1 Introdução

Uma modelo de base de dados distribuídas utilizando sistemas de gestão de bancos de dados, SGBDs, relacionais é uma ferramenta que provê escalabilidade horizontal em soluções utilizando ferramentas que permitam que agregar as funcionalidades do modelo relacional em uma arquitetura onde a persistência dos dados não é feita em uma mesma máquina física ou virtual. Esta abordagem de manutenção dos dados em um ambiente distribuído permite que equipamentos ampliar a capacidade de manipulação e manutenção de dados que compreendem grande consumo de recursos computacionais seja realizado por equipamentos com reduzida capacidade de processamento. A ampliação do número de usuários e da quantidade de dados mantidos pela infraestrutura corporativa pode ser redimensionada de acordo com as demandas que venham a surgir ao longo do tempo sem que haja a necessidade de dimensionamento prévio de processamento, armazenamento e memória.

Um dos grandes desafios na utilização de SGBDs relacionais distribuídos está ligado a própria concepção do modelo que depende do atendimento de premissas concebidas para garantir a integridade dos dados e operações que implicam em processos pouco adequados a ambientes distribuídos do ponto de vista do tempo de execução das tarefas de persistência das informações. A necessidade de garantia de transações utilizando um único ponto onde são mantidas os registros implicam em confirmações e bloqueios de acesso a determinados dados para atender a aspectos de garantia de unicidade das informações. Este modelo de persistência de dados foi concebida em um momento em que a computação ainda possuía um grande enfoque em modelos centralizados de processamento de dados e atendeu muito bem a maioria das demandas até o início da década de 2000 (KOSSMANN, 2000) mas com a proposta de maior escalabilidade de recursos de hardware para atendimento pontual das necessidades de maneira que não exista superdimensionamento de infraestrutura em decorrência de expectativas que podem nem sempre se realizar levantou um questionamento contundente da adequação do modelo centralizado as novas demandas das ferramentas de Tecnologia da Informação e Comunicação, TICs.

Neste panorama de questionamento de modelos de persistência de dados em ambientes distribuídos surgiram propostas e ferramentas que tem sido defendidas como mais adequadas que os tradicionais SGBDs relacionas denominados com bancos de dados noSQL, *not struturad query language*, em tradução livre não SQL, que foram rapidamente aceitos como solução para organizações dotadas de grande demandas de manutenção de informações tais como sistemas de busca na Internet e ambientes de redes sociais. Existem notadas vantagens do ponto de vista de utilização destas ferramentas em ambientes distribuídos pois várias delas foram concebidas justamente para prover performance nestas arquiteturas tecnológicas e são criadas para prover melhor desempenho em situação específicas. Estas ferramentas, no entanto, apresentam um modelo de uso diferenciado para cada

solução não existindo, como no caso dos bancos de dados relacionais SQL, uma forma relativamente semelhante de manipulação de informações, motivo este que impõe grande dependência da aplicação a ferramenta escolhida para a persistência dos dados.

Muitas das soluções de software que tem grande dependência da manutenção e recuperação de informações em banco de dados foram desenvolvidas utilizando o modelo relacional. Muitos dos gestores de projetos e desenvolvedores de aplicação tiveram em sua formação e vida profissional experiências apanas com ferramentas SQL e dado ao relativo sucesso na maioria das implementações utilizando estas ferramentas sequer vislumbram a possibilidade de abordagem de outras ferramentas para soluções que utilizem bancos de dados. A mudança desta visão de utilização de repositórios que não utilizem bancos relacionais implica em uma mudança significativa na abordagem de desenvolvimento de sistemas e pode representar necessidades de reimplementação que demanda custos impraticáveis, a curto prazo, para a maioria das organizações.

