database migration. The IT infrastructure evaluated in this thesis satisfies a real world ITIL environment with hundreds of servers and dozen applications. Communication with responsible persons of application is necessary over the whole duration of the migration.

To ensure to treat near every possible known case of database migration and treat near every possible error, the ITIL environment is a perfect environment to improve iterative the framework and test every way of database migration. The framework supports each way of database migration and covers all problems occurring in an ITIL conform enterprise environment.
As long as a database administrator does his job, so long he has to improve his skills and knowledge steady. Today this comes in mind with the cloud hype. Every manager thinks it is necessary to go to the cloud although the expert reports advice there is no need to get into the cloud, because there is no real benefit for some applications. Some applications underlie a service level agreement that forbid giving sensible data to third party companies for instance. In some cases, it would be more expensive to put all systems into the cloud environment.

From the business demand to the approval, a database migration and all dependencies should be done complete, in time and without loss of data or data quality. This should be the measure for a complete database migration. The most projects are budget and time critical. In most cases, the business allows near no downtime that means that the application and its database also allow no downtime. Such database migration becomes time critical because it is necessary to minimize the downtime. The database migration framework cuts into four phases.

The database migration framework is divided into the following nine use cases and explained in section 3.1 to 3.11.

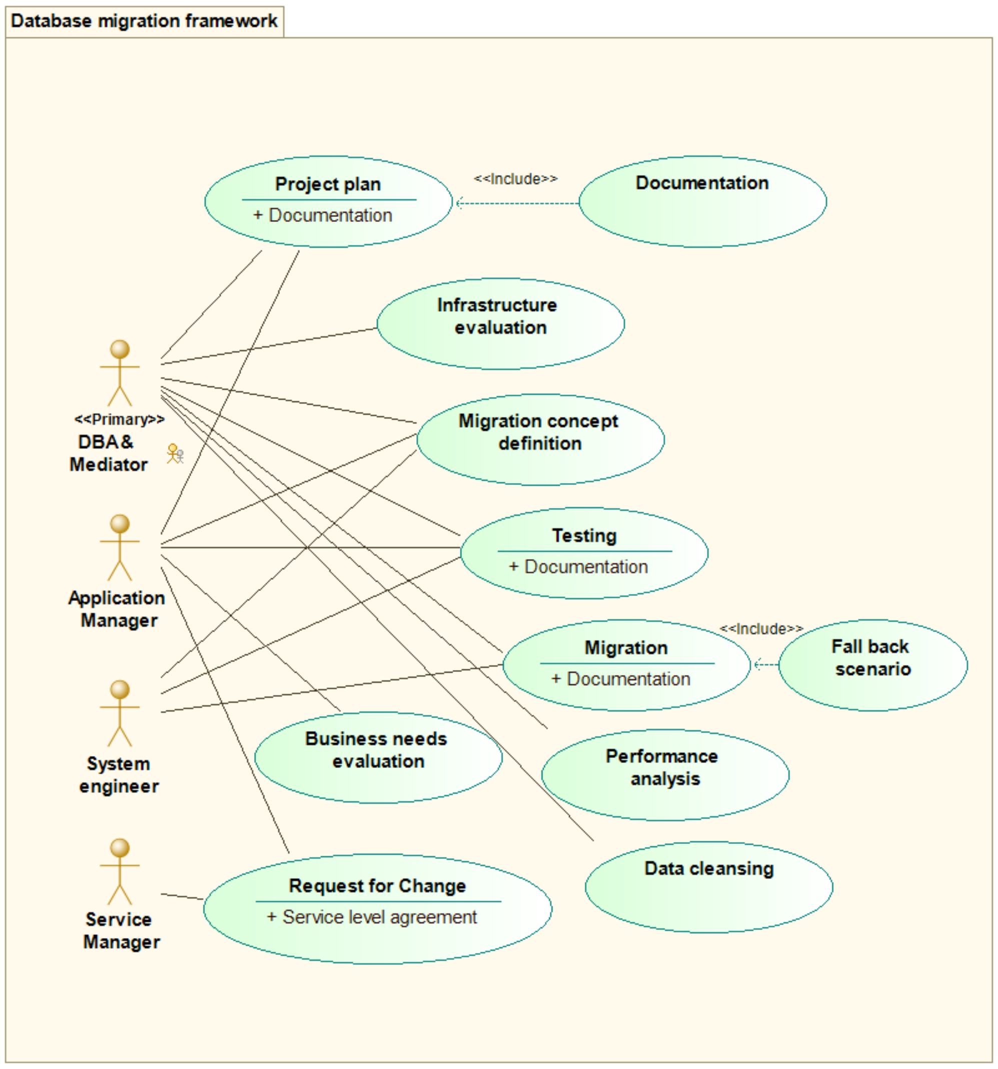


***Figure 9**: Database migration framework – use case diagram*

Each **actor** in the database migration framework can be a single person or a group of team members. In some migration projects it is necessary that only one person acts as a database administrator and there exist projects where a team of database administrators is mandatory. In case of a big infrastructure change where many applications are affected there are many application managers.

The database migration framework has three actors.

The ***actor database administrator*** short called DBA is also mediator between all responsible persons during the migration process and has professional technical skills. He is responsible for the use case migration and has a major function by the technical preparation of the migration concept.

The ***actor application manager*** is project purchaser or building owner and has knowledge about the business needs and the application.

The ***actor service manager*** is the sentinel in the ITIL environment and guarantees that all service level agreements and proceedings are complied.

3.1 Business needs evaluation

A database migration project starts with the creation of the project plan. Responsible for the project plan can be the application manager or the database administrator with the function as mediator. The first step is to evaluate the business needs. This task can be handled by the mediator but in most cases it will be assigned to the application manager.

The actor application manager evaluates the business needs and therefore all necessary application needs and requirements to ensure the application operation. All results have to be added to the project plan. These results build the base for the request for change.

3.2 Request for change (RFC)

In a ITIL environment, it is necessary to proof any sort of change before it will be announced. Some small changes are accepted as a hotfix and can be implemented in the short term without an extensive testing period. However other changes need a time expensive evaluation period.

An announcement is called request for change, short RFC.
Comprehensive changes have to be elaborately tested before they were announced as a request for change. All involved departments have to prove

their necessary tasks to giving green light to the service manager. The definition of the request for change ensure that nothing will be forgotten.

Every change has to meet the service level agreement requirements of the three key system process controls like **change control** process, **problem management** process and **configuration management** process. [14]

Each database migration must meet the needs formulated by the business process and the underlying service-level agreement (SLA) controlled by the service desk with the incident management. If new databases applied, they must comply with the security specifications and standards.

3.3 Infrastructure evaluation

The infrastructure or so called IT infrastructure consists of the server hardware, every sort of software and the application.

A lot of different trigger can

The **application** vendor makes a new version of a software product available with necessary new features or functionality. Pre-requisites and requirements have to be evaluated to meet the requirements of the software vendor and to meet the needs of the ITIL environment.

Application responsible persons needs a new functionality available in the new version of the application and order it by the external or in-house developer division. This could be realized by an Add-on implemented with a hotfix or an application update.

Infrastructure changes in terms of new hardware. In this case, the database and its fundament have to be relocated but there are no changes to be made in the database. Therefore, the database will only be relocated but not migrated.

Infrastructure changes in terms of new database engine. In this case, best practice is a full new installation often called from scratch. First, it will be the newest operating system version installed to meet the pre-requisites requirements of the database engine vendor. Second, the database engine installation is following. Third all newest patch level installation will be done to be up to date. If the database is older, it is possible to adapt an older compatibility level if needed. The new database engine can handle older databases with several compatibility levels.

3.4 Migration concept definition

Evaluation of infrastructure platform changes and evaluation of performance needs are also mandatory like business needs evaluation. The results of all these heights are the base for the migration strategy decision. This includes also the needs for backup and availability strategy.

Every need for change looks different. Also, every database migration looks different. The results from the business needs evaluation and the infrastructure evaluation forms the right migration concept definition.

For the database itself, it is useful to take down following checklist to prevent the migration process from any missing dram.

Independent in terms of the trigger for a database migration a mandatory environment evaluation process is necessary to cover all dependencies and specify the right tool set for data processing and handling.

The **Business process** and the dedicated **Service-level agreement** specifies the importance of a database and furthermore the parameter and thresholds for the monitoring system. Furthermore, if the application has very high status it means that there is nearly no downtime accepted. Downtime can lead to serious problems like security violation in services and in most cases downtime is directly associated with losing a lot of money. Therefore, the business process and its SLA determine the migration process.

Changes in **application** are documented and have to be evaluated in order to treat all necessary data processing tasks. Changes in application possibly require changes in the database. This can be a change for the whole database or only for a few tables or only for one column. This can induce new dependencies between several objects.

New functions in the application and changes in application lead often to changes in data structure. It is necessary to formulate all changes in application and data structure. With this information, it is possible to find the best fitting data migration strategy.

All applications in an ITIL environment are addicted to the application vender in terms of support. So, many migration scripts will be offered by the application vendor in form of T-SQL scripts.

In this migration framework exist three kinds of data manipulation techniques and executives.

Data manipulation techniques

- T-SQL scripts
- Stored procedures
- Integration Services (SSIS) Packages

3.4.1 Application and add-on or plug-in

In most applications there are possibly self-made or third party add-ons or plug-ins. Such custom-made add-ons or plug-ins can be a problem for a migration, if there is no documentation. In such case, a search for the interaction between database and application is necessary. In worst cases, workarounds or expensive reverse engineering is necessary.

3.4.2 Interfaces

needs to be tested and if applicable adapted to ensure the functionality.

3.4.3 Hardware

re-dimensioning becomes mandatory if it is too old or if the application needs more performance for instance. Two possible mistakes are false dimensioning and overcommitting. If the server hardware is false dimensioned, there is no room to provide more performance. Virtual farms are often overcommitted to save money. In most cases, such applications they are not performance critical overcommitting can save money, but in worst cases, it leads to performance problems.

3.4.4 Replication

Every replication has a full documentation which is mandatory to rebuild the replication on a new system if it is necessary.

3.4.5 Cubes

Microsoft SQL Server Analysis Services cubes and its dimensions need to be defined and deployed in the new migrated environment.

Every relation from relational data to cubes or vice versa has to be proved while the testing phase.

3.4.6 Database Jobs

Database jobs must be scripted to hold all information for reproducing all items in the new environment if it is necessary.

3.4.7 Linked server

A Linked server is a logical connection and sight from an SQL Server engine to another SQL Server engine. In some application code there exists application logic in stored procedures abusing linked server to optimize connection issues.

3.4.8 Database growth

is often underestimated. Considering database growth is mandatory for the budget and important to choose the right storage sub-system.

3.4.9 Database performance needs

have to be defined in the documentation to meet further monitoring parameter.

3.4.10 Backup strategy

Tools and scripts, schedules and the right backup strategy are also mandatory topics for the documentation. SQL Server Backups written to a file system can also be a good backup strategy as special software which takes care for the scheduled backups. A mandatory setting for production databases is to operate the database with recovery model *"Full"*. Best practice for most database uses is a full backup from databases every day most executed shortly after midnight. Furthermore, transaction log backups were made every two hours on every day in the week.

3.4.11 Availability strategy

The availability strategy defines the technical solution to guarantee the uptime specified in the service level agreement. Established availability definitions are SLA levels from 99.99% (“four nines”) uptime to 99.9999% (“six nines”) uptime. This means a downtime from 52.60 minutes to 31.56 seconds in a year.
The server hardware can be build disaster tolerant with cluster technology. The SQL Server engine can be operated with *“AlwaysOn”* High Availability Groups or with Log Shipping technology.

3.4.12 Security

The access to the database has to be restricted to a minimum. There exist three types of users. First user is called SQL Server user stored with username, password and permissions directly on the SQL Server engine. Then there are operating system users like a windows user with windows authentication stored in the active directory of the domain controller server. The third user is an active directory group which can hold multiple windows user.

3.5 Project plan

The project plan is the guide for the migration process. From the first Kick-off meeting till the last summary report the project plan holds all information about the project.

3.6 Documentation

In combination with the project plan the documentation holds all detailed data about each project phase and all detailed data about the migration concept, fall back scenario and instructions for the testing phase and detailed information about all test results. The documentation ensures an unobstructed migration.

3.7 Testing

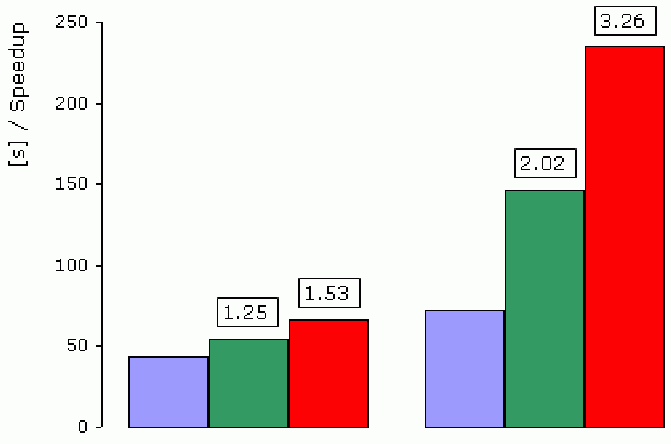
Testing is the most important topic. Pre-testing is mandatory. While testing there is the possibility to rectify incompatibilities and other errors in time. The testing phase happens prior, and sometimes while, the whole migration phase. Experts handle unexpected errors at an early stage.

Whether a planned system change, hotfix or a complete migration of the whole database, there is need for a test environment to proof the implementation process. Every small mistake or forgotten issue can lead to serious problems. It is mandatory necessary to formulate a worst-case scenario to evolve a reliable fall back scenario. The major factor for a successful migration is time. Tasks like data bulk-loads or a whole database restore seems often underestimated.

Best case for testing is a test system environment identical to the production system. In most cases exists a test system environment that is near identical to the production system only in lack of performance in order to save expenses for instance. In addition to the infrastructure test system, an application test system is also necessary. Application manager and system engineers responsible for the application should extensive test the new system and adds detailed results to the documentation. To prevent errors while the go-live phase documentation and communication is very important.

SQL Server Performance Test 1M

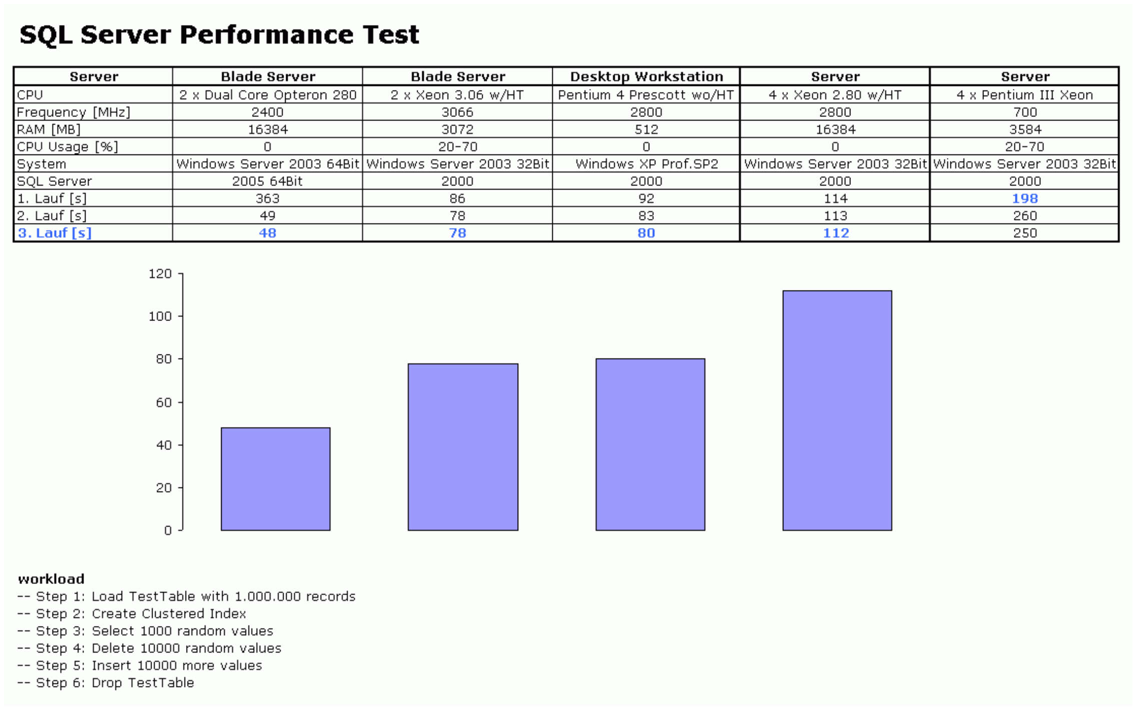
Server	Blade Server	Desktop Workstation
CPU	2 x Dual Core Opteron 280	Pentium 4 Prescott wo/HT
Frequency [MHz]	2400	2800
RAM [MB]	16384	512
CPU Usage [t=0] [%]	0	0
System	Windows Server 2003 64Bit	Windows XP Prof.SP2
SQL Server	2005 64Bit Enterprise	2000 Enterprise
Storage	EMC Symmetrix	Local Disk
3. Lauf [s] x 1	43	72
3. Lauf [s] x 2	54	146
3. Lauf [s] x 3	66	235



workload
```
-- Step 1: Load TestTable with 1.000.000 records
-- Step 2: Create Clustered Index
-- Step 3: Select 1.000 random values
-- Step 4: Delete 10.000 random values
-- Step 5: Insert 10.000 more values
-- Step 6: Drop TestTable
```

Figure 10*: 1 Million row insert Benchmark*

Figure 10 compare benchmark results between a server hardware and a workstation hardware with comparable hardware components. The server hardware can operate the six database operations in round half the time. The scalability of the server hardware shows a speedup of 3.5 by executing the six database operations steps three times at once.


SQL Server Performance Test

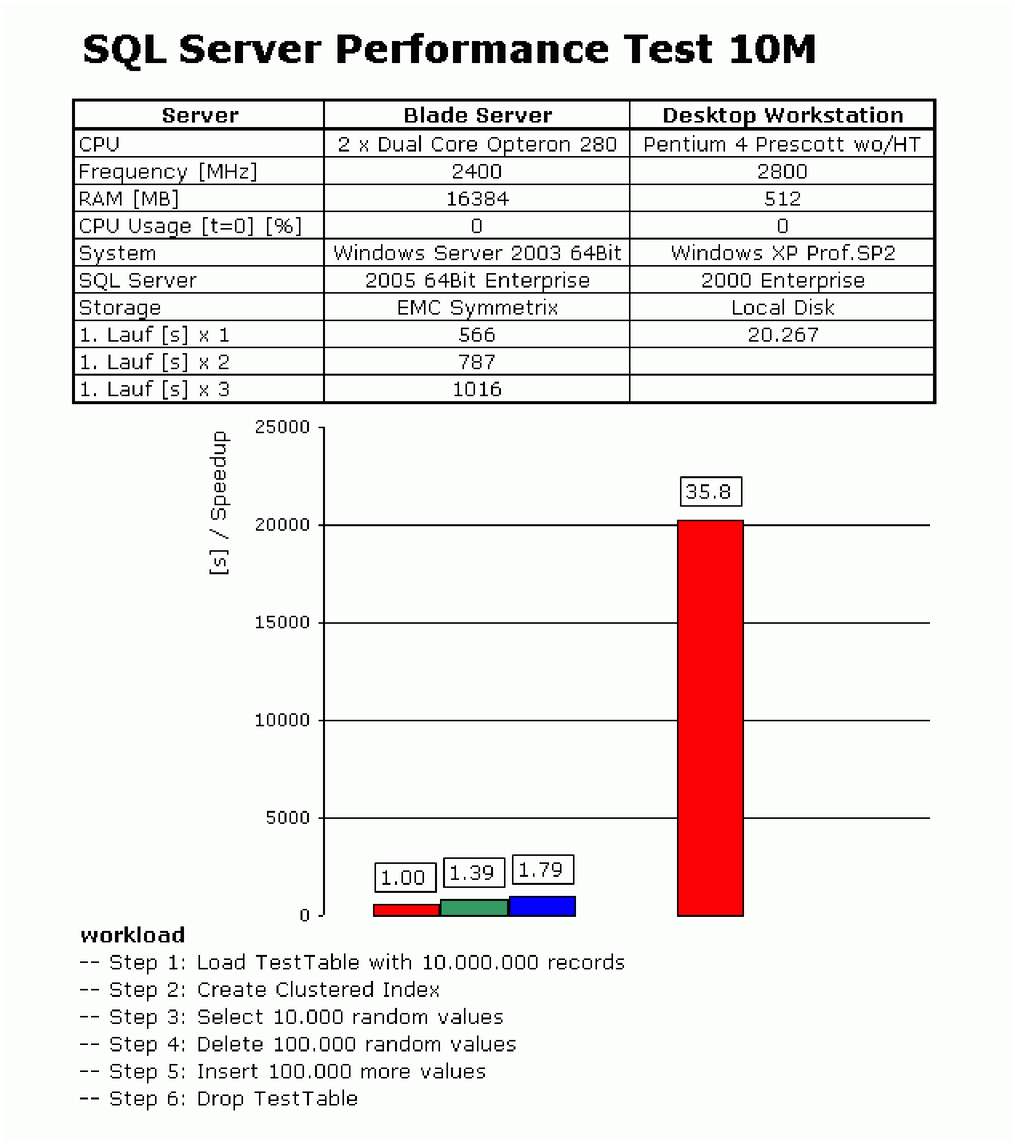
Server	Blade Server	Blade Server	Desktop Workstation	Server	Server
CPU	2 x Dual Core Opteron 280	2 x Xeon 3.06 w/HT	Pentium 4 Prescott wo/HT	4 x Xeon 2.80 w/HT	4 x Pentium III Xeon
Frequency [MHz]	2400	3066	2800	2800	700
RAM [MB]	16384	3072	512	16384	3584
CPU Usage [%]	0	20-70	0	0	20-70
System	Windows Server 2003 64Bit	Windows Server 2003 32Bit	Windows XP Prof.SP2	Windows Server 2003 32Bit	Windows Server 2003 32Bit
SQL Server	2005 64Bit	2000	2000	2000	2000
1. Lauf [s]	363	86	92	114	**198**
2. Lauf [s]	49	78	83	113	260
3. Lauf [s]	**48**	**78**	**80**	**112**	250

workload

```
-- Step 1: Load TestTable with 1.000.000 records
-- Step 2: Create Clustered Index
-- Step 3: Select 1000 random values
-- Step 4: Delete 10000 random values
-- Step 5: Insert 10000 more values
-- Step 6: Drop TestTable
```

Figure 11*: 1 Million row insert Benchmark*

Figure 11 shows results from different server. This benchmark demonstrates the higher computing performance from the newer SQL Server 2005 engine against the older SQL Server 2000 engine.



