Introdução

Nesse relatório serão abordadas características essenciais dos bancos de dados relacionais (SQL), não-relacionais (NoSQL) e de uma nova vertente chamada NewSQL, as vezes citada como novo relacional. A partir dessas observações, é possível entender mais a fundo as diferenças entre os modelos de bancos de dados e seu uso.

NewSQL é um termo cunhado no ano de 2011 pelo pesquisador do laboratório 451 Matthew Aslett para descrever uma nova geração de banco de dados. Esse termo hoje descreve bancos de dados que tentam prover a escalabilidade horizontal dos bancos de dados NoSQL para OLTP juntamente com as garantias de um banco sequencial de consistência com o modelo ACID. Muitas empresas que trabalham com dados sensíveis e que necessitam de um sistema sem inconsistências e com o processamento capaz de escalar horizontalmente tem optado por essa nova alternativa.

Bancos de dados como o Google Spanner, VoltDB e CockroachDB tem crescido em adoção nos últimos anos, provendo serviços de alta confiabilidade para empresas de seguros, mercado financeiro, jogos e mais.

SQL

O termo SQL (Structured Query Language) em si é uma linguagem de programação usado para a manipulação de banco de dados relacional. Porém, atualmente, o termo SQL é mais usado para referenciar próprio banco de dados relacional.

Essa linguagem e estrutura de armazenamento de dados foi inicialmente criada pela IBM na década de 70, baseado no modelo relacional idealizado por Edgar F. Codd. Simplificadamente, esse modelo propõe que todos dados são representados por tuplas e agrupados em relações (tabelas). Assim, é possível utilizar expressões muito similares à álgebra relacional para manipular os dados. Portanto, a linguagem SQL consegue:

- Definir dados (data definition), como criar e modificar tabelas;
- Manipular dados (data manipulation), como inserir dados em relações (tabelas);
- Consultar dados (data query), como visualizar dados de uma relação;
- Controlar dados (data control), como revogar permissões de um usuário.

A linguagem SQL proporciona a realização de transações, que geralmente são um conjunto de operações. Na década de 80, foi cunhado o conceito ACID, que define quatro características de transações para que essas tenham uma maior confiabilidade e causem menos falhas no banco de dados. O nome ACID é um acrônimo das quatro características:

- **A**tomicidade: Todas as operações de uma transição ou são executadas e confirmadas pelo *commit* ou, caso haja falha, nenhuma é confirmada, dando um *rollback* em todas as operações executadas até o momento da falha. Isso é possível graças um sistema de log de dados que gravam as operações executadas antes da transação ser confirmada. não podendo confirmar-se apenas parte da transação.
- **C**onsistência: Regras como primary key, foreign key, cascades e triggers garantem que não haja transações que corrompa o estado do banco de dados.
- **I**solamento: Uma transação em andamento deve ficar isolada à mudanças feitas por outras transações.

- **D**urabilidade: As transações confirmadas não são perdidas em caso de falhas do sistema.

A linguagem padronizada, inspirada na álgebra relacional, e as transações geram grandes vantagens como:

- Facilidade e rapidez na consulta no banco de dados;
- Facilidade na manipulação de vários dados simultaneamente sem que causem falhas no banco de dados;
- Alta integridade dos dados;

Em contrapartida, essas características trazem desvantagens, sendo as principais:

- Necessidade de definir bem previamente a estrutura dos dados, fazendo a sua implementação inicial complexa;
- Dificuldade em mudar a estrutura de dados, requerendo também uma mudança de todos os dados já obtidos;
- Possível perda de dados para que estes estejam de acordo com o padrão da estrutura de dados;
- Majoritariamente, apenas consegue escalar verticalmente, não suportando dados grandes (como Big Data).

Exemplos de bancos de dados SQL são o MySQL, PostgreSQL e MariaDB.

NoSQL

O termo NoSQL se refere aos bancos de dados não-relacionais. Ou seja, principalmente, os bancos de dados não-relacionais não tem tem uma linguagem unificada própria definida e nem um padrão de armazenamento. Esse modelo existe desde quando o armazenamento em computação começou a ser relevante, porém o termo NoSQL foi definido e popularizado por conta da crescente geração de conteúdo na internet.

