Distributed transactions in relational databases

DANIELE APILETTI
POLITECNICO DI TORINO
ACID properties

• Atomicity
  o It requires distributed techniques
    ▪ 2 phase commit

• Consistency
  o Constraints are currently enforced only locally

• Isolation
  o It requires strict 2PL and 2 Phase Commit

• Durability
  o It requires the extension of local procedures to manage atomicity in presence of failure
Other issues

• *Distributed query optimization* is performed by the DBMS receiving the query execution request
  - It partitions the query in subqueries, each addressed to a single DBMS
  - It selects the execution strategy
    - order of operations and execution technique
    - order of operations on different nodes
      - transmission cost may become relevant
    - (optionally) selection of the appropriate replica
  - It coordinates operations on different nodes and information exchange
Atomicity

• All nodes (i.e., DBMS servers) participating to a distributed transaction must implement the same decision (commit or rollback)
  o Coordinated by 2 phase commit protocol

• Failure causes
  o Node failure
  o Network failure which causes lost messages
    ▪ Acknowledgement of messages (ack)
    ▪ Usage of timeout
  o Network partitioning in separate subnetworks
2 Phase Commit protocol

• Objective
  o Coordination of the conclusion of a distributed transaction

• Parallel with a wedding
  o Priest celebrating the wedding
    ▪ Coordinates the agreement
  o Couple to be married
    ▪ Participate to the agreement
2 Phase Commit protocol

• Distributed transaction
  o One coordinator
    ▪ Transaction Manager (TM)
  o Several DBMS servers which take part to the transaction
    ▪ Resource Managers (RM)

• Any participant may take the role of TM
  o Also the client requesting the transaction execution
New log records

- TM and RM have *separate private* logs
- Records in the TM log
  - *Prepare*
    - it contains the identity of all RMs participating to the transaction (Node ID + Process ID)
  - *Global commit/abort*
    - final decision on the transaction outcome
  - *Complete*
    - written at the end of the protocol
New log records

• New records in the RM log
  o *Ready*
    ▪ The RM is willing to perform commit of the transaction
    ▪ The decision *cannot be changed* afterwards
    ▪ The node has to be in a reliable state
      ▫ WAL and commit precedence rules are enforced
      ▫ Resources are locked
    ▪ After this point the RM *loses its autonomy* for the current transaction
2 Phase Commit protocol

RM

Prepare msg

TM

Prepare

LOG
Phase I

1. The TM
   - Writes the prepare record in the log
   - Sends the prepare message to all RM (participants)
   - Sets a timeout, defining maximum waiting time for RM answer
2 Phase Commit protocol
Phase I

• The RMs
  o Wait for the prepare message
  o When they receive it
    ▪ If they are in a reliable state
      ■ Write the ready record in the log
      ■ Send the ready message to the TM
    ▪ If they are not in a reliable state
      ■ Send a not ready message to the TM
      ■ Terminate the protocol
      ■ Perform local rollback
    ▪ If the RM crashed
      ■ No answer is sent
2 Phase Commit protocol
3. The TM

- Collects all incoming messages from the RMs
- If it receives ready from all RMs
  - The commit global decision record is written in the log
- If it receives one or more not ready or the timeout expires
  - The abort global decision record is written in the log
Phase II

1. The TM
   - Sends the global decision to the RMs
   - Sets a timeout for the RM answers
2 Phase Commit protocol

- **Prepare msg**
- **Ready/not ready msg**
- **Global decision**
- **Ack msg**
Phase II

2. The RM
   o Waits for the global decision
   o When it receives it
     ▪ The commit/abort record is written in the log
     ▪ The database is updated
     ▪ An ACK message is sent to the TM
2 Phase Commit protocol

Diagram:
- RM (Prepare msg) to TM (Prepare LOG)
- RM (Ready/not ready msg) to TM (Global Commit/Abort)
- RM (Global decision) to TM (Ack msg)
- RM (Complete) to TM (LOG)

Symbols:
- RM: redo manager
- TM: two-phase commit manager
- DB: database
- LOG: log

Actions:
- Prepare
- Ready/not ready
- Global decision
- Ack
- Complete
3. The TM

- Collects the ACK messages from the RMs
- If *all* ACK messages are received
  - The complete record is written in the log
- If the timeout expires and some ACK messages are missing
  - A new timeout is set
  - The global decision is resent to the RMs which did not answer until all answers are received
2 Phase Commit protocol
Uncertainty window

• Each RM is affected by an *uncertainty window*
  o Start after ready msg is sent
  o End upon receipt of global decision

• Local resources in the RM are locked during the uncertainty window
  o It should be small
Failure of a participant (RM)

• The warm restart procedure is modified with a new case
  o If the last record in the log for transaction T is “ready”, then T does not know the global decision of its TM

• Recovery
  o READY list
    ▪ new list collecting the IDs of all transactions in ready state
  o For all transactions in the ready list, the global decision is asked to the TM at restart
    ▪ Remote recovery request
Failure of the coordinator (TM)

• Messages that can be lost
  o Prepare (outgoing)
  o Ready (incoming)
  o Global decision (outgoing)

• Recovery
  o If the last record in the TM log is prepare
    ▪ The global abort decision is written in the log and sent to all participants
    ▪ Alternative: redo phase I (not implemented)
  o If the last record in the TM log is the global decision
    ▪ Repeat phase II
Network failures

• Any network problem in phase I causes global abort
  o The prepare or the ready msg are not received

• Any network problem in phase II causes the repetition of phase II
  o The global decision or the ACK are not received