Lab 2 JTA – Java Transaction API

- Objective: familiarise with the very basic programming for distributed transactions.
- Task: implement and test a distributed transaction application, e.g. *fund transfer*, using Java Transaction API (JTA).
  - Make sure the principles for distributed transactions are well understood.
  - The program is in fact a toy transaction manager.

Two Phase Commit Protocal (2PC)

- The site at which the transaction originated is the coordinator. Other sites at which sub transactions are executed are subordinates.
- When a user decides to commit a transaction, the commit command is sent to the coordinator for the transaction.
- This initiates the 2PC.

Two Phase Commit Protocal (2PC) – cont'd

- Phase 1: Coordinator sends a prepare message to each subordinate; When subordinate receives a prepare message, it decides to abort or commit its sub transaction; Subordinate forces writes an abort or ready log record and sends a no or yes message to coordinator accordingly.
- Phase 2: If coordinator receives a yes message from all subordinates, force-writes a commit log record and sends commit message to all subordinates. Else force-writes an abort log record and sends an abort message; Subordinates force-write abort or commit log record based on the message they receive.
- In some implementations, after the two phases, subordinates send acknowledgement message to coordinator; After coordinator receives ack messages from all subordinates it writes an end log for the transaction.

Preparation

- JDBC is needed.
  - SAT MySQL: (IP: *10.7.1.127*) http://csse-mysql.xjtlu.edu.cn
  - *You should have had your account (already created by the admin)*
  - JDBC connector:https://dev.mysql.com/downloads/connector/j/
- Create 'distributed' databases on MySQL
  - two relations/tables in your own database, or
  - two relations/tables in different databases

Preparation – cont'd

- Two relations/tables in your own database
  - Not a true distributed database
  - This can be done by yourself
- Two relations/tables in different databases
  - Distributed database
  - Need two people work in collaboration (share each others' database relations)
- Basic relation/table design
  - A table with two columns such as user account (integer) and balance (double)
  - Add more columns if needed
  - Add some sample data for testing

Preparation – cont'd

- Download the file from LM Core, filename: *jta.FundTransferJTA.java*
- Source code
  - Fund transfer from one account at one database to another at the same or different database.
  - *Note: you need to change the source code to include your own database information*, e.g.
    - database URL, port, username, password, etc
    - user account ID
    - Add code to print the amount before and after the transaction, or view the result in the relations.

Task: Fund Transfer

- Run code in the terminal from the lab2 folder
  - Windows:
    - javac -cp mysql-connector-j-8.0.31.jar; FundTransferJTA.java
    - java -cp mysql-connector-j-8.0.31.jar; FundTransferJTA
  - MacOS:
    - javac -cp mysql-connector-j-8.0.31.jar: FundTransferJTA.java
    - java -cp mysql-connector-j-8.0.31.jar:. FundTransferJTA
- Record the output