Inhaltsverzeichnis

Was sind Tablelocks?

Tablelocks = Tabellensperren…

…sind erst einmal nichts Böses. Jede Datenbank bedient sich ihrer, um die Integrität von Daten sicherzustellen.

Ein Beispiel: Anwender A hat einen Debitorenstammsatz auf dem Bildschirm und ändert nun die Straße. Dieser Vorgang wird nun in der Datenbank zurückgespeichert, sodass die neue Straße fest gespeichert ist. Anwender B bearbeitet gerade den gleichen Datensatz und will aber die Telefonnumer verändern. Eigentlich können beide Aktionen unabhängig voneinander durchgeführt werden.

Da aber die Änderung von Anwender A immer den ganzen Datensatz zurückschreibt und auch die Telefonnummeränderung von Anwender B den ganzen Debitorensatz (inkl. Name, Adresse, Straße, neue Telefonnummer, Kreditlimit…) zurückschreibt, kann in diesem Fall Folgendes passieren: Anwender A und Anwender B bekommen den gleichen Satz mit der alten Straße und der alten Telefonnummer auf dem Bildschirm angezeigt.

Anwender A verändert nun nur die Straße (**keine Telefonnummer**), und speichert diesen Satz. Anwender B hat davon nichts mitbekommen. Er hat noch den unveränderten Datensatz mit der **alten Straße** auf dem Bildschirm und ändert nun die **Telefonnummer**. Nun speichert auch er seinen Satz zurück.

Was ist die Folge? Seine Änderung überträgt nun die neue Rufnummer…. **zusammen mit der alten Straße** an die Datenbank. Die Korrektur von A ist verloren.

Wie kann man so eine alltägliche Situation verhindern? Man arbeitet mit Tablelocks: Anwender A bekommt den Datensatz auf den Bildschirm. In diesem Augenblick sperrt Navision / Business Central den Datensatz für Veränderungen. Er wird „gelockt“. Nicht zu verwechseln mit einem Zugriffsprotokoll, dann wird er „geloggt“ (von dem Begriff „**Log**buch“).

Anwender B kann nun erst gar nicht diesen Datensatz anzeigen. Erst wenn A fertig ist, z.B. mit dem Anschauen oder der Straßenänderung, bekommt B den nun wieder freigegeben Satz auf den Bildschirm. Nun geht dieses Spiel bei ihm wieder von Vorne los.

Wodurch entstehen Tablelocks?

„Lesen“ ist in Navision / Business Central NIE ein Problem! Erst beim Schreiben geht der Ärger los. Wobei Löschen, Einfügen und Verändern alle unter „Schreiben“ (=Verändern“) fallen.

Zum Glück ist Navision / Business Central deutlich smarter als viele Datenbanksysteme, die das tatsächlich so handeln wie oben beschrieben. Navision benutzt Optimistic Concurrency: In den allermeisten Fällen wird ein Datensatz nur angezeigt, und nicht verändert.

Und wenn doch mal eine Adresse geändert wird? Statistisch ist es sehr unwahrscheinlich, dass auf einem anderen Computer der **gleiche Datensatz** zur gleichen Zeit verändert wird. Daher lockt Navision in keinem Fall diesen Datensatz, sondern kontrolliert beim Speichern über den Zeitstempel, ob der Datensatz in der Datenbank noch die gleiche Version ist wie der gerade zum Ändern zurückzuspeicherende Datensatz.

Dies wird über das unsichtbar mitgeführte Feld Timestamp in jedem Record einer Navision / Business Central Datenbank getan. Daher kann man auch selbst kein Feld anlegen, welches TimeStamp heißt. „Früher“ ging das noch, ein beliebter „Graue Haare“-Fehler beim Übertragen einer nativen Datenbank auf SQL.

Wenn nun Navision gar keine Locks macht, woher dann das Problem? Obiges Beispiel ist eines von hunderten, bei dem die Entwickler von Navision dem Anwender solche Arbeit abgenommen habe. Es gibt aber auch echte Fälle, wo die Datenbank sicher sein muss, dass sie sich in einem konsistenten Zustand befindet.

