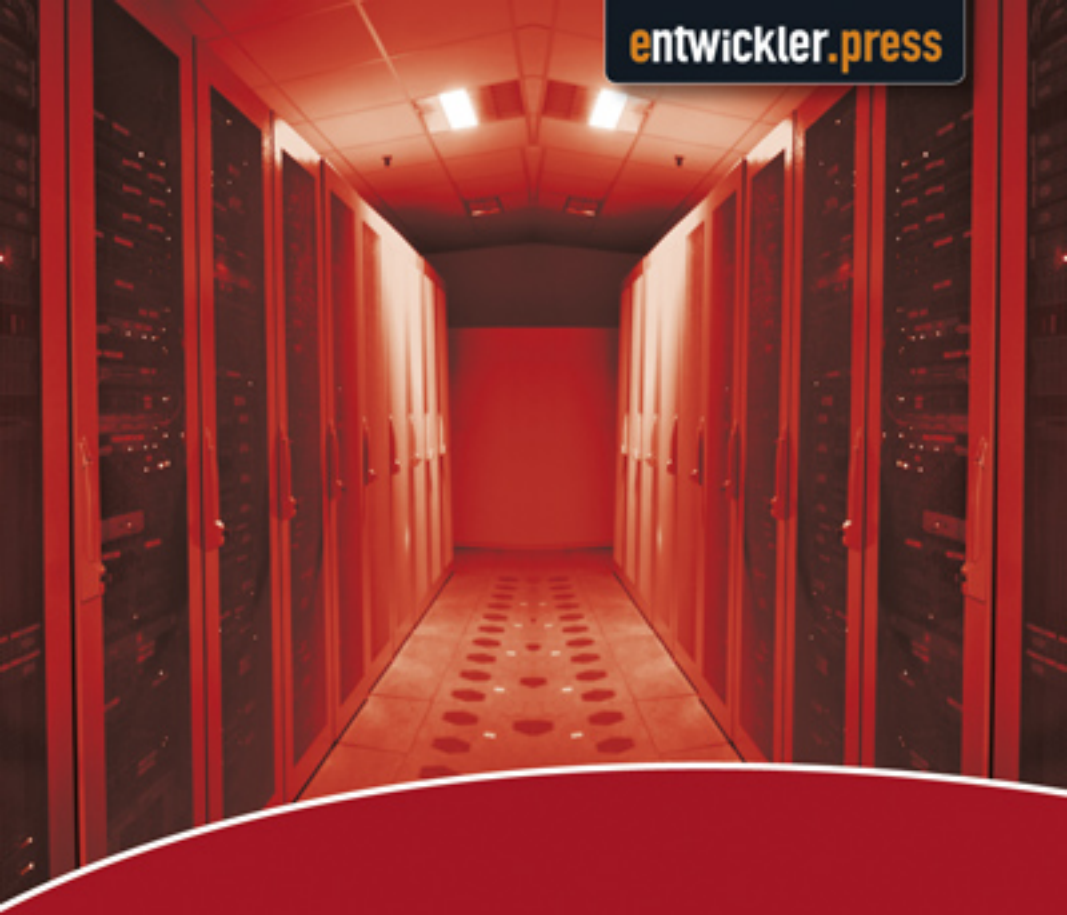

entwickler.press

SQL Server

Performance Ratgeber

Robert Panther

Robert Panther

SQL Server Performance-Ratgeber

Robert Panther

SQL Server Performance-Ratgeber

Datenbankoptimierung für Architekten, Entwickler und Administratoren

entwickler.press

Robert Panther: SQL Server Performance-Ratgeber
Datenbankoptimierung für Architekten, Entwickler und Administratoren
ISBN: 978-3-86802-232-2

Ein Imprint der Software & Support Verlag GmbH

Bibliografische Information der Deutschen Nationalbibliothek
Die Deutsche Nationalbibliothek verzeichnet diese Publikation in der Deutschen Nationalbibliografie; detaillierte bibliografische Daten sind im Internet über http://dnb.d-nb.de abrufbar.

Ihr Kontakt zum Verlag und Lektorat:
Software & Support Verlag GmbH
entwickler.press
Geleitsstraße 14
60599 Frankfurt am Main
Tel: +49(0) 69 63 00 89 - 0
Fax: +49(0) 69 63 00 89 - 89
lektorat@entwickler-press.de
http://www.entwickler-press.de

Projektleitung: Sebastian Burkart
Lektorat und Korrektorat: Rüdiger Glaum
Layout: SatzWERK, Siegen (www.satz-werk.com)
Umschlaggestaltung: Maria Rudi
Belichtung, Druck & Bindung: M.P. Media-Print Informationstechnologie GmbH, Paderborn

Inhaltsverzeichnis

Vorwort

Die Vorgeschichte dieses Buches ist eine sehr lange, die ich hier – um nicht zu sehr ins Detail zu gehen – kurz zusammenfassen möchte:

Bereits im Jahre 1996, als ich meine Diplomarbeit zum Thema Optimierung von Datenbankanwendungen fertig stellte, hatte ich den Plan, dieses Thema auch mal in Form eines Buches zu verarbeiten.

Es folgten Verhandlungen mit diversen Verlagen, die meist darin endeten, dass dieses Thema als zu speziell angesehen wurde. Offensichtlich war die Zeit dafür noch nicht gekommen.

Mein erstes Buch schrieb ich dann Jahre später und zu einem völlig anderen Thema, nämlich über die Programmierung von Pocket-PCs. Das positive Feedback hierzu ließ mich noch ein weiteres Buch sowie zahlreiche Fachartikel in diesem Themenumfeld schreiben, wobei ich hauptberuflich eigentlich immer primär mit der Administration und Programmierung relationaler Datenbanksysteme – insbesondere MS SQL Server – zu tun hatte.

Inzwischen waren auch einige englischsprachige Bücher zur Performanceoptimierung derselben erschienen, aber nur sehr wenige in deutscher Sprache. Es scheint also, dass die Zeit nun endlich reif ist für dieses Buch, das Sie – liebe Leser – gerade in den Händen halten.

