Centro universitário das Faculdades

Metropolitanas Unidas – FMU

Análise e Desenvolvimento de Sistemas

PRÁTICAS DE BANCO DE DADOS

APS

André Bezerra Ribeiro RA: 7343674

Denilson Elias de Souza Junior RA: 3324643

Jéssica Adriana Feitosa RA: 2146934

Juliana dos Santos Lima RA: 3895943

Lucas Silva Rodrigues de Oliveira RA: 3851869

São Paulo

2021

PRÁTICAS DE BANCO DE DADOS

APS

Pesquisa apresentada no curso de graduação de Análise e Desenvolvimento de sistemas das Faculdades Metropolitanas Unidas. Orientador Ademir Avila.

São Paulo

2021

SUMÁRIO

RESUMO

Este estudo, tem como objetivo entender, quando o Banco de Dados NoSQL pode ser utilizado, e o real motivo pelo qual pequenas e grandes empresas como o GOOGLE, tem escolhido esse tipo de banco de dados. Também abordaremos: Em quais momentos o Banco de Dados NoSQL faz sentido nas corporações? Por qual razão o Banco de Dados NoSQL tem ganhado tanto espaço no mercado mundial? Quais são as principais características do Banco de Dados NoSQL? Além de esclarecer as suas principais vantagens e desvantagens.

BANCO DE DADOS NOSQL

Um pouco de história

Antes de falar sobre NoSQL, vamos entender um pouco da história dos sistemas de gerenciamento de dados desde seu surgimento até o aparecimento deste movimento chamado NoSQL. Nos últimos 30 anos temos vivido a «ditadura» do modelo relacional, porém, é importante lembrar que os primeiros sistemas de gerenciamento de bancos de dados não eram baseados em estruturas relacionais, e sim hierárquicas ou baseados em grafo.

O movimento noSQL

O movimento noSQL teve sua origem em junho de 2009, para nomear um encontro promovido por Johan Oskarsson e Eric Evans, que teve como objetivo discutir o crescente surgimento de soluções open source de armazenamento de dados distribuídos não relacionais. Inclusive, é importante entender que noSQL não significa «no SQL», mas sim «not only SQL». Com isso, temos, pela primeira vez na história, uma nova onda surgindo sem o objetivo de substituir por completo o modelo atual. Portanto é bem irônico usar o termo noSQL, criado para nomear um banco de dados relacional, para classificar soluções de armazenamento de dados não relacionais.

O que é NoSQL?

Ferramentas NoSQL fornecem meios mais eficientes de armazenamento de grandes volumes de dados e/ou mecanismos de pesquisa de baixa latência, fatores importantes que precisam ser considerados durante a escolha de uma solução de armazenamento de dados.

Bancos de dados relacionais nem sempre são a melhor opção em cenários onde é necessário armazenar estruturas dinâmicas, tratar grandes volumes de dados ou lidar com estruturas não convencionais como grafos. Em contrapartida, as

tecnologias noSQL oferecem diversas maneiras de tratar estes pontos, inclusive, de forma a trabalhar em conjunto com bancos de dados relacionais. noSQL é um movimento que promove soluções de armazenamento de dados não relacionais. Ele é composto por diversas ferramentas que, de forma particular e específica, resolvem problemas como tratamento de grandes volumes de dados, execução de consultas com baixa latência e modelos flexíveis de armazenamento de dados, como documentos XML ou JSON.

As tecnologias noSQL não têm como objetivo substituir os bancos de dados relacionais, mas apenas propor algumas soluções que em determinados cenários são mais adequadas. Desta forma, é possível trabalhar com tecnologias noSQL e banco de dados relacionais dentro de uma mesma aplicação.

O termo noSQL é bastante abrangente, pois envolve diversas ferramentas, tecnologias, estruturas de dados e arquiteturas. Esta nova buzzword representa muito mais um movimento, ou uma nova escola de pensamento, do que alguma tecnologia em particular.

Escrever sobre este tema não é uma tarefa simples, pois além desta abrangência, é necessário lidar com dois pontos antagônicos: de um lado a desconfiança e o ceticismo sobre algo novo, do outro a excitação gerada por uma nova tecnologia.