Por não ter um padrão único e definido que nem o modelo SQL, um banco de dados NoSQL pode ter diferentes maneiras de armazenar e manipular os dados. Mais comumente, são usados quatro tipos:

- Coluna;
- Documento;
- Chave-valor;
- Grafo.

Independente do tipo, o modelo NoSQL não precisa se preocupar muito com como os dados são guardados, ou seja, é projetado para dados estruturados de qualquer maneira (estruturado, semi-estruturado, não estruturado). Além disso, não segue a fundo as características ACID, abrindo espaço para um modelo mais flexível. Com essas características, o modelo NoSQL proporciona:

- Estrutura de dados flexível:
    - Atualização dinâmica dos schemas;
    - Facilidade no armazenamento;
- Escalabilidade horizontal e alta disponibilidade;
- Alta performance.

Porém, por não ser relacional nem seguir as características ACID, o NoSQL apresenta algumas desvantagens como:

- Não apresentar consistência de integridade dos dados;
- Possível perda de dados de escrita;
- Não ser padronizado, dificultando migrações;
- Sem suporte à análise de dados como o SQL.

Exemplos de bancos de dados NoSQL são Apache Cassandra (colunas), MongoDB (documentos), Couchbase (chave-valor) e OrientDB (grafo)

NewSQL

Usado pela primeira vez em um artigo de 2011, o termo NewSQL se refere a novos bancos de dados que buscam juntar as vantagens de bancos SQL e NoSQL. Os bancos NewSQL usam diversas técnicas que possibilitam a combinação dos dois modelos. As principais técnicas são:

- **Partitioning/Sharding**: Muito utilizada no modelo NoSQL, a partição/fragmentação do banco de dados entre vários nós (*shards*) é a responsável pela escalabilidade horizontal. A diferença entre o sharding do NewSQL é que cada nó roda o mesmo banco de dados, porém só com uma parte dele. Além disso, o nó não é acessado e atualizado independentemente por diferentes aplicações. A maneira de fragmentação pode variar dependendo da biblioteca ou serviço utilizado.
- **Main Memory Storage**: Essa técnica é pouco usada em modelos SQL pois, como este escala verticalmente e a sua tecnologia é antiga, os bancos de dados SQL são majoritariamente projetados para serem guardados em disco. Porém, com o sharding e o barateamento dos custos de memória, cada parte do banco de dados nos shards cabem na memória, não sendo mais necessário utilizar o disco. Assim, no NewSQL, o acesso aos dados ficam ligeiramente mais rápidos sem que haja um gasto extra desproporcionalmente maior que o ganho.
- **Concurrency Control**: O que é feito no modelo SQL com protocolos de concorrência sofisticados, como o Multi-Version Concurrency Control (MVCC) com two-phase locking (2PL). Porém, a maneira com que o grande parte de bancos NewSQL lidam com operações concorrentes é usando protocolos baseados em timestamp para manter uma consistência maior, ou até usando MVCC descentralizados, sendo mais rápidos mas podendo comprometer a consistência.
- **Secondary Indexes**: Além das chaves-primárias do modelo SQL, o NewSQL implementa índices secundários, que são alternativas para consultas mais rápidas, principalmente em consultas que envolvem mais de um shard.
- **Replication**: A replicação é a principal técnica que garante a consistência desejada ao banco de dados. Quando um dado sofre uma atualização, os outros nós precisam ficar cientes da atualização. Portanto, assim como no NoSQL, a sincronia entre os nós é feita a partir de um algoritmo de consenso como o *Paxos* ou o *Raft*.

- **Crash Recovery**: Além de não perder os dados atualizados como em um modelo SQL, o modelo NewSQL se preocupa em minimizar o downtime. Em sistemas sem réplicas, usando o algoritmo *Write-ahead logging* (WAL), a recuperação a partir de um log que grava todas as operações acontecidas, não é necessário recuperar todo o banco de dados, apenas a parte do nó. Em sistemas com réplicas, basta o nó escravo assumir o lugar do mestre. Então o nó com falha é consertado, atinge a sincronia e assume lugar de escravo ou retoma a posição de mestre.

Apesar de todas todas as técnicas, não é possível ter 100% de todas as vantagens dos dois modelos. Por exemplo: um banco pode ter alta consistência dos dados, mas a prioridade para alcançar um consenso entre os nós causaria grande perda na velocidade de execução nas operações.

