ULBRA
CENTRO UNIVERSITÁRIO LUTERANO DE PALMAS
COMUNIDADE EVANGÉLICA LUTERANA "SÃO PAULO"
Recredenciado pela Portaria Ministerial nº 3.607 - D.O.U. nº 202 de 20/10/2005

**Rogério Lopes Ferreira**

# Cluster MySQL: estudo de caso na Fábrica de Software do CEULP/ULBRA

**Palmas - TO**

**2013**

**Rogério Lopes Ferreira**

# Cluster MySQL: estudo de caso na Fábrica de Software do CEULP/ULBRA

Trabalho de Conclusão de Curso (TCC) elaborado e apresentado como requisito parcial para obtenção do título de bacharel em Sistemas de Informação pelo Centro Universitário Luterano de Palmas (CEULP/ULBRA).

Orientador: Prof. M.Sc. Jackson Gomes de Souza.

**Palmas - TO**

**2013**

**Rogério Lopes Ferreira**

# Cluster MySQL: estudo de caso na Fábrica de Software do CEULP/ULBRA

Trabalho de Conclusão de Curso (TCC) elaborado e apresentado como requisito parcial para obtenção do título de bacharel em Sistemas de Informação pelo Centro Universitário Luterano de Palmas (CEULP/ULBRA).

Orientador: Prof. M.Sc. Jackson Gomes de Souza.

**Aprovada em: 06 de Dezembro de 2013.**

**BANCA EXAMINADORA**

_____

Prof. M.Sc. Jackson Gomes de Souza

Centro Universitário Luterano de Palmas

_____

Prof. M.Sc. Fernando Oliveira

Centro Universitário Luterano de Palmas

_____

Prof. M.Sc. Madianita Bogo

Centro Universitário Luterano de Palmas

**Palmas - TO**

**2013**

Dedico este trabalho aos meus queridos e dedicados pais, que não mediram esforços para me fornecer todo amor e apoio em todos os momentos desta etapa de minha vida.

Também é dedicado a minha esposa, por todos os dias de cobranças e apoio.

**AGRADECIMENTOS**

A Deus por ter me dado forças e iluminando meu caminho para que pudesse concluir mais uma etapa da minha vida. Aos meus pais, por todo amor e dedicação.
A minha esposa Deybianne Silva de Araújo e agora também Ferreira, pelo apoio nas horas mais difíceis.
Aos meus amigos Leafarnel, Ícaro, Tony e todos que trabalham no Portal por ajudar não só neste trabalho, mas em vários outros durante a faculdade. Ao meu orientador Jackson Gomes, pelos ensinamentos e dedicação dispensados no auxilio a concretização dessa monografia.
Aos professores que me acompanharam na minha vida acadêmica.
Aos meus colegas de trabalho da TI do SEBRAE, pelo apoio, incentivo e confiança que foram fundamentais para minha formação. A todos aqueles que contribuíram direta ou indiretamente para que esse trabalho fosse realizado meu eterno AGRADECIMENTO.

RESUMO

A tecnologia se expande para proporcionar ambientes mais eficientes quanto a disponibilidade e desempenho dos sistemas. Este trabalho aborda os conceitos de Banco de Dados Distribuídos, Cluster e seus principais tipos, bem como propõe um ambiente Distribuído de Banco de Dados voltado ao ganho de desempenho da Rede Social Conecta Disponibilizado pelo CEULP/ULBRA.

**PALAVRAS-CHAVE: Cluster, Sistemas Distribuído, Banco de Dados, Banco de Dados Distribuído, MySQL Cluster, NDB Cluster.**

## LISTA DE FIGURAS

Figura 1: Exemplo de Modelo de um Cluster ..... 9
Figura 2: Cluster para alta disponibilidade ..... 10
Figura 3: Cluster de alto desempenho ..... 12
Figura 4: Cluster para balanceamento de carga ..... 13
Figura 5: Busca de contatos na agenda telefônica. ..... 15
Figura 6: SGBD e os Metadados. ..... 16
Figura 7: BDD heterogêneos e homogêneos, respectivamente ..... 18
Figura 8: SGBD Global e Local ..... 20
Figura 9: SGBD local assumindo temporariamente o papel de SGBD global ..... 22
Figura 10: Mudanças de estado de uma transação (NAVATHE 2011, p. 506) ..... 23
Figura 11: Balanceamento de carga do SGBDD ..... 30
Figura 12: Fracionamento de transações ..... 31
Figura 13: VMware Workstation (VMware 2013, online) ..... 34
Figura 14: *MySQL* Enterprise Monitor (*MySQL* 2013, online) ..... 38
Figura 15: Medidor de Desempenho do Windows ..... 39
Figura 16: Ambiente Centralizado ..... 47
Figura 17: Ambiente Cluster ..... 50
Figura 18: Divisão de tabelas no Cluster ..... 51

## LISTA DE TABELAS

Tabela 1: Características Físicas dos nós do Cluster ..... 48

Tabela 2: Características Lógicas dos nós do Cluster ..... 48

Tabela 3: Divisão dos testes de estresse ..... 54

Tabela 4: Médias de consumo de recursos e tempos de resposta por usuário do Ambiente Centralizado ..... 55

Tabela 5: Médias de consumo de recursos e tempo de respostas por usuário do Ambiente Distribuído ..... 56

Tabela 6: Médias comparativas de consumo de recursos e tempo de respostas por usuário dos Ambientes Centralizado e Distribuído ..... 57

## Lista de Gráficos

Gráfico 1: Média de utilização de Memória (MB) por usuário nos Ambientes Centralizado e Distribuído ..... 59
Gráfico 2: Média de utilização de Disco (MB/s) por usuário nos Ambientes Centralizado e Distribuído ..... 60
Gráfico 3: Média de utilização de Rede (KB/s) por usuário nos Ambientes Centralizado e Distribuído ..... 61
Gráfico 4: Média de Requisições Atendidas/s por usuário nos Ambientes Centralizado e Distribuído ..... 62
Gráfico 5: Média de Tempos de Resposta por usuário nos Ambientes Centralizado e Distribuído ..... 63
Gráfico 6: Média de utilização de CPU (%) no Ambiente Centralizado..... 71
Gráfico 7: Média de utilização de CPU (%) no Ambiente Centralizado..... 72
Gráfico 8: Média de utilização de Memória (MB) no Ambiente Centralizado..... 73
Gráfico 9: Média de utilização de Disco (MB/s) no Ambiente Centralizado ..... 74
Gráfico 10: Média de utilização de Rede (KB/s) no Ambiente Centralizado ..... 76
Gráfico 11: Média de Requisições Atendidas /s no Ambiente Centralizado ..... 77
Gráfico 12: Média de Tempos de Resposta por usuário no Ambiente Centralizado ..... 78
Gráfico 13: Média de utilização de CPU (%) no Ambiente Distribuído..... 80
Gráfico 14: Média de utilização de Memória (MB) por usuário no Ambiente Distribuído..... 81
Gráfico 15: Média de utilização de Disco (MB/s) por usuário no Ambiente Distribuído..... 82
Gráfico 16: Média de utilização de Rede (KB/s) por usuário no Ambiente Distribuído ..... 84
Gráfico 17: Média de Requisições Atendidas/s por usuário no Ambiente Distribuído..... 85
Gráfico 18: Média de Tempos de Resposta por usuário no Ambiente Distribuído..... 86

# SUMÁRIO

**1 INTRODUÇÃO ..... 6**
**2 REFERENCIAL TEÓRICO ..... 9**
**2.1. Cluster ..... 9**
**2.1.1. Cluster para alta disponibilidade ..... 10**
**2.1.2. Cluster para alto desempenho ..... 11**
**2.1.3. Cluster para balanceamento de carga ..... 13**
**2.2. Banco de Dados ..... 14**
**2.2.1. Sistema de Gerenciamento de Banco de Dados ..... 15**
**2.3. Banco de Dados Distribuídos ..... 17**
**2.3.1. Sistema de Gerenciamento de Banco de Dados Distribuídos ..... 19**
**2.3.2. Gerenciamento de Transações em Bancos de Dados Distribuídos ..... 21**
**2.3.3. Processamento de Consultas Distribuídas ..... 24**
**2.3.3.1. Mapeamento de consultas ..... 24**
**2.3.3.2. Localização ..... 25**
**2.3.3.3. Otimização global da consulta ..... 26**
**2.3.3.4. Execução da consulta local ..... 26**
**2.3.4. Controle de Concorrência e Recuperação ..... 26**
**2.4. Alto Desempenho em Bancos de Dados Distribuídos ..... 29**
**3 MATERIAIS E MÉTODOS ..... 33**
**3.1. Materiais ..... 33**
**3.1.1. *Conecta* ..... 33**
**3.1.2. VMware Workstation ..... 34**
**3.1.3. *MySQL* Cluster ..... 35**
***3.1.4. JMeter* ..... 36**
**3.1.5. *MySQL* Enterprise Monitor ..... 37**
**3.1.6. Medidor de desempenho do Windows ..... 38**
**3.2. Métodos ..... 39**
**4 RESULTADOS E DISCUSSÃO ..... 45**
**4.1. Ambientes virtuais de homologação ..... 45**
**4.1.1. Ambiente Centralizado (Não Cluster) ..... 46**
**4.1.2. Ambiente Distribuído (Cluster) ..... 48**

**4.2. Testes: Visão Geral e parâmetros ..... 52**
**4.2.1. Consumo de recursos do BD e Resultados dos experimentos no ambiente Centralizado ..... 54**
**4.2.2. Consumo de recursos do BD e Resultados dos experimentos no ambiente Distribuído ..... 56**
**4.3. Comparativo dos resultados de desempenho ..... 57**
**5 CONSIDERAÇÕES FINAIS ..... 65**
**6 REFERÊNCIAS BIBLIOGRÁFICAS ..... 67**
**APÊNDICES ..... 70**
**Ambiente Centralizado ..... 71**
**Porcentagem de consumo de CPU ..... 71**
**Consumo de memória RAM ..... 73**
**Consumo de Disco ..... 74**
**Consumo de Rede ..... 75**
**Requisições atendidas ..... 77**
**Tempos de resposta ..... 78**
**Ambiente Distribuído ..... 79**
**Porcentagem de consumo de CPU ..... 79**
**Consumo de memória RAM ..... 80**
**Consumo de Disco ..... 82**
**Consumo de Rede ..... 83**
**Requisições atendidas ..... 85**
**Tempos de resposta ..... 86**

# 1 INTRODUÇÃO

Com a crescente expansão tecnológica e a necessidade de armazenamento, manipulação e análise de dados, os bancos de dados passaram a representar uma das principais formas de persistência de dados. As mídias compartilhadas, as transações bancárias em diversos tipos de dispositivos (inclusive os móveis) e os serviços em nuvem necessitam de bases de dados eficientes e eficazes. Sem eficiência e eficácia em suas bases de dados tais sistemas perderiam muito de sua disponibilidade, apresentando falhas e, consequentemente, inconsistências em suas transações. Estas falhas por indisponibilidade do serviço podem ser consequência de quando uma base de dados recebe uma carga de transações muito alta para ser executada, podendo acarretar, assim, aumento do tempo de resposta, inconsistência dos dados e, até, travamentos do sistema de computação que serve o banco de dados.

O limite máximo da carga de transações de um banco de dados é medido pelos limites dos requisitos físicos de seu servidor, ou seja, a capacidade de processamento disponível pela CPU, a quantidade de memória disponível e as tecnologias de componentes como memória RAM e discos rígidos. Uma alternativa para elevar estes limites seria expandir estes recursos físicos, contudo nem sempre é uma alternativa barata e viável.

Uma das formas de resolver os problemas citados anteriormente é utilizar conceitos de Sistemas Distribuídos. Sistemas Distribuídos são um conjunto de equipamentos independentes e distribuídos de forma que compartilham seus recursos para um determinado fim (COULOURIS et al. 2007 p.16). Eles trazem vantagens como o compartilhamento de recursos e a escalabilidade. Estas vantagens proporcionam ganhos significativos no ambiente em que o sistema distribuído está sendo executada como a utilização de todo o recurso físico do ambiente distribuído em momentos de necessidade e a possibilidade de expansão destes recursos.

Um *cluster* é formado por um conjunto de computadores trabalhando como se fossem um único sistema.

A utilização de bancos de dados distribuídos em *cluster* é uma alternativa para a expansão dos limites de computação. Pelas características do

comportamento de um *cluster* de computadores, não há a necessidade de se ter vários servidores com o mesmo *hardware* ou *software*, ou mesmo servidores com grande poder de processamento, pois as tarefas relacionadas ao banco de dados, como transações de leitura e escrita, são distribuídas no *cluster* e, com isso, há menor carga individual por servidor no *cluster*.

A Fábrica de Software do CEULP/ULBRA disponibiliza o *software Conecta*, que é uma rede social que se encontra em fase de desenvolvimento e testes. Esta rede social tem sua base de dados executada em um banco de dados *MySQL* não distribuído e sua aplicação em um servidor *web Apache*. Atualmente, estes serviços (banco de dados e HTTP) estão em execução em um servidor individual.

Pelo fato da rede Conecta ainda não ter sido disponibilizada oficialmente ao seu público (alunos e professores do CEULP/ULBRA), embora o *software* tenha passado por testes de estresse, não há dados reais da sua utilização. Portanto, a Fábrica de Software trabalha com estimativas de acesso, baseada na utilização de outros *softwares* em utilização no CEULP/ULBRA e na quantidade de alunos e professores (em torno de cinco mil pessoas). Dentre estes, destaca-se o “Acesso interno”, que fornece aos alunos, por exemplo, a funcionalidade de respostas a atividades extracurriculares propostas pelos professores. Neste caso, o acesso médio, em torno de 5.000 (cinco mil) a 7.000 (sete mil) por dia, e chega, em dias de pico (prazos de entregas de respostas dos alunos), de 50.000 (cinquenta mil) a 70.000 (setenta mil) acessos diários (durante dois a três dias)[1].

O ambiente que disponibiliza a rede social *Conecta* foi replicado em um ambiente virtual, buscando através de testes de estresse medir o comportamento da aplicação e sua base de dados através da coleta dos dados referente ao consumo de recursos (CPU, Disco e a média de transações atendidas por segundo). Após as medições foi possível verificar que com uma quantidade de 1000 usuários simultâneos acessando o *Conecta* o atual servidor não suportaria a carga de acessos gerada.

Este trabalho busca responder se o uso de cluster de banco de dados *MySQL* traz aumento de desempenho acima de 10%, onde o ganho será demonstrado em forma de comparativo de consumo de recursos por usuário, o que permitirá maior velocidade para servir conteúdo do banco de dados para o software *Conecta*. O

[1] Estas informações são fornecidas pela Fábrica de Software, obtidas por meio de relatórios de acesso.

valor de porcentagem de ganho de 10% foi um valor definido de forma empírica, sem testes ou estudos.

Este trabalho está organizado da seguinte forma: no capítulo 2 será apresentado o referencial teórico referente à Cluster, Banco de dados, Banco de Dados Distribuídos, Gerenciamento de Transações, Processamento de Consultas, Controle de Concorrência e Recuperação e Alto Desempenho em Bancos de Dados Distribuídos, no capítulo 3 serão apresentados os materiais e métodos utilizados para a elaboração deste trabalho; no capítulo 4 são mostrados os resultados e as discussões; no capítulo 5 serão apresentadas as considerações finais deste trabalho e trabalhos futuros e, por fim, são listadas as referências bibliográficas.

# 2 REFERENCIAL TEÓRICO

Neste trabalho serão apresentados os conceitos de Cluster, Banco de Dados Distribuídos, Gerenciamento de Transações, Processamento de Consultas e Alto desempenho em Banco de Dados Distribuídos. Tais conceitos serão necessários para o entendimento das tecnologias envolvidas na implantação de um Banco de Dados Distribuído na Fábrica de Software do CEULP/ULBRA, com o objetivo de se obter um maior desempenho em uma aplicação web.

## 2.1. Cluster

Segundo Buyya (1999 apud CONTI, 2009):

> Cluster é um tipo de sistema para processamento paralelo e distribuído que consiste de uma coleção de computadores interconectados e trabalhando juntos como um único recurso computacional integrado. Com esse conjunto de computadores é possível realizar processamentos que até então apenas computadores de alto desempenho conseguiam executar.

Esta tecnologia permite a utilização de vários computadores, chamados de *nós*, interligados por uma rede lógica de comunicação, de modo que sejam utilizados como um único equipamento, distribuindo entre seus *nós* as cargas de tarefas a serem processadas. A Figura 1 ilustra uma estrutura de computadores em *cluster*.

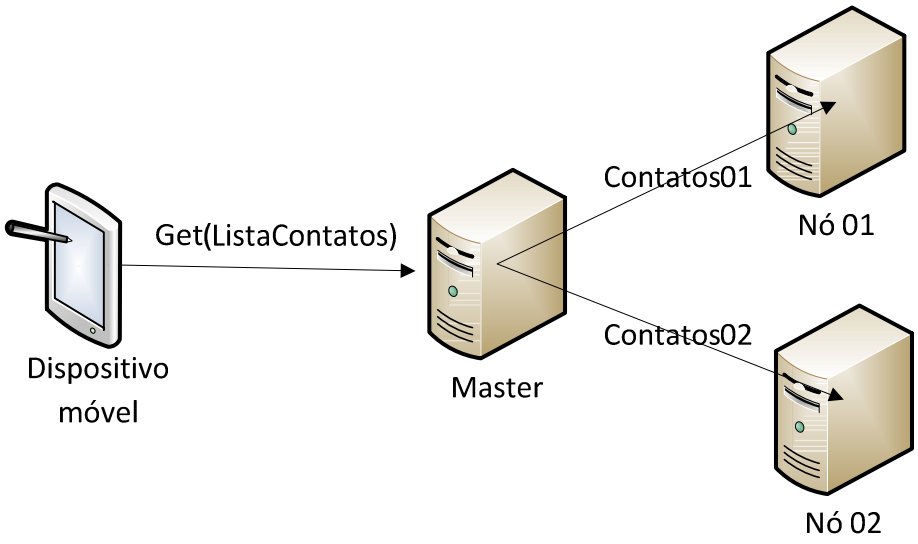


**Figura 1: Exemplo de Modelo de um Cluster**

Conforme apresentado na Figura 1, os dispositivos e sistemas realizam solicitações ao conjunto formado pelo *Cluster*. No exemplo da Figura 1, um

dispositivo móvel solicita ao *cluster* uma lista de contatos. As solicitações chegam ao servidor conhecido como *máster*, que é responsável por distribuir e gerenciar as tarefas a serem realizadas pelos nós. A execução das tarefas nos nós ocorre conforme o tipo do cluster: Cluster para alta disponibilidade, Cluster para alto desempenho e Cluster para balanceamento de carga. Estes serão apresentados nas próximas seções.

### 2.1.1. Cluster para alta disponibilidade

Disponibilidade de sistema é a característica que define que um sistema deve estar disponível quando necessário (SOBRINHO 2009, p.12). Esta característica é essencial, sendo buscada por todos os sistemas.

Segundo Rodrigues (2007, p.11), o objetivo da arquitetura de *Cluster* para alta disponibilidade é evitar falhas de acesso, as quais acontecem, por exemplo, caso um disco venha a se danificar devido ao alto tráfego de informações no conjunto de servidores. Para isso, os dados são constantemente replicados entre os *nós*, permitindo que, ao ocorrer falhas em um dos *nós*, outro [*nó*] possa assumir a tarefa. Uma comunicação contínua dentro do *Cluster* permite identificar os *nós* que não estejam operando normalmente, como apresenta a Figura 2.

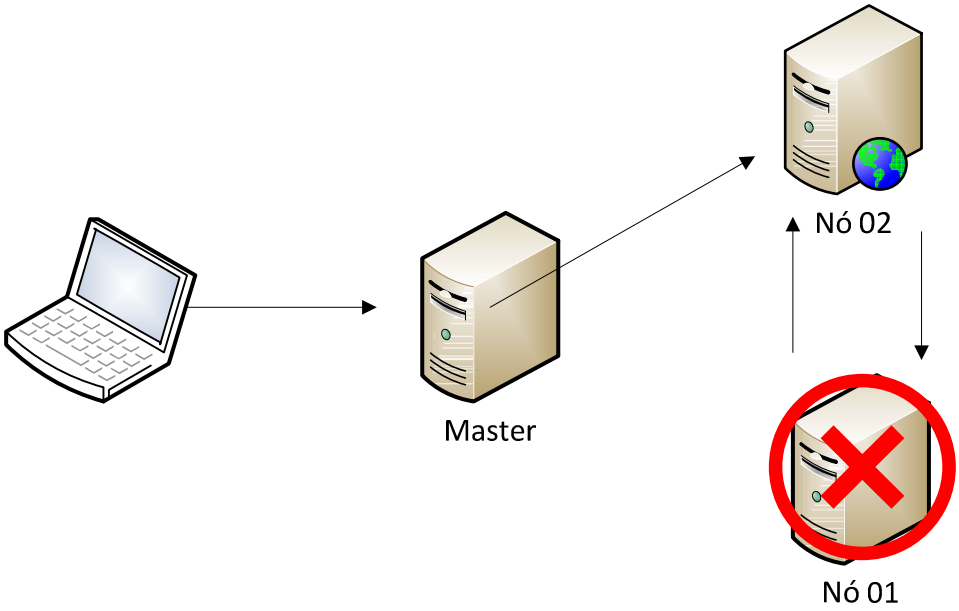


**Figura 2: Cluster para alta disponibilidade**

Conforme apresentado na Figura 2, após ser identificado que o "*Nó 01*" está inoperante, o segundo *nó* ("*Nó 02*") assume seu lugar. Este tipo de cluster opera de forma a garantir ao máximo a disponibilidade do serviço que está sendo disponibilizado.

Pode-se citar como exemplo de utilização deste tipo de cluster uma empresa que disponibiliza um serviço de rastreio de veículos através de um sistema web. Mesmo que o sistema não sofra com grandes cargas de acessos, este sistema necessita estar sempre disponível, pois a falta de disponibilidade deste serviço fará com que, na ocasião de furto de veículo rastreado, sua recuperação seja mais difícil ou, até mesmo, impossibilitada. Portanto, se houver queda do servidor principal que disponibiliza o serviço, outro servidor deve assumir, mantendo a disponibilidade do serviço.

### 2.1.2. Cluster para alto desempenho

A arquitetura de *Cluster* para alto desempenho enfoca no desempenho das aplicações. Segundo Pitanga (2003, p.01), este tipo de cluster caracteriza-se por fracionar o processamento total entre os computadores buscando-se um melhor desempenho. Com este fracionamento do processamento cada *nó* processa somente uma fração do total e, uma vez concluído o processamento, o resultado é devolvido ao *nó máster* para que esta fração possa ser estruturada novamente. Este processo de descentralização da informação aproveita o poder de processamento de todos os computadores disponíveis no *Cluster*, conforme apresenta a Figura 3.

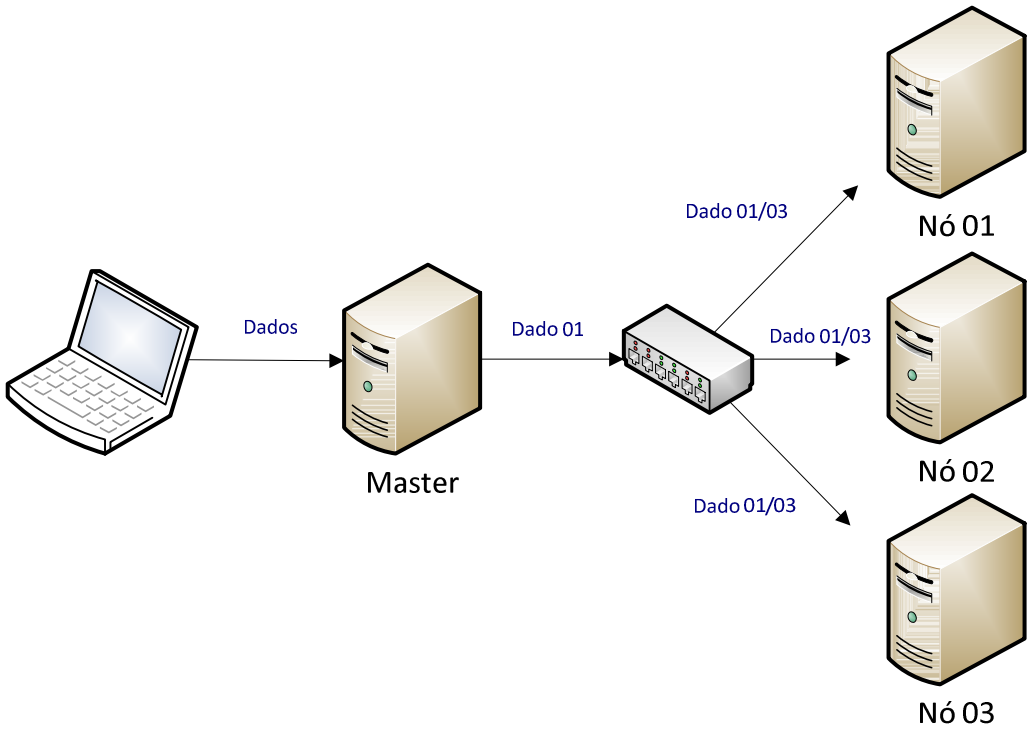


**Figura 3: Cluster de alto desempenho**

Como apresentado na Figura 3, os dados recebidos pelo *nó máster* são fracionados em partes iguais à quantidade de computadores disponíveis para o processamento e então distribuídos entre os *nós* do *Cluster*. Após o fracionamento e distribuição dos dados, os nós processam cada fração que recebeu e as devolve ao nó *máster* onde serão reagrupados e devolvidos ao solicitante.

