UNIVERSIDADE DO EXTREMO SUL CATARINENSE

CURSO DE CIÊNCIA DA COMPUTAÇÃO

FERNANDO RODRIGUES

ANÁLISE DE DESEMPENHO E CARGA EM ACESSOS SIMULTÂNEOS A UM BANCO DE DADOS

CRICIÚMA, DEZEMBRO 2007

FERNANDO RODRIGUES

ANÁLISE DE DESEMPENHO E CARGA EM ACESSOS SIMULTÂNEOS A UM BANCO DE DADOS

Trabalho de Conclusão de Curso apresentado para obtenção do Grau de Bacharel em Ciência da Computação da Universidade do Extremo Sul Catarinense.

Orientador: Prof. MSc. Daniel Pezzi da Cunha

CRICIÚMA, JUNHO DE 2007

FERNANDO RODRIGUES

ANALISE DE DESEMPENHO E CARGA EM ACESSOS SIMULTÂNEOS A UM BANCO DE DADOS

Submetido ao corpo docente do Departamento de Ciência da Computação da Universidade do Extremo Sul Catarinense como um dos requisitos para obtenção do grau de Bacharel em Ciência da Computação.

Prof[a] MSc. Ana Cláudia Garcia Barbosa
Coordenadora do Curso de Ciência da Computação

Banca Examinadora:

Prof. MSc. Daniel Pezzi da Cunha
Orientador

Prof. MSc. Paracelso de Oliveira Caldas

Esp. Marcelo Mazon
Especialista em Banco de dados/ Depto TI-UNESC

Dedico este trabalho ao meu filho Enzo, que ainda não nasceu mais seus chutes no ventre de sua mãe já me dão à alegria de fazer valer a pena.

AGRADECIMENTOS

Agradeço primeiramente a Deus, sem sua luz não teria como chegar até aqui, aos meus Pais Aldo e Zilda, que sempre me deram incentivo e apoio, muitas vezes me fazendo ressurgir com seu amor, a minha esposa Michele que sempre esteve do meu lado em momentos difíceis.

Agradeço, também, aos meus grandes amigos, Moacyr Couto, Flavio, Fabio Tome, Margaret Dagostim, Magno Cardoso, Filetti, Lena e Luana que sempre me incentivaram e me prestarão total apoio quando precisei, ao meu irmão Alexandre que com seus convites de futebol me faziam agüentar a pressão dos dias.

Agradeço a Waldir Preis, Germano, Aline, Vânia, e Junior (Amador) que sempre foram grandes companheiros em horas que tanto precisava.

Também não posso deixar de agradecer a todos os diretores e funcionários das Lojas Adelino pela compreensão e ajuda, a todos os professores da UNESC que sempre me trataram com muito carinho, em especial ao meu professor Daniel Pezzi da Cunha pela orientação, compreensão e incentivo na elaboração e conclusão deste trabalho.

Em geral agradeço a todos os amigos e companheiros de universidade pelos anos juntos convividos, e pelas dificuldades superadas, anos em que jamais serão apagados de minha memória.

“Nunca, jamais desanimes,

embora venham ventos contrários!”

Santa Paulina

RESUMO

O banco de dados é uma importante estrutura a serviço do desenvolvimento de sistemas em todo o mundo, é nele onde todos os dados ficam armazenados, ele também é responsável pelo controle de fluxo de solicitações de usuários, muitas dessas em grande número. O objetivo principal deste trabalho é buscar formas e ferramentas de testar e analisar o limite de acessos simultâneos em que um banco de dados pode suportar. Para tanto ferramentas de teste de carga em banco de dados foram pesquisadas e testes foram executados.

Palavras Chave: Sistemas de Banco de Dados, Controle de concorrência, Programas de teste de carga em SGBD.

ABSTRACT

The database is all over the world an important structure to service of the development of systems, it is in him where all the data are stored, he is also responsible for the control of flow of users' solicitations, many of those in great number. The objective principal of this work is to look for forms and tools of to test and to analyze the limit of simultaneous accesses in that a database can support. For so much tools of load test in database were researched and tests were executed.

Keywords: Database systems, competition Control, Programs of load test in SGBD.

LISTA DE ILUSTRAÇÕES

LISTA DE TABELAS

LISTA DE SIGLAS

CD	*Compact Disc*
CPF	*Cadastro de pessoa Física*
CPU	*Unidade de Processamento Central*
DBA	*Database Administrator*
DDL	*Data Definition Language*
DML	*Data Manipulation Language*
DVD	*Digital Video Disc*
E/S	*Entrada e Saída*
MTTF	*Mean Time to Failure*
OLAP	*On-Line Analytical Processing*
RAID	*Redundant Arrays of Inexpressive Discs*
SGBD	*Sistema Gerenciamento Banco de Dados*
SQL	*Structured Query Language*
WWW	*World Web Wide*

SUMÁRIO

1 INTRODUÇÃO

Atualmente cada vez mais os sistemas com tecnologia de banco de dados relacional fazem parte do cotidiano das empresas, sistemas baseados em arquivos estão gradativamente desaparecendo do mercado, isso se deve a inúmeros fatores onde podemos destacar alguns como, facilidade de desenvolvimento, oferta de mercado etc. Porem o fato das organizações utilizarem banco de dados, não faz com que seus problemas não existam, quando abordamos itens como acessos simultâneo de usuários, não é rara a área técnica deparar-se com situações em que a dada aplicação deixa de responder de forma adequada após uma nova operação do banco de dados.

Em geral isso ocorre quando os testes feitos no novo sistema não consideram o uso intensivo por vários usuários simultaneamente, nesses casos a equipe de desenvolvimento se limitou às questões funcionais, ignorando um aumento súbito de usuários, o que inevitavelmente vai acontecer em um futuro próximo, essa carência de analise e teste preliminar a implementação do banco, faz com que as corporações trabalhem diariamente com o risco iminente de uma pane por quantidade inesperada de acessos.

O principal problema que podemos encontrar é a falta de informação, se tratando de acessos simultâneos, quando esse problema é diagnosticado deve-se existir soluções rápidas e previamente testadas, devido a isso, esse trabalho tem como objetivo fazer um estudo de desempenho e capacidade de carga de acessos simultâneos a modelos de banco de dados mais utilizados no mercado. Analisando a quantidade e capacidade de acessos simultâneos para cada banco de dados e verificando possíveis soluções em cada caso, para esse estudo serão utilizados servidores de banco de dados, softwares de desempenho e carga, bem como modelos de bancos de dados.

1.1 OBJETIVO GERAL

Realizar um estudo de capacidade de acessos simultâneo em sistemas de banco de dados.

1.2 OBJETIVOS ESPECÍFICOS

Os objetivos específicos desta pesquisa consistem em:

a) fazer um levantamento dos principais fatores que influenciam o desempenho de um banco de dados;

b) analisar o comportamento de um determinado SGBD em acessos simultâneo de usuários;

c) pesquisar utilitários de carga e desempenho adequados para identificar limites da capacidade de acessos;

d) verificar os resultados, analisar o comportamento do ambiente de testes;

1.3 JUSTIFICATIVA

Um dos fatores mais importantes no desempenho de um banco de dados é quanto a sua capacidade de suportar vários acessos simultâneos, haja visto que os problemas gerados por sobrecarga de usuários e um SGBD podem trazer situações de extrema preocupação para o administrador de banco de dados de qualquer empresa. Os danos dessa origem podem ser desde uma lentidão excessiva, até a indisponibilidade de acessos ao SGBD, causando prejuízos e incertezas. Devido a isso, o estudo e a pesquisa

sobre sobrecarga e desempenho em acessos simultâneo em banco de dados são necessários, de modo a prevenirem falhas ocasionadas por demandas mal planejadas, alem de obter respostas sobre as limitações do SGBD.

1.4 ESTRUTURA DO TRABALHO

Este trabalho de pesquisa buscou realizar um estudo sobre a real capacidade de acessos simultâneos a um modelo de banco de dados. Para tal, o mesmo é estruturado em 5 capítulos, aonde o capítulo 2 forma a base teórica dos objetos de estudos enumerando informações sobre Bancos de Dados Relacionais, modelos específicos arquiteturas e suas principais características.

No capítulo 3 são levantadas informações a respeito do tratamento de melhorias em que um SGBD pode passar assim como uma coleção de soluções para os problemas de naturezas oriundas dos acessos simultâneos a um SGBD.

O capítulo 5 aborda todo o processo constituído pelo ambiente de testes, informando as métricas utilizadas.

2 SISTEMAS DE BANCO DE DADOS

Um Sistema Gerenciador de Banco de Dados (SGBD) é um conjunto de programas responsáveis pelo gerenciamento de uma base de dados, conforme Silva (1999), o objetivo principal dos SGBDs é estruturar um ambiente eficiente e conveniente para retirar, armazenar e atualizar as informações de arquivos de banco de dados.

Historicamente o primeiro Sistema Gerenciador de Banco de Dados comercial surgiu no final de 1960 com base nos primitivos sistemas de arquivos (SILVA, 1999) na época esses sistemas controlavam os acessos concorrentes por vários usuários ou processos.

Os SGBDs evoluíram desse sistema de arquivos de armazenamento em disco, criando novas estruturas de dados com o objetivo de armazenar dados. Com o passar do tempo novas formas de representação, ou modelos de dados foram utilizadas para descrever a estrutura das informações contidas em seus bancos de dados.

2.1 CONCEITOS E APLICAÇÕES DE UM SGBD

Os Sistemas de Bancos de Dados surgiram em resposta aos métodos mais antigos de gerenciamento computadorizado de dados comerciais. Os bancos de dados são amplamente utilizados em vários setores do comercio, indústria dentre outros. Porem nas últimas quatro décadas, a utilização dos bancos de dados cresceu em todas as suas áreas de utilização (SILBERSCHATZ; KORTH; SUDARSHAN, 2006). Primeiramente havia pouca interação direta entre os sistemas de banco de dados e os usuários, embora os usuários o faziam sem perceber.

Ocorreu então que os sistemas avançaram e através de máquinas de auto-atendimento, interfaces telefônicas computadorizadas e web sites, permitiram que os usuários lidassem e manipulassem diretamente o banco de dados. A partir dessa mudança os sistemas de banco de dados foram gradativamente melhorando em recursos e serviços, no final da década de 90 houve um aumento significante de acessos direto dos usuários a banco de dados.

As organizações converteram muitos de seus serviços que utilizavam o telefone como principal interface de comunicação, para interfaces web que economiza mão de obra e garantia mais agilidade no atendimento.

Ainda que as interfaces de usuários ocultem cada vez mais os detalhes do acessos a um determinado Banco de Dados e a maioria das pessoas nem mesmo tenha consciência de estar lidando com ele, conforme Silberschatz, Korth e Sudarshan (2006) acessar Banco de Dados é parte essencial da vida de quase todo mundo hoje.

2.2 BANCO DE DADOS RELACIONAIS

O Modelo relacional é uma teoria matemática desenvolvida por Edgar Frank Cold que surgiu na década de 70 (VACHELLI, 2001) para descrever como as bases de dados deveriam funcionar, porém devido às pesquisas fortemente exigidas, só na década de 1980 o modelo começou a entrar no mercado, e a partir da década de 1990 se transformou na tecnologia líder do mercado de Banco de dados.

Esse tipo de banco de dados é baseado no modelo relacional e utiliza um conjunto de tabelas para representar os dados e as relações entre eles (SILBERSCHATZ; KORTH; SUDARSHAN, 2006).

De acordo com a arquitetura ANSI/SPARC, os bancos de dados relacionais possuem três camadas: um conjunto de visões compondo o nível externo; uma coleção de estruturas de dados, em relações compondo o nível conceitual, um conjunto de índices ou métodos de acessos a dados armazenados, compondo o nível interno.

A teoria relacional de banco de dados define um conjunto de operações lógicas utilizando a álgebra e o cálculo relacional, tais operações são à base da linguagem SQL. Segundo Vachelli (2004), o modelo surgiu das seguintes necessidades:

a) aumentar a independência de dados nos sistemas gerenciadores de banco de dados;

b) prover um conjunto de funções apoiados em álgebra relacional para armazenamento e recuperação de dados.

Sendo assim, o modelo relacional revelou ser o mais flexível e adequado ao solucionar os vários problemas que se colocam no nível da concepção e implementação da base de dados. Conforme Oliveira (1999) estrutura fundamental do modelo relacional é a relação tabela, ou seja, linha x coluna. O modelo relacional implementa estruturas de dados organizados em relações, porem para trabalhar com essas tabelas, algumas restrições precisam ser impostas para evitar aspectos indesejáveis como: repetição da informação, incapacidade de representar parte da informação e a perda da informação, essas restrições são: Integridade referencial, chaves e integridade de junções de relações.

