

Datenbank-Performance-Experimente in der Lehre

Max Flügel, Alexander Stein, Michael Höding
FH Brandenburg
Magdeburger Straße 50
14770 Brandenburg/Havel
(+49)3381 355 243

[fluegel|steina|hoeding]@fh-brandenburg.de

ABSTRACT

In diesem Beitrag stellen wir unsere Idee und deren Ansätze vor, die die Ausbildung im Bereich Datenbank-Performance mittels eines Praktikums unterstützen sollen. Wichtiges Element des Erkenntnisgewinns ist das eigene Erleben im Rahmen von Experimenten. Im Bereich Datenbank-Performance müssen Experimente und Experimentierumgebung so gestaltet und aufgebaut sein, dass den Erwartungen an Vergleichbarkeit und Wissenszuwachs entsprochen wird.

Keywords

Database Performance, Praktikum, Experiment

1. Einführung und Motivation

Datenbank-Performance ist in Forschung, Lehre und Praxis ein zentraler Betrachtungsgegenstand. Betriebliche Anwendungssysteme müssen durch akzeptable Antwortzeiten einen reibungslosen Betrieb ermöglichen. Datenbankforschung und Datenbankhersteller haben hierzu eine Vielzahl von Methoden und Mechanismen entwickelt und eingeführt. Eine Aufgabe der Lehre ist es, diese Methoden zu vermitteln, um ihre Anwendung zu unterstützen. Zwar gibt es zunehmend Self-Tuning-Mechanismen in Datenbankmanagementsystemen (DBMS), jedoch ist der gut ausgebildete Datenbankadministrator ein wichtiges Erfolgselement des soziotechnischen Gesamtsystems.

Um Studierende für ein Thema zu motivieren und auch zu begeistern, muss ihnen die Möglichkeit gegeben werden, Erkenntnisse in der praktischen Arbeit zu erlangen. Hierzu können Experimente eingesetzt werden, welche sich unter anderem durch selbstständiges Arbeiten auszeichnen. Der Student wird durch eine Aufgabenstellung mit einem in der Theorie beschriebenen Problem bekannt gemacht. Das Experiment hat in seiner Struktur einen festen Rahmen. Dieser unterstützt den Studenten ein Thema methodisch zu betrachten (vgl. [2] S.239). Zusätzlich erlernt der Student neben neuem Wissen auch soziale Kompetenzen (Teamfähigkeit, Konfliktlösung, usw.).

1.1 Problem

In Unternehmen, in denen leistungsstarke Datenbanken und ein gutes Antwortzeitverhalten eine Rolle spielen, kommt es vor, dass bezüglich der Datenbank-Performance voreilige und einseitige Entscheidungen getroffen werden und somit die sorgfältige Betrachtung der Datenbank-Performance zu kurz kommt.

In der Lehre wird oftmals zu wenig Aufschluss über dieses Thema gegeben bzw. in Veranstaltungen lediglich darauf hingewiesen, dass Datenbank-Performance unter anderem bei Unternehmen eine große Rolle spielt oder spielen kann. Welche Beiträge, sprich welche Methoden und/oder Werkzeuge, für eine optimale Performance-Auslastung eingesetzt werden können und wie mit potentiellen Leistungsdefiziten umgegangen werden kann, wird im Rahmen der Lehre viel zu selten betrachtet. Eine Vertiefung durch eigene Übungen findet zudem i.d.R. nicht statt.

Ein weiteres Problem, welches mit der zuvor genannten Sachlage einher geht, ist, dass das Thema Datenbank-Performance als sehr abstrakt und wenig kalkulierbar bzw. greifbar von vielen Studenten eingeschätzt wird. Es ist nicht genau klar, wie die Performance dargestellt und beeinflusst werden kann.

1.2 Zielstellung

Für die genannten Problemfelder sollen in der Lehre ein höherer Stellenwert und die notwendige Akzeptanz erreicht werden. Dabei soll nicht nur die Datenbank-Performance als solche beleuchtet werden. Auch Wirtschaftlichkeitsbetrachtungen der Performance sollen mit einfließen. Kernziel ist es, mögliche Umgangsweisen mit der Performance von Datenbanken deutlich zu machen.

Erreicht werden kann dieser Ansatz durch das praktische Arbeiten mit *Datenbank-Performance-Experimenten in der Lehre*.

In mehreren Übungsveranstaltungen sollen die Studenten eigenständig Praxiserfahrungen sammeln können, um Datenbank-Performance selbst zu erleben. Hierfür soll aufgezeigt werden, wie die Performance gemessen und interpretiert werden kann. Im Nachgang werden Methoden und Werkzeuge vorgestellt, mit denen die Performance optimiert wird. Denkbar wäre auch, dass die Studenten selbst nach geeigneten Möglichkeiten suchen und diese dann anwenden müssen.

