Banco de dados replicados na edge
Uma abordagem híbrida

Ben Hur Faria Reis
Felipe Aguiar Costa
Filipe Silveira Chaves
João Paulo Rocha Celestino
Thiago Monteles De Sousa

INF
INSTITUTO DE INFORMÁTICA

UFG
UNIVERSIDADE FEDERAL DE GOIÁS

Cronograma

	Maio	Junho	Julho	Agosto
Revisão bibliográfica	X	X		
Apresentação interno de métodos de consistência	X	X		
Codificação		X	X	X

X concluído
X em processo

Contextualização

No contexto de replicação de bancos de dados, as abordagens de consistência forte ou eventual oferecem garantias diferentes, sendo que em alguns casos, uma ou outra é mais recomendada. Nesse sentido, o projeto pretende criar um banco de dados replicado de forma que duas abordagens de consistência estejam implementadas, e com base no uso, o usuário pode escolher qual prefere utilizar.

Aspectos do projeto em relação a Sistemas Distribuídos

INF
INSTITUTO DE INFORMÁTICA

Estilos arquiteturais:

Serão aplicados dois estilos arquiteturais de sistemas distribuídos: O estilo cliente-servidor e o peer-to-peer.

Cliente-servidor
Será utilizado para as chamadas de serviços externos aos nós do cluster do banco de dados.

Peer-to-peer
Será utilizada para a comunicação interna de replicação dos dados dentro do cluster.

Aspectos do projeto em relação a Sistemas Distribuídos

INF
INSTITUTO DE INFORMÁTICA

Paradigmas de comunicação:

Serão aplicados dois paradigmas de comunicação de sistemas distribuídos: O RPC e o request-reply com o protocolo HTTP.

RPC

Será utilizado para a comunicação interna entre os nós do cluster.

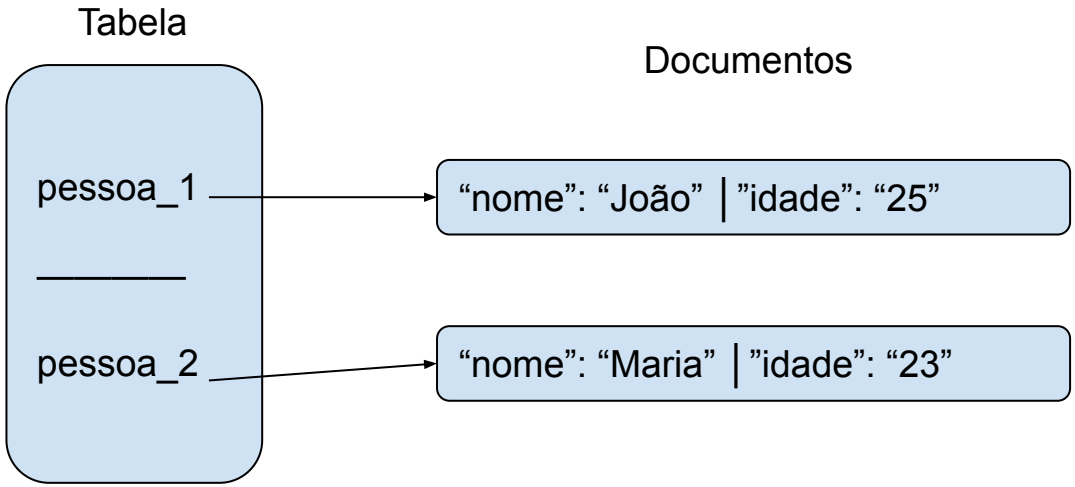
As chamadas nesse caso seriam para a sincronização de alterações nos dados.

Request-reply com o protocolo HTTP

Será utilizada para a comunicação externa entre os clientes e os nós do cluster

Essa comunicação seria para a manipulação externa dos dados, com o tratamento de replicação e consistência sendo feito de forma transparente.

Design dos dados



Tabelas: Cada tabela é uma coleção de documentos, cada documento tem um ID do tipo string, cada tabela pode ser configurada para ter ou consistência forte (RAFT) ou eventual (CRDT)
Documentos: Cada um guarda um mapa chave-valor, as chaves e os valores strings, as chaves podem ser arbitrárias, não tendo um esquema fixo para os dados.