SQL Server Performance Test 10M

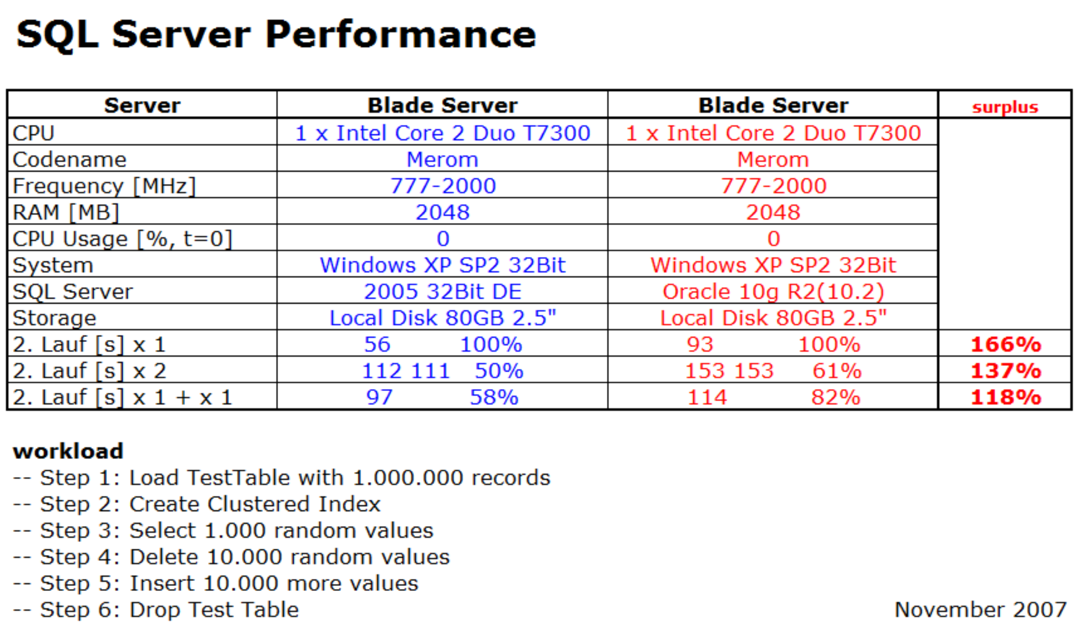
Server	Blade Server	Desktop Workstation
CPU	2 x Dual Core Opteron 280	Pentium 4 Prescott wo/HT
Frequency [MHz]	2400	2800
RAM [MB]	16384	512
CPU Usage [t=0] [%]	0	0
System	Windows Server 2003 64Bit	Windows XP Prof.SP2
SQL Server	2005 64Bit Enterprise	2000 Enterprise
Storage	EMC Symmetrix	Local Disk
1. Lauf [s] x 1	566	20.267
1. Lauf [s] x 2	787	
1. Lauf [s] x 3	1016	

workload

-- Step 1: Load TestTable with 10.000.000 records
-- Step 2: Create Clustered Index
-- Step 3: Select 10.000 random values
-- Step 4: Delete 100.000 random values
-- Step 5: Insert 100.000 more values
-- Step 6: Drop TestTable

***Figure 12**: 10 Million row insert Benchmark*

Figure 12 demonstrates the scalability of the SQL Server 2005 engine in conjunction with the type of processor. Although the server hardware has four processors, the execution of multiple jobs takes more time as expected.



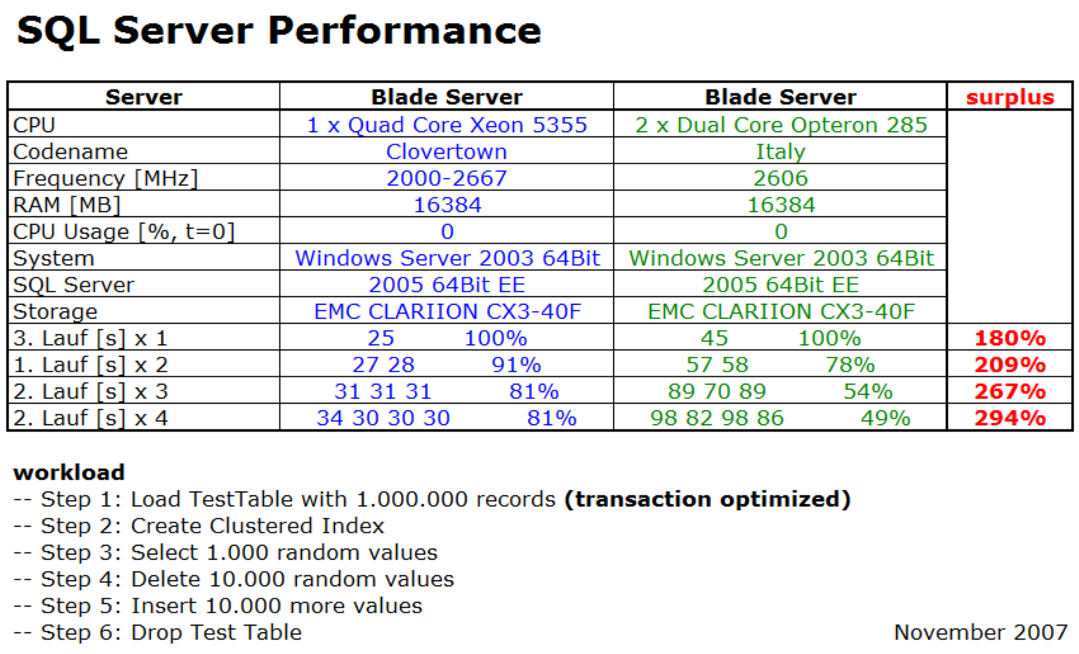
SQL Server Performance

Server	Blade Server	Blade Server	surplus
CPU	1 x Intel Core 2 Duo T7300	1 x Intel Core 2 Duo T7300	
Codename	Merom	Merom	
Frequency [MHz]	777-2000	777-2000	
RAM [MB]	2048	2048	
CPU Usage [%, t=0]	0	0	
System	Windows XP SP2 32Bit	Windows XP SP2 32Bit	
SQL Server	2005 32Bit DE	Oracle 10g R2(10.2)	
Storage	Local Disk 80GB 2.5"	Local Disk 80GB 2.5"	
2. Lauf [s] x 1	56 100%	93 100%	**166%**
2. Lauf [s] x 2	112 111 50%	153 153 61%	**137%**
2. Lauf [s] x 1 + x 1	97 58%	114 82%	**118%**

workload
-- Step 1: Load TestTable with 1.000.000 records
-- Step 2: Create Clustered Index
-- Step 3: Select 1.000 random values
-- Step 4: Delete 10.000 random values
-- Step 5: Insert 10.000 more values
-- Step 6: Drop Test Table

November 2007

***Figure 13**: 1 Million row insert Benchmark*



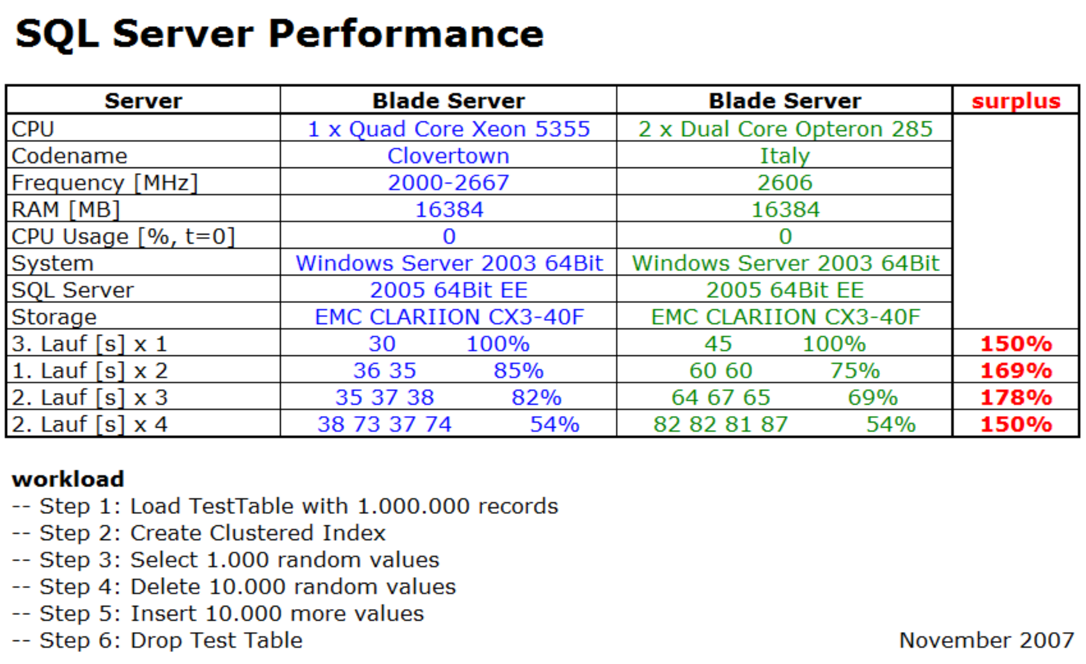
SQL Server Performance

Server	Blade Server	Blade Server	surplus
CPU	1 x Quad Core Xeon 5355	2 x Dual Core Opteron 285	
Codename	Clovertown	Italy	
Frequency [MHz]	2000-2667	2606	
RAM [MB]	16384	16384	
CPU Usage [%, t=0]	0	0	
System	Windows Server 2003 64Bit	Windows Server 2003 64Bit	
SQL Server	2005 64Bit EE	2005 64Bit EE	
Storage	EMC CLARIION CX3-40F	EMC CLARIION CX3-40F	
3. Lauf [s] x 1	25 100%	45 100%	**180%**
1. Lauf [s] x 2	27 28 91%	57 58 78%	**209%**
2. Lauf [s] x 3	31 31 31 81%	89 70 89 54%	**267%**
2. Lauf [s] x 4	34 30 30 30 81%	98 82 98 86 49%	**294%**

workload
-- Step 1: Load TestTable with 1.000.000 records **(transaction optimized)**
-- Step 2: Create Clustered Index
-- Step 3: Select 1.000 random values
-- Step 4: Delete 10.000 random values
-- Step 5: Insert 10.000 more values
-- Step 6: Drop Test Table

November 2007

***Figure 14**: 1 Million row insert Benchmark*



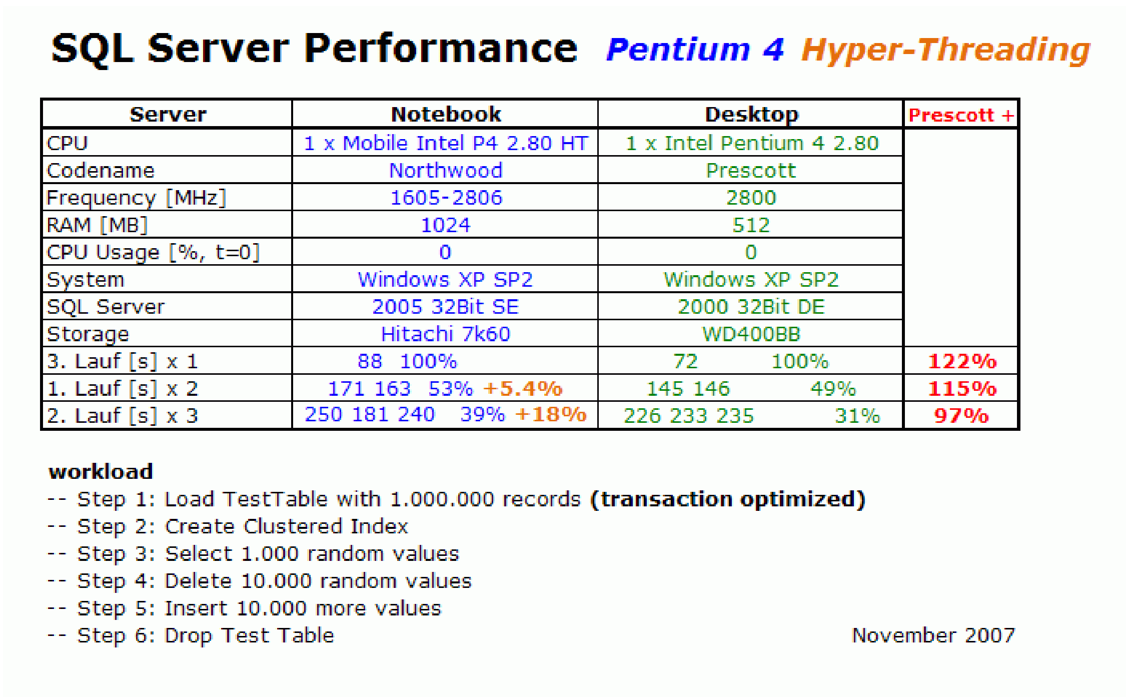
SQL Server Performance

Server	Blade Server	Blade Server	surplus
CPU	1 x Quad Core Xeon 5355	2 x Dual Core Opteron 285	
Codename	Clovertown	Italy	
Frequency [MHz]	2000-2667	2606	
RAM [MB]	16384	16384	
CPU Usage [%, t=0]	0	0	
System	Windows Server 2003 64Bit	Windows Server 2003 64Bit	
SQL Server	2005 64Bit EE	2005 64Bit EE	
Storage	EMC CLARIION CX3-40F	EMC CLARIION CX3-40F	
3. Lauf [s] x 1	30 100%	45 100%	**150%**
1. Lauf [s] x 2	36 35 85%	60 60 75%	**169%**
2. Lauf [s] x 3	35 37 38 82%	64 67 65 69%	**178%**
2. Lauf [s] x 4	38 73 37 74 54%	82 82 81 87 54%	**150%**

workload
-- Step 1: Load TestTable with 1.000.000 records
-- Step 2: Create Clustered Index
-- Step 3: Select 1.000 random values
-- Step 4: Delete 10.000 random values
-- Step 5: Insert 10.000 more values
-- Step 6: Drop Test Table

November 2007

***Figure 15**: 1 Million row insert Benchmark*



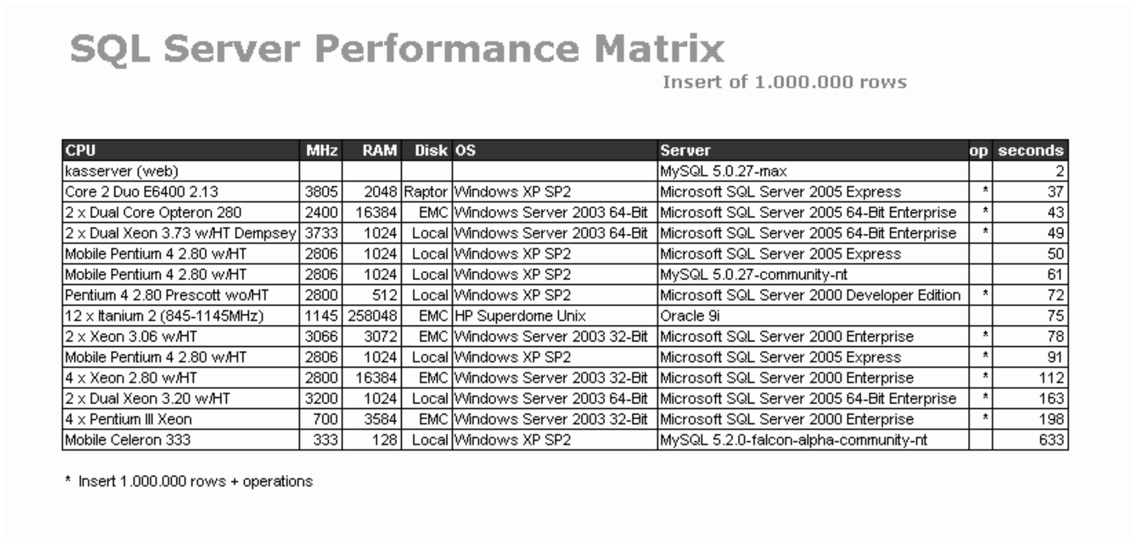
SQL Server Performance *Pentium 4* *Hyper-Threading*

Server	Notebook	Desktop	Prescott +
CPU	1 x Mobile Intel P4 2.80 HT	1 x Intel Pentium 4 2.80	
Codename	Northwood	Prescott	
Frequency [MHz]	1605-2806	2800	
RAM [MB]	1024	512	
CPU Usage [%, t=0]	0	0	
System	Windows XP SP2	Windows XP SP2	
SQL Server	2005 32Bit SE	2000 32Bit DE	
Storage	Hitachi 7k60	WD400BB	
3. Lauf [s] x 1	88 100%	72 100%	**122%**
1. Lauf [s] x 2	171 163 53% **+5.4%**	145 146 49%	**115%**
2. Lauf [s] x 3	250 181 240 39% **+18%**	226 233 235 31%	**97%**

workload
-- Step 1: Load TestTable with 1.000.000 records **(transaction optimized)**
-- Step 2: Create Clustered Index
-- Step 3: Select 1.000 random values
-- Step 4: Delete 10.000 random values
-- Step 5: Insert 10.000 more values
-- Step 6: Drop Test Table

November 2007

***Figure 16**: 1 Million row insert Benchmark*



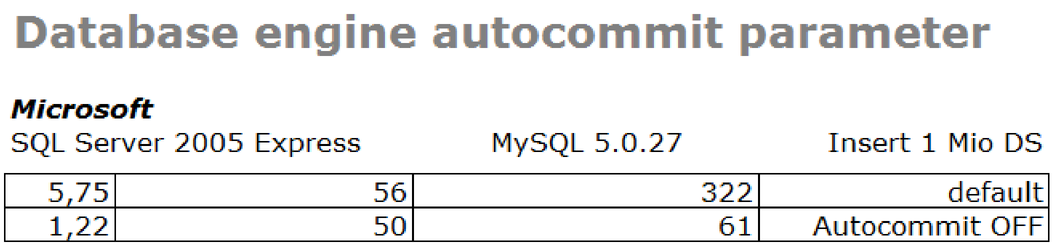
SQL Server Performance Matrix

Insert of 1.000.000 rows

CPU	MHz	RAM	Disk	OS	Server	op	seconds
kasserver (web)					MySQL 5.0.27-max		2
Core 2 Duo E6400 2.13	3805	2048	Raptor	Windows XP SP2	Microsoft SQL Server 2005 Express	*	37
2 x Dual Core Opteron 280	2400	16384	EMC	Windows Server 2003 64-Bit	Microsoft SQL Server 2005 64-Bit Enterprise	*	43
2 x Dual Xeon 3.73 w/HT Dempsey	3733	1024	Local	Windows Server 2003 64-Bit	Microsoft SQL Server 2005 64-Bit Enterprise	*	49
Mobile Pentium 4 2.80 w/HT	2806	1024	Local	Windows XP SP2	Microsoft SQL Server 2005 Express		50
Mobile Pentium 4 2.80 w/HT	2806	1024	Local	Windows XP SP2	MySQL 5.0.27-community-nt		61
Pentium 4 2.80 Prescott wo/HT	2800	512	Local	Windows XP SP2	Microsoft SQL Server 2000 Developer Edition	*	72
12 x Itanium 2 (845-1145MHz)	1145	258048	EMC	HP Superdome Unix	Oracle 9i		75
2 x Xeon 3.06 w/HT	3066	3072	EMC	Windows Server 2003 32-Bit	Microsoft SQL Server 2000 Enterprise	*	78
Mobile Pentium 4 2.80 w/HT	2806	1024	Local	Windows XP SP2	Microsoft SQL Server 2005 Express	*	91
4 x Xeon 2.80 w/HT	2800	16384	EMC	Windows Server 2003 32-Bit	Microsoft SQL Server 2000 Enterprise	*	112
2 x Dual Xeon 3.20 w/HT	3200	1024	Local	Windows Server 2003 64-Bit	Microsoft SQL Server 2005 64-Bit Enterprise	*	163
4 x Pentium III Xeon	700	3584	EMC	Windows Server 2003 32-Bit	Microsoft SQL Server 2000 Enterprise	*	198
Mobile Celeron 333	333	128	Local	Windows XP SP2	MySQL 5.2.0-falcon-alpha-community-nt		633

* Insert 1.000.000 rows + operations

Figure 17*: 1 Million row insert Benchmark*



Database engine autocommit parameter

Microsoft SQL Server 2005 Express		MySQL 5.0.27	Insert 1 Mio DS
5,75	56	322	default
1,22	50	61	Autocommit OFF

Figure 18*: 1 Million row insert Benchmark*

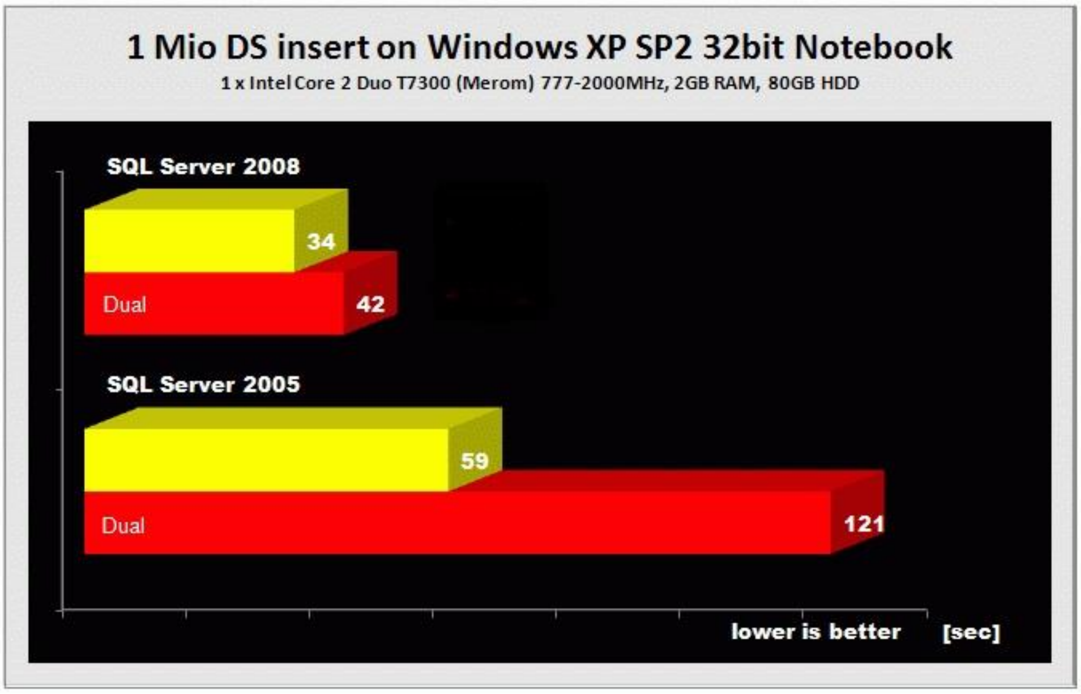


Figure 19*: 1 Million row insert Benchmark*

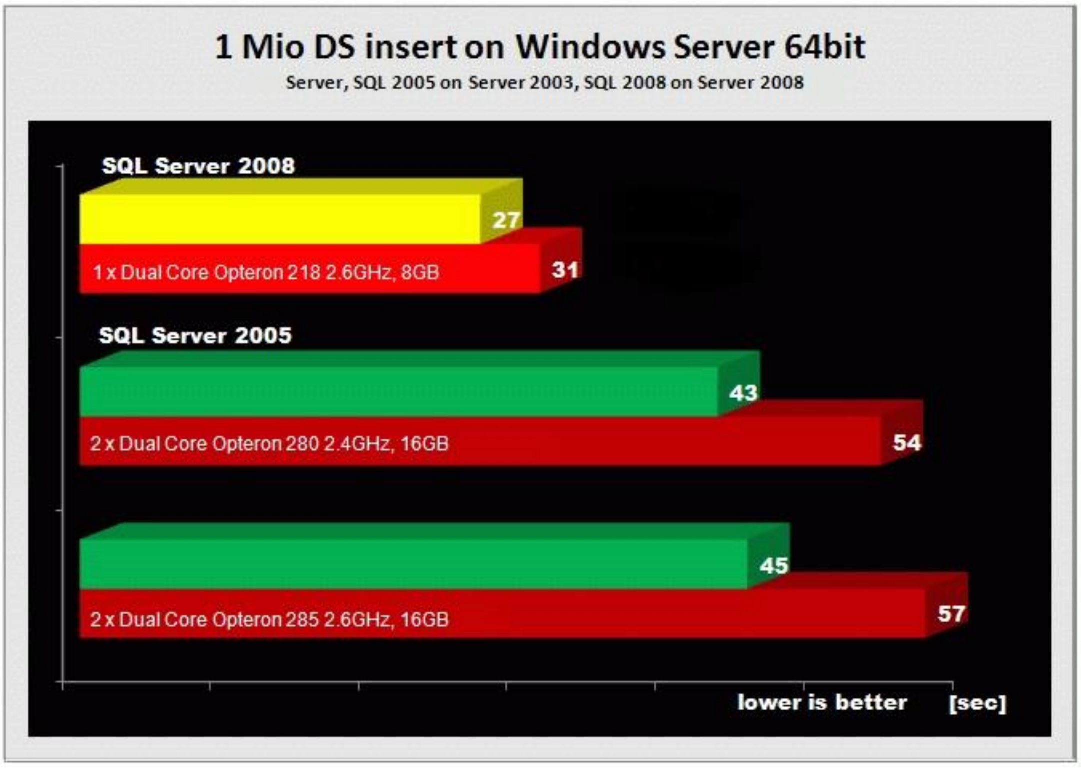


Figure 20*: 1 Million row insert Benchmark*

Figure 13 to figure 20 representing benchmark results to compare SQL Server engine performance and to compare CPU performance of different CPU types with different clock rating. This is very important to calculate SQL query runtimes. A list of benchmark results gives a hint to predict query runtimes. This is necessary and useful to plan the migration period. Furthermore are such benchmark results an important base in the course of performance analysis.