2.3 ARQUITETURA

A arquitetura de um sistema de Banco de dados é bastante influenciada pelo sistema de computador que o mesmo esta trabalhando em conjunto (SILBERSCHATZ; KORTH; SUDARSHAN, 2006).

Diante das arquiteturas disponíveis existem sistemas com processamento centralizado, ou ainda no esquema cliente-servidor aonde a tarefa de execução e consulta é bem dividida. As arquiteturas paralelas também são utilizadas no sistema de banco de dados distribuído, abrangendo varias maquinas geograficamente espalhadas, porem vistas como uma só estrutura de processamento.

A partir dessa diversidade de arquiteturas, os ambientes começam a se tornar cada vez mais flexível no ponto de vista de crescimento da informação. Porem todo esse crescimento leva consigo uma série de problemas de desempenho e capacidade, já que segundo Caratti (2006), as implementações e desenvolvimento muitas vezes não levam em consideração as arquiteturas do cliente que será o usuário final, ficando muito restritas as funcionalidades de um certo módulo, e não preparadas para uma utilização em conjunto com outras arquiteturas e até com outros usuários concorrentes. No próximo sub-capitulo explica-se um pouco de cada arquitetura .

2.3.1 RESUMO DAS ARQUITETURAS

Conforme Date (2001), as arquiteturas podem ser classificadas nos seguintes modelos;

a) plataformas centralizadas: na arquitetura centralizada, existe um computador com grande capacidade de processamento, que é o hospedeiro do SGBD e emuladores para as mais diversas aplicações. Essa arquitetura tem como principal vantagem permitir que muitos usuários manipulem grande volume de dados. Sua principal desvantagem é seu custo, devido a utilização de *mainframes* e serviços centralizados;

b) sistemas de computador pessoal (PC): os computadores pessoais trabalham em modo *stand-alone*, ou seja, executam seus processamentos sozinhos. No inicio esse tipo se serviço era bastante limitado, porém com a gradativa evolução das infra-estruturas de hardware têm-se hoje computadores com uma enorme capacidade de processamento. Eles utilizam o padrão XBase e quando se trata de SGBDs, trabalham como hospedeiros e

terminais ao mesmo tempo, fazendo dessa forma que apenas um aplicativo seja executado na maquina. A simplicidade é a sua principal vantagem dessa arquitetura;

c) banco de dados cliente-servidor: na arquitetura cliente-servidor, o cliente executa as tarefas do aplicativo, ou seja, fornece a interface do usuário (tela e processamento de entrada e saída), já o servidor trabalha e executa as consultas e retorna os resultados ao cliente. Apesar de ser uma arquitetura bastante popular, muitas vezes são necessárias soluções sofisticadas de software que possam trabalhar o tratamento das transações.

A divisão do processamento entre dois sistemas é uma de suas principais vantagens fazendo com que seja uma das mais utilizadas.

d) sistemas de banco de dados distribuídos: nessa arquitetura a informação esta dispersa e distribuída em vários servidores, cada servidor atua com as mesmas características que a arquitetura cliente-servidor, porem as consultas oriundas dos aplicativos são feitas para qualquer servidor de uma maneira indistintamente. Caso a informação solicitada seja mantida por outro servidor ou servidores, o sistema encarrega-se de obter a informação necessária, de uma maneira transparente para o aplicativo, que passa a atuar consultando a rede, independente de conhecer seus servidores. Exemplos são as bases de dados corporativas, em que o volume de informação é muito grande e, por isso, deve ser distribuído em diversos servidores. Porém não é dependente de aspectos lógicos de carga de acessos aos dados, ou base de dados fracamente acoplados, em que uma informação solicitada vai sendo coletada numa propagação de consulta numa cadeia de servidores. A principal característica é do esquema de vários aplicativos consultando a

rede para buscar os dados quando necessários, todavia sem o conhecimento explicito de quais servidores dispõem os dados.

2.4. SQL

Quando os bancos de dados relacionais estavam em desenvolvimento, varias linguagens foram criadas destinadas a sua manipulação. A partir daí, o departamento de pesquisas da IBM, desenvolveu a SQL (Strutured Query Language) como forma de interface para o sistema de Banco de Dados relacional chamado *System R*, em meados dos anos 70 .

Em 1986 o Americam National Institute (ANSI), publicou um padrão SQL, desde então a SQL estabeleceu-se como a linguagem padrão de banco de dados, conforme SILBERSCHATZ, KORTH, SUDARSHAN (2006) a linguagem SQL é composta de várias partes:

a) l**inguagem de definição de dados(DDL):** responsável para fornecer comandos para definir esquemas de relação, excluir relações e modificar esquemas, entre eles pode-se citar os comandos : *create table, create index, alter table, drop table, drop view, drop index;*

b) **linguagem de manipulação dos dados (DML):** a DML da SQL inclui uma linguagem de consulta baseada na álgebra relacional, e também no calculo relacional de tupla, suporta também comandos para inserir, excluir e modificar tuplas no banco de dados, entre eles tem-se os seguintes comandos: *insert, update, delete, select;*

c) i**ntegridade**: a DDL da SQL inclui comandos para especificar restrições de integridade as quais os dados armazenados no banco de dados precisam

satisfazer. As atualizações que violam as restrições de integridade são proibidas;

d) **definição de Views:** a DDL SQL inclui comandos para definir views;

e) **comandos para controle de dados**: Servem para gerenciar o acessos dos usuários a determinadas tabelas, e manter a integridade destas. Entre eles pode-se citar: *commit, rollback, grant e revoke.*

f) **SQL embutida, SQL Dinâmica**: a SQL embutida e a dinâmica, definem como as instruções SQL podem ser incorporadas dentro das linguagens de programação como java, c, c++, pl/i, cobol, pascal e fortram.

A linguagem SQL é muito ampla como foi relatado nas linhas anteriores onde possui uma série de operações e funções, desde uma simples consulta, até a adequação para módulos em desenvolvimento. Não é objetivo deste projeto de pesquisa avançar e especificar as funções SQL, visto que para tal, longas paginas seriam necessárias, e dependendo do caso até um trabalho de TCC completo apenas para o tema.

2.5. FINALIDADES DE UM SGBD

Os métodos mais antigos de gerenciamento computadorizado permitiram o surgimento dos atuais Sistemas de Banco de Dados (SILBERSCHATZ; KORTH; SUDARSHAN, 2006).

Os bancos de dados buscam resolver problemas da organização com relação às informações em que métodos antigos não conseguiam, ou tinham muita dificuldade

em resolver. Manter informações organizadas em Sistemas de Banco de Dados apresenta diversos desafios a resolver, como se pode destacar:

a) redundância e inconsistência de dados: onde se busca minimizar a duplicação de informações e também sua inconsistência, ou seja, as várias cópias dos mesmos dados podem não mais concordar;

b) dificuldade de acessos aos dados: onde se busca certo dado com o menor custo possível aliado a maior rapidez;

c) isolamento de dados: quando o objetivo é não deixar os dados em locais de difícil acessos ou localização;

d) problemas de integridade: os dados precisam em muitas vezes de restrições de consistência para poder manter regras externas, entretanto as restrições exigem certo cuidado em seu gerenciamento;

e) problemas de atomicidade: onde o controle do sucesso ou fracasso de uma transação deve ser monitorado, fazendo com que em caso de falha, a informação não sofra alterações;

f) anomalias de acessos concorrente; para fornecer o desempenho geral do sistema, e uma garantia de resposta mais rápida, muitos sistemas permitem que vários usuários atualizem os dados simultaneamente. Nesse tipo de ambiente a interação das atualizações concorrentes é forte, e pode resultar em dados inconsistentes;

g) problemas de segurança: o controle dos acessos de usuários e suas permissões são de fundamental importância para o SGBD.

Essas dificuldades e desafios exigiram o desenvolvimento dos sistemas de bancos de dados. Com o crescimento do uso da informática os SGBD passaram a

atender uma grande demanda de usuários, muitas vezes não esperada pela equipe de desenvolvimento. O aumento dos serviços de comunicação alterou de maneira considerável o perfil dos sistemas e também dos locais de acessos, pois não basta apenas projetar bancos para acessos locais, mais sim estar disposto a atender uma quantidade externa de usuários e solicitações que podem surgir rapidamente.

3. DESEMPENHO DE UM SGBD

Como um dos maiores fatores responsáveis pelo sucesso ou fracasso de um SGBD, o desempenho é um assunto muito discutido e principalmente esperado. A gestão de desempenho agrupa sob a sua denominação diversos aspectos relativos às funções de melhoria dos procedimentos internos do SGBD. Segundo DATE(2000), o desempenho em Sistemas de computadores se baseiam em 4 pilares básicos que são:

a) **otimização de aplicações:** consiste em incrementar o desempenho do código de uma determinada aplicação;

b) **o diagnóstico:** consiste em determinar as causas da degradação de desempenho de determinadas medidas como *"throghput''* e tempo de resposta";

c) **a gerência de recursos:** consiste em agendar a utilização de recursos de forma ótima, através de balanceamento de carga e de ajustes de parâmetros do sistema;

d) **o planejamento da capacidade**: visa em prover níveis aceitáveis de serviço em um horizonte de longo prazo.

O desenvolvimento de uma aplicação envolve inúmeras tarefas. O aspecto de desempenho é um dos fatores de resposta mais esperados, tanto pelos usuários, como

pela equipe de desenvolvimento. De fato, quando um SGBD é usado, é comum descobrir que o mesmo ficou mais lento do que quando foi projetado, ou se trata de menos transação por segundo do que era preciso, ou também que não suporta certa quantidade de acessos simultâneos.

O banco pode correr riscos de insatisfação do usuário e, em casos extremos pode se tornar completamente inutilizável. Os ajustes de desempenho fazem com que as solicitações de usuários possam ser executadas muito mais rapidamente.

Esses ajustes visam localizar e eliminar os gargalos, alem de buscar a melhor performance com a adoção de hardwares específicos e apropriados. São várias as ações que um desenvolvedor pode fazer para ajustar a aplicação e existem também a responsabilidade do administrador de sistemas de banco de dados onde o mesmo pode tomar medidas para agilizar o processamento de uma aplicação.

Uma ferramenta muito útil para a equipe de desenvolvimento são os chamados *benchmarks*, onde são conjuntos padronizados de tarefas que ajudam a caracterizar o desempenho dos sistemas de banco de dados (SILBERSCHATZ; KORTH;SUDARSHAN, 2006).

Sua utilidade é muito importante, pois dão uma idéia aproximada dos requisitos tanto de hardware como de software, fazendo com que o banco possa ser testado em uma simulação de ambiente semelhante a utilização real onde será utilizada, isso até antes mesmo que a aplicação seja criada.

O ajuste de desempenho de um Banco de dados objetiva buscar a melhor performance em vários parâmetros e opções. Os aspectos podem ser diversos, desde alto nível, como esquemas e transações, até buffers e hardwares como memória e velocidade de rede. Cada um desses topicos permite que sua configuração seja alterada a fim de buscar o melhor desempenho com as garantias que os SGBD podem oferecer.

3.1 GARGALOS

As principais limitações de desempenho na maioria dos SGBD são ocasionadas devido a gargalos que limitam ou atrasam os processos de uma dada aplicação devido a vários fatores como, por exemplo, um pequeno looping dentro de um código SQL pode fazer com que uma transação perca um tempo considerável. A melhoria do desempenho de um componente que não é um gargalo tem pouca contribuição na melhora da velocidade geral do sistema. Ao partir em busca de melhor ajuste para o SGBD primeiramente é preciso tentar descobrir o que está causando, e quais são os gargalos, assim que encontrar, a equipe de desenvolvimento deve eliminá-los melhorando o desempenho dos componentes do sistema.

Quando acontece a remoção de um gargalo, dependendo da forma que o mesmo foi removido, o gargalo pode ser transferido para outra parte do sistema, entende-se sistema nesse caso não só os códigos da aplicação, mas também todo processamento computacional, desde o hardware até o usuário.

Os bancos de dados são sistemas complexos e seu controle de gargalos pode ser pelo enfileiramento, as transações solicitam vários serviços ao SGBD, as solicitações de serviços, passam pelo controle de concorrência, onde para cada serviço uma fila esta associada, aonde também pequenas transações podem perder um precioso tempo de suas execuções na espera dessa fila. Para entender melhor suponha que alguém vá até o supermercado e compre 1 litro de leite, e na frente possui uma outra pessoa com carrinho cheio de compras, o que o supermercado deve fazer para resolver esse gargalo? Colocar um novo caixa? E se o mesmo não tiver espaço interno?