Ziele sind demnach:

- Performance erleben
- Performance-Messmethoden kennenlernen und einsetzen

- Gesamtarchitektur betrachten (beispielsweise zugrundeliegendes System) und analysieren, um „Flaschenhalse“ zu erkennen
- Optimieren der Performance
- Wirtschaftlichkeit von Performance-Lösungen betrachten

Dabei werden fachliche als auch überfachliche Fähigkeiten ausgeprägt.

2. Theoretische Grundlagen

Um Performance zu erleben, sollen Performance-Tests und die didaktische Methode Experiment miteinander verbunden werden. Im folgenden Abschnitt wird auf Performance-Tests und ihre Rolle in den Systemtests eingegangen. Im Anschluss daran werden das Experiment in seiner Methode und die wesentlichen Phasen beschrieben. Abschließend sollen die Zusammenhänge zwischen beiden Themenfeldern aufgezeigt werden.

2.1 Performance-Test

„Die Performance oder Leistung in der Informatik ist die Fähigkeit eines Systems, eine Aufgabe in einer bestimmten Zeit zu erledigen“ ([4] S. 439). Dabei wird Performance durch den Anwender wahrgenommen und muss entlang der Zielgruppen-Anforderungen betrachtet werden (vgl. [4] S. 439). So werden Wartezeiten bei Datenanalysen durch den Nutzer eher akzeptiert, als die Wartezeit bei Internetsuchmaschinen. Die Performance eines Systems lässt sich hingegen durch Tests systematisch nachweisen und im Anschluss gegebenenfalls an die Anforderungen anpassen. Dabei gehören die Performance-Tests zu den Systemtests. Systemtests werden nach H. Balzert in Funktionstest, Leistungstest, Benutzbarkeitstest, Sicherheitstest und Interoperabilitätstest unterteilt. Diese Tests dienen zur Bestimmung der Produktqualität eines Softwaresystems (vgl. [5] S. 503 ff.). Durch Leistungstests können zwei grundsätzliche Fragestellungen beantwortet werden:

- Wie viele Daten kann das System verarbeiten?
- Wie lange braucht das System für die Datenverarbeitung?

Zusätzlich können hier auch die Dimensionen der Belastung eingliedert werden. Hierzu zählen die Tests unter normalen Bedingungen des Regelbetriebs, das Testen im Grenzbereich durch Lasttests und das bewusste Überschreiten des Grenzbereichs durch Stresstests.

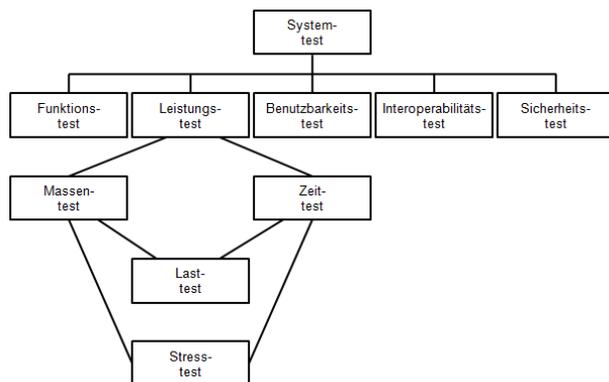


Abbildung 1: Systemtest

Die jeweiligen Tests lassen sich dabei in die Phasen Vorbereitung, Durchführung und Nachbereitung aufteilen (vgl. [6] S. 253). Die Tests werden entsprechend dokumentiert, um daraus Maßnahmen abzuleiten. In der Vergangenheit haben sich für System- bzw. Softwaretests verschiedene Normen und Standards entwickelt. Hier ist unter anderem die DIN 66273 (Vorgehensmodell für die Lastmessung) zu nennen.

2.2 Aufbau des Praktikums

Um Performance für die Studenten erlebbar zu machen, wird begleitend zur Vorlesung ein Praktikum durchgeführt. Dieses Praktikum teilt sich in einzelne Experimente auf. Die Experimente bauen in der Folge aufeinander auf, so erhalten die einzelnen Experimente einen roten Faden. Es werden die wichtigsten Eckpunkte der Performance-Messung berücksichtigt. Innerhalb der einzelnen Experimente sollen bewusst die Stellschrauben an den Systemen geändert werden, um die Wirkung der einzelnen Variablen auf das Verhalten des Systems zu erkennen und anschließend zu bewerten. Der Student soll hierbei selbstständig die Werte der Variablen ändern können.

