UNIVERSIDADE TECNOLÓGICA FEDERAL DO PARANÁ
CAMPUS GUARAPUAVA
CURSO DE TECNOLOGIA EM SISTEMAS PARA INTERNET

SIMONE DOMINICO

TUNING: UM ESTUDO SOBRE A OTIMIZAÇÃO DE DESEMPENHO DE SISTEMAS GERENCIADORES DE BANCO DE DADOS RELACIONAIS SOB CARGA DE TRABALHO DE SUPORTE A DECISÃO

TRABALHO DE CONCLUSÃO DE CURSO

GUARAPUAVA
2013

SIMONE DOMINICO

TUNING: UM ESTUDO SOBRE A OTIMIZAÇÃO DE DESEMPENHO DE SISTEMAS GERENCIADORES DE BANCO DE DADOS RELACIONAIS SOB CARGA DE TRABALHO DE SUPORTE A DECISÃO

Trabalho de Conclusão de Curso de graduação, apresentado à disciplina de Trabalho de Conclusão de Curso 2 do curso de Tecnologia em Sistemas para Internet do Câmpus Guarapuava, da Universidade Tecnológica Federal do Paraná - UTFPR, como requisito parcial para obtenção do título de tecnólogo.

Área de concentração: Banco de Dados

Orientador: Prof. Ms. Tarcizio Alexandre Bini

GUARAPUAVA

2013

Ministério da Educação
Universidade Tecnológica Federal do Paraná
Câmpus Guarapuava
Curso Superior de Tecnologia em Sistemas Para a Internet

ATA DE DEFESA DE MONOGRAFIA DE TRABALHO DE CONCLUSÃO DE CURSO DO CURSO DE TSI

No dia 04 de dezembro de 2013, às 9:30 horas, nas dependências da Universidade Tecnológica Federal do Paraná Câmpus Guarapuava, ocorreu a banca de **defesa da monografia** de Trabalho de Conclusão de Curso intitulada: "**Um Estudo sobre a Otimização de Desempenho de Sistemas Gerenciadores de Banco de Dados Relacionais sob Cargas de Trabalho de Suporte a Decisão**" da acadêmica **Simone Dominico** sob orientação do professor **Prof. Me. Tarcizio Bini** do Curso de Tecnologia em Sistemas para Internet.

Banca Avaliadora	
Membro	**Nome**
Orientador	Prof. Me. Tarcizio Bini
Coorientador	
Avaliador 1	Prof. Me. Paulo Henrique Soares
Avaliador 2	Prof. Me. Diego Marczal

Situação do Trabalho

Situação	(x) Aprovado ( ) Aprovado com ressalvas ( ) Reprovado ( ) Não Compareceu
Encaminhamento do trabalho para biblioteca	(x) Pode ser encaminhado para biblioteca. ( ) Manter sigilo para publicação ou geração de patente.

Guarapuava, 4 de dezembro de 2013.

A Folha de Aprovação assinada encontra-se na Coordenação do Curso (ou Programa).

AGRADECIMENTO

A DEUS, por me conceder esta oportunidade em minha vida, me dando forças e esperanças na realização deste curso.

A meu esposo e filha pela compreensão nesta etapa tão importante em minha vida, me apoiando e auxiliando em todos os momentos necessários, mesmo quando acreditava que seria impossível .

Ao professor Tarcizio Alexandre Bini, por me orientar, nesta etapa decisiva do curso, me auxiliando nas minhas dúvidas nas dificuldades encontradas, tendo paciência e acreditando no meu potencial, dedicando seu tempo sendo meu guia na realização desta monografia.

A coordenação do Curso de Tecnologia em Sistemas para Internet, que diversas vezes deu o apoio necessário em questões acadêmicas e vem cumprindo a cada dia um papel mais essencial no curso. Por meio da coordenação me direciono aos professores que estiveram presentes nestes anos de estudo e repassaram seu conhecimento, e reconheceram meu esforço em todas as etapas da graduação .

A toda comunidade acadêmica da Universidade Tecnologia Federal do Paraná que proporcionou a estrutura necessária para o desenvolvimento das atividade relacionadas ao curso.

Aos professores da banca examinadora pelo tempo dedicado, atenção e contribuições a este trabalho.

Agradeço a todos pelo incentivo e, auxílio nesta etapa essencial para a finalização de mais esta caminhada.

“O desejo de conquista é coisa realmente muito natural e comum; e, sempre que os homens conseguem satisfazê-lo, são louvados, nunca recriminados; mas, quando não conseguem e querem satisfazê-lo de qualquer modo, aí estão o erro e a recriminação”. (Maquiavel, Niccolo)

RESUMO

DOMINICO, Simone. **Tuning: Um Estudo sobre a Otimização de Desempenho de Sistemas Gerenciadores de Banco de Dados Relacionais sob Carga de Trabalho de Suporte a Decisão**. 2013. 57 p. Trabalho de Conclusão de Curso (Tecnólogo em Sistemas para Internet) - Universidade Tecnológica Federal do Paraná. Guarapuava, 2013.

Os Sistemas Gerenciadores de Banco de Dados Relacionais, são cada vez mais exigidos para manipulação de bases de dados complexas e com grandes volumes de dados. Consequentemente apresentam a necessidade de um desempenho mais apurado. Neste contexto, podemos evidenciar o *tuning*, que consiste do fino ajuste de parâmetros do Sistema Gerenciador de Banco de Dados Relacional objetivando a melhoria de seu desempenho. Este trabalho busca realizar um estudo sobre *tuning* em sistema Gerenciador de Banco de Dados Relacional, com foco em carga de trabalho do tipo Online Analytical Processing, extremamente comum nas bases de dados. Para a simulação da carga de trabalho anteriormente citada e também para analisar o desempenho, foi empregado a metodologia do *benchmarks* TPC-H mediante o uso do *benchmark Database Tes*t 3. Foi avaliado neste trabalho, através de técnicas de *tuning*, a sensibilidade dos parâmetros de configuração de um Sistema Gerenciador de Banco de Dados, além de seu comportamento, sob carga de trabalho Online Analytical Processing e verificando os impactos em seu desempenho.

Palavras-Chave: Sistema Gerenciador de Banco de Dados Relacional. Tuning. Banco de Dados. *Online Analytical Processing*. Desempenho. Benchmark

ABSTRACT

DOMINICO, Simone. **Tuning: Um Estudo sobre a Otimização de Desempenho de Sistemas Gerenciadores de Banco de Dados Relacionais sob Carga de Trabalho de Suporte a Decisão**. 2013. 57 p. Trabalho de Conclusão de Curso (Tecnólogo em Sistemas para Internet) - Universidade Tecnológica Federal do Paraná. Guarapuava, 2013.

The Relational Database Management Systems are more and more required for handling complex databases and with large data volumes. Consequently present the need more accurate performance. In this context, we stands out the tuning, which consists of fine adjustment to parameters Relational Database Management Systems with the objective of improve their performance. This work seek to realise a study on tuning in Relational Database Management Systems focusing on workload of the type Online Analytical Processing extremely common in databases. To simulate the workload previously mentioned and also to analyze the performance, we used the methodology of the TPC-H benchmarks by using the benchmark Database Test 3. Was evaluated in this work through tuning techniques, the sensitivity the configuration parameters of a Relational Database Management Systems, and their behavior under workload Online Analytical Processing and verifying impacts on their performance.

Keywords : Relational Database Management Systems. Tuning. Database. Online Analytical Processing . Performance. Benchmark

LISTA DE FIGURAS

LISTA DE TABELAS

LISTA DE SIGLAS

BD	Banco de Dados
DBA	Database Administrator
DBT	Database Test Suite
DSS	Decision Support System
IBM	International Business Machines
SGBD	Sistema Gerenciador de Banco de Dados
SGBDR	Sistema Gerenciador de Banco de Dados Relacional
SQL	Structured Query Language
OLAP	Online Analytical Processing
OLTP	Online Transaction Processing
RF1	Refresh Function 1
RF2	Refresh Function 2
ACID	Atomicidade, Consistência, Isolamento, Durabilidade
TPC	Transaction Performance Council

SUMÁRIO

1 INTRODUÇÃO

Aplicações científicas, acadêmicas e comerciais geram diariamente um grande volume de dados. Esta afirmação é uma justificativa ao esforço despendido por fabricantes de sistemas de armazenamento e pesquisadores na área de banco de dados, com objetivo de desenvolver tecnologias que permitam eficiência no armazenamento, gerenciamento e recuperação de dados.

Devido a essas necessidades, surgiram na década de 60, os primeiros Sistemas Gerenciadores de Banco de Dados (SGBD). Tais aplicações procuram facilitar a manipulação, criação e gestão de base de dados. Neste contexto, a primeira proposta foi idealizada por Charles Bachman, denominado Depósito de Dados Integrados (RAMAKRISHNAN; GEHRKE, 2008).

Na década de 70, foi apresentado por Edgar Codd, uma estrutura de representação dos dados nomeada de modelo relacional (CODD, 1970). Baseado neste modelo surgiram os Sistemas Gerenciadores de Banco de Dados Relacionais (SGBDR). Sua primeira implementação foi denominado *System-R* em 1979 (SELINGER *et al*, 1979). A partir desta aplicação várias melhorias foram e continuam sendo incluídas aos SGBDR, o que os tornam atuais, extremamente confiáveis e utilizados no mercado.

No decorrer dos anos, surgiram diferentes exigências impostas aos SGBDR. Como exemplo, podemos citar as cargas de trabalho do tipo OLTP (*Online Transaction Processing*). Caracterizadas por pequenas transações, geralmente de escrita e leitura intercaladas, são muito comuns em sistemas bancários e *e-comerce*. Também cargas de trabalho do tipo OLAP (*Online Analytical Processing*), compostas por consultas complexas, que resultam no acesso a grandes quantidades de informações e constante acesso a disco.

Para que o SGBDR, apresente bom desempenho frente a diferentes cargas de trabalho, é necessário o ajuste de vários parâmetros de sua configuração, utilizando uma técnica denominada *tuning*. Para que esta seja realizada de forma satisfatória, são exigidos amplos conhecimentos do DBA (*Database Administrator*),

no que diz respeito a base de dados armazenada, características de hardware e do sistema operacional, além de conhecimentos prévios sobre parâmetros de configuração do próprio SGBDR.

Neste contexto, este trabalho busca demonstrar os principais parâmetros de configuração do SGBDR, que quando ajustados melhoram o desempenho das transações em ambientes OLAP, sendo escolhidos alguns parâmetros segundo DEBNATH; LILJA; MOKBEL (2008). Apresentando as regras de *tuning* propostas para cada parâmetro adotado, propiciando assim, a melhor utilização dos recursos existentes no SGBDR e, no hardware onde se encontra alocado.

Como ferramenta de estudo será empregado o SGBDR PostgreSQL (POSTGRESQL, 2012), aplicação de código fonte aberto, distribuído gratuitamente onde se torna possível e viável a aplicação e análise de regras de *tuning*. Para simulação do ambiente OLAP, será utilizado o *benchmark* TPC-H/DBT-3 (TPC BENCHMARK H,1997).

Este trabalho está dividido em 7 capítulos: o Capítulo 2 apresenta os conceitos de SGBDR, assim como as principais cargas de trabalho a ele submetidas e discute o conceito e aplicação de técnicas de *tuning*; o Capítulo 3 aborda a utilização de benchmarks aplicados a SGBDR, dando foco ao *benchmark* TPC-H; o Capítulo 4 apresenta a metodologia utilizada neste trabalho e, as configurações do ambiente de experimentos; No Capítulo 5 estarão detalhados os resultados obtidos em nossos experimentos; o Capítulo 6, por fim apresenta as conclusões alcançadas e menciona os trabalhos futuros.

2 TUNING EM SISTEMAS GERENCIADORES DE BANCO DE DADOS RELACIONAIS

O SGBD, é um software que facilita a manipulação, criação de uma base de dados e seu gerenciamento. Foram criados para auxiliar na administração de grande volume de dados. Os mais utilizados são baseados no modelo relacional de dados, denominados Sistemas Gerenciadores de Banco de Dados Relacional (SGBDR), representando os dados em forma de relações.

Os SGBDR são estudados a várias décadas por pesquisadores e DBA's que buscam aperfeiçoar suas funcionalidades e melhorar o desempenho frente a diferentes tipos de cargas de trabalhos, como OLAP e OLTP. Para isso uma das técnicas utilizadas por DBA's é denominada *tuning*.

Neste contexto, o capítulo abordará, a definição de SGBDR, as características das principais cargas de trabalho. Apresentando também o conceito de *tuning*, seus principais objetivos e quais as necessidades de se otimizar o desempenho do SGBDR.

2.1 SISTEMAS GERENCIADORES DE BANCO DE DADOS RELACIONAIS

O modelo de dados relacional foi proposto em 1970, por Edgar Codd *(CODD, 1972)* no laboratório de pesquisa da IBM (*International Business Machines*), sendo uma referência histórica em desenvolvimento de sistemas gerenciadores de banco de dados. Com isso, na década de 80, o modelo relacional consolidou sua posição como o paradigma dominante de SGBD (RAMAKRISHNAN; GEHRKE, 2008).

Ao final dos anos 70, foi desenvolvida uma linguagem de consulta própria para o banco de dados relacional. Sendo denominada SQL (*Structured Query Language*), tornou ainda mais produtivos os SGBDR. Os usuários submetem requisições ao banco de dados sob forma de instruções SQL. O SGBDR por sua vez, é responsável por executá-las. (RAMAKRISHNAN; GEHRKE, 2008).

O modelo relacional representa os dados, por meio de um conjunto de

relações (tabelas), compostas por linhas (tuplas/registros) e colunas (atributos). Para recuperação dos dados a partir das relações, são utilizados um conjunto de operações derivadas da álgebra relacional *(CODD, 1972).*

A eficiência dos SGBDR na gestão do modelo relacional, e seu aprimoramento através dos anos, o consolidou no mercado com facilidade, transformando-o no principal sistema para gestão de banco de dados. Estando presente nas aplicações mais simples, até as mais complexas, sua eficiência decorre de diversas vantagens (ELMASRI; NAVATHE, 2011). Pode-se citar a linguagem utilizada na sua manipulação (SQL), garantia de integridade dos dados, controle de concorrência, recuperação rápida de falhas, consistência dos dados, segurança, controle de transações e otimização de consultas (BRITO, 2010).

No SGBDR são executadas diferentes tipos de cargas de trabalho, algumas caracterizadas por pequenas atualizações, em contraponto, apresentam a demanda de um grande número de transações que necessitam de confiabilidade e eficiência. Outras cargas de trabalho, exigem tomadas de decisão para o retorno dos dados, aumentando a complexidade do mesmo (RAMAKRISHNAN; GEHRKE, 2008).

