

Relatório do Projeto - DDB Middleware

Disciplina: Laboratório de Sistemas Distribuídos

Integrantes:

Luiz Jordany de Sousa Silva
Ivan Luis Gama Grana
Syanne Karoline Moreira Tavares

Repositório:

https://github.com/oJordany/distributed_db

Visão Geral

O projeto implementa um middleware e nós de banco de dados distribuído. O cliente envia consultas SQL via TCP para o middleware, que descobre o coordenador (eleição Bully) e encaminha todas as requisições. Leituras são executadas pelo coordenador e escritas são replicadas usando 2PC (Two-Phase Commit).

Arquitetura

- Cliente: interface Tkinter para envio de SQL.
- Middleware: ponto de entrada que encaminha consultas para o coordenador.
- Coordenador: nó líder responsável por executar e replicar escritas.
- Nós participantes: executam PREPARE/COMMIT/ROLLBACK e mantêm o banco local.
- Eleição: algoritmo Bully para escolher o coordenador ativo.

Protocolo de Comunicação

A comunicação usa TCP (unicast) com mensagens em JSON contendo três campos: type, payload e checksum. O checksum é um MD5 do payload para verificar integridade.

Principais mensagens:

- ELECTION / ANSWER / COORDINATOR_VICTORY: eleição Bully.
- HEARTBEAT / ACK: detecção de falhas do coordenador.
- GET_COORDINATOR / COORDINATOR_INFO: descoberta do líder pelo middleware.
- EXECUTE_QUERY: leitura executada no coordenador.
- EXECUTE_2PC: escrita disparada no coordenador.
- PREPARE / COMMIT / ROLLBACK: fases do 2PC.
- TX_RESULT: resultado de uma escrita distribuída.

O middleware envia a consulta ao coordenador. Para escritas, o coordenador inicia o 2PC: envia PREPARE a todos os nós, coleta respostas e então envia COMMIT ou ROLLBACK.

Garantia das Propriedades ACID

- Atomicidade: 2PC garante que a escrita é aplicada em todos os nós ou em nenhum.
- Consistência: regras e constraints do banco local são preservadas; falhas abortam a transação.
- Isolamento: o controle é delegado ao MySQL via transações do SQLAlchemy.
- Durabilidade: após COMMIT, o MySQL persiste os dados em disco.

Tipo de Comunicação

A comunicação é unicast (TCP ponto a ponto). Algumas operações enviam mensagens para todos os nós, mas isso é feito por múltiplos unicasts, não por broadcast ou multicast na rede.

Execução

Para testes locais, o projeto fornece um docker-compose com três instâncias MySQL. Os nós usam portas 8001/8002/8003 (servidor do nó) e os bancos usam 3306/3307/3308.

Comandos principais:

- `docker compose up -d`
- `python main_node.py 1` (e repetir para 2 e 3)
- `python main_middleware.py`
- `python client/client_app.py`

Logs de Replicação

Cada nó registra o status das transações de escrita na tabela `replication_logs`, permitindo auditoria do processo de replicação.