Esta é a primeira parte de uma série de três artigos sobre noSQL, que irão apresentar, de forma consciente, clara e objetiva, o que de fato é noSQL. Neste artigo, será apresentado o conteúdo teórico, abordando os temas: história, modelos de dados alternativos ao relacional e arquitetura. Esta parte teórica é de extrema importância para compreensão do movimento como um todo, é fundamental para entender e nortear as escolhas das ferramentas classificadas como noSQL.

Após esta importante visão histórica, iremos agora nos aprofundar nas características das soluções chamadas noSQL. Vamos iniciar esta exploração pelos modelos alternativos ao relacional, pois esta é a porta de entrada para compreender melhor as propostas destas ferramentas noSQL.

Modelos de Dados Alternativos

Estamos habituados com o modelo relacional que é composto basicamente por tabelas, colunas e linhas, e que tem como principais características a integridade dos dados e a necessidade de modelar toda estrutura antes de seu uso.

Estrutura do modelo de família de colunas

Já o componente Família de Colunas tem uma estrutura mais próxima de uma tabela do modelo relacional, onde os dados são armazenados em linhas e organizados em colunas. A Listagem 2 mostra um exemplo, em formato JSON, de como os dados são organizados em uma família de colunas. Por fim, temos a coluna, que é uma tupla composta por nome, timestamp e valor, onde os dados são realmente armazenados. O elemento timestamp permite que uma única coluna armazene diversos valores, adicionando uma outra dimensão aos dados, como podemos ver no exemplo da Listagem 3.

O cenário típico de uso deste modelo está relacionado com a necessidade de lidar com grandes volumes de dados que precisam ser consultados com um tempo de resposta muito baixo, bem como a necessidade de armazenar uma estrutura de dados complexa.

Como temos tratado nossos dados

No início, nós programadores Java, tínhamos apenas um recurso para acessar bases de dados, o JDBC – uma API padrão que nos permitia acessar qualquer banco de dados que disponibilizasse um driver. As tecnologias de ORM trouxeram um ganho significativo de produtividade, contudo, para utilizar da melhor forma essa tecnologia, tivemos que abrir mão de algumas características do modelo relacional. Entretanto, quando começamos a propor isso aos ADs e DBAs, de cara foi rechaçado, pois estávamos «rasgando» o modelo relacional. Após severas discussões, os desenvolvedores acabaram «ganhando», pois conseguiram provar

que seriam mais produtivos, e os DBAs e ADs também não saíram perdendo, pois poderiam criar constraints para implementar as regras de integridade.

Dessa forma, manipular dados ficou muito simples, praticamente uma receita de bolo a ser seguida, o que nos levou a não nos preocupar mais com isso. No entanto, hoje vivemos a era do Big Data, onde um grande volume de dados é gerado a todo instante, e a manipulação deste volume começa a se transformar em um gargalo nos sistemas atuais. E devido à padronização do acesso aos dados através de ferramentas de ORM, acabamos nivelando a utilização dos recursos dos bancos de dados pelo menor denominador comum, e este menor denominador comum, é claro, não inclui operações específicas para melhoria de performance ou a criação de padrões de acesso diferenciados. O objetivo deste texto foi de apenas convidar você a refletir sobre como temos tratado o armazenamento de dados como um cidadão de segunda classe.

Nas próximas seções, vamos explorar algumas das principais características de arquitetura que devem ser analisadas quando avaliamos qualquer solução de armazenamento de dados, inclusive bancos de dados relacionais.

ACID vs BASE

Primeiro vamos deixar claro que as propriedades ACID são importantes, e todas as soluções de armazenamento de dados adorariam implementá-las, mas isso nem sempre é possível. Como mencionado anteriormente, até mesmo alguns bancos de dados relacionais não suportam todas as propriedades ACID em busca de performance e até mesmo simplificação de arquitetura. Existem diversos estudos sendo realizados em busca de equacionar esses pontos, porém ainda muito incipientes. A ideia principal é abrir mão da consistência em favor da disponibilidade e escalabilidade.

