Performace de Query e Escalabilidade no NoSQL: MongoDB vs CouchDB

Allison Alfredo de Oliveira Sampaio[1], **Mara Luci Leite Goulart**[2]

[1]Departamento de Ciência da Computação – Universidade Tecnológica Federal do Paraná (UTFPR)
R. Rosalina Maria Ferreira, 1233 – Vila Carola, Campo Mourão – PR, 87301-899

[2]Departamento de Ciência da Computação – Universidade Tecnológica Federal do Paraná
Campo Mourão, PR, Brasil

allisonsampaiox@gmail.com, maragoulart@alunos.utfpr.edu.br

Abstract.

Resumo.

1. Introdução

O termo NoSQL foi primeiramente utilizado em 1998 como o nome de um banco de dados não relacional de código aberto, sendo criado não com o objetivo de substituir o Modelo Relacional e sim de servir como uma alternativa para atender casos onde haja necessidade de uma maior flexibilidade e distribuição dos dados. Um dos principais fatores que favoreceram seu surgimento foi a natureza dos dados da Internet, a importância de se atingir altos graus no processamento de grandes volumes de dados e a distribuição de sistemas em escala global. Os bancos de dados NoSQL possuem diversas categorias, para o presente estudo foi escolhida a categoria de bancos de dados orientado a documentos, dois clientes conhecidos são o MongoDB e o CouchDB. O CouchDB é um banco de dados de documentos flexível, que suporta JSON e preza pela consistência de dados. O MongoDB é um banco de dados não relacional, e utiliza BJSON para armazenar dados. Possui um significativo suporte de query e índices, o que permite um acesso rápido a grandes plataformas de dados. O Mongo pode acessar dez vezes mais rápido os dados de uma plataforma que excede 50GB do que um banco de dados SQL.

2. Objetivo

O objetivo do trabalho é realizar uma pesquisa com banco de dados NoSQL pré-selecionados para estudo e comparação, selecionando os bancos estudados para o desenvolvimento de uma aplicação para manipular dados armazenados nestes bancos.

3. Conceitos

Em meados dos anos 60 foi introduzido o termo banco de dados que serviria para armazenar informações, uma camada de suporte para os sistemas de informações. O conceito de separar aplicações dos dados era novo e abria possibilidades para aumentar a robustez das aplicações. Em 1970 Edgar Codd propos o modelo relacional que substituiu os modelos hierárquicos e de rede da época, o modelo de Codd tornou-se base para o uso de SQL que permite que os dados sejam armazenados em tabelas. Por volta dos anos 2000, as aplicações começaram a produzir um vasto volume de dados por meio de de aplicações

complexas. Segundo [Juravich 2012] , termo NoSQL não é a abreviação de “no SQL” – sem SQL, significa “not only SQL”, não apenas SQL, bancos de dados NoSQL são soluções de persistência de dados que não seguem o modelo relacional e não utiliza SQL para querying. A primeira característica que faz os bancos de dados NoSQL diferente é a sua estrutura de dados. Dentre as estruturas disponíveis, há a orientada a documentos, que salva os itens sem serem estruturados em um esquema. Essa estrutura de dados é remanescente de um array associativo em PHP. [Juravich 2012] acrescenta que o NoSQL deve ser utilizado quando o esquema e a estrutura dos dados mudam constantemente.

3.1. CouchDB

O CouchDB utiliza uma API HTTP/REST para a manipulação dos dados, portanto a comunicação entre entre cliente e servidor pode ser realizada em qualquer aplicação que possa comunicar-se por HTTP. As operações básicas de um banco de dados são mapeadas para as operações do protocolo HTTP, como GET, PUT, POS T, DELETE, para ler, armazenar, substituir e remover objetos armazenados [Juravich 2012]. Além da API REST/HTTP que provê acesso a qualquer aplicação com suporte ao protocolo HTTP, o CouchDB oferece um servidor web, chamado Futon, que pode ser acessado pelo browser através do endereço /localhost 5984/ utils/, onde é possível gerenciar os bancos e documentos individualmente. A arquitetura do CouchDB consiste em um servidor de banco de dados master que possui um ou mais bancos de dados criados. Os bancos são compostos por objetos, que são documentos do tipo JSON, de esquema livre, ou seja, em uma mesma base de dados os documento não precisam ter os mesmos campos. Entretanto, recomenda-se armazenar documentos semelhantes em um mesmo banco de dados [Redmond and Wilson 2012]. CouchDB suporta múltiplas réplicas do banco de dados rodando em diferentes servidores e fornece um mecanismo para sincronização destes dados. Há um processo, replicador, executando que possibilita informar a origem e o destino do servidor, é realizada uma verificação pela data e caso houve mudanças, a cópia é sobrescrito no servidor slave. As replicações são realizadas de forma assíncrona, de forma que não compromete a velocidade de escrita e leitura [Juravich 2012]. Dentre as principais características do CouchDB, estão:

a Um servidor de banco de dados de documento, acessível via RESTful JSON API.
b Ad-hoc e esquema com um largo espaço de endereço.
c Queries feitas a partir do JavaScript.