Schreiben tut weh…

Dafür Ein neues Beispiel: ein Anwender verbucht einen Auftrag. Das kann schon mal 1-2 Sekunden dauern. In dieser Zeit werden Sachposten geschrieben, gebuchte Lieferscheine und gebuchte Rechnungen, offene Debitorenposten…

In der gleichen Zeit könnte nun ein anderer Anwender diesen Auftrag oder gar diesen Debitoren löschen wollen. Wenn dies erlaubt wäre, so würden am Ende Lieferscheine, offene Posten und Rechnungen zu einem Debitoren existieren, der selbst zu dieser Zeit gar nicht mehr existiert. Je nach zeitlicher Verschiebung hätte es auf dem Debitoren ja noch gar keine Posten gegeben, die ein Löschen verboten hätten!

Navision / Business Central betreibt hier noch viel mehr Aufwand, um soetwas sicher zu vermeiden, aber für unser Beispiel reicht es.
Navision würde nun also beim Verbuchen den Auftrag und den Debitoren sperren („locken“), um genau diesen Zustand zu verhindern. In Echt wird dies anders gemacht, aber das ist hier nicht so wichtig.
Nun gibt es also für eine kurze Zeit einige Tabellen, die gesperrt („zum Schreiben geöffnet“) sind: Aufträge, Auftragszeilen, gebuchte Lieferscheinköpfe, gebuchte Lieferscheinzeilen… und noch ein paar mehr.

Wenn nicht so viele Menschen an dem System arbeiten, ist das auch gar kein Problem: Schließlich sind diese Tabellen schon ein oder zwei Sekunden später wieder frei. (Nicht bei Ihnen? Bei Ihnen braucht ein Auftrag viele Sekunden oder gar Minuten zum Verbuchen? Das ist Schlecht! Das lässt sich sicher beschleunigen!)

Also immer noch kein Problem? Genau. Außer… solche Transaktionen dauern zu lange, oder sperren Tabellen in unterschiedlichen Reihenfolgen. Genau dadurch treten Deadlocks auf.

Deadlocks… der Blick auf das Höllentor

Deadlocks: Angenommen, ein Programm will Debitoren sperren, um danach Aufträge dafür einzutragen. Erst sperrt NAVISION / BUSINESS CENTRAL die Debitoren, dann… ist einen Augenblick Verzögerung. In dieser Zeit startet ein anderes Programm, welches die Aufträge sperrt und bei Bedarf Datenänderungen im Debitoren vornimmt.

Nun kommt das erste Programm an die Stelle, wo es zu seinen Debitoren auch noch die Aufträge sperren will. Das geht nun nicht, weil das **zweite** Programm bereits die Aufträge für sich gesperrt hat. Also wartet das **erste** Programm einfach ganz brav darauf, dass es

auch die Aufträge sperren kann. Es wartet wirklich! Das zweite Programm kommt nun an die Stelle, an der es gerne die Debitoren verändern, also auch sperren möchte. Es fordert die Debitorensperrung von der Datenbank an… doch die sind ja schon gesperrt!

Nun gibt es einen Deadlock: Beide Programme warten auf die jeweils andere, von dem anderen Programm gesperrte Tabelle. Die Sperrung ist -im Prinzip- unendlich, da ja beide Programme jeweils auf die Freigabe der anderen Tabelle aus dem ersten Programm warten. Diese Situation bezeichnet man als „Deadlock“, tote Sperrung: Sie wird – von sich aus- niemals aufgelöst.

Dies erkennt der SQL-Server, und er beendet eine der beiden Abfragen, so dass die andere nun weiter arbeiten kann.

Faustformel: Je schlechter der Programmcode, desto länger die Programmverarbeitung und desto wahrscheinlicher sind Deadlocks. Weil einfach die Schreibsperren zu lange aktiv sind. Performance-Optimierung ist demnach auch immer Deadlock-Vermeidung!