3.8 Migration

The documentation is filled with all test results. The service manager has the system change announced and give green light to the migration team.

The migration team consists of the application manager, the database administrator and the system engineer.

The **complete tool set** consists of scripts for all possible cases occurring while data processing and system tasks. Included are all variants of scripts like T-SQL scripts, PowerShell scripts, Command-line scripts and SSIS packages. The complete tool set is very important for the daily work of a database administrator. Every database administrator has his own complete tool set and it grows with growing experience.

There are several sometimes difficult or tricky needs for data manipulation. For example, some old legacy databases with hundreds of tables with big amounts of datasets do not fit any normal form. There exists no primary key and no constraints. Therefore, without any knowledge about how the application work and handle data, there is no chance to clean data and data structures. With special scripts, it is possible to delete double or multiple datasets according to an identified logic.

3.9 Fall back scenario

The migration concept need a fall back scenario to guarantee the business operation as usual. Worst case scenarios have to be defined and build the base for a fall back scenario. Because of the knowledge of a worst case scenario it should be proved if it is possible to avoid the worst case scenario. This can probably lower the possibility of problem occurring.

3.10 Performance analysis

Slogans like time to market and consolidation in business are often in charge of rising performance problems. These two conflictive strategies accelerate the need for performance. In addition, comes the big rising amount of data. Big data and faster data warehouse solutions are trigger for tremendous gain of data. Technological evolution like faster processor, faster main memory and faster disk storage solutions are very useful and necessary to process the rapidly growing data. However, the data is growing much faster than technological evolution can provide faster hardware solutions.
Therefore, the need for more performance leads mandatory to comprehensive performance analysis to overcome the need for performance and to find the bottlenecks in the environment.

In the course of hardware, change like relocate a database from a hardware server cluster environment to a virtual server machine environment just as relocate a database from an older database engine to a newer one performance analysis becomes indispensable.

The rapidly growing amount of data and the growing demand for reports are one major trigger for the need of performance analysis. The other important trigger is the trend of virtualization related on cost reduction. This leads to the importance of performance analysis.

To get a clue about performance it is necessary to know what amount of data load is a server hardware capable to process in a specific time. Furthermore, it is mandatory to know what amount of instructions is a processor able to execute. A synthetic benchmark result is nice but it is often only a measurement useful to compare a server with another server. Such a benchmark result does not give guaranteed information if the server is capable complex application operations. Only a database administrator with a lot of long year experience is able to extrapolate the performance results to compare it with a well-known real world application.

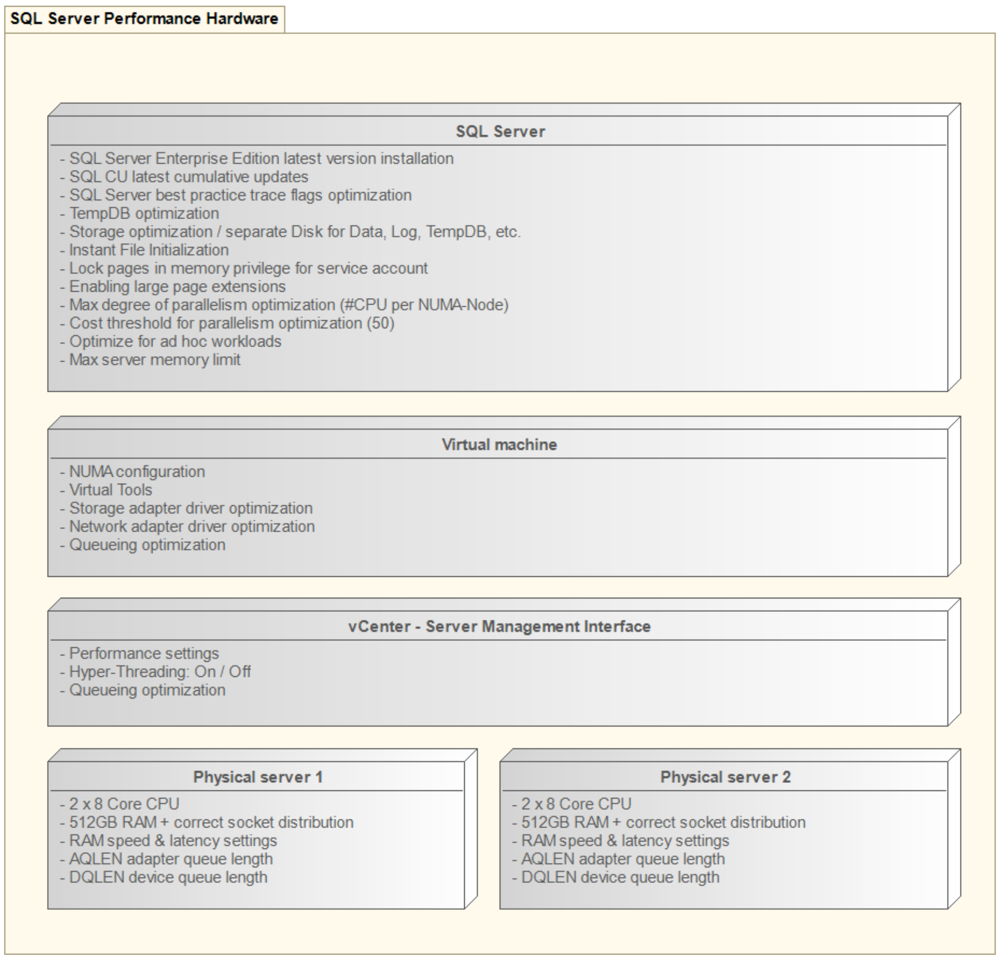
A benchmark result in context with a real life workload is more meaningful to predict data load runtime instead of a result from a synthetic benchmark.

Following benchmark result shows the time a database engine needs to insert one million rows of data on different hardware and processed with different database engines on different operating systems.
These results provide a basis for further runtime calculations in big data environment.

A couple of years ago, for example the timeframe from 2002 to 2010 there were significant processor improvements on the market. Every year processor performance measurably increased also all other hardware components became faster.

A Manager who is responsible for application has no clue about any database operation. He is responsible for the operation of application and he is responsible to follow the business strategy.
To predict runtime behavior, it is necessary to get sense of the server hardware. Very interesting are real world benchmarks like a mass insert of data or other database operations.
The slowest hardware chain link in a server is the disk subsystem. To optimize the whole batch job, it is often necessary to optimize parallel bulk load tests to predict runtime of tasks they were very time consuming.

Nowadays the server hardware is high scalable and with very good performance.
The increasing amount of data and the increasing granularity of data require special techniques to process this big amount of data to satisfy the need for performance.


SQL Server Performance Hardware

SQL Server

- SQL Server Enterprise Edition latest version installation
- SQL CU latest cumulative updates
- SQL Server best practice trace flags optimization
- TempDB optimization
- Storage optimization / separate Disk for Data, Log, TempDB, etc.
- Instant File Initialization
- Lock pages in memory privilege for service account
- Enabling large page extensions
- Max degree of parallelism optimization (#CPU per NUMA-Node)
- Cost threshold for parallelism optimization (50)
- Optimize for ad hoc workloads
- Max server memory limit

Virtual machine

- NUMA configuration
- Virtual Tools
- Storage adapter driver optimization
- Network adapter driver optimization
- Queueing optimization

vCenter - Server Management Interface

- Performance settings
- Hyper-Threading: On / Off
- Queueing optimization

Physical server 1

- 2 x 8 Core CPU
- 512GB RAM + correct socket distribution
- RAM speed & latency settings
- AQLEN adapter queue length
- DQLEN device queue length

Physical server 2

- 2 x 8 Core CPU
- 512GB RAM + correct socket distribution
- RAM speed & latency settings
- AQLEN adapter queue length
- DQLEN device queue length

Figure 21*: SQL Server Performance Hardware*

Figure 21 shows a methodical approach to build a high performance hardware for SQL Server and configure the SQL Server engine for high performance needs as well. On the physical server level, it is necessary to install fast memory and configure the memory according to best practice for performance. Parameter settings for RAM speed and latency should also be set according to best practice for performance. The queueing settings for adapter and device should be synchronized with the parameter settings for queueing on the server management interface and on the virtual machines. The SQL Server engine should be operated with 64KB cluster size on the NTFS file system because the engine handles 8 pages with 8KB blocks for performance. All these parameter settings are the results of testing experience in combination of results of studying best practice whitepapers of all involved hardware and software artifacts.

3.10.1 Classification of performance analysis

3.10.1.1 Hardware performance analysis

Hardware performance is all around measureable hardware components involved by database operations like server components and their settings called mainboard, BIOS, CPU, memory, storage system, network adapter

3.10.1.2 Software performance analysis

Software components like application, application interface and driver can lead to comprehensive performance analysis specially to evaluate the runtime behavior by optimizing multiple parameter simultaneous. In many cases the right connection settings from the application via the network to the database engine helps to optimize the performance and guarantees the transaction commits without timeouts.

3.10.1.3 Database engine performance analysis

Every database engine version and database engine from different manufacturer has several performance criteria and a lot of parameter settings for tuning and fine tuning. There is very much to know about performance behavior and fine-tuning possibilities. Some parameter for SQL Server engine runtime can induce bottlenecks for application while they are mandatory for other applications.

3.10.1.4 Logical query performance analysis

The challenge is to find the key to poor performing queries. Many variants of queries and their combinations needs well-founded knowledge about to build queries and knowledge about the query optimizer in the database engine. Trace flags for the database engine manipulate engine behavior in various use cases.

3.10.2 Procedural problem finding approach

3.10.2.1 Gain information

- Documentation of the problem description and definition.
- Localization of direct involved hardware, database engine version and software.
- Gain detailed information and statistical data about hardware, database engine version and software.

3.10.2.2 Analyses information and statistical data

- Proof gained information to justify best practice and needs.
- Deep database engine analysis on the base of gained data.
- Best practice evaluation in test system.

3.10.2.3 Proposal list with evaluated solutions

- Solution possibility proof in terms of service level agreement.
- Request for change for the chosen solution.
- Implement solution.

3.11 Data cleansing

Data cleansing can be necessary before migrating the data and can also be a possible task after migrating the data.
Sometimes data cleansing is necessary after successful database migration. This task needs an accepted and announced hotfix or a further system change by the change management. In some cases, old data will be defined and backed up externally or in the production database on another file group to be able to view the data in further future. Data masking, data deleting or data encryption are also tasks for data cleansing. Data masking make data unreadable but visible as usual. Often salary data will be masked. This makes the data readable but hides the real meaning or real values of the data.

4 Implementation and case study

The framework for database migration, in this used example, is divided into four phases.

4.1 An application driven database migration

This example shows the approach and implementation of the framework in case of an application driven database migration.

4.1.1 Phase 1: Pre-Requisites and business needs evaluation.

The conceptual framework covers all possible occurring prospects. Therefore, in this case of an application driven database migration the conceptual framework can be used as a guide. Pre-requisites comprehend the technical requirements of the application. The business need evaluation attests a business decision to update an existing application to a newer version containing sensitive improvements to the functionality of the application. In a ITIL environment there are several applications running maybe all from different vendors. State of the art, they were all running on a single SQL Server engine for cost saving.

4.1.2 Phase 2: Migration concept planning

After the business needs and environment evaluation, the right migration strategy guarantees going live in time.

Project Plan. If the database migration happens in the context of a project, the conditions, restrictions and criteria needs respected. The milestone information as well as weekly progress updates updated to the project plan. If the database migration happens in place and is a smaller change, it is although necessary to communicate with the responsible persons. Often small database migration scenarios are full of possible errors because the complexity has been underestimated.

Timeframe for Migration to go live and getting production status is documented in the project plan and should be proofed and communicated many times while the migration.

Monitoring system. In order to guarantee the daily work after migration the proof of change in the monitoring system is necessary. If there are any changes in the alarm escalation chain or new parameter, the monitoring system needs the new information. In some cases, additional fine-tuning of threshold parameter or other dependencies is necessary.

Documentation is mandatory. Every step of the whole migration procedure run into the documentation.

A **Mediator** is needed and in most cases the database administrator is the mediator and responsible for the communication with all responsible persons in addition to the project leader.

4.1.3 Phase 3: Migration testing

The testing phase is very important therefore, it is mandatory to simulate the whole migration in the test environment. Every small error and every unforeseen delay needs documented attention. Also the fall back scenario needs a test to proof this time critical task. Many test environments are not identical to the production environment in terms of size and performance. This lack of performance needs extrapolation to meet the real runtime of the production system. The documented results of the testing phase set the basis for the migration process.
New functionalities to a new version of an application leads often to changes in the database. The change can be necessary in the database schema. Also changes in tables or new tables are possible. When the structure has changed in existing tables it is necessary to rewrite queries. Queries can be stored direct in the application or in stored procedures in the database or in views in the database. Schema changing scripts, stored procedures and SSIS packages have to be executed in a determined order and the results get into the documentation.

4.1.4 Phase 4: Go live and target cleansing

The system change announced by the change management for discussion in the review board meeting needs approval. If there are no objections, the system change announced in the planned downtime listing waits for execution as scheduled. The database administrator and the system engineer execute the migration step by step project plan conform and documentation conform. The application manager is testing the connection and the functionality of the application as in the documentation and testing additions designated.

4.2 Performance Analysis

Performance cannot be static measured. In context with a database environment, or a data-warehouse existing reference values telling how much time a job may take. The whole system is as fast in terms of time consuming as the sum of all involved time taking items as hardware, software, driver, application, parameter, and so on. The need for performance is growing in a faster changing environment. Many variables are involved and need attention to solve time critical problems. In context with database performance problems there can be a checklist formulated to efficiently find the problem causer and offer a fast realizable solution.

Let us assume that a database migration was successful. The application and the database has production status, all is up and running daily business. Then the service management crew gets an incident with the short description: "Application is slow!" In most cases, such lack of performance is a subjective customer perception. Anyhow, the service management process has to prove all possible sources of errors.
Every division has to proof the problem according to its competence. The network services division reports no observable problems and no delays. The Wintel services division reports no errors in the VMware log files and all is up and running. The storage systems are all up and running, no error has occurred and no performance issue is logged or known.

Colleagues from the application services division, also the application manager, knows that they are working with a database, most identified as a black box. They are sure the problem only is with the database. Nevertheless, the database administrator also cannot find any error, neither in the log files nor in any other behavior. All is fine and up and running. The database administrator has to start a time expensive search. Now starts an observation of the application behavior. With profiling tools and special T-SQL queries the database administrator try to find the bottleneck to solve the problem. This is only possible when the database administrator has fundamental in-depth knowledge about advanced performance analysis.

Problem description and definition

At first, it is mandatory to describe and define the problem. The developer or the application responsible, most called application manager, must describe the problem in detail. In most cases, the database administrator is the first contact person who has to deal with the problem. The further function of the DBA is mediator and further problem officer. The DBA analyses the problem in almost the same manner as listed above.

To describe and define the problem the DBA confronted with an acquisition request because the data warehouse loads are very slow for instance. The most requests are emotional driven and not technical associated prepared. The DBA has to be a mediator and starts collecting more detailed information concerning infrastructure system, network subsystem, and application direct and indirect involved and the behavior of the problem.

The DBA is an expert all around database belongings and different database systems. In real life, a DBA has a lot of praxis often serve as a mediator. In a ITIL environment, the DBA has his daily workload and out of his duty many colleagues need his knowledge and his solution performance
Therefore, if a performance problem occurs, the DBA is one of the most involved persons. Directly involved persons to a specific problem are application manager. The application manager can specify the problem and can bring a detailed statistic about the problem behavior. Those statistics are the best basis for further performance analysis.

The DBA starts to collect performance monitor counter and statistical data over an arranged period in case this is possible. Meanwhile the data warehouse developer collects statistical data about the runtime of the affected application over the same arranged period. With the data from the application runtime statistics the database administrator can seek into detail, is able to find the problem, and offers a solution. There are many possible scenarios or so-called solutions. Only good research can recommend the best solution.

At first, let us have a look what is going on at the server hardware. What amount of traffic has the server to handle in this arranged observation period? In general, the database engine should not exceed average CPU utilization over 80% over a long period. Empirical values about possible speed of a server hardware at hand helps to classify the bottleneck. Needful performance monitor counter gives a look on the health of a database server engine and the involved operation system. Listing A.1 lists most important performance monitor counter to give a statement about the health of the observed SQL Server, the IO-subsystem, processor metrics, metrics of the underlying virtual machine processor and memory.

Such a user-defined configuration needs to be installed on the operating system to generate ad-hoc recordings or scheduled recordings to get a result set of a specific critical period in the night from two to three o'clock for instance.

In a data warehouse, there exists a lot of time critical tasks. Huge amounts of data needs transferred from one SQL Server to another SQL Server for further aggregation. If such a data transfer takes too much time as acceptable, the time will definitely be lost because the next data transfer is waiting. Therefore, it could happen that further aggregation cannot happen and important reports not realized.

The challenge begins to find the causer process for the time loss.
It is also possible to reconsider the underlying process. Not rarely a new SQL Server added in terms of load balancing to fulfill the task.

The results of the performance monitor period give a clue about the overall SQL Server health. Analysis of the data copy process gives a necessary illustration about the implementation of the process. A look for the copy parameters, versions and special notes of involved driver and driver settings is necessary to judge the copy process.
Maybe there are better opportunities to copy data, such as special tuned bulk loads or special developed Integration Services Service packages (SSIS packages) or fast load scripts to fulfill the task. Always be aware of parallel settings and parameter. Parallel settings must be totally understood to give a guarantee of success.

TPC Benchmark C is an on-line transaction processing (OLTP) benchmark with complex multiple transaction types. Downloadable for free is an unofficial open source benchmark. It has character of a synthetic benchmark although it simulates a real workload of up to fifty virtual users to give a manageable statement about the measured system.

Notwithstanding that, the benchmark is a synthetic benchmark with real time character, the TPC-C Benchmark used for determining scaling problems in a plenty used environment.

A good tool therefore is called *HammerDB.* It is an open source load-testing tool simulates multiple user transactions like the original TPC-C benchmark. The results of the benchmark measured in transactions per minute over a period, shows the performance a SQL Server system is capable. Variation of the gained results are normal. The benchmark can help to tune the SQL Server and play with parameter like degree of parallelism or cost threshold for parallelism.

Following results made on different server with a simulated workload of 50 users.

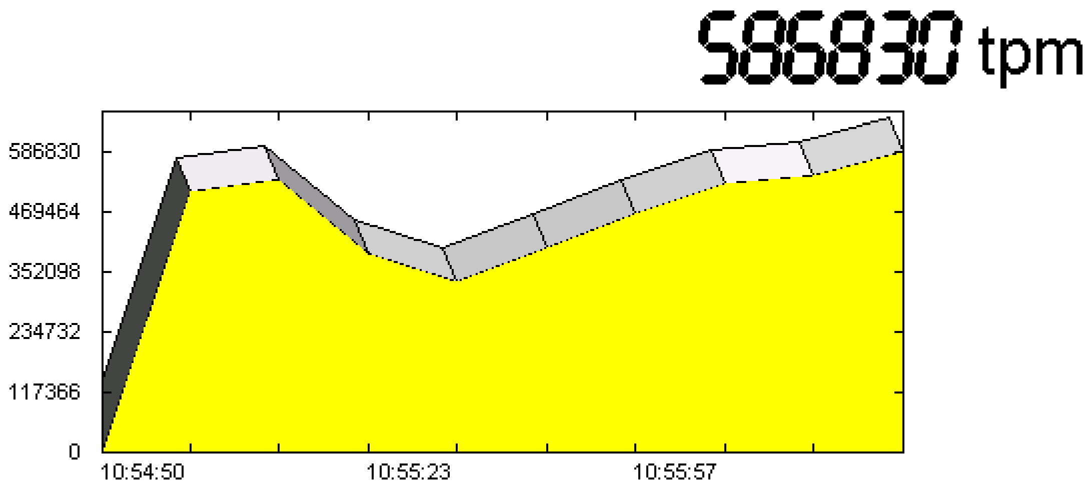


***Figure 22**: HammerDB, inofficial TPC-C benchmark*

In figure 22, a physical test server SQL Server 2008R2 with four CPUs Intel Xeon E5530 @ 2.40GHz brings a reference result of up to 586830 transactions per second. The same test server need 23 seconds for the reference benchmark inserting one millions rows of data.

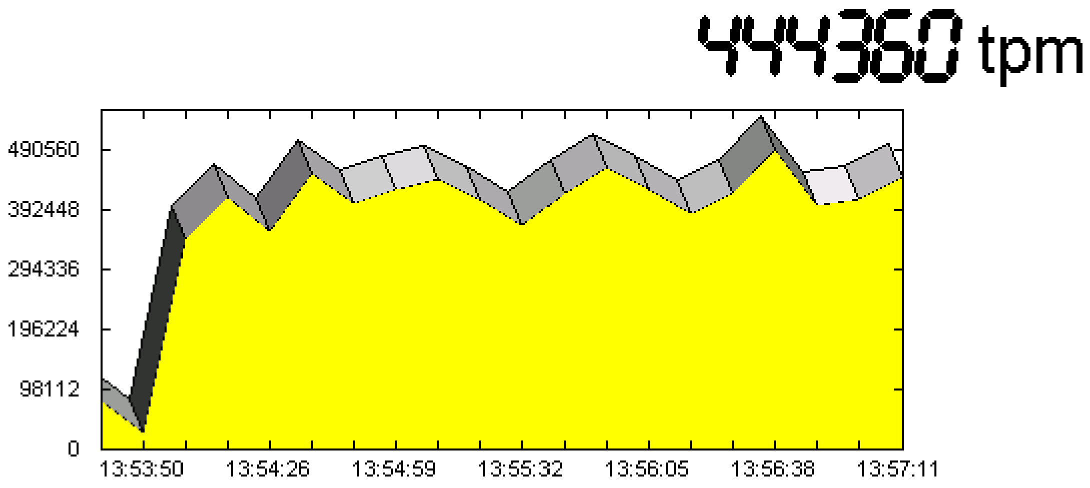
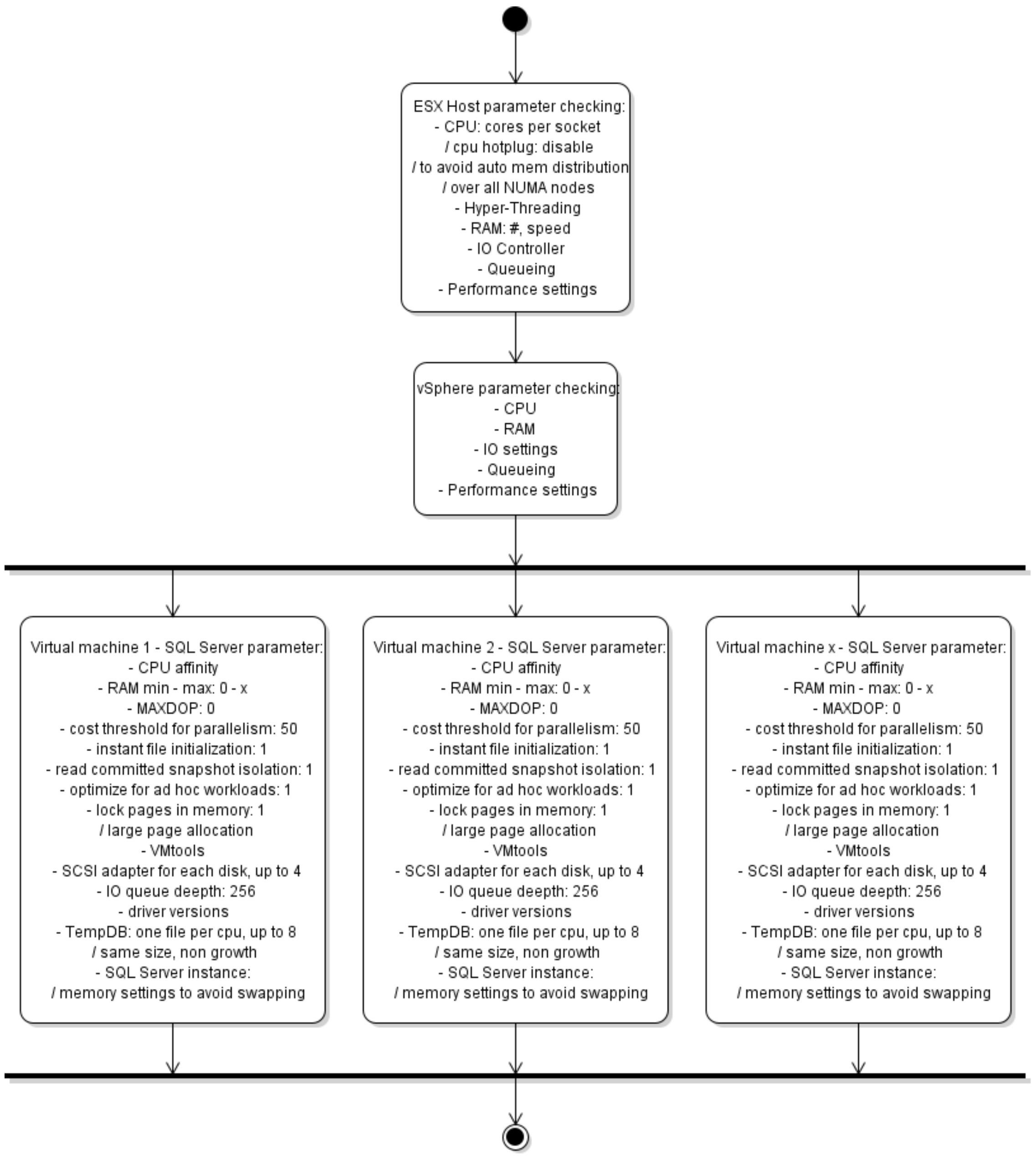


***Figure 23**: HammerDB TPC-C inofficial benchmark*

Figure 23 shows in comparison to figure 22 the results of a newer test server SQL Server 2012: The newer test server has a newer and faster database engine and newer and faster CPUs. Even if the two server are not comparable, the newer one should be faster in every case. The newer test server has Intel Xeon E5-2630 0 @ 2.30 GHz CPUs and is a virtual server under VMware. The one million insert benchmark finished in 36 seconds. This is also a very poor result compared to older legacy server.
Scaling problems caused by overcommitting or also called overprovisioning are supposed to be the cause.
Another test server running on a virtual VMware environment with newer CPUs Intel Xeon X5650 @ 2.67GHz (4 processors) 16 Core finish the one million rows insert benchmark in 21 second. This result will hold for years as the best result under virtual environments although newer machines becomes marginal faster. The reason for this observation is the marginal growing performance per CPU core. The following trend of CPU evolution is discoverable. The amount of CPU cores is increasing and the computing performance is growing at the same time by decreasing clock rate.