3.2. MongoDB

O aplicativo MongoDB é composto por dois tipos de serviços, o processo servidor mongod que é o núcleo do banco de dados e o serviço mongos para autosharding. Sharding é a divisão dos dados em vários nós, sendo utilizado quando faz-se necessário balanceamento de carga . O processo servidor pode rodar tanto em arquiteturas 32 como 64- bit, no entanto, é recomendado o uso de 64-bit, uma vez que o tamanho máximo de um banco de dados é limitado à 2GB no modo 32-bit. Além dos serviços do aplicativo servidor, há também serviço cliente mongo, que é um cliente shell padrão do MongoDB utilizado para conectar-se ao servidor através da porta 27017. Entretanto, há uma grande variedade de drivers oficiais disponíveis, como C, C++, Haskell, Java, PHP, entre outros, todos eles estão sob a licença Apache [Redmond and Wilson 2012]. Também é possível conectar através da interface HTTP/REST na porta 28017, permitindo a manipulação de entradas

via HTTP. O servidor MongoDB pode hospedar mais de um banco de dados, independentes entre si, armazenados separadamente. Um banco de dados contém um uma ou mais coleções constituídas por documentos BSON (Binary JSON), que são estruturados como documentos JSON, com esquema dinâmico, fazendo com que a integração de dados em certos tipos de aplicações sejam mais fáceis e rápidos [Garret 2016]. Os autores acrescentam que consultas são expressas em sintaxe como JSON e são enviadas ao servidor como objetos BSON pelo driver do banco de dados. O modelo permite consultas a todos os documentos dentro de uma coleção, incluindo objetos e matrizes incorporadas. MongoDB não possui transações, tampouco joins, ficando a cargo do desenvolvedor implementá-las, se necessário, em uma aplicação. É possível inclusive relacionar documentos de coleções distintas. Quanto aos recursos disponíveis, vale ressaltar o suporte automático para a realização de Sharding. O sistema de banco de dados pode ser distribuído através de um cluster de máquinas. O cluster consiste em três componentes, nós Shard, servidores de configuração e serviços de roteamento chamados mongos. Os nós Shard são responsáveis por armazenar os dados atuais, podendo inclusive ser replicados para redundância em caso de falha. Os servidores de configuração são usados para armazenar os metadados e rotear informação do cluster de MongoDB. Os Mongos são responsáveis por rotear os processos, otimizando o desempenho das tarefas requisitadas pelos clientes [Garret 2016]. Outro recurso importante é o GridFS, que serve para armazenar e recuperar arquivos que excedam o limite de 16M dos documentos BSON. Os arquivos são divididos em partes e armazenados em uma coleção, em outra coleção são armazenados os metadados do arquivo. É útil para armazenar arquivos grandes, como áudio e vídeo [Redmond and Wilson 2012].

4. Método de Avaliação

O método de avaliação do presente trabalho, será feito por meio do desenvolvimento de uma aplicação web que armazenará um pacote json de 74.5MB contendo 18.000 arquivos json. Os bancos de dados NoSQL utilizam queries dinâmicas, por exemplo, uma query SQL:

```
SELECT * FROM posts
INNER JOIN posts_tags ON posts.id = posts_tags.post_id
INNER JOIN tags ON posts_tags.tag_id == tags.id
WHERE tags.text = 'politics' AND posts.vote_count > 10;
```

E uma query equivalente em MongoDB

```
db.posts.find({'tags': 'politics', 'vote_count': {'$gt': 10}});
```

Aplicaremos um teste de stress de banco de dados quanto a replicação, partindo do princípio do que faz os bancos de dados orientados a documento únicos é a sua capacidade de lidar com documentos sem esquema definido em tabelas como noSQL. Eles possuem a capacidade de replicar suas coleções em diferentes servidores e performar queries em paralelo, se um servidor não está disponível ele faz requisição para o próximo [Redmond and Wilson 2012]. Os autores acrescentam que no caso do CouchDB ele permite múltiplos mestres, já o Mongo considera como sua força principal a abilidade de lidar com um grande volume de dados e múltiplas requisições por meio de replicação e escala horizontal, uma vez que o bancos de dados relacionais utilizam escala vertical.

4.1. Requerimentos do Sistema

O ambiente utilizado para a realização dos testes de desempenho possui um processador Intel Core i3 de sexta geração, sendo a memória de 4GB e o sistema operacional Windows 10 64 bits.

Notebook Inspiron 15 5000 i15-5566-A10P

Sistema Operacional: Windows 10 Home Single Language x64

Processador: Intel(R) Core(TM) i3-6006U CPU @ 2.00GHz

Memória (RAM): 4,00GB

4.2. Instalação MongoDB e CouchDB

O MongoDB é autônomo e não possui outras dependências do sistema. Você pode executar o MongoDB a partir de qualquer escolhida conforme o tutorial de instalação da própria aplicação [MongoDB 2017]. Quanto ao CouchDB não há qualquer pré-requisito extra para a instalação, sendo clara a documentação apresentada no próprio site [CouchDB 2017].

5. Resultados

Para a execução do teste de stress de banco de dados, utilizamos a ferramenta jmitter. O JMeter é uma aplicação implementada em Java projetada para fazer teste de carga de aplicações Cliente/Servidor. Foi originalmente desenhado para realizar testes em aplicações Web, mas, logo foi expandido para fazer outros tipos de testes, tais como em: servidores de banco de dados . Com o JMeter, é possível assegurar se um SGBD está capacitado para suportar uma determinada quantidade de usuários simultâneos e estudar o seu comportamento no que tange à escalabilidade. Neste contexto, entende-se por escalabilidade a capacidade de resposta do sistema em relação à demanda de recursos exigidos dele. Foi utilizada a replicação passiva, ou seja ter um master enviando todas as requisicões e os demais servidores as recebendo. O CouchDB permite a utilização de múltiplos mestres, o que torna algumas réplicas mestres, no entanto, essa configuração não será abordada neste trabalho.

6. Conclusão

7. Images

Referências

CouchDB (2017). *Couch Documentation*. CouchDB.

Garret, D. (2016). *Mongo in Action*. Manning Publications, 1th edition.

Juravich, T. (2012). *CouchDB and PHP Web Development*. Packt Publishing, 1th edition.

MongoDB (2017). *Mongo Documentation*. MongoDB.

Redmond, E. and Wilson, J. R. (2012). *Seven Databases in Seven Weeks*. LLC, 1th edition.