Finden von Tablelocks/Deadlocks

In Navision / Business Central wird so ein aufgelöster Deadlock nur mit einer einfachen Fehlermeldung angezeigt:

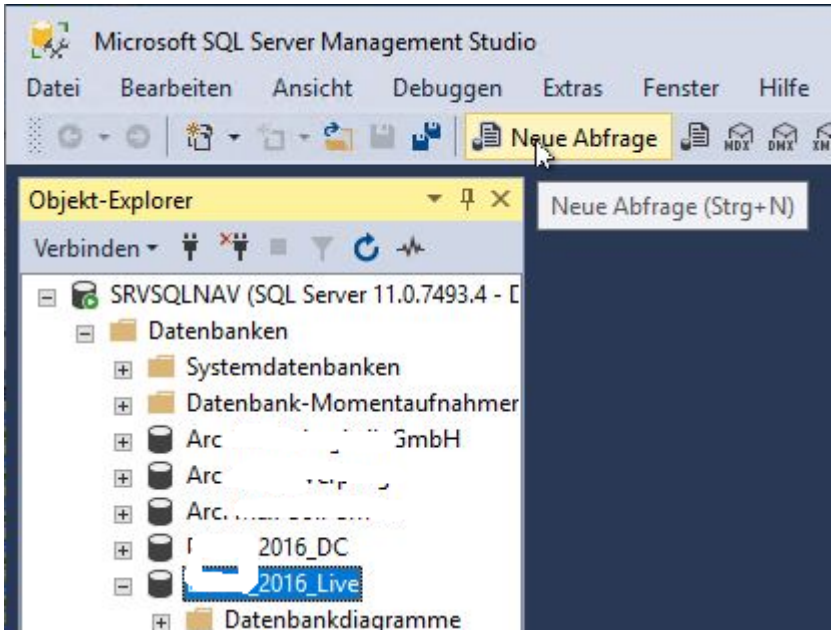
„Der Vorgang konnte nicht abgeschlossen werden, da ein Datensatz in der Tabelle „xxx“ durch einen anderen Benutzer gesperrt wurde“ angezeigt. Führen Sie die Aktion erneut aus. Diese Meldung erscheint immer auf dem Computer, der den Deadlock **verloren** hat.

Hin und wieder sind Deadlocks aber auch über mehrere Programme verteilt, sodass selbst der (SQL-)Server selbst nicht mehr feststellen kann, dass es sich um einen Deadlock handelt. Hier greift dann z.B. die maximale SQL Ausführungszeit.

Bis Navision 2009R2 konnte man noch einfach in der Session-Tabelle nachsehen, welcher Prozess den jeweils anderen blockiert. Durch einen flinken Blick auf den richtigen Bildschirm konnte man das Problem möglichweise schnell lokalisieren… oder seine ausgedehnte Suche starten.

Seit Navision / Business Central 2013 RTC greift aber die Microsoft-Politik, alles nach Möglichkeit komplizierter und schwieriger zu machen. Hier muss man sich dann direkt auf dem SQL-Server austoben. Hier ein paar Methoden zum Analysieren:

Es geht immer im SQL-Server in der aktuellen Datenbank mit einer Abfrage los



Auflisten von blockierenden Usern

exec sp_who2

SQL listet alle Sessions auf. Eine blockierte Session zeigt in der Spalte „Blkby“ an, ob/von welcher Session sie gesperrt wird.
I.d.R. wird man hier die gleiche Session-ID bei mehreren Sessions finden, die alle auf die gleiche blockierende Session warten.

Datenbanklast

Da Deadlocks praktisch immer Hand in Hand mit zu langen Programmlaufzeiten einhergehen, und diese fast immer durch unüberlegte Datenbankabfragen entstehen, ist ein Blick auf die Datenbanklast auch immer ganz praktisch:

select
spid,cmd,waittime,lastwaittype,cpu,physical_io,login_time,last_batch,status,hostname,program_name,nt_username, nt_domain

from master.dbo.sysprocesses where dbid = db_id(‚GewünschteDatenbank‘)

oder

```
SELECT DISTINCT
name AS database_name,
session_id,
host_name,
login_time,
login_name,
reads,
writes
FROM sys.dm_exec_sessions
LEFT OUTER JOIN sys.dm_tran_locks ON sys.dm_exec_sessions.session_id =
sys.dm_tran_locks.request_session_id
INNER JOIN sys.databases ON sys.dm_tran_locks.resource_database_id =
sys.databases.database_id
WHERE resource_type <> ‚DATABASE‘
–AND request_mode LIKE ‚%X%‘
–AND name =’DieInteressanteDatenbank‘
ORDER BY name
```

