Análise Qualitativa do Desempenho de Soluções NewSQL in-memory para Ambientes Centralizados e Distribuídos

Lucas A. Lisboa[1], **João Victor R. Ferro**[1], **Fábio J. Coutinho** [1]

[1]Instituto de Computação – Universidade Federal de Alagoas (UFAL)
Maceió – AL – Brasil

{lal, jvrf, fabio}@ic.ufal.br

Abstract. *NewSQL solutions related to data storage in main memory have gained prominence in this decade, aiming to optimize operational performance for large volumes of data. In this context, VoltDB and MemSQL are known DBMSs considered as references of this new trend. Thus, this paper aims to analyze their performance using the TPC-C benchmark, in a centralized and distributed view. Experiments carried out demonstrated the superior performance of MemSQL over VoltDB in both environments, as well as a significant loss of performance of both software in the distributed environment.*

Resumo. *As soluções NewSQL, atreladas ao armazenamento de dados em memória principal, têm ganhado destaque nesta década, visando otimizar o desempenho operacional para grandes volumes de dados. Nesse contexto, os Sistemas de Gerenciamento de Banco de Dados (SGBDs) VoltDB e MemSQL são tidos como referências de software desta nova tendência. Assim, este artigo visa analisar o desempenho deles, mediante o uso do benchmark TPC-C, tanto em ambiente centralizado, quanto em ambiente distribuído. Os experimentos realizados demonstraram a superioridade de desempenho do MemSQL em relação ao VoltDB em ambos os ambientes, bem como de uma perda significativa de desempenho dos dois softwares no ambiente distribuído.*

1. Introdução

Nos últimos anos, tem surgido cada vez mais aplicações que demandam o processamento de um grande volume de requisições a um baixo tempo de resposta, o que demanda requisitos de escalabilidade e desempenho dos Sistemas de Gerenciamento de Banco de Dados (SGBDs). Tal circunstância permitiu o surgimento dos bancos de dados em memória principal, os quais estão calçados em pilares distintos daqueles estudados, no decorrer de décadas, por sistemas de bancos de dados relacionais. Projetos de SGBDs em que os dados residem em memória discutem novas técnicas para lidar com temas que já foram muito explorados sob a ótica relacional (controle de concorrência, processamento de consultas, durabilidade, recuperação, entre outros), caracterizando o surgimento do NewSQL.

Associado a esse tipo de tecnologia, tem-se desenvolvido soluções de armazenamento *main-memory*, isto é, o armazenamento dos dados na memória principal, diminuindo os custos de acesso, tendo em vista que o tempo de acesso à memória secundária é superior. No entanto, a durabilidade de SGBDs em memória é prejudicada em virtude da volatilidade do meio de armazenamento.

Diante dessa perspectiva, este artigo provê uma análise comparativa do desempenho de SBGDs em memória principal, tendo sido selecionados para este fim o VoltDB e o MemSQL (objetos da pesquisa referente a projeto do ciclo PIBIC 2019-2020). Ambas as soluções de armazenamento foram avaliadas tanto em ambiente centralizado quanto distribuído.

2. Fundamentação Teórica

2.1. NewSQL

As tecnologias de Banco de Dados NewSQL surgem com a premissa de: garantir as propriedades ACID (Atomicidade, Consistência, Isolamento e Durabilidade) dos modelos Relacionais Tradicionais, ao mesmo que buscam conseguir um desempenho similar aos modelos NoSQL. Nesse sentido, [Stonebraker and Cattell 2011] definiram algumas características importantes de um SGBD NewSQL:

- Uso da Linguagem SQL como meio de interação entre Banco de Dados e aplicação;
- Suporte para transações ACID;
- Controle de Concorrência não bloqueante, evitando conflitos entres as leituras e as escritas;
- Arquitetura que visa melhor desempenho por nó de processamento;
- Arquitetura escalável, com memória distribuída;