As filas de entradas e saídas em disco são uma das grandes responsáveis pelo enfileiramento de pequenas linhas de códigos.

Como resultado, é nesse momento que os gargalos aparecem, na forma de longas filas para um determinado serviço em particular, ou de uma forma equivalente, em grandes utilizações para um serviço em particular (SILBERSCHATZ; KORTH;SUDARSHAN, 2006). Como as solicitações acontecem de uma forma aleatória, fica muito difícil controlar os tempos de cada solicitação, se elas fossem espaçadas de forma uniforme isso seria possível, e também se o tempo para atender uma solicitação fosse menor ou igual ao tempo da próxima solicitação não haveria problemas de filas.

A figura 1 mostra o esquema de filas em um banco de dados, onde se demonstra o gerenciador de controle de concorrência atendendo as solicitações de bloqueio, vê-se também o gerenciador de transações controlando as transações com o auxilio do monitor de transações, as transações fazem as solicitações para acessarem o gerenciador de CPU e o gerenciador de disco.

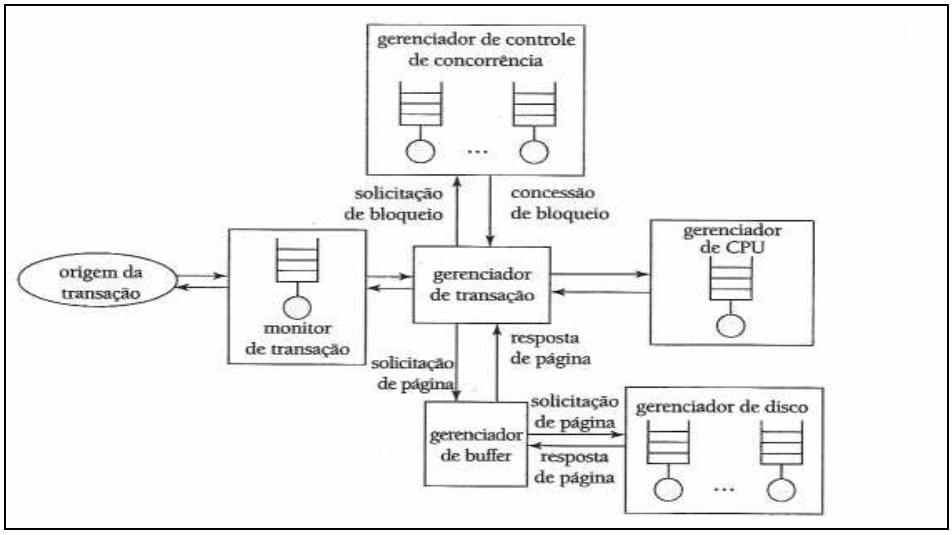


Figura 1 – Filas em um sistema de Banco de Dados
Fonte: Silberschatz, A. Korth, H. Sudarshan, S. (2006, p. 594)

3.2 PARÂMETROS DE AJUSTE

Os níveis de ajuste que os administradores de banco de dados podem implementar são os 3 seguintes:

a) nível de hardware: as opções de ajuste de desempenho no nível de hardware são ações de modificação em sua configuração como o aumento de discos, ou usar os serviços de RAID caso a E/S de disco seja um possível gargalo, a expansão de memória, nesse caso o buffer seria o gargalo a ser eliminado, e até mesmo o processador poderia ser substituído se os administradores verificassem uma insuficiência de processamento do mesmo. No que se refere acessos simultâneo, os equipamentos de rede deveriam ser analisados também, já que são as vias de acessos de comunicação externa;

b) parâmetros do sistema de banco de dados: são os intervalos de pontos de verificação e o tamanho do *buffer*,, cada SGBD possui um conjunto exato de parâmetros, isso é especifico de cada modelo, como o SQL Server, Interbase dentre outros. Os modelos de mercado oferecem uma grande documentação sobre esses ajustes, onde cabe ao DBA verificar a melhor configuração do sistema. Mas sistemas de banco de dados bem projetados realizam essa tarefa automaticamente, buscando sempre o melhor desempenho e cada vez mais ajudando a suportar cargas de acessos;

c) esquema de transações: o projeto do esquema pode ser ajustado pelo administrador, os índices e as transações executadas servem para buscar a melhora do desempenho, nesse nível é comparativamente independente do banco.

Um fator importante a ser lembrado é que os três níveis de ajuste interagem entre sí, onde uma ação efetuada em um nível pode desencadear um gargalo em outro,

cabe ao administrador do banco de dados estar sempre atento às alterações para que não tenha surpresas desagradáveis.

3.3 AJUSTES DE HARDWARE

As operações de E/S de disco também podem ter um grande volume, até em SGBD bem projetados, caso as informações exigidas estiverem em disco. Em analogia pode-se citar os acessos simultâneos para o cadastramento de CPF nos servidores da receita federal, no momento da confirmação do recadastramento isso seria muito exigido. Um fator importante no ajuste de um sistema de processamento de transação é certificar-se de que o subsistema de disco pode lidar com a taxa em que as operações de E/S são exigidas.

Por exemplo: Um disco que admita a um tempo de acessos de cerca de 10 milisegundos, em uma taxa de transferência média de 25 megabytes por segundo (muito comum nos discos da atualidade), dessa forma quando a página for acessada n vezes por segundo, a economia necessária para mantê-la na memória é n vezes esse valor, onde armazenar uma pagina na memória custa: (preço por megabyte de memória*)* dividido por (paginas por megabyte de memória)

Sendo assim, segundo Date(2000) o ponto de equilíbrio seria a seguinte fórmula :

$$n * \frac{\textit{(preço por unidade de disco)}}{\textit{(acessos por segundo por disco)}} = \frac{\textit{(preço por megabyte de memória)}}{\textit{(paginas por megabyte de memória)}}$$

A regra pratica dos 5 minutos nos diz que se uma consulta é usada com mais freqüência do que uma vez em cada 5 minutos, ela deve ser colocada em cache de

memória, a fórmula para o ponto depende muito de fatores como custo de discos e de memória.

Outro aspecto importante de ajuste é sobre a utilização da tecnologia do RAID, onde qual modelo deverá ser adotado? RAID 1 ou RAID 5, a freqüência de acessos é que vai levar para a melhor opção, não se pode esquecer que RAID 5 é mais lento que RAID 1. (SILBERSCHATZ; KORTH;SUDARSHAN) (2006).

3.4 MELHORANDO ÍNDICES

Quando se tem consultas como gargalos, os índices podem passar por ajustes a fim de ganhar a melhora do desempenho, criando índices apropriados sobre as relações. Já se o gargalo for às atualizações, o problema nesse caso é a quantidade de índices, removendo-os, as relações serão atualizadas com mais agilidade, a escolha do tipo de índice também é muito importante, alguns modelos de SGBD admitem diferentes formas de índices, como índice de *Hash* e índices de arvore B, para um numero expressivo de consultas, os índices de arvore B, são sugeridos.

Para auxiliar na criação de índices e também na sua identificação, os assistentes de ajuste dos SGBD servem para ajudar o DBA, utilizando-se de históricos de consultas passadas e atualizações, as recomendações sobre quais índices criar são baseadas em estimativas nesses históricos.

Quando se cita manutenção de desempenho em visões, conforme Date 2000, as visões materializadas podem dar bastante agilidade a certos modelos de consultas, em geral de consultas de agregação. Suponha que em dado momento o sistema requisita com freqüência a quantidade total de empréstimos em uma agência (Obtida pela soma das quantias de empréstimos de todos os empréstimos da agência), a criação de uma

visão materializada armazenando a quantidade total do empréstimo pode agilizar as consultas. Porem as visões materializadas devem ser utilizadas com certa cautela, pois alem da sobrecarga de espaço existe também uma sobrecarga de tempo para manter essas visões. Uma importante questão a ser analisada é qual visão adotar, a manutenção de visão imediata ou a manutenção de visão adiada, a duvida é quais visões materializadas devem ser mantidas..

O administrador do sistema pode fazer a seleção manualmente examinando os tipos de consultas, e descobrindo quais consultas precisam ser executadas mais rapidamente, e quais atualizações podem ser executadas mais lentamente. O método manual de escolha é cansativo porem compensador.

3.5 CONTROLE DE CONCORRÊNCIA

A partir do momento em que varias transações são executadas ao mesmo tempo no banco de dados, as propriedades de isolamento podem não ser mais preservadas, para garantir essa preservação o sistema precisa controlar as transações simultâneas, uma serie de mecanismos são utilizados para efetuar esse controle. Esses mecanismos são chamados de controle de concorrência. Pergunta-se: Porque o controle de concorrência é necessário? A resposta é que muitos problemas podem ocorrer quando as transações concorrentes são executadas sem controle, suponha um sistema de venda de medicamentos on-line onde vários pacientes solicitam seus remédios ao mesmo tempo, um pedido duplicado poderia além de trocar os remédios, ou até mesmo trocar os endereços de entrega. Pode-se levantar os seguintes problemas relatados por Elmasri, Navathe(2002), gerados pela concorrência entra translações:

a) **o problema da perda de atualização:** quando as transações que acessam os mesmos itens do banco de dados tem suas operações entrelaçadas, de tal forma que o valor de algum item seja erradamente informado;

b) **o problema da atualização temporal:** acontece quando uma transação atualiza um item do banco de dados e por algum motivo ocorre uma falha;

c) **o problema de agregação incorreta:** quando um registro esteja sendo disputado entre uma função de agregação e uma atualização, em dado momento a função pode calcular alguns valores antes que eles sejam atualizados, e outros depois que tenham sido atualizados.

Um último problema que pode ocorrer é quando uma transação x lê o mesmo item duas vezes e o item é alterado por uma outra transação entre as duas leituras da transação x.

3.6 A IMPORTÂNCIA DA RECUPERAÇÃO

Toda vez que as transações são enviadas a um SGBD para sua execução, a responsabilidade do SGBD é verificar tudo que ocorre em perfeitas condições, e seu efeito é registrado perfeitamente no banco de dados, ou a transação não tem efeito e nada é gravado.

O SGBD mantém o controle de permissões entre quais transações podem ser executadas antes que outra seja. Caso isso não aconteça, o motivo é uma falha de transação após executar algumas operações, têm-se então alguns tipos de falhas que geralmente são classificadas como falhas de transação do sistema e de mídia. Existem diversas razões possíveis para que uma transação falhe no meio da execução:

a) uma falha do computador: o problema pode ser de hardware ou software;

b) um erro de transação ou de sistema: algum fator na transação pode fazer com que ela falhe, como por exemplo, o excesso de dígitos *(overflow)*, também considera-se que o usuário pode interromper a transação durante a sua execução;

c) erros locais: condições de execução detectadas pela transação, podem ocorrer certas condições que necessitem do cancelamento da transação;

d) imposição do controle de concorrência: o método de controle de concorrência pode decidir abortar a transação;

e) falha de disco: erros de leitura e gravação como a perda de dados devido a disfunções nos blocos;

f) problemas físicos e catástrofes: problemas das mais diversas naturezas, desde uma pane elétrica até um incêndio.

Para assegurar que as transações possam ser executadas sem problemas são utilizados diversos esquemas de controle de concorrência, todos esses esquemas adiam uma operação ou abortam a transação que emitiu a operação.

3.7 PROTOCOLOS BASEADOS EM BLOQUEIOS

Uma forma garantir um bom trabalho das transações, é exigir que os itens de dados sejam acessados de uma maneira mutuamente exclusiva, fazendo que quando uma transação estiver trabalhando com um item de dado nenhuma outra transação possa alterar esse item.

A maneira mais comumente usada é fazer com que a transação efetue um bloqueio sobre o dado para que execute os serviços necessários. Existem diversas formas de como um item de dados pode ser bloqueado, nesse trabalho abordam-se dois modos, o modo de compartilhado e o modo exclusivo.

O modo compartilhado é quando uma transação acessa um item de dado com acessos apenas para leitura, inclusive permitindo que outras transações façam o mesmo, e o modo exclusivo é quando uma transação obtém prioridade e exclusividade sobre um item de dados, ela tem total acessos a ele podendo ler e escrever, fechando o acessos para outras transações.