Das Experiment lässt sich gemäß K. Reich in drei Phasen aufteilen (vgl. [3] S. 7):

- Am Beginn steht die Planungsphase. Hier wird die Vorberachtung umgesetzt. Zudem wird die Fragestellung formuliert, auf deren Basis eine Hypothese aufgestellt werden kann. Weiterhin werden die Randbedingungen des Experiments beschrieben und der Versuchsaufbau geplant.
- Dann erfolgt die Durchführungsphase. In dieser Phase wird die Planung umgesetzt, indem der Praktikumsversuch durchgeführt wird.
- Nach der Durchführung des Experiments erfolgt die Nachberachtung bzw. Auswertungsphase. Hier werden die Ergebnisse überprüft und hinsichtlich der potentiellen Fehlerquellen analysiert.

Daraus wird für die praktische Umsetzung folgender Ablauf abgeleitet:

1. Aufgabe stellen; durch den Lehrenden
2. Vorberachtung; durch den Studierenden
3. Überprüfung; durch den Lehrenden
4. Praktikumsversuch; durch den Studierenden
5. Nachberachtung; durch den Studierenden
6. Evaluation des Experiments; durch den Lehrenden

Bei der Formulierung der Aufgabenstellung sind zwei Extreme denkbar. Zum einen besteht die Möglichkeit, die Aufgabe völlig offen und ohne Restriktionen zu stellen, z.B. „Untersuchen Sie das System in Bezug auf CPU, Arbeitsspeicher und Storage!“. Zum anderen kann durch eine geschlossene Aufgabenstellung die Aufgabe so klar formuliert sein, dass der Student einen fest vorgeschriebenen Weg beim Abarbeiten der Aufgabe einschlägt und garantiert zum Ziel kommt. Beide Varianten haben nach Meinung der Autoren ihre Vor- und Nachteile (siehe Tabelle „Vor- und Nachteile bei der Aufgabenstellung“). Die zweckmäßigere Variante soll im Rahmen des Projekts ermittelt werden.

Tabelle 1: Vor- und Nachteile bei der Aufgabenstellung

Offene Aufgabenstellung	Aufgabenstellung mit Restriktionen
+ Der Student kann sich frei entfalten	+ Ergebnis ist schnell überprüfbar
+ Neue Erkenntnisgewinne	+ Ergebnis ist nachvollziehbar
+ Vielfältige Ergebnisse sind möglich	+ Ergebnisse sind vergleichbar
- Ergebnis ist ungewiss	- Erkenntnisgewinn ist vorge-schrieben
- Ergebnis und Experiment evtl. nicht nachvollziehbar	- Experiment lässt wenig Platz für Eigeninitiative
- Ergebnisse eventuell schwer vergleichbar	- Gänzlich neue Erkenntnis-se/Lösungswege sind kaum möglich

Folgende Themengebiete sind beispielsweise für das Praktikum denkbar:

- Erkundung der Systemeigenschaften
- Einfluss der SQL-Statements
- Nutzen und Wirkung von Indexten
- Arbeiten mit Puffern

Die Experimente werden durch den Studenten vorbereitet. In dieser ersten Phase müssen sich die Studenten die Aufgabenstellung genau versinnbildlichen und eine Zielstellung definieren. Im Anschluss daran gilt es den Lösungsweg zu planen. Dabei sollen theoretische Grundkenntnisse und das spezifische Fachwissen aus den Lehr- und Übungsveranstaltungen ebenso dienen, wie Kenntnisse und Fähigkeiten, welche sich die Studenten für die Aufgabenerfüllung selbstständig aneignen müssen. Desweiteren sollen die Prämissen der Planungsphase laut K. Reich zum Tragen kommen (siehe oben), d.h. die Studenten sollen mittels des erlangten Wissens eine Hypothese parallel zur Aufgabenstellung formulieren und diese Annahme dann durch ihren Praktikumsversuch entweder bestätigen oder widerlegen. Wenn eine Aufgabe beispielsweise verlangt eine bestimmte Datenmenge aus einer Datenbank einerseits mit einem SQL-Statement und andererseits mit Programm-Code (z.B. PHP, Java) abzufragen, dann könnten die Studenten hier die Hypothese aufstellen, dass das Abfragen mittels SQL performanter als das Abfragen mit Programm-Code ist. Als Ergebnis der Vorbetrachtung entsteht der Protokollentwurf.

Erst bei ausreichender Vorbetrachtung des Experiments, wird die Durchführung durch den Lehrenden möglich gemacht, hierzu erfolgt eine Überprüfung. Während des Praktikumsversuchs wird der Vorplanung entsprochen und die Aufgabenstellung praktisch gelöst. Die Ergebnisse werden in einem (Mess-)Protokoll festgehalten.

Nach erfolgreicher Durchführung des Praktikumsversuchs, werden die Ergebnisse in der Auswertungsphase (Nachbetrachtung) zusammengefasst und beurteilt. Die Beurteilung erfolgt hinsich-

tlich der in der Vorbetrachtung getroffenen Hypothese. Hier soll der Student durch eine ehrliche und realistische Einschätzung seiner ursprünglichen Annahme das Resultat seines Versuchs bewerten. An dieser Stelle sollte auch eine Fehleranalyse mit einfließen. Neben der technischen Beurteilung zählen auch die Wirtschaftlichkeitsbetrachtungen (Verhältnis zwischen Nutzen und Aufwand/Kosten) zu den Erfordernissen dieser Phase.

