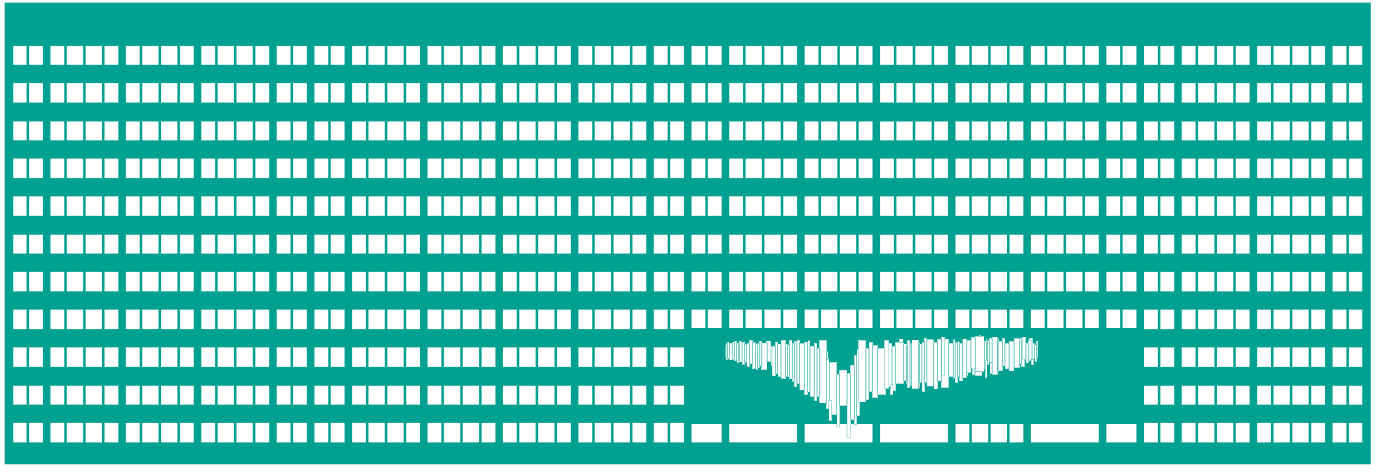
VŠB TECHNICKÁ UNIVERZITA OSTRAVA

VSB TECHNICAL UNIVERSITY OF OSTRAVA



www.vsb.cz

Database and Information Systems

db.cs@vsb.cz

Department of Computer Science
Faculty of Electrical Engineering and Computer Science
VSB - Technical University of Ostrava

2019/2020

Tasks

- Study anomalies of concurrency and transaction isolation levels from 8th lecture. Focus on the isolation levels of transactions in SQL Server.
- Open scripts `t1.sql` and `t2.sql` in Sql Server Management Studio (SSMS).
- Create table `Orders` following script `t1.sql`.
- Execute commands in `t1.sql` and `t2.sql` as instructed in the script `t1.sql`. Put the windows of both connections in SSMS side by side for the easier work with them and their comparison.
- Make sure you understand each step you perform. Execute commands in the specified order, do not skip any step! After each test, make sure that the table texttt orders still contains only 4 records.

Loss of update 1/2

- Let's have a classic reservation system, where users view available items (like seats in a movie theater) and then make a reservation.
  - The user `U1` shows the available seats (`S1` and `S2`) and think about the best place to sit.
  - Meanwhile, the user `U2` logs in and reserves the seat `S1`.
  - After that, the user `U1` also chooses the seat `S1` and makes a reservation.
  - The seat `S1` is reserved twice, or the reservation of the user `U1` is overwritten (depending on the scheme of the reservation DB).

Loss of update 2/2

- How to properly solve this situation in the database system? Can the isolation levels help us with this problem? Try to analyze which features might have different solutions depending on whether the reservation is made with `INSERT` to the `Reservation` table or as `UPDATE` to `Seat(user:int)`
  - Using the isolation level of SERIALIZABLE and starting the transaction while viewing the available locations, the user U2 would never get to the reservation. It is therefore not appropriate. Is it possible to solve it with the SNAPSHOT isolation level?
  - If the reservation is made with UPDATE, then it is a good solution to create a transaction that will check if the location is available (i.e. the user attribute is not set yet) before inserting the reservation.
  - If the reservation is made with INSERT, then a simple solution can be to create a suitable primary key on the table to prevent the user U2 from creating an order.