



Transbase® Version 6.9

Die Transbase® Version 6.9 weist einige Neuerungen auf, die vor allem auf neuen Rechnerarchitekturen, die durch 64-Bit-Prozessoren, Multi-Core-CPUs, große Hauptspeicher und schnelle RAID-Plattensubsysteme charakterisiert sind, zu erheblichen Leistungsverbesserungen führen.

Neuerungen im Bereich Prozessarchitektur sind:

- 64 Bit Support
- Elimination dynamischer TCP/IP Ports
- Replikation und Datenbank-Grids

Neuerung im SQL Kern sind:

- WITH Klausel
- Parallele Queryverarbeitung
- Cache Partitionierung
- CLOBs

Prozessarchitektur

64 Bit Support

Moderne Rechnerarchitekturen verfügen heute über sehr große Hauptspeicher, die wesentlich über die mit 32 Bit adressierbaren 4 GByte hinausgehen. Um diesen Hauptspeicher auch aus einem einzelnen Prozess in vollem Umfang auszunützen, muss ein solcher Prozess 64 Bit breite Adressen verwenden. Seit Version 6.9 ist Transbase® in einer 64-Bit-Version verfügbar, die sich im wesentlichen dadurch von der ebenfalls verfügbaren 32-Bit-Version unterscheidet.

Damit einhergehend wurden einige Systemparameter in ihren Konfigurationsbereichen deutlich erweitert; dies betrifft z.B. die Größe des globalen Transbase® Caches, der nunmehr bis zu 128 x 1 GByte (128 GByte) groß konfiguriert werden kann. Dieser globale, von allen Transbase® Prozessen geteilte Cache wird in jeden Prozessadressraum eingebunden, was

Kontakt

Transaction Software GmbH
Willy-Brandt-Allee 2
81829 München

Tel.: +49 89 / 627 09 - 0
Fax: +49 89 / 627 09 - 11

info@transaction.de
www.transaction.de
www.transbase.de



unmittelbar zu Prozessgrößen jenseits der 4 GByte führt.

Auch der sogenannte lokale Cache könnte in seiner Größe entsprechend größer als 2 GByte konfiguriert werden, wobei hierbei allerdings zu berücksichtigen ist, dass der lokale Cache von jedem einzelnen Prozess angefordert wird und deswegen dessen Größe unbedingt sorgfältig gewählt werden sollte, um zu verhindern, dass der physische Speicher überbeansprucht wird.

Elimination dynamischer TCP/IP Ports

In bisherigen Versionen wurde die Kommunikation zwischen Transbase® Client- und Serverprozess über zwei feste Ports (in der Regel 2024 und 2025) sowie eine gewisse Anzahl dynamischer Ports abgewickelt.

Seit Version 6.9 kann Transbase® so konfiguriert werden, dass nur noch die beiden festen Ports zur Kommunikation verwendet werden. Dies erleichtert den Betrieb von Transbase® erheblich, wenn zwischen Client und Server ein sogenannter Firewall eingezogen ist, weil nur diese Ports freigegeben werden müssen.

Replikation und Datenbank-Grids

Eine Transbase® Datenbank (*Master*) kann durch einen Replikationsprozess (*tbrepl master*) auf beliebig viele Slaves repliziert werden. Ein Slave erstellt mittels des Replikationsprozesses (*tbrepl slave*) zunächst eine Kopie der Master Datenbank und empfängt anschließend kontinuierlich deren Änderungen. Diese Änderungen werden zum frühestmöglichen Zeitpunkt auf dem Slave angewendet, insbesondere nach jedem COMMIT des Masters. Damit kann jeder Slave als Hot-Standby die Masterrolle übernehmen, falls der Master ausfallen sollte. Zusätzlich kann jeder Slave als Datenbank-Server für lesende Anfragen betrieben werden. Damit lässt sich eine dynamische Lastverteilung realisieren.

Die dynamische Lastverteilung wird durch ein sogenanntes Datenbank-Grid realisiert, das Anfragen für eine (logische) Datenbank auf ein Grid von physischen Datenbanken verteilt.

Replikation und Grid werden typischerweise kombiniert: alle schreibenden Anfragen werden direkt zum Master geleitet, alle lesenden Anfragen zu einem Grid, in dem der Master und alle Slaves zusammengefasst sind.

SQL Kern



WITH Klausel

Mit der WITH Klausel können Mehrfachberechnungen innerhalb einer SQL Anfrage vermieden werden. Die allgemeine Form lautet:

```
WITH  a AS (SELECT ...),  
      b AS (SELECT ...),  
      ...  
SELECT ... FROM a, b, <tables>  
...
```

Die in der WITH Klausel definierten Bezeichner dienen als Bezeichner für (temporäre) Zwischenergebnisse und können in der nachfolgenden SQL Anfrage wie persistente Tabellen verwendet werden. Die Definition von b kann sich dabei bereits auf a abstützen usw.