2.1.1 Principais Cargas de Trabalho Submetidas a SGBDR

As cargas de trabalho submetidas aos SGBDR relacionam-se com os diferentes domínios do problema especificado. Estas por sua vez, expõe constantemente a base de dados a transações, caracterizadas por sua carga de trabalho que são definidas a partir das ações realizadas. As cargas de trabalho são um conjunto de consultas e atualizações, avaliadas pela sua frequência de ocorrência (LIFSCHITZ; MILANÉS; SALLES, 2004).

Os ambientes que executam pequenas atualizações diversas vezes concorrentemente, estão presentes em uma grande parte das aplicações. São denominados OLTP, onde processam milhares de informações do banco de dados por dia, em pequenas quantidades. Executam variadas consultas e pequenas atualizações. Normalmente os sistemas OLTP manipulam as mesmas informações

diversas vezes, não se preocupando em analisar os dados. Apresentando constantes acessos à base, o que requer otimizações no desempenho do SGBDR.

Ambientes OLTP, segundo Ramakrishnan e Gehrke (2008, p. 704), estimularam o crescimento dos SGBDR, nas últimas três décadas. Responsáveis por registrar todas as transações em uma determinada operação, disponibilizando completamente seus dados 24 horas por dia e, possuindo características como eficiência e simplicidade.

Atualmente, as atenções estão cada vez mais direcionadas a aplicações que executam uma análise dos dados, antes do seu retorno. Existem ferramentas poderosas, onde as consultas utilizam operadores lógicos de agrupamento e agregação para melhor retorno. Ambientes com tais características são denominados OLAP (RAMAKRISHNAN, GEHRKE, 2008).

O termo OLAP surgiu no ano de 1990, através de Edgar Codd, onde o definiu como uma categoria de processamento de dados, elencando as exigências apresentadas na síntese de informações através dos dados (CODD, 1993). Sendo assim as cargas de trabalho OLAP, estão presentes em diversos servidores de empresas, onde busca-se obter um retorno dos dados mais especifico através da base de dados. Destaca-se assim a necessidade do *tuning*, voltado para cargas de trabalho do tipo OLAP.

2.1.2 OLAP

As cargas OLAP possuem características diferenciadas de outros ambientes encontrados nas empresas. Com o objetivo principal de auxiliar na tomada de decisões. Analisam a base de dados de forma a retornar resultados satisfatórios, permitindo uma visão conceitual multidimensional dos dados.

As aplicações OLAP, são dominadas por consultas *ad-hoc* complexas, isto significa que as variáveis contidas nas consultas sofrem alterações em seus valores a cada execução, fazendo com que a consulta não se repita. Essas consultas envolvem operadores de agrupamento e agregação (RAMAKRISHNAN, GEHRKE,

2008). Os ambientes OLAP representam os dados através de relatórios gerencias, que facilitam a formatação multidimensional dos dados (ALMEIDA, 2004), realizando uma quantidade significativa de leituras sequenciais, em um grande volume de dados.

As cargas de trabalho OLAP estão presentes em *Data Warehouse*, que geralmente armazenam dados em um único repositório. São utilizados para obter através de sua base de dados, relatórios gerencias para suporte à tomada de decisão (ALMEIDA, 2004).

As cargas de trabalho OLAP, exigem que o SGBDR esteja devidamente ajustado para obter um bom desempenho. Uma técnica muito utilizada pelos DBA's, para esta finalidade é o *tuning* do SGBDR.

2.2 TUNING DE SGBD

O conceito de *tuning* vem do inglês "*tune*", podendo ser traduzido como “ajustar”. Considerando banco de dados, realizar *tuning* significa, um ajuste fino em seus parâmetros de configuração, objetivando melhor desempenho.

O *tuning* difere das otimizações de consulta que é um processo automático executado por componentes dos SGBDR, onde converte um comando declarativo em um de execução. O *tuning* se caracteriza por ser realizado por um especialista em desempenho ou o DBA. No SGBDR possui objetivos de executar mais rapidamente as transações no banco, diminuindo o tempo de resposta do mesmo e, melhorando o desempenho geral das transações realizadas (CARNEIRO et AL, 2007).

Os SGBDR, são flexíveis e podem ser ajustados de forma a afetar seu desempenho na execução das tarefas. O desempenho pode ser definido como a eficiência dos sistemas em atingir seus objetivos. Sendo assim, avaliando o tempo de resposta que o mesmo leva para executar suas tarefas. Envolve diferentes fatores, um desses é o tipo da carga de trabalho, executada no SGBDR. Outro é o seu rendimento no processamento das operações (SILBERSCHATZ; KORTH,

2009).

Nos SGBDR, encontra-se a vantagem de modificar seus aspectos internos de configuração, ajustando conforme as necessidades de desempenho. Este envolve um componente que Elmasri e Navathe (2004) denominam de gargalo, ou seja, quando o sistema gasta mais tempo com operações pequenas que deveriam ser rápidas. O ajuste de parâmetros de configuração de um SGBDR envolve monitorar os gargalos do sistema e tentar minimizá-los. Quando minimizados os gargalos pode se obter um melhor desempenho.

Em um SGBDR, os gargalos são disputas pelos mesmos dados, onde estes são extremamente utilizado no sistema. Se as consultas fossem distribuídas ou chegassem de forma que possuíssem um pequeno espaço de tempo entre as execuções, o desempenho seria melhor. Isso não ocorre pois normalmente essas operações são aleatórias (ELMASRI; NAVATHE, 2011).

O *tuning* no SGBDR, é realizado quando se alteram seus parâmetros de configuração. A dificuldade está em, saber quais parâmetros devem ser modificados. Para isso necessita-se realizar um estudo e verificar os parâmetros que afetam os SGBDR em diferentes ambientes. Deve ser estudado cada ambiente e constatado quais parâmetros no SGBDR, melhoram seu desempenho.

Alguns SGBDR como o DB2 da IBM (DB2, 2008), possuem ferramentas que, quando executadas sugerem os valores dos parâmetros. Estas ferramentas não verificam o tipo de carga de trabalho que está sendo executada. A presença de um profissional com conhecimento destes parâmetros, o DBA, é necessário para designar o valor ideal ao parâmetro. Este quando alterado, apresente bons retornos para a carga de trabalho caracterizada na base de dados (MACHADO, 2011).

Para realizar o tuning em um determinado ambiente de banco de dados, o DBA ou especialista em desempenho precisa monitorar a aplicação. Deve-se verificar qual carga de trabalho está sendo utilizada, identificando a possível fonte de gargalo. Há necessidade de conhecer as características de cada aplicação e sistemas envolvidos, para não gerar mais gargalos (ELMASRI; NAVATHE, 2004). Deste modo, pode-se propor alternativas de configuração para o SGBDR tornando o *tuning* eficiente.

Em geral a maioria do SGBDR podem coletar internamente as estatísticas

entre elas: o armazenamento do banco de dados; quantidade de memória utilizado para as relações, índices e espaços de *buffe*r; desempenho de E/S (entradas e saídas); e o tempo da resposta das consultas e transações. Assim pode-se criar um perfil da atividade de um banco de dados. Auxiliando na melhor escolha dos parâmetros de configuração do SGBDR, onde será realizado o *tuning* (POWELL, 2005).

O *tuning* de SGBD procura resolver diretamente problemas como, a concorrência entre as transações, minimizar a sobrecarga de registro de *logs* que aumentam o armazenamentos não necessário de dados, otimizar o escalonamento de processos, e o tamanho de *buffer*, e por fim destinar recursos como disco, memória e processos buscando a eficiência na sua utilização (CARNEIRO; MOREIRA; FREITAS, 2007).

O *tuning* pode ser realizado em diferentes SGBDR, a maioria destes, possuem parâmetros similares que podem ser modificados para alcançar o desempenho almejado. Alguns manuais de SGBD trazem a descrição de parâmetros que podem ser alterados para melhorar seu desempenho, o tempo de execução das consultas e transações do SGBDR (ELMASRI; NAVATHE,2004).

Para auxilar na realização do *tuning*, devemos verificar o desempenho que o SGBDR apresenta na execução da carga de trabalho escolhida. Com a finalidade de mensurar este desempenho utiliza-se softwares denominados de *benchmark*s. Cada *benchmark* possui domínio especifico, propondo métricas relevantes às características do ambiente simulado.

3 BENCHMARKS

Benchmark é o ato de aplicar programas com cargas de trabalho a software e hardware para verificar seu desempenho e comportamento, através de valores pré-estabelecidos, mensurando assim sua performance. Os *benchmarks* são empregados na avaliação de diferentes aplicações, entre eles os SGBD.

Os *benchmarks* aplicados a SGBDs avaliam seus diferentes aspectos e, vem sendo estudado há mais de 30 anos para sua análise e aprimoramento. A próxima seção irá descrever o histórico de *benchmark* de SGBD, apresentado os primeiros projetos propostos e as inovações realizadas ao longo dos anos.

3.1 HISTÓRICO BENCHMARK

O primeiro *benchmark* aplicável a avaliação de SGBDs foi o *Wisconsin* (LIMA, 2008). Desenvolvido por David Dewit em 1983, analisava o desempenho de SGBDs relacionais utilizando-se de consultas e atualização simples e complexas. Devido a sua fácil aplicação e desenvolvimento acadêmico, conseguiu ganhar popularidade rapidamente (PESSOA, 2006).

Em 1985 com as contribuições do *Wisconsin*, surgiu o *benchmark DEBIT CREDIT*, por meio do artigo intitulado "**A Measure of Transaction Processing Power**" (GRAY, 2001). Tal *benchmark* descrevia um teste *online*, criado para avaliar o desempenho de um terminal de débito e crédito. O teste compõe as características básicas da utilização de vários terminais em uma determinada configuração de computador. Diversas operações são simuladas, como saque e deposito de um usuário, verificando quantas ações podem ser concluídas por segundo (BINI, 2010). A base de dados do *DEBIT CREDIT* era composta de: caixas eletrônicos (*ATM's-Automatic Teller Machines*), agências e clientes. As transações utilizadas eram definidas por um algoritmo de ações simples, representado pelos seguintes passos (LIMA, 2008):

- Inicialização da transação;
- Leitura da mensagem proveniente dos caixas eletrônicos;
- Leitura e atualização do registro na conta do cliente;
- Inserção do registro de auditoria no arquivo de histórico;
- Leitura e atualização dos caixas eletrônicos;
- Leitura e atualização dos registros da agência;
- Envio de mensagem do retorno no caixa eletrônico;
- Finalização da transação.

Como as avaliações apresentadas pelos *benchmarks* afetavam diretamente as empresas que produziam SGBDs, em 1987 tais organizações começaram a publicar testes, onde os resultados obtidos em transações por segundo eram muito elevadas. No ano seguinte descobriram que os testes divulgados utilizavam uma variação dos testes de *Debit Credit*, ignorando as ATM's e usuários presentes nos testes do *Debit Credit*. Isso gerou dúvidas em relação à credibilidade dos testes realizados (PESSOA, 2006).

A divergência nos resultados obtidos com o *Debit Credit*, levantou a necessidade de padrões para *benchmark* de avaliação de SGBD, onde todas as análises deveriam ser realizadas respeitando os mesmos conjuntos de métricas. Assim em 1988 iniciou-se uma discussão a respeito da normalização de *benchmarks* aplicáveis a SGBDs. Dessa discussão, surgiu um grupo composto por 8 empresas liderados por Omri Serlin. Assim foi fundando o *Transaction Performance Council* (TPC) (TPC, 2013), uma organização sem fins lucrativos, que apresentaria um papel notável nas definições de testes para SGBD (PESSOA, 2006). Ao término deste ano o conselho já contava com um total de 26 membros.

As primeiras preocupações do TPC, eram definir metodologias para avaliação de SGBD, tendo em vista o “aquecimento” da utilização de *benchmarks* e a necessidade de um padrão na aplicação de testes de performance. Atendendo aos anseios, logo nas primeiras reuniões foi definido o padrão TPC *Benchmark*- A (TPC-A) (TPC-A, 1994) lançado em 1989.

3.2 BENCHMARK TPC

O TPC oferece padrões de verificações de performance SGBDs para diferentes ambientes de trabalho, considerando quesitos de confiabilidade, integridade e consistência de dados (BINI, 2010). Os ambientes simulam a utilização do sistema pelo usuário e entradas de serviços, realizando transações entre uma aplicação e a base de dados, intermediadas pelo SGBD. O conselho TPC define métricas para avaliação de desempenho das transações no SGBD utilizado, através da verificação no número de transações concluídas em um espaço de tempo definido por: *transações concluídas por segundo* (tps) e *transações concluídas por minuto* (tpm) (TPC, 2013).

Após a criação do TPC-A, o TPC lançou diversas versões de *benchmark* aplicadas a diferentes ambientes, os quais são apresentados de forma sintetizada na Tabela 1 . A classificação como obsoleto ou não é do próprio desenvolvedor.

Tabela 1: Benchmark TPC

Benchmark TPC-Obsoletos	**Ano de desenvolvimento**	**Aplicação**
TPC-A	1989	Processamento de transações em tempo real (OLTP)
TPC-B	1990	Carga de trabalho
TPC-D	1995	Suporte a decisão (OLAP)
TPC-R	1999	Suporte a decisão (OLAP)
TPC-W	1999	Transações web
TPC-APP	2004	Servidor aplicações/web
TPC-C	1992	Processamento de Transações em Tempo Real (OLTP)
TPC-DS	2012	Suporte a decisão (OLAP)

Benchmark TPC-Obsoletos	Ano de desenvolvimento	Aplicação
TPC-H	1999	Suporte a decisão (OLAP)
TPC-E	2007	Processamento de transações em tempo real (OLTP)
TPC-VMS	2012	Suporte a decisão (OLAP) sob máquinas virtuais.

Fonte: TPC (2013).

Conforme é possível observar na Tabela 1, os *benchmarks* criados pelo TPC encontram-se em constante atualização buscando prover maior credibilidades aos seus resultados. Nas próximas seções será dado foco a *benchmark* TPC-H os quais foi empregado nos experimentos realizados neste trabalho.

3.2.1 TPC-H

O TPC-H foi desenvolvido no ano de 1999, simulando um sistema de apoio a decisão (*DSS - Decision Support System*) ou Business Intelegence (TPC-H, 1997). Este benchmark atualmente encontra-se na versão 2.8 e é composto por um conjunto de consultas orientado a negócios do tipo *ad-hoc, realizadas no banco de dados com alto grau de complexidade*. Tanto as consultas quanto os dados apresentadas pelo TPC-H, procuram expressar relevância com o comportamento em banco de dados de indústrias englobando vários seguimentos, incluindo o que cada um possui em comum (BINI, 2010).

3.2.2 Banco de Dados do Benchmark TPC-H

O esquema do banco apresentado pelo *benchmark* TPC-H, é constituído por oito tabelas, ilustradas na Figura 1, que representa também suas respectivas chaves primarias e estrangeiras.