É fundamental que cada transação solicite um bloqueio em um modo adequado sobre o dado. A transação só poderá prosseguir com a operação depois que receber a permissão do SGBD. O modo de compartilhamento é compatível apenas com o próprio modo compartilhado, e não como o modo exclusivo, isso quer dizer, quando uma transação solicita o modo compartilhado, outras transações podem simultaneamente buscar o mesmo item de dado, neste caso pode existir vários bloqueios em modo de compartilhamento trabalhando ao mesmo tempo.

Já as solicitações de bloqueio exclusivo que estiverem na fila esperando para operar com o mesmo item de dados, precisam aguardar até que os bloqueios do modo compartilhamento sejam liberados. A instrução executada para buscar o bloqueio compartilhado sobre o item de dados é a instrução *lock-s,* de forma semelhante para o bloqueio exclusivo tem-se a instrução *lock-x*, a instrução *unlock* desbloqueia o item de dado após os bloqueios de cada transação.

3.8 CONCESSÃO DE BLOQUEIOS

Quando ocorre uma solicitação de um bloqueio sobre um item de dados em determinado modo, e em nenhum momento outra transação solicita o mesmo item não gerando um conflito, aí então o bloqueio concedido.

Porem em algumas situações deve-se ter certo cuidado a fim de evitar uma longa espera de transações, quando na seguinte situação, em uma determinada aplicação, um item de dados precisa ser modificado por vários usuários diariamente, em dado momento uma transação solicita um bloqueio no modo compartilhado, logo após o gerente operacional da empresa entra com uma transação com solicitação de bloqueio exclusivo, nesse intervalo de tempo outra transação solicita um bloqueio compartilhado, essa cadeia de solicitações de bloqueios compartilhados pode fazer com que o gerente operacional, que foi o originário da transação que esta solicitando o bloqueio em modo exclusivo entre em estado de espera tão grande que seja considerado estagnado.

O SGBD pode evitar que isso aconteça tomando as seguintes providencias:

a) que não haja outra transação que manterá um conflito de bloqueio com o ultimo solicitante;

b) que não exista outra transação que esteja esperando por um bloqueio sobre o item de dado e que fez sua solicitação antes da segunda solicitação.

Dessa forma uma solicitação de bloqueio nunca será bloqueada ou estagnada por uma transação que mais tarde solicitou outro bloqueio.

3.9 PROTOCOLO DE BLOQUEIO EM DUAS FASES

Conforme Elmasri, Navathe(2002), uma transação segue o protocolo de bloqueio em duas fases se todas as operações de bloqueio precedem a primeira operação de desbloqueio na transação. Podendo essa transação ser dividida em duas fases, a fase do crescimento onde uma transação pode obter bloqueios, mais não pode liberar qualquer bloqueio.

Na fase de encolhimento uma transação pode liberar bloqueios, mais não pode obter novos bloqueios. O controle dessa lógica tende as transações operarem dessa forma: quando uma transação esta na fase de crescimento, a transação adquire bloqueios conforme sua necessidade, quando ela libera um bloqueio ela entra na fase do encolhimento, não podendo mais emitir bloqueios.

Existem ainda modificações no protocolo de bloqueio de duas fases que trabalham com bloqueio estrito em duas fases. Esse protocolo requer que todos os bloqueios no modo exclusivo realizados por uma transação sejam mantidos até que essa transação seja confirmada. Essa tarefa visa garantir que qualquer item de dado escrito por uma transação não confirmada sejam bloqueado no modo exclusivo, até que a transação seja confirmada.

Outro modelo de protocolo de bloqueio em duas fases é o protocolo de bloqueio rigoroso em duas fases, que ordena e exige em suas regras que todos os bloqueios sejam mantidos até que a transação seja confirmada, sendo assim pode-se verificar que com o bloqueio rigoroso em duas fases, as transações podem ser seriadas na ordem em que são confirmadas. Os sistemas de banco de dados comerciais usam de forma muito freqüente o bloqueio rigoroso em duas fases.

3.10 PROTOCOLO BASEADO EM *TIMESTAMP*

Uma outra forma de determinar a ordem é selecionar com antecedência uma ordenação entre as transições. O método de ordenação por *timestamp* é o mais adequado para fazer isso, associa-se um valo*r* fixo e exclusivo, esse *valor* é atribuído pelo sistema de banco de dados antes que a transação inicie sua execução. Se uma transação tiver recebido o *timestamp* ts(x) e uma nova transação ts(y) entrar no sistema, então ts(x) < ts(y). Para resolver isso dois métodos existentes são propostos.

O primeiro é usar o valor do clock do sistema como *timestamp*, ou seja o *timestamp* de uma transação é igual ao valor do clock quando a transação entra no sistema . O outro método é usar o contador lógico que é incrementado após um novo *timestamp* ter sido atribuído, ou seja o *timestamp* de uma transação é igual ao valor do contador quando a transação entra no sistema.

Tem-se ainda formas de ordenação por *timestamp* onde existe a garantia que quaisquer operação read e write em conflito sejam executadas em ordem de *timestamp.* Porem demonstrar a operação de tais protocolos não é o foco deste trabalho haja visto que o assunto complementaria uma vasta documentação especial apenas para isso .

3.11 PROTOCOLO BASEADO EM VALIDAÇÃO

Em duas transições que buscam o bloqueio de transações somente leitura, a quantidade de conflitos tende a ser baixa, mesmo assim muitas dessas transações, se executadas sem a supervisão de um esquema de controle de concorrência, deixariam o sistema em um estado de inconsistência.

Um esquema pode gerar uma concorrência impondo sobrecarga de execução de código gerando assim um possível atraso as transações. Pode ser melhor usar um

esquema alternativo que impõe menos overhead, pois não pode-se saber com antecedência quais as transações estarão envolvidas em um conflito, fica difícil ter uma redução de *overhead*. Um esquema de monitoramento precisa ser usado para obter esse conhecimento, ou seja, um esquema de validação é um método apropriado quando a grande maioria das transações são transações somente leitura. Como são somente leitura a taxa de conflitos entre essas duas transações é baixa. Um *timestamp* fixo exclusivo é associado a cada transação no sistema. O *timestamp* é quem determina a ordem da seriação, nesse esquema uma seriação nunca é adiada, porem ela precisa passar por um teste de validação para ser concluída. Caso o resultado do teste seja negativo o estado atual é novamente carregado pelo sistema.

Diversas outras técnicas existentes auxiliam o controle da concorrência, entre elas pode citar, a granularidade multiplica bloqueios de intenção, protocolos de bloqueios por granularidade múltipla, como também técnicas especiais de controle de concorrência podem ser desenvolvidas para estruturas de dados especiais na busca de diminuir a concorrência, aumentando o desempenho das transações.

4 ACESSOS SIMULTÂNEO

Conforme a abordagem deste trabalho de pesquisa, segundo Caratti(2006), um dos fatores mais críticos na utilização de um ambiente de que envolve a utilização de banco de dados diz respeito ao seu desempenho, a alteração ou inclusão de uma, ou mais consultas em um sistema legado, a implantação de um novo módulo, ou o crescimento inesperado de usuários, são fatores que podem comprometer o desempenho de um SGBD.

Quando se refere à alteração de um sistema legado, não é difícil o departamento de desenvolvimento encontrar situações em que uma determinada aplicação deixa de responder de forma adequada após uma alteração ou inclusão de uma nova operação de Banco de Dados.

O maior problema que nota-se nesses casos é de origem de planejamento, onde o desenvolvimento não se preocupou, ou não dispunha de ferramentas e conhecimento para testes preliminares de carga e desempenho. Já em relação a implementação de um novo sistema, é normal que a equipe de desenvolvimento seja questionada quanto a capacidade do servidor de banco de dados, suportar ou não a carga de acessos de seu futuro cliente. Se tratando disso é comum encontrar respostas vagas quanto a métodos e respostas de testes em Bancos de Dados.

No capitulo a seguir demonstra-se o resultado de uma pequena pesquisa de campo onde se busca visualizar um pouco do gerenciamento de problemas dessa natureza.

4.1. PESQUISA DE CAMPO

Para melhor compreender a importância deste trabalho de pesquisa, uma pesquisa de campo em empresas de desenvolvimento da região de Criciúma e Ararangua foi executada entre os dias 15/11/2007 a 20/11/2007. Um conjunto de quatro perguntas foi exposto a desenvolvedores ou gerentes de desenvolvimento, em um total de 4 empresas sólidas, e com Bancos de Dados em pleno funcionamento no mercado local, segue as perguntas com as opções de resposta:

a)Quanto tempo o seu Produto (Sistema, Aplicação) está no mercado?

Respostas Disponíveis: 1 ano, 2 anos, 3 anos, 4 anos, 5 anos, 6 anos, 7 anos, mais de 7 anos;

b) Sua empresa possui rotinas de testes de carga em Banco de Dados?

Respostas Disponíveis: sim, não;

c) Seus produtos já passaram por problemas de acessos simultâneo ?

Respostas Disponíveis: sim, não;

d) Selecione nas opções abaixo o que você acha de um estudo de soluções e ferramentas para trabalhar com o problema do desempenho e acessos simultâneo?

Respostas Disponíveis: importante, necessário, menos importante, não é importante.

4.1.1 RESULTADO DA PESQUISA.

A pesquisa retornou os seguintes resultados conforme a seqüências de tabelas abaixo, os nomes das empresas foram substituídos por números a fim de preservar as informações e os formulários originais juntamente com as assinaturas dos responsáveis se encontram em poder do autor desse trabalho de pesquisa.

Tabela 1: Questionário de respostas empresa A

Cidade Empresa: Turvo Data Pesquisa: 16/11/2007	**Contato: Fabio**
Pergunta	**Resposta**
a) Quanto tempo o seu produto (Sistema) está no mercado ?	4 anos
b) Sua empresa possui rotinas de testes de carga em Banco de Dados?	Não
c) Seus produtos já passaram por problemas de acessos simultâneo?	Sim
d) Selecione nas caixas abaixo o que você acha de um estudo de soluções e ferramentas para trabalhar com o problema de acessos simultâneo e desempenho em Banco de Dados?	Necessário

Tabela 2: Questionário de Respostas empresa B

Cidade Empresa: Tubarão Data Pesquisa : 16/11/2007	Contato: Roberto
Pergunta	**Resposta**
a) Quanto tempo o seu produto (Sistema) está no mercado?	5 anos
b) Sua empresa possui rotinas de testes de carga em Banco de Dados?	Sim
c) Seus produtos já passaram por problemas de acessos simultâneo?	Sim
d) Selecione nas caixas abaixo o que você acha de um estudo de soluções e ferramentas para trabalhar com o problema de acessos simultâneo e desempenho em Banco de Dados?	Importante

Tabela 3: Questionário de Respostas empresa C

Cidade Empresa: Turvo Data Pesquisa: 16/11/2007	Contato: Helder
Pergunta	**Resposta**
a) Quanto tempo o seu produto (Sistema) está no mercado?	4 anos
b) Sua empresa possui rotinas de testes de carga em Banco de Dados?	Não
c) Seus produtos já passaram por problemas de acessos simultâneo?	Sim
d) Selecione nas caixas abaixo o que você acha de um estudo de soluções e ferramentas para trabalhar com o problema de acessos simultâneo e desempenho em Banco de Dados?	Necessário

Tabela 4: Questionário de Respostas empresa D

Cidade Empresa: Criciuma Data Pesquisa: 19/11/2007	Contato: Tiago
Pergunta	**Resposta**
a) Quanto tempo o seu produto (Sistema) está no mercado?	Mais de 7 anos
b) Sua empresa possui rotinas de testes de carga em Banco de Dados?	Não
c) Seus produtos já passaram por problemas de acessos simultâneo?	Sim
d) Selecione nas caixas abaixo o que você acha de um estudo de soluções e ferramentas para trabalhar com o problema de acessos simultâneo e desempenho em Banco de Dados?	Necessário

4.1.2 GRÁFICO DOS RESULTADOS

Das empresas visitadas, gráficos foram gerados com a finalidade de visualizar o panorama de como elas tratam do problema de acessos simultâneo.

