UNIVERSIDADE FEDERAL RURAL DE PERNAMBUCO UNIDADE ACADÊMICA DE EDUCAÇÃO A DISTÂNCIA E TECNOLOGIA BACHARELADO EM SISTEMAS DE INFORMAÇÃO

BACHARELADO EM SISTEMAS DE INFORMAÇÃO

**ALEXSANDRO MATIAS DE ALMEIDA**

**MEDIÇÃO DO GANHO DE DESEMPENHO APÓS OTIMIZAÇÕES NO BANCO DE DADOS MYSQL**

PALMARES, 2021

**ALEXSANDRO MATIAS DE ALMEIDA**

**MEDIÇÃO DO GANHO DE DESEMPENHO APÓS OTIMIZAÇÕES NO BANCO DE DADOS MYSQL**

Trabalho de Conclusão de Curso apresentado à Coordenação do Curso de Bacharelado em Sistemas de informação, pelo aluno ALEXSANDRO MATIAS DE ALMEIDA, sob orientação da professora Dra. Juliana Regueira Basto Diniz, para conclusão do Curso de Sistemas de Informação.

PALMARES, 2021

**AGRADECIMENTOS**

Agradeço, primeiramente, a Deus pela oportunidade dessa formação e por ter me dado saúde e força para superar as dificuldades.

A esta faculdade, seu corpo docente, direção, administração e funcionários que sempre me serviram da melhor maneira possível nos momentos que precisei.

A minha mãe, irmã e esposa, por todo amor, paciência, incentivo e apoio incondicional que me deram durante a minha jornada no curso.

E a todos que direta ou indiretamente fizeram parte da minha formação, o meu muito obrigado.

**RESUMO**

O presente estudo analisa o quanto de desempenho é ganho quando realizados diversos ajustes de configuração no Banco de Dados MySQL, processo esse chamado de tuning. Para realização deste teste de performance será utilizado um modelo internacional para benchmark chamado TPC-H que serve para medição de carga de trabalho auxiliando no suporte à decisão. Para isso, serão criadas duas bases dados: A primeira apenas seguindo o modelo de criação e população das tabelas indicado pelo TPC-H. Já no segundo serão realizadas as configurações possíveis para as consultas ao banco sejam as mais performáticas possíveis. A partir dos dados coletados, serão descritos os valores dos ganhos percentuais nas consultas a essa base de dados otimizada.

**Palavras-Chave:** MySQL, Desempenho, TPC-H.

**ABSTRACT**

The present study analyzes how much performance is gained when several configuration adjustments are made in the MySQL Database, a process called tuning. To perform this performance test, an international benchmark model called TPC-H will be used to measure workload, helping to support the decision. For that, two databases will be created: The first just following the model of creation and population of the tables indicated by TPC-H. In the second one, the possible configurations for queries to the bank will be made as well as possible. From the collected data, the values of the percentage gains in the consultations to this optimized database will be described

**Keywords:** Relacional Databases, Performance.

**LISTA DE SIGLAS**

BD – Banco de Dados.

DDL - Data Definition Language.

DML - Data Manipulation Language.

SGBD – Sistema de Gerenciamento de Banco de Dados.

SGBDR – Sistema de Gerenciamento de Banco de Dados Relacional.

SQL – Structured Query Language.

MER – Modelo Entidade Relacionamento

**SUMÁRIO**

[1. INTRODUÇÃO 7](#_Toc52307560)

[1.1 OBJETIVOS 7](#_Toc52307561)

[1.1.1 OBJETIVO GERAL 7](#_Toc52307562)

[1.1.2 OBJETIVOS ESPECÍFICOS 7](#_Toc52307563)

[1.2 JUSTIFICATIVA 7](#_Toc52307564)

[2. FUNDAMENTAÇÃO TEÓRICA 8](#_Toc52307565)

[2.1 DADOS 9](#_Toc52307566)

[2.2 INFORMAÇÃO 9](#_Toc52307567)

[2.3 BANCO DE DADOS 9](#_Toc52307568)

[2.4 SISTEMA DE GERENCIAMENTO DE BANCO DE DADOS 9](#_Toc52307569)

[2.4.1 BANCO DE DADOS MySQL 10](#_Toc52307570)