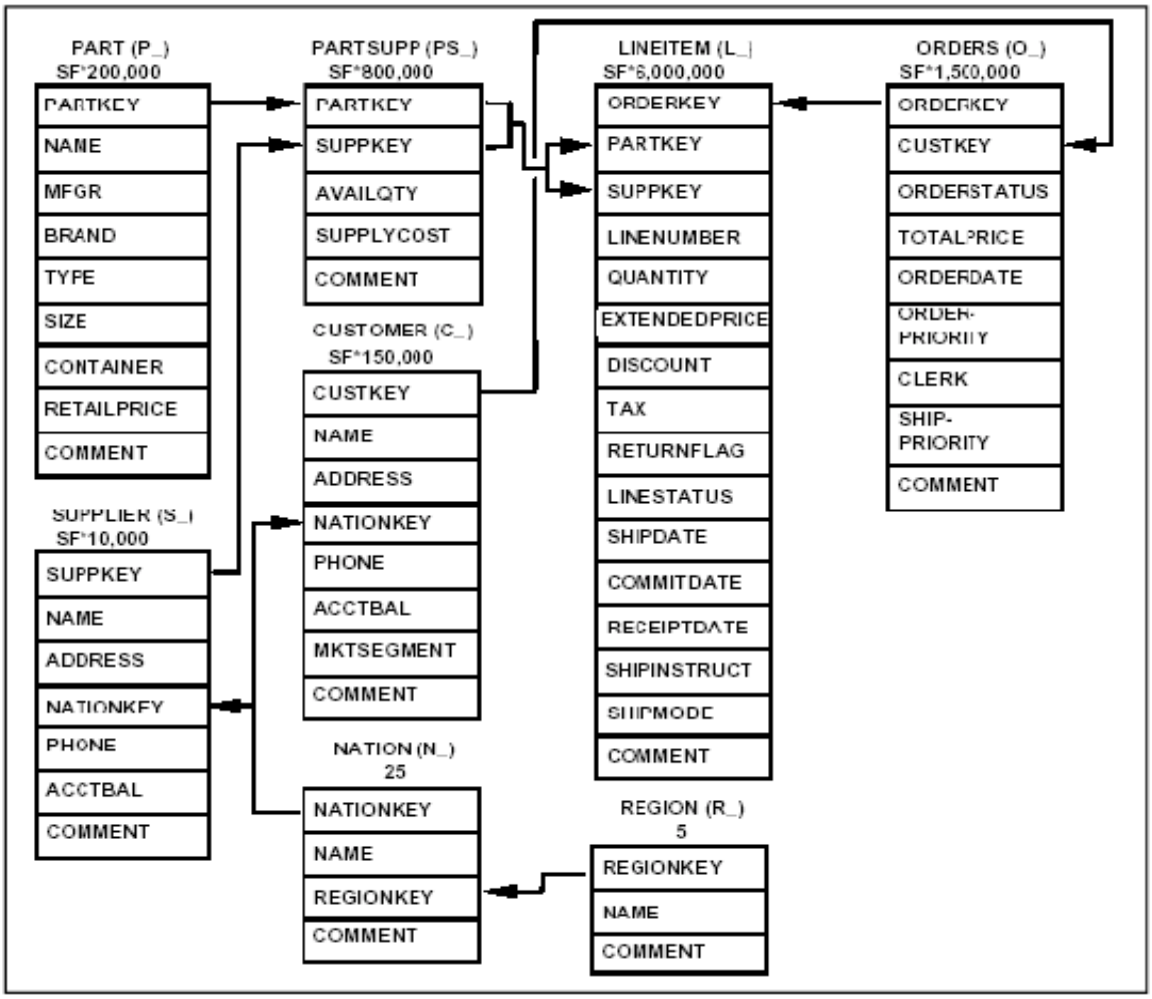

Figura 1: Esquema Banco de Dados Benchmark TPC-H 2.8.0

Fonte: TPC-H (1997).

As relações que compõem o esquema do banco de dados do *benchmark* TPC-H são divididas em *tabelas* de dimensões que apresentam as descrições dos elementos de negócios, sendo representados por *CUSTOMER, NATION*, *PART*, *PARTSUPP*, *REGION* e *SUPPLIER*. Também em tabelas de fatos armazenando

medições a respeito dos negócios e as ligações das tabelas de dimensões. Sendo as tabelas de fatos representadas por *ORDERS e LINEITEM* (BINI,2010).

A geração dos dados para popular a base do *benchmark* TPC-H, é realizada através da ferramenta denominada de DBGEN (*database generator*), escrita em ANSI C, buscando facilitar a sua utilização em diferentes plataformas (ALMEIDA, 2004). O DBGEN utiliza comando em linhas de texto a fim de auxiliar a geração de dados em banco de dados e arquivos de texto (BINI, 2010).

Utilizando o DBGEN podemos definir um fator de escala para geração dos dados que pode ser 1GB, 10GB, 30GB, 100GB, 300GB, 1000GB, 3000GB, 10000GB, 30000GB e 100000GB. A menor escala a ser definida para gerar uma base de dados para testes é de 1GB (TPC-H, 1997). O fator de escala escolhido para a geração do banco de dados, é multiplicado pelo tamanho de cada tabela, como se pode ver na Figura 1.

Tabela 2: Tamanho Estimado Banco de Dados Fator Escala 10GB (em tuplas)

Tabela	**Tuplas**	**Tuplas * FS**
Supplier	10.000	100.000
Part	200.000	2.000.000
Partsupp	800.000	8.000.000
Customer	150.000	1.500.000
Orders	1.500.000	15.000.000
Lineitem	6.001.215	59.986.052
Nation	25	25
Region	5	5
Total	8.661.245	86.586.132

Fonte: TPC benchmark H (1997).

A Tabela 2 demonstra que a tabela *NATION* e *REGION* possui tamanho das tuplas fixas, e a tabela *LINEITEM*, não deriva da multiplicação do fator de escala, sua quantidade de registros é escolhido aleatoriamente (TPC benchmark H, 1997). (ALMEIDA, 2010).

O TPC-H possui um aplicativo denominado QGEN (*query generator*),

utilizado para geração de vinte e duas consultas aplicadas ao banco de dados gerado, estas consultas são executadas de forma aleatória e seus parâmetros se alteram durante a execução (BINI, 2010).

3.2.3 Funções de Atualização e testes TPC-H

Após a geração de dados, e das vinte e duas consultas definidas, pode-se aplicar as funções de manipulação da base de dados RF1 (*Refresh Function* 1) e RF2 *(Refresh Function 2)* que compõe o TPC-H. A função RF1, é responsável pela inclusão de dados, e a função RF2 realiza exclusões. Os dados utilizados por essas funções são gerados pelo DBGEN.

A função RF1, insere dados na tabela ORDERS, a cada dado inserido nesta tabela, são realizadas de uma a sete inclusões na *LINEITEM*. A função RF2 exclui os dados, com base nos arquivos de dados gerados pelo TPC-H, onde são disponibilizadas chaves para realizar tais operações.

Além de funções de atualização existem também os testes para medir o desempenho do banco de dados, sendo eles:

- *Power* (teste de poder) – executa as funções de atualização em fluxo separado e programado antes e após as vinte e duas consultas definidas, avaliando a potência apresentada pelo sistema na execução de cada consulta, com a simulação de somente um usuário (TPC-H, 1997).
- *Throughput* (teste de produtividade) – realiza a medição da capacidade que o sistema possui para processar as consultas em maior quantidade e menor espaço de tempo, em várias sessões ou *query-streams*, executando as 2 funções de manipulação e as vinte e duas consultas. A quantidade de *query-streams* a ser executadas está diretamente ligada ao fator de escala escolhido especificado na Tabela 3 (BINI, 2010). As funções de atualizações executadas juntas podem ser denominadas de *refresh stream* (TPC-H, 1997).
- *Performance* - executado após o *power* e o *throughput*, utiliza as informações geradas, tendo em vista a capacidade de processamentos das consultas,

analisando a performance do SGBD (BINI, 2010).

Tabela 3: Query-Streams conforme fator de escala

Fator escala	Número de Streams
1	2
10	3
30	4
100	5
300	6
1000	7
3000	8
10000	9
30000	10
100000	11

Fonte: TPC benchmark H (1997).

Cada função de atualização RF1 e RF2 devem ser executadas em conjunto para garantir a quantidade inicial de dados inseridos no banco, enquanto RF1 insere dados, a função RF2 realiza a exclusão dos dados não necessários.

Os resultados obtidos pelo teste de *power* são utilizados para calcular o poder de processamento do SGBDR na execução da consulta simples. Utiliza-se do fator de escala em que o banco foi criado, o intervalo de tempo em segundos das consulta e de cada execução da função de atualização, para se obter TPC-H *Power@Size*.

Os resultados gerados no teste *throughput* são utilizados para medir a capacidade do sistema de executar as consultas e funções de atualização, em um menor tempo em ambiente multiusuário. Neste teste, o fator de escala definido influência na sequência de execução das consultas, denominadas de *streams,* sendo executadas as 22 consultas a cada *stream.* Para verificar a métrica TPC-H *Throughput@Size*, utiliza-se o número de *streams* multiplicado pelo número de consultas (TPC-H, 1997).

As especificações do TPC, exigem que a base de dados utilizada seja a mesma nos dois testes, ou seja tanto no *power* quanto no *throughput test*. A

performance do SGBD é calculada com base nos dois testes realizados sobre o mesmo banco de dados (ALMEIDA, 2010).

3.2.4 Métricas do Benchmark TPC-H

Os resultados das métricas *throughput@Size* e *power@Size* definidas pelo *benchmark* TPC-H, são utilizados para formar a principal métrica do TPC-H *Composite Queries-Per-Hour* (QphH), ou consultas compostas por hora, obtida através da média geométrica dos dois resultados. Quanto maior o valor obtido pelo QphH, melhor o desempenho encontrado (TPC-H, 1997).

A métrica *preço desempenho* utiliza o resultado encontrado na QphH, avaliando o preço total do sistema (*hardware* e *software*). Quanto menor o valor obtido por essa métrica mais rentável se torna o sistema.

Os resultados alcançados em diferentes fatores de escala, são distintos, assim o TPC-H sugere não compará-los entre si. Por não ser uma base de dados de mesmo tamanho, podem surgir diferenças nos resultados encontrados através das métricas. Da mesma forma a modificação das consultas não são permitidas pelo TPC.

Conforme Almeida (2010), a execução do *benchmark* TPC-H, no SGBDR PostgreSQL, não é possível devido a interrupção da execução do teste por *timeout*. Isso ocorre devido o PostgreSQL não criar um plano de execução satisfatório para a consulta 19 (dezenove). Neste caso, existem os *benchmarks* criados pela Linux *Foudation*, os *Toolkit Database Test*. Estes *benchmarks* são baseados nas especificações do TPC, sendo código fonte aberto e com pequenas diferenças.

3.3 BENCHMARKS DBT

Os *benchmarks* DBT apresentam ferramentas para criação da base de

dados, população e geração de relatórios de desempenho dos testes. Atualmente quem mantêm os trabalhos relacionados com os *benchmarks* DBT's, é a comunidade do SGBDR PostgreSQL (NASCIMENTO, 2013).

Apesar dos *benchmarks* DBT serem totalmente baseados nas especificações dos *benchmarks* TPCs, os testes realizados por ambos não podem ser comparados. Os cinco kits de testes DBTs são:

Tollkit Database Teste-1 (DBT-1): este kit foi baseado nas especificações do TPC-W, simulando um ambiente web de uma livraria com navegação e compra de mercadorias online.

Tollkit Database Teste-2 (DBT-2): baseado nas especificações do TPC-C, emulando um fornecedor de peças, onde são realizados atualizações de informações de clientes e estoque, além de consultas.

Tollkit Database Teste-3 (DBT-3): desenvolvido através do TPC-H, simulando ambiente de suporte e decisão, as consultas são do tipo ad-hoc. Diferencia-se do TPC-H, por não possuir a métrica *preço desempenho. Possui mais opções na escolha do fator de escala comparado ao TPC-H*. Não realiza o teste ACID(Atomicidade, Consistência, Isolamento e Durabilidade) (BINI, 2010).

Tollkit Database Teste-4 (DBT-4): apresenta a simulação de ambiente *bussines-to-bussines* web com servidor de aplicação, baseado no TPC-APP .

Tollkit Database Teste-5 (DBT-5): uma implementação de código aberto do TPC-E, apresenta a simulação de um ambiente OLTP, emulando uma corretora de ações.

Os benchmarks DBTs possuem a característica de permitir modificar as consultas definidas, adequando as necessidades encontradas. Logo o TPC proíbe tal modificação. Isto torna os benchmarks DBTs uma escolha alternativa para teste de desempenho no SGBDR PostgreSQL.

4 METODOLOGIA

Neste trabalho será verificado quais os parâmetros de configuração do SGBDR, quando realizado o *tuning*, apresentam maior sensibilidade considerando cargas de trabalho do tipo OLAP. Designando configurações de *tuning* para estes parâmetros e analisando qual destas melhorou o desempenho do SGBDR.

Para a realização do *tuning* é necessário possuir uma base de dados que possua as características necessárias, neste caso OLAP. Na simulação desta carga de trabalho, será utilizado o *benchmark* DBT3 baseado nas especificações do TPC-H. Assim este capítulo apresentará a metodologia utilizada para o desenvolvimento deste estudo, descrevendo os métodos e as técnicas empregadas, os parâmetros escolhidos para realizar o *tuning*, bem como a configuração do *benchmark* utilizado.

4.1 AMBIENTE EXPERIMENTAL

Para execução dos testes, foi utilizado o computador com Processador Intel Core i5-2410M CPU@2.30GHz, 64 bits. Este possuía 5,7 GB de memória RAM, e disco rígido de 500 GB. O sistema operacional instalado era o Linux Ubuntu 12.04 LTS com Kernel 3.5.0-41, X86_64. Neste ambiente foi instalado como ferramenta de estudo o SGBDR PostgreSQL na versão 9.2, obtido diretamente do site do desenvolvedor, e será tratado na próxima subseção.

4.1.1 PostgreSQL

O SGBDR PostgreSQL, foi criado na Universidade da Califórnia, em Berkeley, pelo Departamento de Ciência da Computação no ano de 1977, com o nome de Ingres. Atualmente o PostgreSQL é um software de código aberto, mantido pela sua comunidade, preocupada sempre em corrigir os erros apresentados, bem como a

atualização das versões. É considerado o SGBDR de código aberto mais robusto, executando em diferentes sistemas operacionais e plataformas (ALMEIDA, 2010).