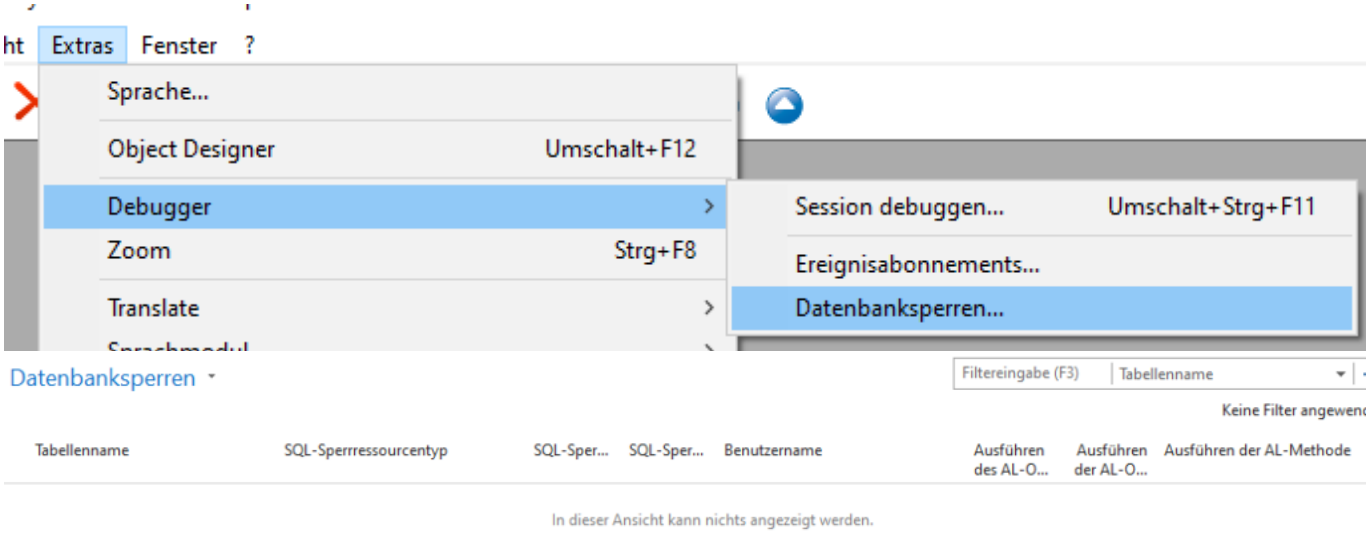
Während reads und writes im 5-Stelligen Bereich meist keine Aufmerksamkeit wert sind, sollte man bei 6-stelligen schon mal nachsehen, und bei 7-stelligen Zugriffen sollten die Alarmglocken klingeln… natürlich auch in Abhängigkeit der bisherigen Laufzeit des jeweiligen Prozesses.

1,5 Mio Schreibzugriffe für einen EDIfact Connector, der seit 12 Tagen online ist, ist eine andere Zahl als 500.000 Lesezugriffe bei einem User, der sich vor 5 Minuten angemeldet hat. Performance Troubleshooting braucht viel Erfahrung und Geduld.