Eine typische Verwendung der WITH Klausel ergibt sich für Reports, die zunächst Detailsätze anzeigen und anschließend deren Summe. Eine solche Anfrage würde sich dann folgendermaßen schreiben lassen:

```
WITH detail AS (SELECT ...)  
SELECT 0, * FROM detail  
UNION  
SELECT 1, 'TOTAL', SUM(A), AVG(B), ... FROM detail  
  
ORDER BY 1
```

Die WITH Klausel ist syntaktisch identisch mit der bei Oracle verfügbaren WITH Klausel.

Parallele Queryverarbeitung

Mit der Verbreitung von Multi-Core-CPU's bis hin in Notebooks kommt der gezielten Ausnutzung solcher Ressourcen eine steigende Bedeutung zu. Während bei vielen Prozessen schon das Betriebssystem für die Ausnutzung der CPU's sorgt, ist es im Fall geringer Prozessparallelität Aufgabe des einzelnen Prozesses, mehrere CPU's gleichzeitig auszunutzen. Dies geschieht typischerweise durch Zerlegen des Prozesses in mehrere Threads, die dann parallel an einer einzelnen Query arbeiten können.

Bei der Zerlegung in Threads gibt es zwei grundverschiedene Ansätze:

- Zerlegung in verschiedenartige Threads, z.B. IO-Threads, Restriktions-Threads, Sortier-Threads, etc.; hierbei sind der Parallelität gewisse Grenzen gesetzt, die sich durch die verschiedenen Funktionen der Threads ergeben.
- Zerlegung in mehrere gleichartige Threads; hier kann die Parallelität fast beliebig verändert werden, soweit eine gewisse Datenparallelität gegeben ist.



Transbase® verwendet sowohl die erstgenannte, vor allem aber die letztgenannte Art der Parallelisierung. Dazu gibt es einen speziellen Operatorbaumknoten „ASYNC“, der sämtliche Aspekte der Parallelverarbeitung in sich konzentriert. Dieser Knoten stellt einerseits die Tupelpuffer bereit, in die darunterliegende Knoten ihre Tupel ablegen und aus denen darüberliegende Knoten ihre Tupel abholen. Andererseits kreiert und beendet dieser Knoten die Threads und synchronisiert sie in Bezug auf die Pufferzugriffe. Wie viele Threads ein solcher Knoten generiert, hängt dabei dynamisch von der Verarbeitungsgeschwindigkeit der Threads ab: Ergibt sich an einer Stelle im Operatorbaum ein „Stau“, so werden weitere Threads kreiert, um den Stau abzubauen. Auf diese Weise ergibt sich ein dynamisches Gleichgewicht der Threads mit der Folge, dass sie weder wegen voller noch wegen leerer Puffer zu oft warten müssen.

Die Parallelisierung wird im folgenden Operatorbaum beispielhaft skizziert:

Für die Version 6.9 wurde die Parallelverarbeitung weiter ausgebaut: sie erstreckt sich jetzt insbesondere auch auf die IO-relevanten Phasen der Queryverarbeitung. Dadurch wird automatisch auch der IO-Durchsatz deutlich