O PostgreSQL, apresenta características como suporte a linguagem SQL, consultas complexas, criação de funções, transações e controle de concorrência. Também suporta variados tipos de dados, operadores, funções agregadas e linguagens procedurais, além de oferecer uma interface gráfica para a manipulação denominada pgAdmin (PGADMIN, 2013). Tornando-se assim um SGBDR bastante atrativo para utilização, por oferecer tantas funcionalidades. Isso justifica, a escolha do PostgreSQL para a execução dos testes com cargas OLAP em nosso trabalho.

4.2 PARÂMETROS DE CONFIGURAÇÃO DO SISTEMA GERENCIADOR DO BANCO DE DADOS RELACIONAL

Os SGBDR possuem variados parâmetros de configuração, onde pode-se modificá-los para obter um melhor desempenho, nas transações realizadas. Em geral, possuem valores padrões oriundos de seu desenvolvimento. O SGBDR PostgreSQL, possui aproximadamente cem parâmetros de configuração, que aceitam valores dos tipo: booleano, inteiro, ponto flutuante ou cadeia de caracteres. Os mesmos podem ser encontrados em um arquivo chamado *postgresql.conf*.

Alguns parâmetros são alterados antes mesmo que o servidor seja iniciado, é o caso dos parâmetros que definem o tipo de conexão realizada no PostgreSQL. Como o *max_connections*, que define o número de conexões aceitadas no SGBDR, por padrão possui 100 (cem) conexões (SMITH, 2010).

Outros parâmetros importantes na realização do *tuning*, são aqueles que definem a memória compartilhada pelo SGBDR, conforme o hardware onde ele se encontra instalado, e também os parâmetros que influenciam no planejamento das consultas. Neste parâmetros pode-se definir valores eficientes para melhorar o seu desempenho.

Para a execução do *tuning* considerando o número de parâmetros que este possui, percebe-se as possibilidades existentes para configuração, são inúmeros valores para cada parâmetro. Pensando no SGBDR PostgreSQL DEBNATH; LILJA;

MOKBEL (2008), exemplificam, se fosse escolhido apenas dois valores para cada parâmetro existente no SGBDR, considerando a carga de trabalho, a quantidade dos testes realizados seriam 2^{100}, um trabalho que levaria muito tempo e esforço.

Observando as afirmações de DEBNATH; LILJA; MOKBEL (2008), optou-se por escolher os parâmetros que afetam diretamente a execução da carga de trabalho OLAP. DEBNATH; LILJA; MOKBEL (2008) apresentam em seu trabalho um *ranking* dos principais parâmetros que melhoram o desempenho do SGBDR frente a cargas de trabalho OLAP.

Com base no ranking apresentando por DEBNATH; LILJA; MOKBEL (2008), os parâmetros escolhidos para aplicar o tuning são: *shared_buffers*, *effective_cache_size* e *work_mem.* Estes parâmetros se encontram nas primeiras posições deste *ranking*. Sendo assim, as próximas subseções descrevem onde cada parâmetro escolhido afeta o desempenho do SGBDR PostgreSQL, bem como os valores utilizados nos experimentos realizados.

4.2.1 Parâmetro shared_buffers

O PostgreSQL não realiza a alteração dos dados diretamente no disco, solicita uma parte da memória cache do sistema para realização das operações. O *shared_buffers* é o parâmetro que define qual parte da memória será disponível para manipular uma relação. O valor pré definido na instalação do SGBDR, corresponde a 24MB (megabytes), com seu valor mínimo especificado em 128kB (kilobyte) (BERKUS; DAITHANKAR, 2006).

Este parâmetro deve ser alterado de forma que, não utilize toda a memória disponível. Caso contrário obrigaria a execução do PostgreSQL a utilizar memória *swap*, o que levará o desempenho do banco de dados a diminuir. Definindo um valor muito pequeno para *shared_buffers*, o SGBDR não consegue realizar as operações.

Para realizar a alteração do parâmetro *shared_buffers* para mais de 32MB, existe a necessidade de aumentar o segmento de memória compartilhada definido no sistema operacional Linux. A memoria compartilhada é parte do sistema

operacional onde vários processos dividem um único pedaço desta para comunicar-se entre si. Caso este valor seja pequeno o PostgreSQL não executará e apontará um erro para este parâmetro, sendo prudente a alteração da variável *SHMMAX n*o sistema operacional, que define o máximo de memória compartilhada pelo sistema (SMITH, 2010).

4.2.2 Parâmetro *work_mem*

Outro parâmetro que considera a memória disponível pelo hardware é o *work_mem. E*ste delega a memória a ser utilizada por cada operação interna de classificação e ordenação. É importante em sistemas onde existe a presença de múltiplas operações com características complexas, tais como cargas de trabalho OLAP. A quantidade de conexões permitidas no SGBDR PostgreSQL é considerado neste parâmetro.

Cada conexão realizada paralelamente utilizará o valor definido para as operações individualmente. O valor não pode ultrapassar 1/4 da memória do sistema, considerando o número de conexões multiplicado pelo valor que deseja atribuir, e o número de ações realizadas paralelamente (BERKUS; DAITHANKAR, 2006).

Se a consulta executada, necessita de uma grande classificação dos dados, o PostgreSQL, realiza uma avaliação da dimensão das mesmas e, após compara com a memória disponível existente para este parâmetro. O valor padrão é de 1MB, podendo ter o mínimo de 64 Kb. É preciso considerar a memória que foi disponibilizada para o *shared_buffers*, para designar parte do seu restante ao *work_mem*, estimando este valor com base no números de clientes conectados (SMITH, 2010).

4.2.3 Parâmetro effective_cache_size

Este parâmetro está diretamente relacionado com o planejador de consultas do PostgreSQL. Através do mesmo, o SGBDR verifica se os planos executados são os esperados para alocar na memoria definida. A presença de valores muito baixos, não executam os índices definidos nas consultas, assim o planejador de consultas não pode utilizá-los e escolherá outro plano.

Em uma máquina dedicada à execução do SGBDR, é recomendado definir o valor do parâmetro *effective_cache_size* sendo a metade da memória RAM disponível, considerado um valor mais agressivo para a execução. Sendo 3/4 da memória, é um valor mais razoável, verificando a quantidade de clientes que utilizam o mesmo índice (SMITH, 2010).

O valor padrão definido no PostgreSQL é de 128MB. Não possui valor mínimo especificado pelo SGBDR, mas não é recomendado diminuir muito este valor, devido ao PostgreSQL não ter memória o suficiente para executar um melhor planejamento das consultas.

Estes parâmetro se encontram em outros SGBDR, com a mesma funcionalidade, mas denominados de forma diferente. Assim a próxima subseção discutirá estes parâmetros em outros SGBDR conhecidos.

4.2.4 Similaridade entre os Parâmetros dos SGBDR

Como a ferramenta de estudo escolhida foi o SGBDR PostgreSQL, a Tabela 4, apresentará um comparativo dos seus parâmetros de configuração com o MySQL (MYSQL, 2013) e ORACLE (ORACLE, 2013) que se encontram atualmente na lista dos mais populares SGBDR (DB-ENGINE, 2013).

Tabela 4: Comparativo de Parâmetros de Configuração PostgreSQL, MySQL e ORACLE.

PostgreSQL	**MySQL**	**ORACLE**
shared_buffers	*query_cache_size*	*db_cache_size*
work_mem	*join_cache_size/ sort_cache_size*	*sort_area_size/* *join_area_size* *key_area_size*
effective_cache_size	*key_cache_size/*	optimizer_index_cost_adj / *key_area_size*
max_connection	*max_connection*	max_pool_size

Fonte: MySQL (2013), ORACLE (2013).

Na Tabela 4 podemos perceber que o SGBDR MySQL e ORACLE , possuem parâmetros com funcionalidades similares com os presentes no PostgreSQL. Os parâmetros do MySQL, estão presentes em um arquivo denominado my.cnf. No SGBDR ORACLE utiliza uma ferramenta denominada SQLPLUS (SQLPLUS, 2005), que determina a alteração de seus parâmetros através de linha de comando.

Após apresentar a similaridade dos parâmetros do PostgreSQL, com o SGBDR MySQL e ORACLE, será apresentado os critérios de escolha do valores para cada parâmetro analisado.

4.2.5 Valores Definidos para Tuning do PostgreSQL

Considerando as cargas de trabalho OLAP, foi observado as recomendações definidas na documentação do PostgreSQL (POSTGRESQL, 2013), e as sugestões da referência SMITH (2010). Os dados apresentados na Tabela 5, demonstram os valores utilizados pelos parâmetros *shared_buffers*, *work_mem* e *effective_cache_size* nos experimentos realizados.

Tabela 5: Valores Utilizados nos Experimentos de Tuning

shared_buffer	work_mem	effective_cache_size
12 MB	512 kB	64 MB
24 MB*		128 MB*
48 MB	1 MB*	512 MB
64 MB		896 MB
96 MB	2 MB	1024 MB
128 MB		1390 MB
256 MB	3 MB	2048 MB
512 MB		2365 MB
	4 MB	3072 MB

Fonte: Autor (2013).

Nota: * Valor Padrão utilizado pelo SGBDR PostgreSQL.

Os experimentos realizados no parâmetro *shared_buffers,* considerando a carga de trabalho OLAP que segundo GURGEL (2013), os valores definidos para este parâmetro necessitam ser menores, devido ser uma carga de trabalho que realiza mais consultas na base de dados. Neste contexto foram determinados para *shared_buffers,* valores entre 12 MB e 512 MB como máximo, com intenção de verificar o comportamento do SGBDR frente a estes valores.

Para o parâmetro *work_mem* os valores definidos nos experimentos se baseiam, nas característica apresentadas pelo *work_mem, que* define a memória utilizada para operação de ordenação e classificação*,* que influenciam o desempenho do SGBDR em cargas de trabalho OLAP. Como esta carga de trabalho, utiliza muitos operadores lógicos de agrupamento e agregação, um valor alto é mais favorável (GURGEL, 2010). Nos testes realizados não foram definidos valores maiores que 4 MB, pois a memória disponível impediria a execução das operações. Para determinar este número máximo, foi considerando o padrão para conexões igual a 100 (cem) definidas no PostgreSQL no parâmetro *max_connection.*

No parâmetro *effective_cache_size*, foram especificados nove valores, estes iniciando do minimo de 64 MB, e chegando ao máximo de 3072 MB, sendo

correspondente a metade da memória RAM disponível pelo hardware. Segundo SMITH (2010), quando está somente executando o PostgreSQL, 3072 MB é um valor definido como agressivo para a realização do *tuning*. Entretanto foram realizados experimentos com variados valores entre o máximo possível, sendo metade da RAM e um valor minimo estabelecido para verificar o desempenho do SGBDR, executando com carga de trabalho OLAP.

Os valores adotados nos testes para *shared_buffers*, *work_mem* e *effective_cache_size*, foram obtidos através da analise da capacidade do hardware utilizado, e verificando em cada teste qual foi o desempenho alcançado frente a carga de trabalho OLAP.

Os experimentos foram executados separadamente para cada valor definido nos parâmetros escolhidos. Os testes foram iniciados com o *work_mem*, neste foram realizados 4 experimentos. Após foram empregados os 7 testes com o *shared_buffers,* e por fim com no parâmetro *effective_cache_size* foram aplicados 8 testes. A cada experimentos realizado foi coletado o desempenho apresentado pelo PostgreSQL. Após a execução de todos os testes foram analisados os resultados a fim de verificar qual valor melhorou o desempenho do SGBDR.

Após a realização individual dos testes, foram definidos 3 combinações dos parâmetros: *shared_buffers*, *effective_cache_size* e *work_men,* para verificar se é possível alcançar um melhor desempenho do SGBDR PostgreSQL, com a combinação realizada. *Os valores definidos para cada parâmetro se* baseou nos 19 experimentos executados. Assim foram adotados os valores:

- *shared_buffers* = 24MB*, work_mem* = 2MB e *effective_cache_size* = 512MB.
- *shared_buffers* = 768MB*, work_mem* = 2MB e *effective_cache_size* = 1GB.
- *shared_buffers* = 128MB*, work_mem* = 1MB e *effective_cache_size* = 1600MB.

Com estes valores foram executados mais 3 experimentos, verificando o desempenho apresentado em cada teste, e em qual combinação foi alcançado o objetivo de aumentar o desempenho do SGBDR, diminuindo seu tempo de execução das operações, baseando nos valores *default* definidos no PostgreSQL.

4.3 DEFINIÇÃO DA BASE DE DADOS UTILIZADA NOS EXPERIMENTOS

Para a simulação da carga de trabalho OLAP foi utilizado o benchmark DBT3 baseado na metodologia do benchmark TPC-H tratado no capítulo 3. Antes da criação da base de dados foi realizado a configuração do benchmark para a execução do testes para o PostgreSQL. Todas as configurações realizadas para a utilização do benchmark, foram direcionadas a diretórios especificados, para facilitar a verificação do avanço dos experimentos.

Após as configurações, foi determinado a população da base de dados com fator de escala de 10 GB. As métricas que compõe o DBT3 foram todas aplicadas nos teste. Antes da execução dos testes foi conferido a geração das 22 (vinte e duas) consultas[1]. Após a configuração do ambiente de testes, foram iniciados os experimentos, sendo separados os resultados obtidos em cada parâmetro.

Na demonstração dos resultados dos parâmetros shared_buffers, effective_cache_size e work_mem, optou-se por realizar a média dos testes *throughput* e *power*. Segundo ALMEIDA (2010), a composição dos resultados são obtidos através dos dois testes, assim foi considerado viável a apresentação da média dos mesmos.

1 https://github.com/Simone-Dominico/consultasBenchmarkDBT3

5 APRESENTAÇÃO E DISCUSSÃO DOS RESULTADOS

A primeira aplicação do *benchmark* DBT3, no SGBDR, foi utilizada para verificar o desempenho do PostgreSQL, frente a carga de trabalho OLAP, com os valores dos parâmetros oriundos de seu desenvolvimento (valores default). Este resultado está apresentado na Figura 2, onde representa o tempo de execução de cada consulta realizada nos testes das métricas *power* e *throughput*.