DataBases\ Características	***VoltDB***	***NuoDB***	***CockroachDB***	***MemSQL***
Categoria	Nova Arquitetura Databases	Nova Arquitetura Databases	Nova forma de armazenamento MySQL	Nova Arquitetura Databases
Disponibilidade	Código Aberto. Edição Comercial.	Código Fechado. Edição dos Desenvolvedores.	Código Aberto. Edição da comunidade.	Código Aberto. Edição da comunidade.
Desenvolvido por	VoltDB Inc. Michael Stonebraker, 2010	Jim starkey in 2012	Spencer Kimball at cockroach labs in 2014	MemSQL Inc. in 2013
Empresas que usam	Nokia, Ericsson, Flipkart, hp	UAExchange, Forbes, Kodiak	Ninguém até o momento	Intel, Microsoft Azure
Consistência	Forte	Eventual	Forte	Forte
Tipo de armazenamento	Em memória	Em memória	Em memória	Em memória em linhas/ armazenamento em coluna de disco
Concorrência de Controle	Não, modelo único de threaded	Yes, MVCC	Sim, otimização do controle de concorrência	Yes, MVCC
Escalabilidade	Bidirecional, scale-up, scale-out	scale-out	Horizontal, escala automática	Altamente escalável horizontalmente e durável

Figura 1. Tabela Comparativa dos SGBDs NewSQL por [Kaur and Sachdeva 2017] (em tradução livre)

Dessa forma, o VoltDB e o MemSQL foram escolhidos pelo fato de apresentarem maior facilidade de uso, código aberto, consistência forte e de empresas de grande relevância para o mercado de *software* adotarem tais soluções, como demonstrado na Figura 1.

2.2. VoltDB

O SGBD VoltDB, criado por Michael Stonebraker, Sam Madden e Daniel Abadi, utiliza o armazenamento *in-memory* e, consequentemente, em sua configuração padrão, os dados não tem durabilidade, ou seja, se a conexão com o VoltDb for encerrada os dados serão perdidos. Pode-se alterar essa opção, escolhendo o nível de durabilidade almejada, levando em consideração os custos computacionais dessa decisão.

Outro fator importante do funcionamento do VoltDB é o sistema de particionamento e balanceamento de dados [VoltDB 2020]. Quando configurado em ambiente distribuído, um nó mestre fica responsável por coordenar a distribuição da carga dos dados visando tornar as transações independentes entre si. Para isso, ele tenta deixar dados codependentes em um único nó, isto é, os dados que possuem dependência transacional entre si (como, por exemplo, um valor e seu identificador) são armazenados fisicamente próximos um do outro, para que a quantidade de conflitos de dados diminua [Stonebraker and Weisberg 2013]. Assim, os criadores definem três tipos de transações, as quais estão representadas na Figura 2:

Single-node transactions (Grande maioria): São transações que podem ser executadas em um único nó do cluster;

One-shot transactions (Pequeno Número): São transações que ocorrem em paralelo entre nós distintos, mas que podem ser executadas simultaneamente sem ocasionar conflito;

General transactions (Minoria): São transações que envolvem mais de um nó, com dependência entre eles.

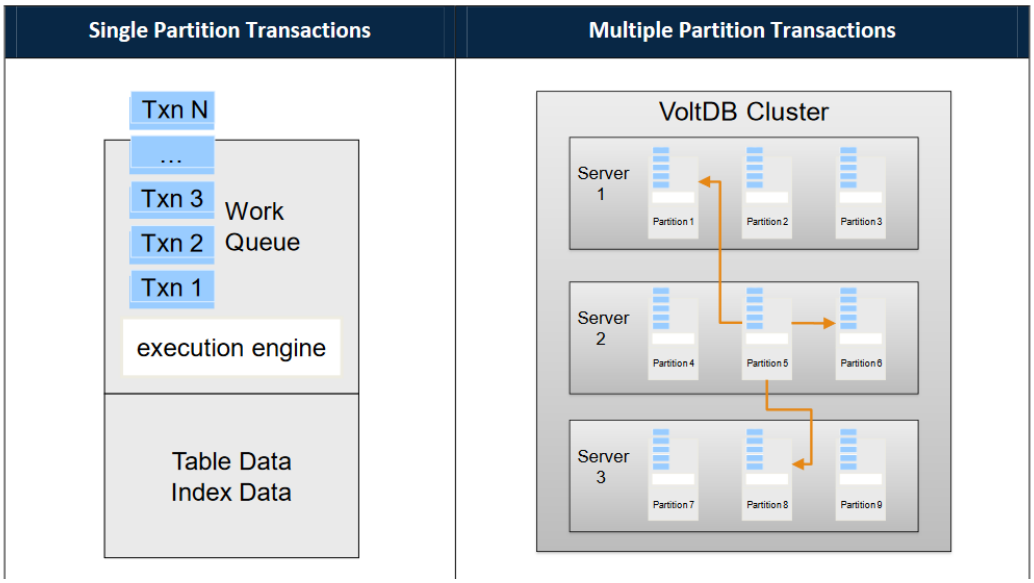

Figura 2. Arquitetura Simplificada do VoltDB [VoltDB 2013]

2.3. MemSQL

MemSQL é altamente escalável, tendo suporte tanto para cargas transacionais quanto analíticas [MemSQL 2020]. Em ambiente transacional, utiliza-se o armazenamento *rowstore* com os dados mantidos em memória principal e a durabilidade sendo garantida

por um *Write Ahead Log* (WAL). Dados analíticos também podem ser armazenados como *columnstore* - formato otimizado para processamento *Online Analytical Processing* (OLAP).