Entenda que abrir mão de consistência não quer dizer que seus dados estarão sempre inconsistentes, significa apenas que seus dados podem ficar inconsistentes por um pequeno período de tempo. Um bom exemplo para entender o estado de inconsistência temporário é através da operação de transferência de valores entre bancos. Após a confirmação da operação do banco destino, os dados retornam ao

estado consistente. Em resumo, em sistemas ACID o dado é modificado de estado consistente para consistente a cada operação, já no BASE os dados estão em estado de fluxo, isto é, em constante modificação.

Para finalizar, vamos deixar claro que existem ferramentas noSQL com propriedades ACID, um exemplo é o Neo4j.

Modelo de Persistência

Grande parte dos bancos de dados relacionais atuais tem como meio de persistência principal o disco, o que, em conjunto com logs e outras técnicas, são capazes de oferecer garantia de durabilidade dos dados. Esta técnica aumenta significativamente a velocidade de acesso aos dados, entretanto seu uso acarreta alguns problemas. O principal deles é o risco de perda de dados que pode ser causado por uma corrupção no arquivo, ocasionado por um crash na máquina. Ferramentas que utilizam esta técnica aconselham replicar os dados em mais de uma máquina, para diminuir o risco de perda de dados.

Mas o disco não é o único local onde podemos alocar dados. A memória é um local quase perfeito para «armazenar» dados. Com ela conseguimos obter o menor nível de latência, e nos dias atuais não é difícil encontrar servidores com 32GB ou até mesmo 64GB de memória – para algumas aplicações esta quantidade de memória é suficiente para alocar praticamente todo o banco de dados. Infelizmente a memória é volátil, ou seja, se a máquina for desligada, ou até mesmo se ocorrer uma falha na aplicação e ela for finalizada, você perderá todos os dados.

Modelo de Distribuição

Para isso existem algumas técnicas, e uma das mais comuns é a distribuição e replicação dos dados em mais de uma máquina. Para implantar este cenário é fundamental entender quais são os recursos que sua ferramenta de armazenamento de dados disponibiliza, tais como replicação master/slave ou até master/master. Porém, na vida real é um «pouco» diferente, e a melhor forma de saber como efetivamente todo este processo ocorre é através de testes reais e benchmarks, de preferência com os dados de sua aplicação, para comprovar o funcionamento de cada ferramenta no seu ambiente. Existem ainda ferramentas que

oferecem o particionamento automático dos dados, que nada mais é do que a capacidade de dividir e distribuir os dados de uma grande base para bases menores em diversas máquinas.

Particionar dados consiste em dividir um grande banco de dados em bancos de dados menores independentes. Para esta divisão é utilizado algum fator, por exemplo, a letra inicial do nome do usuário. Em bancos não relacionais como chave-valor, documento ou família de colunas, esta técnica não tem impactos negativos, uma vez que não existe relacionamento no modelo. Já para o caso do modelo de grafo, aplicar esta técnica não é nada fácil, pois os nós e links são criados de forma dinâmica e altamente acoplados, tornando o particionamento de dados uma tarefa muito complicada.

Escalabilidade e Elasticidade

Quando precisamos tratar grandes volumes de dados, precisamos ter a capacidade de criar clusters não só para nossos servidores de aplicação, mas também para as nossas ferramentas de armazenamento de dados. Apesar de grande parte das ferramentas de bancos de dados relacionais oferecer algum mecanismo de escalabilidade horizontal, o modelo relacional se comporta melhor em um modelo de escalabilidade vertical. Em contrapartida, algumas ferramentas noSQL têm seu comportamento planejado para trabalhar de forma mais otimizada em escala horizontal. Em favor da escalabilidade vertical tem sua simplicidade, pois inicialmente é mais simples adicionar mais hardware sem ter que modificar o software.

Já a escalabilidade horizontal tem como maior vantagem a capacidade de expansão linear, pois sempre é possível adicionar novas máquinas. Entretanto, criar sistemas para trabalhar de forma distribuída é bem mais complexo.