Pode-se citar como exemplo de utilização deste tipo de cluster uma empresa que disponibiliza um buscador online, onde seus acessos são realizados por milhares de pessoas simultaneamente. Devido à enorme quantidade de acessos, o sistema passa a apresentar tempos de resposta maiores que os apresentados costumeiramente, necessitando assim distribuir as tarefas a serem processadas. As requisições são recebidas pelo servidor principal e então distribuídas entre seus servidores escravos ou secundários ou simplesmente nós, dividindo assim todas as tarefas a serem realizadas, onde cada nó receberá uma fração desta requisição a ser processada e devolvendo o resultado para o servidor principal em menor tempo que seria se estivesse realizando todo o processamento sozinho.

### 2.1.3. Cluster para balanceamento de carga

A arquitetura de *Cluster* para balanceamento de carga é voltada para o balanceamento da carga a ser processada, ou seja, o balanceamento do montante de dados e requisições a serem processados. Segundo Rodrigues (2007, p.12), este tipo de *Cluster* caracteriza-se por distribuir a carga de dados a serem processados entre seus *nós*, de maneira a não sobrecarregar os computadores e garantirem a disponibilidade do serviço. Um balanceador de carga é utilizado conforme demonstra a Figura 4.

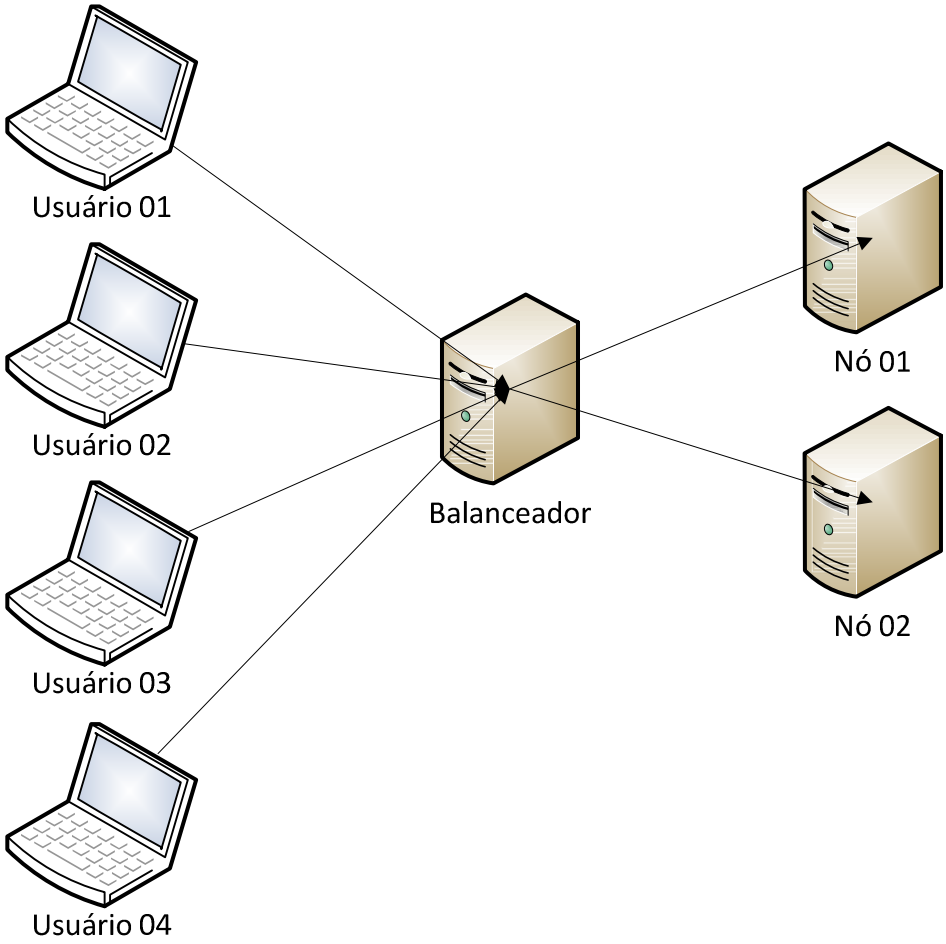


**Figura 4: Cluster para balanceamento de carga**

Conforme apresentado na Figura 4, as várias requisições são enviadas pelos usuários ao balanceador, que as distribui entre os computadores do *Cluster*, reduzindo assim a indisponibilização do serviço por falha em um dos servidores.

Pode-se citar como exemplo de utilização deste tipo de cluster uma empresa que disponibiliza um serviço de vendas de produtos, através de um site de vendas. O sistema sofre com muitos acessos simultaneamente e necessita estar sempre

disponível. O balanceador receberá as solicitações de transações e encaminhará ao nó disponível com menor carga naquele momento, reduzindo assim a possibilidade de deixar o serviço indisponível.

Por fim observa-se que a tecnologia cluster é uma alternativa a ganho de poder de processamento e disponibilidade. Seus benefícios são utilizados por grandes empresas que disponibilizam sistemas a grande quantidade de usuários, onde, buscam sempre garantir alto desempenho e disponibilidade em seus serviços.

Cluster também proporcionam escalabilidade, que segundo Amazon (2007), o conceito representa a capacidade de expansão dos recursos de um sistema na mesma proporção que o sistema ganha em desempenho. Com escalabilidade, ao se necessitar de mais recursos como CPU, memória ou disco, será necessário somente adicionar mais nós ao cluster, somando-se assim os recursos físicos dos nós aos recursos totais disponíveis do sistema.

Uma utilização muito comum da tecnologia cluster se dá em Banco de Dados Distribuídos, buscando-se maior disponibilidade e/ou desempenho, onde os conceitos de Banco de Dados e suas tecnologias serão apresentados nas próximas seções.

## 2.2. Banco de Dados

Bancos de dados são utilizados em aplicações diversas para se obtiver persistência nos dados armazenados. Por exemplo, são utilizados em um sistema bancário no qual, ao ser feita uma movimentação bancária é realizada uma modificação na base de dados onde estão armazenadas as informações das contas correntes de seus clientes.

Navathe et al. (2011, p.03) definem Banco de Dados como sendo uma coleção de dados relacionados. Estes dados estão alocados em equipamentos eletrônicos, mais comumente servidores, onde são acessados por aplicações que necessitam de seus dados.

Com a utilização de bancos de dados as aplicações podem armazenar todos os tipos de dados, possibilitando assim a recuperação, inserção, atualização e exclusão destas informações a qualquer momento.

Como exemplo de banco de dados pode-se apresentar a utilização da agenda de contatos de um celular ou *smartphone*, onde podem ser inseridos novos contatos,

além de atualizar, buscar e apagar contatos existentes. Os contatos são armazenados em um banco de dados dentro do aparelho, e assim ficam disponíveis para outras aplicações. Ao se buscar um contato nesta lista é realizada uma transação de busca na base de dados do aparelho, conforme será apresentado na Figura 5.

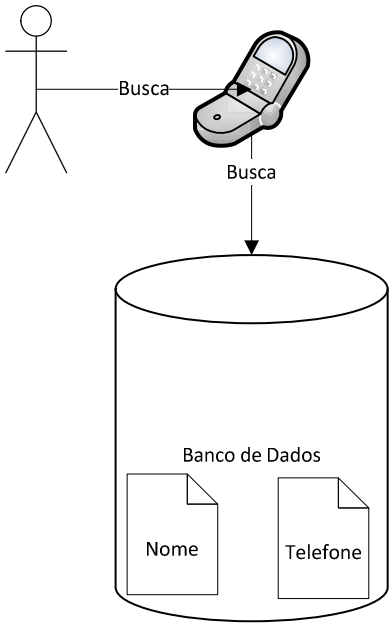


**Figura 5: Busca de contatos na agenda telefônica.**

Conforme apresentado na Figura 5, uma base de dados com informações dos contatos, composta basicamente por nome e telefone, se encontra no dispositivo de comunicação, possibilitando desta forma o acesso a estes dados pelo usuário.

Uma aplicação pode manipular diretamente os dados em um banco de dados, essa manipulação consiste em criar, apagar, atualizar e buscar os dados, contudo um Sistema de Gerenciamento de Banco de Dados (SGBD) reduz enormemente sua utilização e gerenciamento, como será apresentado no próximo tópico.

### 2.2.1. Sistema de Gerenciamento de Banco de Dados

Um SGBD contempla o conjunto de ferramentas necessárias para a utilização e gerenciamento de um banco de dados. Heuser (1998, p.04) define SGBD como sendo um software que incorpora as funções de definição, recuperação e alteração de dados em um banco de dados. O SGBD possibilita as definições de criação de um banco de dados como tipo, estrutura e restrições dos dados a serem

armazenados. O SGBD armazena estas definições nos *metadados,* uma espécie de dicionário, os quais são acessados sempre que uma transação for solicitada (TAKAI 3.et al. 2005, p.15). O processo de acesso aos metadados é ilustrado pela Figura 6.

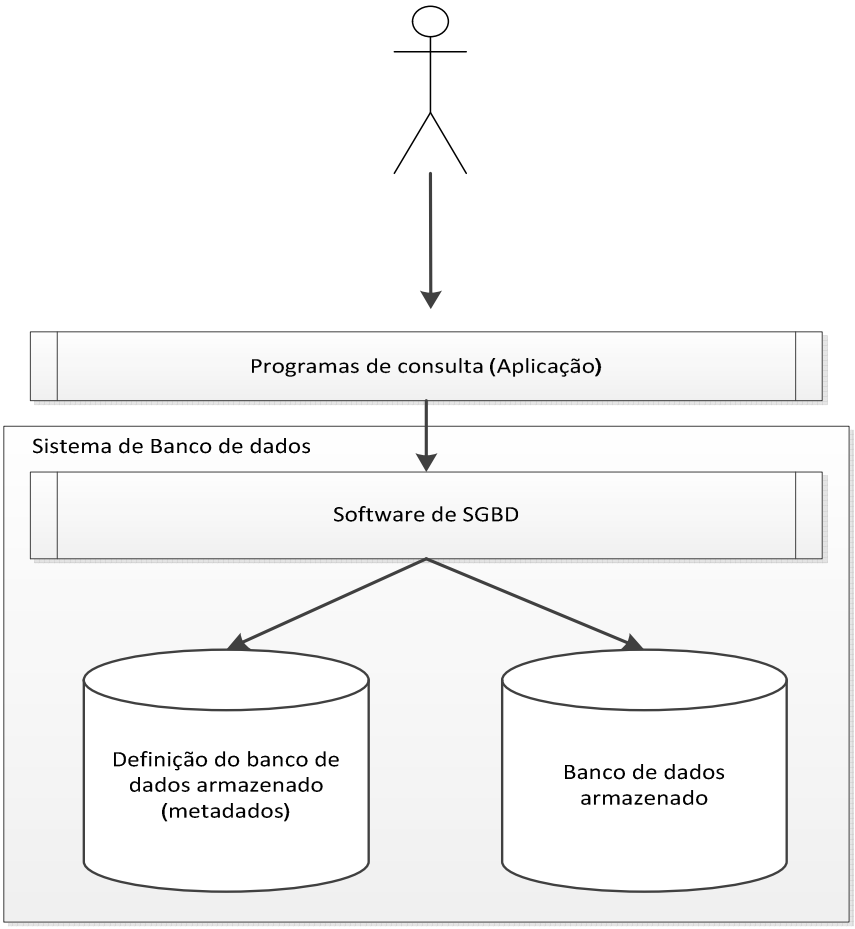


**Figura 6: SGBD e os Metadados.**

Conforme apresentado na Figura 6, os metadados são mantidos com os dados, pois suas informações são essenciais durante o acesso à base de dados utilizada. Os metadados disponibilizam informações sobre os dados, como descrição das tabelas e tipo dos dados, que são componentes do BD, facilitando assim a manipulação no BD pelo SGBD. Estes metadados são utilizados pelo próprio SGBD durante a manipulação dos dados no banco, sendo transparente para a aplicação que solicita os dados.

O SGBD busca realizar o gerenciamento do BD de forma transparente para a aplicação, utilizando tecnologias e métodos de forma a facilitar a manipulação dos dados. Esta facilidade de manipulação e gerenciamento dos dados do BD, oferecida

pelo SGBD, busca principalmente justificar sua utilização visto que um sistema não necessita obrigatoriamente do SGBD para manipular o BD. O BD pode ser utilizado diretamente pela própria aplicação, contudo esse acesso sem intermédio do SGBD torna a manipulação dos dados bem mais trabalhosa, pois todas as técnicas de gerenciamento de transações, controle de concorrência e recuperação de falhas, que serão apresentadas respectivamente nas seções 2.3.2 e 2.3.4, deverão ser gerenciadas pela própria aplicação.

A manipulação dos dados por meio do SGBD compreende, então, processos como consultas e atualizações das bases de dados, com o auxílio dos metadados.

O SGBD também proporciona a proteção do banco de dados, que contempla a proteção contra falhas de *hardware*, *software* e segurança contra acessos não autorizados. Esta proteção provê técnicas de manipulação dos dados da base de dados que permitem, ao haver falhas ou parada brusca, como uma queda de energia, manter o máximo da integridade dos dados do banco. O SGBD também implementa métodos de segurança que garantem a confidencialidade e integridade dos dados manipulados (CASANOVA et al. 1999, p.25), os quais serão melhor apresentados na seção **Gerenciamento de Transações em Bancos de Dados Distribuídos**.

## 2.3. Banco de Dados Distribuídos

Segundo Ozsu (1999, p.13), Banco de Dados Distribuídos (BDD) são uma coleção de vários bancos de dados logicamente inter-relacionados, distribuídos ao longo de um sistema de redes de computadores. Este tipo de banco de dados caracteriza-se por possuir várias bases de dados, podendo ser homogêneas ou heterogêneas:

- **BDDs homogêneos** comportam somente um tipo de banco de dados. Por exemplo, um *Cluster* de BD Microsoft SQL Server.
- **BDDs heterogêneos** comportam mais de um tipo de banco de dados. Por exemplo, um *Cluster* de banco de dados Microsoft SQL Server e BD Oracle.

A Figura 7 a seguir ilustra os exemplos citados, onde uma estrutura dos dois tipos de BDD é apresentada.

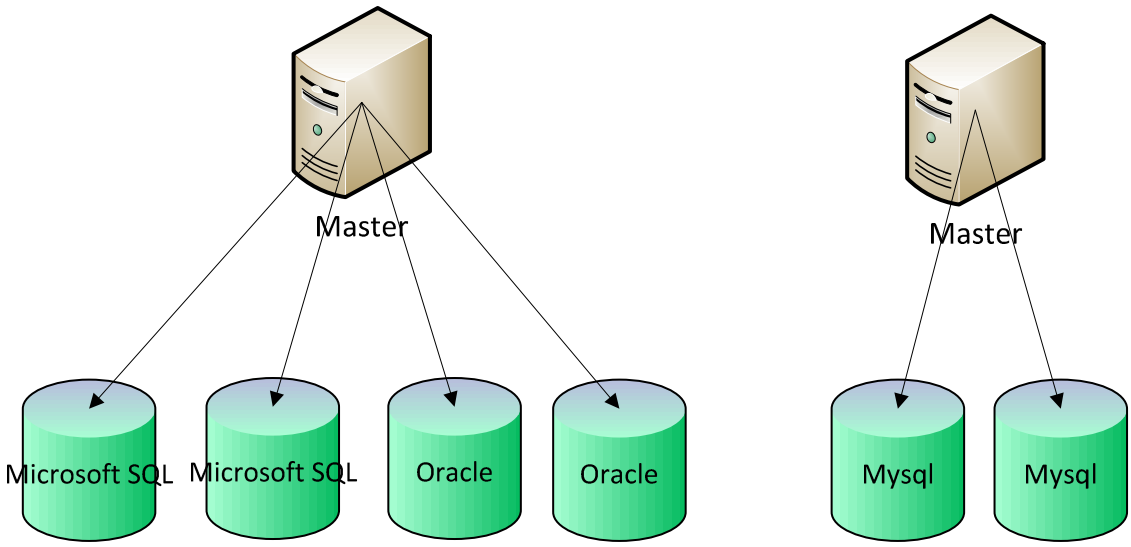


**Figura 7: BDD heterogêneos e homogêneos, respectivamente**

Conforme ilustrado na Figura 7, os BDDs heterogêneos são compostos por bases de dados de tipos diferentes de um *Cluster* e os BDDs homogêneos são formados pelos mesmos tipos de banco de dados no *Cluster*.

Segundo Navathe et al. (2011, p.589) “a tecnologia de BDD é o resultado de uma fusão de duas tecnologias: tecnologia de banco de dados e tecnologia de rede e comunicação de dados”. O processamento distribuído passou a ser possível com a utilização das redes de computadores, que forneceram a tecnologia necessária para interligação dos nós. Esta interligação permitiu a distribuição dos dados a serem processados a cada computador na rede.

A utilização de BDD traz consigo algumas vantagens, como:

- **Maior confiabilidade e disponibilidade** – Bancos distribuídos implementam isolamento de falhas. Assim, somente partes do banco são afetadas ao ocorrerem problemas em um sistema;
- **Maior desempenho** – O Sistema de Gerenciamento de Banco de Dados Distribuídos (SGBDD), que será apresentado mais detalhadamente no próximo tópico, distribui o banco de dados de forma a deixar partes do banco próximas de onde é mais requisitada ou importante, reduzindo assim principalmente o tráfego de dados nas transações (Alto desempenho em BD será mais detalhado na seção 2.4);

- **Escalabilidade** – Se torna muito mais simples seu crescimento tanto em quantidade de dados quanto em expansão dos recursos físicos.

Estes conceitos foram apresentados anteriormente na seção 2.1 e serão abordados novamente na seção 2.4, pois se tratam de conceitos gerais de sistemas distribuídos.

Um BDD é gerenciado de forma semelhante ao Banco de Dados Centralizado, por um sistema de gerenciamento chamado de Sistema de Gerenciamento de Banco de Dados Distribuídos (SGBDD), responsável por tornar a distribuição transparente para todos os usuários.

### 2.3.1. Sistema de Gerenciamento de Banco de Dados Distribuídos

Um sistema de BDD apresenta características semelhantes aos Bancos de Dados Centralizados (BDC), inclusive em seu gerenciamento. Segundo Casanova et al. (1999, p.11),

> "Sistemas de gerência de bancos de dados distribuídos estendem as facilidades usuais de gerência de dados de tal forma que o armazenamento de um banco de dados possa ser dividido ao longo dos nós de uma rede de comunicação de dados, sem que com isto os usuários percam uma visão integrada do banco."

O SGBDD reduz drasticamente tarefas de desenvolvedores de aplicações ao implementar transparência em sua utilização. Com a utilização do SGBDD os programadores não mais necessitam se preocupar com a complexidade da camada de comunicação com o BDD, já que o próprio sistema de gerência se encarrega de realizar todas as particularidades de um sistema distribuído, como o gerenciamento de transações, processamento de consultas e controle de concorrência e recuperação.

De modo geral um SGBDD pode ser comparado a um conjunto de SGBDs locais, interligados e gerenciados por um SGBD global em rede, conforme apresenta a Figura 8.

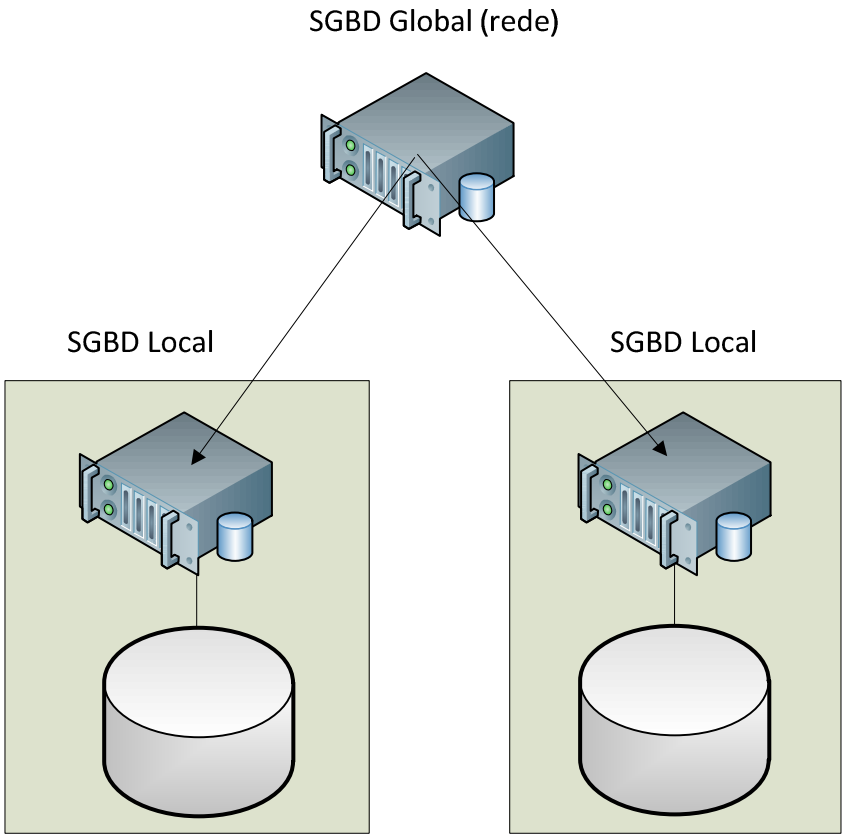


**Figura 8: SGBD Global e Local**

Conforme apresentado na Figura 8, o SGBD Local é gerenciado por outro SGBD considerado global, sendo que o SGBD global gerencia todo o conjunto de SGBDs locais através de uma rede de computadores. Estes dois SGBDs formam o SGBDD, que trabalha de forma a apresentar os resultados das suas partes de forma unificada.

Os SGBDs locais são centralizados e gerenciam de forma autônoma sua base de dados local, exceto quando recebem solicitações de usuários locais em relação a dados que estão em outro nó. Já o SGBD Global trabalha como uma camada entre o sistema e os SGBD locais, gerenciando todas as requisições e encaminhando aos SGBD locais dos nós correspondentes.

Algumas das funcionalidades que o SGBD, centralizado e distribuído, realiza e que serão abordadas neste trabalho é o Gerenciamento de Transações, Processamento de Consultas e o Controle de Concorrência e Recuperação, que serão apresentados nas próximas seções.

### 2.3.2. Gerenciamento de Transações em Bancos de Dados Distribuídos

O gerenciamento de transações é realizado pelo SGBD (CASANOVA et al. 1999 p.14). Segundo Navathe (2011, p.611), as transações possuem propriedades chamadas de propriedades ACID (Atomicidade, Consistência, Isolamento e Durabilidade), que são atribuídas pelos métodos de recuperação e controle de concorrência do SGBD, que será apresentado na seção 2.3.4:

- **Atomicidade** – é a propriedade que define uma transação ser atômica, de modo que ela deva ser executada no seu total ou não ser executada, não podendo ser realizado somente parte da transação;

- **Preservação da consistência** – é a propriedade que define uma transação de, ao ser executada, preservar o estado consistente do banco de dados, buscando assim garantir integridade dos dados manipulados;

- **Isolamento** – é a propriedade que define uma transação de não ser interferida por outras transações simultaneamente, buscando garantir confidencialidade dos dados manipulados. A confidencialidade é necessária para evitar que os dados sejam modificados antes de ser recebida em seu destino;

- **Durabilidade ou permanência** – todas as mudanças do banco de dados pelas transações confirmadas precisam ter persistência, desde que não haja falha na transação.

O gerenciamento de transações, juntamente com o gerenciador de controle de concorrência e recuperação do SGBD, que será apresentado na seção 2.3.4, é responsável por garantir as propriedades ACID das transações. Estas propriedades contribuem para garantir a confidencialidade e integridade dos dados manipulados.