No aspecto da distribuição de dados, MemSQL faz uso de uma arquitetura *shared-nothing*, utilizando *sharding* automático para balancear a carga entre os nós do *cluster*, isto é, o SGBD particiona as tabelas de maneira horizontal a fim de otimizar a carga em seus nós. Em relação à replicação são oferecidas as opções *redundancy-1* (os dados não tem cópias extras) e *redundancy-2*, que funciona com grupos de disponibilidade (*availability groups*). Os grupos permitem que nós compartilhem dados de forma pareada, sendo mestres para algumas partições e escravos para outras, ou seja, caso um nó falhe, seus dados não se tornarão indisponíveis.

Além disso, outros pontos que influenciam no desempenho do MemSQL é a separação da arquitetura em duas camadas. Nesse sentido, os agregadores lidam com metadados, roteiam consultas e agregam seus resultados e os nós folha executam as consultas designadas aos mesmos, como pode ser observado na Figura 3, bem como é usado o *Multiversion Concurrency Control* (MVCC) e estruturas de dados livres de bloqueios para os controles de concorrência.

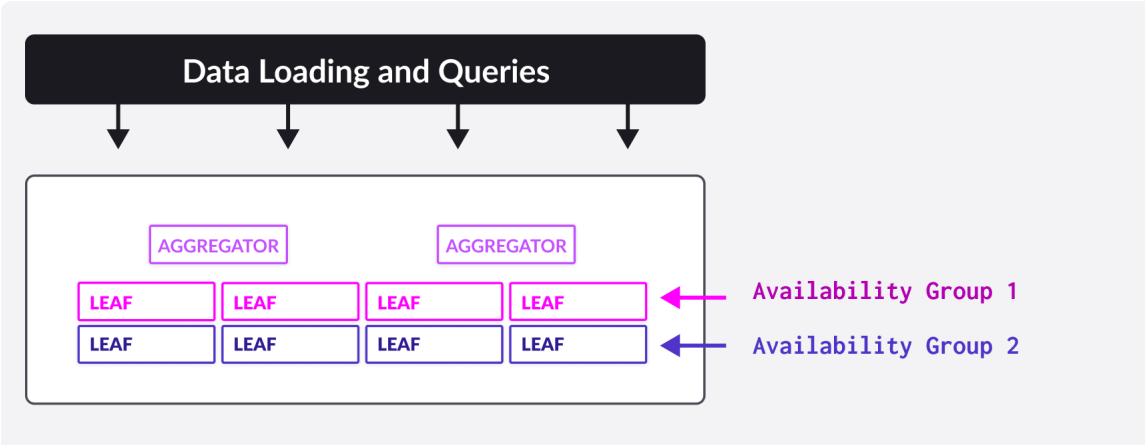

Figura 3. Arquitetura Simplificada do MemSQL [MemSQL 2020]

2.4. Trabalhos Correlatos

Inicialmente, foi realizada uma pesquisa em periódicos da área e demais fontes bibliográficas a fim de selecionar trabalhos que abordassem a análise de desempenho das tecnologias de armazenamento abordadas em nosso projeto. Também foram estudados os principais *benchmarks* utilizados nesses trabalhos de pesquisa.

Dentre os trabalhos encontrados, destacam-se o de [Lemos 2014] que faz uma análise do SGBD NuoDB em três ambientes, com um, dois e quatro nós, utilizando o TPC-C, TATP, YCSB e Twitter (todos OLTP-Bench) como benchmarks escolhidos. O trabalho de [Kaur and Sachdeva 2017] utiliza apenas um tipo de ambiente de teste (centralizado) e não usa *benchmarks* conhecidos, optando por métricas e base de dados próprios, analisando os SGBDs CockroachDB, MemSQL,NuoDB e VoltDB. Em [Knob 2018] também é utilizado um único tipo de ambiente, neste caso, distribuído

com os *benchmarks* YCSB, Voter e TPC-H, sendo avaliados os mesmos SGBDs de [Kaur and Sachdeva 2017].