Bevor ich Sie nun dem eigentlichen Buchtext überlasse, möchte ich noch einigen Menschen danken, die dazu beigetragen haben, dass dieses Buch etwa 14 Jahre nach der ursprünglichen Idee dann doch noch erschienen ist:

Der erste Dank geht an die Mitarbeiter von entwickler.press und dem Software & Support Verlag. Hier möchte ich insbesondere Sebastian Meyen hervorheben, der mir durch konstruktive Gespräche in der Konzeptionsphase des Buches geholfen hat, dem Buch eine klarer definierte Zielrichtung zu geben. Aber auch meinem Lektor Sebastian Burkart, der mich während des Schreibens betreut hat, bin ich zu Dank verpflichtet. Dies insbesondere dafür, dass er mehrere Terminverschiebungen gegenüber dem Verlag erfolgreich vertreten hat, die notwendig waren, um das doch recht tief gehende Thema in der richtigen Qualität aufbereiten zu können.

Dazu gilt mein Dank allen anderen Leuten, die im Hintergrund mitgeholfen haben, seien es weitere Mitarbeiter des Software & Support Verlags, aber auch Freunde und Kollegen, die mich in verschiedenen Diskussionen mit weiteren Anregungen und Detailideen versorgt haben. Hier möchte ich insbesondere Thomas Küppers nennen, der mir viele hilfreiche Anregungen zu den ersten beiden Kapiteln gegeben hat, sowie Wolfgang Braun, der für mich das Kapitel acht qualitätsgesichert hat.

Ein weiterer Dank geht an Erik Franz, der mich als ehemaliger Verlagsleiter von entwickler.press im Vorfeld darin unterstützt hat, das Thema bei diesem Verlag zu platzieren.

Ach ja, und wie es so schön heißt: „last but not least“ ein großes Dankeschön an meine Frau Birgit, die im vergangenen Jahr sicherlich zu wenig Aufmerksamkeit von mir bekam, weil ich neben meinem „normalen“ Job als IT-Berater gleich zwei Buchprojekte im SQL-Server-Umfeld zu bewältigen hatte.

Aber nun wünsche ich viel Spaß beim Lesen des SQL-Server-Performanceratgebers.

Robert Panther

Königstein im Januar 2010

E Einleitung

E.1 Warum dieses Buch?

Datenbanken bilden das zentrale Rückgrat von fast jeder Businessanwendung. Ein Thema, was dabei oft vernachlässigt wird – oder zumindest zu spät Beachtung findet –, ist die Performanceoptimierung.

Es gibt mittlerweile ja zumindest in englischer Sprache einiges an Literatur zum Thema Datenbankperformance. Der wesentliche Unterschied an dem Buch, das Sie gerade in den Händen halten, ist aber – abgesehen von der deutschen Sprache – die bewusste Aufteilung nach Zielgruppen. Bei immer größeren und komplexeren Anwendungen kommt es nur noch selten vor, dass ein und dieselbe Person den Server aufsetzt, die Datenbank sowie dazugehörige Software entwirft, programmiert und später auch noch wartet.

Daher ist die Struktur – insbesondere der ersten drei Abschnitte – dieses Buches konsequent so gegliedert, dass IT-Profis, die eine bestimmte Rolle ausüben, schnell die für sie relevanten Informationen finden, ohne sich diese umständlich aus verschiedenen Quellen zusammensuchen zu müssen.

Aber auch im Rest des Buches steht die Praxisorientierung im Vordergrund, denn dort werden praxiserprobte Vorgehensweisen geschildert, mit denen Sie eine gute Performance Ihrer Datenbank bzw. Datenbankanwendung erreichen oder wiederherstellen können.

Was Sie von diesem Buch nicht erwarten sollten, sind vollständige Auflistungen aller möglichen Detaileinstellungen, um noch die letzten 0,02% Performanceverbesserung zu erreichen. Dafür gibt es sicherlich andere Quellen und dies würde auch dem pragmatischen Ansatz des Textes widersprechen. Schließlich sollen hier auf möglichst knappem Raum die wesentlichen Informationen untergebracht werden, anstatt sich in endlosen Details zu verlieren, die hunderte von Seiten füllen, dabei aber nur noch marginale Performancevorteile erzielen.

Man kann dieses Buch also eher als eine Art SQL-Server-Performance-Basiswerk sehen, im Sinne von:

> *Was jeder SQL-Server-Datenbankadministrator, -architekt oder -entwickler über Performanceoptimierung wissen sollte.*

Dabei werden bewusst auch einige Grundlagenthemen (wie beispielsweise die Funktionsweise von Indizes) angesprochen, um sicherzustellen, dass diese Grundlagen nicht nur vage verstanden, sondern auch wirklich verinnerlicht wurden.

Oft habe ich in der Praxis erlebt, dass Leute über ein oberflächliches Wissen von sehr vielen Details verfügen, die wirklichen Grundlagen – und damit auch Zusammenhänge – aber nie ganz verstanden haben. Aus falsch verstandener „Professionalität“ kann es dann vor-

kommen, dass dieser Zustand dauerhaft anhält, weil man nur ungern die Schwäche zeigt, eventuell eine Wissenslücke in einem Grundlagenthema zu offenbaren. In diesem Sinne kann man das vorliegende Buch auch unter dem folgenden Aspekt sehen:

> *Was Sie schon immer über SQL-Server-Performance wissen wollten, aber nie zu fragen gewagt haben!*

Ich hoffe, Sie lassen sich von dieser – zugegebenermaßen etwas provokanten – Aussage nicht zu sehr abschrecken, denn diese trifft natürlich (hoffentlich) nur auf einige wenige zu und die Grundlagenthemen füllen daher auch nur einen Teil des gesamten Textes. Auch für erfahrene SQL-Server-Profis sollten zahlreiche interessante Informationen und Anregungen im Text zu finden sein.

Dies bringt mich zur wichtigen Frage:

E.2 Für wen ist dieses Buch gedacht?

Dieses Buch richtet sich an alle, die direkt oder indirekt mit SQL-Server-Datenbanken zu tun haben. Damit sind neben Datenbankadministratoren, Datenbankarchitekten und Datenbankentwicklern auch Anwendungsentwickler und Systemadministratoren angesprochen.