No contexto de BDD, é preciso notar que, no gerenciamento de transações distribuídas, é adicionado o gerenciador de transação global, além do gerenciador de transações local, já existente. Este gerenciador permite que um *nó* local assuma temporariamente o estado de global para coordenar a execução das operações em banco de dados entre vários outros *nós* distribuídos, conforme exemplificado na Figura 9:

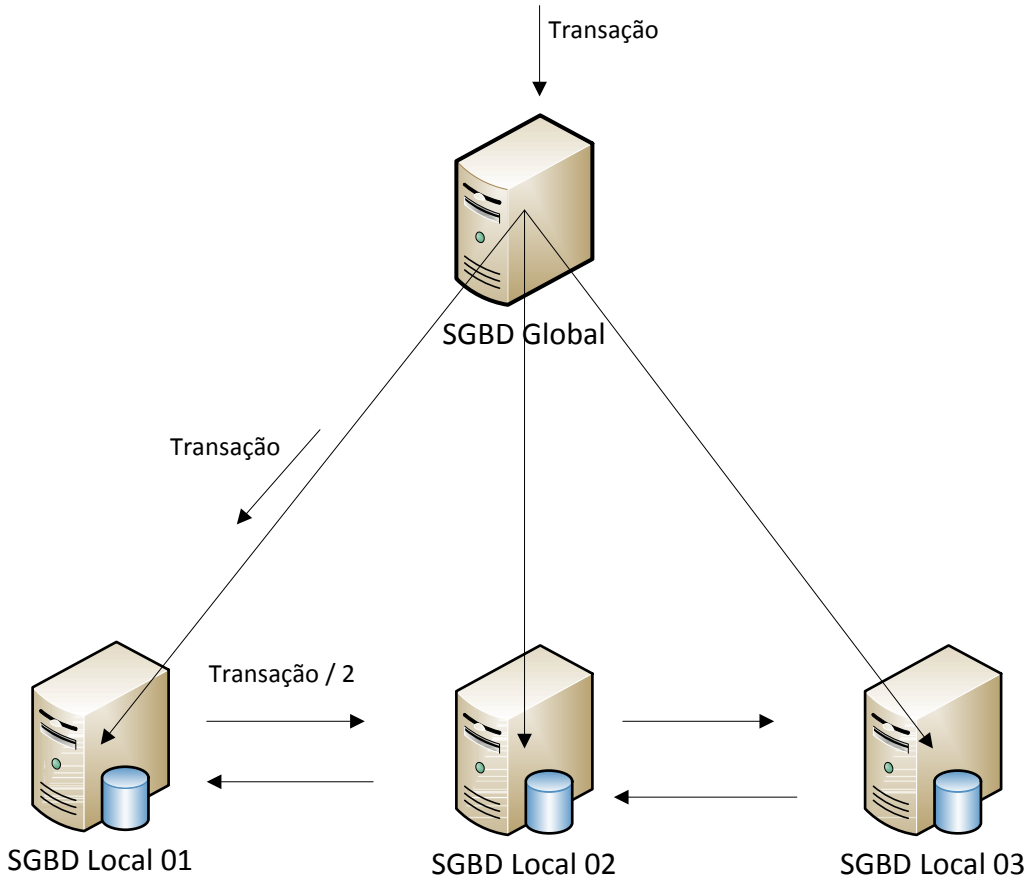


**Figura 9: SGBD local assumindo temporariamente o papel de SGBD global**

Conforme apresentado na Figura 9, uma solicitação recebida pelo nó que comporta o SGBD global é encaminhada ao nó local *SGBD Local 01*, onde se encontram os dados solicitados. O nó local tem autonomia sobre seus dados, contudo, havendo a necessidade deste buscar mais dados em outros nós, assumirá temporariamente funções do nó global a fim de solicitar diretamente estes dados de outro nó, através da fração de transação "Transação /2" ao *SGBD Local 02*, para compor o resultado da transação solicitada.

Os gerenciadores de transações disponibilizam funcionalidades em forma de transações SQL para as aplicações (NAVATHE 2011, p. 611), como:

- **BEGIN_TRANSACTION** – representa o início da transação sendo executada;
- **READ e WRITE** – definem as transações de escrita e leitura no BD;
- **END_TRANSACTION** – sinaliza que a transação de escrita e leitura terminou e marca o final da transação sendo executada. Neste ponto, se houver

alguma violação ou falha, a transação pode ser abortada antes de realizar o *commit*;

- **COMMIT_TRANSACTION** – define que a transação foi bem sucedida e sinaliza que pode ser realizado o *commit* na base de dados, ou seja, as alterações realizadas pela transação são persistidas no banco de dados;

- **ROLLBACK** ou **ABORT** – sinalizam que houve falha na operação; neste caso, quaisquer alterações realizadas no BD precisarão ser desfeitas.

A Figura 10 apresenta as mudanças de estado de uma transação.

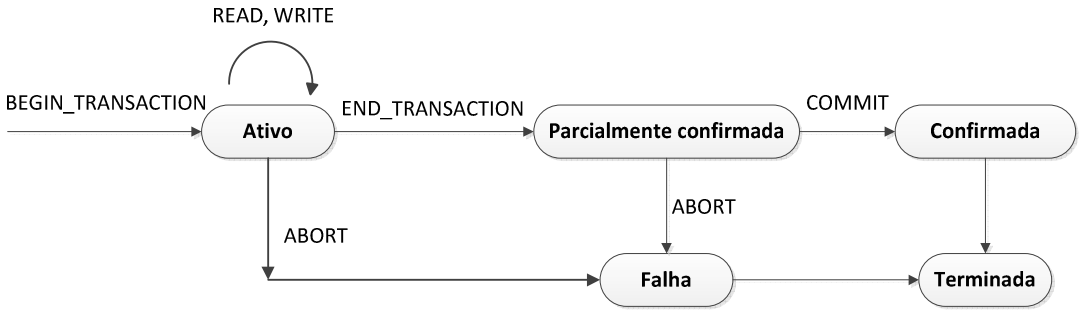


**Figura 10: Mudanças de estado de uma transação (NAVATHE 2011, p. 506)**

A Figura 10 apresenta as mudanças de estado de uma transação durante sua execução:

1- A transação é iniciada com estado “ativo”, neste estado ela executa suas operações de escrita (*write)* e leitura *(read)*, após finalizar as operações ela entra no estado “parcialmente confirmada”. Se houver erros neste estado a transação passará para o estado de “falha”;
2- No estado “parcialmente confirmada” os protocolos de recuperação verificam se não haverá problemas no momento de registrar permanentemente as operações, após esta verificação ser bem sucedida a transação passa para o estado “confirmada”. Se houver erros neste estado a transação passará para o estado de “falha”;
3- No estado de “confirmada” as mudanças realizadas pela transação são gravadas permanentemente no BD. Ao finalizar o processo de gravação a transação sai do sistema e recebe o estado “terminada”.

Uma transação pode receber o estado de “falha” se uma das verificações encontrar problemas, abortar ou se houver falhas no sistema. Estando neste estado, se a transação já realizou mudanças na base de dados, o SGBD realiza ações do controle de recuperação, que será apresentado na seção 2.3.4, para executar processos de desfazer as mudanças no BD.

### 2.3.3. Processamento de Consultas Distribuídas

Segundo Casanova et al. (1999, p.36), processamento de consultas distribuídas corresponde à tradução de pedidos, formulados em uma linguagem de alto nível, para sequências de ações sobre os dados armazenados nos vários bancos de dados locais. Processamento de consultas realiza então uma tradução das requisições SQL e as transcreve em um formato de baixo nível para ser entendido e processado pelo BDD.

O processamento de consultas distribuídas tem como uma de suas finalidades realizar o mapeamento das consultas de alto nível para um grupo de operações de BD em partes fragmentadas. Uma consulta de BDD é então processada nos estágios (NAVATHE, 2011, p.605):

1. Mapeamento de consultas;
2. Localização;
3. Otimização global da consulta; e
4. Execução da consulta local.

Cada um destes estágios é descrito nas seções a seguir.

#### 2.3.3.1. Mapeamento de consultas

Neste estágio a consulta é lida e traduzida de linguagem de consulta para linguagem de álgebra relacional. Esta tradução não leva em consideração a replicação dos dados, sendo normalizada, analisada quanto a erros de semântica, simplificada e reescrita em forma algébrica. Este estágio é composto pelos seguintes sub-estágios:

- **Normalização –** transcreve a consulta em consulta normalizada, utilizando operadores lógicos como *AND* e *OR*. Exemplo:

  Forma original:

```
Select nome from pessoa where pessoa.nome = "rogerio" and
pessoa.idade = 30
```

Forma normalizada:

```
(pessoa.nome = rogerio) ^ (pessoa.idade = 30)
```

- **Análise** – verifica a consulta normalizada a fim de localizar erros de semântica e corrigi-los. Exemplo:

  Forma de consulta com erro:

```
Select valor from vendas where vendas.data_venda < 2013
```

  No exemplo citado o campo data_venda é do tipo *datetime* e sua comparação é com um valor do tipo inteiro (número).

- **Simplificação** – simplifica a consulta eliminando possíveis redundâncias. Exemplo:

  Forma de consulta redundante:

```
Select nome from pessoa where pessoa.idade < 20 and
pessoa.idade < 25 and pessoa.sexo = masculino
```

  Forma simplificada:

```
Select nome from pessoa where pessoa.idade < 20 and
pessoa.sexo = masculino
```

- **Reescrita** – reescreve a consulta em álgebra relacional. Exemplo:

  Consulta:

```
Select idade from pessoa where nome = "pedro"
```

  Álgebra relacional:

$$\prod_{idade}(\sigma_{nome=pedro}(pessoa))$$

#### 2.3.3.2. Localização

Neste estágio ocorre o mapeamento dos dados que serão analisados pela consulta. Esse mapeamento leva em consideração a fragmentação dos dados no BDD

O resultado do mapeamento da consulta é utilizado para se medir o custo de execução de cada fragmento de consulta, gerando assim estratégias de execução.

Cada uma das estratégias de execução geradas armazenam estes custos de execução, o que possibilita a execução do próximo estágio.

#### 2.3.3.3. Otimização global da consulta

Neste estágio ocorre a seleção da melhor estratégia dentre as geradas na etapa anterior. A melhor estratégia normalmente consiste em ser aquela com menor tempo de execução, sendo obtido através de cálculos estimados de utilização dos recursos como processador, I/O de disco e consumo de rede, já que BDD são interligados por uma rede de computadores.

#### 2.3.3.4. Execução da consulta local

Neste estágio ocorre a execução da consulta em todos os nós que comportam os fragmentos necessários da consulta. Sendo que cada fragmento é visto em seu nó como uma consulta local, sendo seu resultado encaminhado para o SGBDD global após sua execução para ser reagrupado.

Resumindo os processos, pode-se verificar que uma consulta é analisada, otimizada, fragmentada e é verificado onde cada fragmento será processado, distribuído, processado e reconstruído novamente no SGBDD global.

### 2.3.4. Controle de Concorrência e Recuperação

Os dados de um BDD podem sofrer comumente acessos simultâneos, necessitando assim de que haja algum recurso de garantia e consistência [dos dados]. Segundo Casanova et al. (1999, p.14), “controle de concorrência visa a garantir que, em toda execução simultânea de um grupo de transações, cada uma seja executada como se fosse a única do sistema”. Isso significa que as transações concorrentes não devem gerar anomalias entre si, ou seja, uma transação não pode modificar informações que estão sendo modificadas por outra transação (Isolamento), causando assim inconsistência nos dados.

Existem alguns mecanismos para o controle de concorrência, como o *locking* (bloqueio) e o *timestamping* (identificador).

O mecanismo de *locking* ou *lock* é o mecanismo de controle de concorrência mais amplamente difundido (TANENBAUM, 1992 apud SHIBAYAMA, 2004, p.30). Este mecanismo consiste no bloqueio dos recursos que serão utilizados pelo

processo da transação, como recursos de escrita ou leitura no BD. Contudo, outros processos não poderão acessar tais recursos até os recursos serem liberados (bloqueio ser desfeito). Exemplo:

Um sistema recebe duas transações chamadas de “Transação A” e “Transação B”, nesta sequência. As duas transações necessitam utilizar o recurso chamado de “Recurso X” em sua execução. Por ordem de chegada a execução será esta:

Transação A inicia
Recurso X bloqueado
Transação A utiliza Recurso X
Recurso X desbloqueado
Transação A finalizada

Transação B iniciada
Recurso X bloqueado
Transação B utiliza Recurso X
Recurso X desbloqueado
Transação B finalizada

Conforme apresentado, a “Transação A” realiza o bloqueio do “Recurso X”, adquirindo assim acesso restrito ao recurso necessário para sua utilização. Somente após a utilização do recurso pela “Transação A”, o “Recurso X” é então desbloqueado para ser utilizado por outra transação, neste caso, a “Transação B”.

O mecanismo de *timestamping* (TS) consiste na designação de um valor, chamado de *timestamping* ou *timestamp*, para cada transação (SILBERSCHATZ, et al. 1999, p.627). A transação que primeiro chegar ao sistema terá prioridade na execução. São utilizadas algumas formas de se obter o TS em sistemas distribuídos:

1. A hora do relógio de um servidor global, ou seja o TS de uma transação é a exata hora que ela entra no sistema. A necessidade de se ter um servidor global é devida a possíveis diferenças de horários entre os nós;
2. Utilizar um contador para definir a numeração TS da transação quando entra no sistema.

O TS de uma transação determina a ordem de execução, assim o sistema precisa garantir a execução serializada da transação de acordo com seu TS.

Contudo, em sistemas distribuídos, o TS de transações paralelas pode apresentar inconsistências. O custo de se garantir desempenho de comunicação entre os nós é elevado, possibilitando assim pequenos atrasos nas trocas de informações e, com isso, inconsistências nas filas das transações. Um nó pode estar com seu horário diferenciado em alguns segundos de outro nó, causando assim inconsistência na fila de transações.

Segundo Silberschatz (1999, p.511), a recuperação de falhas de transação representa a restauração do banco de dados a seu último estado consistente antes do momento da falha. A recuperação busca manter a integridade do banco de dados mesmo que isso represente a perda das últimas transações realizadas. Duas técnicas podem ser citadas para recuperação de falhas de transação: atualização adiada e atualização imediata (NAVATHE, 2011, p.644):

A técnica de **atualização adiada** consiste em não atualizar fisicamente o banco de dados até que a transação alcance seu ponto de confirmação, somente após alcançar este estado as atualizações são registradas no banco de dados. Enquanto a transação está a caminho de seu estado de confirmação todas as suas atualizações de transação são registradas temporariamente em um buffer do SGBD e registradas no log. Caso uma transação falhe antes de atingir seu estado de confirmação, ela não alterou o BD e, portanto, não haverá a necessidade de se executar o UNDO (Desfazer). O UNDO é uma função do SGBD que realiza a ação de desfazer algo no banco que, neste caso, seria desfazer as alterações gravadas no BD de forma persistente. Um REDO (Refazer) pode ser necessário para se reiniciar a transação gravada no log e que falhou. O REDO é uma função do SGBD que realiza a ação de refazer algo no banco que, neste caso, seria refazer a transação. A técnica de atualização adiada é também conhecida como algoritmo NO-UNDO/REDO (SILBERSCHATZ 1999, p.522).

A técnica de **atualização imediata** consiste em atualizar o BD por algumas transações ou por todas as transações antes que elas alcancem seu estado de confirmada. Contudo estas transações também necessitam ser gravadas em log antes de serem gravadas diretamente no BD, a fim de possibilitar a recuperação se houver falha. Havendo falha na situação em que somente algumas gravações no BD foram realizadas antes da confirmação da transação, será necessário executa tanto

o UNDO quanto o REDO. Portanto, esta técnica é conhecida como algoritmo UNDO/REDO. Havendo falha na situação em que todas as gravações no BD foram realizadas antes da confirmação da transação, será necessário executar apenas o UNDO e não o REDO, sendo conhecido assim como algoritmo UNDO/NO-REDO (NAVATHE, 2011, p.644):.

O resultado da recuperação de uma falha do BD pelo SGBD deve ser igual ao estado do BD quando não há falha, mantendo assim a atomicidade do banco.

O controle de concorrência e recuperação são então controles implementados pelo SGBD, buscando garantir a consistência do BD. Sendo de extrema importância para a utilização em sistemas distribuídos, devido à grande quantidade de transações concorrentes em suas bases de dados que podem estar distribuídas geograficamente.

## 2.4. Alto Desempenho em Bancos de Dados Distribuídos

Segundo SHIBAYAMA (2004, p.15), dois pontos trazem grandes vantagens para obter alto desempenho em BDD: fragmentação do BDD e paralelismo.

A **fragmentação do BDD** consiste em alocar as partes em locais próximos de seus pontos de utilização mais comuns, reduzindo significativamente o tempo de comunicação entre os nós onde estão armazenados os dados e seus sistemas, além de permitir cada nó manipule uma parte do BD facilitando sua administração. A possibilidade de autonomia local é frequentemente uma das maiores vantagens de BDD (CASANOVA et al. 1999 p.03). Por exemplo, considerando que uma empresa possui um BDD contendo um de seus nós na cidade de Araguaína e o outro nó do mesmo banco na cidade de Palmas. Conforme será apresentado na Figura 11.

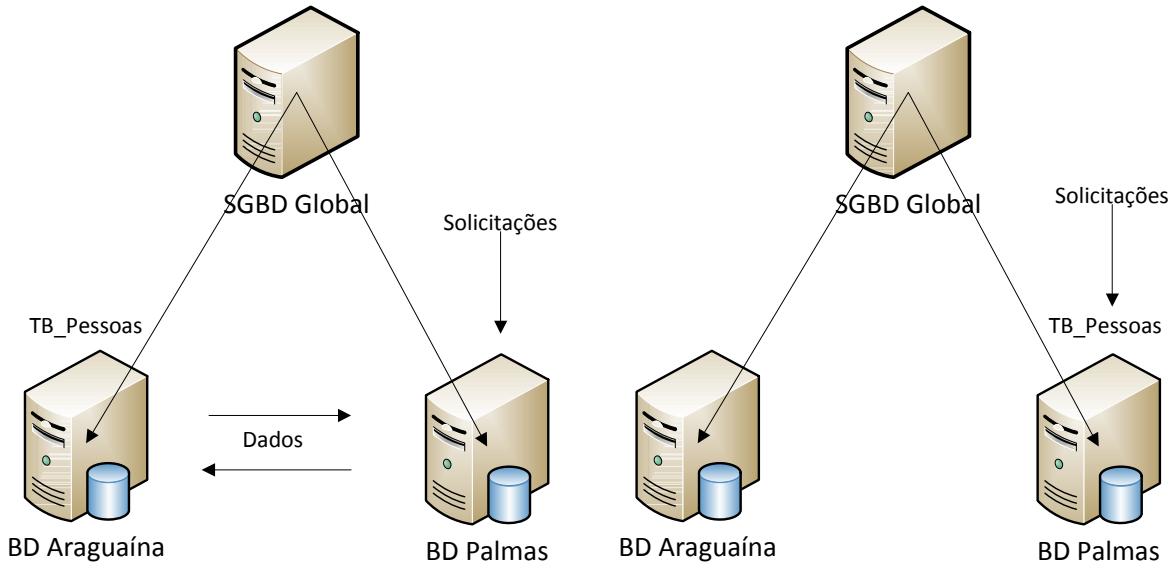


**Figura 11: Balanceamento de carga do SGBDD**

Como mostrado na Figura 11, o SGBDD deste banco analisará os acessos a sua base distribuída para identificar os consumos de recursos em relação às operações realizadas, identificando assim que a “tabela Pessoas” do banco está sendo mais acessada pela cidade de Palmas. Então o SGBDD alocará a “tabela Pessoas” no nó da cidade de Palmas, o que irá, principalmente, reduzir o tráfego de rede da cidade de Palmas para Araguaína.

O **paralelismo** entre consultas e intraconsultas possibilita executar várias consultas em paralelo, como também, fragmentar uma consulta em várias subconsultas ou fragmentos de consulta para serem executadas em locais diferentes. O paralelismo intraconsulta, segundo Wada et al. (2003, p.15), permite reduzir o tempo de execução de uma operação, pois divide a consulta em fragmentos menores permitindo serem processadas por diferentes processadores. Em BDD estas operações são divididas entre os nós e não entre os processadores como em um BDC. Por exemplo, um sistema de controle de alunos e turmas de uma universidade recebeu uma consulta de busca em sua base de dados distribuída. Esta consulta está realizando a busca de todos os alunos de todas as turmas do BDD. Conforme será apresentado na Figura 12:

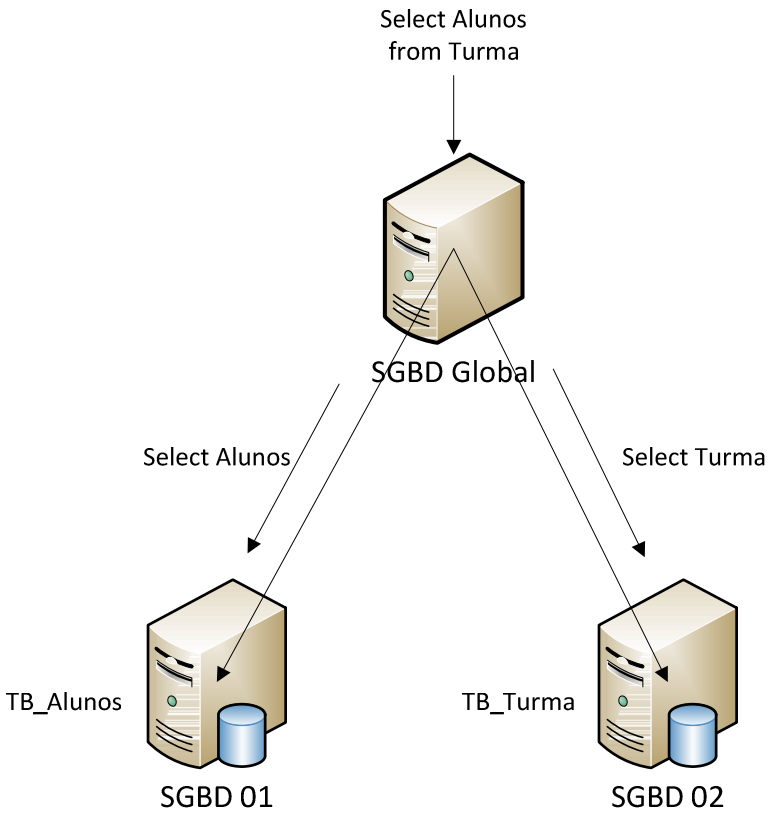


**Figura 12: Fracionamento de transações**

Como ilustrado na Figura 12, o SGBDD quebra a consulta em subconsultas para serem processadas por mais nós. Essa fragmentação reduz o tempo de busca dos dados no BD local, permitindo assim uma melhora de desempenho na execução de transações.

A escalabilidade, característica presente em sistemas distribuídos, é uma forte aliada na justificativa da utilização de cluster em BD voltados para ganho de desempenho.

A utilização de BDD + SGBDD na rede Social *Conecta* possibilitará ganho em desempenho do sistema em questão. Permitindo assim que o sistema suporte um maior número de usuários e transações simultâneas em relação a capacidade atual. Pode-se considerar ainda que o ganho em desempenho também traz ganho em disponibilidade, pois um sistema que tem seus limites de desempenho aumentados possibilitará menor chance de se tornar indisponível por sobrecarga de utilização.

A possibilidade de expandir a capacidade de processamento do sistema também (Escalabilidade) é um ponto positivo na utilização de BDD + na rede Social *Conecta*. Acompanhando o crescimento de utilização do sistema poderá ocorrer a

expansão da capacidade de processamento alocando-se mais nós ao Cluster existente.

# 3 MATERIAIS E MÉTODOS

Nesta seção serão apresentados os materiais e métodos utilizados para se desenvolver este trabalho.

## 3.1. Materiais

O *software Conecta* foi o ambiente utilizado para se realizar os testes de estresse. Utilizou-se a ferramenta de virtualização *VMware Workstation* para criar os ambientes virtuais de homologação para a realização dos experimentos, sendo um centralizado e um distribuído. O *Cluster* foi criado com a ferramenta *MySQL Cluster*. A ferramenta *JMeter* foi utilizada para simular os acessos durante os testes de estresse ao BD. O *MySQL Enterprise Monitor* foi utilizado para capturar os dados dos recursos diretamente do BD da aplicação, evitando assim a inconsistência destes valores. O *Medidor de Desempenho do Windows* também foi utilizado para capturar os dados de consumo de recursos, contudo estes dados são de todo o sistema e não somente do BD. A seguir, cada ferramenta será apresentada com mais detalhes.

### 3.1.1. *Conecta*

A Fábrica de Software do CEULP/ULBRA disponibiliza um *software* que utiliza sua base de dados de maneira centralizada, sendo este software uma rede social acadêmica com o nome de *Conecta*. O *Conecta* é uma rede social para gestão do conhecimento. Implantada no CEULP/ULBRA para servir como canal de comunicação e interação virtual entre alunos e professores.

Este *software* está disponibilizado através de um servidor Apache, e sua base de dados está alocada em um servidor *MySQL*. O *Conecta* utiliza algumas tecnologias de ganho de desempenho como o *Memcached,* que armazena informações frequentemente usadas para evitar vários carregamentos da mesma informação em memória ou disco.

O *Conecta* disponibiliza algumas funcionalidades como, lista de amigos da rede, possibilita postagens de conteúdos como textos e imagens, permite comentar

postagens de amigos e criação e gerenciamento de grupos públicos e privados de usuários.

### 3.1.2. VMware Workstation

O *VMware Workstation* é um *software* para criação e gerenciamento de máquinas virtuais com suporte às principais plataformas de mercado (VMware 2013). Ele possibilita criar e gerenciar máquinas virtuais trazendo um desempenho aos seus *hosts* bem próximo do real, em máquinas físicas, e dando suporte a várias plataformas, como mostrado na Figura 13.

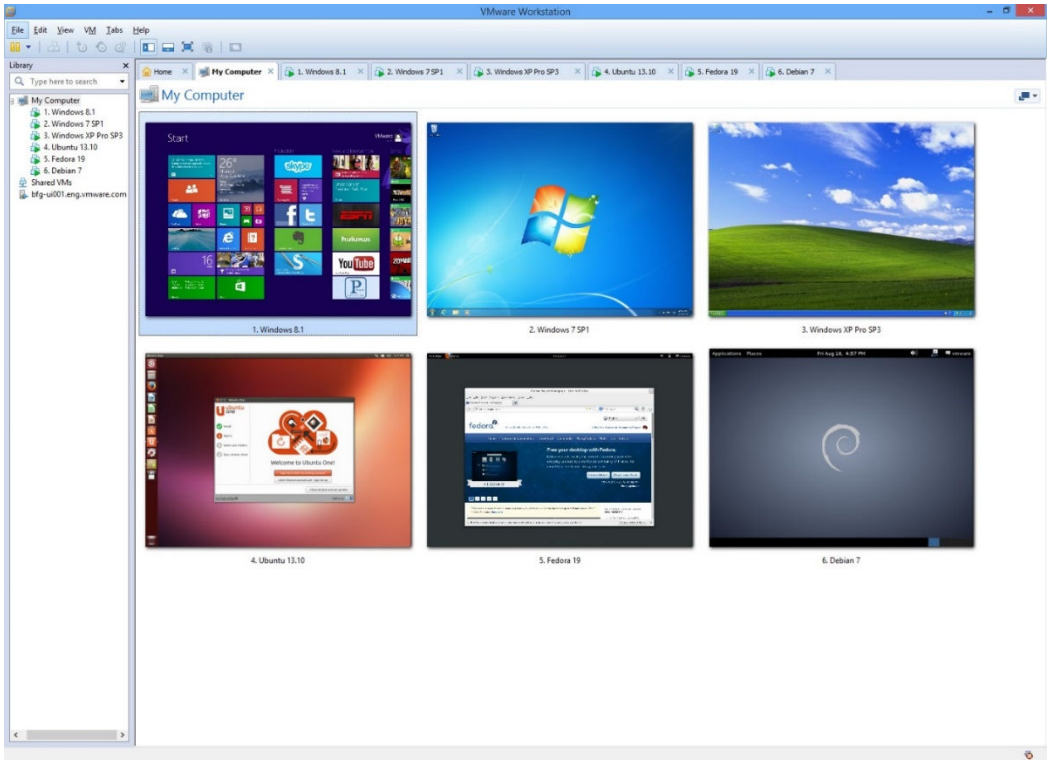


**Figura 13: VMware Workstation (VMware 2013, online)**

A Figura 13 demonstra que o *VMware Workstation* dá suporte a várias plataformas de mercado, tanto proprietárias quanto de código aberto, como *Microsoft Windows* e variadas distribuições do *Linux*.