In Navision / Business Central Debugger ab 2017

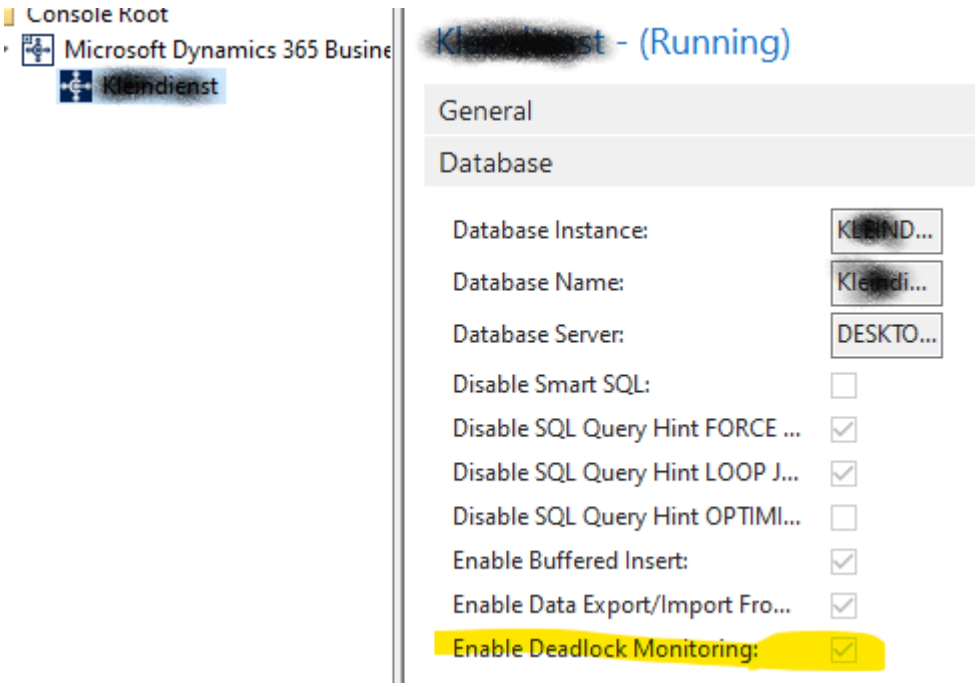
Hier wurde das Locking Monitoring bereits in den Debugger mit eingebaut! Navision / Business Central benötigt hierfür aber Änderungen im SQL-Server, damit dieser auch die notwendigen Daten herausrückt.

Wenn Navision / Business Central selbst den Server installiert hat, sind diese Einstellungen bereits vorgenommen!

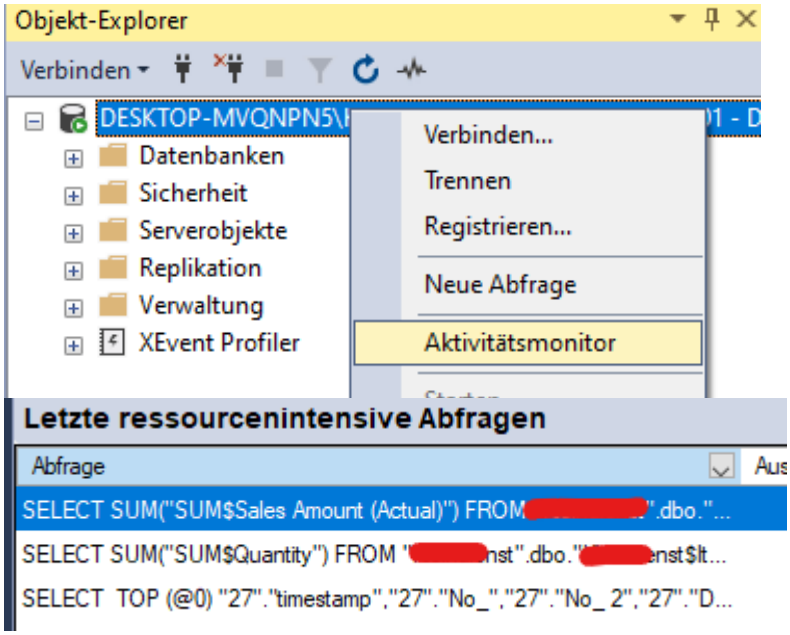


Vornehmen der Anpassungen im SQL_Server, wenn noch nicht erfolgt:
Im MSSMS Server/Eigenschaft -> Berechtigungen, für die Anmeldung unter der der Navisiondienst läuft:
Beliebige Ereignissitzung ändern: Erteilen.
Serverstatus anzeigen: Erteilen.

Diesen Benutzer über Server/Sicherheit/Anmeldung der gewünschten Datenbank als Owner zuordnen.
In der Navision / Business Central Einstellung das Deadlock Monitoring aktivieren:



Für Performance Trouble Shooting kann auch der Aktivitätsmonitor hilfreich sein. Er ist über das MSSMS erreichbar:



[Hier gibt es noch weitere Tools aus dem Aktivitätenmonitor]