Die Begriffe Administratoren, Architekten und Entwickler werden jedoch in jedem Unternehmen im Detail etwas anders interpretiert und damit auch gelebt. Oft wird sogar noch eine feinere Unterteilung vorgenommen, sodass sich folgende Rollen ergeben können:

- Systemadministratoren
- Datenbankadministratoren
- Softwarearchitekten
- Datenbankarchitekten
- Software-/Anwendungsentwickler
- Datenbankentwickler

Zwischen den Softwarearchitekten und den -entwicklern ist mit den Software- bzw. Anwendungsdesignern sogar noch eine weitere Rolle möglich. Der Einfachheit halber unterscheide ich hier jedoch lediglich zwischen drei grundlegenden Rollen, die jeweils für einen bestimmten Aspekt der Optimierung zuständig sind.

- Administratoren: Optimierung der Systemumgebung
- Architekten: Optimierung der Datenhaltung
- Entwickler: Optimierung des Datenzugriffs

Dabei sind die Grenzen der Zuständigkeiten oft fließend (so sind Datenbankadministratoren beispielsweise auch für die Datenhaltung mit verantwortlich, allerdings auf einer anderen Ebene als die Datenbankarchitekten, die das Datenmodell entwerfen). Dadurch wird aber auch klar, dass eine enge Zusammenarbeit der verschiedenen Rollen ein wesentlicher Faktor für den Gesamterfolg einer guten (und performanten) Datenbankanwendung ist.

In diesem Sinne können die bereits erwähnten Grundlagenthemen auch dazu genutzt werden, sich zumindest die performancerelevanten Grundlagen der jeweils anderen Rollen anzueignen. Dies erleichtert die Kommunikation und damit auch die Zusammenarbeit ungemein, da auf diesem Wege Administratoren, Architekten und Entwickler dieselbe Sprache sprechen.

E.3 Erforderliche Kenntnisse und Voraussetzungen

Dieses Buch soll bewusst keine Einführung in Administration oder Programmierung des Microsoft SQL Servers darstellen. Dazu gibt es einerseits bereits genügend andere gute Bücher und andererseits würde dies auch den inhaltlichen Rahmen sprengen und vom eigentlichen Thema ablenken.

Es werden daher Grundkenntnisse in der Nutzung des Microsoft SQL Servers vorausgesetzt. Je nach Rolle (Administrator, DB-Entwickler etc.) sollten Sie in der Lage sein, einen SQL Server einzurichten, Datenbanken mit Tabellen und Indizes anzulegen und grundlegende SQL-Statements (vor allem *SELECT*, *INSERT*, *UPDATE*, *DELETE*) zu formulieren.

Auch die Verwendung der wichtigsten Tools sollte Ihnen dabei geläufig sein: SQL Server Enterprise Manager (bis SQL Server 2000) bzw. SQL Server Management Studio (ab SQL Server 2005). Eine Kenntnis der spezielleren (performancerelevanten) Zusatztools wie beispielsweise SQL Server Profiler oder SQL-Server-Optimierungsratgeber ist dagegen nicht erforderlich, da die Verwendung dieser Tools im Buch erklärt wird.

Generell spielt es keine Rolle, ob Sie auf einem einzelnen Arbeitsplatz mit der kostenfreien Express Edition von SQL Server oder in einem großen Netzwerk mit mehreren Enterprise Editionen von SQL Server arbeiten, denn die meisten Optimierungsansätze treffen für alle Varianten des Produkts zu. Dort, wo sich einzelne Themen nur auf einige Editionen von SQL Server beziehen, wird explizit darauf hingewiesen. Dasselbe gilt für die verwendete Version von SQL Server. Dabei wird im Zweifelsfall von der aktuellsten Produktversion (SQL Server 2008) ausgegangen, jedoch wird an den entsprechenden Stellen auch auf die Besonderheiten von SQL Server 2005 (und teilweise auch SQL Server 2000) hingewiesen.

Um wirklich alle hier vorgestellten Tools verfügbar zu haben, verwenden Sie idealerweise die Developer oder Enterprise Edition von SQL Server 2008. Dabei ist die Developer Edition eine – insbesondere im Vergleich zur Enterprise Edition – sehr günstige Variante, die allerdings nur für Entwicklungszwecke lizenziert ist. Von der Enterprise Edition bietet Microsoft aber auch eine 180-Tage-Testversion an, die Sie im Internet kostenfrei herunterladen können.

Damit die gezeigten Abfragen und Beispiele möglichst gut nachzuvollziehen sind, wird in den meisten Fällen die AdventureWorks-Beispieldatenbank (in der 2008er-Version) verwendet. Diese wird seit einiger Zeit nicht mehr mit dem SQL Server ausgeliefert, ist aber als kostenloser Download über die Codeplex-Website verfügbar.

Downloadlink: *http://www.codeplex.com/SqlServerSamples*

E.4 Aufbau des Buches

Der gesamte Text des Buches ist – abgesehen vom Anhang – in vier große Abschnitte unterteilt. Davon tragen die ersten drei Abschnitte des Buches den bereits erwähnten Zielgruppen Rechnung, da hier für jede Gruppe ein Abschnitt zu finden ist, der die wichtigsten Faktoren beschreibt, die aus dem jeweiligen Blickwinkel zu beachten sind.

- *Teil A: Optimierung für Administratoren*
- *Teil B: Optimierung für Architekten*
- *Teil C: Optimierung für Entwickler*
- *Teil D: Optimieren einer bestehenden Datenbank(-anwendung)*

Durch diese Unterteilung nach Zielgruppen können Sie schneller den Teil des Buches finden, der für Sie in Ihrer alltäglichen Arbeit die größte Bedeutung hat. Wenn Sie also beispielsweise Datenbankanwendungen entwerfen, starten Sie doch gleich mit den Kapiteln für Datenbankarchitekten. Als Anwendungsprogrammierer können Sie sich stattdessen gezielt auf die Kapitel für Entwickler stürzen.