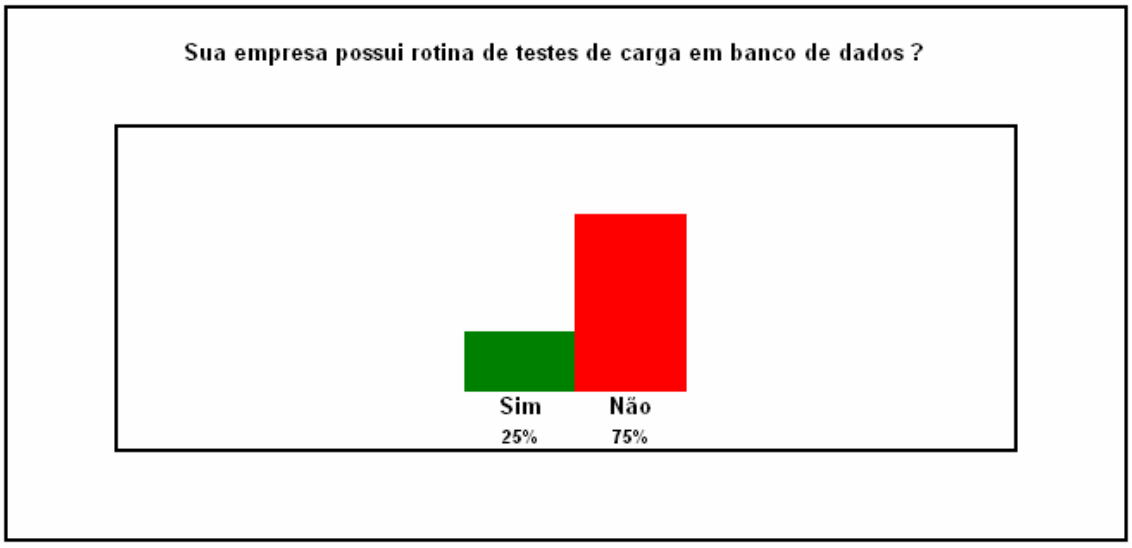


Figura 2. Gráfico da pesquisa de rotina de testes.

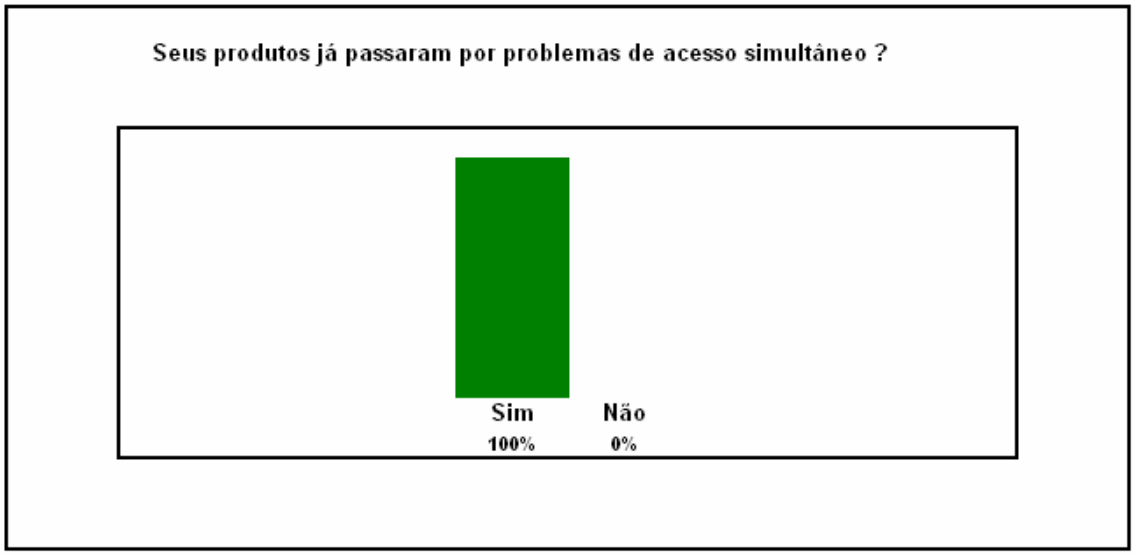


Figura 3. Gráfico da pesquisa de problemas de acessos simultâneo.

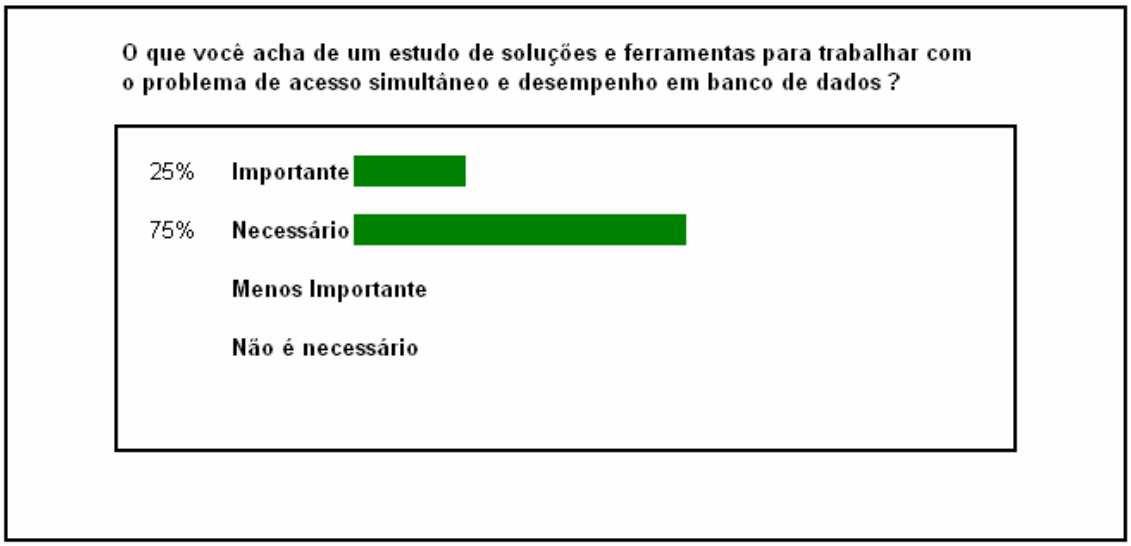


Figura 4. Gráfico da pesquisa sobre o estudo de soluções.

4.2 SOFTWARES DE ANALISE E DESEMPENHO DE CARGA

Apesar de não estar ainda muito difundido entre os desenvolvedores (SILBERCHATZ, KORTH, SUDARSCHAN 2006), no mercado é possível encontrar alguns fornecedores que de alguma forma podem auxiliar o administrador de banco de dados na execução de testes de capacitação, como anteriormente citado muitas empresas de desenvolvimento não dispunham de um grande numero de estações para executar testes.

Uma solução interna poderia ser o desenvolvimento de módulos ou de aplicativos específicos para esse fim. Porem nem todas as empresas podem dispensar tempo e profissionais para esse objetivo, mesmo sabendo de sua importância, as necessidades do dia a dia e o atendimento aos clientes falam mais alto nesse momento.

Diante dessa realidade, poder ter acessos a produtos destinados para essa finalidade é um ganho de um tempo considerável, após buscar algumas ferramentas no mercado tem-se alguns aplicativos interessantes aonde se pode citar: a Quest

Software(www.quest.com), Bmc Software (www.bmc.com), a CA tem o Unicenter NeuMICS Resource Management Capacity Planner Option (www.ca.com), o OpenSTA (www.opensta.org) e a Apache (www.apache.org).

É bom ressaltar que as ferramentas citadas aqui não têm necessariamente as mesmas Funcionalidades, isto é, algumas são bem mais abrangentes que outras e merecem um estudo detalhado para a adequação de suas necessidades.

Também não é objetivo desse trabalho de pesquisa fazer uma comparação entre elas, já que isso seria muito difícil dado o foco que cada uma possui, dentre os fornecedores citados para esse estudo será apresentado o JMETER, da Apache (www.apache.org) como a ferramenta padrão para a execução dos testes de carga e desempenho aqui realizados.

4.3 APACHE JMETER

O JMETER é uma aplicação implementada em Java projetada para executar testes de carga de aplicações Cliente/Servidor. Foi originalmente projetada para realizar testes em aplicações web, mas logo foi expandido para fazer outros tipos de testes tais como em : servidores de bancos de dados (via JDBC), servidores de FTP, JNDI, LDAP e Web Services. 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 a escalabilidade.

Nesse contexto entende-se por escalabilidade a capacidade de resposta de sistema em relação a demanda de recursos exigidos dele. Aumentar a escalabilidade de um sistema de Banco de Dados consiste em expandir a sua capacidade de processamento a armazenamento de registros, seja melhorando a capacidade do

hardware ou otimizando o uso dos recursos do sistema. O JMETER é uma ferramenta "Open Source" distribuído sobre a licença ASF, seu download pode ser feito diretamente pelo site da Apache (http://jakarta.apache.org) (CARATTI 2006).

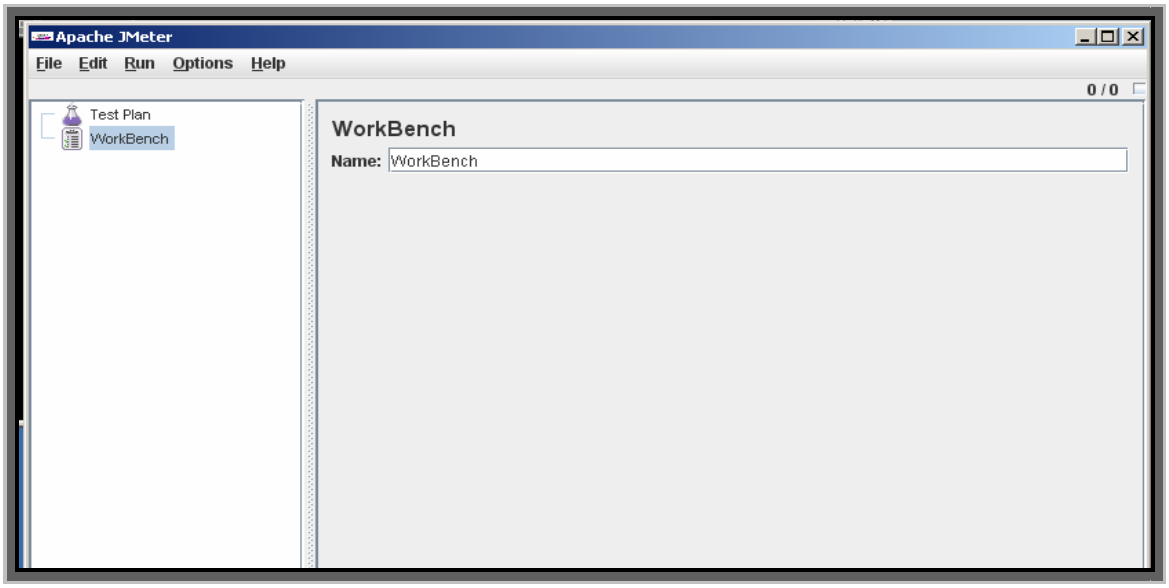


Figura 5: Aplicativo da Apache JMETER

O JMETER é um aplicativo fácil de manipular, principalmente porque divide em testes separados o que o usuário necessita, tem-se, por exemplo, um *Tread Group* que em sua tela é possível informar a quantidade de usuários em que se deseja simular e o comportamento desses usuários quanto a utilização de requisições conforme mostra figura 6, seus principais campos de configuração são os seguintes conforme tabela 5.