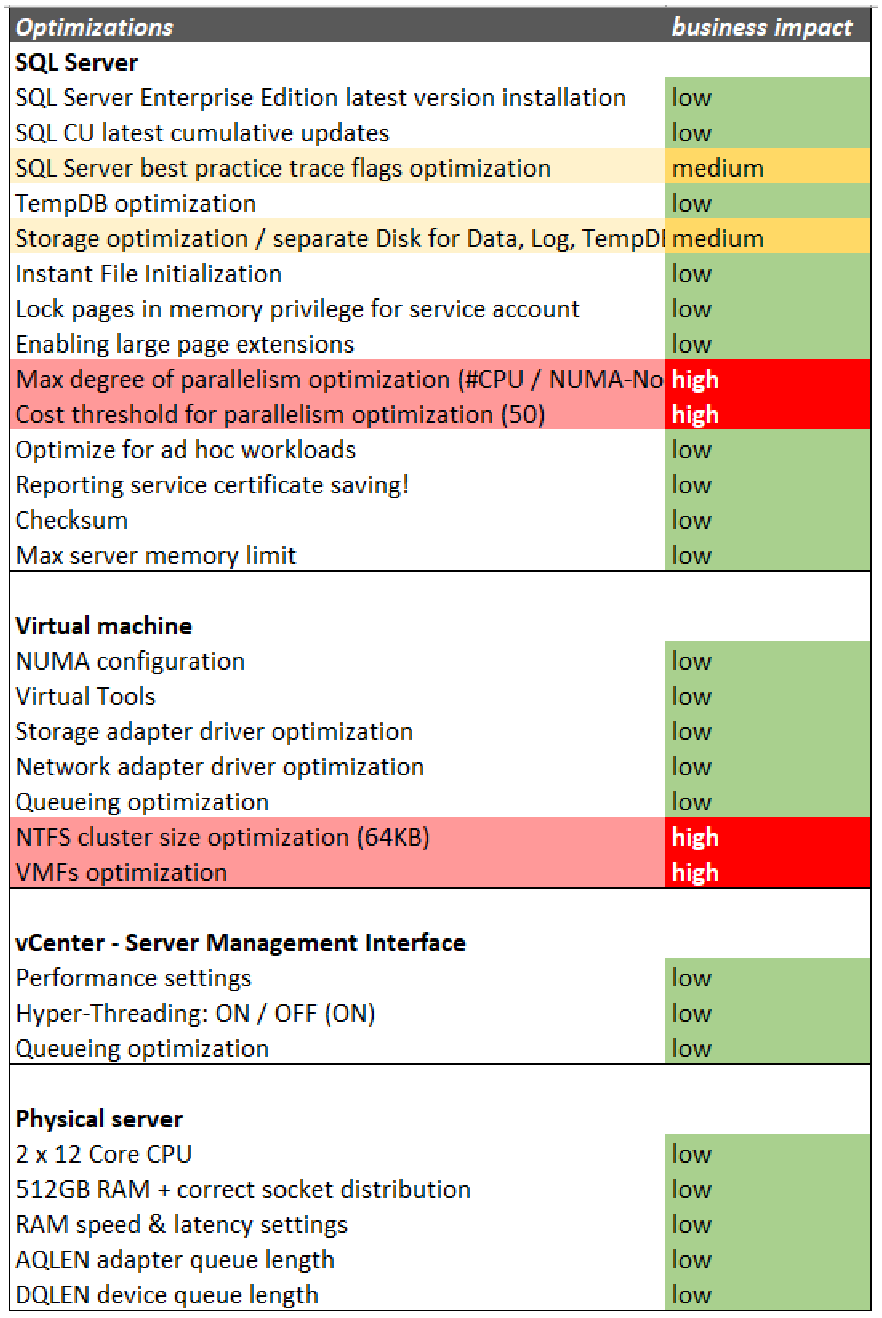
4.2.1 Best practice guide to ensure a performance base for SQL Server mapped in a UML activity diagram



ESX Host parameter checking:
- CPU: cores per socket
/ cpu hotplug: disable
/ to avoid auto mem distribution
/ over all NUMA nodes
- Hyper-Threading
- RAM: #, speed
- IO Controller
- Queueing
- Performance settings
vSphere parameter checking:
- CPU
- RAM
- IO settings
- Queueing
- Performance settings
Virtual machine 1 - SQL Server parameter:
- CPU affinity
- RAM min - max: 0 - x
- MAXDOP: 0
- cost threshold for parallelism: 50
- instant file initialization: 1
- read committed snapshot isolation: 1
- optimize for ad hoc workloads: 1
- lock pages in memory: 1
/ large page allocation
- VMtools
- SCSI adapter for each disk, up to 4
- IO queue deepth: 256
- driver versions
- TempDB: one file per cpu, up to 8
/ same size, non growth
- SQL Server instance:
/ memory settings to avoid swapping
Virtual machine 2 - SQL Server parameter:
- CPU affinity
- RAM min - max: 0 - x
- MAXDOP: 0
- cost threshold for parallelism: 50
- instant file initialization: 1
- read committed snapshot isolation: 1
- optimize for ad hoc workloads: 1
- lock pages in memory: 1
/ large page allocation
- VMtools
- SCSI adapter for each disk, up to 4
- IO queue deepth: 256
- driver versions
- TempDB: one file per cpu, up to 8
/ same size, non growth
- SQL Server instance:
/ memory settings to avoid swapping
Virtual machine x - SQL Server parameter:
- CPU affinity
- RAM min - max: 0 - x
- MAXDOP: 0
- cost threshold for parallelism: 50
- instant file initialization: 1
- read committed snapshot isolation: 1
- optimize for ad hoc workloads: 1
- lock pages in memory: 1
/ large page allocation
- VMtools
- SCSI adapter for each disk, up to 4
- IO queue deepth: 256
- driver versions
- TempDB: one file per cpu, up to 8
/ same size, non growth
- SQL Server instance:
/ memory settings to avoid swapping

***Figure 24**: Best practice guide to ensure a performance base for SQL Server*

Figure 24 shows a UML sequence diagram. It is a proved best practice guide to design a hardware with state of the art performance settings. The HOST hardware server should always have the latest processor technology. Each vendor has its own parameter for performance. The working memory in a CPU called Level 1, level 2 and level 3 cache is the fastest memory followed by the random access memory followed by the storage system. For most business critical applications or real time data ware house systems the storage system is an array of the fastest SSD drives on the market. Best practice settings guarantee the optimum and minimize the loss of performance between the involved hardware and software items.



Optimizations	*business impact*
SQL Server	
SQL Server Enterprise Edition latest version installation	low
SQL CU latest cumulative updates	low
SQL Server best practice trace flags optimization	medium
TempDB optimization	low
Storage optimization / separate Disk for Data, Log, TempDI	medium
Instant File Initialization	low
Lock pages in memory privilege for service account	low
Enabling large page extensions	low
Max degree of parallelism optimization (#CPU / NUMA-No	high
Cost threshold for parallelism optimization (50)	high
Optimize for ad hoc workloads	low
Reporting service certificate saving!	low
Checksum	low
Max server memory limit	low
Virtual machine	
NUMA configuration	low
Virtual Tools	low
Storage adapter driver optimization	low
Network adapter driver optimization	low
Queueing optimization	low
NTFS cluster size optimization (64KB)	high
VMFs optimization	high
vCenter - Server Management Interface	
Performance settings	low
Hyper-Threading: ON / OFF (ON)	low
Queueing optimization	low
Physical server	
2 x 12 Core CPU	low
512GB RAM + correct socket distribution	low
RAM speed & latency settings	low
AQLEN adapter queue length	low
DQLEN device queue length	low

Figure 25*: Best practice guide to ensure a performance base for SQL Server*

In figure 25 the activity diagram in figure 24 will be supplemented with the status of business impact and helps to plan the downtimes and priorities in detail.

To proof, demonstrate and clarify the boasted claim to provide a best practice guide for performance, an implementation of the best practice guide to ensure a performance base for SQL Server will be shown.

The hardware for testing is a state of the art notebook. SQL Server chosen is the newest version 2019 Preview announced these days on the beginning of Ignite on Monday 24 of September 2018. Testing platform chosen is Docker for Microsoft Windows 10 64-bit and the RedHat Linux docker container with SQL Server 2019 Preview. Management Console chosen is the latest standard Microsoft SQL Server Management Studio v18.0 Preview 4.

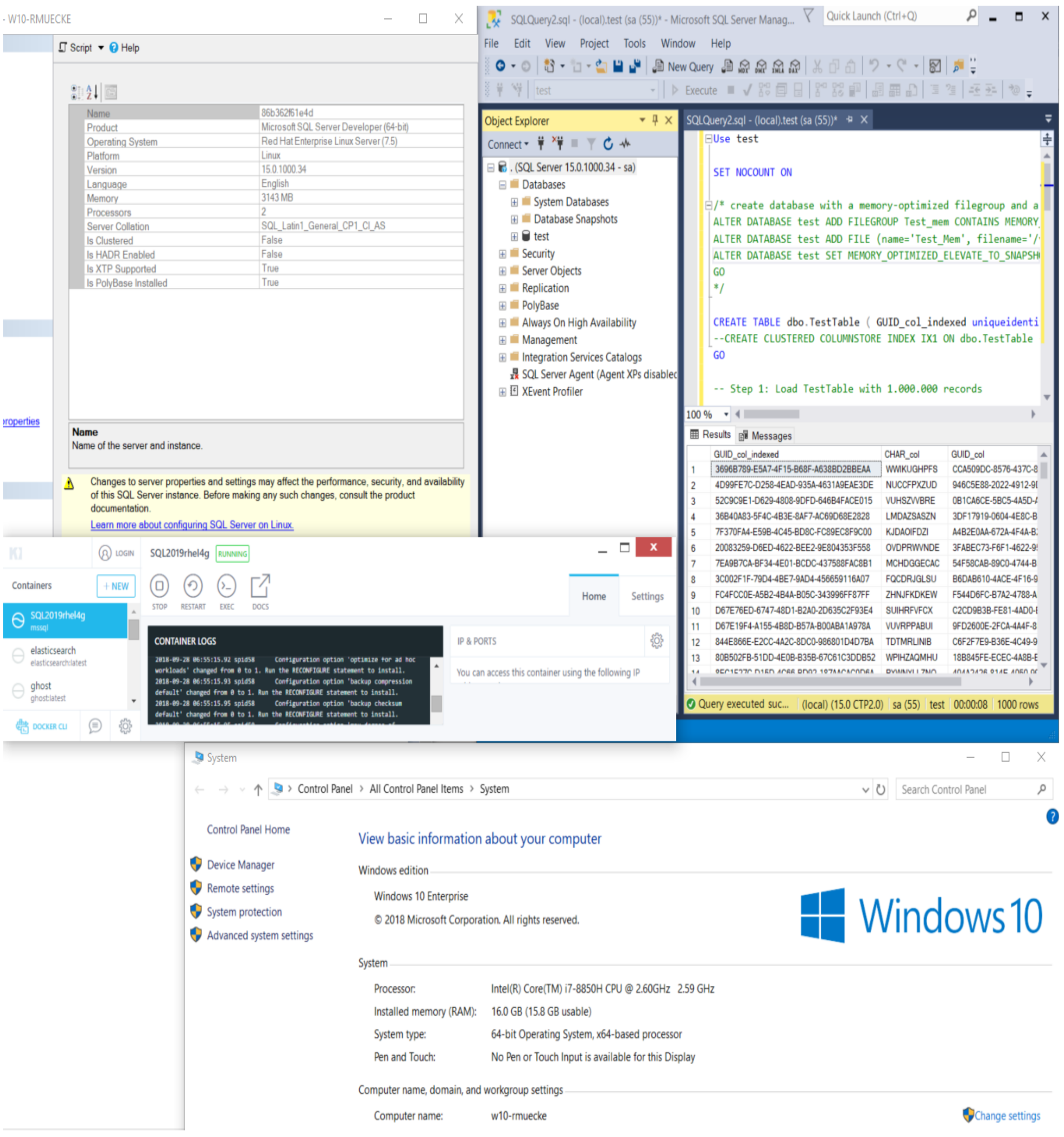
Stress Test scenario is the well-known and tested one million rows insert benchmark clarified in detail in the Appendix in this thesis. The benchmark script was especially over-worked for this benchmark scenario to test and show the possibility and performance of an in memory table in the new SQL Server.

The result is very surprising. The performance of a state of the art notebook is double the performance of a server a few years old. Out of the scope for this claim is the scalability. This means that a notebook never ever can be compared with a server. But the integer performance overall for ad-hoc queries is more as satisfying.

Figure 26 shows all involved software items. The base is the Docker software running the docker container providing the RedHat Linux with SQL Server 2019 Preview. The lastest SQL Server Management Studio was used to configure the SQL Server and execute the optimized T-SQL script on the SQL Server.

The T-SQL scripts are to be found in the Appendix.

The standard T-SQL benchmark script was executed and finished in 10 seconds. The T-SQL benchmark script optimized for in memory tables was executed and finished in only 7 seconds. For the optimized T-SQL benchmark script executed two times in parallel the SQL Server needed 14 seconds.



Name	86b362f61e4d
Product	Microsoft SQL Server Developer (64-bit)
Operating System	Red Hat Enterprise Linux Server (7.5)
Platform	Linux
Version	15.0.1000.34
Language	English
Memory	3143 MB
Processors	2
Server Collation	SQL_Latin1_General_CP1_CI_AS
Is Clustered	False
Is HADR Enabled	False
Is XTP Supported	True
Is PolyBase Installed	True

Name
Name of the server and instance.

Changes to server properties and settings may affect the performance, security, and availability of this SQL Server instance. Before making any such changes, consult the product documentation.
Learn more about configuring SQL Server on Linux.

SQLQuery2.sql - (local).test (sa (55))* - Microsoft SQL Server Manag...

```
Use test

SET NOCOUNT ON

/* create database with a memory-optimized filegroup and a
ALTER DATABASE test ADD FILEGROUP Test_mem CONTAINS MEMORY_
ALTER DATABASE test ADD FILE (name='Test_Mem', filename='/
ALTER DATABASE test SET MEMORY_OPTIMIZED_ELEVATE_TO_SNAPSH
GO
*/

CREATE TABLE dbo.TestTable ( GUID_col_indexed uniqueidenti
--CREATE CLUSTERED COLUMNSTORE INDEX IX1 ON dbo.TestTable
GO

-- Step 1: Load TestTable with 1.000.000 records
```

	GUID_col_indexed	CHAR_col	GUID_col
1	3696B789-E5A7-4F15-B68F-A638BD2BBEAA	WWIKUGHPFS	CCA509DC-8576-437C-8
2	4D99FE7C-D258-4EAD-935A-4631A9EAE3DE	NUCCFPXZUD	946C5E88-2022-4912-9
3	52C9C9E1-D629-4808-9DFD-646B4FACE015	VUHSZVVBRE	0B1CA6CE-5BC5-4A5D-
4	36B40A83-5F4C-4B3E-8AF7-AC69D68E2828	LMDAZSASZN	3DF17919-0604-4E8C-B
5	7F370FA4-E59B-4C45-BD8C-FC89EC8F9C00	KJDAOIFDZI	A4B2E0AA-672A-4F4A-B
6	20083259-D6ED-4622-BEE2-9E804353F558	OVDPRWVNDE	3FABEC73-F6F1-4622-9
7	7EA9B7CA-BF34-4E01-BCDC-437588FAC8B1	MCHDGGECAC	54F58CAB-89C0-4744-B
8	3C002F1F-79D4-4BE7-9AD4-456659116A07	FQCDRJGLSU	B6DAB610-4ACE-4F16-9
9	FC4FCC0E-A5B2-4B4A-B05C-343996FF87FF	ZHNJFKDKEW	F544D6FC-B7A2-4788-A
10	D67E76ED-6747-48D1-B2A0-2D635C2F93E4	SUIHRFVFCX	C2CD9B3B-FE81-4AD0-
11	D67E19F4-A155-4B8D-B57A-B00ABA1A978A	VUVRPPABUI	9FD2600E-2FCA-4A4F-8
12	844E866E-E2CC-4A2C-8DC0-986801D4D7BA	TDTMRLINIB	C6F2F7E9-B36E-4C49-9
13	80B502FB-51DD-4E0B-B35B-67C61C3DDB52	WPIHZAQMHU	18B845FE-ECEC-4A8B-

Query executed suc... | (local) (15.0 CTP2.0) | sa (55) | test | 00:00:08 | 1000 rows

SQL2019rhel4g RUNNING

CONTAINER LOGS

```
2018-09-28 06:55:15.92 spid58      Configuration option 'optimize for ad hoc
workloads' changed from 0 to 1. Run the RECONFIGURE statement to install.
2018-09-28 06:55:15.93 spid58      Configuration option 'backup compression
default' changed from 0 to 1. Run the RECONFIGURE statement to install.
2018-09-28 06:55:15.95 spid58      Configuration option 'backup checksum
default' changed from 0 to 1. Run the RECONFIGURE statement to install.
```

You can access this container using the following IP

View basic information about your computer

Windows edition

Windows 10 Enterprise

© 2018 Microsoft Corporation. All rights reserved.

System

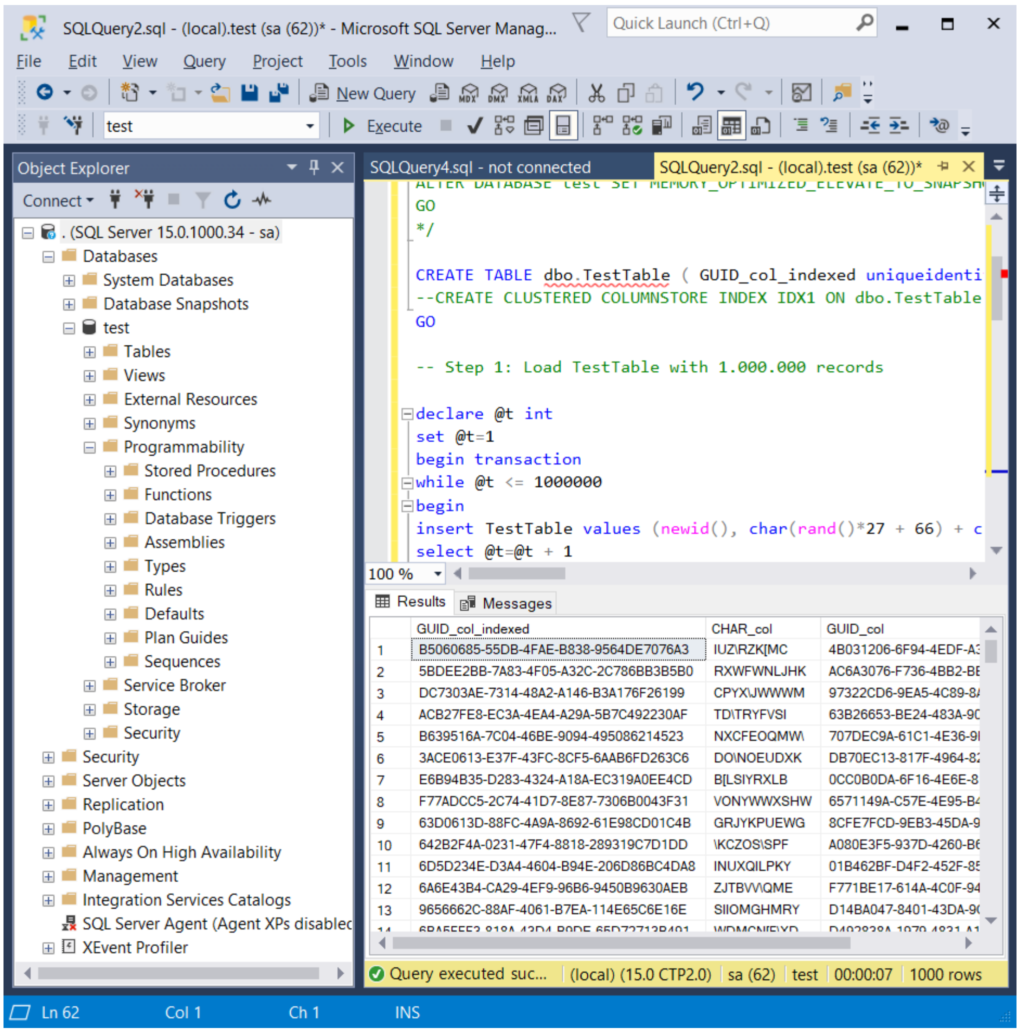
Processor:	Intel(R) Core(TM) i7-8850H CPU @ 2.60GHz 2.59 GHz
Installed memory (RAM):	16.0 GB (15.8 GB usable)
System type:	64-bit Operating System, x64-based processor
Pen and Touch:	No Pen or Touch Input is available for this Display

Computer name, domain, and workgroup settings

Computer name:	w10-rmuecke

Figure 26: Best practice guide to ensure a performance base for SQL Server proven in praxis

The hardware system and also the database engine has been optimized as defined in figure 24 and figure 25 to offer maximum performance and to minimize queueing.



```
GO
*/

CREATE TABLE dbo.TestTable ( GUID_col_indexed uniqueidenti
--CREATE CLUSTERED COLUMNSTORE INDEX IDX1 ON dbo.TestTable
GO

-- Step 1: Load TestTable with 1.000.000 records

declare @t int
set @t=1
begin transaction
while @t <= 1000000
begin
insert TestTable values (newid(), char(rand()*27 + 66) + c
select @t=@t + 1
```

	GUID_col_indexed	CHAR_col	GUID_col
1	B5060685-55DB-4FAE-B838-9564DE7076A3	IUZ\RZK[MC	4B031206-6F94-4EDF-A3
2	5BDEE2BB-7A83-4F05-A32C-2C786BB3B5B0	RXWFWNLJHK	AC6A3076-F736-4BB2-BE
3	DC7303AE-7314-48A2-A146-B3A176F26199	CPYX\JWWWM	97322CD6-9EA5-4C89-8A
4	ACB27FE8-EC3A-4EA4-A29A-5B7C492230AF	TD\TRYFVSI	63B26653-BE24-483A-9C
5	B639516A-7C04-46BE-9094-495086214523	NXCFEOQMW\	707DEC9A-61C1-4E36-9I
6	3ACE0613-E37F-43FC-8CF5-6AAB6FD263C6	DO\NOEUDXK	DB70EC13-817F-4964-82
7	E6B94B35-D283-4324-A18A-EC319A0EE4CD	B[LSIYRXLB	0CC0B0DA-6F16-4E6E-8
8	F77ADCC5-2C74-41D7-8E87-7306B0043F31	VONYWWXSHW	6571149A-C57E-4E95-B4
9	63D0613D-88FC-4A9A-8692-61E98CD01C4B	GRJYKPUEWG	8CFE7FCD-9EB3-45DA-9
10	642B2F4A-0231-47F4-8818-289319C7D1DD	\KCZOS\SPF	A080E3F5-937D-4260-B6
11	6D5D234E-D3A4-4604-B94E-206D86BC4DA8	INUXQILPKY	01B462BF-D4F2-452F-85
12	6A6E43B4-CA29-4EF9-96B6-9450B9630AEB	ZJTBVV\QME	F771BE17-614A-4C0F-94
13	9656662C-88AF-4061-B7EA-114E65C6E16E	SIIOMGHMRY	D14BA047-8401-43DA-9C

Query executed suc... | (local) (15.0 CTP2.0) | sa (62) | test | 00:00:07 | 1000 rows

Figure 27*: Best practice guide to ensure a performance base for SQL Server proven in praxis, detailed benchmark result*

Figure 27 shows a detailed window with the benchmark result of 7 seconds as shown in figure 26.