Und wenn Sie tatsächlich alle diese Rollen in einem Projekt ausüben sollten (was gerade bei kleineren Projekten leicht der Fall sein kann), so können Sie natürlich auch alle Kapitel lesen. Dabei orientiert sich die Reihenfolge der Kapitel so weit wie möglich an einem üblichen Software-Entwicklungszyklus. Erst wird der entsprechende Datenbankserver aufgesetzt, anschließend wird die Datenbank entworfen. Im nächsten Schritt erfolgt die Implementierung durch die Programmierer, bevor die Anwendung letzten Endes in den Produktivbetrieb übergeht. Lediglich das Kapitel, in dem die regelmäßige Wartung einer Datenbank beschrieben wird, weicht hiervon ab und ist bereits in *Teil A* zu finden, da hier auch die anderen für Administratoren relevanten Themen behandelt sind.

Der vierte Teil des Buches widmet sich der Optimierung von bestehenden Anwendungen. Hier erfahren Sie, wie man am besten vorgeht, um aus einer bereits bestehenden und produktiv eingesetzten Datenbankanwendung durch möglichst geringe Änderungen das Maximum an Performancegewinn zu erzielen. Dabei werden die verschiedenen Zielgruppen wieder zusammengeführt, da hier eine Zusammenarbeit von Administratoren, Architekten und Entwicklern gefragt ist, um möglichst effektive Ergebnisse zu erzielen.

Zum Schluss folgt dann noch ein Anhang, in dem ein Überblick über die wichtigsten Tools zur Performanceoptimierung zu finden ist. Außerdem sind hier auch eine Auflistung von Quellen zu finden, aus denen man weitere Informationen zum Thema Datenbankperformance erhalten kann, sowie ein kurzes Glossar, in dem die wichtigsten Begriffe und Abkürzungen aufgelistet sind.

E.5 Schreibweisen

Damit der Buchtext besser lesbar ist, werden verschiedene typografische Konventionen für bestimmte Inhalte verwendet. Diese möchte ich hier kurz auflisten:

Verweise auf andere Kapitel werden in dieser Form dargestellt: siehe *Einleitung – Schreibweisen*

Verweise auf Webseiten werden wie folgt dargestellt: *http://www.PantherPhotoPress.de*

Für Zitate ist die folgende Schreibweise vorgesehen!

Eine ähnliche Schreibweise wird für Texte verwendet, die auf dem Bildschirm zu sehen sind, aber auch für *allgemeine Hervorhebungen von Begriffen, die im Text neu eingeführt werden*.

Quelltextausschnitte werden *in dieser Schrift* wiedergegeben. Dieselbe Schrift wird auch für Datenbankobjekte (Tabellen, gespeicherte Prozeduren etc.) sowie Datei- und Verzeichnisnamen verwendet.

```
SELECT * FROM Person.Address

SELECT DISTINCT * FROM Person.Address
-- Hin und wieder sind Teile von Listings durch Fettdruck hervorgehoben

SELECT * FROM Tabellenname
-- Platzhalter, die durch die eigentlichen Objektnamen zu ersetzen sind,
-- werden kursiv dargestellt
```

Listing E.1: Ganze Quelltexte werden in der oben gezeigten Schreibweise dargestellt

Gelegentlich findet sich auch ein Kasten, in dem ein Exkurs (z.B. eine Hintergrundinformation), ein besonders hervorzuhebender Hinweis oder Tipp dargestellt ist.

Bezüglich der Sprachwahl für Fachbegriffe aus dem SQL-Server-Umfeld versuche ich möglichst, sowohl die deutschsprachige als auch die englischsprachige Bezeichnung zu erwähnen, da in vielen – teilweise selbst deutschsprachigen – Texten oft die englischen Bezeichnungen bevorzugt verwendet werden. In vielen deutschen Unternehmen gibt es sogar generelle Richtlinien, dass auf Servern ausschließlich die englischsprachigen Softwareversionen zu installieren sind, da für diese Updates und Patches oft etwas früher verfügbar sind. Beim SQL Server war dies in der letzten Zeit zwar weniger der Fall (hier erschienen die wichtigsten Sprachvarianten gleichzeitig und lediglich exotische Sprachen kamen später dazu). Wenn Sie aber zu einem bestimmten Begriff nach Informationen im Internet suchen, finden Sie definitiv deutlich mehr Treffer, wenn Sie nach den englischen Bezeichnungen suchen. Von daher schadet es also sicher nicht, auch diese zu kennen, selbst wenn Sie mit der deutschen Software arbeiten.

E.6 Website zum Buch

Auf meiner Website *http://www.PantherComputing.de* ist ein Unterbereich zu diesem Buch zu finden. Dort werden Ergänzungen und Fehlerkorrekturen (falls jemand einen Fehler findet, wovon ich ausgehe, denn „Nobody is perfect!“), auch weitere Tipps und Verweise auf zusätzliche Informationen (sowohl im Internet als auch Buchempfehlungen) zu finden sein. Dieser Bereich ist momentan noch recht überschaubar, da ich mit dem Schrei-

ben dieses Buches beschäftigt war, wird im Laufe der Zeit aber sicherlich deutlich mehr Inhalt bekommen, also schauen Sie öfter mal vorbei!

E.7 Über den Autor/Kontakt zum Autor

Damit Sie wissen, wer für diesen Text verantwortlich ist, möchte ich mich kurz vorstellen:

Ich beschäftige mich mit dem Microsoft SQL Server bereits, seit im Jahre 1995 mit der Version 6.0 die erste echte Microsoft-Version des Produkts veröffentlicht wurde. Seit dieser Zeit war auch immer wieder die Datenbank-Performanceoptimierung ein zentrales Thema für mich. So kam es auch, dass ich dies zum Thema meiner Diplomarbeit an der FH-Darmstadt wählte, die ich im Jahre 1996 erfolgreich abschloss.