3. Metodologia

Neste trabalho, foi realizada a comparação dos SGBDs MemSQL e VoltDB utilizando o *benchmark* TPC-C em ambiente centralizado e distribuído. Este cenário de execução desses SGBDs, de forma centralizada e distribuída, distingue nossa análise daquelas realizadas pelos trabalhos descritos anteriormente.

3.1. Ambiente de Testes

Foram criados dois ambientes para os testes. O primeiro, sendo centralizado, foi com a utilização de um único computador com processador Ryzen 5 2400G com 4 núcleos de 3.6GHz, duas memórias RAM de 8gb DDR4 2133 MHz, disco rígido de 1TB 7200RPM e sistema operacional Ubuntu 18.04. Já o segundo, sendo distribuído, foi executado na mesma interface de rede com taxa de transmissão entre 10 a 100 Mb/s, em que foram utilizadas as seguintes máquinas:

- **Computador 1:** Dell Inspiron INS-3470-M40 8a Geração Intel Core i7 8GB RAM 1 TB HD (6 núcleos, 12MB Cache, até 4.6 GHz) ;
- **Computador 2:** HP Compaq DC5850 AMD Phenom X4 4GB RAM 250GB HD (2.3 GHz);
- **Computador 3:** Pauta Connect BB8701A MoBo Itautec SM3322 AMD Phenom X2 4GB RAM 320GB HD (2.0 GHz).

Durantes os testes, o Computador 1 serviu como host para o sistema distribuído. Todos os computadores também utilizaram o sistema operacional Ubuntu 18.04.

Acerca das versões dos SGBDs: no caso do VoltDB, foi utilizada a versão versão 9.1 (lançada em 08 de Agosto de 2019), disponibilizada no Github; já no MemSQL, foi utilizada a versão 7.0 (lançada em agosto de 2018), disponibilizada no site da MemSQL Inc. Ambos os *softwares* foram utilizados em suas configurações padrões.

3.2. Benchmark TPC-C

Neste trabalho, a comparação realizada entre os SGBDs fez uso do *benchmark* TPC-C, composto por um conjunto de cinco transações de leitura e escrita, que simulam as atividades encontradas em um ambiente OLTP complexo de um sistema atacadista. O funcionamento do TPC-C prevê um número espalhado de armazéns (*warehouses*) e distritos(*districts*), os quais determinam o tamanho da organização a partir da composição de armazéns que ela possui. Ou seja, conforme ela cresce mais armazéns e distritos são criados e cada armazém mantém estoque de todos os cem mil produtos e cada um de seus distritos atende a três mil clientes (*customers*) [Mendes 2006].

São realizadas 5 operações:

- *New Order*
- *Payment*
- *Delivery*
- *Order Status*
- *Stock Level*

As operações mais executadas são a *New Order* (leitura/escrita) e *Payment* (leitura/escrita), as quais impõem uma carga considerável ao sistema. Para efeito da análise realizada neste trabalho, foram avaliados o *thoughput* (operações por segundo) e a latência média (microssegundos) de cada SGBD.