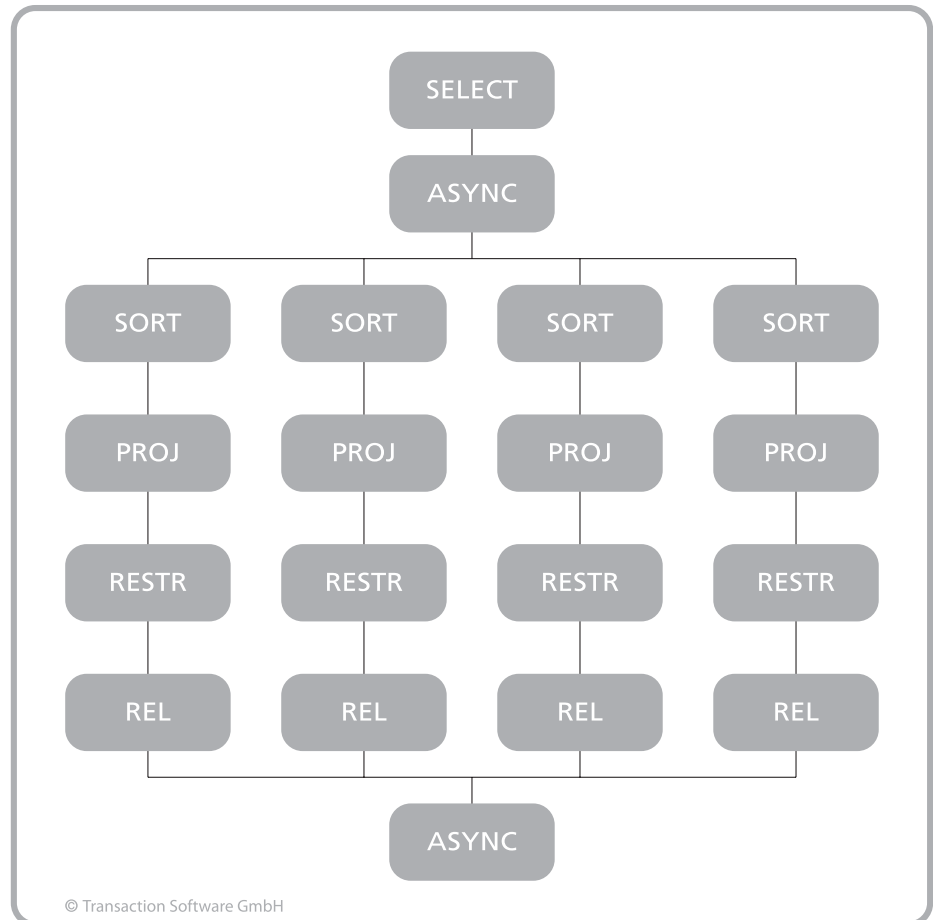


Abbildung 1:
Dynamisches Multithreading



gesteigert, so dass große RAID-Systeme optimal ausgelastet werden können. IO-relevant sind insbesondere die B-Baum-Algorithmen für den Blattzugriff (Sequential Scan und Intervallzugriff), die Hypercube-Algorithmen (multidimensionaler Zugriff) sowie die Sortieralgorithmen (die je nach Sortiervolumen eine hohe IO-Last erzeugen können).

Cache Partitionierung

Der Zugriff auf den Transbase® Cache muss zwischen Threads und Prozessen synchronisiert werden. Dazu gibt es Datenstrukturen, die nur exklusiv betreten werden dürfen, d.h. von höchstens einem Thread eines Transbase® Prozesses. Da die parallele Queryverarbeitung durch viele Threads in einem Prozess erreicht wird, ergibt sich daraus ein mit wachsender Threadanzahl immer deutlicherer Engpass. Wenn z.B. zum Fixieren und zum Freigeben einer Datenbankseite im Cache exklusiver Code von jeweils 10 µsec durchlaufen werden muss, ergibt sich automatisch eine Beschränkung von 50.000 Seitenoperationen je Sekunde für das Gesamtsystem, völlig unabhängig wie lange ein einzelner Thread zum Behandeln einer Seite benötigt.

Um diesen Engpass zu entschärfen, kann die Länge der exklusiven Codesequenzen verkürzt werden; dies würde den Engpass jedoch nicht qualitativ verändern. Daher wurde in Version 6.9 eine Partitionierung des globalen Caches vorgenommen, die das gleichzeitige Betreten von Cachepartitionen ermöglicht und so den Engpass praktisch gänzlich beseitigt. Derzeit können bis zu 128 Partitionen konfiguriert werden, so dass damit – mit den obigen Annahmen – bis zu 6.400.000 Seitenoperationen je Sekunde möglich werden. Jede Cachepartition ist in ihrer Größe auf maximal 1 GByte beschränkt; so ergibt sich eine maximale Gesamtgröße von 128 GByte für den Transbase® Cache.

CLOBs

Ein neuer Spaltentyp CLOB ist verfügbar, der sich zur Speicherung großer Textstücke eignet und praktisch eine unbeschränkte Ausweitung des Datentyps VARCHAR(*) darstellt (welcher ja in der Länge begrenzt ist, entweder auf die maximale Stringlänge oder auf die maximale Seitengröße).

Implementiert ist ein CLOB wie ein BLOB, allerdings kann er nur CHARACTER Daten aufnehmen. Bei der Speicherung von CLOBs wird deren Inhalt in die Codepage der Datenbank konvertiert, bei der Abholung wird er in die vom Client gewünschte Codepage konvertiert.

Eine CLOB Spalte kann mittels Volltextindex indiziert werden.

Kontakt

Transaction Software GmbH
Willy-Brandt-Allee 2
81829 München

Tel.: +49 89 / 627 09 - 0
Fax: +49 89 / 627 09 - 11

info@transaction.de
www.transaction.de
www.transbase.de