Abschließend wird das Experiment vom Lehrenden evaluiert. Zum einen wird im Rahmen der Evaluation die Einhaltung der formalen Vorgehensweise geprüft, hier die Phasen des Experiments und die Dokumentation. Zum anderen wird die Plausibilität begutachtet. An dieser Stelle müssen die Vor- und Nachbetrachtung in einem logischen Zusammenhang stehen, d.h. wurde in der Nachbetrachtung auch auf genau das eingegangen, was in der Vorbetrachtung geplant wurde. Das setzt natürlich voraus, dass in der Durchführungsphase auch das umgesetzt wurde, was in der Vorbetrachtungsphase angesetzt wurde.

Der Evaluationsaufwand steigt natürlich auch mit steigendem Grad einer offenen Aufgabenstellung. Hier ist ganz besonders darauf zu achten, wie die Studierenden vorgegangen sind und wie plausibel der eingeschlagene Lösungsweg ist. Näher betrachtet werden muss an dieser Stelle der wissenschaftliche Anspruch des Lösungswegs.

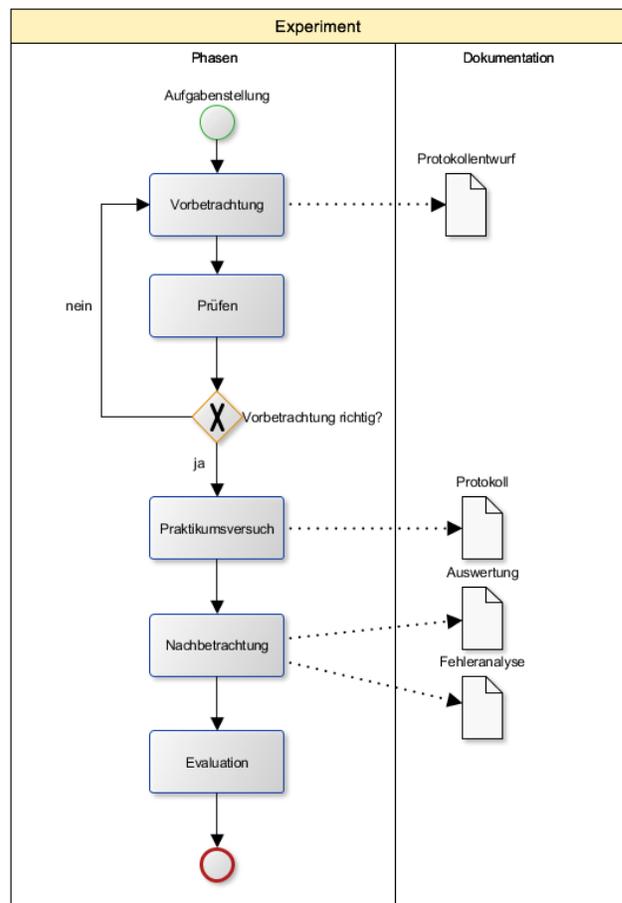


Abbildung 2: Ablauf eines Experiments

Für die Durchführung der Praktikumsversuche soll ein im Vorfeld festgelegter Wochentag, der Versuchstag, dienen. Die Festlegung auf einen solchen Versuchstag und den übrigen „normalen“ Tagen erfolgt einerseits aus lehrbedingten und andererseits aus technischen Gründen. Die Studenten sollen Erfahrungen mit einem technisch „großen“ Umfeld sammeln können. Das bedeutet, dass sie die Performance von umfangreichen Datenbanken in leistungsstarken Experimentierumgebungen (Servern) testen sollen. An den Versuchstagen stehen diese Experimentierumgebungen zur Verfügung. An den übrigen Tagen haben die Studenten Zugriff auf wesentlich „kleinere“ Umgebungen, um ihre Praktikumsversuche vorbereiten bzw. nach den Versuchen Nacharbeiten durchführen zu können. Genaueres zum Zusammenspiel der beiden Umgebungsarten ist im vierten Kapitel nachzulesen.

Die Zeit zwischen den Versuchstagen soll durch den Studenten für die Vorbereitung und Nachbereitung seines Versuchs dienen. Die jeweiligen Schritte der Experimente werden durch den Studenten dokumentiert und lassen sich zum Abschluss des Praktikums als Basis für die Prüfungsnote nutzen.

