Projeto de Formatura – 2024

PCS - Departamento de Engenharia de Computação e Sistemas Digitais

Engenharia de Computação

Tema:

Banco de dados chave-valor distribuído

Contexto e Motivação

O crescimento exponencial da internet e a demanda por processamento de grandes volumes de dados tornaram sistemas tradicionais centralizados insuficientes para muitas aplicações modernas. A computação em nuvem e os sistemas distribuídos emergiram como soluções eficazes, permitindo escalabilidade horizontal, melhor uso de recursos computacionais e tolerância a falhas.

Em particular, sistemas de gerenciamento de bancos de dados se beneficiam significativamente dessa abordagem, pois a distribuição de carga entre múltiplos nós melhora o desempenho e a resiliência. A pesquisa e desenvolvimento nessa área é essencial para sistemas que necessitem de alta performance e façam um grande consumo de dados. Como exemplo, podemos citar aplicações de aprendizado de máquina ou com muitos usuários.

Objetivo

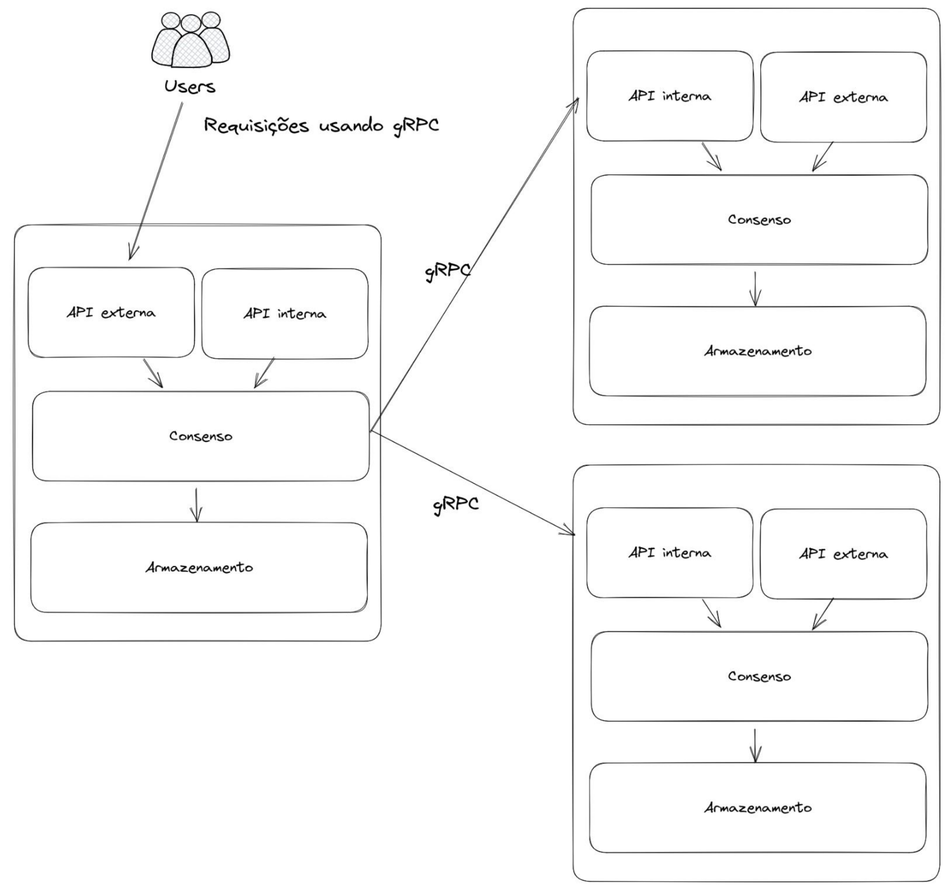
Desenvolver um sistema de banco de dados distribuído baseado no modelo chave-valor que seja capaz de: Lidar com grandes volumes de requisições, mantendo alto desempenho. Garantir consistência forte dos dados através da implementação de um algoritmo de consenso. Oferecer tolerância a falhas, permitindo a recuperação sem perda de dados em caso de falhas de nós. Permitir escalabilidade horizontal, facilitando a adição de novos nós conforme a necessidade.

Desenvolvimento

O projeto foi desenvolvido seguindo etapas bem definidas e uma arquitetura em camadas:

1. Camada de API: Interface com o usuário para operações de leitura e escrita.
2. Camada de Consenso: Implementa o algoritmo Raft para coordenação entre os nós.
3. Camada de Armazenamento: Responsável pela persistência dos dados em disco.

O sistema teve foi desenvolvido em golang e distribui os nós usando o algoritmo Raft para garantir consistência e tolerância ao particionamento de rede, mesmo que isso custe a sua disponibilidade. A figura abaixo exemplifica como diferentes nós se comunicam neste sistema:



Resultados

No sistema, foi implementado o algoritmo de Raft, de forma que cada nó persiste informações do termo atual (rodada da eleição, que contém informações sobre votos, liderança) e logs de chamadas à API.

De início, no termo de número 1 (um), não há nenhum líder. Após um timeout aleatório um nó pode se candidatar a ser líder e pede votos para outros nós. Caso haja mais deu um candidato,vence o nó que conseguir a maioria dos votos primeiro.

Periodicamente o líder enviar “heartbeats”, informando os outros nós que segue funcionando e novas informações que devem ser salvas.

Há um algoritmo de Two-Phase Commit, que garante que os dados só podem ser salvos após todos os nós do sistema concordarem com a nova entrada, garantindo consistência e que, em caso de falha da rede, não haja informações divergentes entre diferentes partes do sistema.

A comunicação entre nós é feita utilizando o protocolo gRPC e permite uma melhor eficiência, além de boa definição da API.

Tecnologias e utilizadas:

- Linguagem de Programação: Golang, escolhida por seu desempenho elevado e suporte a concorrência através de goroutines;
- Protocolo de comunicação: gRPC, escolhido pelo suporte a dados binários e melhor desempenho para sistemas distribuídos;
- Arquivo de configuração: YAML, escolhido pela facilidade de leitura e ser amplamente adotado para configurações de sistemas;
- Controle de Concorrência: Mutexes para impedir condições de corrida;
- Observabilidade: Ferramentas de logging estruturado para garantir entendimento fácil do sistema.

Integrantes: - Eduardo Niza Minosso
- Eduardo Thomaz dos Santos

Professor(a) Orientador(a): Prof. Dr. Jorge Rady