A avaliação do desempenho de modelos distribuídos utilizando banco de dados relacionais é um importante fator na decisão na escolha de implementar esta abordagem para a melhoria da qualidade de sistemas que apresentem a necessidade de prover crescimento de infraestrutura ou distribuição das informações em ambientes geograficamente distantes. Consolidar informações relativas ao desempenho da solução utilizando distribuição dos dados permite a tomada de decisões de implementação que permitam a geração de resultados mais consistentes e adequados a necessidade de escalabilidade permitindo tanto a definição da arquitetura mais adequada ao projeto quanto avaliar os impactos no desenvolvimento da solução. Em casos em que já existe uma implementação baseada em modelos centralizados testes de performance permitem avaliar os impactos de alterações na implementação dos sistemas e definições de estratégias que promovam menor impacto na adequação da solução podendo inclusive apontar a inviabilidade de determinados projetos adequarem-se ao modelo distribuído (LIN et al., 2011).

1.1 Objetivos

1.1.1 Objetivos gerais

Avaliar o desempenho, sobre a ótica de tempo de execução, de bancos de dados relacionais em ambientes distribuídos.

1.1.2 Objetivo específico

Gerar estratégias de testes de tempo de execução de consultas e inserção de dados em bancos de dados relacionais centralizados e distribuídos.

Construir uma ferramenta capaz de automatizar os teste para geração de dados para serem avaliados.

Criar um modelo de comparação de dados dependendo da infraestrutura e quantidade de dados utilizados durante os testes.

2 Trabalhos correlacionados

3 Metodologia

Este trabalho é de caráter exploratório e pretende apresentar um modelo de identificação de desempenho na utilização de bancos de dados relacionais implementados de forma distribuída. Para a obtenção do modelo é necessária a construção de uma ferramenta capaz de realizar testes de manutenção e recuperação de informações em sistemas instalados em equipamentos distintos para a produção de dados relativos ao tempo de execução das instruções. Os dados gerados pela ferramenta serão utilizados para a identificação de relações em relação aos tempos de execução e a infraestrutura disponibilizada para estes testes.

Para a realização dos testes foi utilizado um banco de dados relacionais distribuído sobre licença livre, o Postgres SQL, de tal foma que permite a construção de ambiente similar para qualquer pesquisador que pretenda repetir os procedimentos utilizados neste trabalho. Não existe, no entanto, nenhum empecilho em relação a utilização de outras soluções de SGBD que possivelmente produziram valores diferentes em relação aos obtidos na experiência aqui proposta mas que cumprem rigorosamente com os objetivos de avaliação de desempenho. O modelo de avaliação produzido neste trabalho apresenta fatores de ajustes para que possam ser feitas avaliações de comportamento utilizando implementações distintas de SGBD.

Os testes de desempenho aplicados consistem no envio instruções para vários servidores de bancos de dados utilizando processamento paralelo das instruções e recuperação das informações retornadas pelos servidores acionados. Para os testes propostos foram criadas quatro tabelas dotadas de relacionamentos entre elas, sem imposição de regras de relacionamentos como chaves estrangeiras, e com o uso de chaves primárias em todas as tabelas utilizadas para os testes. Para todas as tarefas foram utilizadas instruções do padrão SQL/ISO sem utilização de otimizações de consultas particulares do SGBD.

A implementação construída para os testes utiliza uma estratégia de sincronização de relógios dos servidores utilizando a escala de micro segundos para que garantir diferenciação entre o tempo de execução das consultas e a latência de rede. Esta sincronização de

relógios permite a criação de variáveis que indicam o tempo gasto para a recuperação de dados e tempo que estes levam para percorrer a rede produzindo valores para a avaliação de infraestruturas distintas. Para esta sincronização a estratégia utilizada foi a recuperação do tempo do processador de cadas um dos servidores antes do início das transações e manutenção dos mesmos em uma tabela mantida em memória na implementação que executa as chamadas aos outros servidores.

A partir dos resultados obtidos pelos testes foi construído um modelo que representa os fatores de correção de tempo de execução das instruções em ambiente distribuído para serem utilizadas na equação que estabelece valores para comparação com as mesmas instruções executadas no modelo tradicional de SQL utilizando uma única máquina onde são mantidos todos os dados do sistema. As instruções executadas incluem agrupamentos de registros, *joins*, que são processadas de forma independente quando realizadas em ambiente distribuído. O banco de dados utilizado no teste não utiliza implementação de regras de integridade referencial nativas do banco de dados mas prevê testes que impõe regras de relacionamento em inserções e atualizações distribuídas utilizando validação de campos com instruções implementadas no próprio aplicativo de testes.