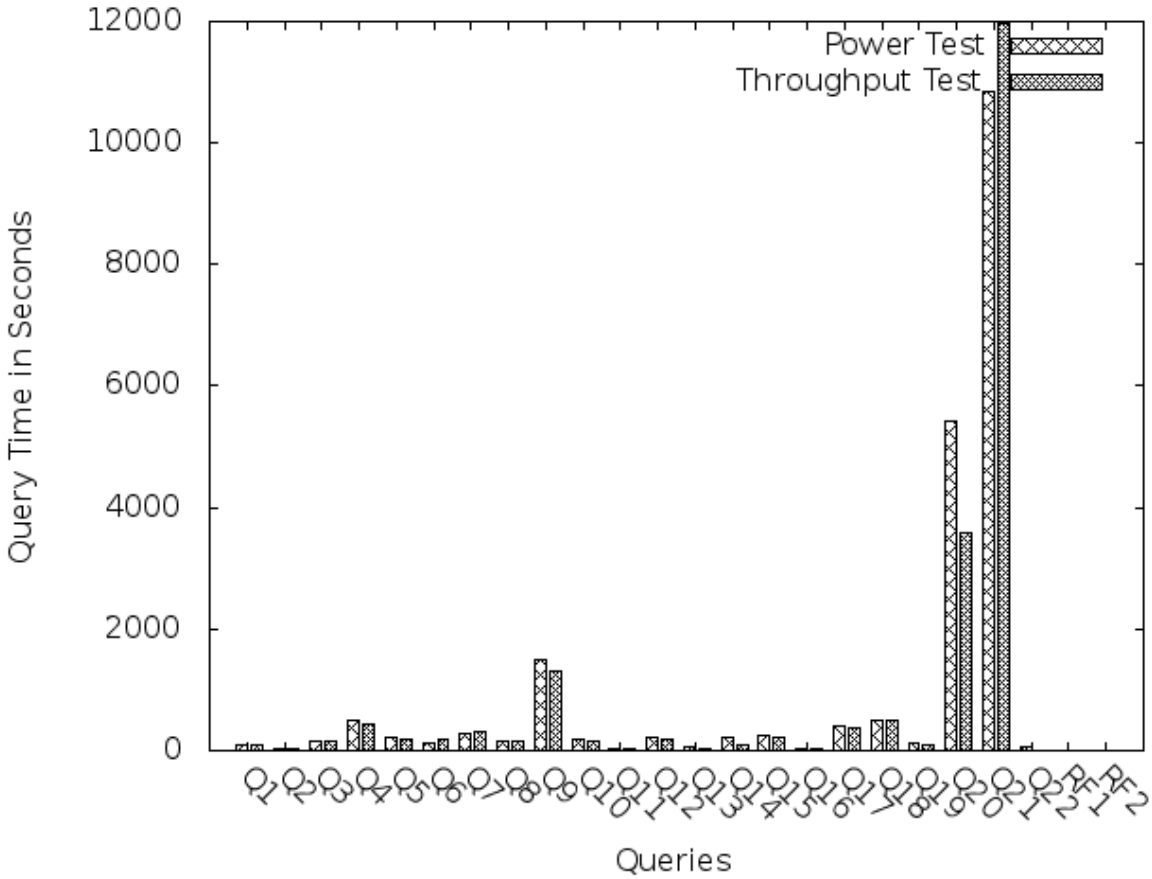

Figura 2: Resultado Teste Inicial com Valores Padrões do PostgreSQL.

As consultas Q20 e Q21, foram as consultas que mais demandaram tempo na execução do teste inicial. Na consulta Q20 o teste *power* ultrapassou o teste *throughput* , e na consulta Q21 ocorreu o contrário. Os tempos de execução de cada consulta também podem ser acompanhados na Tabela 6.

Tabela 6: Resultados Teste DBT3 SGBDR PostgreSQL sem Tuning

Consulta	Teste de Power (em segundos)	Teste de Throughput (em segundos)
Q1	100.358296	99.042219
Q2	45.243961	38.86521
Q3	161.53704	169.844929
Q4	493.196589	451.861331
Q5	233.655766	197.921477
Q6	131.412476	178.588036
Q7	285.937235	313.107826
Q8	142.125963	149.574532
Q9	1486.108473	1300.770594
Q10	180.243213	159.705597
Q11	18.773567	19.501342
Q12	224.810694	197.476793
Q13	47.454546	41.551147
Q14	218.374197	94.868884
Q15	245.010988	206.998242
Q16	27.436112	15.689121
Q17	411.878945	387.91971
Q18	513.853005	503.462202
Q19	110.059642	98.819988
Q20	5435.241496	3579.165034
Q21	10855.029671	11976.348298
Q22	57.803651	5.684184
RF1	0.859154	5.708168
RF2	0.958253	0.133979

Após verificar a performance do SGBDR PostgreSQL sem *tuning*, foram iniciados os testes considerando os parâmetros *shared_buffers, effective_cache_size* e *work_mem,* com objetivo de avaliar os valores definidos para cada parâmetro, onde verifica-se qual regra de *tuning* apresentaria impactos positivos para o PostgreSQL, considerando a carga de trabalho OLAP. Nas subseções serão descritos os resultados obtidos em cada parâmetro, e algumas combinações de valores de *shared_buffers*, *effective_cache_size* e *work_mem.*

5.1 RESULTADOS SHARED_BUFFERS

No parâmetro *shared_buffers* foram utilizados os valores definidos no Capítulo 4. Após a execução de todos os testes e a obtenção da média dos tempos de execução para cada regra de *tuning*, podem ser verificados seus impacto no SGBDR PostgreSQL na Figura 3.

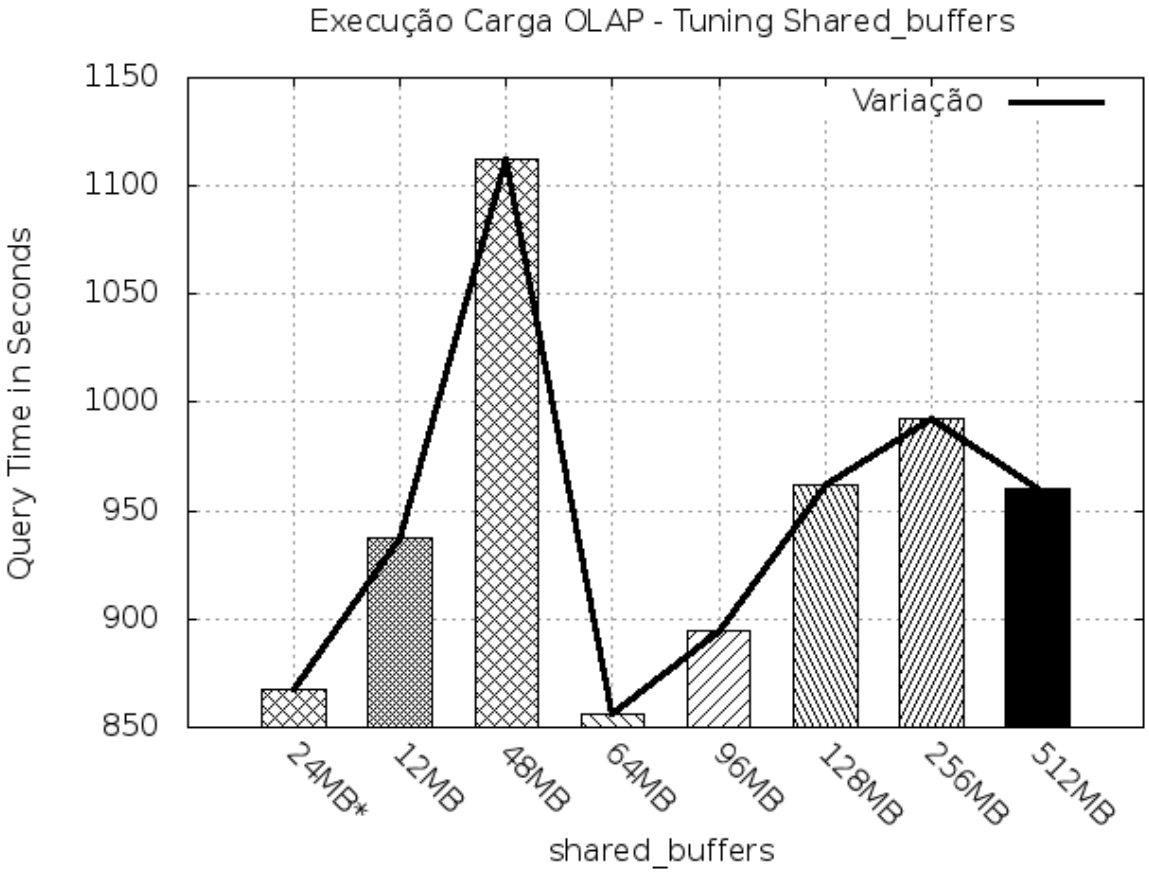

Figura 3: Resultados Tuning shared_buffers

Nota: * Valor padrão PostgreSQL.

Na Figura 3, podemos ver a variação nos testes de *tuning* considerando o parâmetro *shared_buffers,* através da média de execução das consultas das métricas *power* e *throughput.* Verifica-se que a melhor execução do SGBDR PostgreSQL, foi obtida quando especificado o valor de 64MB. Assim, executando os testes *power* e *throughput* com média de 856.18 segundos, considerando a média

do valor padrão 24 MB que executou os testes em 867.08 segundos.

Para observar o tempo de execução que cada consulta apresentou individualmente nos testes *power* e *throughput, quando obtido o* melhor desempenho com a regra de *tuning* definida com valor 64MB no parâmetro *shared_buffers.* Na Tabela 7, pode-se comparar com o teste realizado no PostgreSQL sem *tuning*. Assim verificando individualmente o desempenho das consultas.

Tabela 7: Resultados do T*uning* no Parâmetro *shared_buffers*.

Consultas	**PostgreSQL sem tuning**		**Tuning PostgreSQL**	
	Teste de *Power* (em segundos)	**Teste de *Throughput* (em segundos)**	**Teste de *Power* (em segundos)**	**Teste de *Throughput* (em segundos)**
Q1	100.358296	99.042219	100.759647	98.895567
Q2	45.243961	38.86521	40.473993	51.629161
Q3	161.53704	169.844929	167.747888	172.929906
Q4	493.196589	451.861331	491.424827	471.46422
Q5	233.655766	197.921477	205.672727	201.9719
Q6	131.412476	178.588036	138.207678	182.638138
Q7	285.937235	313.107826	289.213735	320.631303
Q8	142.125963	149.574532	158.898564	168.327519
Q9	1486.108473	1300.770594	1369.902995	1290.906736
Q10	180.243213	159.705597	158.728914	176.571995
Q11	18.773567	19.501342	32.400442	19.470769
Q12	224.810694	197.476793	231.228546	203.617232
Q13	47.454546	41.551147	57.119859	47.528894
Q14	218.374197	94.868884	105.348277	103.947046
Q15	245.010988	206.998242	240.211237	221.433356
Q16	27.436112	15.689121	29.343648	17.776928
Q17	411.878945	387.91971	401.374367	405.805133
Q18	513.853005	503.462202	483.077036	506.271025
Q19	110.059642	98.819988	111.502032	109.491936
Q20	5435.241496	3579.165034	5349.213059	3470.697295
Q21	10855.029671	11976.348298	11684.074047	10931.22835
Q22	57.803651	5.684184	60.008471	7.816833
RF1	0.859154	5.708168	4.435907	4.214973
RF2	0.958253	0.133979	0.944443	0.201612

Para a melhor visualização dos resultados da Tabela 7 do *tuning* do parâmetro *shared_buffers* com valor definido em 64 MB, pode-se observar a Figura 4.

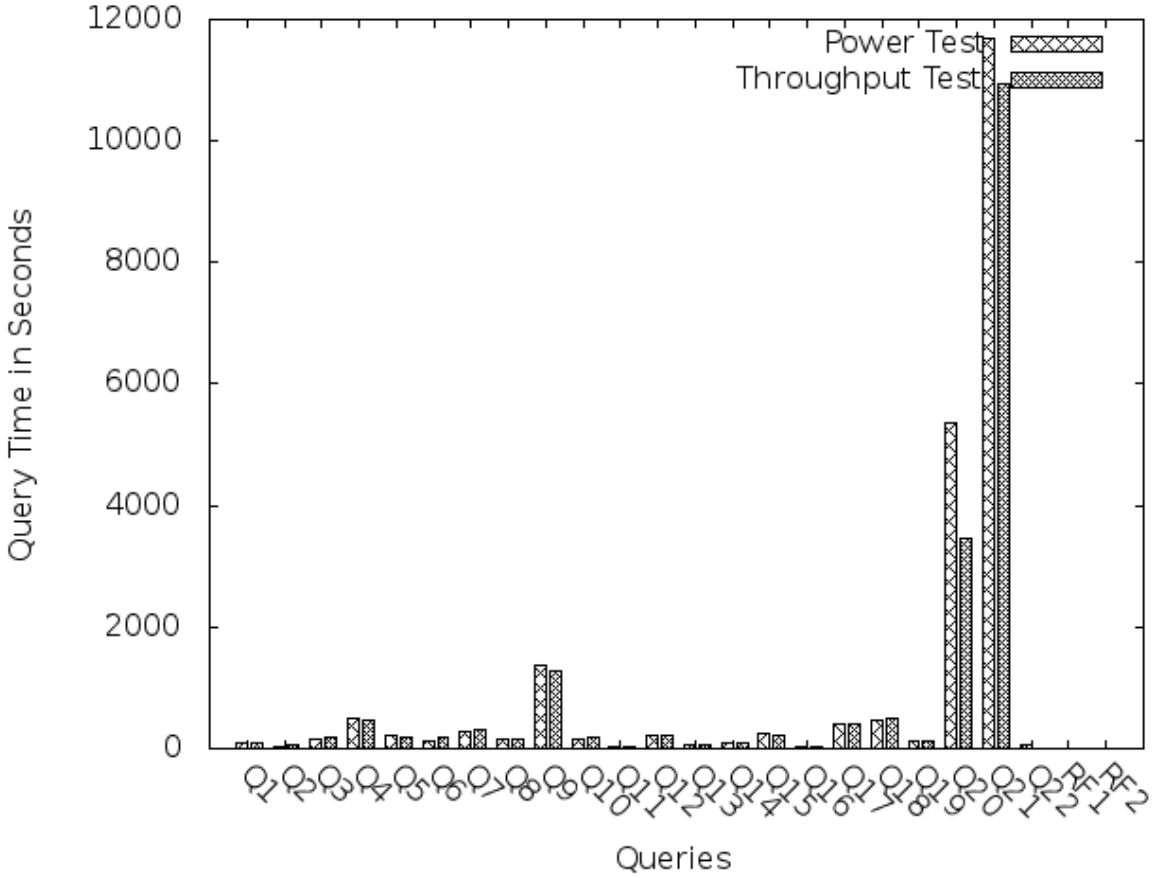

Figura 4: Resultados shared_buffers para Tuning de 64 MB

Através da analise dos resultados apresentados na Figura 4 e na Tabela 7, podemos verificar a performance do SGBDR PostgreSQL após o *tuning*. Observa-se que as consultas apresentaram melhor tempo no teste *power,* sendo desta juntamente com a função RF2. Sendo elas: Q2, Q4, Q5, Q9, Q10, Q14, Q15, Q17, Q18 e Q20. No teste *throughput,* as consultas Q1, Q9, Q11, Q20, Q21 e as funções RF1 e RF2 apresentaram melhor desempenho. Destacando a consulta Q20, que nos dois testes diminui quase 100 segundos.