4.3 Scaling problems under virtual environments and case study

The following case scenarios and the problems occurring therein are real world problem scenarios in the described ITIL environment.

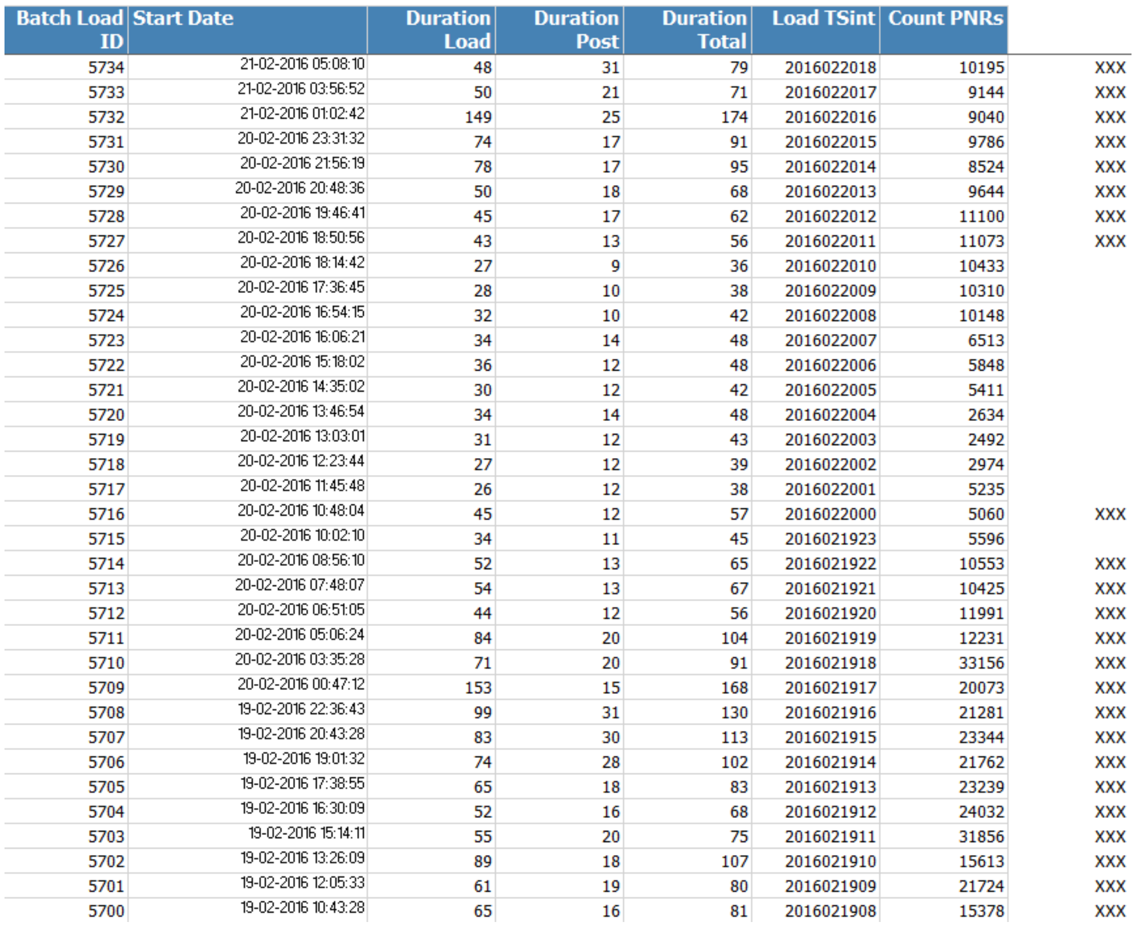
4.3.1 Case scenario 1: Overcommitting or overprovisioning

A virtual environment is known to consume overhead costs. This means that a virtual processor in a virtual machine cannot provide its full performance like it does on a physical machine. In general, such overhead costs are about a few percent.

Overcommitting or overprovisioning happens if one or several virtual machines are taking more count of CPU or more memory as the underlying host server can offer.
For example, if a host server has four CPUs and 32GB of memory and have to serve two virtual machines with four CPUs and 32GB for each as well.
In most cases this will be no problem in daily work, if neither of the two virtual machines will be operated at its performance limits.
If the CPU will be overcommitted, heavy wait times will occur. In worst cases timeouts or inacceptable application behavior will occur. If the memory will be overcommitted, heavy swapping will occur. In worst cases the virtual machine will not response anymore and have to be rebooted.

4.3.2 Case scenario 2: Scaling problems on IO sub-system

It is very hard to find scaling problems on a storage system when all is fine and up and running. All main performance counter seems to be all right. No errors will occur in any log file. Only the notations of the application manager show deviations respectively spikes in runtime behavior of a data warehouse load.



Batch Load ID	Start Date	Duration Load	Duration Post	Duration Total	Load TSint	Count PNRs	
5734	21-02-2016 05:08:10	48	31	79	2016022018	10195	XXX
5733	21-02-2016 03:56:52	50	21	71	2016022017	9144	XXX
5732	21-02-2016 01:02:42	149	25	174	2016022016	9040	XXX
5731	20-02-2016 23:31:32	74	17	91	2016022015	9786	XXX
5730	20-02-2016 21:56:19	78	17	95	2016022014	8524	XXX
5729	20-02-2016 20:48:36	50	18	68	2016022013	9644	XXX
5728	20-02-2016 19:46:41	45	17	62	2016022012	11100	XXX
5727	20-02-2016 18:50:56	43	13	56	2016022011	11073	XXX
5726	20-02-2016 18:14:42	27	9	36	2016022010	10433	
5725	20-02-2016 17:36:45	28	10	38	2016022009	10310	
5724	20-02-2016 16:54:15	32	10	42	2016022008	10148	
5723	20-02-2016 16:06:21	34	14	48	2016022007	6513	
5722	20-02-2016 15:18:02	36	12	48	2016022006	5848	
5721	20-02-2016 14:35:02	30	12	42	2016022005	5411	
5720	20-02-2016 13:46:54	34	14	48	2016022004	2634	
5719	20-02-2016 13:03:01	31	12	43	2016022003	2492	
5718	20-02-2016 12:23:44	27	12	39	2016022002	2974	
5717	20-02-2016 11:45:48	26	12	38	2016022001	5235	
5716	20-02-2016 10:48:04	45	12	57	2016022000	5060	XXX
5715	20-02-2016 10:02:10	34	11	45	2016021923	5596	
5714	20-02-2016 08:56:10	52	13	65	2016021922	10553	XXX
5713	20-02-2016 07:48:07	54	13	67	2016021921	10425	XXX
5712	20-02-2016 06:51:05	44	12	56	2016021920	11991	XXX
5711	20-02-2016 05:06:24	84	20	104	2016021919	12231	XXX
5710	20-02-2016 03:35:28	71	20	91	2016021918	33156	XXX
5709	20-02-2016 00:47:12	153	15	168	2016021917	20073	XXX
5708	19-02-2016 22:36:43	99	31	130	2016021916	21281	XXX
5707	19-02-2016 20:43:28	83	30	113	2016021915	23344	XXX
5706	19-02-2016 19:01:32	74	28	102	2016021914	21762	XXX
5705	19-02-2016 17:38:55	65	18	83	2016021913	23239	XXX
5704	19-02-2016 16:30:09	52	16	68	2016021912	24032	XXX
5703	19-02-2016 15:14:11	55	20	75	2016021911	31856	XXX
5702	19-02-2016 13:26:09	89	18	107	2016021910	15613	XXX
5701	19-02-2016 12:05:33	61	19	80	2016021909	21724	XXX
5700	19-02-2016 10:43:28	65	16	81	2016021908	15378	XXX

Figure 28*: Best practice to ensure a performance base in SQL Server*

Figure shows spikes in runtime behavior in the column “Duration Total”.

Nothing has changed in the application and nothing has changed in the loading process.
Now it is time to put some stress to the server and do a few tests to check the server health.
The slowest item in a server is the disk subsystem.

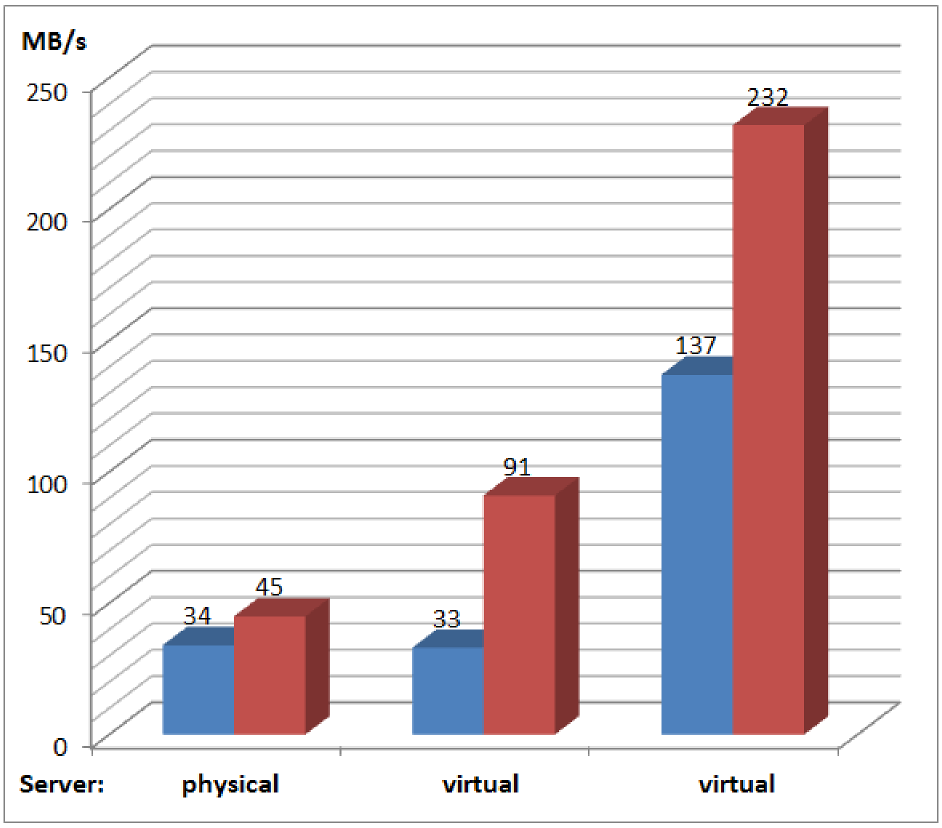


***Figure 29**: scaling problems on IO sub system*

To find the bottleneck also the IO sub-system needs analysis.
A very simple and not time-consuming benchmark is copying a reference data file. Although the three server are different and the three IO sub-systems are different too, a copy benchmark is a good and meaningful test tool. The reference data file in this case is a small database backup file with 5.7GB of size. The file copied once and then four times on the same server within the same IO sub-system.
The calculated data transfer rates form the basis to compare all copy processes.

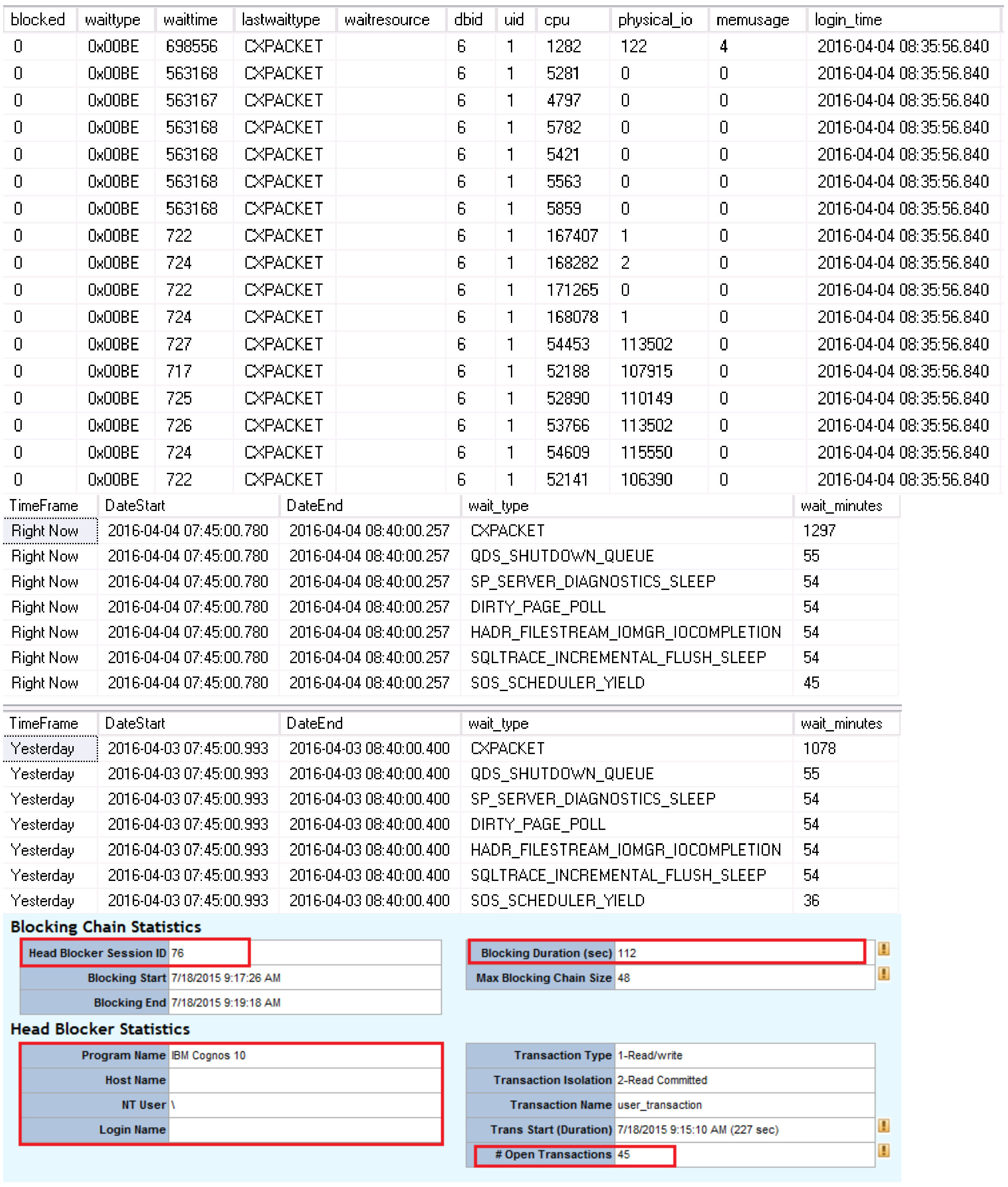
In this case, the problem causer becomes apparently. On the physical server, the data transfer rate for copying the reference file once is 45MB/s compared with 34MB/s when the reference file copied four times concurrent. Because of the known characteristic performance and scalability of the IO sub-system, the two results should be near identical but a decreased performance of 75% is acceptable.

The negative performance impact on both virtual machines is measurably bigger with 36% and 59% data transfer rate compared to the single copy job.

A look at the driver and settings of the IO controller in the virtual environment debunk the problem causer. False settings and driver versions in combination with overcommitting in virtual environments can lead to such performance problems hard to find.

4.3.3 Case scenario 3: Parallelism and skewed parallelism

There are many ways to use parallelism and combinations of such ways. The application can force using parallelism with query options or co called query hints. The SQL Server engine can be optimized for parallelism with parameter. In SQL Server one task will be split into a lot of tasks so called threads. The threads they are waiting to be merged are marked with the wait type CXPACKET. In newer versions like SQL Server 2016 SP2 and SQL Server 2017 RTM CU3 consumer threads are no longer registered as CXPACKET waits, they are called CXCONSUMER waits.



blocked	waittype	waittime	lastwaittype	waitresource	dbid	uid	cpu	physical_io	memusage	login_time
0	0x00BE	698556	CXPACKET		6	1	1282	122	4	2016-04-04 08:35:56.840
0	0x00BE	563168	CXPACKET		6	1	5281	0	0	2016-04-04 08:35:56.840
0	0x00BE	563167	CXPACKET		6	1	4797	0	0	2016-04-04 08:35:56.840
0	0x00BE	563168	CXPACKET		6	1	5782	0	0	2016-04-04 08:35:56.840
0	0x00BE	563168	CXPACKET		6	1	5421	0	0	2016-04-04 08:35:56.840
0	0x00BE	563168	CXPACKET		6	1	5563	0	0	2016-04-04 08:35:56.840
0	0x00BE	563168	CXPACKET		6	1	5859	0	0	2016-04-04 08:35:56.840
0	0x00BE	722	CXPACKET		6	1	167407	1	0	2016-04-04 08:35:56.840
0	0x00BE	724	CXPACKET		6	1	168282	2	0	2016-04-04 08:35:56.840
0	0x00BE	722	CXPACKET		6	1	171265	0	0	2016-04-04 08:35:56.840
0	0x00BE	724	CXPACKET		6	1	168078	1	0	2016-04-04 08:35:56.840
0	0x00BE	727	CXPACKET		6	1	54453	113502	0	2016-04-04 08:35:56.840
0	0x00BE	717	CXPACKET		6	1	52188	107915	0	2016-04-04 08:35:56.840
0	0x00BE	725	CXPACKET		6	1	52890	110149	0	2016-04-04 08:35:56.840
0	0x00BE	726	CXPACKET		6	1	53766	113502	0	2016-04-04 08:35:56.840
0	0x00BE	724	CXPACKET		6	1	54609	115550	0	2016-04-04 08:35:56.840
0	0x00BE	722	CXPACKET		6	1	52141	106390	0	2016-04-04 08:35:56.840

TimeFrame	DateStart	DateEnd	wait_type	wait_minutes
Right Now	2016-04-04 07:45:00.780	2016-04-04 08:40:00.257	CXPACKET	1297
Right Now	2016-04-04 07:45:00.780	2016-04-04 08:40:00.257	QDS_SHUTDOWN_QUEUE	55
Right Now	2016-04-04 07:45:00.780	2016-04-04 08:40:00.257	SP_SERVER_DIAGNOSTICS_SLEEP	54
Right Now	2016-04-04 07:45:00.780	2016-04-04 08:40:00.257	DIRTY_PAGE_POLL	54
Right Now	2016-04-04 07:45:00.780	2016-04-04 08:40:00.257	HADR_FILESTREAM_IOMGR_IOCOMPLETION	54
Right Now	2016-04-04 07:45:00.780	2016-04-04 08:40:00.257	SQLTRACE_INCREMENTAL_FLUSH_SLEEP	54
Right Now	2016-04-04 07:45:00.780	2016-04-04 08:40:00.257	SOS_SCHEDULER_YIELD	45

TimeFrame	DateStart	DateEnd	wait_type	wait_minutes
Yesterday	2016-04-03 07:45:00.993	2016-04-03 08:40:00.400	CXPACKET	1078
Yesterday	2016-04-03 07:45:00.993	2016-04-03 08:40:00.400	QDS_SHUTDOWN_QUEUE	55
Yesterday	2016-04-03 07:45:00.993	2016-04-03 08:40:00.400	SP_SERVER_DIAGNOSTICS_SLEEP	54
Yesterday	2016-04-03 07:45:00.993	2016-04-03 08:40:00.400	DIRTY_PAGE_POLL	54
Yesterday	2016-04-03 07:45:00.993	2016-04-03 08:40:00.400	HADR_FILESTREAM_IOMGR_IOCOMPLETION	54
Yesterday	2016-04-03 07:45:00.993	2016-04-03 08:40:00.400	SQLTRACE_INCREMENTAL_FLUSH_SLEEP	54
Yesterday	2016-04-03 07:45:00.993	2016-04-03 08:40:00.400	SOS_SCHEDULER_YIELD	36

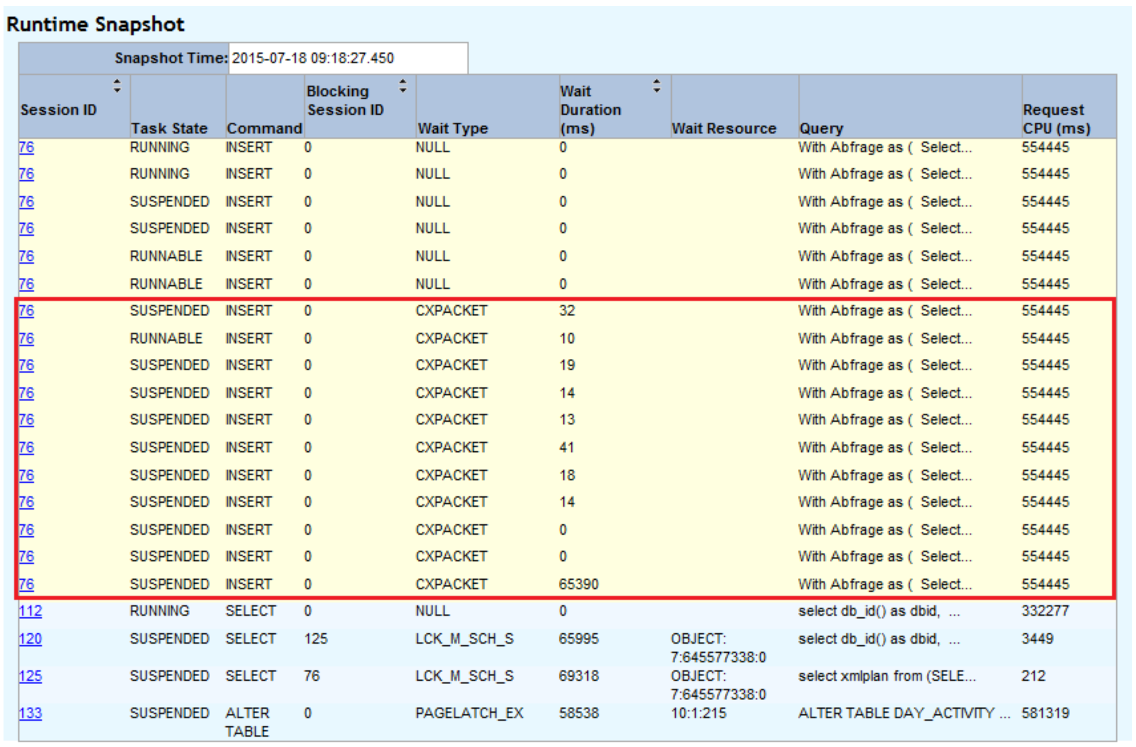
Blocking Chain Statistics

Head Blocker Session ID	76	Blocking Duration (sec)	112
Blocking Start	7/18/2015 9:17:26 AM	Max Blocking Chain Size	48
Blocking End	7/18/2015 9:19:18 AM		