Tabela 5: indicadores de configuração de um Thread Group

Campo	Descrição
Name	Nome do Thread Group
Number of Threads	Cada Thread considera-se um usuário que pode executar as requisições SQL
Ramp Up Period	Diz ao JMETER quantos segundos ele tem pra colocar todas as Threads em execução
Loop Count	Numero de vezes que o Thread Group ira executar

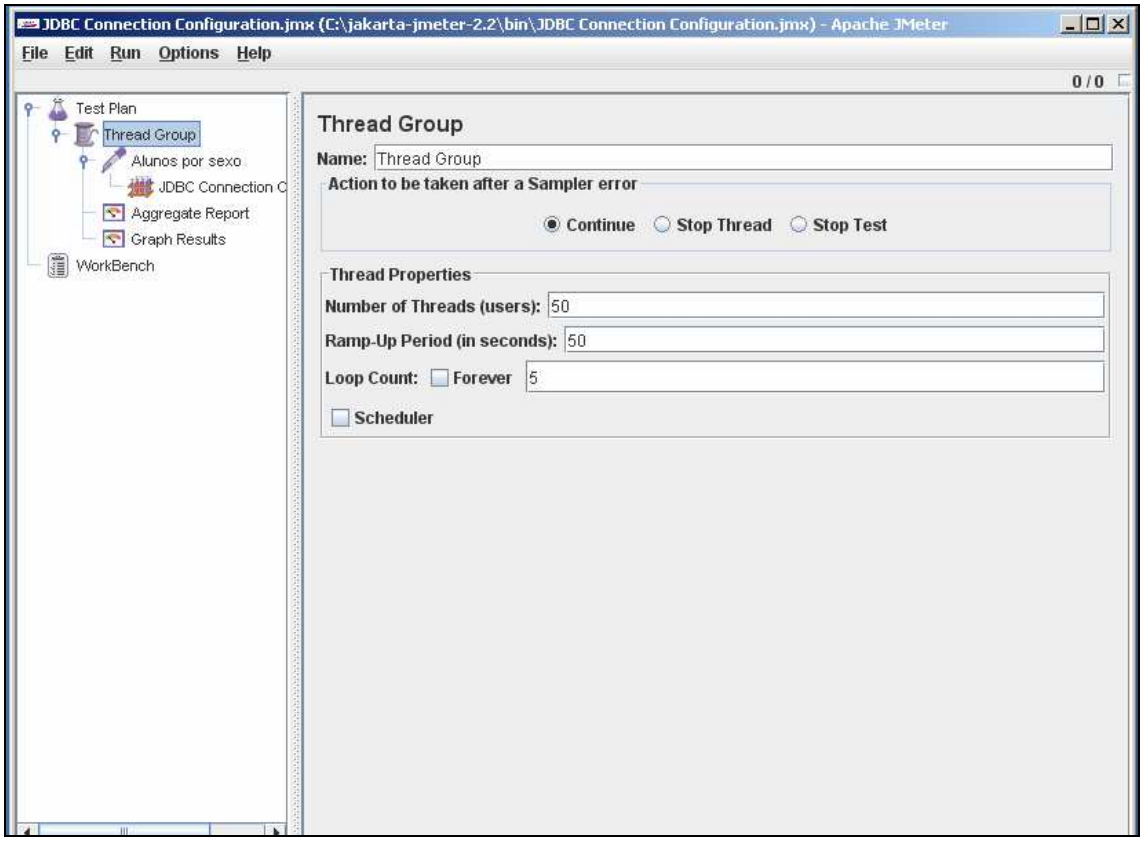


Figura 6: Tela JMETER Thread Group

A etapa de conexão do JMETER com Banco de Dados que será levado ao teste de carga também precisa de atenção, para isso o JMETER monta um plano de configuração chamado de JDBC Connection Configuration aonde nesse espaço todas as informações sobre a conexão são disponibilizadas, conforme mostra figura 7, note que na parte esquerda do software todo o plano de configuração de teste é separado por serviços, o que facilita muito o entendimento de cada serviço criado.

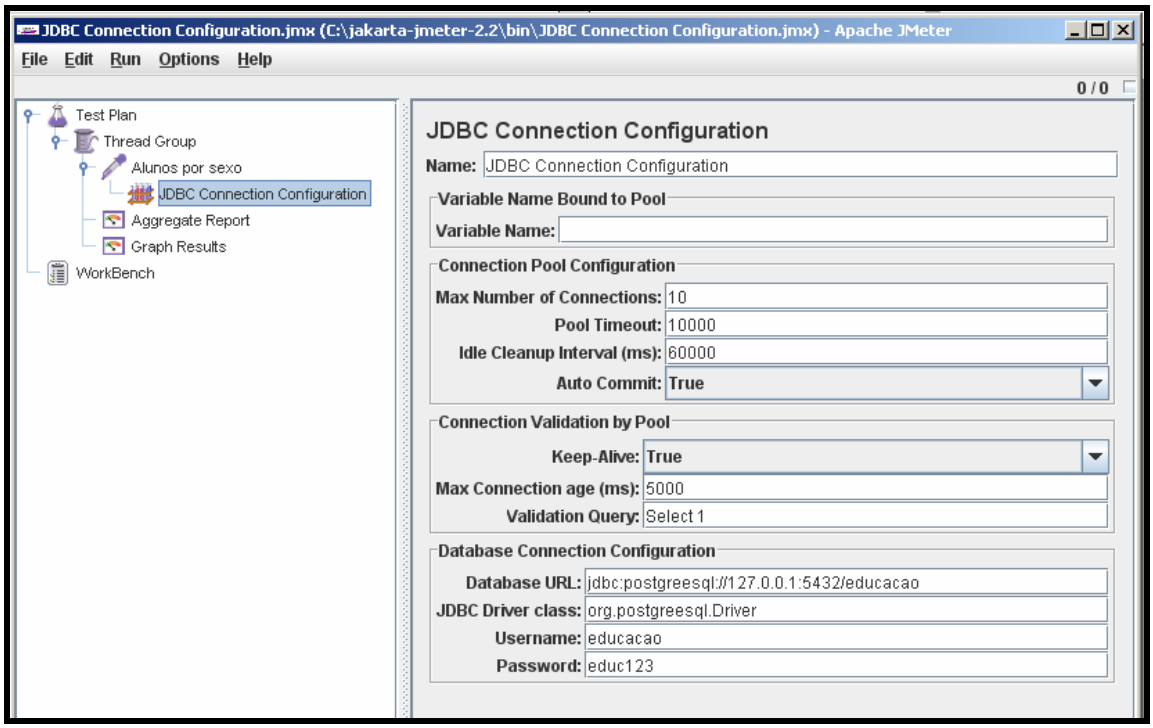


Figura 7: Tela JMETER JDBC Connection Configuration

Como operações SQL serão testadas no banco, o JMETER precisa disponibilizar para o usuário um local aonde esses scripts possam ser inseridos e manipulados, o nome desse local se chama JDBC Request, aonde esta disponível um espaço para as Querys testadas, segundo mostra a figura 8.

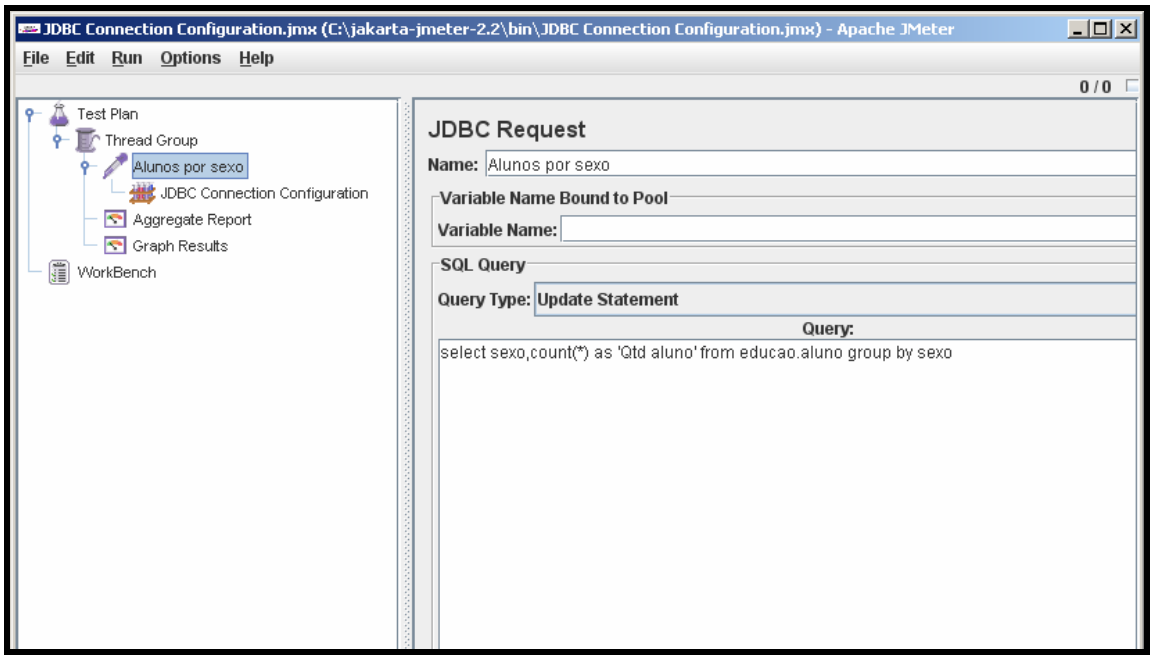


Figura 8: Tela JMETER JDBC Request

4.4 PREPARANDO O AMBIENTE DE TESTES

Para poder utilizar o JMETER em um teste de stress no Banco de Dados, é necessário a instalação do driver JDBC implementado para seu SGBD. A maioria dos grandes fornecedores de SGBD fornece o seu próprio driver JDBC.Somente para citar alguns: IBM DB2, ORACLE, SQL Server, Sybase, Interbase, PostgreSQL, Firebird e MySQL.

Nesse trabalho de pesquisa será utilizado o JDBC para PostgreSQL, já que será em PostgreSQL o Banco de Dados "cobaia" que será manipulado e levado a carga máxima de stress. A infra estrutura de hardware está composta com dois computadores um para cada grupo de testes, tem-se na **configuração 1** : Computador Padrão IBM-PC, Processador Intel Pentium III 750 Mhz, 1 Gb Memória Ram, HD com capacidade de 20 Gb, e na **configuração 2** : Computador Padrão IBM-PC, Processador Intel Celeron 1.6 Ghz, 512 Gb Memória Ram, HD com capacidade de 20 Gb

4.5 DESCRIÇÃO DOS TESTES

Para o plano de testes foram impostas varias métricas em um estudo preliminar, nunca é uma tarefa fácil buscar uma métrica que atenda a quase todas as necessidades, devido que cada desenvolvedor pode possuir um foco e um desafio diferenciado. Contudo o objetivo principal deste trabalho de pesquisa é realizar um estudo da capacidade de acessos simultâneo que um Banco de Dados pode suportar, buscando a partir de uma serie de métodos e ferramentas o limite de seus recursos, sejam eles de software ou de hardware.

A realização dos testes também deverá servir como base de estudo para buscar fatores que podem influenciar no desempenho de um servidor de Banco de Dados, neste caso busca-se fatores externos ao Banco, como problema de limitações de hardware, influencia de softwares ou até de usuários.

Para realização dos testes como já foi relatado será utilizado o JMETER, o JMETER terá como missão aplicar simular carga em um Banco de Dados PostgreSQL previamente preparado com 5 tabelas referentes a um cadastro de alunos, para isso separamos os testes em duas baterias distintas:

a)teste de carga de usuários consulta(T01): o teste consiste em simular o envio de requisições ao Banco de Dados, essas requisições serão de inserção e consulta de sexo em um cadastro de aluno.

Sintaxe SQL do teste de consulta:

select sexo, count(*) as "qtd aluno" from educação. aluno group by sexo;

Sintaxe SQL do teste de inserção:

insert into aluno (nis, nome, sexo, endereco, municipio, fone, filiacao, datanasc) values (null, 'teste', 'm', null, 'sobral', null, null, null);

Configuração Utilizada: 1

Objetivo do Teste: Buscar o limite de acessos aceitos, esse limite pode ser de estafa de software ou de hardware.

b)teste de desempenho(T02): nesse teste o servidor de Banco de Dados será levado a uma simulação de carga utilizando consulta, porem dessa vez um agente de dificuldade foi imposto : Execução de um desfragmentador no

servidor de banco de dados, essa dificuldade foi imposta a fim de retratar um processo iniciado involuntariamente para um servidor.

Sintaxe SQL da consulta:

select sexo, count(*) as "qtd aluno" from educação. aluno group by sexo

Configuração Utilizada: 1

Objetivo do Teste: buscar fatores que comprometem o desempenho do Banco de Dados.

c)teste de desempenho de hardware(T03): Nesse modelo de teste**, a** alteração da capacidade de memória será analisada, a fim de diagnosticar se o aumento da capacidade de memória realmente leva consigo o aumento do indicador de vazão *(Throughputs),* o teste consiste em simular 30 usuários executando uma consulta em um servidor com capacidade inicial de 512 Mb de memória Ram, depois o mesmo teste será executado com respectivamente 1 Gb e 2 Gb de memória.

Sintaxe SQL da consulta:

select sexo, count(*) as "qtd aluno" from educação. aluno group by sexo

Configuração Utilizada: 2

Objetivo do Teste: Verificar o ganho de vazão (Throughputs) em upgrade de memória.

d)teste de desempenho com concorrência de Software(T04): nesse teste o servidor de Banco de Dados será levado a uma simulação de carga de 30 usuários executando uma consulta, porem dessa vez alguns serviços concorrentes de recursos serão executados em paralelo com o Banco de

Dados, os serviços são os seguintes : execução varredura de antivírus, execução Backup completo do disco rígido, Atualização padrão de software antivírus. Esses serviços concorrentes visam simular um ambiente normal de utilização de um servidor de Banco de Dados em escritórios e em pequenas empresas.