5.2 RESULTADOS WORK_MEM

No parâmetro *work_mem* foram realizados quatro testes, os resultados obtidos estão apresentados na Figura 5, onde se encontra as médias dos experimentos executados, e do valor padrão do PostgreSQL.

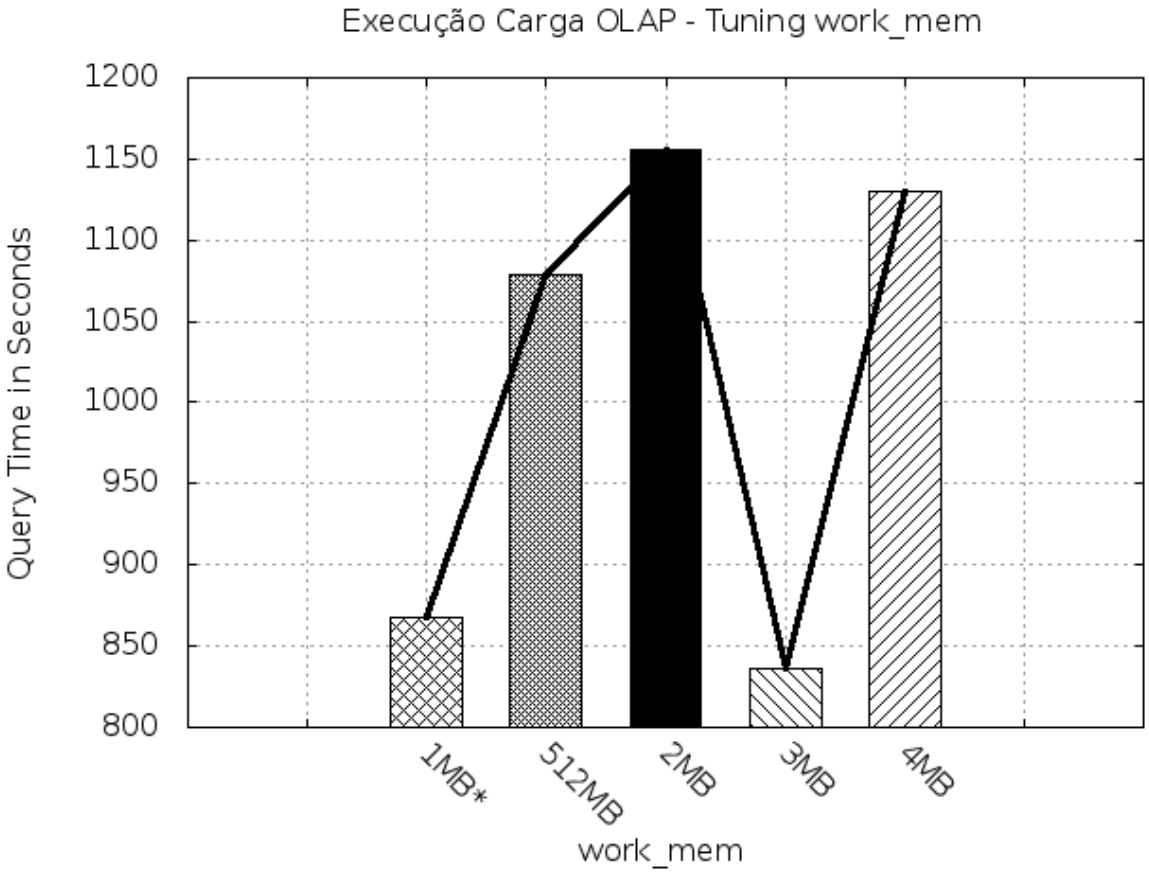

Figura 5: Resultados Tuning work_mem
Nota: * Valor padrão PostgreSQL.

O tuning no *work_mem* com melhor desempenho, foi o valor definido em 3MB. Neste teste a média de execução foi de 836.14 segundos. A execução de cada consulta e das funções RF1 e RF2, no melhor desempenho obtido podem observados na Tabela 8. Ainda pode ser comparado com o teste realizado com os valores padrão do PostgreSQL, sem *tuning*.

Tabela 8: Resultados do *Tuning* no Parâmetro *work_mem.*

Consultas	PostgreSQL sem tuning		Tuning PostgreSQL	
	Teste de *Power* (em segundos)	**Teste de *Throughput* (em segundos)**	**Teste de *Power* (em segundos)**	**Teste de *Throughput* (em segundos)**
Q1	100.358296	99.042219	100.835132	101.633993
Q2	45.243961	38.86521	46.737533	44.186747
Q3	161.53704	169.844929	152.224346	167.18512
Q4	493.196589	451.861331	448.921342	427.100821
Q5	233.655766	197.921477	158.856734	159.769144
Q6	131.412476	178.588036	155.487456	212.018508
Q7	285.937235	313.107826	267.296203	300.335107
Q8	142.125963	149.574532	152.76923	171.44569
Q9	1486.108473	1300.770594	1673.722889	1489.197011
Q10	180.243213	159.705597	159.054835	179.227204
Q11	18.773567	19.501342	32.996934	27.305504
Q12	224.810694	197.476793	249.934723	227.396241
Q13	47.454546	41.551147	47.384747	44.906094
Q14	218.374197	94.868884	95.989846	103.350003
Q15	245.010988	206.998242	275.236488	253.510391
Q16	27.436112	15.689121	28.002339	21.286305
Q17	411.878945	387.91971	429.647277	398.753025
Q18	513.853005	503.462202	463.716017	453.282392
Q19	110.059642	98.819988	108.8566	112.401987
Q20	5435.241496	3579.165034	5026.777541	3390.280723
Q21	10855.029671	11976.348298	10818.696732	10873.762551
Q22	57.803651	5.684184	65.877246	8.113382
RF1	0.859154	5.708168	4.173043	3.994046
RF2	0.958253	0.133979	1.31406	0.175078

Analisando a Tabela 8, podemos perceber que 50% das consultas apresentaram um melhor desempenho com o *tuning.* Novamente o teste *power, obteve* melhores resultados, sendo este obtido nas consultas: Q3, Q4, Q5, Q7, Q10, Q13, Q14, Q18, Q19, Q20 e Q1. O teste *throughput* apresentou melhor desempenho nas consultas: Q3, Q4, Q5, Q7, Q18, Q20, Q21 e na função RF1. Em muitas consultas, o tempo de execução diminui em mais de 100 segundos

comparado com o desempenho sem *tuning*, e as consultas Q20 e Q21, onde seu tempo de execução era mais elevado, melhorou seu desempenho tanto no teste *power* e *throughput*. A execução das consultas são apresentadas na Figura 6 , onde são demonstradas individualmente seu tempo de execução.

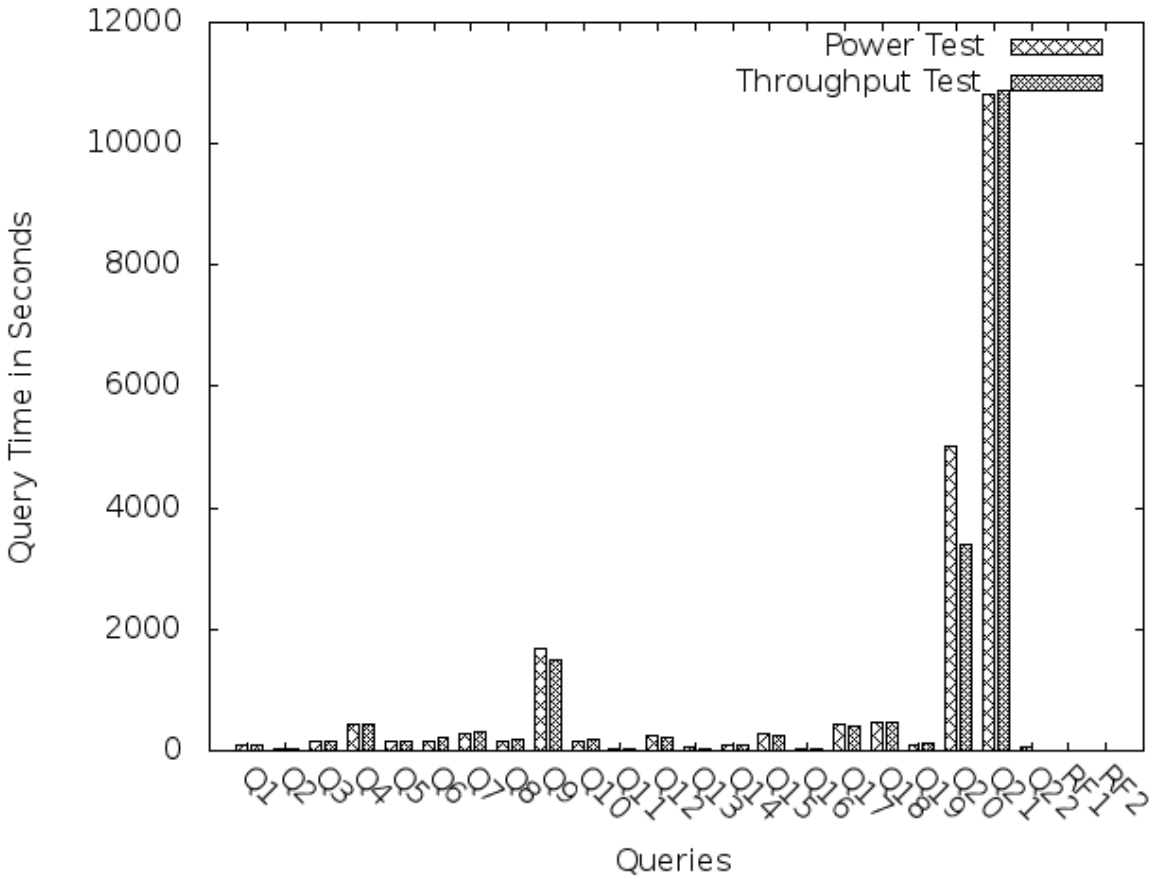

Figura 6: Resultados work_mem para Tuning de 3 MB

Assim podemos constatar que as consultas realizadas nos testes *power* e *throughput* com a configuração de *tuning* no parâmetro *work_mem* com valor de 3 MB obteve uma melhor execução em relação o valor padrão definido pelo PostgreSQL.

5.3 RESULTADOS EFFECTIVE_CACHE_SIZE

No parâmetro *effective_cache_size* foram executados 8 experimentos, estes podem ser analisados na Figura 7, onde apresenta a média na execução das configurações de *tuning* definidas para o parâmetro.

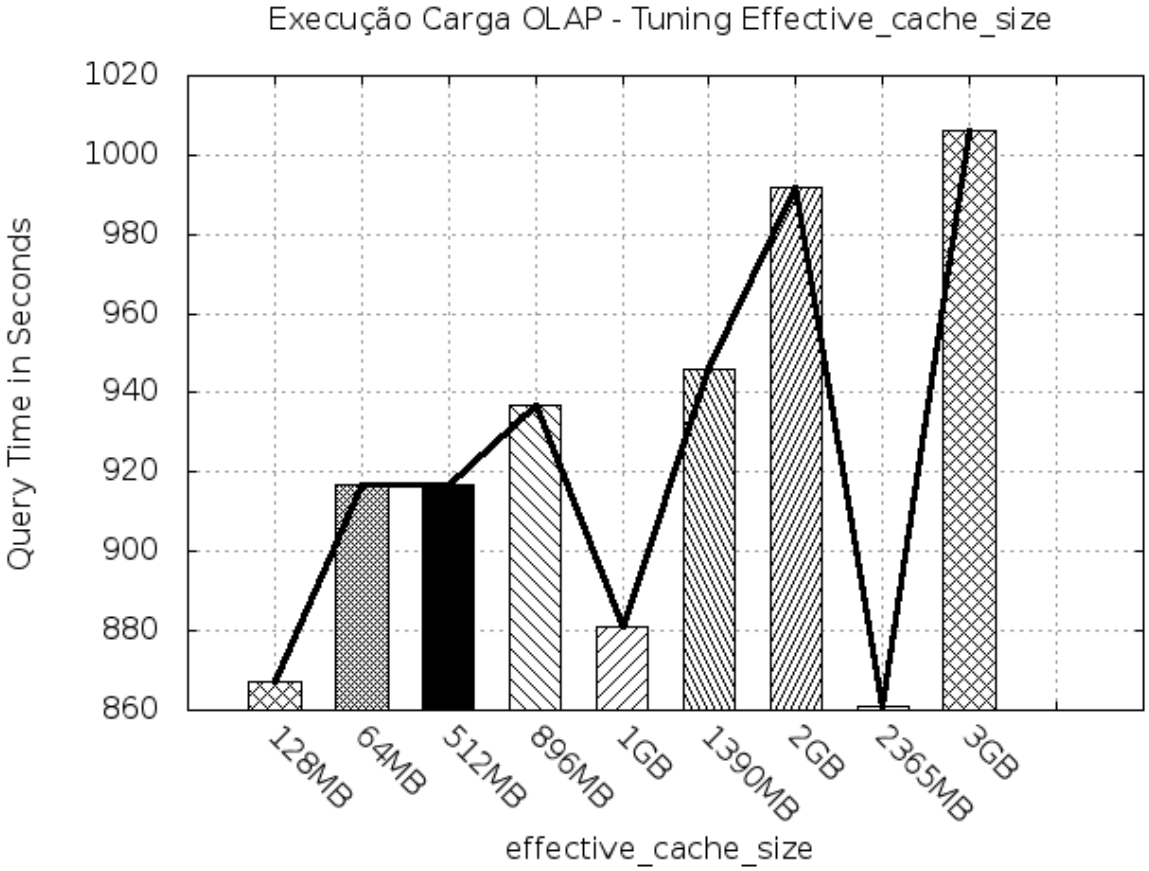

Figura 7: Resultados Tuning effective_cache_size
Nota: * Valor padrão PostgreSQL.

O melhor desempenho encontrado nas regras de *tuning* definidas, foi obtido quando o valor de *effective_cache_size* aumentou para 2365MB, correspondendo a 41% da memoria RAM disponível. Pode-se constatar na Tabela 9 o total da execução individual de cada consulta, quando definido o parâmetro com 2365MB. O valor para *effective_cache_size* definido em 3072MB que corresponde a 50% da memória do hardware, aumentou sua média de execução, diminuindo o

desempenho do SGBDR.

O melhor desempenho foi encontrado na regra de *tuning com valor de 2365MB*, que pode ser verificado na Tabela 9, onde representa a execução das consulta individualmente, juntamente com a execução sem *tuning* presente na mesma tabela.

Tabela 9: Resultados do *Tuning* no Parâmetro *effective_cache_size*.