2.3 Zusammenhang zwischen Performance-Test und Experimenten

In der Informatik kann man Performance-Tests zu den Experimenten zählen (vgl. [7] S. 233 ff.). Wie in jedem Test folgen auch Performance-Tests einer bestimmten Abfolge. Tests werden geplant und im Anschluss durchgeführt. Die Ergebnisse des Tests werden dokumentiert und ausgewertet. Genauso wie Experimente müssen Tests auch vergleichbar und reproduzierbar sein. Wenn man diesem allgemeinen Vorgehen folgt, lassen sich hier Parallelen zum Experiment ziehen. Die Studenten sollen diese Parallelen erkennen und sie für den Aufbau der Experimente nutzen.

3. Organisation der Lehre

Nachdem die Begriffe Experiment und Performance-Test beschrieben sind, soll nun gezeigt werden, wie das Experiment die Lehre beeinflusst. Es wird dabei auf die Aufteilung der in der Lehrveranstaltung teilnehmenden Studenten und die daraus resultierenden Gruppenstärken ebenso eingegangen, wie auf die Rolle des Lehrenden und des Studenten bei dieser Form der praktischen Übung

3.1 Aufteilung der Lehrveranstaltung

Für die Experimente sollen sich die Studierenden in Gruppen zusammenfinden. Die Arbeit im Team wird erfahrungsgemäß bessere Ergebnisse hervorbringen als Einzelarbeit, da sich die Studenten mit ihren unterschiedlichen Wissensständen austauschen und somit ergänzen können. Zudem ist die Bildung von Gruppen der Ressourcenplanung dienlich, wie in Kapitel vier ersichtlich wird.

In der Fachhochschule Brandenburg im Studiengang Wirtschaftsinformatik wird pro Semester mit einer Studierendenzahl von bis zu 80 Studenten ausgegangen. Dabei ist eine Aufteilung in maximal 20 Gruppen noch handhabbar. So können diese 20 Gruppen ihre Praktikumsversuche (am Versuchstag) in maximal 4 Blöcken á 90 Minuten durchführen. Es ergibt sich folgende Matrix der Gruppenstärke und Aufteilung:

Tabelle 2: Aufteilung der Gruppen

	Studierendenzahl		
	40	60	80
2	20	30	40
Gruppenstärke 3	12*3+1*4	20	24*3+2*4
4	10	15	20
5	8	12	16

Es wird für die weitere Planung von einer maximalen Gruppenzahl von 20 Gruppen ausgegangen, d.h. bei einer Studierendenzahl von 40-80 Studierenden eine Gruppenstärke von 2-4 Studierende je Gruppe. Innerhalb der einzelnen Gruppen müssen dann die aus dem Einzelexperiment abgeleiteten Aufgaben entsprechend aufgeteilt werden. Ist die Studierendenzahl über 80, muss die Gruppengröße auf 5 erhöht werden, um die Gesamtgruppenzahl von 20 nicht zu überschreiten.

3.2 Rolle des Studierenden

Der Studierende wird während der Experimente selbst aktiv. Durch entsprechende theoretische Vor- und Nachbetrachtungen wird er ein optimales Verhältnis zwischen Theorie und Praxis erhalten. Er schult neben den rein fachlichen Fähigkeiten auch andere Kompetenzen. So muss er gerade in der Zusammenarbeit mit anderen Studierenden seine Kommunikationsfähigkeit schulen und auch sich selbst und die Gruppe in der er arbeitet organisieren. Bei der jeweiligen Gruppenorganisation wird vorausgesetzt, dass sich die Studierenden eigenständig für die Aufgaben einteilen, bei denen ihre fachlichen Stärken liegen (z.B. Dokumentation, Programmierung, System-Administration). Als primäre Anforderung an den Studierenden muss dessen Interesse, neue Sachverhalte zu entdecken, geweckt werden (vgl. [3] S. 12). Die Fähigkeit, eventuell auftretende Konflikte frühzeitig zu erkennen und diese abzustellen, wird als sekundäre Anforderung angesehen.

3.3 Rolle des Lehrenden

Dem Lehrenden wird während des Experiments eher die passiv unterstützende Rolle zu teil. Er führt in den Vorlesungen in die Thematik ein und vermittelt dabei die theoretischen Grundlagen (vgl. [3] S. 11). Während der Experimente wird er beratend zur Seite stehen und dem Studenten bei Bedarf Denkanstöße geben. Er muss nach der Vorbereitungsphase eine erste Überprüfung des Aufbaus und des Experiments durchführen, bevor er den Praktikumsversuch durchführen lässt. Es wäre hier vorstellbar Lernplattformen wie Moodle [11] einzusetzen, um in einem kurzen Wissenstest die Vorbereitung zu überprüfen. Auch die Ergebnisse des Praktikumsversuchs (z.B. das Messprotokoll) können via Moodle verwaltet und überprüft werden.

4. Ressourcenplanung

Um den Studentengruppen die Möglichkeit der praktischen Arbeit zu geben, sind Experimentierumgebungen auf und mit speziellen Systemen (Hardware und Software) notwendig.