Sintaxe SQL da consulta:

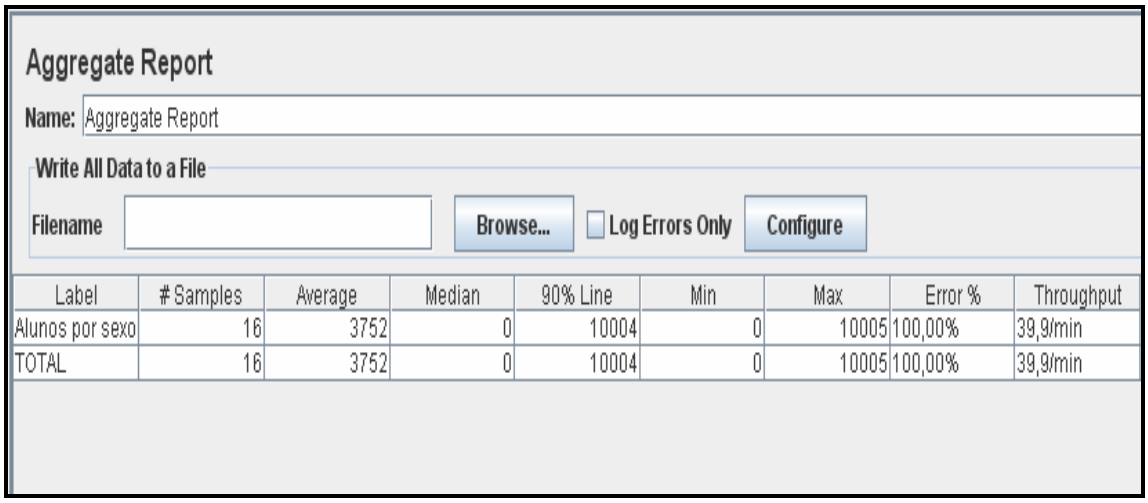
select sexo, count(*) as "qtd aluno" from educação. aluno group by sexo

Configuração Utilizada: 1

Objetivo do Teste: analisar fatores que comprometem o desempenho do Banco de Dados.

5. INTERPRETAÇÃO DOS RESULTADOS

O JMETER retorna uma série de relatórios para análise dos testes, é possível analisar cada item enviado ao servidor através de um relatório chamado *Aggregate Report*, os valores de cada consulta podem ser verificados, inclusive separados por consulta, conforme mostra figura 9, também é possível acompanhar a evolução dos testes através de gráfico em tempo real de execução, esse recurso é o Graph Results, conforme figura 10.



Label	# Samples	Average	Median	90% Line	Min	Max	Error %	Throughput
Alunos por sexo	16	3752	0	10004	0	10005	100,00%	39,9/min
TOTAL	16	3752	0	10004	0	10005	100,00%	39,9/min

Figura 9: Relatório JMETER Aggregate Report

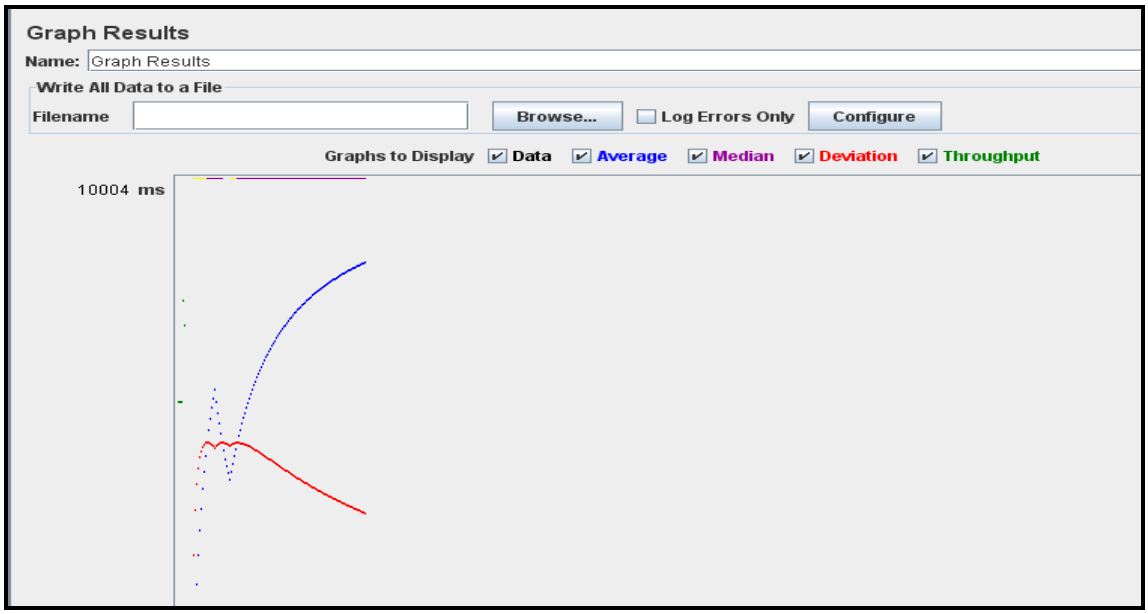


Figura 10: Gráfico JMETER Grafic Results

Para melhor compreender a execução dos testes feitos pelo JMETER precisa-se analisar a função de cada indicador, conforme tabela 6, a partir da resposta dos indicadores já é suficiente para se ter uma base de informações para verificar o desempenho de um Banco de Dados e sua capacidade de carga de usuários.

Tabela 6: indicadores de resposta JMETER

Indicador	Descrição
Samplers	Número de Execuções
Average	Tempo médio de Resposta em Milisengundos
Median	Tempo mediano de resposta
90% Line	Indica a porcentagem de requisições atendidas em um tempo menor que a coluna para de consulta
Min	Menor tempo de resposta
Max	Maior tempo de resposta
Error100%	Lista a porcentagem de erro em atendimento a requisições
Throughput	Numero de requisições atendidas por segundo

5.1 RESULTADO TESTE DE CARGA DE USUÁRIO (T01)

Conforme o resultado obtido pela aplicação JMETER, o Banco de Dados foi remetido à consulta inicial de 10 usuários simultaneamente e a partir da primeira bateria, novas baterias eram executadas, aumentando 10 usuários em cada ciclo de baterias, com o objetivo de conhecer o limite do Banco de Dados o JMETER reportou os seguintes resultados do teste T01 aonde * marca o limite suportado.

a) teste T01 baseado em consulta SQL :

Sintaxe aplicada: *select sexo, count(*) as “qtd aluno” from educação. aluno group by sexo;*

Tabela 7: resultado teste T01 baseado em consultas

Indicador	Carga de Usuarios							
	10	20	30	40	50	53	54	55
Samplers	100	200	450	650	900	1200	0	0
Average	8006	8506	8894	9082	9227	9338	0	0
Median	10004	10004	10004	10004	10004	10004	0	0
90% Line	10005	10005	10005	10005	10005	10005	0	0
Max	10035	10035	10035	10045	10045	10045	0	0
Error100%	0%	0%	0%	0%	0%	100%	100%	100%
Throughput	49,2	41,0	48,4	49,3	53,4	1,0*	0	0
Consumo de Memória Ram	569	571	579	608	723	740*	740	740

A seguir tem-se o gráfico em relação ao throughput, ou seja, a vazão de cada consulta por segundo (CARATTI 2006), a coluna vermelha informa o limite de 53 usuários em que o Banco de Dados suportou.

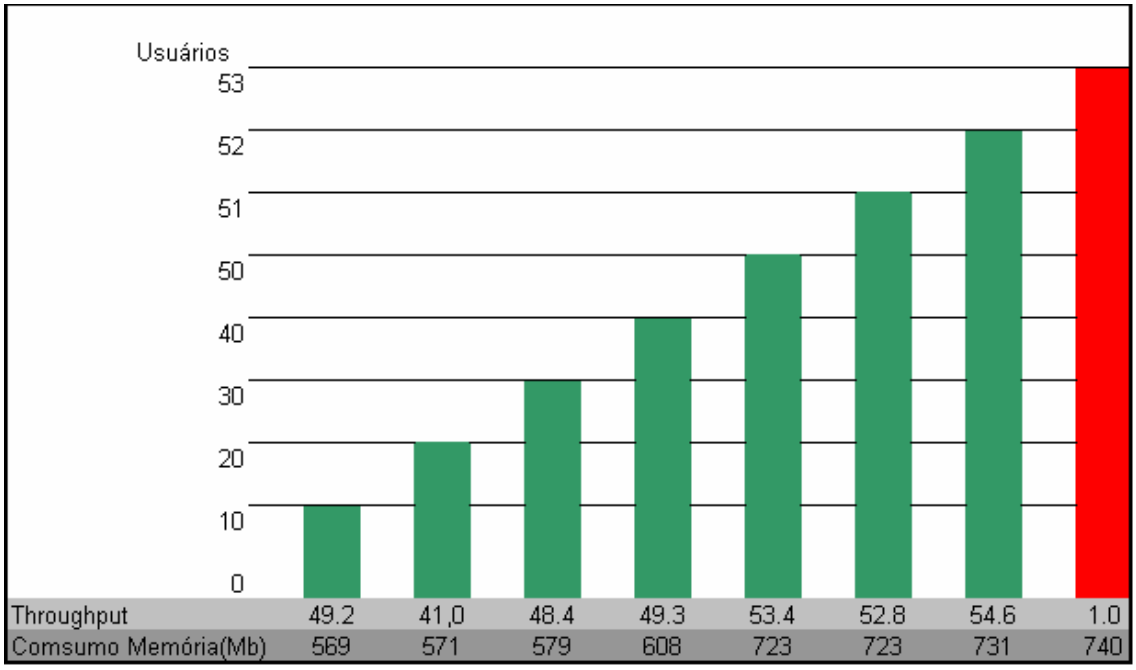


Figura 11: Gráfico de analise teste T01 baseado em consultas

b) teste T01 baseado em inserção:

Sintaxe aplicada: *insert into aluno (nis, nome, sexo, endereco, municipio, fone, filiacao, datanasc) values (null, 'teste', 'm', null, 'sobral', null, null, null);*

Tabela 8: resultado teste T01 baseado em inserção

Indicador	Carga de Usuarios							
	10	20	30	40	50	51	52	53
Samplers	100	300	450	650	900	2050	0	0
Average	8012	8674	8895	9083	9228	9473	0	0
Median	10004	10004	10004	10004	10004	10004	0	0
90% Line	10025	10005	10005	10005	10005	10024	0	0
Max	10075	10075	10075	10075	10154	10154	0	0
Error100%	0%	0%	0%	0%	0%	100%	100%	100%
Throughput	44,7	43,8	9,6	13,2	17,4	14,8*	0	0
Consumo de Memória Ram(Mb)	593	600	610	649	717	720*	0	0

A amostragem do gráfico abaixo que analisa as inserções, identificou que o banco chegou ao limite ao número de 51 usuários.