EMPRESA QUE UTILIZA O NOSQL (GOOGLE)

Google é uma empresa multinacional de serviços online e software dos Estados Unidos. O Google hospeda e desenvolve uma série de serviços e produtos baseados na internet e gera lucro principalmente através da publicidade pelo AdWords.

Desde sua incorporação culminou em uma cadeia de outros produtos, aquisições e parcerias que vão além do núcleo inicial como motor de buscas. A empresa oferece softwares de produtividade online, como o software de e-mail Gmail, e ferramentas de redes sociais, incluindo o fracassado Google+ e os descontinuados Google Buzz e Orkut. Os produtos do Google se estendem à área de trabalho, com aplicativos como o navegador Google Chrome, o programa de organização de edição de fotografias Picasa e o aplicativo de mensagens instantâneas Google Talk. Notavelmente, o Google também lidera o desenvolvimento do sistema operacional móvel para smartphones Android, usado em celulares de marcas como Samsung, Motorola, LG, HTC, Huawei e Xiaomi.

Em 2015, o Google tinha mais de 2,2 bilhões de usuários, porém essa contagem parou, portanto, é complicado de dimensionar. Sem dúvidas, trata-se do maior buscador e talvez seja a empresa de tecnologia que mais cresce atualmente.

PRINCIPAIS MOTIVOS PARA USAR NOSQL

Bases de dados NoSQL têm sido adotadas por um grande número de empresas, especialmente aquelas com plataformas de serviços como Google, que oferecem a Google NoSQL Big Data Database Service e outras como o Facebook e Amazon.

Quando comparados a bancos de dados relacionais – RBDMS, sistemas NoSQL oferecem mais escalabilidade, melhor performance em queries e flexibilidade para manipular dados que podem apresentar mudanças em seu formato. No artigo de hoje falaremos sobre essa tecnologia e seus benefícios.

Document Model

Enquanto em bancos de dados relacionais os dados são estruturados em tabelas e os dados são gravados em linhas, no document model, cada unidade de informação é armazenada em um documento com a notação JSON. Em cada documento são inseridos os campos com os tipos de informação correspondente como nome, geolocalização, dados monetários, entre outros. Quando se faz uma busca nesse tipo de base de dados, pode-se especificar quais dados se pretende encontrar em cada campo ou simplesmente buscar a informação em todos os documentos, sem alguma especificação de qual tipo de dado se pretende encontrar.

Graph Model

Organizam dados de um mesmo tipo em nodos. Eles são ligados entre si por conectores que os relacionam. Esse modelo permite ricas representações de informações complexas como conexões em redes sociais, topologias de rede e redes de logística.

Key-Value Stores

Utiliza um simples modelo de dados em que cada informação tem um nome ou chave que podem ser usadas nas queries.

Wide Column

Wide column ou família de colunas utiliza um amplo mapa multidimensional para armazenar os dados. Cada unidade de informação pode ser armazenada em uma ou mais colunas que são agrupadas em famílias e os dados são obtidos por chave primária por coluna de família. O Google Big Table usa esse modelo de armazenamento de dados.

NoSql x SQL

Em bancos, as tabelas e o relacionamento entre elas requerem cuidado tanto para modelagem quanto para atualizações.
Bancos de dados NoSQL oferecem mais flexibilidade para gravação, não precisam obedecer a uma concepção prévia de seu formato. Novas propriedades podem ser adicionadas a uma entidade do sistema no instante de gravação. Com isso, o time de desenvolvimento amplia a agilidade para criar e testar novas funções sobre informações novas.

Outra vantagem é a escalabilidade. Sistemas NoSQL são projetados para permitirem a distribuição de dados em múltiplos servidores — clustering — sem prejuízo à performance e à integridade dos dados.

Para tanto, utiliza o modelo sharding, no qual cada servidor armazena um subconjunto de informações ou replication em que os dados são copiados em diversos servidores de tal forma que cada bit de dados fica presente em múltiplos lugares.