Esta ferramenta foi utilizada para criar e manter os ambientes virtuais de homologação, permitindo assim que todos os experimentos pudessem ser realizados

sem riscos de se danificar o real ambiente do sistema *Conecta*. Serão apresentados mais detalhes dos ambientes no capítulo 4.

### 3.1.3. *MySQL* Cluster

O *MySQL Cluster* é um Banco de Dados ACID, escalável em tempo real, compatível com banco de dados transacional e que combina a disponibilidade de 99,999% com o baixo custo de aplicações de código aberto (*MySQL* 2013a).

O *MySQL Cluster* é um banco de dados distribuído, podendo ser composto por vários nós (*MySQL* 2013b):

1. Nó SQL – é o nó do servidor *MySQL* que responde como o banco *MySQL*; é nele que são executadas as consultas SQL;
2. Nó de dados – é o nó onde são armazenados os dados do banco, podendo este nó ser *single-thread* (voltado para processadores que trabalhem somente com uma *thread*) ou *multi-thread* (voltado para processadores que trabalhem com várias *threads*);
3. Nó de gerenciamento – é o nó que permite o gerenciamento dos nós do cluster e também é responsável por monitorar a disponibilidade dos nós;
4. Nó API – é o nó cliente responsável por acessar os dados do banco através de uma API em vez de usar o SQL. Este nó não será utilizado no presente trabalho.

A junção destes componentes forma o *MySQL Cluster*, de modo que disponibilidade e desempenho estão sempre presentes em todas as estruturas formadas pelos nós.

Nem toda aplicação obtém ganho de desempenho utilizando *MySQL Cluster*. Devido as tabelas do *MySQL Cluster* serem do tipo *NDB Cluster* e não *InnoDB*, e nem todas as características do tipo *InnoDB* ainda serem suportadas pela *engine NDB Cluster*, algumas aplicações podem apresentar perdas significativas de desempenho se utilizarem este ambiente. Devido à necessidade de alto desempenho de banco de dados as aplicações ideais para o uso do *MySQL Cluster* são (*MySQL* 2013c):

- Plataformas de negociação / sistemas
- Gateways de pagamento
- Gerenciamento de perfil de usuário
- Vários jogos online
- Serviços IMS
- DHCP para o acesso de banda larga
- VoIP e videoconferência

O *MySQL* Cluster foi utilizado para distribuir a base de dados da aplicação *Conecta*, sendo possível a realização dos experimentos de desempenho na aplicação com foco no desempenho de sua base de dados distribuída.

### *3.1.4. JMeter*

O software *JMeter* é uma aplicação de código aberto projetado para carregar testes em ambientes web (*Apache* 2013). Ele tem capacidade de carregar vários tipos de testes em servidores como:

- *Web – HTTP, HTTPS;*
- *FTP;*
- Banco de dados via *JDBC;*
- *LDAP;*
- *SMTP, POP3* e *IMAP*; e
- *TCP.*

Os tipos de testes suportados pela ferramenta podem ser (Lagares 2013):

- Estresse – Busca estressar o sistema afim de se medir a capacidade máxima de sua utilização mediante condições adversas de *software* ou *hardware*.
- Carga – Busca medir o desempenho do sistema à medida que seu número de utilizadores aumenta.

O JMeter foi utilizado no presente trabalho na execução dos testes de estresse ao sistema e seu BD, sendo que este foi usado para fazer testes de estresse para se entender o comportamento do software durante requisições.

### 3.1.5. *MySQL* Enterprise Monitor

O *MySQL* Enterprise Monitor é um sistema de monitoramento de bancos de dados *MySQL* (*MySQL* 2013d). Ele utiliza-se de um agente chamado de *MySQL* Enterprise Agent, que é executado dentro do banco de dados *MySQL*, para capturar e repostar várias informações dobre a saúde e desempenho do BD.

Através de gráficos esta ferramenta está constantemente informando e alertando sobre pontos críticos dos bancos que está monitorando, conforme será apresentado na Figura 14.

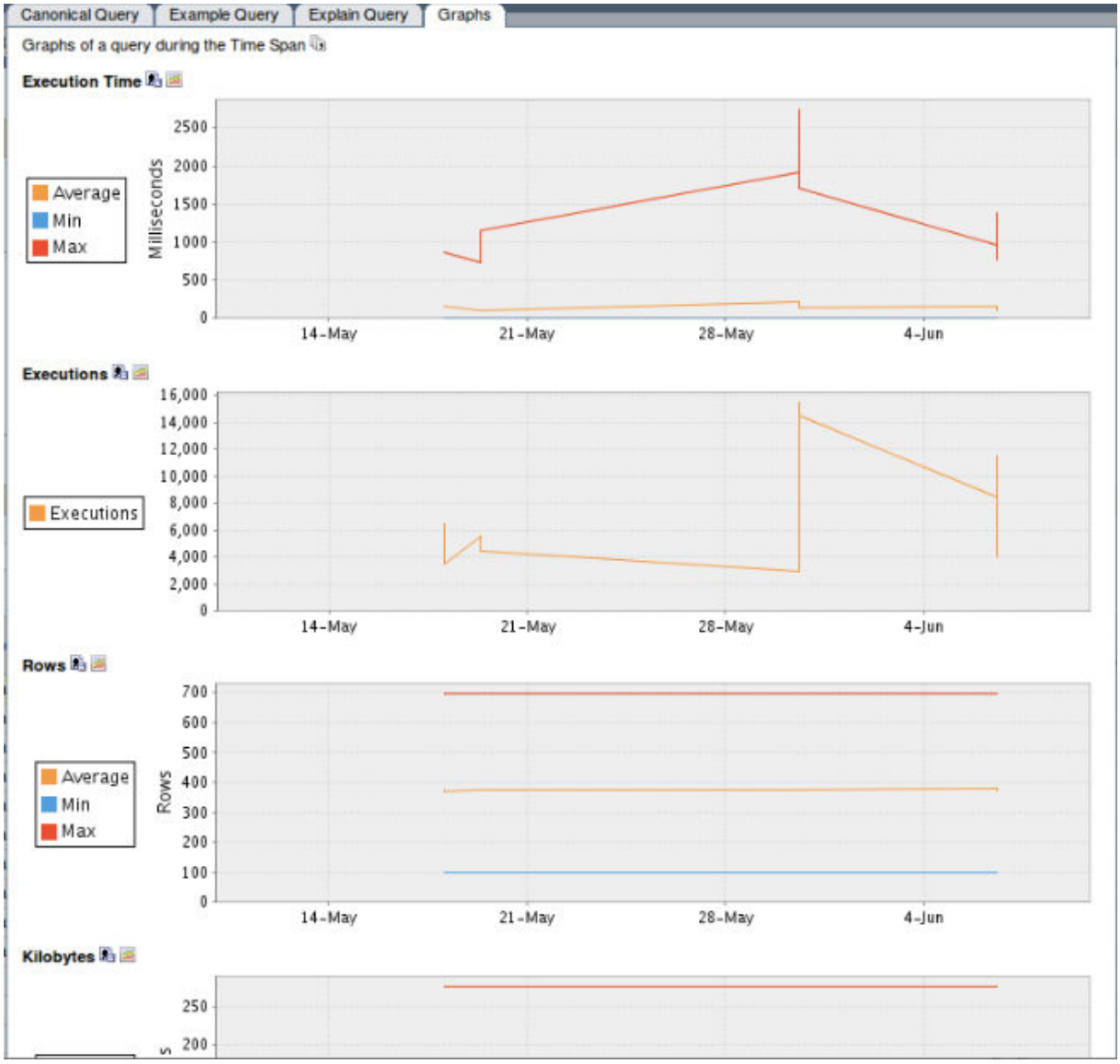


**Figura 14: *MySQL* Enterprise Monitor (*MySQL* 2013, online)**

Como apresentado na Figura 14, a ferramenta disponibiliza vários gráficos de monitoramento do status dos banco de dados. Dentre o conjunto de itens monitorados pela ferramenta através de seu agente, está o consumo de recursos físicos do equipamento pelo BD. Esta foi a principal ferramenta utilizada na captação do consumo de recursos físicos pelo banco de dados durante os experimentos realizados nos dois ambientes de homologação.

### 3.1.6. Medidor de desempenho do Windows

O medidor de desempenho do Windows é uma ferramenta para exibição de dados de desempenho em tempo real ou armazenamento em arquivos de log (Microsoft 2013). Ele possibilita a verificação de dados de desempenho em gráficos, histogramas ou relatórios. Os dados coletados podem ser apresentados em gráficos temporais permitindo assim a visualização do consumo de recursos no decorrer do tempo de utilização, conforme apresentado na Figura 15.

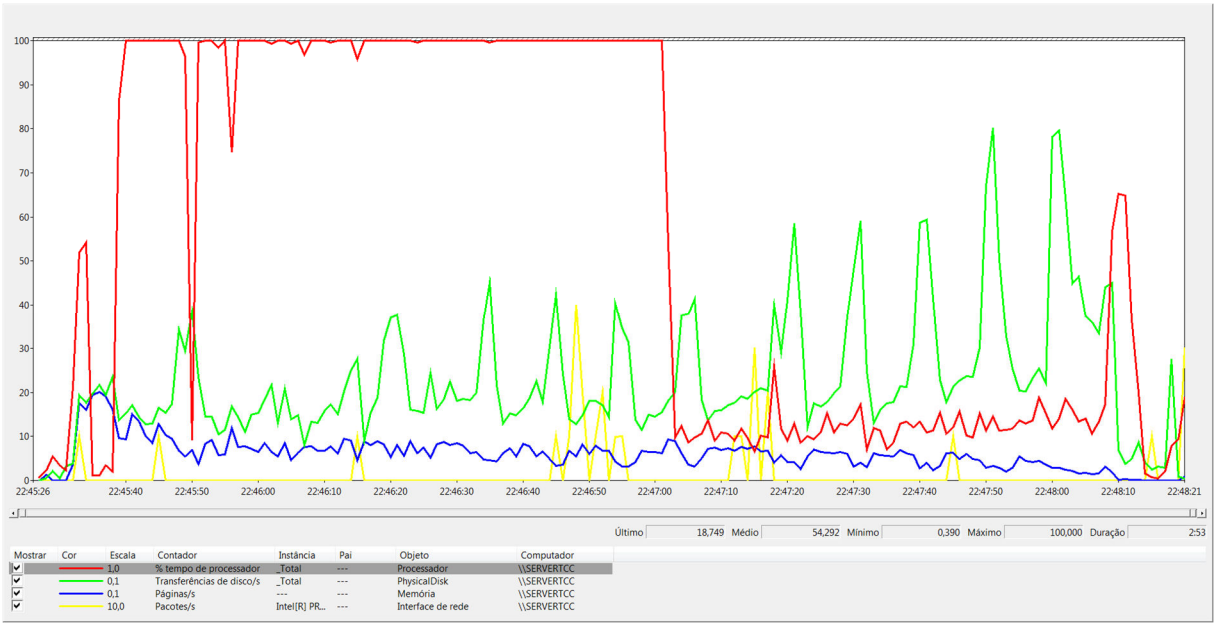


**Figura 15: Medidor de Desempenho do Windows**

Conforme apresentado na Figura 15, o medidor de desempenho exibe graficamente os dados dos coletores de desempenho do Windows como: porcentagem de uso do processador, transferências de disco por segundo, páginas de memória por segundo e pacotes por segundo transmitidos pela rede.

O Medidor de Desempenho do Windows foi utilizado neste trabalho para captação de dados sobre consumo dos recursos físicos do sistema testado que não puderam ser captados pela ferramenta *MySQL* Enterprise Monitor, ou seja, a quantidade de memória RAM consumida durante os experimentos.

## 3.2. Métodos

Para elaboração do presente trabalho foram realizados os seguintes passos:

1. Levantamento inicial da atual situação do Banco de Dados *Conecta*;
2. Estudo e Escrita da Revisão de Literatura;
3. Correção da escrita da Revisão de Literatura e entrega do Projeto;
4. Criação de dois ambientes de homologação (Centralizado e Distribuído), para realização dos experimentos;
5. Pesquisa e Estudo da Ferramenta de Medição do Banco de Dados;
6. Simulação dos acessos do software *Conecta* em sua base de dados nos dois ambientes;
7. Coleta do consumo de recursos de banco de dados do software *Conecta* nos ambientes de homologação;
8. Comparativo dos resultados de desempenho das simulações;
9. Escrita dos resultados encontrados; e
10. Correção da escrita dos Resultados Encontrados e entrega do Projeto.

Para facilitar o entendimento, estes passos serão detalhados nos parágrafos seguintes.

No levantamento inicial da atual situação do Banco de Dados *Conecta* foi realizado um estudo preliminar da capacidade de suportar requisições simultâneas de usuários a Rede Social *Conecta* à medida que esse número de usuários aumenta. Este estudo preliminar buscou demonstrar como a aplicação e seu banco de dados se comportariam com um número crescente de acessos. Esta capacidade foi expressada através do consumo de recursos (CPU, Disco e Rede) pelas transações, a fim de se obter os valores aproximados de consumo de recursos do servidor pelos usuários conectados. Foram então realizados alguns testes de estresse no sistema, através da ferramenta *Jmeter*, que simula alguns grupos de usuários realizando requisições simultaneamente. Os grupos de usuários foram compostos por 01, 10, 20 e 50 usuários. Por meio destes experimentos, foi possível estimar a quantidade de recursos necessária ao servidor para suportar uma quantidade de 5000 a 7000 usuários simultâneos, quantidade estabelecida juntamente com a coordenação da Fábrica de Software.

No Estudo e na Escrita da Revisão de Literatura foram abordados conceitos necessários ao entendimento do referido trabalho, a saber:

1. Banco de Dados;
2. Bancos de Dados Distribuídos;

3. Sistema de Gerencia de Banco de Dados;
4. Sistema de Gerencia de Banco de Dados Distribuídos;
5. Gerenciamento de Transações;
6. Processamento de Consultas;
7. Controle de Concorrência e Recuperação;
8. Cluster; e
9. Alto Desempenho em Bancos de Dados Distribuídos.

O tipo de cluster “Cluster de alto desempenho” foi o utilizado no escopo deste trabalho.

Na etapa de criação de dois ambientes de homologação (Centralizado e Distribuído), foram criados dois ambientes virtuais com base no ambiente real do sistema *Conecta*, sendo que um deles comportou o BD de forma centralizada e o segundo de forma distribuída. Os ambientes de homologação foram criados afim de serem utilizados para se hospedar o sistema *Conecta* e sua base de dados, estes ambientes seriam então utilizados para se realizar os experimentos de desempenho.

Foi escolhido o valor de 10% para se comparar o ganho do ambiente distribuído sobre o ambiente centralizado, onde o ambiente distribuído pode apresentar um ganho em desempenho superior ou igual a 10% em relação ao ambiente centralizado ou o ambiente distribuído pode apresentar um ganho em desempenho inferior a 10% em relação ao ambiente centralizado. O valor de porcentagem de ganho de 10% foi um valor definido, apenas, de forma empírica, sem testes ou estudos.

Na etapa de levantamento e estudo de qual ferramenta de medição de desempenho de banco de dados seria utilizada, foi escolhida e analisada uma ferramenta para realizar a medição do desempenho do banco de dados do software *Conecta*. A ferramenta possibilitou a captura dos dados referentes ao tempo de execução, quantidade de requisições atendidas e consumo de recursos de cada transação no banco de dados. Os parâmetros dos recursos medidos por transação são: porcentagem de consumo de processador (CPU), média de consumo de memória RAM, média de consumo de disco e média de consumo de rede. Também foram analisados outros dois fatores importantes nos ambientes que são: média de requisições SQL atendidas por segundo e tempo de resposta de cada solicitação realizada. Foi desconsiderado o fato dos experimentos serem realizados em

ambientes virtualizados em um computador hospedeiro, não necessitando de uma rede de computadores, sendo assim os tempos de resposta comportaram apenas o tempo de execução de cada solicitação ao *software,* uma vez que limitações de rede que existem no ambiente Real não estariam disponíveis no ambiente virtualizado.

No levantamento sobre o consumo de recursos de banco de dados do software *Conecta*, foi realizado, através das ferramentas de medição *MySQL* Enterprise Monitor e Medidor de desempenho do Windows, o levantamento de consumo de recursos de banco de dados, a fim de se obter um demonstrativo de requisitos de sistema necessários por usuário, ou seja, o quanto representa em recursos computacionais mais um usuário utilizando o sistema *Conecta*. Este demonstrativo comportou os valores em recursos computacionais do custo de um usuário no sistema *Conecta*, onde poderá ser utilizado para se escalar o quanto de recursos será necessário para se comportar um número maior de usuários ao sistema.

Na etapa de simulação dos acessos do software *Conecta* à sua base de dados, a simulação comportou grupos de usuários, realizando acessos simultaneamente aos servidores do sistema, com intuito de medir o tempo de resposta e o consumo de recursos de cada transação nos ambientes centralizado e distribuído, respetivamente. Esta medição buscou apresentar o desempenho do sistema distribuído em relação ao centralizado. Os grupos de usuários criados para cada teste comportaram um grupo com carga de um usuário apenas acessando o sistema e um grupo com carga de cem usuários simultâneos acessando o sistema. Estes grupos de usuários foram necessários para se obter o comportamento do sistema no gerenciamento dos recursos com um usuário e com mais usuários simultâneos, onde os valores de um e cem usuários foi escolhido apenas, de forma empírica, sem testes ou estudos.

Foram criados dois perfis de usuários, sendo o primeiro chamado de “perfil básico”, que compreende os usuários que comumente realizam somente visualizações no sistema. Os usuários deste perfil percorreram o seguinte caminho no sistema:

- Acessaram a página de *login*;
- Autenticaram-se;
- Acessaram a página de seu perfil;
- Visualizaram a página de perfil de um amigo;

- Retornaram a sua página de perfil; e
- Saíram do sistema.

O segundo perfil de usuários compreende os usuários que comumente publicam conteúdo além de visualizações no sistema. Os usuários deste perfil percorreram o seguinte caminho:

- Acessaram a página de *login* do sistema;
- Autenticaram-se;
- Acessaram a página de seu perfil;
- Visualizaram a página de perfil de um amigo;
- Acessaram a página de conteúdo de seu amigo;
- Comentaram uma postagem do amigo;
- Retornaram a sua página de perfil;
- Realizaram uma postagem em sua página de perfil; e
- Saíram do sistema.

Os dois perfis de usuários foram criados afim de se obter os resultados das métricas de desempenho para os dois tipos de usuários, sendo que a diferença entre estes perfis é a existência de inserções no sistema, o que mostrará como o sistema se comporta nos experimentos de consumo de recursos com grupos diferentes de usuários.

A metodologia de análise dos testes e escolha dos recursos e parâmetros foi baseada no método USE - *Utilization, Saturation e Errors* (GREGG 2012, p.05), no qual, para cada recurso escolhido, são analisados sua utilização, saturação e taxa de erros.

A Utilização compreendeu o tempo em que cada recurso estaria sendo utilizado ou, em alguns casos, como a memória RAM, a porcentagem utilizada em relação ao que o recurso disponibiliza.

A Saturação compreendeu o tempo da fila de espera formada quando a capacidade máxima de um recurso foi alcançada, ou o tamanho dessa fila, sendo que nenhum dos testes apresentou utilização máxima de recursos ou fila de esperas.

Os Erros compreenderam a quantidade de falhas no sistema, que podem ser causadas pela indisponibilidade de recursos, por exemplo, quando há uma

sobrecarga do sistema ao atender muitas solicitações de usuários, sendo que nenhum dos recursos apresentou erros.

Para obtenção dos valores dos resultados das métricas por usuário nos dois ambientes foram realizadas os cálculos de média dos valores. Com estas médias foi possível realizar com mais facilidade a próxima etapa que foi o comparativo dos resultados por ambiente.

Foi apresentado, então, o comparativo dos resultados de desempenho dos dois ambientes, levando em consideração o consumo por usuário de recursos, média de transações atendidas por segundo e tempos de resposta das transações em cada um dos ambientes. Este comparativo tem como objetivo mostrar a porcentagem de ganho de desempenho do ambiente distribuído em relação ao ambiente centralizado. No comparativo foi verificado o quanto, em porcentagem de ganho ou perda, cada recurso foi mais utilizado ou menos utilizado por ambiente.

Foi então escrito e revisado os resultados obtidos nos experimentos realizados, a fim de estes dados serem utilizados pela Fábrica de Software do CEULP/ULBRA para auxiliar na implantação e utilização de um ambiente distribuído e bem como para futuras medições de requisitos do Software *Conecta*.

# 4 RESULTADOS E DISCUSSÃO

O presente capítulo aborda os resultados alcançados com a realização do presente trabalho. É importante lembrar que, em resumo, o trabalho envolveu a obtenção de informações sobre desempenho em dois ambientes de homologação (centralizado e distribuído), bem como os experimentos realizados em cada um deles visando demonstrar, de forma quantitativa, as diferenças de desempenho entre os dois ambientes.

## 4.1. Ambientes virtuais de homologação

Através do estudo das tecnologias envolvidas, vistas no capítulo 2, foram criados dois ambientes independentes para o sistema *Conecta* e seu banco de dados. Um destes ambientes está no formato Distribuído (Cluster) e o segundo está em formato Centralizado (Não Cluster).

Os ambientes foram criados em plataforma virtual, por causa da indisponibilidade de equipamentos físicos (computadores), visto a necessidade de se dispor de um número de cinco computadores para a realização deste trabalho.

Visto que na virtualização ocorre uma distribuição dos recursos físicos locais entre as máquinas virtuais, o desempenho total do servidor de virtualização também é distribuído entre esses *hosts*, fazendo com que durante uma sobrecarga de utilização destes recursos o desempenho dos equipamentos virtuais que estão sendo executados seja afetado diretamente. Buscando minimizar essa perda, o equipamento utilizado para hospedar os ambientes virtuais comportava configurações de recursos superiores a quantidade total de recursos necessários ao conjunto de servidores que formavam estes ambientes. Tanto no ambiente não Cluster quando o Cluster, o computador utilizado oferecia um número de 8 processadores, sendo 4 lógicos e 4 físicos, 16 GB de memória RAM, 02 discos rígidos internos com velocidade de 7200 rotações por minuto e barramento Sata 2 e uma placa de rede de barramento Gigabit.

O ambiente centralizado implementado era composto por 1 servidor virtual alocado em um disco físico secundário sendo que este disco era utilizado somente para o servidor virtual. O Cluster implementado era composto por 04 servidores

virtuais, sendo que 02 deles estavam em um dos discos rígidos e os outros 02 servidores estavam no outro disco rígido, distribuindo, assim, o acesso a disco e reduzindo gargalos durante os experimentos.

### 4.1.1. Ambiente Centralizado (Não Cluster)

O ambiente centralizado (virtual) foi criado com base no ambiente real que comporta a aplicação *Conecta* na Fábrica de Software do CEULP/ULBRA. A composição de hardware deste ambiente de homologação foi definida buscando-se uma proximidade com o servidor da instituição, já a composição de software não pode ser replicada devido a plataforma do ambiente real ser fechada (MAC OS), sendo replicados apenas os serviços necessários para disponibilização da aplicação *Conecta*, como o serviço web e de banco de dados.

O ambiente não Cluster foi formado por um servidor virtual com as seguintes características físicas:

1. Processador de 04 núcleos de 2.3 GHz de frequência;
2. Memória RAM de capacidade máxima de 3 GB;
3. Disco Rígido de capacidade máxima de 30 GB; e
4. Paca de Rede com barramento 10/100.

Este ambiente virtual também foi formado por características lógicas:

1. Sistema Operacional Microsoft Windows Server 2008 Enterprise SP2;
2. Java 7 Update 25;
3. Apache 2.4.3; e
4. *MySQL* Server 5.6 (InnoDB);

A Figura 16 apresenta o formato do ambiente centralizado.

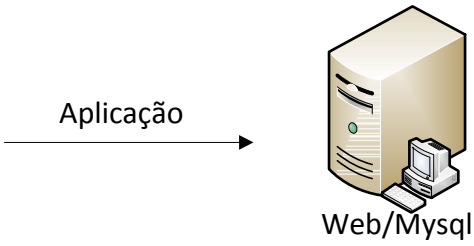


**Figura 16: Ambiente Centralizado**

Conforme apresentado pela Figura 16, o ambiente centralizado comportava em um único servidor todos os serviços necessários para se disponibilizar a aplicação, sendo estes o servidor web e o servidor de banco de dados MySQL.