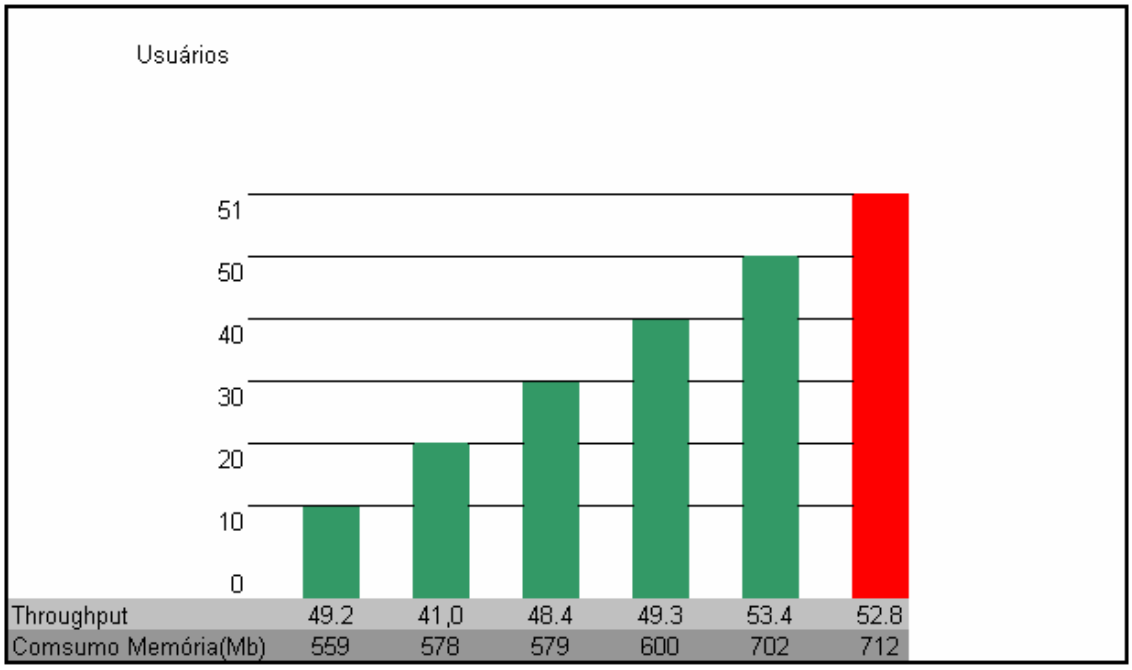


Figura 12: Gráfico de analise teste T01 em inserção

5.2 RESULTADO TESTE DE DESEMPENHO (T02)

Conforme a imposição do agente de dificuldade, o JMETER retornou os seguintes valores abaixo:

a)teste T02 agente de dificuldade: desfragmentador em execução

Sintaxe aplicada: *select sexo, count(*) as "qtd aluno" from educação. aluno group by sexo;*

Tabela 9: resultado teste T02 com agente de dificuldade

Indicador	Carga de Usuarios							
	10	20	30	40	45	48	47	48
Samplers	100	220	350	550	790	820	0	0
Average	8018	8516	8873	9107	9255	9266	0	0
Median	10004	10004	10004	10004	10004	10004	0	0
90% Line	10005	10005	10034	10035	10025	10025	0	0
Max	10966	10966	10966	10966	10966	10966	0	0
Error100%	100%	100%	100%	100%	100%	100%	0	0
Throughput	27,6	29,2	37,0	44,6	47,7	52,5	0	0
Consumo de Memória Ram(Mb)	595	654	730	748	779	0	0	0

De acordo com a bateria de testes, a limitação de CPU foi o principal fator que fez com que o Banco de Dados não atendesse mais usuários, ao impor a desfragmentação em paralelo com o teste o desempenho da CPU caiu consideravelmente, aumentando sua carga de processamento e limitando assim a entrada de novos usuários, conforme mostra figura 13

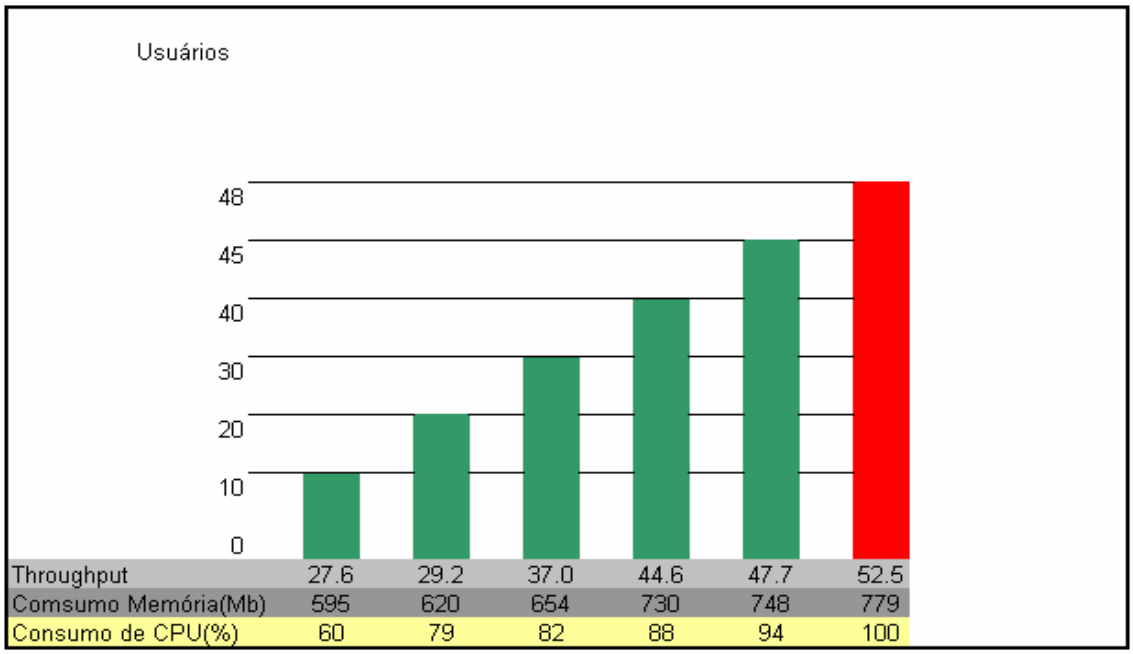


Figura 13: Gráfico de analise do teste T02, com agente de dificuldade.

5.3 RESULTADO TESTE DE DESEMPENHO DE HARDWARE (T03)

No teste de desempenho de hardware o resultado demonstrou um tanto quanto a força da lógica, que quanto mais recurso, mais capacidade. Conforme a tabela 9 mostra o aumento da capacidade de memória também aumentou a vazão do Banco de Dados

Tabela 10: resultado teste T03

Indicadores	Capacidade Mem		
	512 Mb	1 Gb	2 Gb
Samplers	75	75	75
Average	2262	2263	2265
Median	1522	1522	1523
90% Line	5247	5247	5247
Max	10575	10576	10576
Error100%	0%	0%	0%
Throughput	10,6	15,6	18,4
Consumo de Memória Ram(Mb)	430	440	448
Samplers	75	75	75

5.4 RESULTADO TESTE DE DESEMPENHO COM CONCORRÊNCIA DE SOFTWARE (T04)

Ao testar o banco de dados em paralelo a serviços concorrentes foi analisado conforme a tabela 11 abaixo que o serviço que mais causou perda de desempenho em vazão (Throughput) ao Banco de Dados foi o Backup, seguido pela varredura de vírus, e por ultimo a atualização de software, nesse caso em específico, atualização software antivírus, também demostrado na figura 14 em forma de participação da "quantidade" de vazão.

Sintaxe aplicada: *select sexo, count(*) as "qtd aluno" from educação. aluno group by sexo;*

Tabela 11: resultado teste T04

	Serviços Concorrentes			
	Antivírus	Backup	Atualização de Software	Sem serviço concorrente
Throughput	6,2	5,8	6,9	7,8
Consumo de Memória Ram(Mb)	624	615	690	587

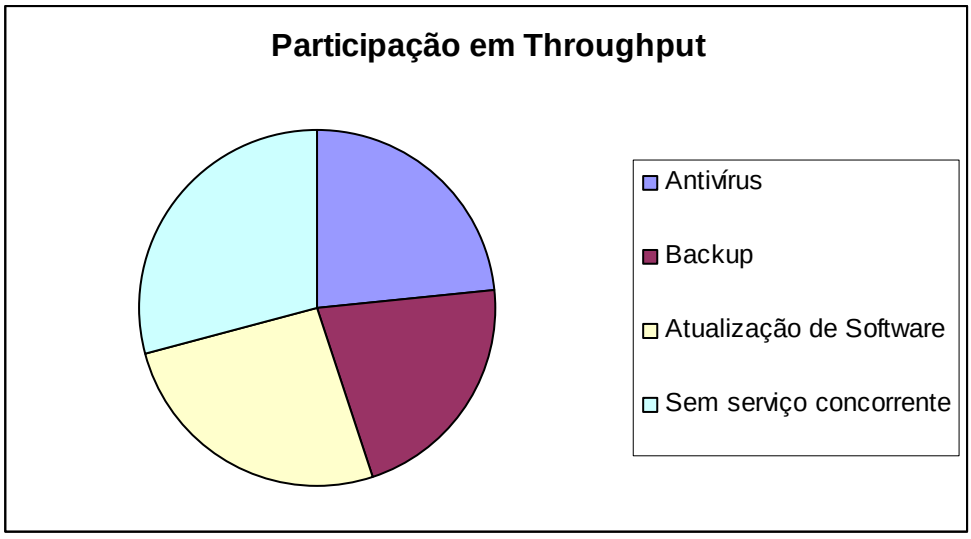


Figura 14: Gráfico de analise do teste T04, com serviços concorrentes.

5.5 RESULTADO FINAL DOS TESTES

Tendo em vista os resultados apresentados nas seções anteriores, prova-se que foram encontrados métodos e ferramentas a fim de buscar o numero de usuários em que certo Banco de Dados pode suportar, também foi paralelamente demonstrado como uma aplicação, quando executada em um servidor de Banco de Dados, contribui para a diminuição de seus recursos, limitando assim o seu desempenho e capacidade de suportar mais usuários.

Prova-se também que o software JMETER é uma importante ferramenta para análise de desempenho e carga, fundamental para o setor de Desenvolvimento e Engenharia de Software.

CONCLUSÃO

Este trabalho apresentou um estudo sobre as formas e métodos de encontrar em certo banco de dados o máximo de usuários em que ele é capaz de suportar, uma extensa pesquisa foi realizada sobre os bancos de dados, formas de controle de concorrência, desempenho e otimização.

Para o objetivo, foram pesquisados modelos de aplicações cujo suas características eram de testar, analisar e documentar informações a respeito do acessos simultâneo e desempenho de um banco de dados, contudo a aplicação JMETER desenvolvida pela Apache, serviu como ferramenta principal em toda a rotina de testes que foram executadas no andamento do trabalho.

Uma pesquisa entre empresas de desenvolvimento também foi realizada, a fim de buscar uma visão das necessidades de tal estudo, buscou-se através de agentes externos, fatores que influenciam no desempenho de um servidor de banco de dados, e que contribuem para a falta de recursos do mesmo.

Infelizmente esse trabalho não é totalmente conclusivo, pois há ainda muitas metodologias e métricas diferentes para se testar o desempenho e acessos simultâneos a um banco de dados. Como trabalhos futuros sugerem-se

a) analise do desempenho em banco de dados com outros aplicativos;

b) análise de otimização em SQL para o mesmo fim;

c) analise de capacidade e carga em banco de dados distribuídos.

REFERÊNCIAS

NEVES, Denise Lemes Fernandes**. PostgreSQL : comceitos e aplicações**. São Paulo: Érica, 2002.

KORTH, Henry F.; SILBERSCHATZ, Abraham. **Sistema de bancos de dados**. 2 ed. rev. São Paulo: Makron Books, 1995**.**

GARCIA-MOLINA, Hector; ULLMAN, Jeffrey D.; WIDOM, Jennifer. **Implementação de sistemas de bancos de dados**. Rio de Janeiro: Ed. Campus, 2001

MACHADO, Felipe Nery Rodrigues; ABREU, Maurício Pereira de. **Projeto de banco de dados: uma visão prática.** 8.ed São Paulo: Érica, 2002

PLEW, Ronald R.; STEPHENS, Ryan K. **Aprenda em 24 horas SQL**. Rio de Janeiro: Ed. Campus, 2000.

DATE, C. J. **Guia para o padrão SQL**. Rio de Janeiro: Ed. Campus, 1998

MANZANO, José Augusto N. G. **Estudo dirigido : SQL**. São Paulo: Érica, 2002.

CARATTI, Ricardo Lima . **Teste de capacitação do BD com Jmeter** . São Paulo : Revista SQL Magazine . Setembro 2006

ELMASRI, NAVATHE**. Sistema de bancos de dados fundamentos e aplicações** . 3 ed. rev. São Paulo: ltc, 2002.

SILBERSCHATZ, KORTH,SUDARSHAN. **Sistema de bancos de dados**. 5 ed. São Paulo: Ed.Campus, 2006.

DATE, C. J. **Introdução s Sistemas de Banco de Dados.** 7 . Rio de Janeiro: Ed. Campus, 2000

SILVA, Rosângela. **Bancos de Dados Geográficos**: uma análise das arquiteturas dual (Spring) e integrada (Oracle Spatial). São Paulo: Editora USP, 2002.

OLIVEIRA, Adelise Generini. **SQL para delphi4**. Florianópolis: Visual Books, 1999.

SONÁGLIO, Wagner Comin.2007. 128 f. **Analise comparativa do desempenho dos métodos de acessos para dados espaciais:Estudo de caso com o postgis**.Trabalho de conclusão de curso- Curso de Ciências da Computação, Universidade do Extremo Sul Catarinese, Criciúma, Santa Catarina,2007.

SURIAN, Jorge. NICOCHELLI, Luiz. BOGHI, Cláudio. **Banco de dados e SQL:** Disponível em www.imasters.com.br/tutoriais. Acessos em 30 set.2007