Assim, sistemas que apresentam rápido crescimento do volume de dados armazenados têm mais facilidade para escalarem seus projetos.
Bancos de dados relacionais foram projetados para rodar em apenas um servidor para preservar a integridade das tabelas e de seu mapeamento, o que os torna ineficientes em clusterings.

NoSQL e RGBDS em um mesmo projeto

Ambos podem ser usados juntos em um projeto, inclusive a definição mais utilizada para NoSQL atualmente é Not only SQL — não apenas SQL.

Quando juntos, as informações que exigem maior integridade de dados para transações financeiras, por exemplo, são manipuladas em bancos RGBDS.
Já as informações que são geradas como maior velocidade e que serão usadas para análises em Big Data — aplicações em internet das coisas, análises de tendências em redes sociais — são armazenadas em bancos NoSQL.

Bases de dados NoSQL trazem muitos benefícios para uma aplicação. Facilitam a escalabilidade, oferecem flexibilidade para alterações no formato dos dados e tornam análises e buscas de dados muito mais eficientes. A

Google usa esse modelo, com tecnologia Bigtable em seu mecanismo de busca na web, Google Earth e também em sua solução em Cloud, a Google NoSQL Database.

CONCLUSÃO

O Google está incluído na lista das grandes empresas que utilizam NoSQL, porque é uma empresa que inova a cada dia. Isso faz com que os seus bancos de dados precisem ser não relacionais, para garantir uma performance melhor.Especialmente no quesito de conseguir fazer a manipulação dos dados que podem até apresentar algumas mudanças no seu formato. Outra vantagem é a escalabilidade, ou seja, a chance de ter uma plenitude de rendimento.

Podemos definir noSQL como o fim dos bancos de dados relacionais como bala de prata, estimulando a escolha de ferramentas que melhor se adequem às necessidades de nossas aplicações. A maior contribuição do movimento noSQL foi trazer à tona um componente arquitetural importante e que estava sendo negligenciado, o modo como armazenamos e tratamos nossos dados.Bancos de dados NoSQL oferecem mais flexibilidade para gravação, não precisam obedecer a uma concepção prévia de seu formato.

Para tanto, utiliza o modelo sharding, no qual cada servidor armazena um subconjunto de informações ou replication em que os dados são copiados em diversos servidores de tal forma que cada bit de dados fica presente em múltiplos lugares. Assim, sistemas que apresentam rápido crescimento do volume de dados armazenados têm mais facilidade para escalarem seus projetos. Bancos de dados relacionais foram projetados para rodar em apenas um servidor para preservar a integridade das tabelas e de seu mapeamento, o que os torna ineficientes em clusterings.

Por esses e outros motivos, diversas empresas têm adotado o NoSQL em suas corporações, a cada vez mais.

REFERÊNCIAS

GOOGLE NoSQL: as principais vantagens de um banco de dados não relacional. [S. l.], 13 jul. 2017. Disponível em: https://www.qinetwork.com.br/google-nosql-e-suas-vantagens/#:~:text=Facilitam%20a%20escalabilidade%2C%20oferecem%20flexibilidade,Cloud%2C%20a%20Google%20NoSQL%20Database. Acesso em: 30 abr. 2021.

O QUE é NoSQL? Bancos de dados não relacionais de alta performance com modelos de dados flexíveis. [S. l.], 2021. Disponível em: https://aws.amazon.com/pt/nosql/#:~:text=NoSQL%20(n%C3%A3o%20relacional),SQL%20Server%2C%20MySQL%20e%20PostgreSQL.&text=Muitas%20vezes%2C%20o%20termo%20%E2%80%9CNoSQL,intercambi%C3%A1vel%20com%20%E2%80%9Cn%C3%A3o%20relacional%E2%80%9D. Acesso em: 5 maio 2021.

O QUE é noSQL? - Java Magazine 86. [S. l.], 2010. Disponível em: https://www.devmedia.com.br/o-que-e-nosql-java-magazine-86/18777. Acesso em: 5 maio 2021.

GOOGLE. [S. l.], 2021. Disponível em: https://pt.wikipedia.org/wiki/Google. Acesso em: 10 maio 2021.