Head Blocker Statistics

Program Name	IBM Cognos 10	Transaction Type	1-Read/write
Host Name		Transaction Isolation	2-Read Committed
NT User	\	Transaction Name	user_transaction
Login Name		Trans Start (Duration)	7/18/2015 9:15:10 AM (227 sec)
		# Open Transactions	45

***Figure 30**: Best practice to ensure a performance base in SQL Server*



Runtime Snapshot

Snapshot Time: 2015-07-18 09:18:27.450

Session ID	Task State	Command	Blocking Session ID	Wait Type	Wait Duration (ms)	Wait Resource	Query	Request CPU (ms)
76	RUNNING	INSERT	0	NULL	0		With Abfrage as (Select...	554445
76	RUNNING	INSERT	0	NULL	0		With Abfrage as (Select...	554445
76	SUSPENDED	INSERT	0	NULL	0		With Abfrage as (Select...	554445
76	SUSPENDED	INSERT	0	NULL	0		With Abfrage as (Select...	554445
76	RUNNABLE	INSERT	0	NULL	0		With Abfrage as (Select...	554445
76	RUNNABLE	INSERT	0	NULL	0		With Abfrage as (Select...	554445
76	SUSPENDED	INSERT	0	CXPACKET	32		With Abfrage as (Select...	554445
76	RUNNABLE	INSERT	0	CXPACKET	10		With Abfrage as (Select...	554445
76	SUSPENDED	INSERT	0	CXPACKET	19		With Abfrage as (Select...	554445
76	SUSPENDED	INSERT	0	CXPACKET	14		With Abfrage as (Select...	554445
76	SUSPENDED	INSERT	0	CXPACKET	13		With Abfrage as (Select...	554445
76	SUSPENDED	INSERT	0	CXPACKET	41		With Abfrage as (Select...	554445
76	SUSPENDED	INSERT	0	CXPACKET	18		With Abfrage as (Select...	554445
76	SUSPENDED	INSERT	0	CXPACKET	14		With Abfrage as (Select...	554445
76	SUSPENDED	INSERT	0	CXPACKET	0		With Abfrage as (Select...	554445
76	SUSPENDED	INSERT	0	CXPACKET	0		With Abfrage as (Select...	554445
76	SUSPENDED	INSERT	0	CXPACKET	65390		With Abfrage as (Select...	554445
112	RUNNING	SELECT	0	NULL	0		select db_id() as dbid, ...	332277
120	SUSPENDED	SELECT	125	LCK_M_SCH_S	65995	OBJECT: 7:645577338:0	select db_id() as dbid, ...	3449
125	SUSPENDED	SELECT	76	LCK_M_SCH_S	69318	OBJECT: 7:645577338:0	select xmlplan from (SELE...	212
133	SUSPENDED	ALTER TABLE	0	PAGELATCH_EX	58538	10:1:215	ALTER TABLE DAY_ACTIVITY ...	581319

***Figure 31**: Best practice to ensure a performance base in SQL*

Huge insert statements not running optimal parallel shown in figure 31.

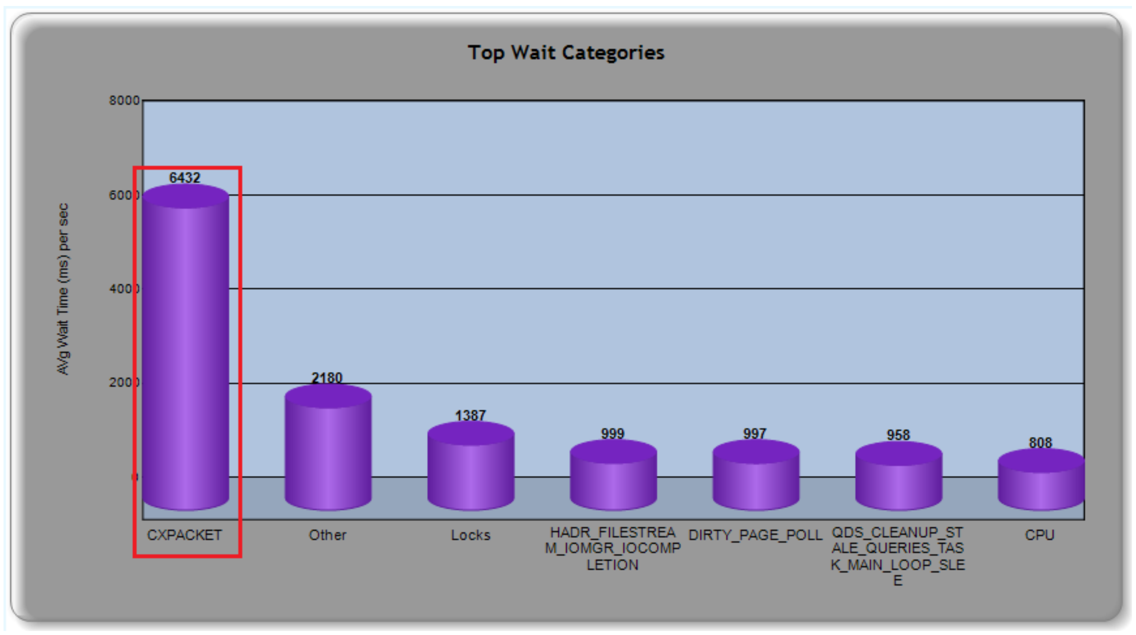


***Figure 32**: waiting parallel threads for main process*

If statistical information about the table content is not up to date the database engine calculates false costs for the execution plan because the estimated costs are not the real costs. In case of a higher count of data rows in a table as the table really holds the optimizer will give one thread all work and all other threads are waiting with the wait type CXPACKET for instance. This is called skewed parallelism.
Figure 30, figure 31 and figure 32 showing a lot of waiting threads coming from huge select statements, huge insert statements and huge update statements automatically generated with the IBM Cognos reporting software.

A lot of threads they are waiting can be optimized with the right setting of the SQL Server parameter *'max degree of parallelism'* and additional in direct relationship with the parameter *'cost threshold for parallelism'*.

Therefore, it is necessary to evaluate the most occurring queries and the most expensive queries. A good tool, coming with the SQL Server, is the SQL Server profiler. With the aid of the SQL Profiler such problematic queries can be found. Another good approach is the use of *'Extended events'* in the SQL Server engine or T-SQL scripts to find the most expensive statements.
It is often very effective to set the SQL Server engine parameter *'cost threshold for parallelism'* especially to a higher value as default. This prevents queries with an estimated runtime below the value of the parameter *'cost threshold for parallelism'* of running parallel. Only expensive queries with an estimated runtime above the parameter *'cost threshold for parallelism'* were executed parallel.

A main task of a DBA is often query rewriting for performance optimization.

4.3.4 Case scenario 4: SQL Server engine tuning

SQL Server engine performance optimization in a newer version are common. Some technological innovation is superior and deserve attention like the new *CheckScanner* design in SQL Server engine 2016. Compared to previous SQL Server engines with *MultiObjectScanner* design the newer design scales 7 times better. So a DBCC scales 7x better and shrinks the maintenance window dramatically which frees a lot of necessary time for other nightly jobs like backup or bulk loads for data warehouse reasons.

There exist many requests about slow job runtimes or poor performing long running queries.

Back to the problem of the poor performing data warehouse SQL Server. The following figure shows the importance of memory for the SQL Server.

A very important performance counter is page life expectancy, shortened PLE. PLE should never be less than 300 seconds in general or for each NUMA CPU. In many discussions, a value of 300 seconds or lower describes a memory pressure problem. There is a lot of controversy about the Buffer Manager Performance object counter Page Life Expectancy. This is a controversial, but versatile discussed predication. There are many different views about this performance counter. A simple DBCC CHECKDB command, responsible to proof a database, can cause this performance counter to fall and predict serious performance troubles to the whole system but there is no hard problem. However, in most cases, it is a good tool to determine serious memory pressure.
Following query results the value for PLE.

```
SELECT [object_name],
[counter_name],
[cntr_value] FROM sys.dm_os_performance_counters
WHERE [object_name] LIKE '%Manager%'
AND [counter_name] = 'Page life expectancy'
```

Following query results the value for CPU Pressure. It shows the signal_wait_time percentage. The value should not exceed 4%. Values above 4% or often 20% indicates CPU pressure. The SQL Server need one or more CPUs to gain performance.

```
SELECT signal_wait_time_ms=SUM(signal_wait_time_ms)
,'%signal (cpu) waits' = CAST(100.0 * SUM(signal_wait_time_ms) / SUM
(wait_time_ms) AS NUMERIC(20,2))
,resource_wait_time_ms=SUM(wait_time_ms - signal_wait_time_ms)
,'%resource waits'= CAST(100.0 * SUM(wait_time_ms - signal_wait_time_ms) /
SUM (wait_time_ms) AS NUMERIC(20,2))
FROM sys.dm_os_wait_stats
```

Since symmetric multiprocessing (SMP) has been very popular over the last years, many improvements were made to parallel processing. One important improvement for the SQL Server is the breakdown of processor architecture in hardware NUMA nodes. A couple of processors have his own front side bus

with his own local memory. This concept allows faster memory access. Each processor also has access to non-local memory from other NUMA nodes by lack of performance.
Calculating the right PLE value on machines with four NUMA nodes for example with caution. Page life expectancy is calculated by adding the PLE for each NUMA buffer pool and then calculating the harmonic mean, not the arithmetic mean.

The arithmetic mean of three values 2, 3 and 6 for example is 3.6 periodical. The harmonic mean of these three values is 3. It seems the two results are comparable.

$$\left(\frac{2^{-1}+3^{-1}+6^{-1}}{3}\right)^{-1} = \frac{3}{\frac{1}{2}+\frac{1}{3}+\frac{1}{6}} = \frac{3}{1} = 3.$$

***Figure 33**: harmonic mean*

The following example shows the necessity to calculate the right PLE with harmonic mean and not arithmetic mean, for instance on a machine with 4 NUMA nodes and a PLE of each being 4000.

(4000+4000+4000+4000)/4 = 4000 PLE, calculated with arithmetic mean.
4/(1/4000+1/4000+1/4000+1/4000) = 4000 PLE, calculated with harmonic mean.

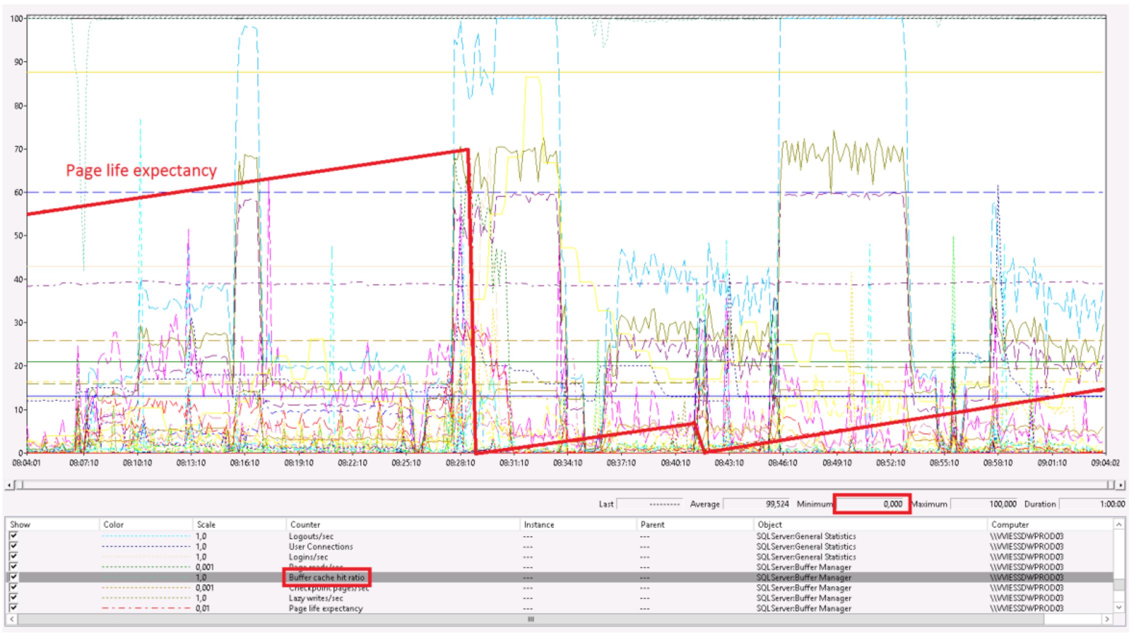
The results are identical if there is no memory pressure on any of the four NUMA nodes. Nevertheless, how does it look, if one NUMA node has memory pressure and the PLE falls to 100 seconds.

(100+4000+4000+4000)/4 = 3025 PLE, calculated with arithmetic mean.
4/(1/100+1/4000+1/4000+1/4000) = 372 PLE, calculated with harmonic mean.

The overall low PLE result leads to overreacting to performance issues on all NUMA nodes, but only one NUMA node is affected. Best practice is to check all Buffer Node: PLE counters for all NUMA nodes, and have a look at the lazywriter threads in sys.dm_exec_requests.

Every row from the result of the following T-SQL script stands for the page life expectancy of a single NUMA node. This helps to calculate the real page life expectancy.

```sql
SELECT *
FROM sys.dm_os_performance_counters
WHERE object_name = ‘SQLServer:Buffer Node’
AND counter_name = ‘Page life expectancy’
```



Last ---------- Average 99,524 Minimum 0,000 Maximum 100,000 Duration 1:00:00

Show	Color	Scale	Counter	Instance	Parent	Object	Computer
✓	┄	1,0	Logouts/sec	---	---	SQLServer:General Statistics	\\WIESSDWPROD03
✓	┄	1,0	User Connections	---	---	SQLServer:General Statistics	\\WIESSDWPROD03
✓	┄	1,0	Logins/sec	---	---	SQLServer:General Statistics	\\WIESSDWPROD03
✓	┄	0,001	Page reads/sec	---	---	SQLServer:Buffer Manager	\\WIESSDWPROD03
✓	┄	1,0	Buffer cache hit ratio	---	---	SQLServer:Buffer Manager	\\WIESSDWPROD03
✓	┄	0,001	Checkpoint pages/sec	---	---	SQLServer:Buffer Manager	\\WIESSDWPROD03
✓	┄	1,0	Lazy writes/sec	---	---	SQLServer:Buffer Manager	\\WIESSDWPROD03
✓	┄	0,01	Page life expectancy	---	---	SQLServer:Buffer Manager	\\WIESSDWPROD03

***Figure 34**: performance monitor*

Figure 34 highlights a real memory problem. Real observation of the whole system can give a clue about the real problem. The minimum, average and maximum values for the performance counter buffer cache hit ratio should always be around 99% then the SQL Server runs healthy. A minimum value of 0 in combination with value 0 for the page life expectancy demonstrates a massive performance problem forming memory pressure. Such dramatically memory pressure needs doubling the memory for the SQL Server.

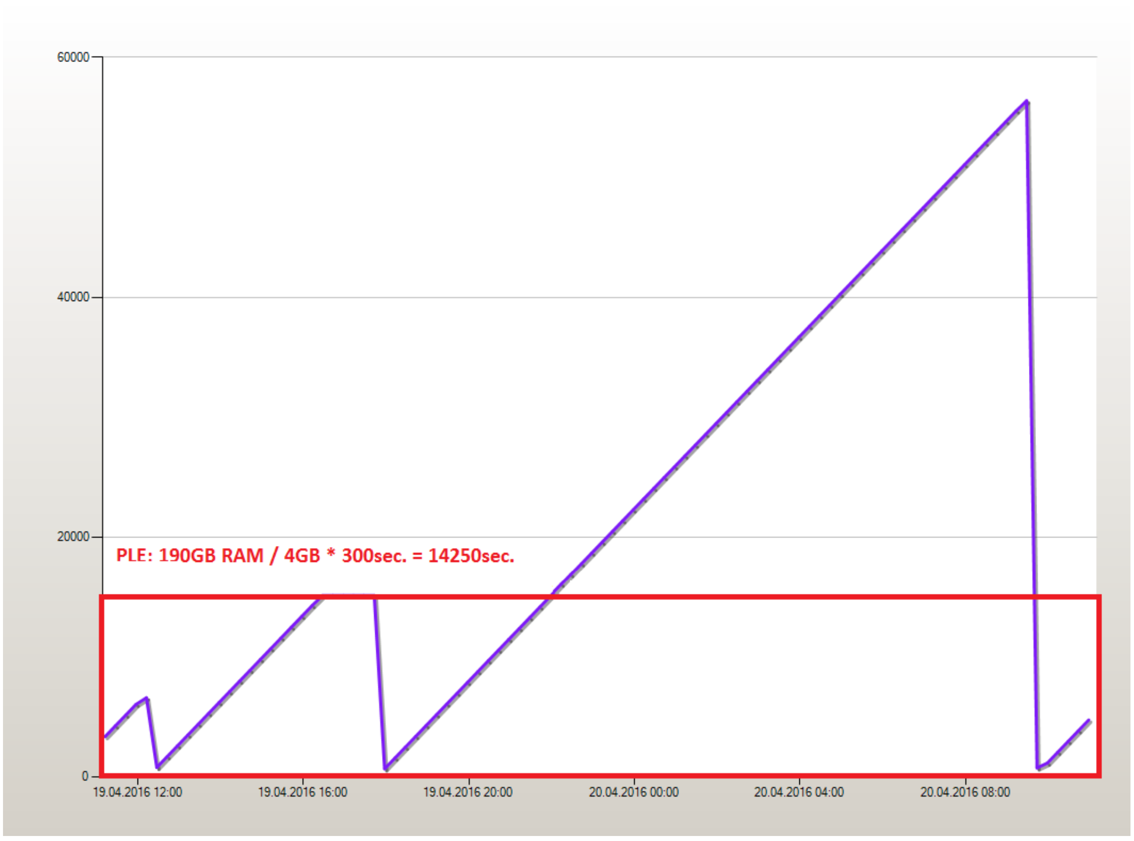


Figure 35*: performance counter PLE*

Figure 35 shows the calculated healthy working range for the page life expectancy parameter. The blue line should always be above the red square. In big data warehouse environments where huge data loads or bulk inserts happens, the page life expectancy value can sink to a very low value but this is a known behavior and needs no further attention because when the load is done the pages in memory will be buffered as usual. The memory size should be dimensioned in a tolerant way that such huge loads do not have an impact for other planned processes in the SQL Server.
Naturally it is better that the value for the PLE metric is high and steady, therefore things that can make PLE metric drop like index rebuilds, DBCC CHECKDB, queries with large memory grants or queries that modify a lot of pages forcing them to get flushed to disk have to be optimal scheduled to avoid self-made memory problems.

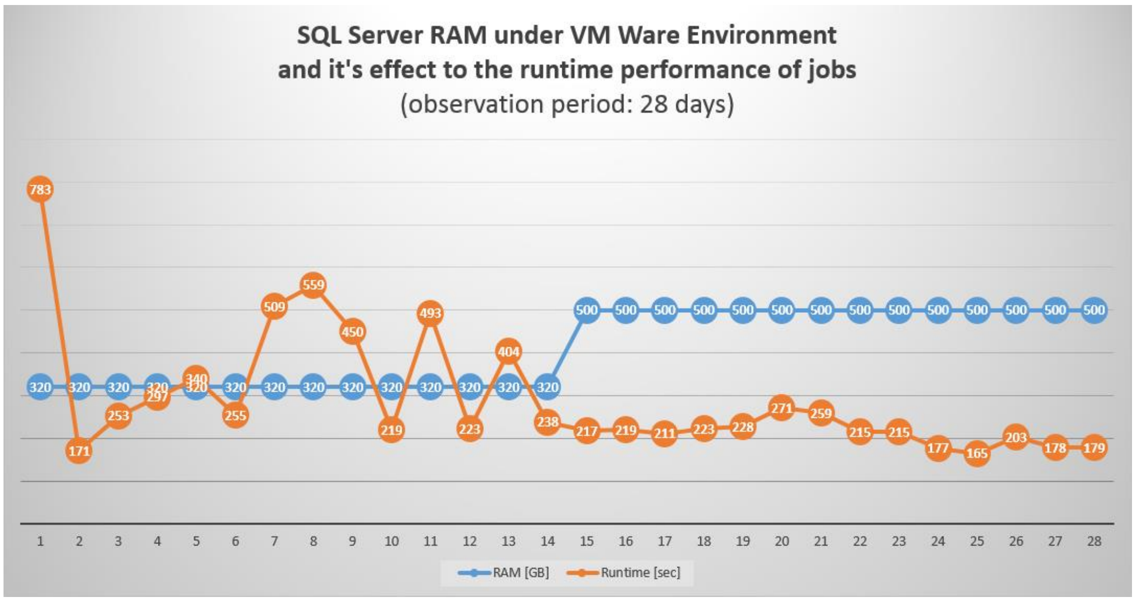


***Figure 36**: nightly run duration and RAM impact*

The figure 36 above shows the runtime of a data warehouse job in seconds multiplied by the factor one hundred. The last job in the observation period on observation day 28 needed 17941 seconds or by other words 5 hours. This is acceptable and guarantees that the nightly run is done the following tomorrow. Spike values like 78343 let the job take 21 hours and 46 minutes to be done and this time is not acceptable for a nightly run.

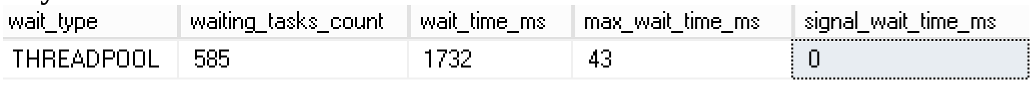
Performance problems estimated in cloud environments because of latency per transaction.

Interfaces to other applications on different operating systems and different environments need especially attention.

The DBA has in depth knowledge about query tuning and can give substantiated tuning tips to developer. Often small tips like avoiding cursors in T-SQL scripting helps to avoid performance problems.

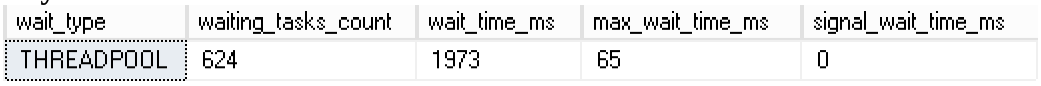
SQL Server has his own operating system. The scheduler coordinates the execution of processes called workers. Every CPU has a scheduler.

Physical Server 1



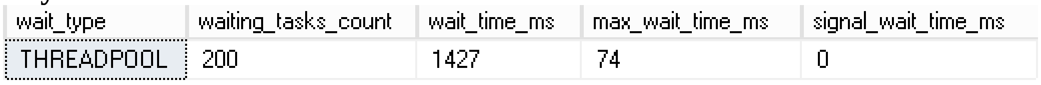
wait_type	waiting_tasks_count	wait_time_ms	max_wait_time_ms	signal_wait_time_ms
THREADPOOL	585	1732	43	0

Physical Server 2



wait_type	waiting_tasks_count	wait_time_ms	max_wait_time_ms	signal_wait_time_ms
THREADPOOL	624	1973	65	0

Physical Server 3

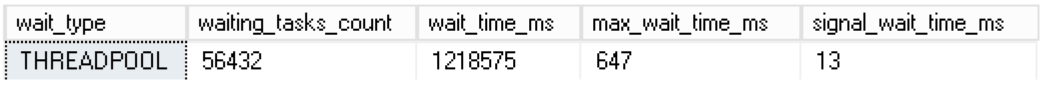


wait_type	waiting_tasks_count	wait_time_ms	max_wait_time_ms	signal_wait_time_ms
THREADPOOL	200	1427	74	0

The results above come from queries taken on production systems while heavy loads and heavy transactions. All systems have zero signal wait time and very low max wait time. On physical server environment all seems up and running well.