In den Jahren danach kamen zahlreiche Projekte mit verschiedenen Versionen des SQL Servers (6.0, 6.5, 7.0, 2000, 2005 und 2008), bei denen ich alle möglichen Rollen (DB-Administrator, DB-Architekt, DB-Entwickler, Anwendungsentwickler, Projektleiter und Trainer) einnahm und so das Produkt aus verschiedenen Blickwinkeln besser kennen lernen konnte. Inzwischen bin ich als Senior Consultant für die Logica Deutschland GmbH & Co. KG mit dem Schwerpunkt SQL Server tätig.

Daneben schreibe ich Fachartikel und Bücher zu meinen Spezialgebieten SQL Server und Anwendungsentwicklung für Mobile Devices. Hin und wieder bin ich zu diesen Themen auch als Sprecher auf Konferenzen wie beispielsweise BASTA! und SQLCON aktiv.

Für Fragen oder Feedback zu diesem Buch können Sie mit mir über folgende Adresse in Kontakt treten: *sqlserver@panthercomputing.de*

Auch wenn es manchmal projektbedingt etwas länger dauern kann, versuche ich jede Mail zu beantworten. Erwarten Sie aber nicht, zu jedem Performanceproblem eine allumfassende Antwort zu bekommen, mit der Sie das Problem im Nu beheben können.

Dies ist erfahrungsgemäß in den wenigsten Fällen möglich, ohne das Gesamtsystem aus Server und Anwendung ausführlich unter die Lupe zu nehmen und im Detail zu analysieren. Bei der Performance von SQL-Datenbanken spielen so viele Faktoren eine Rolle, dass sich zwar leicht exakte Regeln finden lassen, was man nicht tun sollte, aber nur selten solche, die besagen, wie man die optimale (also maximal mögliche) Performance erreicht. Und letzten Endes bin auch ich – obwohl ich glaube, den SQL Server mittlerweile recht gut zu kennen – sicherlich nicht allwissend. Denn das Produkt ist inzwischen so komplex geworden, dass ich auch nach jahrelanger Beschäftigung damit immer wieder ein paar neue Details entdecke.

Sollten Sie mal auf jemanden treffen, der von sich behauptet, über den SQL Server alles zu wissen, sollten Sie diesem sehr skeptisch begegnen. Mit größter Wahrscheinlichkeit kennt er das Produkt so wenig, dass er nicht einmal weiß, was er alles nicht weiß. Und sollte sich wider Erwarten doch herausstellen, dass er tatsächlich alles zu diesem Produkt weiß, schicken Sie mir bitte seine Adresse. Ich hätte da sicherlich ein paar Fragen, die schon lange auf eine Antwort warten. ;-)

Teil A

Optimierung für Administratoren

Dieser Teil des Buches fasst die Optimierungsansätze zusammen, die sowohl für Systemadministratoren als auch für Datenbankadministratoren relevant sind. Je nach Unternehmens- bzw. Projektgröße können sich diese Rollen auch stark überschneiden oder sogar von denselben Personen wahrgenommen werden.

Dabei richtet sich *Kapitel 1 – Vorbereitung der Systemumgebung* primär an Systemadministratoren, während *Kapitel 2 Regelmäßige Wartungsarbeiten* vor allem für Datenbankadministratoren interessant sein dürfte.

1 Vorbereitung der Systemumgebung

1.1 Hardwareumgebung

Es ist weit verbreiteter Irrglaube, dass sich Datenbank-Performanceprobleme am einfachsten durch banale Hardwareaufrüstung lösen lassen. Insbesondere dann, wenn es darum geht, eine bestehende Umgebung zu optimieren, sollte dies eigentlich das letzte Mittel sein.

Das setzt natürlich voraus, dass bei der Ersteinrichtung des Servers eine angemessene Hardware zur Verfügung stand. Daher wird dieses Thema auch gleich zu Beginn des Buches behandelt.

Neben den in diesem Kapitel folgenden Informationen empfehle ich vor einer Neubestellung von Hardware auch einen kurzen Blick in den Windows-Server-Katalog. Hier sind alle zugelassenen Server für den Betrieb aufgeführt, wodurch auch gewährleistet wird, dass diese im Problemfall von Microsoft supported werden:

http://www.windowsservercatalog.com

1.1.1 Prozessor

Lange Zeit war die Angabe der Taktfrequenz das wesentliche Merkmal, um die Leistungsfähigkeit eines Prozessors zu beschreiben. Im Laufe der Jahre wurden die Frequenzen der Prozessoren immer weiter gesteigert, so dass diese von den 4,77 MHz der ersten PCs bis hin zu 2–3 GHz nahezu um den Faktor 1.000 gestiegen sind. Inzwischen sind die derzeit bekannten physikalischen Möglichkeiten zur Steigerung des Prozessortaktes jedoch weitgehend ausgeschöpft. Stattdessen wird die Entwicklung von parallelisierten Systemen bzw. Prozessoren mit höherer Transistordichte vorangetrieben, um eine weitere Leistungssteigerung zu erreichen.

So kommt es, dass bei typischen Unternehmensservern normalerweise mehrere Prozessoren verfügbar sind und sogar viele aktuelle Arbeitsplatzrechner – aufgrund moderner Prozessortechnologie – oft über bis zu vier Prozessorkerne verfügen, die in einem einzigen Prozessor enthalten sind. Dabei macht es keinen Unterschied, ob zwei Prozessoren mit jeweils zwei Kernen oder ein Prozessor mit vier Kernen im System arbeiten, entscheidend ist die Gesamtzahl der Prozessorkerne (engl. Cores).

Ob diese Prozessorkerne dann auch effektiv genutzt werden können, hängt davon ab, ob dies auch vom verwendeten Betriebssystem und der darauf laufenden Software unterstützt wird. Eine Software, die nicht multiprozessorfähig ist, wird auch auf einem System mit acht Prozessorkernen nicht schneller laufen als mit einem Prozessorkern desselben Typs.

Im Falle von SQL Server muss man sich darum allerdings weniger Sorgen machen. Lediglich die Anzahl der nutzbaren Prozessorkerne ist je nach verwendeter SQL Server Edition eventuell begrenzt. So nutzt die Express Edition nur einen Prozessorkern, während die Enterprise Edition mit beliebig vielen Prozessorkernen umgehen kann. (Einen genauen Überblick zu diesen Einschränkungen finden Sie weiter unten beim Abschnitt zur Auswahl der richtigen SQL Server Edition.)