Consultas	PostgreSQL sem tuning		Tuning PostgreSQL	
	Teste de *Power* (em segundos)	Teste de *Throughput* (em segundos)	Teste de *Power* (em segundos)	Teste de *Throughput* (em segundos)
Q1	100.358296	99.042219	99.359801	102.111705
Q2	45.243961	38.86521	44.532099	51.052733
Q3	161.53704	169.844929	129.527674	163.30104
Q4	493.196589	451.861331	451.045053	450.336179
Q5	233.655766	197.921477	144.784178	185.676944
Q6	131.412476	178.588036	152.848492	211.328536
Q7	285.937235	313.107826	244.935062	391.558706
Q8	142.125963	149.574532	164.646094	187.964426
Q9	1486.108473	1300.770594	1436.510604	1567.739552
Q10	180.243213	159.705597	142.874839	182.386199
Q11	18.773567	19.501342	212.307116	211.425347
Q12	224.810694	197.476793	251.177225	221.850076
Q13	47.454546	41.551147	41.925761	46.477133
Q14	218.374197	94.868884	101.152632	107.325527
Q15	245.010988	206.998242	291.348222	243.510914
Q16	27.436112	15.689121	28.318251	27.077844
Q17	411.878945	387.91971	397.077136	398.490531
Q18	513.853005	503.462202	476.635311	472.448971
Q19	110.059642	98.819988	98.203111	112.915633
Q20	5435.241496	3579.165034	3987.787326	4688.864939
Q21	10855.029671	11976.348298	11386.069479	10923.979969
Q22	57.803651	5.684184	79.145536	8.704819
RF1	0.859154	5.708168	4.4607	5.443177
RF2	0.958253	0.133979	1.024243	0.452385

Na Tabela 9 podemos observar, que a consulta Q3 melhorou seu

desempenho tanto no teste *power* e *throughput.* Outra questão importante que pode ser verificada, são as 14 consultas que apresentaram um desempenho superior ao valor padrão correspondendo a 64% das 22 consultas existentes no *benchmark* DBT3, sendo estas: Q1, Q2, Q3, Q4, Q5, Q7, Q9, Q10, Q13, Q14, Q17, Q18, Q19 e Q20. O teste de *throughput* obteve resultados satisfatórios nas consultas: Q3, Q4, Q5, Q18, Q21 e a função RF1. A Figura 8 demonstra por meio de gráfico, o desempenho individual das consultas na melhor regra de *tuning* que definiu o valor de *effective_cache_size* com 2365MB.

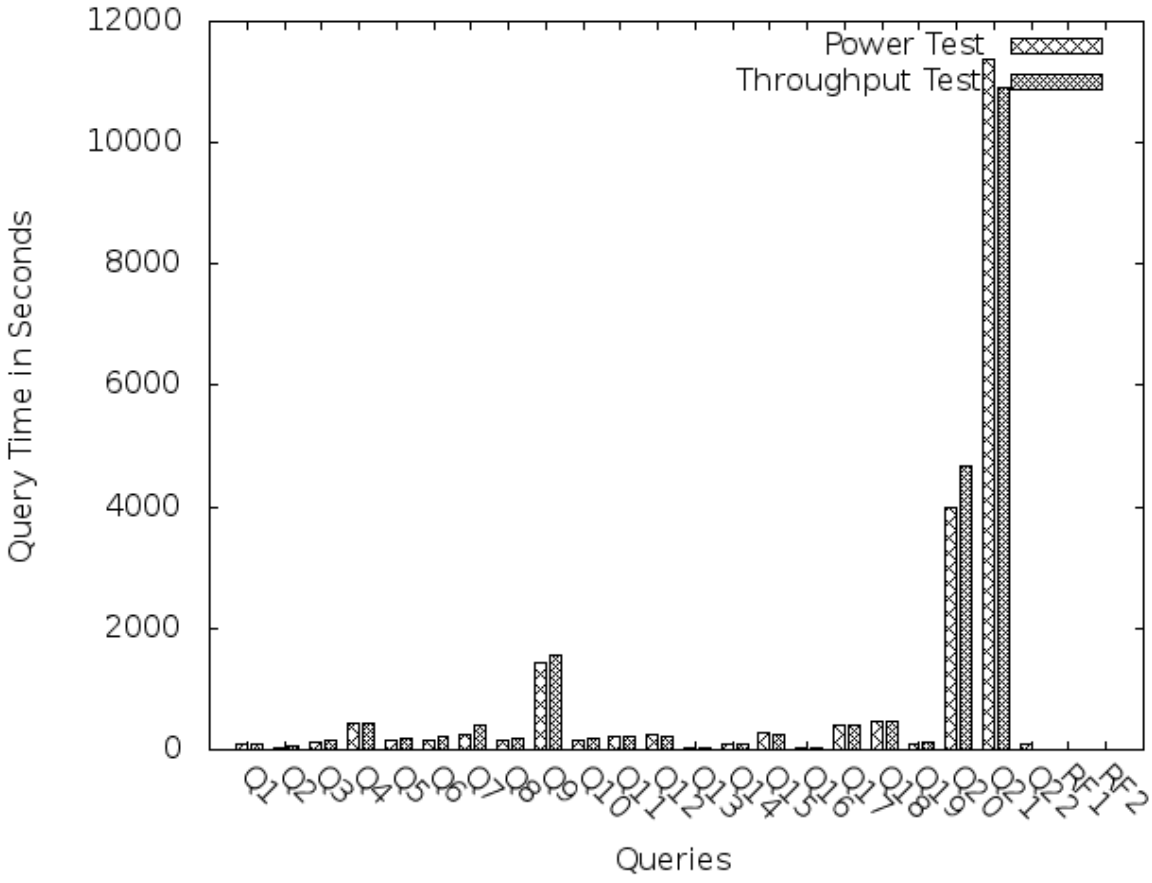

Figura 8: Resultados effective_cache_size para Tuning de 3 MB

Os resultados obtidos com *shared_buffers, effective_cache_size e work_men*, estimularam a realização de experimentos realizando algumas combinações destes parâmetros, a próxima seção apresentará os resultados obtidos com estas regras de *tuning*.

5.4 COMBINAÇÕES DOS PARÂMETROS ESCOLHIDOS

A combinação de parâmetros é complexa, devido a infinidade de valores que podem ser adotados para cada parâmetro escolhido. Neste sentido foram designados algumas configurações de *tuning*. Com isto foram realizados três experimentos estes são demonstrados através de sua média de execução na Figura 9.

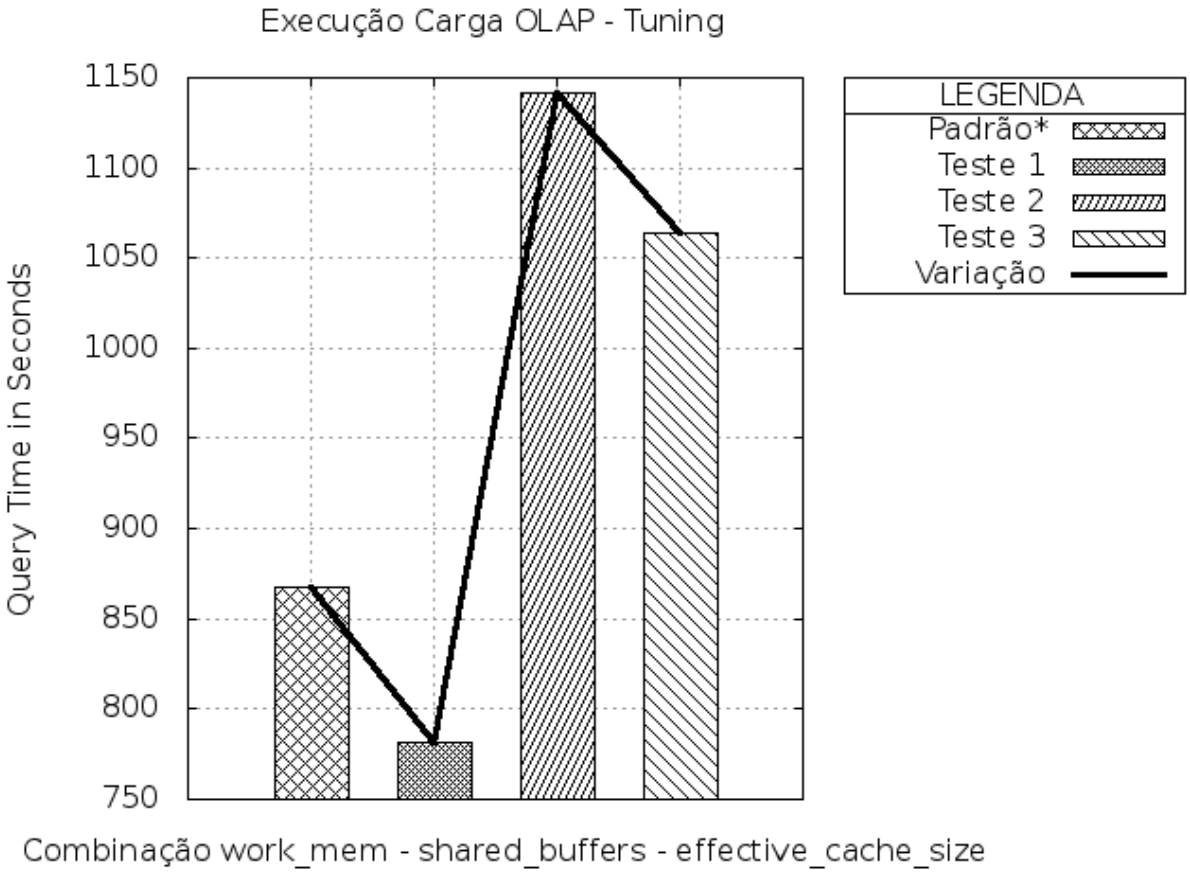

Figura 9: Resultado tuning da combinação do shared_buffers, effective_cache_size e work_mem.

Nota: * Valor padrão PostgreSQL.

Na Figura 9, no *Teste 1* foram utilizados os valores de: *shared_buffers* 24 MB, *work_mem* com 2 MB e *effective_cache_size* com 512MB. O *Teste 2* compõe por *shared_buffers* com 768MB, *work_mem* com 2MB e *effective_cache_size* com 1GB. O *Teste 3* realizado com *shared_buffers* 128MB, *work_mem* 1MB e

effective_cache_size com 1600MB. Analisando a Figura 9, percebemos que o *Teste 1*, apresentou o melhor desempenho que os valores padrões dos parâmetros definidos no SGBDR PostgreSQL, com sua média de execução reduzida em 85.38 segundos. Podendo ser observada na Tabela 10 os resultados individuais de cada consulta executa nos testes *power* e *throughput*, que pode ser comparado com os dados do SGBDR sem tuning.

Tabela 10: Comparação dos Resultados da Combinação dos Parâmetros com o Padrão do PostgreSQL

Consultas	**PostgreSQL sem tuning**		**Tuning PostgreSQL**	
	Teste de *Power* (em segundos)	**Teste de *Throughput* (em segundos)**	**Teste de *Power* (em segundos)**	**Teste de *Throughput* (em segundos)**
Q1	100.358296	99.042219	100.939355	99.7388
Q2	45.243961	38.86521	41.300646	54.483579
Q3	161.53704	169.844929	152.765668	166.901828
Q4	493.196589	451.861331	446.462703	491.874753
Q5	233.655766	197.921477	158.324913	167.389877
Q6	131.412476	178.588036	153.017478	175.638345
Q7	285.937235	313.107826	295.39121	280.476276
Q8	142.125963	149.574532	146.721005	160.688924
Q9	1486.108473	1300.770594	1309.441579	1282.589329
Q10	180.243213	159.705597	153.141267	172.66668
Q11	18.773567	19.501342	32.604518	21.168263
Q12	224.810694	197.476793	255.889536	198.239258
Q13	47.454546	41.551147	52.162421	47.202325
Q14	218.374197	94.868884	110.525236	100.751761
Q15	245.010988	206.998242	275.167344	215.366848
Q16	27.436112	15.689121	27.282757	22.529925
Q17	411.878945	387.91971	413.900216	354.349841
Q18	513.853005	503.462202	480.314191	470.741806
Q19	110.059642	98.819988	104.794954	109.80316
Q20	5435.241496	3579.165034	5850.565955	3563.459071
Q21	10855.029671	11976.348298	11358.415411	10592.484334
Q22	57.803651	5.684184	57.304979	8.375477
RF1	0.859154	5.708168	4.394718	4.368672
RF2	0.958253	0.133979	1.07999	0.116568

A tabela 10, exibe o desempenho de execução das consultas individualmente, através dela podemos perceber que na métrica de *power* as consultas Q3, Q5, Q6, Q7, Q9, Q10, Q14, Q16, Q18, Q19 e Q22, apresentaram melhor desempenho sendo 50% do total das consultas neste teste. No *throughput* 41% das consultas executaram em menor tempo, sendo estas a Q3, Q5, Q6, Q7, Q9, Q17, Q18, Q20, Q21, juntamente com as funções de RF1 e RF2. Podendo ser identificado visualmente na Figura 10.

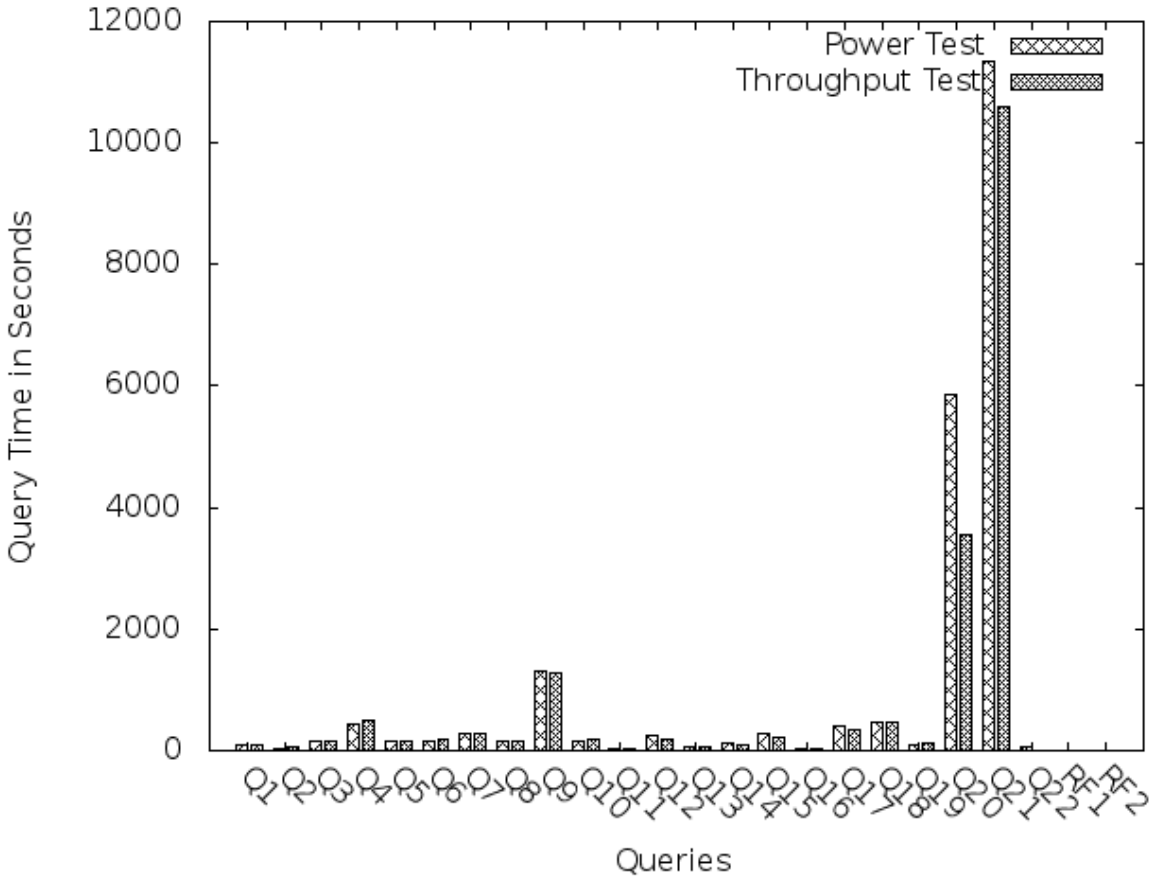

Figura 10: Resultados Combinações de valores nos Parâmetros para Tuning

A Figura 10 demonstrado desempenho individual das consultas realizadas tanto no teste *power* quanto *throughput* . Nas consultas Q21 e Q20 apresentaram melhor desempenho no *throughput test*.