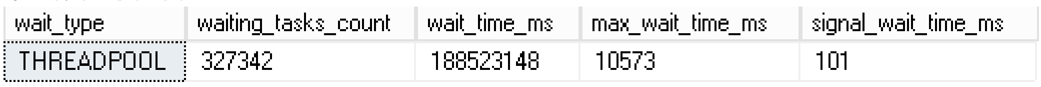
All virtual servers show problems in form of queueing as shown below. There are sensitive wait times on every virtual server.
In worst cases SQL Server is not able to execute any more requests inside the engine, because no free worker threads are available.
This needs attention and further deep analysis. A common fix is to increase the "max worker threads" setting. But this can lead to other problems like memory issues because every worker uses 2MB of memory and if the "max worker threads" will be increased other problems like RESOURCE_SEMAPHORE_QUERY_COMPILE waits can occur.
Another common causer for THREADPOOL waits can be blocking or a big amount of connections trying to run queries.
A good monitoring system in mandatory to overcome such problems.

Virtual Server 1



wait_type	waiting_tasks_count	wait_time_ms	max_wait_time_ms	signal_wait_time_ms
THREADPOOL	56432	1218575	647	13

Virtual Server 2



wait_type	waiting_tasks_count	wait_time_ms	max_wait_time_ms	signal_wait_time_ms
THREADPOOL	327342	188523148	10573	101

Auto update table statistics can lead to performance problems if goes from compatibility level 2012 to 2014 on a SQL Server 2014 Instance. Such fine tuning has to be tested in the test environment. In cases of big tables, it can be necessary to disable the auto update table statistics option and install a maintenance plan for this task running at night where the server has lesser load for instance.
Another problem is the data transfer between different database systems. Sometimes it is necessary to transfer big data loads from an Oracle system to an SQL Server system vice versa. To provide a maximum data transfer performance bulk copy is used with some fine tuning. In general, fast load is necessary to transfer data from or to Oracle Instance.

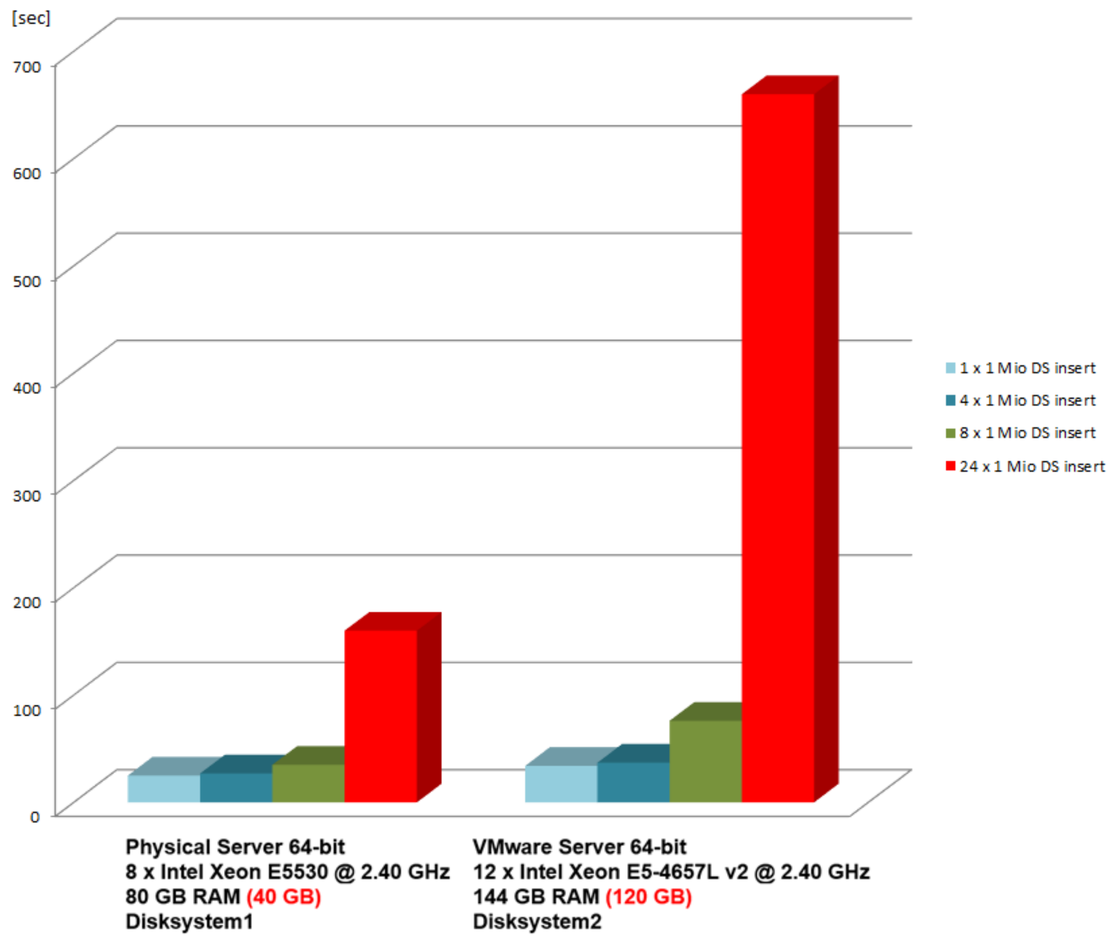


Figure 37*: multiple SQL insert duration and RAM impact*

The figure above shows the performance problem of the virtual server and let assume memory pressure.
The T-SQL script for benchmarking consists of several steps but the main step is simply insert one million random data rows in a table. Because the TempDB is the database on the SQL Server instance that is optimal installed and provides the maximum performance of the SQL Server instance, this is a good place to get optimal performance measure results.

The virtual server has faster processors and the disk subsystem is two to five times faster than the disk subsystem of the physical reference server.

This benchmark put a measureable stress to a server. It is possible to compare two or more systems, but in this case, it highlights the bottleneck for any further performance research.
The virtual server needs 660 seconds to insert 24 times one million data rows. This is eye-catching because the expected time consumption should be around half the time or lesser.
With 230GB RAM for the SQL Server the virtual server needs only 240 seconds to insert 24 times one million data rows. This is an expected result.

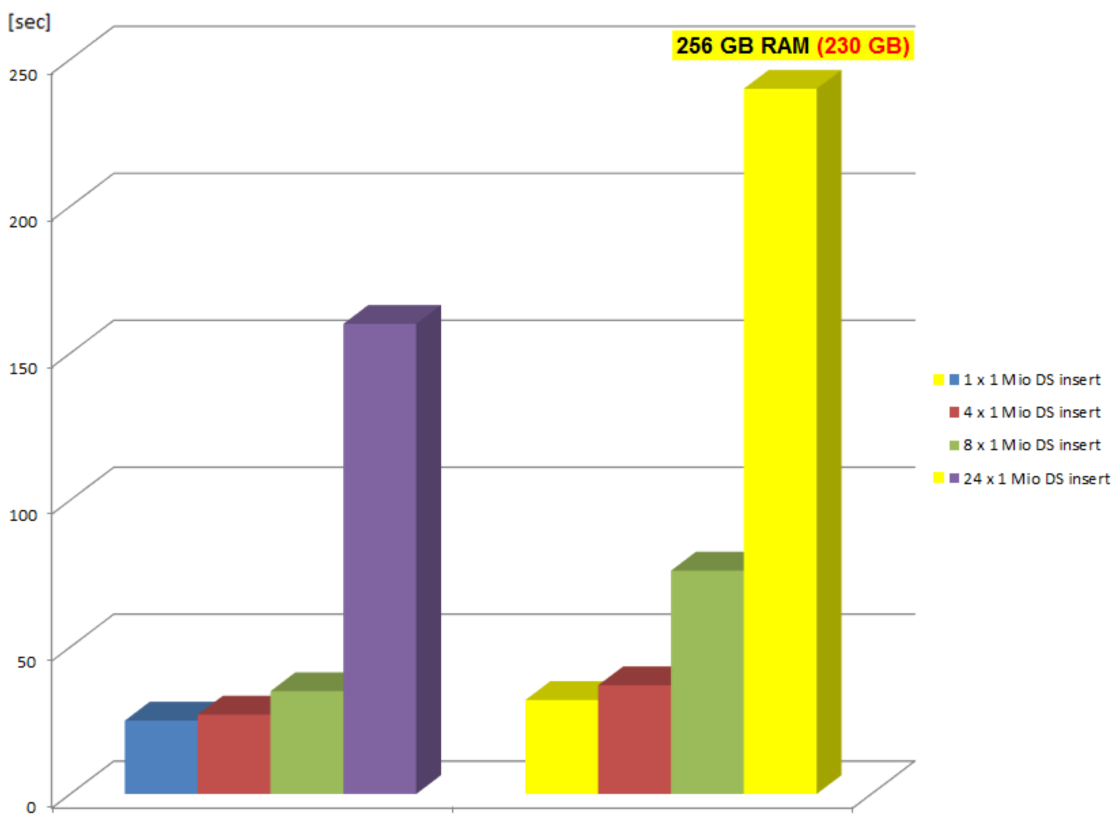
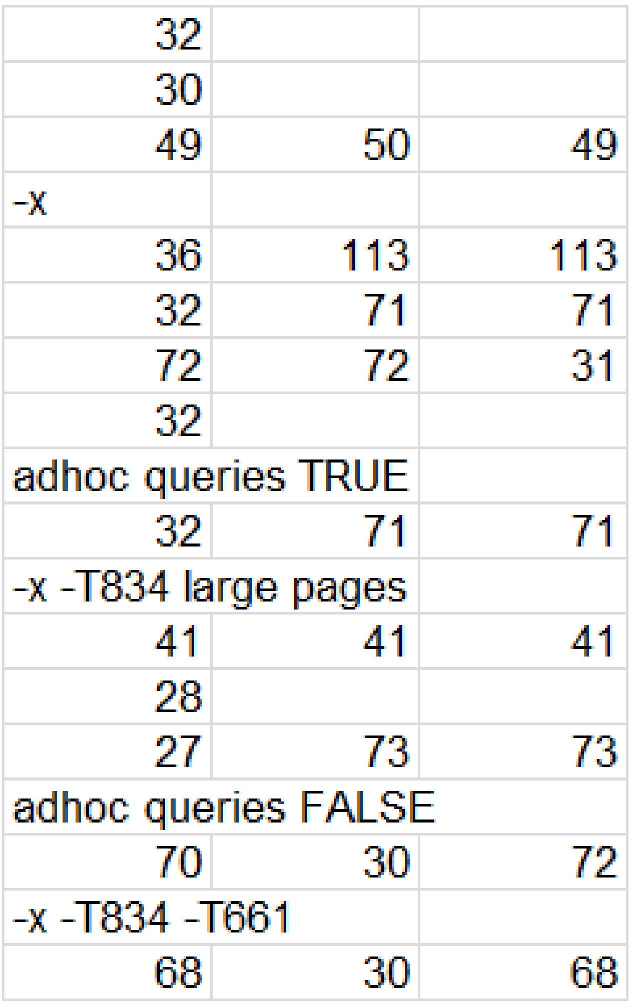


***Figure 38**: multiple SQL insert duration and RAM impact*

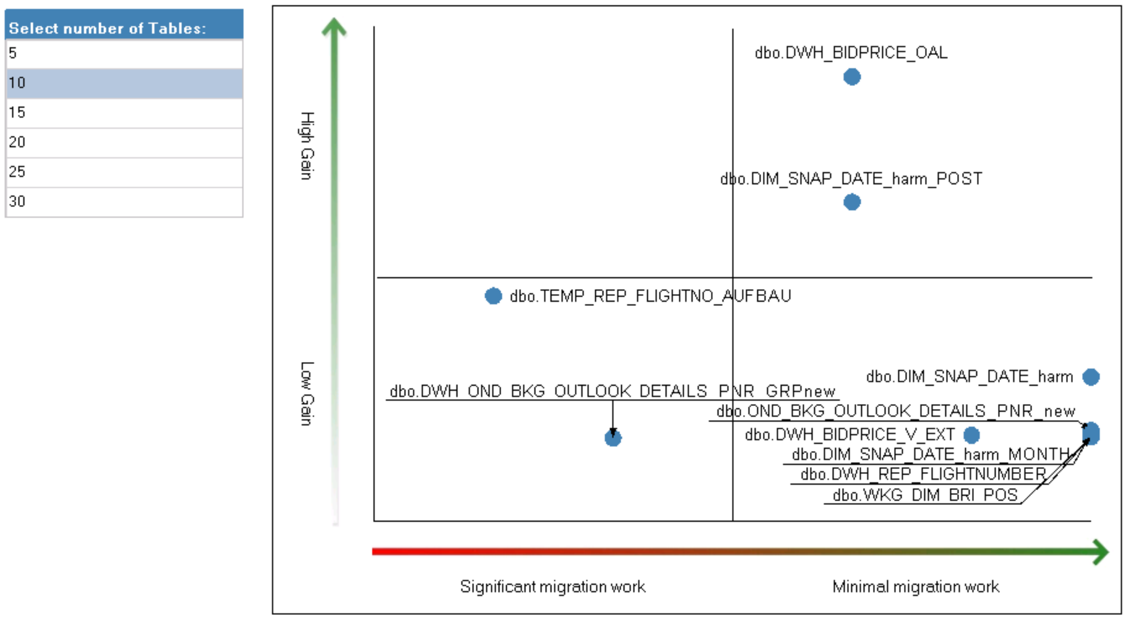
The figure above shows the necessity for the right benchmarking tool or benchmarking approach. In this case, the more stress given to the server the more eye-catching performance analysis results were obtained. This means that the more load a server has to complete the more bottleneck will be visible.



32		
30		
49	50	49
-x		
36	113	113
32	71	71
72	72	31
32		
adhoc queries TRUE		
32	71	71
-x -T834 large pages		
41	41	41
28		
27	73	73
adhoc queries FALSE		
70	30	72
-x -T834 -T661		
68	30	68

***Figure 39**: SQL Server trace flag testing for performance*

Figure 39 shows a test scenario to prove SQL Server performance settings and trace flags in combination with the aid of the described one million row insert benchmark. Such testing is also known as Pre-testing to clarify the SQL Server engine performance and to find the right combination of performance settings for a specific application.


Select number of Tables:
5
10
15
20
25
30
High Gain
Low Gain
dbo.DWH_BIDPRICE_OAL
dbo.DIM_SNAP_DATE_harm_POST
dbo.TEMP_REP_FLIGHTNO_AUFBAU
dbo.DIM_SNAP_DATE_harm
dbo.DWH_OND_BKG_OUTLOOK_DETAILS_PNR_GRPnew
dbo.OND_BKG_OUTLOOK_DETAILS_PNR_new
dbo.DWH_BIDPRICE_V_EXT
dbo.DIM_SNAP_DATE_harm_MONTH
dbo.DWH_REP_FLIGHTNUMBER
dbo.WKG_DIM_BRI_POS
Significant migration work
Minimal migration work

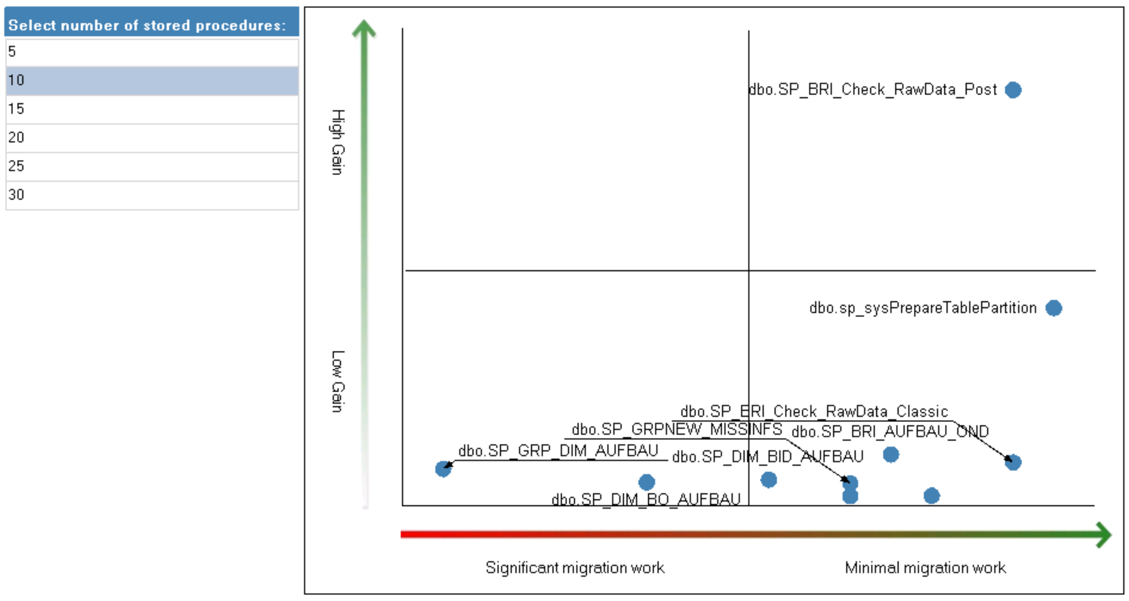
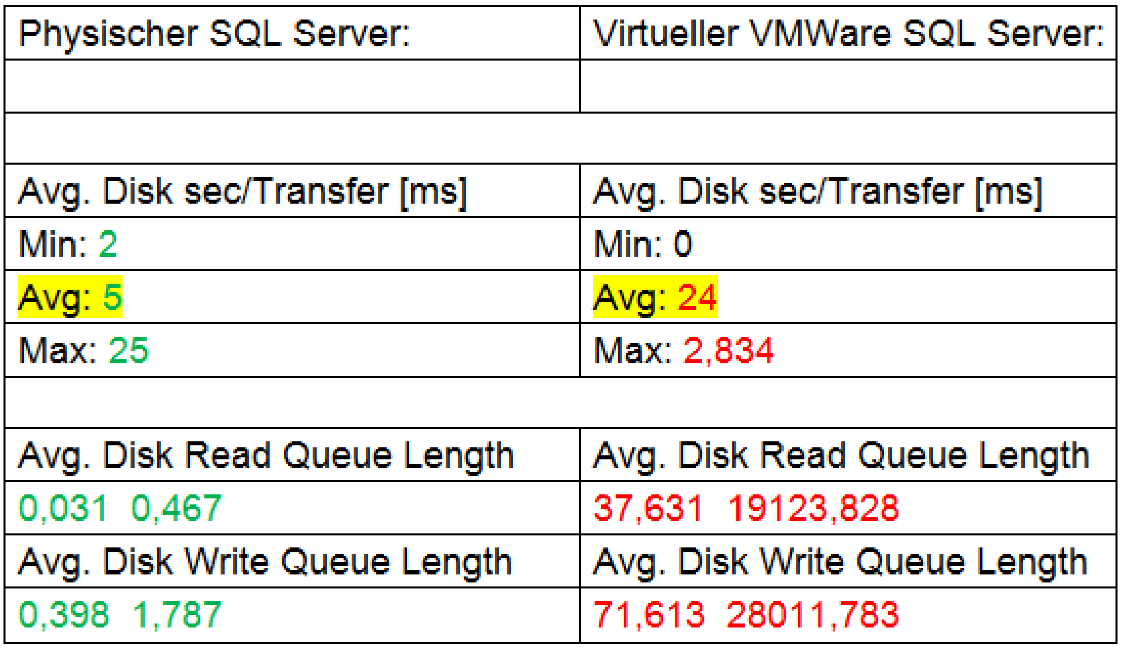


Figure 40*: SQL Server stored procedure effort evaluation*

In figure 40 a build in report in SQL Server Management Studio called "*Transaction performance analysis overview*" in SQL Server is shown. This report provides details of table performance statistics over a period of time and includes the access characteristics of the queries on the table as well as detailed contention statistics including latches and locks information.

Another problem can be with the storage system as following shown.

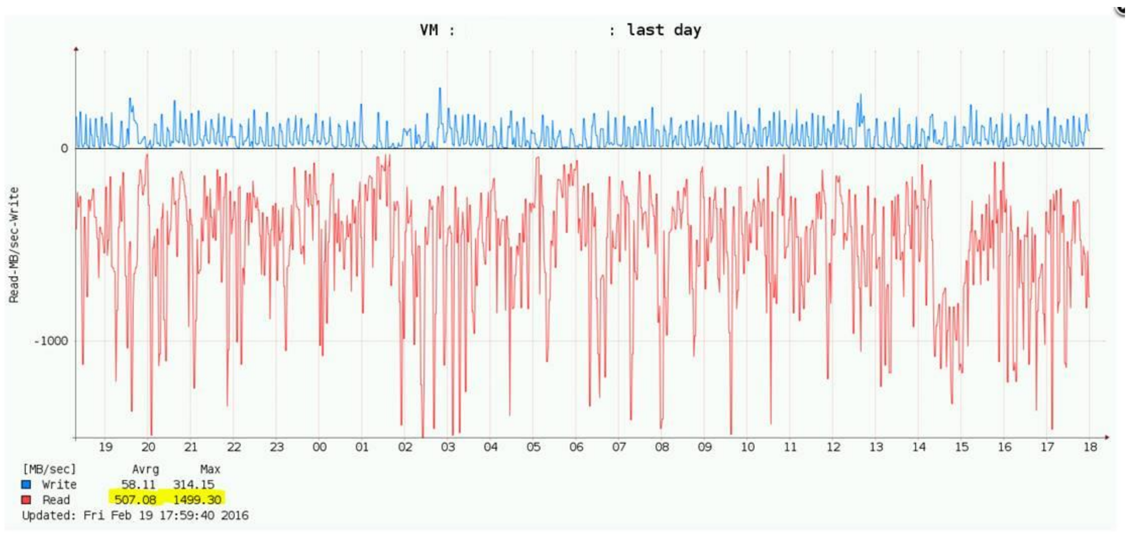


Physischer SQL Server:	Virtueller VMWare SQL Server:
Avg. Disk sec/Transfer [ms]	Avg. Disk sec/Transfer [ms]
Min: 2	Min: 0
Avg: 5	Avg: 24
Max: 25	Max: 2,834
Avg. Disk Read Queue Length	Avg. Disk Read Queue Length
0,031 0,467	37,631 19123,828
Avg. Disk Write Queue Length	Avg. Disk Write Queue Length
0,398 1,787	71,613 28011,783

Figure 41*: Storage sub system performance problems*

Best practice guidelines recommend an average value for the observed parameter Disk sec/Transfer of four milliseconds. On a physical environment there is no problem running with a Disk sec/Transfer value of 5 milliseconds or a little bit slower. Real performance problems are occurring if the Disk sec/Transfer value exceeds the factor 5 or more to the recommended value as shown in the figure 41 above. Furthermore, the maximum value also rises to a very critical amount.

In such case, it is really hard to identify the real problem causer. The virtual server system shown in figure 41 had a queueing problem because the storage adapter configuration was misconfigured. The correct new configuration repaired this problem.



```
[MB/sec]       Avrg      Max
🟦 Write       58.11    314.15
🟥 Read       507.08   1499.30
Updated: Fri Feb 19 17:59:40 2016
```

***Figure 42**: Storage sub system performance problems*

The figure above shows a big average transfer rate on the virtual server. It seems all ok because such a high transfer rate for read and write operations is a proof of a state of the art storage system satisfying every need.
A big data warehouse consists of twenty SQL Server and five Analysis Server for instance. Each server has its own defined workload. Often are three server involved to prepare data for further aggregation. If there is a performance problem, the search for the reason can be tricky and expensive.

4.3.5 Case scenario 5: NUMA Node misconfiguration

As shown and described in figure 25 it is very important to configure the hardware along the best practice guide. With the increasing number of processor core in a CPU the last years it becomes important to configure the whole system to operate it with the maximum performance.