In der Praxis ist es generell sinnvoll und üblich, ein Mehrprozessorsystem für die Serverhardware zu nutzen. Im einfachsten Fall können Betriebssystem und SQL-Server-Dienst auf verschiedenen Prozessorkernen laufen, aber auch die parallele Abarbeitung von verschiedenen Datenbankabfragen auf verschiedenen Prozessorkernen ist möglich. Der Geschwindigkeitszuwachs, der durch weitere Prozessorkerne erzielbar ist, steigt jedoch nicht linear, da durch die Verteilung von Prozessen auf Prozessorkerne auch ein zusätzlicher Verwaltungsaufwand entsteht. Man sollte sich also keine Hoffnung machen, dass eine Datenbank auf einem Server mit acht Kernen doppelt so schnell läuft als auf einem vergleichbaren System mit vier Kernen. In welchem Ausmaß sich die Leistung durch zusätzliche Prozessorkerne steigern lässt, hängt hauptsächlich davon ab, wie rechenintensiv die Datenbankanwendungen sind und wie viele Prozesse normalerweise gleichzeitig auf dem Server laufen.

Ein weiterer wichtiger Aspekt bei der Wahl des richtigen Prozessors ist die Größe des prozessorinternen Cache-Speichers. So kann es vorkommen, dass ein Prozessor mit einem größeren internen Cache eine bessere Gesamtleistung bringt als ein vergleichbarer Prozessor mit einer höheren Taktfrequenz, aber kleinerem CPU-Cache.

1.1.2 Hauptspeicher

Die besten Aussichten auf einen spürbaren Performancegewinn durch Hardwareaufrüstung hat man beim Hauptspeicher (oder kurz: *RAM* für *Random Access Memory*). Das belegt schon ein beliebter Running Joke unter Datenbankadministratoren:

> *Frage: »How much memory does SQL Server need?«*
> *Antwort: »MORE!«*

Das fängt schon bei der Nutzung durch das Betriebssystem an, denn wenn zu wenig physikalischer Hauptspeicher vorhanden ist, nutzt Windows eine Auslagerungsdatei auf der Festplatte, um auf diesem Weg zusätzlichen Speicher bereitzustellen. Da Festplatten natürlich weit langsamer sind als RAM-Speicher, bremst dies die Gesamtleistung des Systems natürlich deutlich aus.

Deutlich wird dieser Geschwindigkeitsunterschied schon, wenn man sich die Einheiten ansieht, in denen die typischen Zugriffszeiten für RAM-Speicher und Festplatten angegeben werden. Während die Zugriffszeiten für Festplatten in Millisekunden (ms) angegeben sind, bewegen sich die Zugriffszeiten von RAM-Speicher im Nanosekunden-Bereich (ns).(1 ms = 1 000 000 ns)

Diese Tatsache kann man natürlich auch zum Vorteil nutzen, indem man durch großzügig bemessenen Hauptspeicher ein Puffern (Caching) von oft verwendeten Daten ermöglicht, die sonst durch wesentlich langsamere Festplattenzugriffe gelesen werden müssten. So ist es bei gut konfigurierten Datenbankservern durchaus normal, dass über 90 % der Daten aus dem Hauptspeicher gelesen werden.

Diese Kette lässt sich sogar noch fortsetzen, wenn man den weiter oben bereits erwähnten prozessorinternen CPU-Cache berücksichtigt. Dieser ist durch seine Integration in den Prozessor wiederum um ein Vielfaches schneller ansprechbar als der normale Hauptspeicher, sodass man auch hier von einem möglichst großen Cache profitieren kann. Allerdings lässt sich der Prozessorcache im Gegensatz zum Hauptspeicher natürlich nicht mehr nachträglich aufrüsten.

Für eine Aufrüstung des Hauptspeichers spricht außerdem, dass die RAM-Preise seit einigen Jahren so niedrig sind, dass Hauptspeicher auch eine der günstigsten Varianten für eine Hardwareaufrüstung darstellen. Allerdings müssen – je nach Gesamtgröße des Hauptspeichers – auch einige Voraussetzungen bei Betriebssystem und SQL Server erfüllt sein, damit dieser auch genutzt werden kann. Diese werden in *Kapitel 1.3.1 – Auswahl der richtigen Edition* behandelt.

1.1.3 Festplatten

Da der langsame Zugriff auf Festplatten einer der typischen Flaschenhälse in der Gesamtperformance eines Datenbankservers darstellt, gibt es auch in diesem Bereich einige Ansätze, dieses Übel so weit wie möglich zu minimieren.

Parallele Zugriffe durch Verteilung auf mehrere Platten ermöglichen

Wie bei den Prozessoren basieren diese meist auf Ausnutzung von parallelen Zugriffen. Dies bedeutet im einfachsten Fall, dass Betriebssystem und SQL-Server-Datenbanken auf verschiedenen Festplatten (nicht Partitionen, sondern physikalisch getrennten Platten) untergebracht werden. Dadurch können die Daten nahezu parallel gelesen und geschrieben werden. Gleichzeitig wird beim Lesen von größeren Datenblöcken die Notwendigkeit minimiert, den Schreib-/Lesekopf häufig neu zu positionieren. Dieses Prinzip lässt sich nahezu beliebig fortsetzen, sodass für eine optimale Konfiguration separate Datenträger für folgende Dateien verwendet werden:

- System (Betriebssystem & SQL Server)
- Windows-Auslagerungsdatei
- Datenbankdateien
- Protokolldateien
- TempDB (Datenbank & Protokoll)
- Backupdateien

Wenn weitere Platten verfügbar sind, lassen sich auch noch Datenbank- und Protokolldatei der TempDB trennen sowie verschiedene Datenbanken auf separate Platten verteilen. Sollten Filestreams verwendet werden, können auch diese auf eine weitere Platte gelegt werden.