O servidor Apache comportou a aplicação *Conecta* com todas as funcionalidades então implementadas até a data de 12 de maio de 2013, bem como sua base de dados no *MySQL* Server. Semelhantemente ao ambiente real da aplicação *Conecta* disponibilizado pela Fábrica de Software do CEULP/ULBRA, tanto a aplicação web quanto sua base de dados se encontravam no mesmo servidor.

O Servidor MySQL tem seu modo de gerenciar o controle de concorrência e recuperação de transações definido pelo módulo de armazenamento, podendo ser InnoDB e/ou NDB Cluster. O controle de concorrência no módulo InnoDB do *MySQL* ocorre por mecanismo de *lock* (BIANCHI 2008), sendo que o bloqueio ocorre por linha que está sendo acessada. Quando um processo vai escrever em uma tabela, o mecanismo de controle de concorrência bloqueia somente a linha da tabela que será modificada, deixando livre as demais linhas para serem utilizadas por outros processos, após a escrita a linha volta a ser desbloqueada. Tal mecanismo proporciona um nível balanceado entre desempenho, evitando que toda a tabela seja bloqueada e vários processos entrem em fila de espera, e segurança, evitando que outro processo acesse a mesma linha que está sendo modificada garantindo a consistência dos dados.

O controle de recuperação no InnoDB é o NO-UNDO/REDO, executando assim toda a transação um uma memória secundária (cache) e depois que a transação chega ao seu ponto de confirmação ela é escrita no disco (*MySQL* 2013e).

### 4.1.2. Ambiente Distribuído (Cluster)

O ambiente distribuído também foi criado com base no ambiente real, contudo foram utilizados recursos de ambientes distribuídos, sendo neste caso um Cluster de computadores.

O ambiente Cluster caracteriza-se por ser um ambiente Homogêneo e foi formado por quatro servidores virtuais com as características físicas apresentadas na Tabela 1.

**Tabela 1: Características Físicas dos nós do Cluster**

| Servidores | Processador | Memória RAM | Disco | Rede |
|---|---|---|---|---|
| Nó SQL | 04 núcleos de 2,3 GHZ | 03 GB | 30 GB | 2 * 10/100 |
| Nós de dados | 02 núcleos de 2,3 GHZ | 02 GB | 30 GB | 10/100 |
| Nó de Gerenciamento | 01 núcleo de 2,3 GHZ | 02 GB | 30 GB | 10/100 |

Conforme apresentado pela Tabela 1, os nós de dados compartilham as mesmas configurações físicas, já o nó de gerenciamento, que não necessita de grande desempenho, apresenta uma redução na quantidade de núcleos. O nó SQL apresenta a quantidade de recursos equivalente a existente no ambiente centralizado.

Este ambiente virtual foi configurado com as características lógicas apresentadas na Tabela 2.

**Tabela 2: Características Lógicas dos nós do Cluster**

| Servidores | S.O. | Java | MySQL Cluster | Servidor Web |
|---|---|---|---|---|
| Nó SQL | Windows Server 2008 Ent SP2 | 7 Update 25 | mysqld (7.3.2) | Apache 2.4.3 |
| Nós de dados | Windows Server 2008 Ent SP2 | 7 Update 25 | ndbmtd (7.3.2) | - |
| Nó de Gerenciamento | Windows Server 2008 Ent SP2 | 7 Update 25 | ndb_mgmd (7.3.2) | - |

Conforme observado na Tabela 2, cada nó do Cluster compartilha as mesmas características de *software*, se diferenciando apenas na existência do servidor web no nó SQL e nos módulos do MySQL Cluster que cada nó necessita para se criar o Cluster de banco de dados.

Cada Nó do Cluster tem seu módulo específico do *MySQL Cluster*, sendo que o nó SQL executa o módulo *mysqld.exe,* que é o servidor *MySQL*, este módulo é quem realiza a comunicação entre a aplicação e os nós de dados, tornando todo o processo transparente.

O nó de Dados executa o módulo *ndbmtd.exe* que é responsável por carregar e sincronizar os dados com os outros nós de dados do Cluster e também executar as instruções *sql multithread,* este módulo é quem realmente realiza as tarefas principais do banco, ou seja, é ele quem realiza as manipulações dos dados durante as consultas, sendo que quanto mais nós de dados existirem num cluster maior será a quantidade de recursos compartilhados e melhores a chances de se obter mais desempenho.

O nó de Gerenciamento executa o módulo *ndb_mgmd.exe* que é responsável por criar o Cluster e monitorar, dentre outras funções, a disponibilidade de cada nó.

O formato do Cluster compreende um nó SQL que responderá pelo banco de dados para a aplicação, dois nós de Dados onde estarão as bases de dados e um nó de gerenciamento do Cluster que cria e gerencia o Cluster e seus nós, conforme a Figura 17.

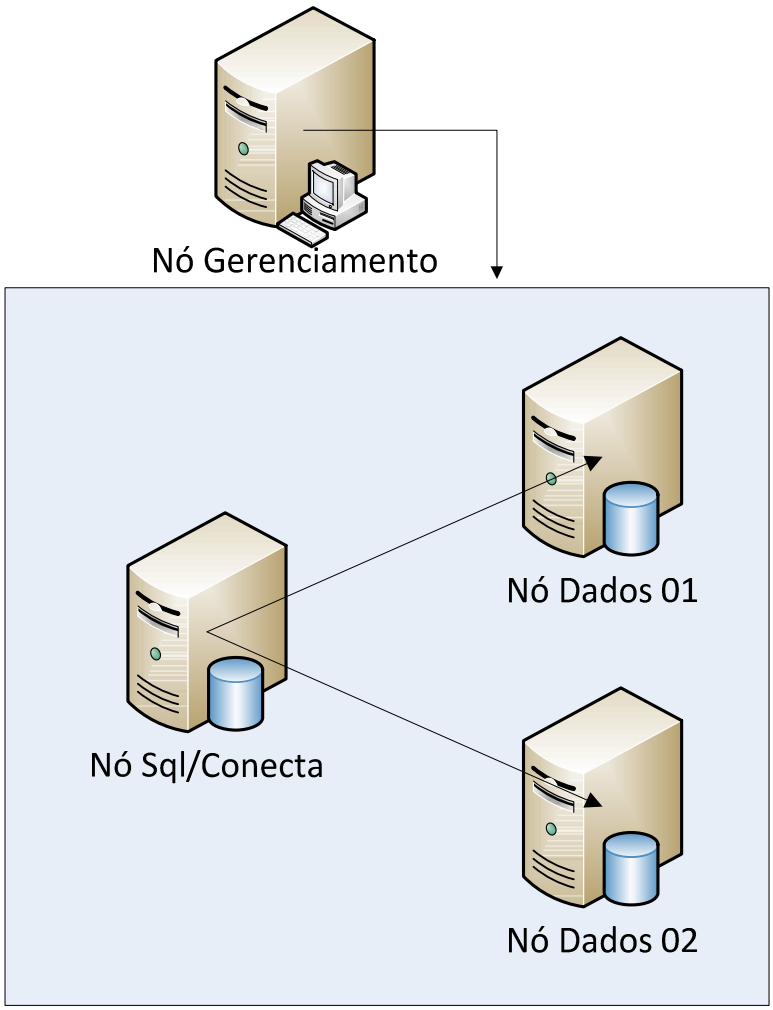


**Figura 17: Ambiente Cluster**

Conforme apresentado na Figura 17, o único nó SQL se comunica com os outros dois nós de dados, sendo que os três nós são gerenciados pelo nó de gerenciamento. Este grupo de nós trabalhando em conjunto forma o Cluster de banco de dados.

Este formato trabalha com as tabelas do banco de dados de maneira equivalente a um Cluster de Desempenho, o qual divide as tabelas em partes iguais na quantidade de nós de dados disponíveis ao nó SQL. Com esta divisão cada nó de dados recebe uma parte da tabela para ser armazenado em seu disco, conforme mostrado na Figura 18.

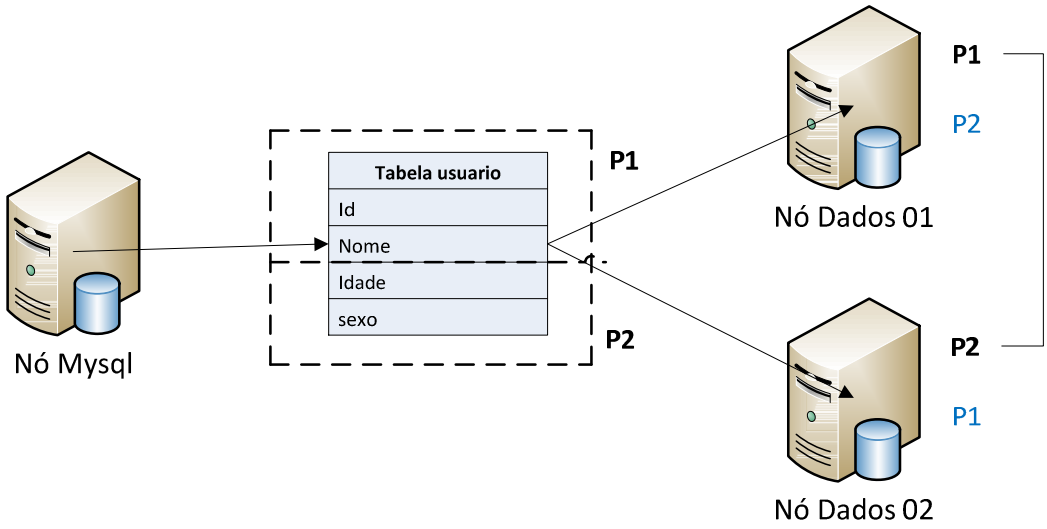


**Figura 18: Divisão de tabelas no Cluster**

Conforme apresentado na Figura 18, as tabelas do banco de dados são divididas na quantidade dos nós de dados (P1 = Parte 1 e P2 = Parte 2), sendo então cada parte dessa divisão alocada em um nó de dados. É também armazenada uma cópia da outra parte da tabela em cada nó de dados, permitindo, assim, que o nó de dados possa responder por toda a tabela em um momento de indisponibilidade do segundo nó de dados.

Esta configuração de replicação dos dados entre os nós está predefinida no Mysql Cluster, bastando informar, no momento de sua criação, quais nós serão de dados, qual nó será o nó SQL e qual será o nó de gerenciamento do Cluster. As tabelas criadas neste ambiente devem conter o argumento “ENGINE=NDBCLUSTER” ou “ENGINE=NDB” em seu script de criação, fazendo com que o Mysql Cluster fracione cada tabela criada na quantidade de partes proporcional à quantidade de nós de dados disponíveis no Cluster. Assim, se o Cluster dispuser de quatro nós de dados, cada tabela será fracionada em quatro partes, sendo que cada parte será armazenada em um dos nós de dados. A sincronização destes dados é realizada em pares, ou seja, os pares de nós de dados estão constantemente sincronizando sua base dados, mantendo, assim, a consistência destes dados. Cada nó de dados pertencente a um par de sincronia comporta uma fração da tabela e uma cópia da fração de seu outro par.

O controle de concorrência no módulo NDB Cluster do Mysql Cluster também ocorre por mecanismo de *lock*, onde o bloqueio por linha ocorre na linha que está sendo acessada.

O controle de recuperação no NDB Cluster também é o NO-UNDO/REDO, ou seja, executando toda a transação um uma memória secundária (cache) e depois que a transação chega ao seu ponto de confirmação ela é escrita no disco, após ser escrita no disco com sucesso ela é replicada para os nós de dados, sendo que havendo uma falha na transação não afetará os demais nós de dados (Mysql 2013f).

Como controle de alto desempenho em banco de dados distribuídos, o Mysql Cluster comporta funções Adaptive Query Localization (AQL), que minimizam o tráfego de rede entre os nós de dados durante consultas muito longas acessando na base local do primeiro nó de dados os dados do segundo nó de dados (Mysql 2013c).

## 4.2. Testes: Visão Geral e parâmetros

De acordo com Hannemann et al. (2010, p.09), o desempenho de um sistema pode ser definido de diversas maneiras, sendo que o sistema de melhor desempenho pode ser:

1. Aquele que executa em menor tempo um determinado programa;
2. Aquele que executa o maior número de programas num determinado tempo;
3. Aquele que executa um programa consumindo menos recursos físicos;
4. Aquele que executa um programa com menor tempo de resposta para o cliente.

Neste trabalho foram abordados os tipos 3 e 4, sendo respectivamente o sistema que economiza recursos físicos e o sistema mais rápido para o usuário.

Para se alcançar estes resultados foi necessário um plano de execução conhecido como metodologia.

A metodologia seguida para a realização dos experimentos foi a metodologia USE, como já visto no capítulo anterior. Sendo que, como primeira etapa do processo, foram escolhidos os seguintes recursos a serem analisados nos testes:

1. CPU - Porcentagem total de uso de processador utilizada pelo banco de dados;
2. Memória RAM – Quantidade em Megabytes de utilização da memória RAM pela aplicação e pelo banco de dados;
3. Disco – Quantidade em Megabytes de utilização do disco rígido pelo banco de dados;
4. Rede – Quantidade em Megabytes do tráfego de rede pelo banco de dados;
5. Requisições SQL – Quantidade de requisições SQL por segundo que o banco de dados conseguiu atender; e
6. Tempo de Resposta – Tempos de resposta em milissegundos de cada solicitação do sistema.

Após a escolha dos requisitos a serem testados, foram definidos dois perfis de usuários para acesso ao sistema *Conecta*, sendo que o primeiro realizava apenas visualizações (chamado de perfil básico) e o segundo realizava, além de visualizações, postagens de conteúdos textuais (chamado de perfil médio). Cada um destes perfis representava um grupo de usuário comum em uma rede social, sendo que suas particularidades, como o conjunto de visualizações e postagens, podem representar diferenças no consumo de cada teste.

Visto que os testes a serem realizados eram testes de estresse, foram definidas então as cargas a serem utilizadas nos testes: carga de 1 usuário realizando acesso ao sistema e carga de 100 usuários realizando acessos o sistema. Cada carga de teste mesclada com os grupos de usuários culminou em 4 cenários de testes, que foram respectivamente:

1. 1 usuário do perfil básico realizando acesso ao sistema;
2. 100 usuários do perfil básico realizando acesso ao sistema;
3. 1 usuário do perfil médio realizando acesso ao sistema; e
4. 100 usuários do perfil médio realizando acesso ao sistema.

Os testes foram realizados em cada um dos quatro cenários citados com uma frequência de 10 vezes, sendo que em cada um dos testes foi realizado a medição dos consumos de recursos, tempos de respostas e quantidade de requisições atendidas por segundo. Os experimentos nos quatro cenários foram realizados nos

dois ambientes de homologação, gerando, assim, um montante de 80 testes, que podem ser melhor visualizados na Tabela 3.

**Tabela 3: Divisão dos testes de estresse**

| Ambiente | Perfil | Carga | Cenários | Qtd Testes |
|---|---|---|---|---|
| Centralizado | Básico | 1 acesso | Cenário 1 | 10 |
| Centralizado | Básico | 100 acessos | Cenário 2 | 10 |
| Centralizado | Médio | 1 acesso | Cenário 3 | 10 |
| Centralizado | Médio | 100 acessos | Cenário 4 | 10 |
| Distribuído | Básico | 1 acesso | Cenário 1 | 10 |
| Distribuído | Básico | 100 acessos | Cenário 2 | 10 |
| Distribuído | Médio | 1 acesso | Cenário 3 | 10 |
| Distribuído | Médio | 100 acessos | Cenário 4 | 10 |
| | | | Total de Testes > | **80** |

Conforme apresentado na Tabela 3, os 80 testes estão distribuídos entre os quatro cenários de testes, sendo que cada um dos cenários comportou 10 testes. Em cada teste obteve-se valores dos 06 requisitos de sistema, que foram selecionados para comportar os valores de métricas de desempenho. Foi utilizado a média aritmética dos valores de cada requisito, ou seja, havendo 10 resultados de testes de um requisito é calculado então uma média aritmética destes valores reduzindo a apenas um valor de resultado de teste do mesmo requisito.

Após a escolha dos requisitos a serem analisados, foram realizadas as configurações nas ferramentas de testes e medições e então iniciado nos dois ambientes de homologação os testes de estresse em cada um dos quatro cenários.

Com a ajuda das ferramentas de testes e medições selecionadas, foram coletados os valores de consumo de recursos pelo banco de dados em cada um dos quatro cenários dos ambientes. Os valores coletados de cada teste em cada ambiente foram analisados seguindo-se as etapas da metodologia USE e serão descritos individualmente nos tópicos 4.2.1 e 4.2.2, sendo então comparados no tópico 4.3.

### 4.2.1. Consumo de recursos do BD e Resultados dos experimentos no ambiente Centralizado

O ambiente não cluster, ou Centralizado, comportava um único servidor que disponibilizava a aplicação web *Conecta* e sua base de dados, sendo que seus recursos eram compartilhados entre a aplicação o sua base de dados além do próprio sistema operacional.

Após a realização dos testes de estresse no ambiente centralizado com os quatro cenários de teste, foram coletados resultados de consumo de recursos e tempos de respostas, sendo que em cada cenário não foram observados erros ou saturação. Um erro neste ambiente pode ser uma solicitação não atendida. Todas as solicitações realizadas durante os experimentos foram atendidas pelo sistema *Conecta* e sua base de dados, sendo que as cargas utilizadas nos testes de estresse não foram voltadas ao máximo desempenho do sistema e sim para medir a média de desempenho e consumo de recursos pelo banco de dados em relação a quantidade de usuários no sistema.

Após a realização dos experimentos, foram coletadas as métricas geradas pelo ambiente nos quatro cenários propostos, as quais foram geradas através de médias dos valores de consumo de cada recurso e tempos de resposta das requisições, sendo apresentados pela Tabela 4.

**Tabela 4: Médias de consumo de recursos e tempos de resposta por usuário do Ambiente Centralizado**

| | CPU (%) | Memória (MB) | Disco (MB/s) | Rede (KB/s) | Requisições atendidas/s | Tempo de resposta (ms) |
|---|---|---|---|---|---|---|
| **Cenário 1** | 5,44 | 5,65 | 0,38 | 0,64 | 0,15 | 135,10 |
| **Cenário 2** | 0,78 | 0,12 | 0,01 | 0,19 | 0,08 | 65,34 |
| **Cenário 3** | 20,38 | 1,79 | 0,43 | 0,88 | 0,20 | 125,57 |
| **Cenário 4** | 0,62 | 0,34 | 0,01 | 0,20 | 0,13 | 41,82 |

Conforme apresentado na Tabela 4, as métricas foram dispostas de acordo com o cenário a que pertence, sendo que seus valores foram compostos pelas médias (de consumo de recursos e tempos de resposta) **por usuário** dos resultados dos experimentos. A Tabela 4 será explicada de forma detalhada nos apêndices deste trabalho.

### 4.2.2. Consumo de recursos do BD e Resultados dos experimentos no ambiente Distribuído

O ambiente Cluster ou Distribuído comportava um conjunto de servidores que disponibilizavam a aplicação *web Conecta* e sua base de dados, sendo que os recursos de cada servidor eram utilizados de forma distribuída como um único sistema.

Após a realização dos experimentos no ambiente distribuído nos quatro cenários, foram coletados dados de consumo de recursos e tempos de respostas, sendo que em cada cenário não foram observados erros ou saturação, pois todas as solicitações realizadas durante os experimentos foram atendidas pelo sistema *Conecta* e sua base de dados.

Após a realização dos experimentos foram coletadas as métricas geradas pelo ambiente nos quatro cenários propostos. Estas métricas foram geradas através de médias dos valores de consumo de cada recurso e tempos de resposta das requisições, sendo adicionados a Tabela 5.

**Tabela 5: Médias de consumo de recursos e tempo de respostas por usuário do Ambiente Distribuído**

| | CPU (%) | Memória (MB) | Disco (MB/s) | Rede (KB/s) | Requisições atendidas/s | Tempo de resposta (ms) |
|---|---|---|---|---|---|---|
| **Cenário 1** | 1,92 | 6,45 | 4,94 | 71,27 | 3,39 | 181,04 |
| **Cenário 2** | 0,24 | 0,13 | 0,04 | 1,34 | 0,19 | 55,09 |
| **Cenário 3** | 3,10 | 19,24 | 8,20 | 68,43 | 2,54 | 323,80 |
| **Cenário 4** | 0,02 | 0,01 | 0,04 | 0,72 | 0,04 | 10,01 |

Conforme apresentado na Tabela 5, as métricas foram dispostas de acordo com o cenário a que pertencem, sendo que seus valores foram compostos pelas médias de consumo de recursos por usuário dos resultados dos experimentos. A Tabela 5 será explicada de forma detalhada nos apêndices deste trabalho.

## 4.3. Comparativo dos resultados de desempenho

Após a realização dos experimentos nos quatro cenários dos ambientes Centralizado e Distribuído, os valores das métricas dos dois ambientes foram comparadas, buscando mostrar quantitativamente qual ambiente apresentou melhor desempenho.

O comparativo das métricas dos resultados foi composto pelos valores dos requisitos nos cenários 2 e 4 de cada ambiente, devido serem estes os cenários que comportaram usuários simultaneamente e, também, terem apresentado melhor gerenciamento dos recursos. Os valores foram distribuídos e apresentados na Tabela 6.

**Tabela 6: Médias comparativas de consumo de recursos e tempo de respostas por usuário dos Ambientes Centralizado e Distribuído**

| Ambientes | Cenários | CPU (%) | Memória (MB) | Disco (MB/s) | Rede (KB/s) | Requisições atendidas/s | Tempo de resposta (ms) |
|---|---|---|---|---|---|---|---|
| Centralizado | Cenário 2 | 0,78 | 0,12 | 0,01 | 0,19 | 0,08 | 65,34 |
| Centralizado | Cenário 4 | 0,62 | 0,34 | 0,01 | 0,20 | 0,13 | 41,82 |
| Distribuído | Cenário 2 | 0,24 | 0,13 | 0,04 | 1,34 | 0,19 | 55,09 |
| Distribuído | Cenário 4 | 0,02 | 0,01 | 0,04 | 0,72 | 0,04 | 10,01 |

Conforme apresentado na Tabela 6, as métricas foram dispostas de acordo com o cenário a que pertence, sendo que, semelhantemente às tabelas anteriores, seus valores foram compostos pelas médias por usuário dos resultados dos experimentos. A Tabela 6 será explicada de forma detalhada nos próximos subtópicos.

### Porcentagem de consumo de CPU

Pode ser observado que os valores deste recurso apresentaram uma redução em sua utilização entre os cenários da tabela, sendo que os maiores valores de utilização encontra-se nos primeiros cenários do ambiente centralizado e seguem decrescendo à medida que se aproxima dos últimos cenários do ambiente

distribuído. Sendo assim o cenário 2 do ambiente distribuído apresentou uma redução de 69,23% no consumo de CPU em relação ao cenário 2 do ambiente centralizado. Estes dados mostram que o ambiente Distribuído obteve maior otimização na utilização do recurso CPU entre os cenários que compartilham o mesmo número de usuários simultâneos e a presença de inserções na aplicação.

Já o cenário 4 do ambiente distribuído apresentou uma redução de 96,77% no consumo de CPU em relação ao cenário 4 do ambiente centralizado. Este ganho de desempenho no referido recurso foi 28,45% maior em relação ao ganho de desempenho do cenário 2 do ambiente distribuído sobre o cenário 2 do ambiente centralizado. O ganho de desempenho do cenário 4 no ambiente distribuído em relação ao cenário 4 do ambiente centralizado indica que a utilização deste recurso pelo ambiente distribuído nos cenários com acessos simultâneos e inserção no sistema foi melhor gerenciado pelas ferramentas de otimização deste ambiente. Este ganho em desempenho também é resultado da fragmentação e distribuição das cargas de consultas entre os nós de dados do ambiente distribuído.

### Consumo de memória RAM

Visando facilitar a visualização dos valores referentes ao comparativo do consumo de memória RAM nos dois ambientes da Tabela 6, estes valores foram extraídos para o Gráfico 1.