Arquitetura

PUT /<table>

Cria uma tabela (indicando se será eventual ou forte)

DELETE /<table>

Remove uma tabela

GET /<table>/<doc id>

Retorna o mapa de chaves e valores de um documento

PUT /<table>/<doc id> (body: keys/values)

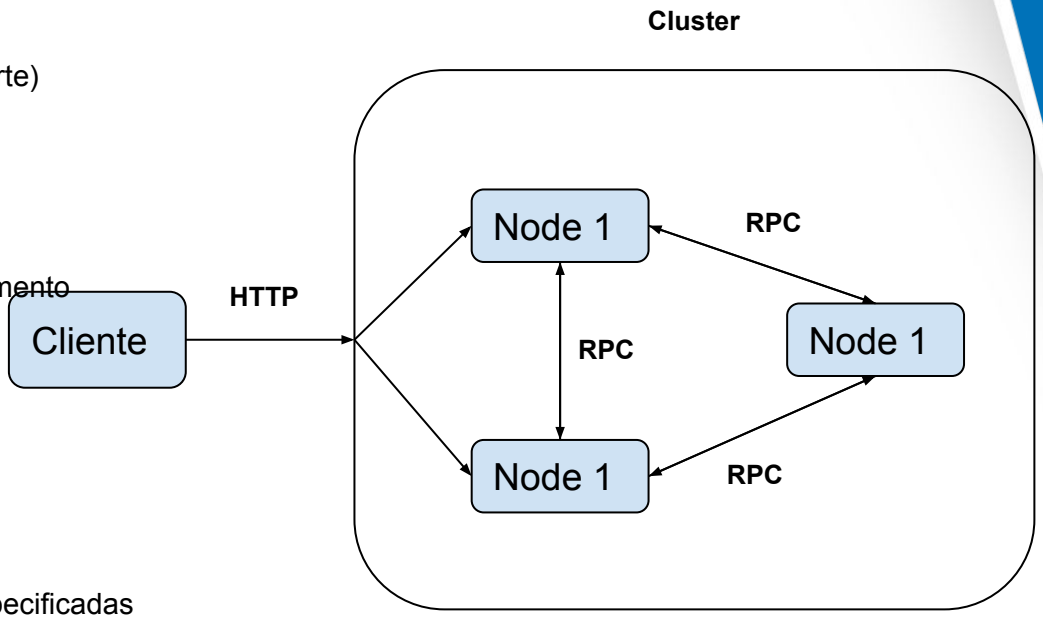
Substitui o documento inteiro, apagando outras chaves não especificadas nesta chamada

PATCH /<table>/<doc id> (body: keys/values)

Substitui valores dentro de um documento, mantendo chaves já existentes que não foram especificadas nesta chamada

DELETE /<table>/<doc id>

Remove um documento



Conflict-free Replicated Data Types (CRDT)

Tipos de dados replicados sem conflitos

Propriedades:

- **Livre de conflitos:** operações de atualização de dados são realizadas independentemente em diferentes réplicas, sem conflitos.
- **Comutatividade:** a ordem de aplicação das operações não importa, uma vez que elas são aplicadas o resultado é o mesmo.

Operação *merge*: `c = merge(a, b)`

Existem CRDTs específicos para diferentes tipos de dados, como: registros, arrays, mapas.

No projeto serão utilizados CRDTs para os tipos de registro e mapa.

Replicated And Fault-Tolerant (Raft)

Raft é um tipo de consistência forte que chega a um consenso por meio de um líder eleito.

Algumas características principais envolvem:

- **Líder e seguidores:** O Raft divide os nós em líder e seguidores, onde o líder coordena as operações e os seguidores seguem as instruções do líder.
- **Eleição de líder:** O algoritmo utiliza um processo de votação para eleger um novo líder quando o líder atual falha.
- **Commit de operações:** As operações no Raft são confirmadas somente após receberem confirmações da maioria dos nós, garantindo a consistência dos dados.
- **Tolerância a falhas:** Utilizando timeouts e mecanismos de eleição para se recuperar de falhas e manter a disponibilidade do sistema.