Für kleinere Abteilungsserver stehen in der Regel aber eher deutlich weniger Plattensysteme zur Verfügung, sodass man sich hier darauf beschränken muss, die Dateien voneinander zu trennen, die am häufigsten gemeinsam benötigt werden. Dies sind vor allem System-, Datenbank-, Protokolldateien und TempDB. Oft bringt es schon einen spürbaren Performancegewinn, die eher leselastigen Dateien (Systemdateien und Datenbank) von den schreiblastigeren (Protokoll und TempDB) zu trennen.

Die Verteilung der Dateien wird natürlich durch den Datenbankadministrator vorgenommen, der Systemadministrator muss aber die entsprechenden Plattensysteme zur Verfügung stellen, sodass auch hier eine enge Zusammenarbeit erforderlich ist.

Einsatz von RAID-Systemen für Sicherheit und Performance

Ursprünglich primär entwickelt, um die Datensicherheit zu erhöhen, sind heute in fast allen produktiven Serverumgebungen sogenannte *RAID*-Systeme im Einsatz. Die Abkürzung RAID stand ursprünglich für *Redundant Array of Inexpensive Disks* (inzwischen gilt die marketingtauglichere Variante *Redundant Array of Independent Disks*) und bezeichnet damit den Zusammenschluss von mehreren preiswerten (oder unabhängigen) Festplatten, auf denen die Daten redundant gespeichert werden, um beispielsweise den Komplettausfall einer einzigen Platte ohne Datenverlust abfangen zu können. Durch die redundante Speicherung kommt als positiver Nebeneffekt hinzu, dass zumindest Lesezugriffe mitunter deutlich beschleunigt werden, da die Daten nun parallel gelesen werden können. Je nach verwendeter RAID-Variante kann sich dies allerdings negativ auf die Schreib-Performance auswirken, da die redundanten Daten ja auch zusätzlich geschrieben werden müssen.

In den RAID-Implementierungen ist zwischen einem Software- und einem Hardware-RAID zu unterscheiden. Das Software-RAID wird durch eine Software (z.B. das Betriebssystem) gebildet. Im Falle des Hardware-RAID ist ein spezieller Controller für die Bildung des RAID verantwortlich. Auf dieser Controller Card ist ebenfalls Software in einem Speicherbaustein abgelegt, die jedoch nicht durch das Betriebssystem gesteuert wird.

Allen RAID-Varianten gemeinsam ist die Tatsache, dass alle Laufwerke eines RAID-Verbundes zusammen als ein logisches Laufwerk im Betriebssystem erscheinen. Die verschiedenen RAID-Varianten werden als RAID-Level bezeichnet, von denen ich die gängigsten drei kurz beschreiben will.

RAID 0 (Striping) – Geschwindigkeit ohne Netz und doppelten Boden RAID 0 verdient eigentlich nur den Namen AID (oder auch „0-redundant AID“), denn der Aspekt der redundanten Datenspeicherung wurde hier weggelassen. Eigentlich werden hierbei lediglich zwei physikalische Datenträger zu einem logischen Datenträger vereint. Eine erhöhte Ausfallsicherheit ist hier also nicht gegeben. Man bedient sich dafür allerdings eines einfachen Tricks, um zumindest einen Performancegewinn zu erhalten: Und zwar werden die Daten nicht fortlaufend auf den Platten abgelegt, sondern blockweise abwechselnd – quasi im Reißverschlussverfahren. Dadurch können sowohl beim Lesen als auch beim Schreiben beide Platten parallel genutzt werden. Nachteilig ist dabei aller-

dings die Tatsache, dass bei einem Plattendefekt gleich alle Daten verloren sind, da jede Datei ihre Daten auf beide Platten verteilt hat. (Lediglich dann, wenn eine Datei klein genug ist, um komplett in einen der – meist 64 KB großen – Blöcke zu passen, wäre diese rein theoretisch noch zu retten, wenn die andere Platte ausfällt.)

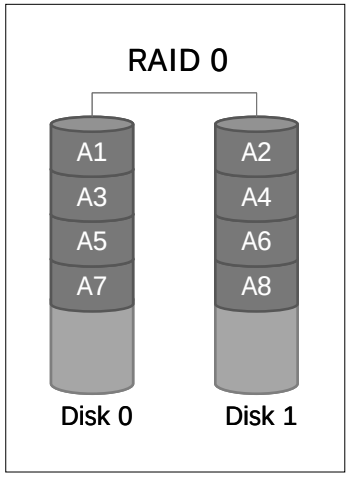

Abbildung 1.1: Aufbau eines RAID-0-Systems

Der typische Einsatz eines RAID-0-Systems ist also ein Szenario, in dem viele Daten schnell gelesen oder geschrieben werden (optimalerweise sequenziell) und die Ausfallsicherheit eher eine untergeordnete Rolle spielt. Bei einem SQL Server wäre dies beispielsweise geeignet, um die TempDB zu speichern, da diese nach einem Neustart des SQL-Server-Dienstes ohnehin geleert wird und daher keine dauerhaft relevanten Daten enthält.

RAID 1 (Mirroring) – Spieg'lein, Spieg'lein an der Wand ... RAID 1 ist ebenso einfach erklärt wie RAID 0, handelt es sich hierbei lediglich um zwei (oder mehr) Platten, die den exakt identischen Inhalt speichern. Somit ist automatisch eine gewisse Ausfallsicherheit gegeben, denn bei einem Plattendefekt kann jede Platte alle anderen jederzeit vollends ersetzen.

Allerdings geht dies auch stark auf Kosten der Kapazität, denn die gesamte nutzbare Kapazität entspricht der Speicherkapazität der kleinsten Platte im Verbund. Daher werden bei RAID 1 meist auch nur zwei (möglichst gleich große) Festplatten verwendet, um möglichst wenig Platz zu verschwenden. Dennoch wird selbst in dieser Konfiguration natürlich die doppelte Menge der eigentlich benötigten Kapazität erforderlich.