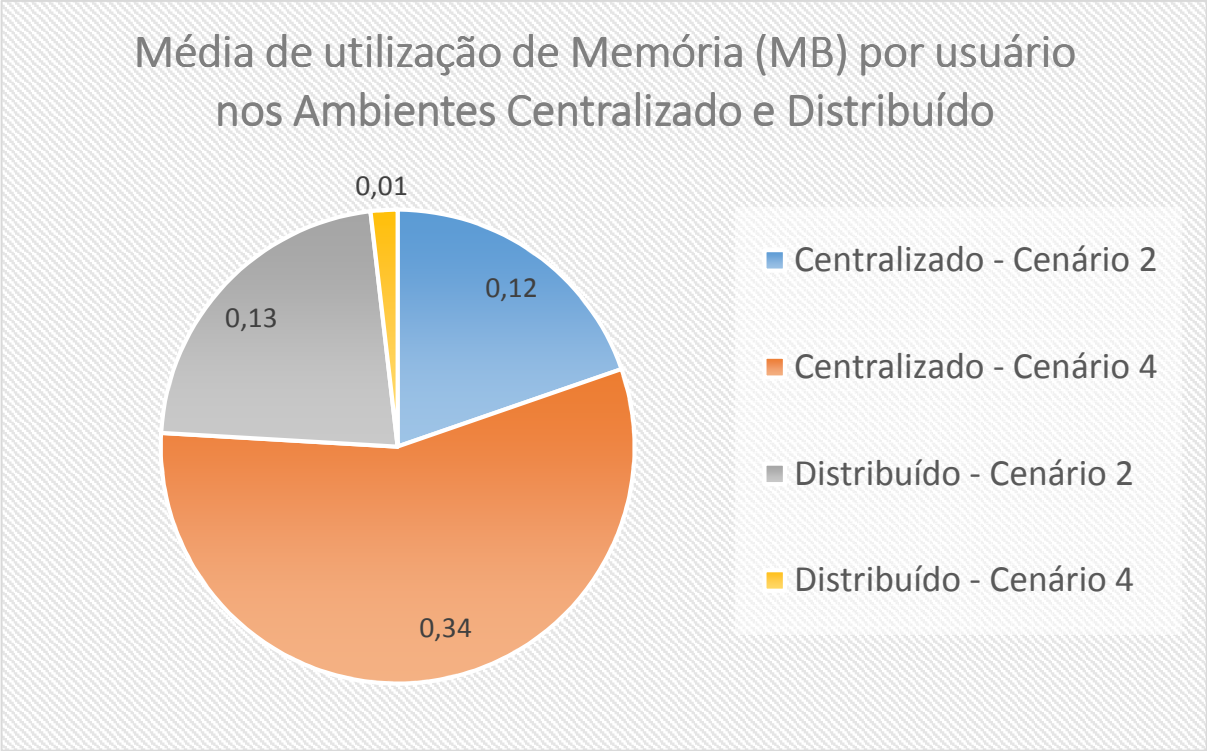


**Gráfico 1: Média de utilização de Memória (MB) por usuário nos Ambientes Centralizado e Distribuído**

Conforme exposto pelo Gráfico 1, o cenário 2 do ambiente distribuído apresentou um aumento de 7,69% no consumo do recurso de memória RAM em relação ao mesmo cenário do ambiente centralizado. Estes valores demonstram que o recurso *Memcached* não obteve uma melhora significativa no gerenciamento do consumo deste recurso por usuário no sistema, sendo que os cenários só comportam visualizações de conteúdo.

Observa-se que cenário 4 do ambiente distribuído obteve uma redução de 97,05% no consumo de memória RAM em relação ao cenário 4 do ambiente centralizado, indicando assim que os recursos de otimização como o *Memcached*, apresentaram melhor desempenho no gerenciamento deste recurso nos ambientes que comportavam inserções no sistema.

### Consumo de Disco

Visando facilitar a visualização dos valores referente ao comparativo do consumo de disco nos dois ambientes da Tabela 6, estes valores foram extraídos para o Gráfico 2.

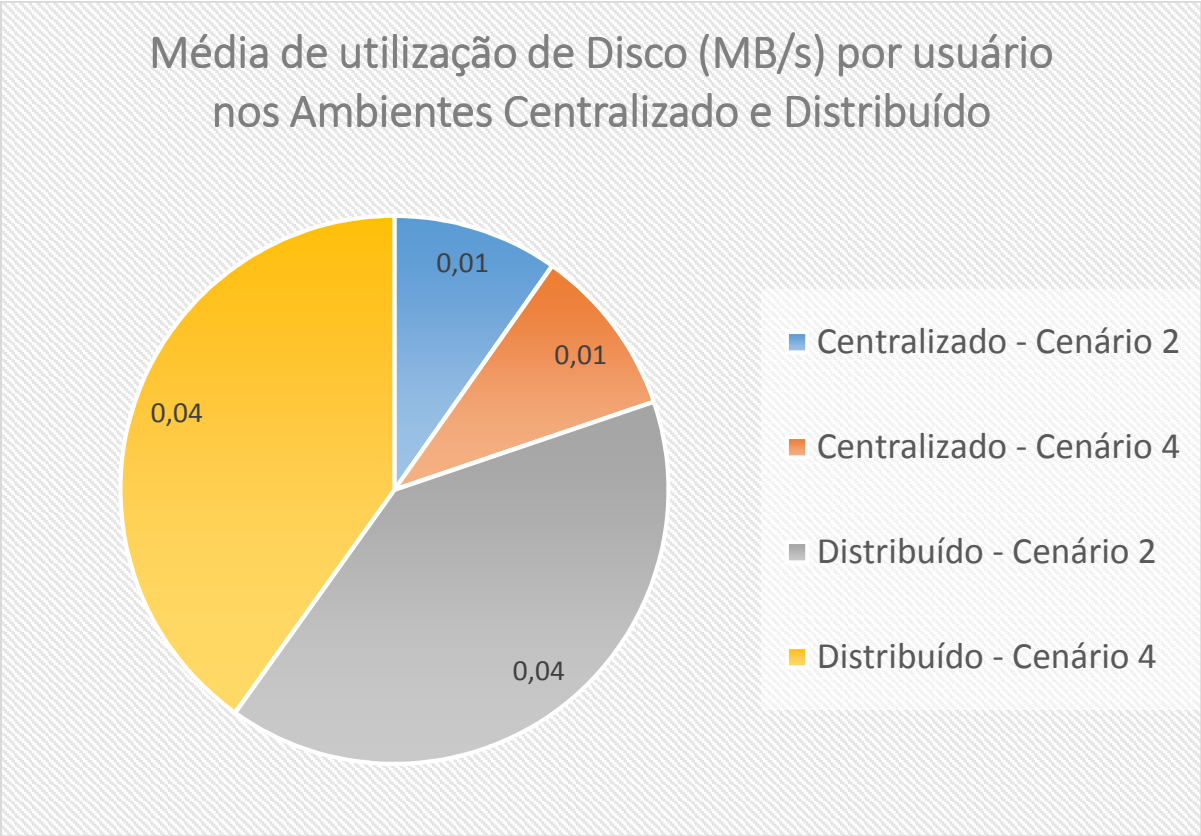


**Gráfico 2: Média de utilização de Disco (MB/s) por usuário nos Ambientes Centralizado e Distribuído**

Conforme apresentado pelo Gráfico 2, os cenários 2 e 4 do ambiente distribuído apresentaram um aumento de 75% no consumo de disco por usuário em relação aos cenários 2 e 4 do ambiente centralizado. Este aumento é pincipalmente devido ao processo de sincronização dos dados do banco entre os nós de dados do Cluster, onde cada alteração na base de dados realizada pelas instruções SQL *Insert e Update* que a aplicação *Conecta* envia ao banco é realizada na memória e depois no disco, gerando assim um processo de sincronização entre os nós afim de se manter a integridade dos dados fracionados. Ao final do processo de sincronia todos os nós de dados que continham uma fração dos dados que foram modificados ou inseridos apresentaram utilização no disco.

**Consumo de Rede**

Visando facilitar a visualização dos valores referente ao comparativo do consumo de rede nos dois ambientes da Tabela 6, estes valores foram extraídos para o Gráfico 3.

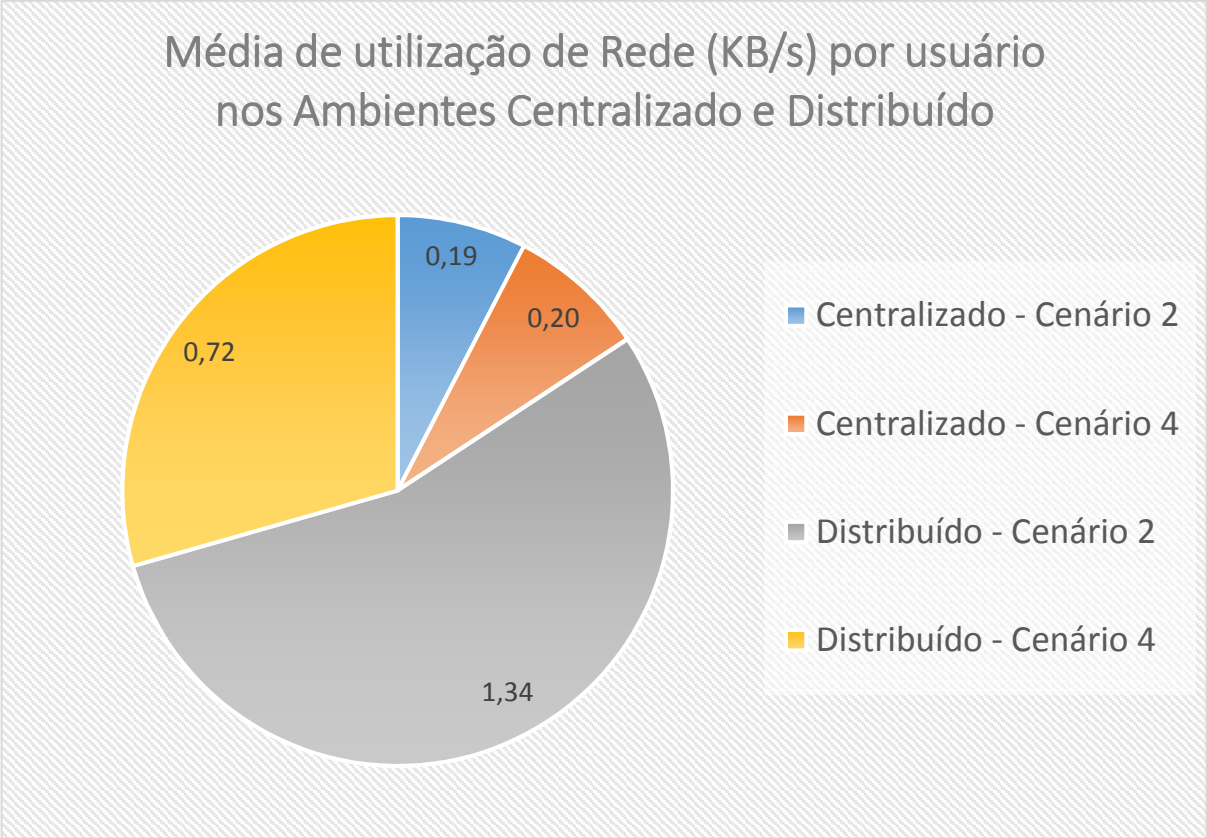


**Gráfico 3: Média de utilização de Rede (KB/s) por usuário nos Ambientes Centralizado e Distribuído**

Conforme apresentado pelo Gráfico 3, o cenário 2 do ambiente distribuído apresentou 85,82% de aumento no consumo de rede em relação ao cenário 2 do ambiente centralizado. Já o cenário 4 do ambiente distribuído apresentou 72,22% de aumento deste mesmo recurso em relação ao cenário 4 do ambiente centralizado. Estes valores mostram que a sincronização dos dados dos nós de dados através da rede provocou um aumento significativo no total de consumo do recurso referido.

Entretanto esse aumento no consumo de rede se acentuou no cenário que comportou inserções no sistema, o que provocou maior volume de tráfego de dados na rede.

### Requisições atendidas

Visando facilitar a visualização dos valores referentes ao comparativo da quantidade de requisições atendidas por segundo nos dois ambientes da Tabela 6, estes valores foram extraídos para o Gráfico 4.

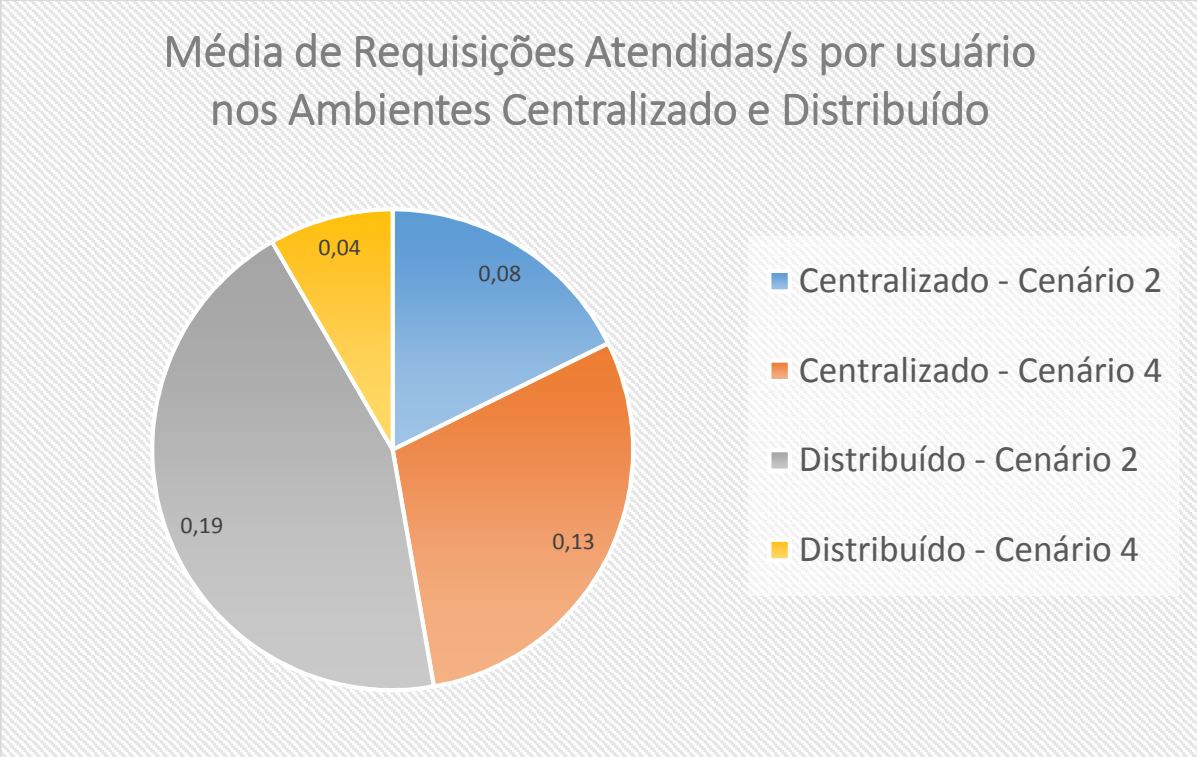


**Gráfico 4: Média de Requisições Atendidas/s por usuário nos Ambientes Centralizado e Distribuído**

Conforme apresentado pelo Gráfico 4, o cenário 2 do ambiente distribuído obteve um aumento no número de requisições SQL atendidas por segundo de 57,89% em relação ao cenário 2 do ambiente centralizado, isso demonstrou que

nestes cenários o ambiente distribuído atendeu um número maior de requisições por segundo em relação ao ambiente centralizado.

Observando os dados dos cenários 4 pode-se verificar que, o cenário 4 do ambiente distribuído apresentou 69,23% de redução no número de requisições atendidas por segundo em relação ao cenário 4 do ambiente centralizado. Este comportamento indica que em cenários que comportaram inserções no sistema o ambiente distribuído apresentou perda de desempenho neste requisito.

### Tempos de resposta

Visando facilitar a visualização dos valores referentes ao comparativo de tempos de resposta nos dois ambientes da Tabela 6, estes valores foram extraídos para o Gráfico 5.

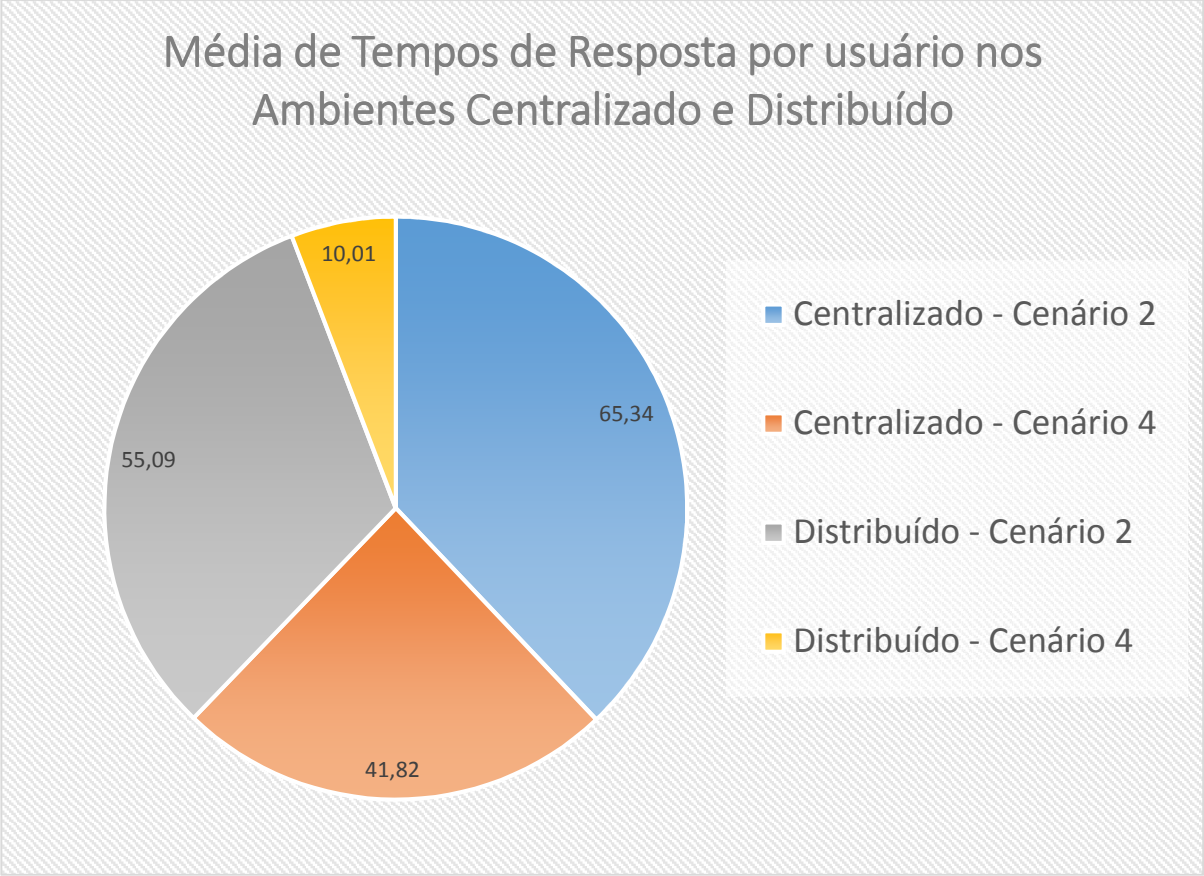


**Gráfico 5: Média de Tempos de Resposta por usuário nos Ambientes Centralizado e Distribuído**

Conforme apresentado no Gráfico 5, observa-se que no cenário 2 do ambiente distribuído obteve-se uma redução nos tempos de respostas de 15,68% em relação ao mesmo cenário do ambiente centralizado. Com tempos menores significa que os usuários esperaram menos tempo para visualizar o resultado de suas ações no sistema.

Observa-se também que no cenário 4 do ambiente distribuído a redução nos tempos de respostas obtida foi de 76,06% em relação ao cenário 4 do ambiente centralizado. Estas reduções nos tempos de resposta das solicitações é o resultado do gerenciamento otimizado dos recursos pela aplicação e o *cluster* com a ajuda das ferramentas de otimização.

Após a análise do comparativo de dados dos dois ambientes, foi possível verificar que, de modo geral, o ambiente centralizado obteve uma desempenho superior nos recursos de consumo de disco, rede e quantidade de requisições atendidas por segundo, já o ambiente distribuído superou o ambiente centralizado nos recursos de CPU, memória RAM e tempos de resposta.

# 5 CONSIDERAÇÕES FINAIS

Foi constatado que a utilização da tecnologia de *Cluster* para banco de dados se mostrou vantajosa, por apresentar algumas características de grande valor ao ambiente, como a escalabilidade, disponibilidade e alto desempenho. O modelo de *Cluster* escolhido para ser utilizado neste trabalho foi um modelo não voltado para disponibilidade e sim para alto desempenho, mas este modelo ainda trouxe nativamente recursos e comportamentos que garantem alta disponibilidade ao ambiente.

No sistema *Conecta* foi observado que a utilização desta tecnologia pode trazer melhoras no desempenho da aplicação no que se refere ao banco de dados, trazendo nos experimentos ganhos em relação ao ambiente centralizado de: 81% de CPU, 68% de memória RAM e 39% em tempos de resposta. Observou-se que alguns recursos passaram a ser mais consumidos no ambiente distribuído, como o consumo de disco (-305%) e de rede (-436%), sendo que este consumo se justifica para se obter maior desempenho quanto a tempo de resposta e principalmente disponibilidade do ambiente.

Foi observado, também, que nos dois ambientes o gerenciamento dos recursos analisados se mostrou mais otimizado nos cenários que tiveram vários usuários realizando acessos (acessos simultâneos) e em alguns casos juntamente com a presença de ações de inserções na aplicação por estes usuários.

Com a utilização de um ambiente Cluster para o banco de dados com maior número de nós de dados interligados por uma rede de baixa latência, pode-se obter melhores resultados de desempenho e disponibilidade. Um cenário de maior porte não foi possível de ser utilizado devido à indisponibilidade de equipamentos para o trabalho, sendo necessário então a utilização de ambientes virtuais para se alcançar os resultados esperados, devido este ambiente ter sido virtualizado uma expansão do modelo proposto traria uma perda superior ao ganho do desempenho, tornando a solução inviável.

Como trabalhos futuros pode-se propor o estudo e criação de ambientes *híbridos*, sendo formado por ambientes distribuídos em conjunto com ambientes centralizados, sendo que este ambiente *hibrido* poderia trazer o melhor dos dois mundos, o centralizado e o distribuído. Uma outra possibilidade de trabalhos futuros seria o estudo e criação de um ambiente Cluster buscando analisar o ganho % de desempenho do ambiente à medida que são adicionados novos nós de dados e nós SQL.

# 6 REFERÊNCIAS BIBLIOGRÁFICAS

AMAZON. **Amazon Architecture**, 2007. Disponível em: <http://highscalability.com/amazon-architecture>, Acessado em 14 de Junho de 2013 as 15h54m.

Apache. **The Apache Software Foundation**, 2013. Disponível em: <http://jmeter.apache.org>, Acessado em 25 out 2013 as 22h06m.

BIANCHI, Wagner. **MySQL INNODB – Introdução e principais características**, 2008. Disponível em: <http://imasters.com.br/artigo/8065>, Acessado em 01 nov. 2013 as 23h57m.

BUYYA, R.; **High Performance Cluster Computing: Architectures and Systems** vol. 01 Editora Pearson Prentice Hall, 1999.

CASANOVA, M. A.; MOURA, A. V. **Princípios de Sistemas de Gerência de Banco de Dados Distribuídos** 1ª Ed. Editora Campus, 1999.

COULOURIS, G.; DOLLIMORE, J.; KINDBERG, T. **Sistemas Distribuídos: Conceitos e Projeto** 4ª Ed. Editora Bookman, 2007.

ELMASRI, R.; NAVATHE, S. B. **Sistemas de Banco de Dados** 6ª ed. Editora Pearson Education do Brasil, 2011.

GREGG, Brendan. **Thinking Methodically about Performance**, 2012. Disponível em: <http://delivery.acm.org/10.1145/2420000/2413037/p40-gregg.pdf>, Acessado em 31 out. 2013 as 21h16m.

HANNEMANN, D.; Silva, E. R.; Silva, K. da C. **Avaliação de Desempenho Utilizando Técnicas de Virtualização Completa em Cluster Web**, 2010. Disponível em:

<http://www.bcc.unama.br/index.php?option=com_docman&task=doc_download&gid=68&Itemid=76>, Acessado em 29 out. 2013 as 19h50m.

HEUSER, C. A. **Projeto de Banco de Dados** 4ª ed. Editora Digital Source, 1998.

LAGARES, V.; Eliza, R. **Testes de Desempenho, Carga e Stress**, 2013.Disponível em: <http://www.devmedia.com.br/testes-de-desempenho-carga-e-stress-revista-java-magazine-110/26546#>, Acessado em 23 nov. 2013 as 01h50m.

Microsoft. **Usando o Monitor de Desempenho**, 2013. Disponível em: <http://technet.microsoft.com/pt-br/library/cc749115.aspx>, Acessado em 27 out 2013 as 15h04m.

MySQL. **Defining MySQL Cluster Data Nodes**, 2013f. Disponível em: <http://dev.mysql.com/doc/refman/5.1/en/mysql-cluster-ndbd-definition.html>, Acessado em 02 nov. 2013 as 00h25m.

MySQL. **MySQL Cluster Evaluation Guide** 2013a. Disponível em: <http://www.mysql.com/why-mysql/white-papers/mysql-cluster-evaluation-guide>, Acesso em 25 set 2013 as 21h11m.

MySQL. **MySQL Enterprise Monitor**, 2013d. Disponível em: <http://dev.mysql.com/doc/refman/5.5/en/mysql-enterprise-monitor.html>, Acesso em 25 out 2013 as 22h21m.

MySQL. **Using the MySQL Cluster Auto-Installer** 2013b. Disponível em: <http://dev.mysql.com/doc/refman/5.6/en/mysql-cluster-install-auto-using.html>, Acesso em 15 set 2013 as 13h30m.

MySQL. **Optimizing Performance of the MySQL Cluster Database**, 2013c. Disponível em: <http://www.mysql.com/why-mysql/white-papers/mysql_wp_cluster_performance.php>, Acessado em 21 ago. 2013 as 21h29m.

MySQL. **The InnoDB Recovery Process**, 2013e. Disponível em: <http://dev.mysql.com/doc/refman/5.7/en/innodb-recovery.html>, Acessado em 02 nov. 2013 as 00h10m.

OZSU, M.T.; VALDURIEZ, P. **Princípios de Sistemas de Banco de Dados Distribuídos** 2ª ed. Editora Prentice Hall, 1999.

PITANGA, Marcos; **Computação em Cluster**. Disponível em: <http://www.clubedohardware.com.br/artigos/153>, Acesso em 13 mai 2013 as 22h10m.