NUMA (non-uniform memory access) is a method of configuring a cluster of processors with the goal to share memory locally. This improves performance and brings the ability to expanded the system. A NUMA node has up to eight cores.

Following an extract of a NUMA node misconfiguration, done with the tool **Coreinfo** from sysinternals.
https://docs.microsoft.com/de-de/sysinternals/

```
Intel(R) Xeon(R) CPU E5-2697 v4 @ 2.30GHz
Intel64 Family 6 Model 79 Stepping 1, GenuineIntel
Microcode signature: 0B00001A
HTT             *       Hyperthreading enabled
HYPERVISOR      *       Hypervisor is present
VMX             -       Supports Intel hardware-assisted virtualization
SVM             -       Supports AMD hardware-assisted virtualization
X64             *       Supports 64-bit mode

Logical to Physical Processor Map:
*-----------------------  Physical Processor 0
-*----------------------  Physical Processor 1
--*---------------------  Physical Processor 2
---*--------------------  Physical Processor 3
----*-------------------  Physical Processor 4
-----*------------------  Physical Processor 5
------*-----------------  Physical Processor 6
-------*----------------  Physical Processor 7
--------*---------------  Physical Processor 8
---------*--------------  Physical Processor 9
----------*-------------  Physical Processor 10
-----------*------------  Physical Processor 11
------------*-----------  Physical Processor 12
-------------*----------  Physical Processor 13
--------------*---------  Physical Processor 14
---------------*--------  Physical Processor 15
----------------*-------  Physical Processor 16
-----------------*------  Physical Processor 17
------------------*-----  Physical Processor 18
-------------------*----  Physical Processor 19
--------------------*---  Physical Processor 20
---------------------*--  Physical Processor 21
----------------------*-  Physical Processor 22
-----------------------*  Physical Processor 23

Logical Processor to Socket Map:
******------------------  Socket 0
------******------------  Socket 1
------------******------  Socket 2
------------------******  Socket 3

Logical Processor to NUMA Node Map:
******------------------  NUMA Node 0
------******------------  NUMA Node 1
------------******------  NUMA Node 2
```

```
------------------******  NUMA Node 3

Calculating Cross-NUMA Node Access Cost...
Approximate Cross-NUMA Node Access Cost (relative to fastest):
     00  01  02  03
00: 1.2 1.7 1.7 1.7
01: 1.0 1.6 1.7 1.6
02: 1.7 1.6 1.6 1.5
03: 1.6 1.1 1.3 1.4

Logical Processor to Cache Map:
******------------------  Data Cache         0, Level 1,   32 KB, Assoc   8, LineSize  64
******------------------  Instruction Cache  0, Level 1,   32 KB, Assoc   8, LineSize  64
******------------------  Unified Cache      0, Level 2,  256 KB, Assoc   8, LineSize  64
******------------------  Unified Cache      1, Level 3,   45 MB, Assoc  20, LineSize  64
------******------------  Data Cache         1, Level 1,   32 KB, Assoc   8, LineSize  64
------******------------  Instruction Cache  1, Level 1,   32 KB, Assoc   8, LineSize  64
------******------------  Unified Cache      2, Level 2,  256 KB, Assoc   8, LineSize  64
------******------------  Unified Cache      3, Level 3,   45 MB, Assoc  20, LineSize  64
------------******------  Data Cache         2, Level 1,   32 KB, Assoc   8, LineSize  64
------------******------  Instruction Cache  2, Level 1,   32 KB, Assoc   8, LineSize  64
------------******------  Unified Cache      4, Level 2,  256 KB, Assoc   8, LineSize  64
------------******------  Unified Cache      5, Level 3,   45 MB, Assoc  20, LineSize  64
------------------******  Data Cache         3, Level 1,   32 KB, Assoc   8, LineSize  64
------------------******  Instruction Cache  3, Level 1,   32 KB, Assoc   8, LineSize  64
------------------******  Unified Cache      6, Level 2,  256 KB, Assoc   8, LineSize  64
------------------******  Unified Cache      7, Level 3,   45 MB, Assoc  20, LineSize  64
```

The red marked lines show the performance loss of this configuration. If a processor of one NUMA node needs a look up to a memory of another NUMA node for instance, there is a performance loss of up to 70 percent.
The major problem of this misconfiguration is the fact that there occurs no error. The system seems to be running well without any error, but a lot of processing performance will be wasted.

Intel provides an online database with detailed information about every Intel processor. A look up in this database shows that this processor has 18 cores.
https://ark.intel.com/de/products/91755/Intel-Xeon-Processor-E5-2697-v4-45M-Cache-2-30-GHz-

That means that this misconfigured system has two processors of the type Intel(R) Xeon(R) CPU E5-2697 v4 and each has 18 cores.
One server will be operated with 24 cores separated in four NUMA nodes and each NUMA node holds six cores. This leads to performance loss up to 70%.

In addition, the important feature “CPU hotplug” should be disabled. This can increase memory performance and reduces latency.

5 Conclusion and outlook

This chapter summarizes the thesis and its results. The acquired knowledge and the limitations of the presented solution were described.

5.1 Conclusion

Lessons learned from a lot of real life database migration scenarios show that communication and documentation in an ITIL environment is a major task to guarantee success.

The important factors for a successful complete database migration are:

- Communication all the time of project duration.
- Pre-Testing and Full-Testing before migration.
- Performance Analysis to meet the requirements.
- Complete toolset to ensure system independence.
- Personnel with expert skills in all divisions.

The database migration framework in this thesis offers all necessary use cases and procedural methods to cover all occurring migration demands in a ITIL environment. Data conversion techniques, used in this thesis for SQL Server, as discussed in [10] are specified and included in the database systems as tools. Modern database systems provide a lot of useful included tools to interoperate and communicate with other database systems and applications.

Performance analysis are mandatory for system hardware and for database engine. It is useless to try optimizing a database when the monitoring system brings permanent alerts with high values of parameter like *"OS average disk queue length"* or *"Average disk sec / Transfer"* for example when the fundament such a virtual host and the virtual machine is not optimized and analysed. To prevent such scenarios, the best practice guide developed in this thesis is a good basis to build high performance database systems.

5.2 Framework Limitations

A fully-automated complete database migration in most cases does not exist. Every migration process underlies many error prune factors to success. The framework needs always to be adapted to meet the requirements of the environment. The framework never raises a claim to fulfill all solutions to any problem. It is a living process and with this intention this framework will become better and faster to solve a new problem over time.

5.3 Outook

Business processes are changing faster. Therefore, applications have to be adapted faster. Amount of data is growing rapidly. Virtualization for cost saving is getting mandatory. All these factors do more stress to an IT infrastructure than ever. Nevertheless, there is a big problem. The IT is a cost factor in every company. In a very big successful company, it is very a dangerous mistake to think it is necessary to save money in the IT. The IT always act as an enabler for business. Because of the costs of every IT department lies by one to two percent of the whole costs, capital investment is necessary to the IT department. In the banking sector, the IT is a mandatory important necessity and well understood for instance.

Performance needs are growing rapidly and that is in combination with rapidly growing data an underestimated problem. Regardless of which environment, if it is a small environment or a big data warehouse, there can occur situations or system behavior, which needs serious performance tuning. In addition, a mediator role between business demand and infrastructure becomes more and more important to achieve the faster changing business goal. Such role can be called application operation officer or application manager for instance.

A APPENDIX

Scripts

Windows Performance Monitor related items for Performance evaluation on SQL Server

```
"\Memory\Available MBytes"
"\Memory\Free System Page Table Entries"
"\Memory\Pages Input/sec"
"\Memory\Pages/sec"
"\SQLServer:Access Methods\Full Scans/sec"
"\SQLServer:Access Methods\Page Splits/sec"
"\SQLServer:Access Methods\Workfiles Created/sec"
"\SQLServer:Access Methods\Worktables Created/sec"
"\SQLServer:Buffer Manager\Buffer cache hit ratio"
"\SQLServer:Buffer Manager\Checkpoint pages/sec"
"\SQLServer:Buffer Manager\Free pages"
"\SQLServer:Buffer Manager\Lazy writes/sec"
"\SQLServer:Buffer Manager\Page life expectancy"
"\SQLServer:Buffer Manager\Page reads/sec"
"\SQLServer:Buffer Manager\Page writes/sec"
"\SQLServer:Buffer Manager\Stolen pages"
"\SQLServer:General Statistics\Logins/sec"
"\SQLServer:General Statistics\Logouts/sec"
"\SQLServer:General Statistics\User Connections"
"\SQLServer:Latches\Average Latch Wait Time (ms)"
"\SQLServer:Locks(_Total)\Average Wait Time (ms)"
"\SQLServer:Locks(_Total)\Lock Requests/sec"
"\SQLServer:Locks(_Total)\Number of Deadlocks/sec"
"\SQLServer:Memory Manager\Target Server Memory (KB)"
"\SQLServer:Memory Manager\Total Server Memory (KB)"
"\SQLServer:SQL Statistics\Batch Requests/sec"
"\SQLServer:SQL Statistics\SQL Compilations/sec"
"\SQLServer:SQL Statistics\SQL Re-Compilations/sec"
"\Paging File(_Total)\% Usage"
"\Paging File(_Total)\% Usage Peak"
"\PhysicalDisk(_Total)\Avg. Disk Read Queue Length"
"\PhysicalDisk(_Total)\Avg. Disk sec/Read"
"\PhysicalDisk(_Total)\Avg. Disk sec/Transfer"
"\PhysicalDisk(_Total)\Avg. Disk sec/Write"
```

```
"\PhysicalDisk(_Total)\Avg. Disk Write Queue Length"
"\Process(sqlservr)\% Privileged Time"
"\Process(sqlservr)\% Processor Time"
"\Processor(_Total)\% Privileged Time"
"\Processor(_Total)\% Processor Time"
"\System\Context Switches/sec"
"\System\Processor Queue Length"
"\VM Processor(*)\Limit in MHz"
"\VM Processor(*)\Reservation in MHz"
"\VM Processor(*)\Shares"
"\VM Processor(*)\CPU stolen time"
"\VM Processor(*)\% Processor Time"
"\VM Processor(*)\Effective VM Speed in MHz"
"\VM Processor(*)\Host processor speed in MHz"
"\VM Memory\Memory Active in MB"
"\VM Memory\Memory Ballooned in MB"
"\VM Memory\Memory Limit in MB"
"\VM Memory\Memory Mapped in MB"
"\VM Memory\Memory Overhead in MB"
"\VM Memory\Memory Reservation in MB"
"\VM Memory\Memory Shared in MB"
"\VM Memory\Memory Shared Saved in MB"
"\VM Memory\Memory Shares"
"\VM Memory\Memory Swapped in MB"
"\VM Memory\Memory Target Size"
"\VM Memory\Memory Used in MB"
```
Listing A.1: *Windows Performance Monitor related items for SQL Server*

Automatic installation of the performance monitor items

```
logman create counter SQL2014Perf -f bin  -b 11/07/2014
11:00:00  -E 11/07/2014 11:10:00  -si 01 -v mmddhhmm -o
"D:\MUECKE\SQL2014Perf" -cf
"D:\MUECKE\SQL2014BaselineCounters.config" -u DOMAIN\USER *
```
Listing A.2: *Automatic installation of the performance monitor items*

Insert of one million rows benchmark for performance analysis (SQL Server 2005, 2008, 2008R2, 2012, 2014, 2016, 2017, 2019)

```
-- Time

IF OBJECT_ID('T1') IS NOT NULL
DROP TABLE T1;
GO

CREATE TABLE T1 (id int, dt datetime);
GO
```

```sql
INSERT INTO T1(id, dt) VALUES (1, CURRENT_TIMESTAMP);
GO

-- Init

Use TEST

SET NOCOUNT ON

IF OBJECT_ID('dbo.TestTable', 'U') IS NOT NULL DROP TABLE dbo.TestTable
CREATE TABLE dbo.TestTable ( GUID_col_indexed uniqueidentifier, CHAR_col
char(10), GUID_col uniqueidentifier, Zeit datetime)
CREATE CLUSTERED INDEX IDX1 ON dbo.TestTable ( GUID_col_indexed )
GO

-- Step 1: Load TestTable with 1.000.000 records

declare @t int
set @t=1
begin transaction
while @t <= 1000000
begin
insert TestTable values (newid(), char(rand()*27 + 66) + char(rand()*27
+ 66) + char(rand()*27 + 66) + char(rand()*27 + 66) + char(rand()*27 +
66) + char(rand()*27 + 66) + char(rand()*27 + 66) + char(rand()*27 + 66)
+ char(rand()*27 + 66)+ char(rand()*27 + 66), newid(), sysdatetime())
select @t=@t + 1
end
commit transaction
go

-- Step 2: Select 1000 random values

select top 1000 * from TestTable order by GUID_col
go

-- Step 3: Delete 10000 random values

delete TestTable where GUID_col in (select top 10000 GUID_col from
TestTable order by GUID_col desc)
go

-- Step 4: Insert 10000 more values

declare @t int
set @t=1
begin transaction
while @t <= 10000
begin
insert TestTable values (newid(), char(rand()*27 + 66) + char(rand()*27
+ 66) + char(rand()*27 + 66) + char(rand()*27 + 66) + char(rand()*27 +
66) + char(rand()*27 + 66) + char(rand()*27 + 66) + char(rand()*27 + 66)
+ char(rand()*27 + 66)+ char(rand()*27 + 66), newid(), sysdatetime())
```

```
select @t=@t + 1
end
commit transaction

-- Step 5: Löschen der TestTable.

DROP TABLE dbo.TestTable
GO

INSERT INTO T1(id, dt) VALUES (10, CURRENT_TIMESTAMP);
GO

  declare @s datetime
  SELECT @s=dt
    FROM [TEST].[dbo].[T1] where id=1
  declare @e datetime
  SELECT @e=dt
    FROM [TEST].[dbo].[T1] where id=10

select @e-@s

INSERT INTO T1(id, dt) VALUES (101, @e-@s);
GO
```
Listing A.3*: Insert of one million rows (SQL Server)*

1.cmd (Windows)

(2.cmd, and so on are analog to 1.cmd with increasing numbered files like perftran02.sql and perftran02.txt)

```
sqlcmd -S SERVERNAME -d TEST -i perftran01.sql -o
perftran01.txt
```
Listing A.4*: Command to start the insert script of one million rows*

System near parallel start of the insert script A.3 24 times simultaneously (Windows)

```
time <enter.txt >timebegin.txt
start /REALTIME 1.cmd
start /REALTIME 2.cmd
start /REALTIME 3.cmd
start /REALTIME 4.cmd
start /REALTIME 5.cmd
start /REALTIME 6.cmd
start /REALTIME 7.cmd
start /REALTIME 8.cmd
start /REALTIME 9.cmd
start /REALTIME 10.cmd
start /REALTIME 11.cmd
start /REALTIME 12.cmd
start /REALTIME 13.cmd
start /REALTIME 14.cmd
start /REALTIME 15.cmd
start /REALTIME 16.cmd
start /REALTIME 17.cmd
start /REALTIME 18.cmd
start /REALTIME 19.cmd
start /REALTIME 20.cmd
start /REALTIME 21.cmd
start /REALTIME 22.cmd
start /REALTIME 23.cmd
start /REALTIME 24.cmd
```
Listing A.5*: Simultaneously start of the insert script A.3*

Insert of one million rows benchmark for performance analysis (Oracle)

```
CREATE TABLE TestTable (GUID_col_indexed varchar(46), CHAR_col
char(100), GUID_col varchar(46) unique);

CREATE UNIQUE INDEX GUID_col_indexed ON TestTable (GUID_col_indexed);

  begin
    for i in 1..1000000 loop
      insert into TestTable values (i, 'Oracle_SQL',i);
    end loop;
    commit;
  end;
  /

drop table TestTable;
```
Listing A.6*: Insert of one million rows (Oracle)*

Insert of one million rows benchmark for performance analysis with select and delete (Oracle 9i – 12c (12.3.0.1)

```
-- open sqlplus and start the script
-- sql> @c:\c-platte\dba_all\sql\benchmark.sql

alter session set nls_date_format = 'yyyy-mm-dd hh24:mi:ss';

-- create logfile
spool c:\c-platte\dba_all\sql\benchmark.log
-- show database name
```

```sql
select name from v$database;

set timing on
set pagesize 1500
set linesize 255

-- Table drop
DROP TABLE TestTable;

-- Startzeit anzeigen
select sysdate Start_Zeit from sys.dual;

-- Table create
CREATE GLOBAL TEMPORARY TABLE TESTTABLE (GUID_COL_INDEXED VARCHAR2(46),
CHAR_COL CHAR(10), GUID_COL VARCHAR2(46), CONSTRAINT SYS_C_TESTTABLE
UNIQUE(GUID_COL));
-- create unique index on first column
CREATE UNIQUE INDEX GUID_col_indexed ON TestTable (GUID_col_indexed);

-- step 1
-- insert 1 Million Data in Testtable
  begin
    for i in 1..1000000 loop
      insert into TestTable values (i, 'Oracle_SQL',i);
    end loop;
  end;
/

-- Step 2: Select first 1000 rows
select * from TestTable where rownum < 1001 order by GUID_col ;

-- Step 3: Delete first 10000 rows
delete TestTable where GUID_col in (select GUID_col from TestTable i
where rownum < 10001);

-- step 4
-- insert 10000 Data in Testtable
  begin
    for i in 1..10000 loop
      insert into TestTable values (i, 'Oracle_SQL',i);
    end loop;
  end;
/

-- Endzeit anzeigen
select sysdate End_Zeit from sys.dual;

-- Table drop
DROP TABLE TestTable;

spool off
exit
/
```
***Listing A.7**: Insert of one million rows with select and delete (Oracle)*

Insert of one million rows benchmark for performance analysis (SQL Server 2019 for RedHat Linux on Docker for Windows 10 64-bit)

```
Use test

SET NOCOUNT ON

/* create database with a memory-optimized filegroup and a container.
ALTER DATABASE test ADD FILEGROUP Test_mem CONTAINS
MEMORY_OPTIMIZED_DATA
ALTER DATABASE test ADD FILE (name='Test_Mem',
filename='/var/opt/mssql/data/Test_mem') TO FILEGROUP Test_mem
ALTER DATABASE test SET MEMORY_OPTIMIZED_ELEVATE_TO_SNAPSHOT=ON
GO
*/

CREATE TABLE dbo.TestTable ( GUID_col_indexed uniqueidentifier PRIMARY
KEY NONCLUSTERED HASH WITH (BUCKET_COUNT=7500000), CHAR_col char(10),
GUID_col uniqueidentifier) WITH (MEMORY_OPTIMIZED=ON,
DURABILITY=SCHEMA_AND_DATA)
--CREATE CLUSTERED COLUMNSTORE INDEX IDX1 ON dbo.TestTable --WITH
(ONLINE=ON)--, funktioniert nicht bei MEMORY OPTIMIZED TABLES! MEM-
TABLES auch nicht in TempDB erlaubt!
GO

-- Step 1: Load TestTable with 1.000.000 records

declare @t int
set @t=1
begin transaction
while @t <= 1000000
begin
insert TestTable values (newid(), char(rand()*27 + 66) + char(rand()*27
+ 66) + char(rand()*27 + 66) + char(rand()*27 + 66) + char(rand()*27 +
66) + char(rand()*27 + 66) + char(rand()*27 + 66) + char(rand()*27 + 66)
+ char(rand()*27 + 66)+ char(rand()*27 + 66), newid())
select @t=@t + 1
end
commit transaction
go

-- Step 2: Select 1000 random values

select top 1000 * from TestTable order by GUID_col
go

-- Step 3: Delete 10000 random values

delete TestTable where GUID_col in (select top 10000 GUID_col from
TestTable order by GUID_col desc)
go

-- Step 4: Insert 10000 more values
```

```sql
declare @t int
set @t=1
begin transaction
while @t <= 10000
begin
insert TestTable values (newid(), char(rand()*27 + 66) + char(rand()*27
+ 66) + char(rand()*27 + 66) + char(rand()*27 + 66) + char(rand()*27 +
66) + char(rand()*27 + 66) + char(rand()*27 + 66) + char(rand()*27 + 66)
+ char(rand()*27 + 66)+ char(rand()*27 + 66), newid())
select @t=@t + 1
end
commit transaction

-- Step 5: Drop Table, (IF OBJECT_ID('dbo.TestTable', 'U') IS NOT NULL
DROP TABLE dbo.TestTable funktioniert nicht bei MEMORY OPTIMIZED
TABLES!)
DROP TABLE dbo.TestTable
GO

/*
DBCC MEMORYSTATUS
DBCC FREESYSTEMCACHE ('ALL');
DBCC FREESESSIONCACHE
DBCC FREEPROCCACHE
DBCC DROPCLEANBUFFERS
*/
```
Listing A.8*: Insert of one million rows in memory optimized table (SQL Server 2019 for RedHat Linux on Docker for Windows 10)*

Bibliography

[1] Takahiro Hara, Kaname Harumoto, Masahiko Tsukamoto, Shojiro Nishio, Jun Okui. Main Memory Database for Supporting Database Migration, 1997

[2] Yusuke Yoshida, Masayoshi Aritsugi, Yoshinari Kanamori. Performance Evaluation of Combining Data Migration and Method Migration in Object Database Environments, 2002

[3] Reynold S. Xin, Patrick Dantressangle, Sam Lightstone, William McLaren, Steve Schormann, Maria Schwenger. MEET DB2: Automated Database Migration Evaluation, 2010

[4] Lixian Xing, Yanhong Li. Design and Application of Data Migration System in Heterogeneous Database, 2010

[5] Carlo Curino, Hyun J. Moon, Carlo Zaniolo. Automating Database Schema Evolution in Information System Upgrades, 2009

[6] Carlo A. Curino, Hyun J. Moon, Carlo Zaniolo. Graceful Database Schema Evolution: the PRISM Workbench, 2008

[7] M. Hitz, G. Kappel, E. Kapsammer, W. Retschitzegger. UML@Work, dpunkt.verlag, Heidelberg, 2005

[8] Florian Matthes, Christopher Schulz. Towards an integrated data migration process model, Garching b. München, April 2011

[9] Database and Expert Systems Applications 19th International Conference, DEXA 2008, Turin, Italy, September 1-5, 2008, Proceedings. Editors: Bhowmick, Sourav S., Küng, Josef, Wagner, Roland (Eds.)

[10] Abdelsalam Maatuk, Akhtar Ali, and Nick Rossiter. Relational Database Migration: A Perspective, 2008

[11] Advances in Service-Oriented and Cloud Computing. Workshops of ESOCC 2015, Taormina, Italy, September 15-17, 2015, Revised Selected Papers

[12] Caio H. Costa, Paulo H. M. Maia, Nabor C. Mendonca, and Lincoln S.Rocha. Supporting Partial Database Migration to the Cloud Using Non-Intrusive Software Adaptations: An Experience Report, Sept. 2015

[13] Clemens Swoboda, TU Wien Wirtschaftsinformatik Diplomarbeit 2008, TeCa-4-DaMi – Entwicklung eines Frameworks für die testgetriebene Datenmigration.

[14] Office of Government Commerce. ITIL The Official Introduction to the ITIL Service Lifecycle. TSO, U.K, 2007/08

[15] Alex Yates. Critiquing two different approaches to delivering databases: Migrations vs state, 2015

[16] Samir Behara. Database Delivery – State based vs Migration based, 2018

Glossary

Failover. In the case of a system hardware failure or a software failure in the service, the service will start on another cluster node. The process where one service is lead over to a healthy cluster node is called failover.

Acronyms

DBA	Database Administrator
ITIL	IT Infrastructure Library
SLA	Service-level-agreement
SQL	Structured Query Language
UML	Unified Modeling Language
SYS	System Administrator
CPU	Central Processing Unit
RAM	Random Access Memory
NUMA	Non-Uniform Memory Access
T-SQL	Transact-SQL
OLAP	Online Analytical Processing
SSD	Solid State Drive
SMO	Schema Modification Operators
SSIS	SQL Server Integration Services
DTS	Data Transformation Services
ETL	Extract, Transform, Load – Process
ELT	Extract, Load, Transform – Process