In Bezug auf Performance ist RAID 1 besonders beim Lesen schnell, denn auch hier kann von allen beteiligten Platten parallel gelesen werden. Da alle Platten dieselben Informationen vorhalten, ist es sogar egal, welche Daten woher gelesen werden, sodass die schnelleren Platten nicht warten müssen, bis die langsameren fertig sind, sondern stattdessen weitere Daten lesen können. Beim Schreiben ist dagegen kein Performancegewinn zu erhoffen, da alle Schreiboperationen gleichzeitig auf allen Platten ausgeführt werden müssen. Aufgrund der notwendigen Synchronisierung ist hierbei sogar mit einem minimalen Performanceverlust zu rechnen.

Hier begegnen wir also zum ersten Mal einem immer wiederkehrenden Prinzip der Performanceoptimierung: „Redundante Datenhaltung erhöht die Lesegeschwindigkeit auf Kosten der Schreibgeschwindigkeit!“ (Wobei der negative Einfluss auf die Schreibgeschwindigkeit in diesem Fall wirklich vernachlässigbar klein ist.)

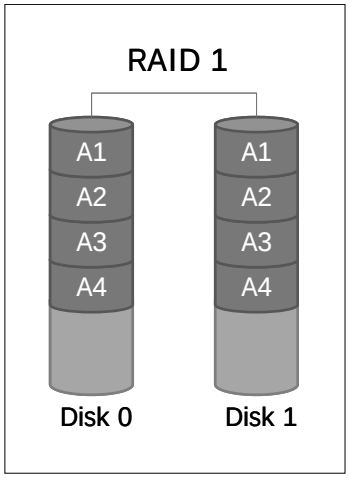

Abbildung 1.2: Aufbau eines RAID-1-Systems

RAID 1 eignet sich also für Daten, die vor allem schnell gelesen werden sollen und dabei eine gewisse Ausfallsicherheit benötigen. Bei einem Datenbankserver trifft dies am ehesten auf die Systempartition zu, auf der das Betriebssystem und die SQL-Server-Software gespeichert sind.

RAID 5 – Performant und redundant Das RAID-Level 5 bietet sowohl Ausfallsicherheit als auch Performance und das bei vergleichsweise geringen Eingeständnissen bezüglich der nutzbaren Gesamtkapazität. Das verwendete Prinzip basiert darauf, dass bei einer bestimmten Anzahl von Festplatten jeweils eine zusätzliche Platte Paritätsinformationen (quasi eine Prüfsumme, basierend auf einer XOR-Verknüpfung) speichert. Dadurch ist beim Ausfall einer beliebigen Platte im Verbund der fehlende Wert durch die Paritätsinformation rekonstruierbar. Um die Platten möglichst gleichmäßig auszulasten, wird – je nach Position des Blocks auf der Platte – jeweils eine andere Platte zur Speicherung der Paritätsinformationen verwendet.

Beim Lesen der Daten kann teilweise ein deutlicher Performancegewinn erzielt werden, da auf mehrere Platten parallel zugegriffen werden kann. Dafür muss man beim Schreiben mit geringen Performanceeinbußen rechnen, da neben dem Schreiben des eigentlichen Werts

auch alle entsprechenden Blöcke auf den anderen Platten gelesen und die Paritätsinformation aktualisiert werden muss. Im Gegensatz zu RAID 1 ergibt sich ein Performancenachteil im Fehlerfall. Denn sobald eine Platte ausgefallen ist, müssen die Blöcke auf allen verbleibenden Platten gelesen werden, um die Daten des fehlenden Blocks zu rekonstruieren. Bei RAID 1 kann dagegen die verbleibende Platte mit normaler Geschwindigkeit weiter genutzt werden. Erst wenn die defekte Platte ausgetauscht wird, entstehen zusätzliche Performanceeinbußen, da diese dann erst einmal mit den richtigen Daten versorgt werden muss (dies gilt dann aber sowohl für RAID 1 als auch für RAID 5).

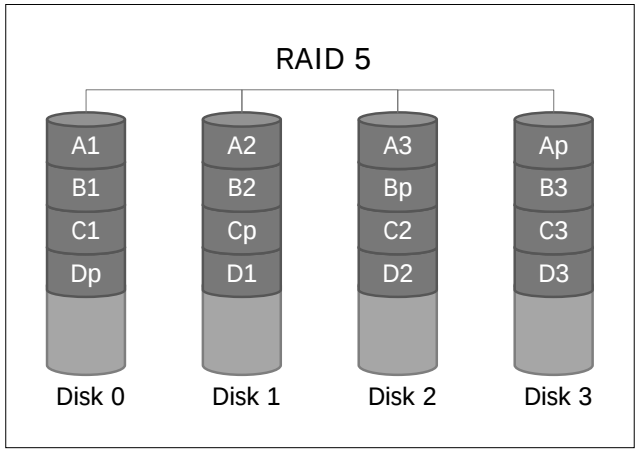

Abbildung 1.3: Aufbau eines RAID-5-Systems

Aufgrund der Kombination von Ausfallsicherheit und Performancegewinn ist RAID 5 normalerweise die bevorzugte Variante zum Speichern von Datenbank-Backups. Für das Speichern von Datenbank- und Protokolldateien ist RAID 5 ebenfalls gut geeignet, hier gibt es jedoch noch Alternativen, die eine deutlich bessere Performance bieten.

RAID 01 und RAID 10 – „gespiegelte Streifen" und „gestreifte Spiegel" Bei diesen beiden Varianten handelt es sich um die Kombination aus RAID 0 und RAID 1. Dabei werden bei RAID 01 mehrere RAID 0-Systeme zu einem RAID 1-System zusammengeschlossen. Bei RAID 10 dagegen werden mehrere RAID-1-Systeme zu einem RAID-0-System zusammengefügt.

Beide Varianten bieten sowohl die Ausfallsicherheit von RAID 1 als auch die gesteigerte Performance (sowohl für schreibende als auch für lesende Zugriffe) von RAID 0. Auf der anderen Seite kommt hier natürlich auch der RAID-1-typische Nachteil zum Tragen, dass die doppelte Plattenkapazität benötigt wird. Wenn man mit diesem Manko leben kann, bieten diese beiden Varianten eine hervorragende Basis für Ausfallsicherheit in Kombination mit optimaler Performance. Damit eignen sich diese kombinierten Varianten bestens zum Speichern von Datenbank- und Protokolldateien.