SHIBAYAMA, E. T. **Estudo Comparativo entre Bancos de Dados Distribuídos**. 2004. 65 p. Monografia (Graduação em Ciências da Computação) – Universidade Estadual de Londrina, Londrina.

SILBERSCHATZ, A.; KORTH, H. F.; SUDARSHAN, S. **Sistema de Banco de Dados** 3ª ed. Editora Makron Books, 1999.

SOBRINHO, David Lojudice. **Avaliação de Modelos de Arquitetura de Web Sites de Alta Escalabilidade**. 2009. 54 p. Monografia (Pós-Graduação em Tecnologia em Análise e Projeto de Sistemas) – Faculdade de Tecnologia de São Paulo, São Paulo.

TANENBAUM, Andrew S.; STEEN, Maarten Van. **Distributed Systems: Principles and Paradigms** 2ª ed. Editora Pearson Prentice Hall, 2006.

TAKAI, O.K.; ITALIANO, I.C.; FERREIRA, J.E. **Introdução à Banco de Dados**, 2005. Disponível em <http://www.ime.usp.br/~jef/apostila.pdf>. Acessado em 25 mar 2013 as 20h15m.

WADA, E. Y.; ARAÚJO, E. S.; SARAIVA, M. S.; AGUIAR, T. L. **Banco de Dados Paralelos**, 2003. Disponível em <http://nobios.por.com.br/trabalhos/Bancos%20de%20Dados%20Paralelos.pdf>. Acessado em 30 mai 2013.

VMware. **VMware Workstation**, 2013. Disponível em: <http://www.vmware.com/br/products/workstation>, Acessado em 25 out 2013 as 20h34m.

# APÊNDICES

Aqui serão detalhados os resultados dos testes realizados nos dois ambientes de homologação.

## Ambiente Centralizado

### Porcentagem de consumo de CPU

O primeiro recurso analisado é o consumo de CPU pelo banco de dados. Os valores aqui apresentados representam o total de utilização em porcentagem do consumo do processador. Para facilitar a visualização dos dados referentes ao consumo de CPU da Tabela 4, estes valores foram extraídos para o Gráfico 6 abaixo.

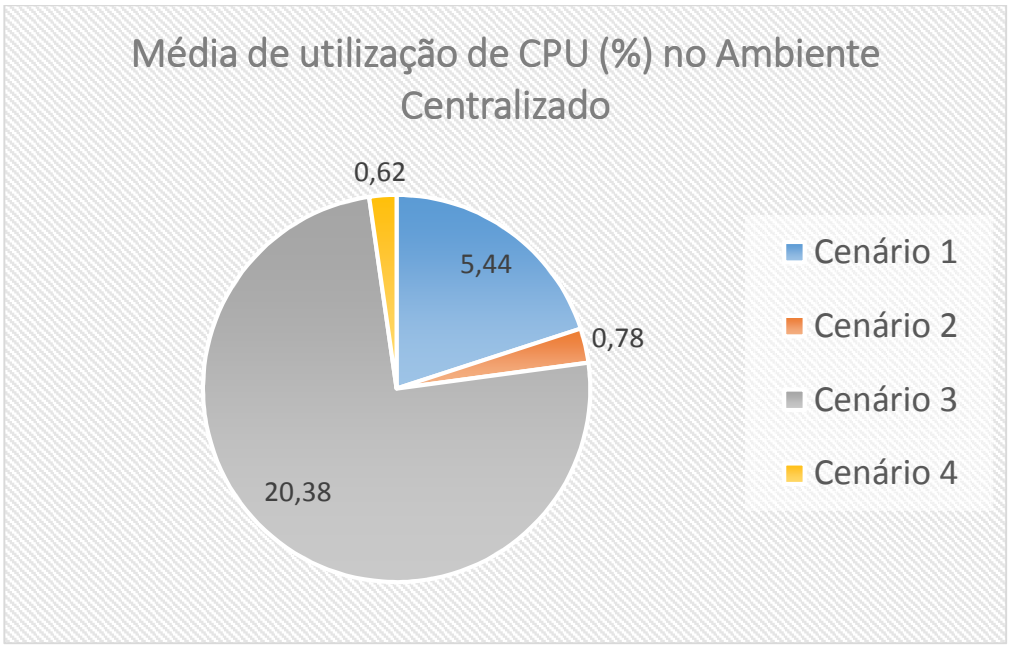


**Gráfico 6: Média de utilização de CPU (%) no Ambiente Centralizado**

Analisando os cenários apresentados pelo Gráfico 6 pode-se verificar que o consumo de CPU do cenário 2 em relação ao cenário 1 não apresentou um crescimento linear, ou seja, no cenário 2 o crescimento de utilização deste recurso não acompanhou os valores encontrados por usuário encontrados no cenário 1, sendo assim, no cenário 2 um usuário no sistema consumiu um valor 85,66% menor

de CPU em relação ao cenário 1. Esta diferença de consumo neste cenário mostra que o sistema MySQL realizou o gerenciamento do recurso CPU de forma otimizada.

No cenário 4, onde um usuário teve um consumo de CPU 96,97% menor em relação ao cenário 3, pode-se observar a mesma otimização no consumo de CPU encontrada no cenário 2. Outra análise a ser observada nestes cenários é a média de consumo de CPU do cenário 3 em relação ao cenário 1, onde o cenário 3 consumiu 274,51% mais processamento que o cenário 1, sendo que os cenários 3 e 4 tiveram como principal diferença a existência de postagens de conteúdos no sistema em relação aos cenários 1 e 2. Esta variação não se observa no cenário 4 em relação ao cenário 2, conforme apresentado no Gráfico 7.

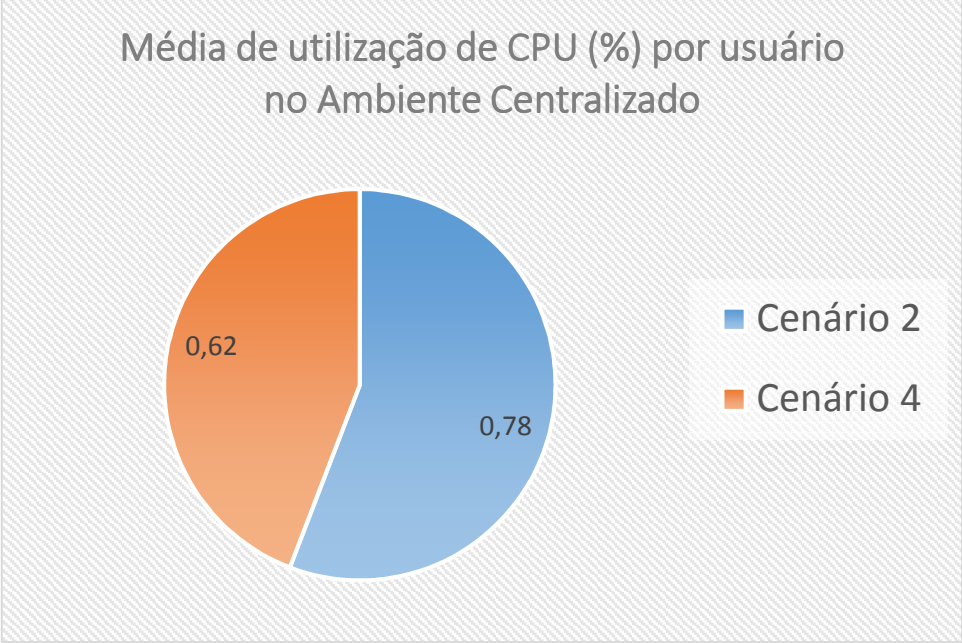


**Gráfico 7: Média de utilização de CPU (%) no Ambiente Centralizado**

Conforme exposto no Gráfico 7, o cenário 4 apresenta uma redução de 20,88% no consumo de CPU por usuário em relação ao cenário 2. Estes comportamentos mostram que o ambiente Centralizado apresentou uma utilização otimizada do recurso CPU nos cenários que comportavam usuários simultâneos que pertencem ao perfil de usuários que realizam postagens no sistema. Portanto o cenário 4 obteve melhor desempenho em relação ao cenário 2.

### Consumo de memória RAM

O próximo recurso analisado foi o consumo de memória RAM, onde seus valores foram coletados de todo o sistema (S.O. + Aplicação + Banco de Dados) e não somente do banco de dados, devido a ferramenta *MySQL Monitor* não disponibilizar estes dados apenas do banco de dados. Para facilitar a visualização dos dados referente ao consumo de memória RAM da Tabela 4, estes valores foram extraídos para o Gráfico 8 abaixo.

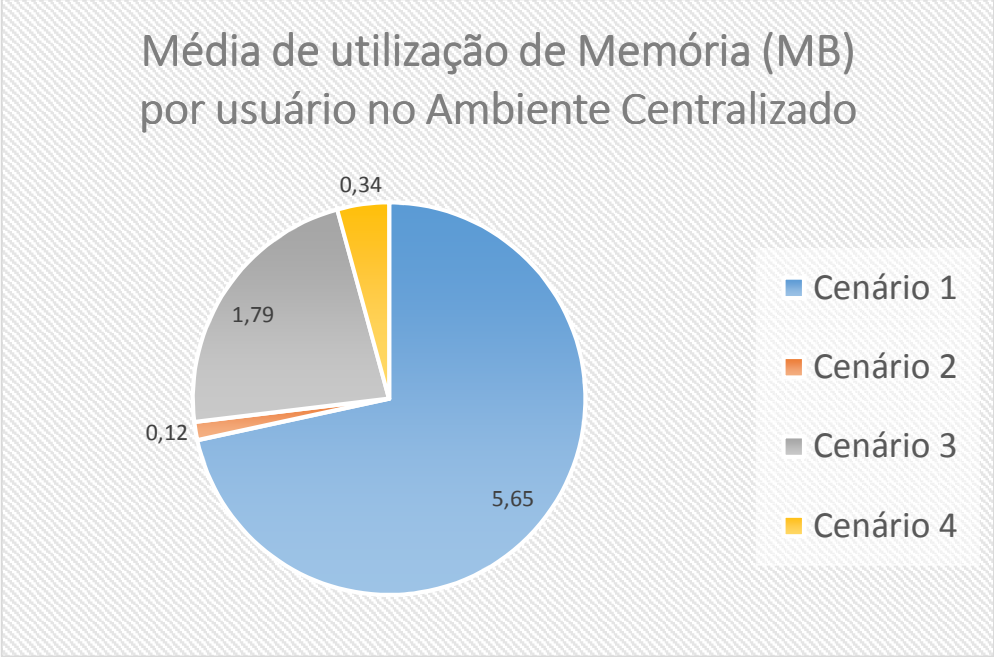


**Gráfico 8: Média de utilização de Memória (MB) no Ambiente Centralizado**

Conforme apresentado no Gráfico 8, pode ser verificado que no cenário 2 a média de utilização de memória RAM por usuário no sistema foi 98,05% menor em relação ao cenário 1. Esta variação na utilização deste recurso nos cenários que comportam usuários simultâneos pode ser explicada pelo gerenciamento de memória de alguns recursos da aplicação como o *Memcached*.

O cenário 4 também apresentou uma otimização no consumo de memória RAM por usuário, contudo esta otimização não se apresentou tão acentuada como no cenário 2, sendo este consumo por usuário apenas 18,43% inferior ao encontrado no cenário 3. Observa-se então que os recursos de otimização como o *memcached* apresentaram um menor índice de ganho de desempenho nos cenários

que comportaram inserções no sistema, em comparação ao ganho de desempenho encontrado no cenário 2 em relação ao cenário 1.

Observa-se também que o cenário 3 obteve uma redução de 31,68% no consumo de memória RAM em relação ao cenário 1. Esta redução na utilização do recurso mostra que seu gerenciamento foi realizado de forma otimizada nos cenários que comportavam apenas um usuário no sistema e que houvesse inserções.

Quanto ao cenário 4 pode ser observada uma redução de 97,19% no consumo de memória RAM por usuário em relação ao cenário 2. Esta redução no consumo de memória RAM mostra que o recurso passou a ser melhor gerenciado quando houve acessos simultâneos mesmo quando estes cenários apresentam utilização de disco através das inserções presentes nos cenários 3 e 4.

### Consumo de Disco

O próximo recurso analisado foi o consumo de disco. Os valores aqui apresentados representam a quantidade de dados que entram e saem do disco em *Mega Bytes* por segundo. Para facilitar a visualização dos dados referentes ao consumo de disco da Tabela 4, estes valores foram extraídos para o Gráfico 9 abaixo.

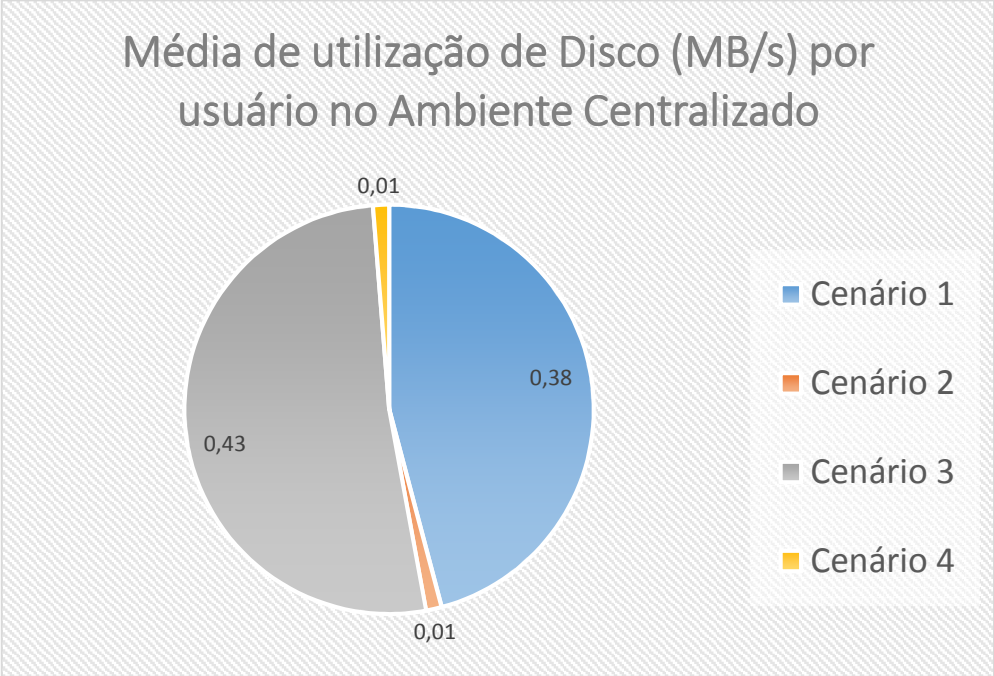


**Gráfico 9: Média de utilização de Disco (MB/s) no Ambiente Centralizado**

Conforme apresentado no Gráfico 9, no cenário 2 o consumo de disco por usuário é de 0,01 MB/s, ou 10 KB/s, de utilização de disco por usuário. Este valor é 97,36% menor que o valor consumido por usuário no cenário 1. Contudo, tanto o cenário 1 quanto o cenário 2 não são compostos por perfis de usuários que realizam requisições que consomem este tipo de recurso, sendo assim estes valores são em grande maioria de valores sendo lidos do banco de dados. Mais uma vez verifica-se que o MySQL realiza um gerenciamento otimizado do recurso à medida que mais usuários acessam o sistema.

Observa-se, também, um aumento de 98,05% de utilização de disco por usuário no cenário 3 em relação ao cenário 4. Estes cenários apresentaram uma variação menor do gerenciamento do recurso em relação ao encontrado no cenário 2, contudo os cenários 3 e 4 são formados por perfis de usuários que realizam ações que utilizam disco, sendo que esta utilização de disco é composta por duas inserções de comentários no sistema que geraram relativamente poucas entradas no banco de dados, como pode ser analisado principalmente pela diferença de 1,09% entre o cenário 4 em relação ao cenário 2.

No cenário 4, em relação ao cenário 2, tem-se um aumento de 1,01% no consumo de disco por usuário. O que mais uma vez mostra que a quantidade de dados inseridos no banco de dados em cada um dos experimentos foi gerenciada de forma otimizada.

Esta otimização de gerenciamento do consumo de disco não se mostrou tão acentuada no cenário 3, que apresentou um aumento de 11,67% no consumo de disco em relação ao cenário 1.

**Consumo de Rede**

O próximo recurso analisado foi o consumo de Rede pelo banco de dados. Estes dados compreendem o consumo de Rede em KB/s pelo banco de dados durante os experimentos realizados, sendo que, devido ao ambiente ser virtualizado, os dados dos experimentos não sofreram variação por ruídos ou outros redutores de desempenho presentes nos ambientes físicos. Para facilitar a visualização dos

dados referentes ao consumo de Rede da Tabela 4, estes valores foram extraídos para o Gráfico 10.

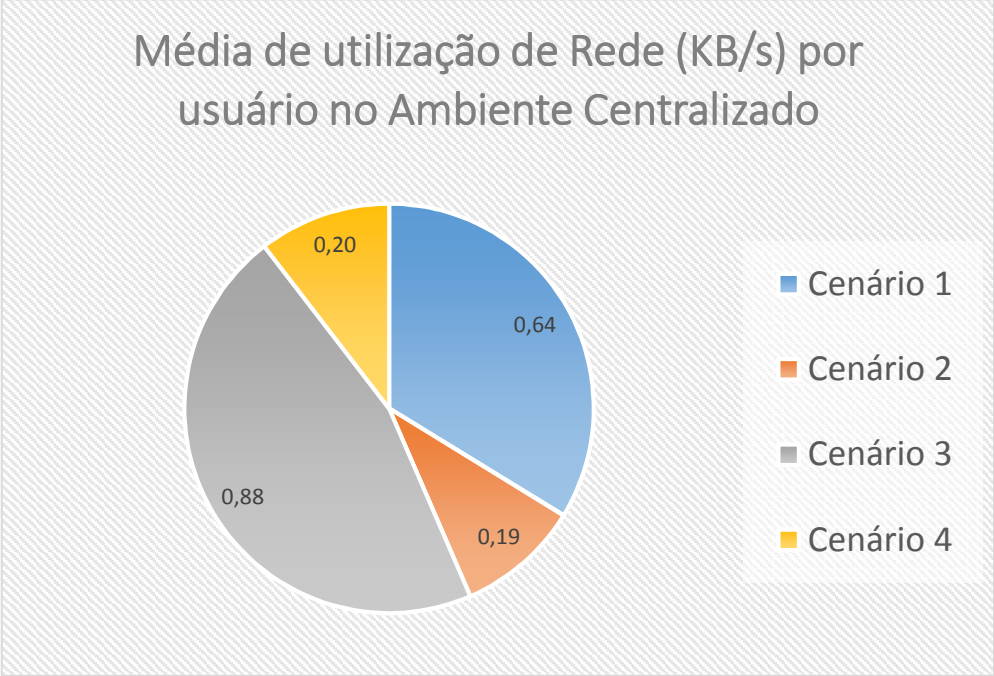


**Gráfico 10: Média de utilização de Rede (KB/s) no Ambiente Centralizado**

Pode ser observado no Gráfico 10 que a vazão de dados no cenário 2 foi 71,87% menor em relação ao cenário 1. Um aumento similar é encontrado entre os cenários 4 e 3, onde o cenário 4 apresentou uma redução de 78,40% em seu consumo de banda de Rede por usuário em relação ao cenário 3. Esta pequena variação no consumo deste recurso nos experimentos que comportavam usuários simultâneos mostra que seu gerenciamento não sofreu grandes alterações em relação a variação de perfil de usuário.

No cenário 3 é possível verificar um aumento de 27,27% na vazão de Rede em relação ao cenário 1, sendo que no cenário 4 esta diferença por usuário só chegou a 5,26% em relação ao cenário 2. Com estes valores é possível observar que o recurso de Rede foi utilizado de forma otimizada nos cenários que comportavam usuários simultâneos e não apresentavam perfis de usuários que realizavam postagens no sistema.

**Requisições atendidas**

Outra métrica analisada foi a quantidade de requisições SQL atendidas por segundo pelo banco de dados. Esta métrica representa a quantidade média de instruções *Select, Insert, Update, Replace, Delete* e *Call* realizadas no banco de dados pela aplicação. Para facilitar a visualização dos dados referente ao consumo de Rede da Tabela 4, estes valores foram extraídos para o Gráfico 11.

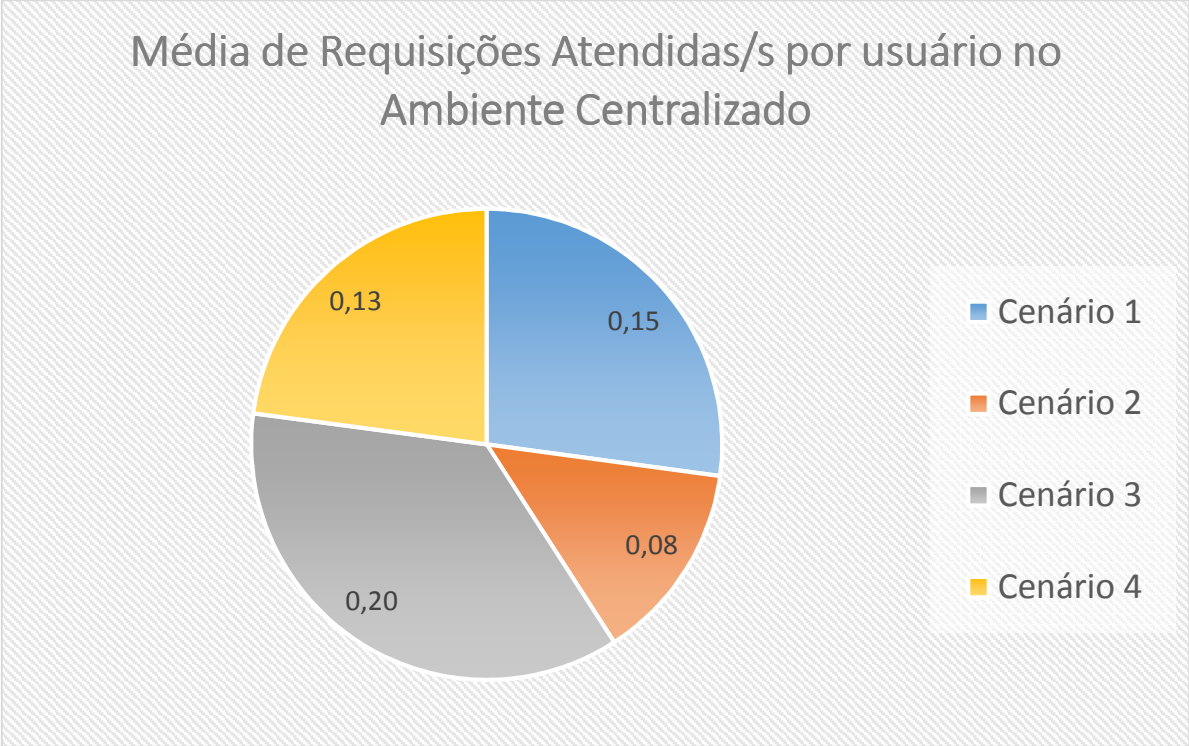


**Gráfico 11: Média de Requisições Atendidas /s no Ambiente Centralizado**

Observa-se que, no Gráfico 11, o cenário 2 apresenta uma taxa de requisições atendidas por segundo por usuário 46,66% inferior a encontrada no cenário 1. O cenário 4 apresentou uma taxa 40% superior de requisições atendidas por usuário em relação ao cenário 3. Observando estes valores pode-se verificar que mesmo o sistema tendo um gerenciamento mais otimizado de alguns recursos em relação a outros, de acordo com os tipos de perfis de usuários, a quantidade de requisições atendidas por segundo pelo sistema manteve uma média de otimização entre os cenários.

Durante a coleta dos dados foi verificado que as únicas requisições SQL encontradas nos perfis analisados foram *Select, Insert* e *Update* para os 4 cenários,

sendo que nos cenários 1 e 2 foi verificado um maior índice de *Select* e nos cenários 3 e 4 foi verificado um aumento no índice de *Insert* e *Update*. Este comportamento do sistema se justifica pelo fato dos cenários 1 e 2 comportarem apenas visualizações ao sistema, já nos cenários 3 e 4 comportarem inserções de dados.

**Tempos de resposta**

Os tempos de resposta compreende o tempo entre o envio da solicitação à aplicação *web* e sua resposta ao cliente. Para facilitar a visualização dos valores referente ao tempo de resposta da Tabela 4, estes valores foram extraídos para o Gráfico 12.