5.5 DISCUSSÃO DOS RESULTADOS

Analisando os resultados apresentados podemos verificar que entre as regras definidas de *tuning* em cada parâmetro adotado, foi possível identificar a melhor configuração individual, bem como a combinação que aumentam o desempenho do SGBDR PostgreSQL na execução de carga de trabalho OLAP. Sendo possível quantificar em segundos o melhor tempo obtido com o *tuning*.

Foi constatado que os parâmetros escolhidos *shared_buffers*, *effective_cache_size* e *work_mem* quando adotado as regras de *tuning* pode se obter uma configuração para cada parâmetro. Onde através destas o SGBDR PostgreSQL apresenta um melhor desempenho. As configurações de *tuning* que demonstraram um resultado satisfatório quando especificado para *shared_buffers* 64MB, *work_mem* 3MB e *effective_cache_size* 2365MB.

No parâmetro shared_buffers podemos verificar que um valor muito alto especificado para este não é satisfatório, sendo encontrado um equilíbrio na execução das consultas quando adotado a regra de *tuning, que* define o valor do parâmetro a menos de 5% da memória RAM disponível. A média tempo da execução das consultas diminui em 10.90 segundos em relação ao valor padrão do PostgreSQL.

No parâmetro *work_mem,* o melhor desempenho apresentados pelo SGBDR foi quando a regra de *tuning* definia o valor de 3MB para o parâmetro. Os valores muito altos para *work_mem* não podem apresentar beneficio se ultrapassarem as capacidades do hardware e também não podem ser menores de modo que o SGBDR não consiga executar as consultas. E a melhor configuração de *tuning* encontrada diminui a média de execução das consultas em 30.94 segundos em relação ao padrão definido pelo PostgreSQL.

O parâmetro *effective_cache_size,* obteve melhor desempenho quando a configuração de *tuning* definiu o valor de 2365MB, sendo 41% da memória RAM disponível. Observando que os valores altos para *effective_cache_size* favorecem o melhor desempenho na execução de cargas OLAP no SGBDR PostgreSQL. E valores muito baixos não favorecem a execução das consultas, aumentando assim o

tempo total na realização das transações conforme especifica a Figura 7. A regra de *tuning* com melhor desempenho executou suas consultas em média 6.04 segundos a menos que o desempenho apresentado no valor padrão definido no SGBDR.

Quando foi realizado a combinação dos parâmetros, obteve-se melhor desempenho no SGBDR PostgreSQL, na regra de *tuning* que definia valores para *shared_buffers* 24 MB, *work_mem* com 2 MB e *effective_cache_size* com 512MB . Está combinação diminui a média do tempo de execução das consultas em 85,36 segundos, melhorando o desempenho do SGBDR PostgreSQL.

Após analisar os experimentos demonstrados nos resultados podemos obter uma conclusão final para este trabalho, sendo descrita na sequência no Capítulo 6.

6 CONCLUSÃO E TRABALHOS FUTUROS

Atualmente os SGBDR são os mais populares e os mais utilizados para gerenciar base de dados, devido a desempenhar de modo eficiente tarefas como concorrência, recuperação e manipulação de grandes volumes de dados. Os SGBDR possuem um bom desempenho na realização das operações e transações no banco de dados, mais uma vantagem que podemos levantar para evidenciar sua posição de SGBD mais utilizado. Pode-se destacar a flexibilidade encontrada nos SGBDR pelo DBA's, para aumentar ainda mais o seu desempenho frente a diferentes cargas de trabalho, com a utilização da técnica de tuning.

O *tuning* realizado em SGBDR, é de extrema importância, pois através dele podemos melhorar o desempenho do SGBDR, configurando o mesmo para utilizar de forma mais eficiente os seus recursos e os oferecidos pelo hardware onde este se encontra. Uma técnica que exige conhecimentos sobre o SGBDR e também a carga de trabalho que está sendo aplicada no banco de dados, a fim de verificar qual parâmetro de configuração deve ser modificado para não piorar o desempenho do SGBDR.

Para alcançar um melhor desempenho do SGBDR utilizando o *tuning*, podemos simular as cargas de trabalho e monitorar o seu desempenho. A utilização deste método, possibilita alcançar aspectos mais específicos do desempenho do SGBDR frente a carga de trabalho utilizada. Entretanto exige um maior conhecimento de softwares específicos como os *benchmarks,* e demanda tempo excessivo para analisar cada *tuning* realizado, aumentando assim sua complexidade. Os experimentos executados para analisar o *tuning* no SGBDR podem levar dias dependendo do tamanho da base de dados, neste trabalho por exemplo foi definindo a base de dados com 10GB, cada experimento levou em média 15 horas para serem executados.

O *tuning* dos parâmetros de configuração do SGBDR, muitas vezes podem ser obter resultados negativos, que consequentemente diminuem o desempenho do SGBDR. Assim o *tuning* necessita de atenção redobrada para sua realização, pois um *tuning* mal feito prejudicaria o SGBDR, ao invés de melhorá-lo. A escolha do

parâmetro a ser modificado, deve ser obtida através do monitoramento do SGBDR, verificando seu tempo de resposta para as requisições a ele impostas e sua produtividade, definida pela quantidade de trabalho executada em um período de tempo pelo SGBDR.

Assim as afirmações anteriores demonstram a complexidade na realização do *tuning*, sendo ainda maior se analisarmos o número de parâmetros presentes em cada SGBDR e as combinações de valores que podem ser direcionadas a cada um destes. Considerando que o *tuning* é realizado manualmente nos SGBDR.

Com a realização do *tuning* neste trabalho foi possível detectar os parâmetros que apresentam a maior sensibilidade quando alterados para carga de trabalho do tipo OLAP. Demonstrando assim três parâmetros de configuração presentes no SGBDR PostgreSQL, que afetam o desempenho do mesmo frente a carga de trabalho estudada.

Um trabalho futuro a ser desenvolvido é considerar a carga de trabalho OLTP citada neste trabalho, onde pode-se pesquisar sobre os parâmetros de configuração nos SGBDR que podem ser sensíveis a tal carga de trabalho, buscando identificar qual *benchmark* que simula um ambiente OLTP, melhorando assim desempenho.

Podemos verificar, a migração de SGBDR para ambientes virtualizados, sendo uma tendência que apresenta inúmeras vantagens entre elas, a redução de custos diretos, bem como a melhor utilização da capacidade dos recursos existentes. Neste sentido evidencia como trabalho futuro, estudar como é realizado o *tuning* do SGBDR nos ambientes virtualizados, que deve considerar que o mesmo compete pelos recursos existentes nestes ambientes.

7 REFERÊNCIAS

ALMEIDA, Eduardo Cunha de. E**studo de Viabilidade de uma Plataforma de Baixo Custo para Data Warehouse,** 2004. Dissertação (Mestrado em Informática) – Programa de Pós-graduação em Informática da Universidade Federal do Paraná, 2004.

BINI, Tarcizio A. **Aplicação do Algoritmo de Kruskal na otimização de consultas com Múltiplas Junções Relacionais**, Capítulo 2, 2009. 97 f. Dissertação (Mestrado em Informática) – Programa de Pós-graduação em Informática da Universidade Federal do Paraná, 2009.

BRITO, Ricardo W**. Bancos de Dados NoSQL x SGBDs Relacionais: Análise Comparativa.** FFB, 2010. Disponível em:<http://www.infobrasil.inf.br/userfiles/27-05-S4-1-68840-Bancos%20de%20Dados%20NoSQL.pdf >. Acesso: 01/06/2012.

COOD, E. F. **A Relational Model of Data fo Large Shared Data Banks.** IBM Research Laboratory, San Jose, California, 1970.

Codd E.F., Codd S.B., and Salley C.T. "Providing OLAP (On-line Analytical Processing) to User-Analysts: An IT Mandate". Codd & Date, Inc 1993. < http://www.fpm.com/refer/codd.html >

COOD, E. F.. **Relational completeness of data base sublanguages**. In: R. Rustin (ed.): Database Systems: 65-98, Prentice Hall and IBM Research Report RJ 987, San Jose, California, 1972.

DATE, C. J. **Introdução a Sistemas de Bancos de Dados**, 1ª edição. Editora Campus, 2004.

DB-ENGINE. **DB-Engines Ranking**, 2013. Disponível em: <http://db-engines.com/en/ranking>. Acesso em: 13/11/2013.

DB2. **Manual DB2**. 2008. Disponível em: <http://www-01.ibm.com/support/docview.wss?uid=swg2700955> Acesso em: 01/11/2013.

DEBNATH, B. K., LILJA, D. J., and MOKBEL, M. F. (2008). **Sard: A statistical approach for ranking database tuning parameters**. In Proceedings of the 2008 IEEE 24th Inter-national Conference on Data Engineering Workshop, ICDEW ’08, Washington, DC, USA. IEEE Computer Society.

ELMASRI, Ramez; NAVATHE, Shamkant B. **Fundamentals of Database Systems, 4th edition**. Addison Wesley, 2004.

ELMASRI, Ramez; NAVATHE, Shamkant B. **Fundamentals of Database Systems, 6th edition**. Addison Wesley, 2011.

GRAY, Jim. **A Measure of Transaction Processing Power**, 2001**.** Disponível em < http://infolab.usc.edu/csci599/Fall2008/papers/c-2a.pdf >

GURGEL, Flavio. **Como Lidar com Cargas de Trabalhos Mistas.** PGBDR, 2013. Disponível em: <http://pgbr.postgresql.org.br/2013/slides/24.pdf>. Acesso em: 05/09/2013.

LIMA, Murilo R. de. **Execução Distribuídas de Benckmarks Em Sistemas de Banco de Dados Relacionais**. Dissertação (Mestrado em Informática) – Programa de Pós-graduação em Informática da Universidade Federal do Paraná, 2008.

MENDES, Marcelo R. N. **Análise de Desempenho de Sistemas OLTP Utilizando o Benchmark TPC-C**. UFPE, 2006. Disponível em: <http;//www.cin.ufpe.br/~tg/2006-1/mrnm.pdf > acesso: 02/06/2013.

MYSQL. **MySQL Reference Manual**. 2013. Disponível em: <http://dev.mysql.com/doc/refman/5.6/en/> Acesso em: 10/11/2013.

NASCIMENTO, RILSON. **DBT-5: Uma Implementação de Código Aberto do TPC-E para Avaliação de Desempenho de Sistemas de Processamento Transacional**. 2008, 115 f. Dissertação (Mestrado em Ciência da Computação) - Programa de Pós-graduação em Ciência da Computação do Centro de Informática da Universidade Federal de Pernambuco, 2009.

ORACLE. **Documentation Library.** 2013. Disponível em:< http://www.oracle.com/pls/db102/homepage> . Acesso em: 09/11/2013.

PESSOA, Fabio A. R. **Analise e aperfeiçoamento de benchmark OLPT para Avaliação de desempenho em SGBD.** Dissertação (mestrado) – Universidade Federal de Pernambuco. CIn. Ciência da Computação, 2006**.**

PGADMIN. **PGAdmin PostgreSQL Tools.** 2013. Disponível em: <https://www.pgadmin.org>. Acesso em: 20/11/2013.

POSTGRESQL. **Manual PostgreSQL**. 2012. Disponível em:

<http://www.postgresql.org/docs/9.2/static/> Acesso em: 01/06/2013.

POWELL, Gavin. **Beginning Database Design**. Indianápolis. Indiana: John Wiley & Sons, 2006.

RAMAKRISHNAN, R; GEHRKE, J. **Sistemas de Gerenciamento de Bancos de Dados**, 3ª edição. Editora Bookman, 2008.

Selinger P. G.;Astrahan M. M.; Chamberlin D. D.; Lorie D. D.; e Price T. G. **Access path selection in a relational database management system. SIGMOD '79: Proceedings of the 1979 ACM SIGMOD international conference on Management of data**, páginas 23-34, New York, NY, USA, 1979. ACM.

SILBERSCHATZ, Abraham; KORTH, Henry F.; SUDARSHAN, S. **Sistema de banco de dados**. Rio de Janeiro: Elsevier, 2006.

SMITH, Gregory. **PostgreSQL 9.0 High Performance .** Reino Unido, 2010. Editora Packt Publishing Ltd .

SOURCEFORGE. **Database Test Suite,** 2011. Disponível em:<http://sourceforge.net/apps/mediawiki/osdldbt/index.php?title=Main_Page>. Acesso em: 01/06/2013.

SQLPLUS. **SQLPlus Guia do Usuário e de Referência**, 2005. Disponível em: <http://docs.oracle.com/cd/B19306_1/server.102/b14357/ch3.htm>. Acesso em: 23/11/2013.

TPC. **Transaction Performance Council**. Disponível em:< http://www.tpc.org/ >. Acesso em: 20/03/2013.

TPC Benchmark A. **Manual TPC-A.** São Jose, 1994. Disponível em: <http://www.tpc.org/tpca/spec/tpca_current.pdf>. Acesso em: 20/03/2013

TPC Benchmark H. **Manual TPC-H**. San Francisco, 1997. p, 136. Disponível em:<http://www.tpc.org/tpch/spec/tpch2.15.0.pdf >. Acesso em: 20/03/2013