4.1 Einsatz von Virtualisierung

Für die Lehre der Messung von Datenbank-Performance ist es wichtig, dass in einem Umfeld gearbeitet wird, in dem keine oder nur sehr wenige Leistungsschwankungen bzgl. der Hardware zu erwarten sind. Grundsätzlich kann es eine mögliche Lösung sein, in die bestehende Infrastruktur der Hochschule einen zusätzlichen Server oder ein Server-Blade zu integrieren, welcher von anderen Servern zumindest Software-seitig abgekapselt wird.

Nun stellt das gleichzeitige Arbeiten der Studentengruppen auf einem und demselben Server keine gute Lösung dar, da kaum feststellbar ist, welche Server-Ressource oder Software gerade von einer bestimmten Gruppe beansprucht wird. Hier sind ein eindeutiges Messen der Performance und somit vergleichbare Experimente nicht möglich.

Die Lösung stellt hier die Server-Virtualisierung dar. „Die Virtualisierung abstrahiert die physische Hardware vom Betriebssystem. Die klassische 1:1-Verbindung von Hardware und Betriebssystem, die man vor allem in der PC-Welt kennt, wird aufgehoben; dazwischen wird eine neue Schicht gelegt, die diese Abstraktion vornimmt, die Virtualisierungsschicht.“ ([10] S. 33) Die Aufgabe des Virtualisierens für die Experimentierumgebung übernimmt das VMware-Produkt vSphere mit seinem ESX Server (siehe nachfolgende Abbildung).

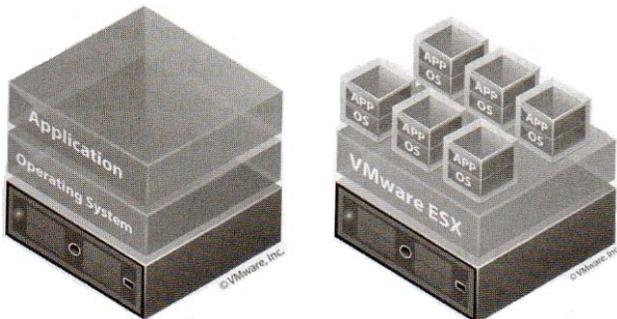


Abbildung 3: Konventioneller Server-Aufbau (links) versus Server-Virtualisierung (rechts) ([10] S. 33)

Durch die Virtualisierung können auf dem Server mehrere virtuelle Maschinen (VMs) eingerichtet werden, innerhalb derer die Studenten dann ihre Experimente unter gleichen Rahmenbedingungen (Vereinheitlichung der Hardware bzw. virtuellen Hardware) durchführen können. Jede VM erhält dann ihr eigene Softwareumgebung, wie Betriebssystem (engl.: Operating System oder kurz: OS), Datenbank und weitere Applikationen (App), wie beispielsweise Performance-Tools. Dies ist in der obigen Abbildung durch die kleinen Würfel dargestellt.

Die zum Einsatz kommenden VMs lassen sich in Entwicklungssysteme und Versuchssysteme unterteilen. Die Systeme unterscheiden sich im Wesentlichen durch ihr Leistungsspektrum und ihre Benutzung.

Für die Planung des Versuchsaufbaus kommen Entwicklungssysteme zum Einsatz. Das sind kleine VMs, sprich VMs mit einer geringeren Leistung und Konfiguration. Jede Gruppe erhält eine solche Entwicklungs-VM an der sie Änderungen vornehmen kann bzw. bestimmte Teilaufgaben bereits im Vorfeld testen kann. Diese Systeme stehen den Gruppen zwischen den einzelnen Versuchstagen zur Verfügung. Die Entwicklungssysteme erhalten eine kleine Datenbank. Während die Entwicklungs-VMs aktiv

sind, bleiben die Versuchs-VMs inaktiv (siehe nachfolgende Abbildung).

Die eigentlichen Praktikumsversuche werden auf den Versuchssystemen mit großen VMs realisiert. Diese VMs besitzen eine große Datenbank und deutlich mehr Leistung und ermöglichen Performance-Tests in großem Umfang. Zeitlich werden die Versuchs-VMs so geschaltet, dass an den Versuchstagen die Entwicklungs-VMs der Gruppen abgeschaltet und die Versuchs-VMs aktiviert werden.

Nach den Praktikumsversuchen, werden die Systeme entsprechend umgedreht wieder aktiviert bzw. deaktiviert.

In der nachfolgenden Abbildung sind die schematische Aufteilung der unterschiedlichen VM-Gruppen und deren Aktivität ersichtlich.