Quando ocorrem testes que consideram junções ou agrupamento de dados a solução adotada neste trabalho foi da realização da solução destas implementadas na própria aplicação de testes. Os valores de tempo gasto nestas operações foram obtidos sem customização de algoritmos que executem estas tarefas o que abre a possibilidade de obtenção de melhores resultados em consultas que utilizem junções de tabelas através da implementação de algoritmos mais eficientes na realização de tarefas de junção e ordenação de dados.

A aplicação dos testes pela ferramenta de validação utilizaram tabelas com diferentes dimensionamentos de número de colunas e quantidade de registros por tabela para serem comparados através de funções lineares ou logarítmicas indicando tendências de melhoria ou piora dos tempos de execução. Os valores obtidos foram testados novamente para comprovação dos valores no modelo teórico construído a partir dos experimentos empíricos passando-se a incluir a variável de utilização compartilhada de recursos, simulando a utilização simultânea de grupos de usuários ao mesmo banco de dados para observar o impacto de compartilhamento de recursos no modelo obtido em acessos únicos.

As instruções SQL a serem executadas foram representadas em uma tabela e para cada uma das instruções foi relacionado um mnemônico que a representa. Os resultados das execuções dos testes também foram representados em tabelas que apresentam o mnemônico da instrução, a quantidade de registros contidos na tabela do bando de dados, o tempo de execução da instrução e o tempo de latência da rede para retornar os valores solicitados.

A analise dos resultados considerou os valores médios obtidos para o tempo de execução quando foram realizados testes com mais de três servidores pois os resultados mais relevantes em implementações do modelo são os dados obtidos no pior e melhor caso que são sempre representados nas tabelas. A utilização deste método de organização ocorre pela necessidade de avaliar a latência da consulta mais demorada para apresentação completa dos resultados em um sistema, o melhor caso pode ser um indicador útil em relação a informações de possibilidades de melhorias na infraestrutura. A verificação individualizada dos dados dados vai além do escopo deste trabalho mas pode ser obtida pela replicação desta metodologia utilizando a ferramenta de testes responsável pela geração dos dados aqui avaliados.

O sistema implementado para a realização dos testes está distribuído sobre a licença GPL e pode ser acessado no repositório git no site git hub e pode ser recuperado com a instrução git github:XXXXXXXX. Toda implementação utilizou ferramentas distribuídas como software livre, foi implementada em linguagem C++ e compilado com o compilador gcc em sistema operacional Linux utilizando as distribuições Ubuntu e Debian. A opção da linguagem C++ ocorreu para permitir melhor desempenho da aplicação e maior independência de implementações de máquinas virtuais ou interpretadores, a distribuição do código fonte sem nenhuma compilação oferece ao implementador que deseje repetir a experiência uma compilação customizada para a arquitetura além de ajustes no código para atendimento de demandas específicas.

4 Analise dos dados

As instruções utilizadas nas consultas de teste visam cobrir apenas conjuntos de instruções SQL mais simples mas os resultados obtidos permitem suposições de tempo de execução de consultas mais complexas utilizando-se modelos que representem cada uma das instruções separadamente.

Mnemônico	Instrução
TSA	SELECT * FROM test
TSC	SELECT count(test_id) FROM test
TSW	SELECT * FROM test WHERE test_id = 20
TSL	SELECT * FROM test WHERE value_string like '1 %'
CSA	SELECT * FROM test_child1
CSJ	SELECT * FROM test_child1 t1, test t where t.parent_id = t1.id
BSA	SELECT * FROM test_child_blob
MSA	SELECT * FROM test_child_multi_col

Os servidores de banco de dados utilizados para a realização dos testes foram men-

surados em relação a sua capacidade de processamento de instruções SQL utilizando a instrução identificada pelo mnemônico TSC e a velocidade de rede em relação ao tempo médio de execução de cem instruções ping. A arquitetura do equipamento é irrelevante para o estudo proposto e foram considerados apenas o tempo de rede independentemente da infraestrutura para a geração dos resultados que serão aplicados como base das validações.