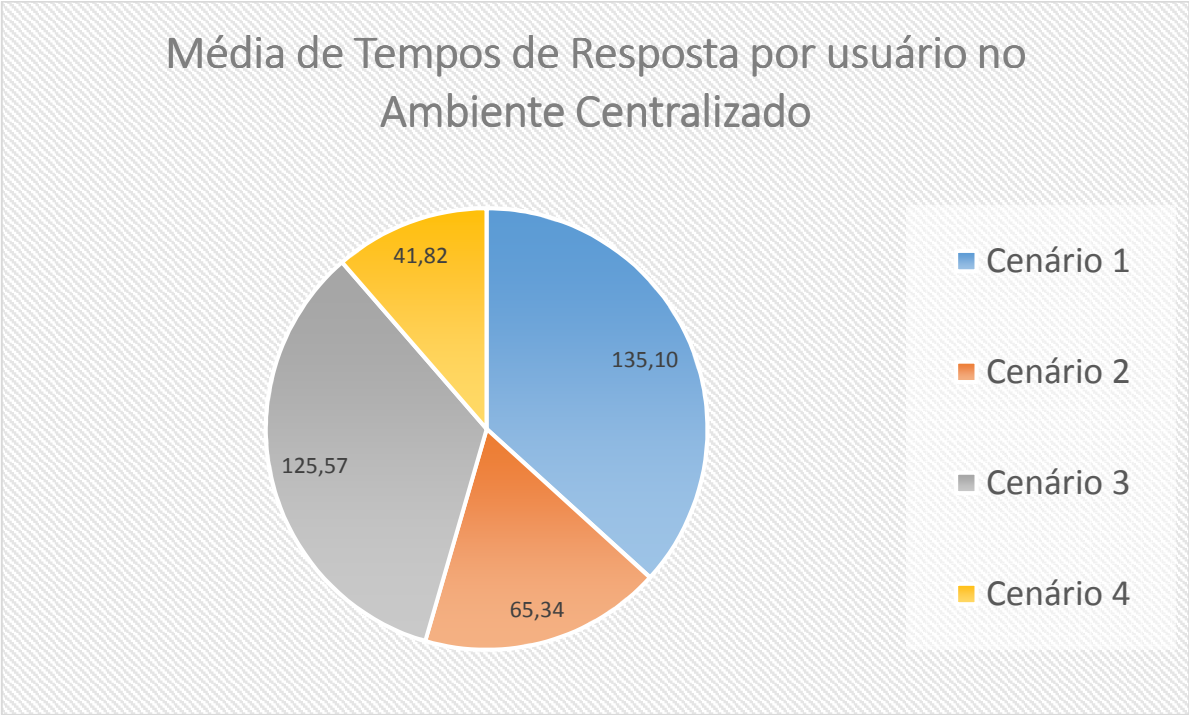


**Gráfico 12: Média de Tempos de Resposta por usuário no Ambiente Centralizado**

Observa-se no Gráfico 12 que em relação ao tempo médio de respostas, considerando o cenário 1 e o cenário 2, o segundo apresentou novamente um ganho de otimização, onde no cenário 2 a média de tempos de resposta por usuário da aplicação é 51,63% menor em relação ao cenário 1. Observa-se, também, que no cenário 4 houve uma melhora no tempo de resposta por usuário de 66,69% em relação ao cenário 3. Estes ganhos são o resultado do gerenciamento otimizado do

sistema *web* em conjunto com seu banco de dados MySQL e as ferramentas de otimização presentes como o *memcached.*

Em resumo, pode ser observado que nos resultados dos experimentos analisados, a otimização dos recursos se encontra mais acentuada nos cenários que apresentam usuários simultâneos e pertencentes ao perfil que comporta inserções de dados no sistema.

## Ambiente Distribuído

### Porcentagem de consumo de CPU

O primeiro recurso do ambiente Distribuído a ser analisado foi o consumo de CPU. Este consumo compreende a porcentagem de utilização de processador do ambiente, sendo que os valores coletados pela ferramenta MySQL Monitor são compostas pela utilização de CPU de todos os nós de dados e nós SQL presentes no Cluster. Para facilitar a visualização dos dados referentes ao consumo de CPU da Tabela 5, estes valores foram extraídos para o Gráfico 13 abaixo.

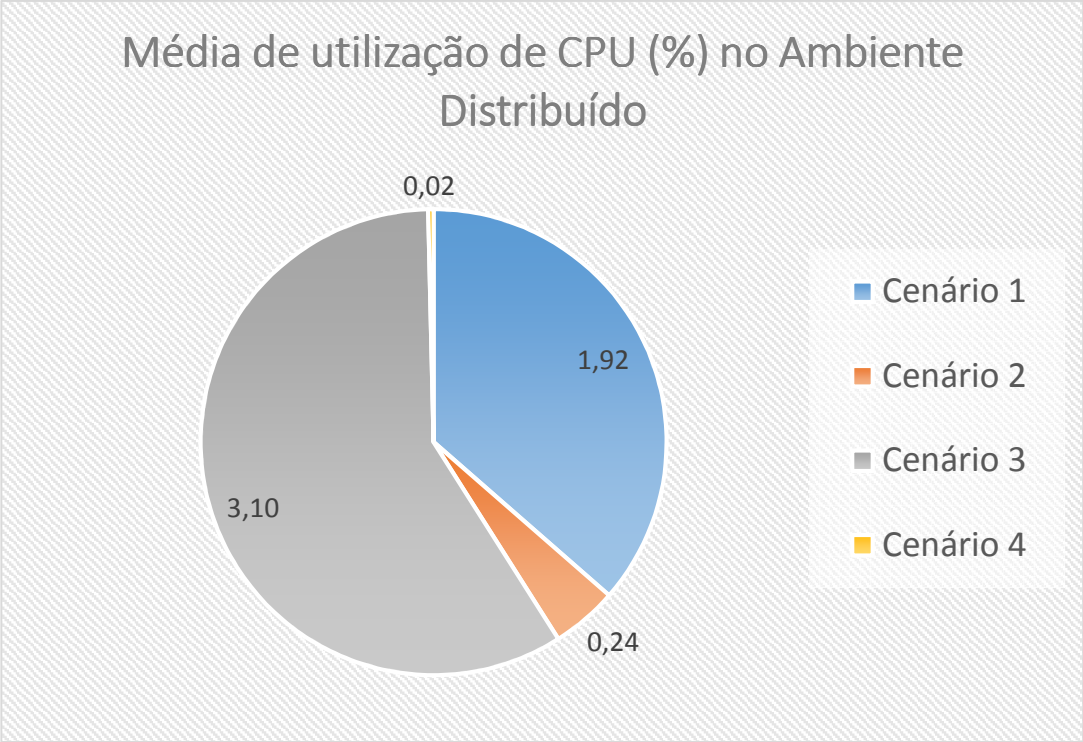


**Gráfico 13: Média de utilização de CPU (%) no Ambiente Distribuído**

Observando os valores de consumo de CPU presentes no Gráfico 13, verifica-se que o cenário 2 teve uma redução de 87,5% no consumo deste recurso por usuário em relação ao cenário 1. Sendo que este ganho acentua-se no cenário 4, apresentando uma redução de 99,35% deste consumo em relação ao cenário 3. Tais valores indicam que os recursos de otimização do ambiente distribuído, como a utilização de módulos voltados para utilização de várias *threads*, obtiveram melhor gerenciamento do recurso de CPU nos ambientes de maior acesso.

O cenário 3 apresentou um aumento de 38,06% em relação ao cenário 1. Já o cenário 4 apresentou uma redução de 91,66% em relação ao cenário 2. Esta inversão no consumo indica que o gerenciamento do recurso ocorreu de forma mais otimizada no ambiente que comportava usuários simultâneos, mostrando também que em cenários com poucos usuários e que comportavam inserções no sistema obteve-se aumento do consumo de recursos.

**Consumo de memória RAM**

O próximo recurso analisado foi o consumo de memória RAM no ambiente Distribuído, sendo que os dados compreendem a média de utilização de memória RAM nos nós SQL e de Dados que pertencem ao Cluster. Os valores deste recurso foram coletados de todo o sistema (S.O. + Aplicação + Banco de Dados) e não somente do banco de dados, devido à ferramenta *MySQL Monitor* não disponibilizar estes dados apenas do banco de dados.

Para facilitar a visualização dos valores referente ao consumo de memória RAM da Tabela 5, estes valores foram extraídos para o Gráfico 14.

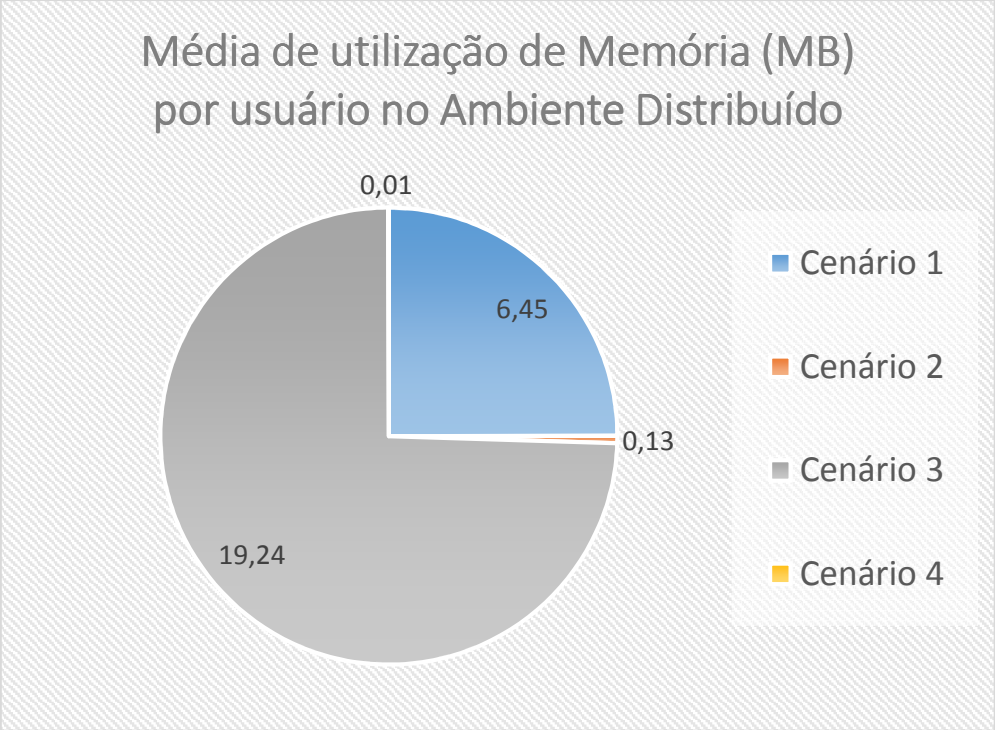


**Gráfico 14: Média de utilização de Memória (MB) por usuário no Ambiente Distribuído**

Conforme apresentado no Gráfico 14, ocorreu uma redução de 97,98% no consumo de recurso de memória RAM no cenário 2 em relação ao cenário 1. Sendo que o cenário 4 apresentou 99,94% de redução no consumo deste recurso por usuário em relação ao cenário 3. Este comportamento apresentado nos cenários que comportam usuários simultâneos mais uma vez mostra que as ferramentas de gerenciamento e otimização, como o *memcached*, comportam melhores resultados nestes tipos de cenários.

O cenário 3 apresentou um aumento de 66,47% no consumo de memória RAM por usuário em relação ao cenário 1. Já o cenário 4 apresentou uma redução de 92,30% no consumo do recurso em relação ao cenário 2. Os resultados indicam novamente que este recurso não é bem gerenciado pelo ambiente nos cenários que comportam inserções no sistema, sendo que sua otimização ocorre nos cenários de acessos simultâneos.

### Consumo de Disco

O próximo recurso a ser analisado foi o consumo de disco. Estes dados compreendem o consumo de disco em *Mega Bytes* por segundo pelo banco de dados de todos os nós de Dados e SQL que comportam o Cluster. Para facilitar a visualização dos valores referente ao consumo de disco da Tabela 5, estes valores foram extraídos para o Gráfico 15.

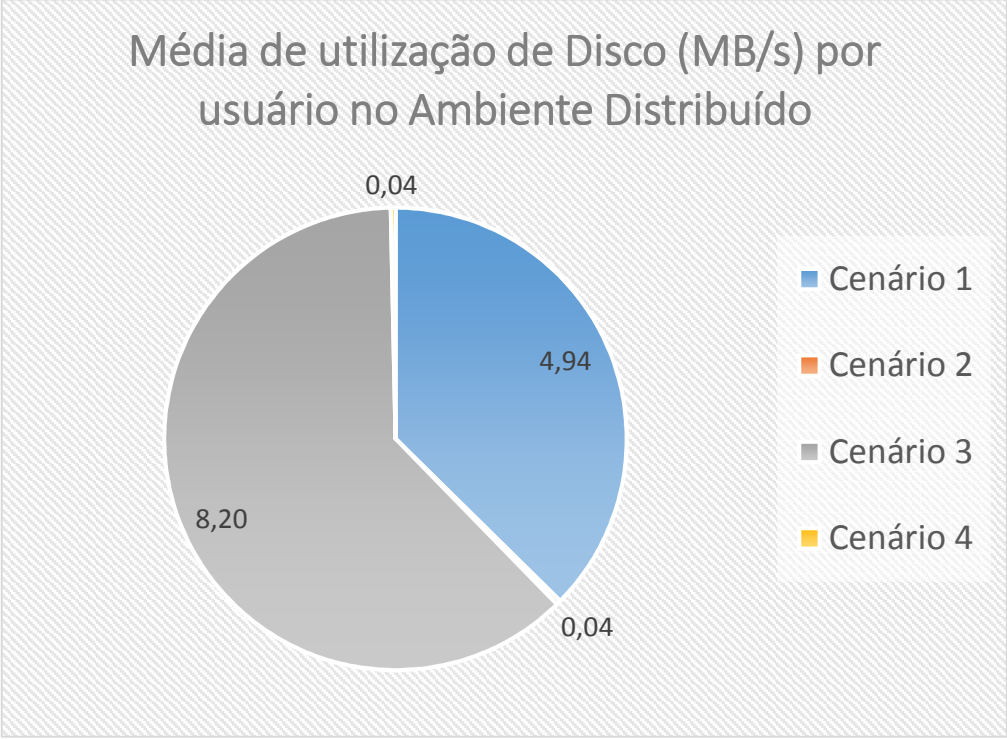


**Gráfico 15: Média de utilização de Disco (MB/s) por usuário no Ambiente Distribuído**

Conforme apresentado pelo Gráfico 15, o cenário 2 obteve uma redução de 99,19% de utilização de disco em relação ao cenário 1. Já o cenário 4 apresentou

uma redução de 99,51% de utilização do mesmo recurso em relação ao cenário 3. A otimização no gerenciamento do recurso novamente demonstrou ser mais acentuada nos ambientes de acessos simultâneos.

Observando o cenário 3, verifica-se que este apresentou um aumento de 39,75% de utilização de disco por usuário em relação ao cenário 1. E o cenário 4 não apresentou variação de consumo deste recurso em comparação com o cenário 2. Observa-se que o gerenciamento do recurso disco obteve um ganho de otimização no cenário que comportava inserções em conjunto com a ausência de acessos simultâneos, entretanto no cenário que comportava acessos simultâneos juntamente com inserções ao sistema a média de consumo deste recurso se manteve, o que ainda é considerado um ganho de desempenho pois no cenário 4 tem-se a existência de inserções no sistema em relação ao cenário 2 e está diferença não representou aumento no consumo deste recurso.

**Consumo de Rede**

O próximo recurso analisado foi o consumo de Rede pelo banco de dados. Este recurso compreende a utilização de dados em *Kilo Bytes* nas interfaces de rede de todos os nós de dados e SQL quem comportam o Cluster. Para facilitar a visualização dos valores referente ao consumo de disco da Tabela 5, estes valores foram extraídos para o Gráfico 16.

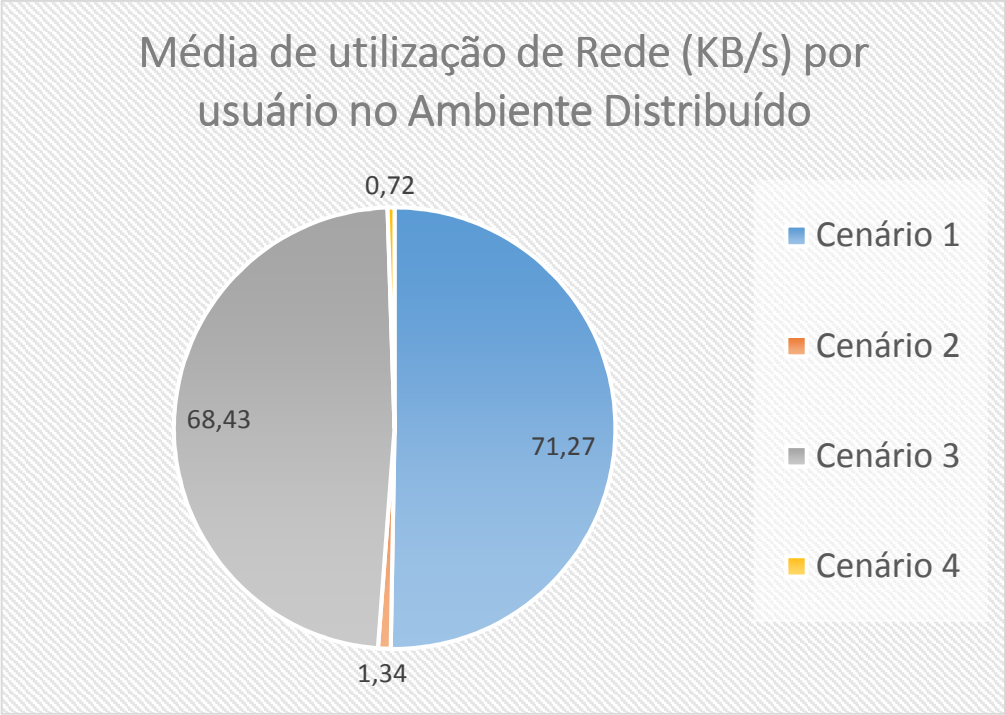


**Gráfico 16: Média de utilização de Rede (KB/s) por usuário no Ambiente Distribuído**

Conforme apresentado pelo Gráfico 16, observa-se que no cenário 2 ocorre redução de 98,11% de utilização por usuário do consumo do recurso de rede em relação ao cenário 1. No cenário 4 ocorre uma redução bem próxima a encontrada no cenário 2 em relação ao cenário 1, sendo essa redução no consumo de rede de 98,94% em relação ao cenário 3. No ambiente distribuído tem-se a utilização das funções AQL, que reduzem os saltos na rede realizando as consultas nos nós com menor custo de execução, estas consultas são realizadas nos fragmentos locais do nó de dados. Com a utilização destas funções tem-se um melhor aproveitamento da utilização do recurso de rede, sendo que o ganho em desempenho foi melhor observado nos ambientes de acessos simultâneos.

No cenário 3 observa-se uma redução de 3,98% no consumo de rede por usuário em relação ao cenário 1. Já no cenário 4 a diferença de consumo deste recurso foi de 46,26% de redução em relação ao cenário 2. Mostrando com estes valores que o ambiente trouxe melhor gerenciamento dos recursos nos cenários que comportaram inserções no sistema.

**Requisições atendidas**

Outra métrica analisada foi a quantidade de requisições SQL atendidas por segundo pelo banco de dados. Esta métrica representa a quantidade média de instruções SQL como *Select, Insert, Update, Replace, Delete* e *Call* realizadas em todos os nós de dados e SQL que comportam o Cluster. Para facilitar a visualização dos dados referente ao consumo de Rede da Tabela 5, estes valores foram extraídos para o Gráfico 17.

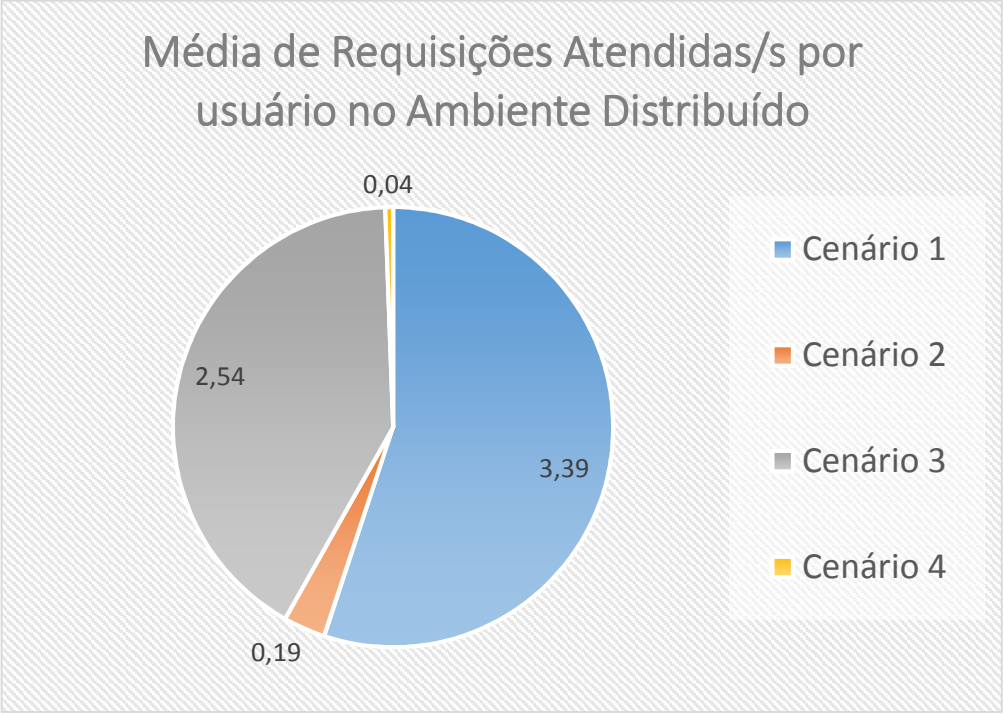


**Gráfico 17: Média de Requisições Atendidas/s por usuário no Ambiente Distribuído**

Conforme apresentado pelo Gráfico 17, o cenário 2 apresentou uma redução na quantidade de requisições atendidas por segundo de 94,39% em relação ao cenário 1. Já o cenário 4 obteve uma redução de 98,42% em relação ao cenário 3 nesta mesma métrica. Observa-se que a quantidade média de requisições atendidas por segundo obteve uma redução significativa nos ambientes de acessos simultâneos. Estes dados podem indicar que todas as ferramentas de gerenciamento e otimização de desempenho encontradas no ambiente não puderam

garantir um melhor desempenho nesta métrica para os cenários de acessos simultâneos.

O cenário 3 obteve uma redução de 25,07% na quantidade de requisições atendidas por segundo em relação ao cenário 1, e o cenário 4 apresentou uma redução de 78,94% em relação ao cenário 2. Analisando estes dados pode-se verificar que o sistema continuou a perder desempenho nesta, se tornando mais acentuado nos cenários que comportam inserções no sistema.

### Tempos de resposta

A última métrica analisada foi o tempo de resposta do sistema. Os tempos de resposta compreende o tempo entre o envio da solicitação a aplicação web e sua resposta ao cliente. Para facilitar a visualização dos valores referente ao tempo de resposta da Tabela 5, estes valores foram extraídos para o Gráfico 18.

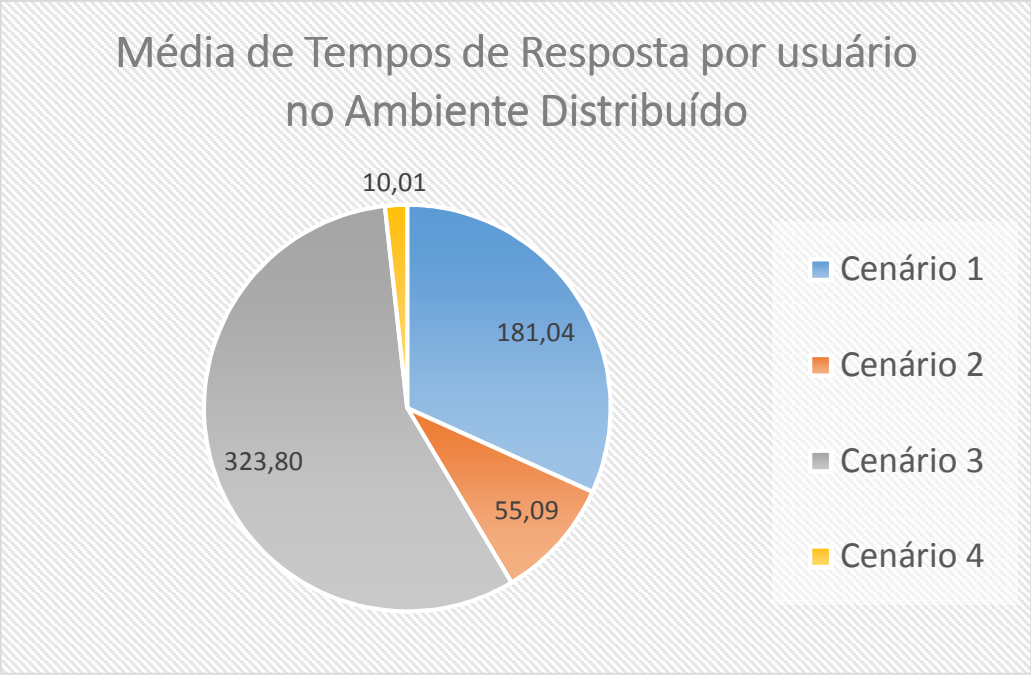


**Gráfico 18: Média de Tempos de Resposta por usuário no Ambiente Distribuído**

Conforme apresentado pelo Gráfico 18, pode ser verificado que o cenário 2 obteve uma redução nos tempos de resposta de 69,57% em relação ao cenário 1. Já

o cenário 4 obteve uma redução de 96,90% nos tempos de respostas em relação ao cenário 3. Através destes dados pode-se verificar que mesmo o ambiente distribuído não conseguindo obter ganho de desempenho nas requisições atendidas por segundo nos ambientes de acessos simultâneos, os tempos de resposta do usuário nestes cenários apresentou melhora. Sendo assim o ambiente distribuído obteve melhor desempenho nos cenários de acessos simultâneos, sendo que este ganho se tornou mais acentuado nos cenários que apresentavam inserções no sistema.

Observa-se no cenário 3 que este obteve um aumento no tempo de resposta de 44,08% em relação ao cenário 1 e o cenário 4 obteve 81,82% de redução no tempo de resposta em relação ao cenário 2. Estes valores indicam que o ambiente se comportou do forma otimizada no cenário com acessos simultâneos e que comportava cenários com inserções no sistema. Já no cenário que não comportava acessos simultâneos mas comportava inserções no sistema, este ambiente apresentou perda de desempenho.