

DbiUebung-001 -- Oracle (not Entity) Framework

Todo:

Führen Sie die Tests mit `numberOfParallelThreads=1` aus. Verhält sich der Code wie erwartet?

Analysieren Sie den gegebenen Code auf Probleme. Wie manifestieren sich die Probleme bei den beiden Tests?

Beheben Sie die Nebenläufigkeitsprobleme mittels den gegebenen Locking-Arten und behandeln Sie jene Fehler, die dabei auftreten können.

- Zeilen-Locks
- Table-Lock
- Optimistisches Locking

- Bei nur einem Thread verhält sich der code wie erwartet, jedes query wird nacheinander abgearbeitet, bei mehreren parallelen Threads können jetzt bereits Probleme auftreten.

Ein Zeilen-Lock scheint deadlocks nicht zu verhindern und selbst mit dem lock-object im c# code, treten ab und zu deadlocks auf (je nach cpu auslastung)

- Ich fügte, wie bereits genannt "FOR UPDATE":

```
// Query with one resulting line
cmd.CommandText = "SELECT balance FROM konto WHERE kid=" + source + " FOR UPDATE"; // Line-Lock - for Update
```

- sowie eine Lock-variable `_locker` zum code hinzu:

```
#region FIELDS
private static readonly object _locker = new(); // lock-object to ensure save code execution
```

- Als letzte Maßnahme erweiterte ich den code um einen vollständigen Table-lock, welcher die deadlocks letztendlich verhindert:

```
cmd.CommandText = "LOCK TABLE konto IN EXCLUSIVE MODE"; // Table lock - to prevent deadlocks
cmd.ExecuteNonQuery(); // while running multiple threads parallel
```

(auch bei vielen parallelen Threads, stimmen die Transaktionen und blockieren sich nun nichtmehr)

- OPTIMISTIC "Locking":

```
// Optimistic "locking": Number of changed lines == 0
int modifiedRows = cmd.ExecuteNonQuery();
if (modifiedRows == 0)
{
    cmd.CommandText = "UPDATE konto SET balance = balance + " + amount + " WHERE kid=" + dest;
    cmd.ExecuteNonQuery();
}
```

- PESIMISTIC Locking:

```
// Pesimistic locking: Lock the whole Table
cmd.CommandText = "LOCK TABLE konto IN EXCLUSIVE MODE"; // Table lock - to prevent deadlocks
cmd.ExecuteNonQuery(); // while running multiple threads parallel
```

Program ran through, with 10 parallel threads:

The screenshot shows two windows. The left window is a database query result titled 'Arbeitsblatt' with a 'Query Builder' tab. It displays the SQL query 'SELECT * FROM KONTO;' and the results of the query. The results are as follows:

KID	KNAME	BALANCE
1	-1 BANK	100000
2	0 KING	490000
3	1 SCOTT	200000
4	2 ANTON	200000
5	3 HERBERT	10000

The right window is a terminal window titled 'D:\Repositories\DbiUr'. It shows the output of a program that transfers money 1000 times. The output is as follows:

```
Transferred Money, times: 998
Transferred Money, times: 992
Transferred Money, times: 993
Transferred Money, times: 997
Transferred Money, times: 999
Transferred Money, times: 993
Transferred Money, times: 998
Transferred Money, times: 994
Transferred Money, times: 994
Transferred Money, times: 999
Transferred Money, times: 995
Transferred Money, times: 995
Transferred Money, times: 996
Transferred Money, times: 996
Transferred Money, times: 997
Transferred Money, times: 997
Transferred Money, times: 998
Transferred Money, times: 998
Transferred Money, times: 999
Transferred Money, times: 999
All threads have finished.

-----Oracle Database Assignment - Locking-----

Enter the amount of Threads to use:
Which test would you like to run?
1 - Transfer Money from King to Herbert
2 - Transfer Money randomly
0 - Exit
```