4. Resultados e Discussões

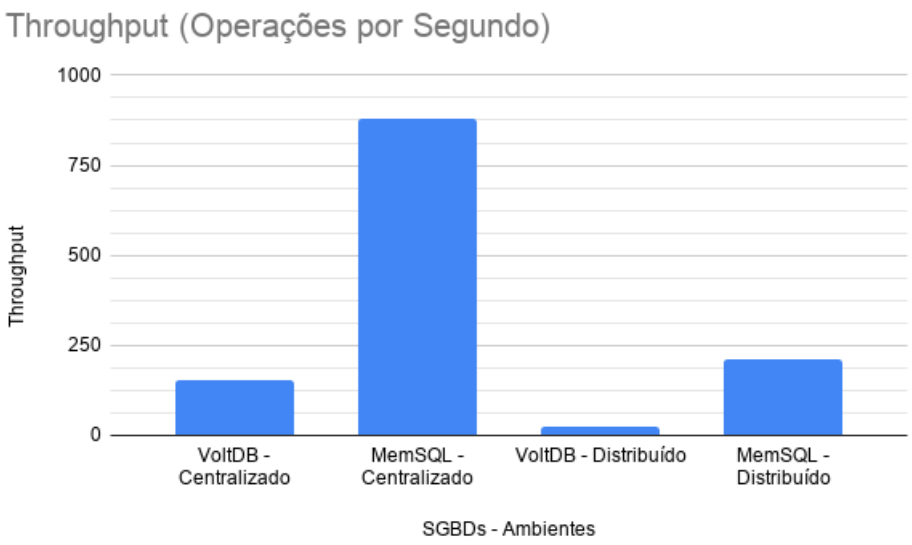

Figura 4. *Throughput*

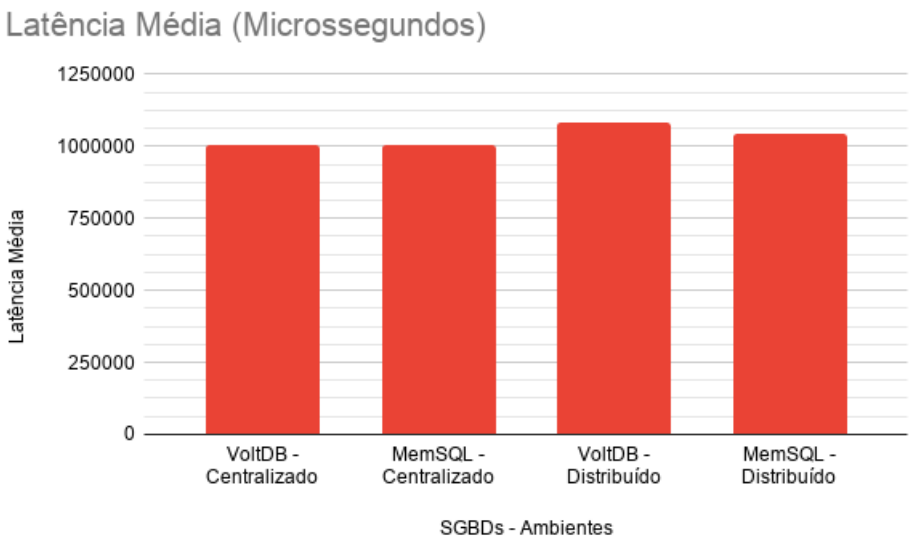

Figura 5. Latência

Como demonstrado na Figura 4, em ambiente centralizado, o VoltDB teve cerca de 153,389 transações por segundo, como *throughput*, enquanto o MemSQL obteve 881,36, ou seja, uma superioridade operacional de 474,59%. Já as latências médias de ambos,

observadas na Figura 5, foram bastante similares, com 1006173,6 microssegundos para o VoltDB e 1007021,4 para o MemSQL, representando uma diferença de apenas 0,084%.

Em ambiente distribuído, foi observado que o desempenho de ambos, em relação ao throughput, teve uma redução bastante significativa. O VoltDB passou a realizar 22,333 operações por segundo, o que representa uma redução de 85,44% em sua capacidade operacional em relação ao ambiente centralizado. Já o SGBD MemSQL obteve o valor de 213,406 operações por segundo, o que significa uma redução de 75,79%. Verifica-se que a diferença entre os dois SGBDs continua grande neste quesito, resultando em 855,56% de aumento operacional do MemSQL em relação ao VoltDB, isto é, um aumento de 66,30% na diferença relativa entre os dois software avaliados. A latência continua próxima, havendo apenas um pequeno aumento em ambos, com o VoltDB apresentando 1081359,0 microssegundos e MemSQL 1041234,5, o que significa uma diferença de apenas 3,71%, isto é, 3,79% de aumento na diferença relativa.

Um aspecto que poderia explicar a discrepância entre os valores de *throughput* é a durabilidade, uma vez que os SGBDs seguem estratégias distintas para a preservação dos dados, o que pode ser muito custoso dependendo de como é implementado. Por exemplo, no VoltDB, se configurado para salvar um log de comando a cada transação, em sistemas em que ocorrem muitas transações, vai ter uma perda de desempenho, mas se configurado para salvar os logs em intervalos de tempo então essa perda de desempenho vai ser menor, porém na segunda opção tem menor segurança em caso de falha. Essa perda de desempenho se agrava na medida em que o tamanho dos dados aumentam, o que significa a grande perda de desempenho no *benchmark* TPC-C.

Outro importante fator para a superioridade do MemSQL no *throughput* é sua arquitetura em camadas de nodos agregadores, otimizando as operações de leitura, e o mecanismo MVCC, evitando o bloqueio de dados no controle de concorrência. Assim, o SGBD consegue realizar operações paralelizadas de leitura com mais facilidade.