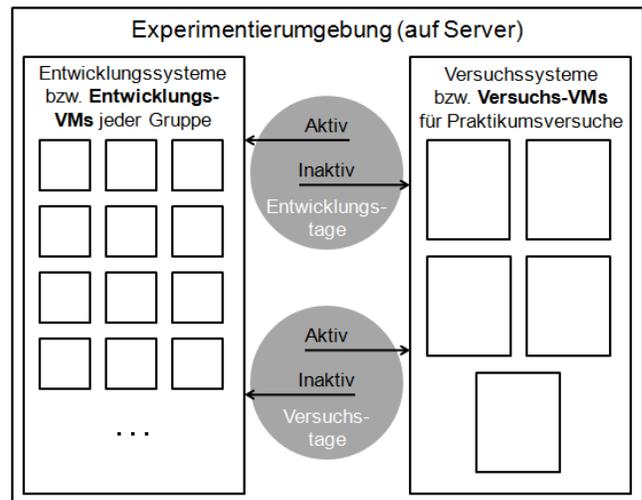


Abbildung 4: Aufbau der Experimentierumgebung

Durch Anfertigen von Clones (Klone) der ersten angelegten VM pro VM-Gruppe kann schnell die gewünschte VM-Anzahl erreicht werden, ohne jede VM einzeln auf herkömmlichem Wege anlegen zu müssen.

Die zugrundeliegenden Ressourcen, sprich die physische Hardware des Servers, wird unter den jeweils aktiven VMs fest aufgeteilt.

Mit der Virtualisierungs-Lösung können folgende positive Nebeneffekte für das Praktikum erreicht werden:

- Performance-Steigerung der Systeme
- Berücksichtigung von Aspekten der Green-IT (Einsparung von Hardware, Verringerung des Stromverbrauchs und Kühlaufwands)
- Gute Administrierbarkeit der Experimentierumgebung (Änderungen an den VMs, Zurücksetzen der VMs durch Snapshots) ([10] S. 34 ff.)

Die genannten Nebeneffekte können den Studenten in der Lehre kommuniziert bzw. selbst von ihnen erfahren werden.

Zudem kann als positiv angesehen werden, dass bei eventuellen Schäden am System lediglich eine VM betroffen sein wird und nicht der Server bzw. die Hardware direkt. Die beschädigte VM kann dann wieder in den Urzustand zurückgesetzt werden.

4.2 Konfiguration der virtuellen Maschinen

Ausgehend von der Gruppenplanung aus dem Abschnitt 3.1 kann die nachfolgende Konfiguration für die VMs abgeleitet werden. Hierbei wird lediglich auf die Zusicherungen von Arbeitsspeicher und Festplattenkapazität eingegangen.

Den Entwicklungs-VMs wird während der Entwicklungszeit 4 GB Arbeitsspeicher zugesichert. Daraus resultiert ein Arbeitsspeicherbedarf von 80 GB. Wird an dieser Stelle mit einer Reserve kalkuliert, kommt man, in Anbetracht der gängigen Konfiguration von RAM-Modulen, auf einen erforderlichen Arbeitsspeicher von 96 GB. Mit den 16 GB Reservespeicher kann unter anderem die Gruppenzahl bei Bedarf erhöht werden (4 Gruppen mehr mit wiederum jeweils 4 GB RAM). Die Entwicklungs-VMs werden mit einer kleinen Datenbank mit max. 4 GB Datenvolumen ausgestattet (für vorbereitende Tests). Weiterhin wird zusätzlicher Storage von 6 GB für das Betriebssystem, das DBMS und weitere Tools (Performance-Tools) beansprucht. Eine Reserve von 2 GB Speicher wird ebenfalls eingeplant. Daraus folgt für jede Entwicklungs-VM ein Storage-Bedarf von 12 GB. Bei 20 Gruppen bedeutet das einen Storage von 240 GB. Rechnet man die 4 potentiellen Entwicklungssysteme aus der Reserve hinzu, führt dies zu einem Speicherbedarf von 288 GB für die Entwicklungs-VMs.

Für die Versuchs-VMs sollen, wie in Abbildung 4 erkenntlich ist, 5 Systeme zur Verfügung stehen. Da zu den Versuchstagen sämtliche Ressourcen den Versuchs-VMs zur Verfügung stehen sollen, kann der gesamte Arbeitsspeicher (96 GB) entsprechend aufgeteilt werden. Das führt zu einem Arbeitsspeicher pro Versuchs-VM von rund 19 GB. In Sachen Storage-Kapazität sollen je Versuchssystem 80 GB vorgehalten werden (für Betriebssystem, DB, DBMS und Tools). Das führt zu einem Storage-Bedarf von 400 GB für die Versuchssysteme.

4.3 Server-Hardware

Nachfolgend werden Eckpunkte der Server-Hardware genannt.

