

# Deadlock-Fehler

Bei einem Schreibbefehl auf einen Datenbank-Transaktion erhält die Datenbank-Zeile einen Deadlock. Der Deadlock wird nach Abschluss der Transaktion wieder aufgehoben. Wird während des Deadlocks ein weiterer Schreibbefehl auf den Datenbank-Eintrag ausgeführt, wird ein Deadlock-Fehler ausgegeben. Der Fehler entsteht somit auf der Datenbank.

Die umgesetzten Lösungen können lediglich versuchen, Schreibbefehle auf die Datenbank zu minimieren und ein entsprechendes Fehler-Handling anzubieten. Die Gentable-Funktion "**gentable.insertIntoDB()**" schreibt zum Beispiel die Rechnungs-Positionen in die Tabelle "**invoice\_posting\_pos**". Die Funktion wird bei jedem Speichern einmalig ausgeführt. Alle Rechnungs-Positionen aller Rechnungen werden in diese Datenbank-Tabelle geschrieben. Die Tabelle wächst demnach immer weiter und bei Lösch- und Update- Befehlen wird es immer aufwendiger, den gewünschten Datensatz zu ermitteln. Entsprechend ist es wichtig, Wartungsoperationen bei einer SQL Datenbank einzuplanen!

Das Schreiben in die Datenbank kann über den Parameter "**GentableSuppressWritingDataIntoDb**" unterdrückt werden. Bei Rechnungen mit Bestellbezug werden die korrekten Positionsdaten in der Tabelle allerdings zwingend für die 3-Way-Match-Prüfungen benötigt. Entsprechend ist das Unterdrücken der Schreibbefehle keine generelle Lösung.

Die folgende Beispiel-Fehlerausgabe stammt von einem MS SQL-Server. Als Lösung schlägt der SQL-Server vor, die Transaktion erneut auszuführen.

```
ReturnCode: -1
SqlState: 40001
NativeError: 1205
ErrorMsg: [Microsoft][ODBC SQL Server Driver][SQL Server]Transaction (Process ID 92) was
deadlocked on lock | communication buffer resources with another process and has been chosen
as the deadlock victim. Rerun the transaction.
.. this.Result:=false
```

Ab der Invoice-Version 2.0.200 wird dieser Vorschlag befolgt. Bei einem Schreibfehler in die Datenbank wird ein `util.sleep()` ausgeführt und der Schreibbefehl wird wiederholt.

---

Revision #3

Created 2 July 2024 10:06:36 by Markus Meisner

Updated 2 July 2024 10:50:20 by Markus Meisner