Mnemônico	Ping	TSC
M1	0.8	0.12
M2	0.8	0.12
M3	0.8	0.12
M4	0.8	0.12
M5	0.8	0.12
M5	0.8	0.12
M7	0.8	0.12
M8	0.8	0.12

Para avaliar o comportamento da execução considerando o crescimento do número de registros e relação do número de campos retornados nas consultas foram obtidos os seguintes resultados médios da execução.

Mnemônico	Registros	TL	TLP	TLM	TLMd	TRP	TRM	TRMd	TA
TSA	10K	0.1	0.1	0.00	0.1	0.1	0.00	0.1	0.1
CSA	10K	0.1	0.1	0.00	0.1	0.1	0.00	0.1	0.1
TSA	100K	0.3	0.2	0.00	0.1	0.1	0.00	0.1	0.1
CSA	100K	0.3	0.2	0.00	0.1	0.1	0.00	0.1	0.1
TSA	1000K	0.5	0.3	0.00	0.1	0.1	0.00	0.1	0.1
CSA	1000K	0.5	0.3	0.00	0.1	0.1	0.00	0.1	0.1
TSW	10K	0.1	0.1	0.00	0.1	0.1	0.00	0.1	0.1
TSW	10K	0.1	0.1	0.00	0.1	0.1	0.00	0.1	0.1
TSW	100K	0.3	0.2	0.00	0.1	0.1	0.00	0.1	0.1
TSL	10K	0.1	0.1	0.00	0.1	0.1	0.00	0.1	0.1
TSL	10K	0.1	0.1	0.00	0.1	0.1	0.00	0.1	0.1
TSL	100K	0.3	0.2	0.00	0.1	0.1	0.00	0.1	0.1
CSA	10K	0.1	0.1	0.00	0.1	0.1	0.00	0.1	0.1
CSA	10K	0.1	0.1	0.00	0.1	0.1	0.00	0.1	0.1
CSA	100K	0.3	0.2	0.00	0.1	0.1	0.00	0.1	0.1
CSJ	10K	0.1	0.1	0.00	0.1	0.1	0.00	0.1	0.1
CSJ	10K	0.1	0.1	0.00	0.1	0.1	0.00	0.1	0.1
CSJ	100K	0.3	0.2	0.00	0.1	0.1	0.00	0.1	0.1
BSA	10K	0.1	0.1	0.00	0.1	0.1	0.00	0.1	0.1
BSA	10K	0.1	0.1	0.00	0.1	0.1	0.00	0.1	0.1
BSA	100K	0.3	0.2	0.00	0.1	0.1	0.00	0.1	0.1
MSA	10K	0.1	0.1	0.00	0.1	0.1	0.00	0.1	0.1
MSA	10K	0.1	0.1	0.00	0.1	0.1	0.00	0.1	0.1
MSA	100K	0.3	0.2	0.00	0.1	0.1	0.00	0.1	0.1

5 Resultados

Utilizar uma arquitetura de servidores heterogênea implica em redução do tempo de execução de todo sistema desenvolvido para consultas compartilhadas pois a dependência de respostas de todos os nós que compreendem o sistema implica que o tempo de resposta da consulta distribuída é o tempo de execução do nó mais lento que atende a requisição. O critério de utilização de servidores com menor capacidade de processamento pode ser uma solução viável quando estes mantém quantidades de registros significativamente menores que os equipamentos mais rápidos. Para consultas que retornam poucos dados a velocidade da rede tem pouco impacto no tempo de execução total de consultas mas representa um impacto maior quando ocorrem muitos resultados no retorno da mesma.

6 Considerações Finais e Trabalhos Futuros

Referências

KOSSMANN, D. The state of the art in distributed query processing. *ACM Computing Surveys (CSUR)*, ACM, v. 32, n. 4, p. 422–469, 2000.

LIN, L. et al. Tenzing a sql implementation on the mapreduce framework. Citeseer, 2011.