Für das Praktikum kommt ein Server-Blade zum Einsatz. Dieses Blade besteht aus zwei Prozessoren (CPUs) mit jeweils 6 Kernen und 3.46 GHz. Desweiteren besitzt der Server, wie aus der Kalkulation hervorgeht, 96 GB RAM. Der Gesamt-Storage, welcher in den Berechnungen bestimmt wurde, beträgt ca. 700 GB. Hierfür stehen Hardware-seitig drei 450 GB Fibre Channel Festplatten zur Verfügung. Die großzügige Reserve soll eventuellen späteren Anforderungen gerecht werden.

4.4 Werkzeuge der Performance-Messung

Die Performance-Messung hat verschiedene Aspekte. Es müssen hier Methoden und Werkzeuge für die Umsetzung voneinander abgegrenzt werden. So werden durch die Methoden einzelne Verfahrensansätze für die Tests beschrieben. Mit „Werkzeuge“ sind hier die Tools zur Messung der entsprechenden Werte zusammengefasst. Es gibt ein weites Spektrum an Tools, wobei immer die Aktualität und Verwendbarkeit der Werkzeuge geprüft werden sollte. Auf der einen Seite können Tools genutzt werden, welche in den jeweiligen Systemen, wie beispielsweise im Betriebssystem oder im DBMS, vorhanden sind oder es können zusätzliche Werkzeuge zum Einsatz kommen, wie z.B. Nagios [12]. Für die Wahl der Methoden und Werkzeuge bestehen mehrere Möglichkeiten, zum einen können feste Vorgaben durch den Lehrenden gemacht werden und zum anderen kann der Einsatz freigestellt

werden, so dass in der Vorlesung ein Pool an Methoden und Werkzeugen vorgestellt wird und der Student entscheidet im Rahmen des Experiments, wie er vorgehen möchte. In der weiteren Arbeit muss geprüft werden, welche der beiden Methoden den Studenten am besten beim Lernprozess unterstützt.

5. Fazit und Ausblick

Performance ist wichtig. Das ist jedem, der mit Datenbanksystemen zu tun hat, klar. Den Studierenden ein Gefühl für diese Thematik zu geben, das ist eine der Herausforderungen der Lehre. Um diese Herausforderung zu meistern, bieten sich Experimente an. Dadurch kann der Student praktische Erfahrungen im Bereich Datenbank-Performance erlangen. Damit den Studenten einheitliche Umgebungen zum Experimentieren zur Verfügung gestellt werden, bieten sich virtualisierte Systeme an. Dieser Rahmen ist bekannt und darüber sind sich die Autoren einig. In Zukunft muss gezeigt werden, wie die Aufgabenstellungen für die Studenten zu formulieren sind und wie die Performance konkret zu messen ist. Im Studiengang Wirtschaftsinformatik muss neben den technischen Details der Systeme auch immer die Wirtschaftlichkeit betrachtet werden. Auch dieser Bereich muss in Zukunft genauer betrachtet werden.

6. REFERENCES

- [1] Euler D.; Hahn A., Wirtschaftsdidaktik, 2004, 1. Auflage, Haupt Verlag, Bern
- [2] Schubert S.; Schwill 1., Didaktik der Informatik, 2004, 1. Auflage, Spektrum Akademischer Verlag, München
- [3] Reich K., Experiment, 2008, <http://methodenpool.uni-koeln.de/download/experiment.pdf> Abruf: 28.03.2012
- [4] Faustmann; Höding; Klein; Zimmermann, Oracle- Datenbankadministration für SAP, 2007, 1. Auflage, Galileo Press, Bonn
- [5] Balzert H.; Lehrbuch der Software- Technik, 1998 Spektrum Akademischer Verlag GmbH, Heidelberg; Berlin
- [6] Schlimm N.; Novakovic M.; Spielmann R.; Knierim T., Performance-Analyse und -Optimierung in der Softwareentwicklung, 2007, Informatik Spektrum, Springer-Verlag
- [7] Claus V.; Schwill A., Duden der Informatik, 2001, 3. Auflage, Duden Verlag, Mannheim
- [8] Müller T., Leistungsmessung und -bewertung, 2004, Seminararbeit, Westfälische Wilhelms-Universität Münster, <http://www.wi.uni-muenster.de/pi/lehre/ss04/SeminarTesten/Leistungsmessung.pdf> Abruf: 28.03.2012
- [9] Barrett D., Linux kurz und gut, 2004, 1. Auflage, O'Reilly Verlag, Köln
- [10] Zimmer D.; Wöhrmann B.; Schäfer C.; Baumgart G.; Wischer S.; Kügow O., VMware vSphere 4 - Das umfassende Handbuch, 2010, 1. Auflage, Galileo Press, Bonn
- [11] <http://moodle.de/> Abruf: 18.05.2012
- [12] Barth; Schneemann; Oestreicher, Nagios: System- und Netzwerk-Monitoring, 2012, 3. Auflage, Open Source Press