Considerando o ambiente distribuído, a piora no desempenho pode ser explicada a partir da necessidade dos SGBDs garantirem a propriedade de atomicidade, exigindo uma maior comunicação entre os nós para que não ocorra nenhuma inconsistência [Lemos 2014]. A própria arquitetura do VoltDB corrobora isso, por focar em ter um ambiente otimizado para as transações chamadas *single-node* e *one-shot*, logo, abdica-se de ter um bom desempenho no outro tipo de transação (*general*), sendo que o TPC-C executa majoritariamente este tipo.

Os resultados apresentados neste trabalho, considerando todos os ambientes executados, corroboram com os que foram obtidos em [Kaur and Sachdeva 2017] no que tange o desempenho do VoltDB e o MemSQL, mesmo que partindo de metodologias distintas. Outro ponto a se destacar é a perda de desempenho entre ambiente centralizado e distribuído, encontrada por [Lemos 2014], sendo também observada aqui, para ambos os SGBDs analisados, podendo, assim, caracterizar um possível padrão dos SGBDs *newSQL in-memory*. Por fim, é interessante observar que em todos os testes realizados por [Knob 2018] também houve superioridade, tanto no *throughoput*, quanto na latência média, do MemSQL em relação ao VoltDB, ou seja, o *benchmark* TPC-C reproduz a mesma tendência observada nos *benchmarks* Voter, TPC-H e YCSB.

5. Conclusão

Mediante os resultados observados, é possível afirmar que o SGBD MemSQL possui superioridade de desempenho em relação ao VoltDB para o conjunto de operações avaliadas pelo TPC-C, tanto em ambiente centralizado, quanto em ambiente distribuído. Assim, dentre os *softwares* avaliados, infere-se que o MemSQL seja o mais indicado para o sistema financeiro de lojas atacadistas, por ser o foco de análise do *benchmark* utilizado.

Além disso, há a perda de desempenho nos dois SGBDs na transição para o sistema distribuído, no que tange o conjunto de operações do TPC-C. Isso indica que eles não estão otimizados para transações que necessitem acesso em múltiplos nós.

5.1. Trabalhos Futuros

Como trabalhos futuros, pode-se destacar a ampliação da quantidade de SGBDs NewSQL in-memory analisados, como NuoDB e CockroachDB, além de realizar testes a partir de outros *benchmarks*, a exemplo de YCSB, Voter, JPAB e LinkBench. Outra importante questão é aprofundar a análise realizada a fim de observar em quais transações do TPC-C há perda maior de desempenho, em especial no caso do VoltDB.

Referências

Kaur, K. and Sachdeva, M. (2017). Performance evaluation of newsql databases. In *2017 International Conference on Inventive Systems and Control (ICISC)*, pages 1–5. IEEE.

Knob, R. R. (2018). Análise e benchmarking das soluções newsql: cockroachdb, memsql, nuodb e voltdb. TCC (Graduação) -Curso de Sistema da Informação, Universidade Federal de Santa Catarina.

Lemos, Pedro Henrique dos Santos; Figueiredo, P. S. (2014). Uma análise dos novos sistemas de bancos de dados relacionais escaláveis. TCC (Graduação) - Curso de Engenharia de Computação e Informação, Universidade Federal do Rio de Janeiro, Rio de Janeiro.

MemSQL, I. (2020). Memsql documentation. Disponível em: https://docs.memsql.com/ . Acesso em: 14 mai. 2020.

Mendes, M. R. N. (2006). Análise de desempenho de sistemas oltp utilizando o benchmark tpc-c. TCC (Graduação) - Curso de Ciência da Computação, Universidade Federal de Pernambuco, Recife, 2006.

Stonebraker, M. and Cattell, R. (2011). 10 rules for scalable performance in'simple operation'datastores. *Communications of the ACM*, 54(6):72–80.

Stonebraker, M. and Weisberg, A. (2013). The voltdb main memory dbms. *IEEE Data Eng. Bull.*, 36(2):21–27.

VoltDB, I. (2013). Voltdb technical overview. Disponível em: http://www.odbms.org/wp-content/uploads/2013/11/VoltDBTechnicalOverview.pdf . Acesso em: 14 mai. 2020.

VoltDB, I. (2020). Using voltdb. Disponível em: https://docs.voltdb.com/UsingVoltDB/ . Acesso em: 14 mai. 2020.