O maior inconveniente dos bancos NewSQL atualmente é o preço. Com nós descentralizados, é necessário ter diferentes locais físicos e cada um com sua própria equipe de administração e seus custos.

SQL x NoSQL x NewSQL

A tabela abaixo mostra as principais diferenças entre os três modelos de banco de dados:

	SQL	**NoSQL**	**NewSQL**
Relacional	Sim	Não	Sim
"Schema"	Baseado em tabelas que contêm colunas e linhas. Ligações entre tabelas precisam de integridade referencial.	Pode ser baseado em documentos, pares de chave-valor, colunas ou grafos. Não exigem um Schema padronizado.	Baseado em tabelas que contêm colunas e linhas. Ligações entre tabelas precisam de integridade referencial.
- Flexibilidade do Schema	Baixa	Alto	Moderado
- Propriedades ACID	Sim	Não	Sim, mas não à risca
- Consistência da integridade dos dados	Alta	Baixa	Depende
- Consultas analíticas	Sim	Não	Sim
Suporte para particionamento	Não	Sim	Sim
- Escalabilidade	Vertical	Horizontal	Horizontal

Performance	Limitado ao subsistema do disco	Limitado pelo tamanho do shard e a latência de rede	Limitado pelo tamanho do shard e a latência de rede
Desempenho com crescimento de dados	Rápido	Rápido	Muito rápido
Sobrecarga de desempenho	Alta	Moderada	Depende
Tolerância a falhas	Baixa	Alta	Depende

Exemplo de NewSQL - Google Spanner

O Google Spanner é um DBaaS recente, que faz parte da Google platform desde 2017. Como a própria Google define, é um: "*serviço de banco de dados escalonável, de nível corporativo, globalmente distribuído e fortemente consistente criado para a nuvem, especificamente para combinar os benefícios da estrutura de banco de dados relacional com a escala horizontal não relacional. Essa combinação resulta em transações de alto desempenho e alta consistência entre linhas, regiões e continentes.*"

Ele utiliza o modelo SQL, criando schemas e tabelas. Na questão de sharding, ele divide as tabelas e guarda cada parte em diferentes nós , podendo ter redundância entre os nós. Essa redundância entre nós faz com que, se houver um nó com falha, o sistema continue funcionando. Os dados são replicados a partir do esquema de replicação Paxos com até três tipos de réplicas (read-write, read-only e witness), facilitando a chegada à um consenso entre dados em todo o sistema, além de prover mais velocidade às queries, com base na divisão de carga. Com diferentes tipos de réplicas, é possível fazer diferentes tipos de transações sem necessariamente submeter o sistema inteiro à mudanças e/ou estado de lock. Assim, é possível realizar ainda mais acessos ao banco de dados sem comprometer a integridade dos dados. Essas características tornam o Spanner uma alternativa confiável ao SQL, ao mesmo tempo que comporta a escalabilidade horizontal de bancos NoSQL. Porém, essas vantagens são acompanhadas de um alto preço, de em média 1 dólar por hora para cada nó para máquinas simples.

Conclusão

O modelo NewSQL é uma revolução na área de banco de dados, porém ainda não tem tanto crédito quanto os outros dois modelos pelo fato de ser uma tecnologia recente. Porém, deve-se ficar de olho pois o NewSQL tem se mostrado capaz de oferecer um equilíbrio customizado das vantagens dos modelos de SQL e NoSQL e, a medida de seu crescimento e popularização, ele terá cada vez mais suporte e será mais fácil e viável para implementar.

Bibliografia

https://db.cs.cmu.edu/papers/2016/pavlo-newsql-sigmodrec2016.pdf
https://en.wikipedia.org/wiki/SQL

https://acodez.in/sql-and-nosql-an-overview/
https://medium.com/devtranslate/diferencas-entre-sql-e-nosql-51311f9069bd
https://aws.amazon.com/pt/nosql/
https://en.wikipedia.org/wiki/NoSQL
https://www.softwaretestinghelp.com/sql-vs-nosql/
https://www.tutorialspoint.com/dbms/dbms_concurrency_control.htm
https://blog.geekhunter.com.br/sql-nosql-newsql-qual-banco-de-dados-usar/#Vantagens_e_desvantagens_SQL_NoSQL_e_NewSQL
https://cloud.google.com/spanner/