[2.5 COMANDOS SQL 11](#_Toc52307571)

[2.5.1 DDL DATA DEFINITION LANGUAGE 11](#_Toc52307572)

[2.5.2 DML DATA MANIPULATION LANGUAGE 11](#_Toc52307573)

[2.5.3 DCL DATA CONTROL LANGUAGE 11](#_Toc52307574)

[3. PERFORMANCE 12](#_Toc52307575)

[3.1 TESTE DE CARGA OU VOLUME 12](#_Toc52307576)

[3.2 TESTE DE CAPACIDADE 12](#_Toc52307577)

[4. PROCEDIMENTOS METODOLÓGICOS 12](#_Toc52307578)

[4.1 METODOLOGIA 12](#_Toc52307579)

[4.2 TIPO DE PESQUISA 13](#_Toc52307580)

[4.3 AMBIENTE DE TESTES 13](#_Toc52307581)

[4.4 COLETA DE DADOS 14](#_Toc52307582)

[5. ANÁLISE DOS DADOS DA PESQUISA (OBTIDOS) 14](#_Toc52307583)

[6. CONSIDERAÇÕES FINAIS 14](#_Toc52307584)

[7. REFERÊNCIAS 15](#_Toc52307585)

1. INTRODUÇÃO

A utilização dos bancos de dados relacionais é indispensável em sistemas de informação para aplicações no campo empresarial. Isso se torna visível quando cinco dos bancos dados mais utilizados no mercado nos últimos cinco anos, quatro deles são relacionais. Estes são representados em ordem crescente em Oracle, MySQL, Microsoft SQL Server, PostgreSQL e MongoDB (DB-Engines, 2020). Para este estudo será adotado Sistemas de Gerenciamento de Banco de Dados (SGBD) gratuitos, uma vez que existe a Cláusula DeWittde que proíbe pesquisadores e cientistas divulgarem explicitamente os nomes de seus sistemas em trabalhos acadêmicos (Brian Moran, 2003).

Então, já que os banco MySQL está entre os cinco mais utilizados no mercado (DB-Engines, 2020), se torna interessante conhecimento mais específico quanto à performance dos mesmo no que diz respeito ao tempo gasto nas operações de inserção, busca, alteração e exclusão de dados, uso de CPU e utilização de memória RAM ao aplicar uma carga de dados no sistema.

* 1. OBJETIVOS

Serão apresentados os objetivos que nortearam este Relatório do Estágio Supervisionado III.

* + 1. OBJETIVO GERAL

Analisar a performance de bancos de dados relacionais através de testes de carga utilizando o benchmark padrão TPC-H.

* + 1. OBJETIVOS ESPECÍFICOS
* Realizar criação e população das tabelas nos bancos de dados MySQL utilizando o modelo TPC-H;
* Realizar medição de consultas aos registros;
* Analisar os resultados obtidos;
  1. JUSTIFICATIVA

O aumento crescente da utilização dos bancos de dados é atribuído aos avanços nas tecnologias de sistemas de informação que consideram os dados das empresas como um bem intangível e extremamente valioso. Essas ferramentas têm auxiliado as empresas a guiar o seu planejamento estratégico. Assim, com o crescimento exponencial desses dados, a disponibilidade, integridade, confidencialidade e forma de armazenamento são imprescindíveis neste processo de competitividade empresarial. Outra aplicação direta do uso de banco de dados é o conceito de Big Data que vem se consolidando como a base para o desenvolvimento de novas tecnologias capazes de manipular grandes conjuntos de dados visando também promover subsídios para a tomada de decisão. Dito isto, algumas características importantes na implementação desses bancos de dados para o mercado:

* Performance aceitável na manipulação a grandes volumes de dados;
* Flexibilizar a atual estrutura baseada em modelo de dados e metadados utilizada pelos bancos relacionais;
* Escalabilidade do modelo do banco de dados implementado.

Este trabalho terá na seção 2 serão apresentados os conceitos básicos, e aplicação desses bancos de dados, assim como será explanado os comandos que viabilizam os testes. Na seção 3 será apresentada a metodologia e preparação do ambiente para o teste de carga, apresentando quais os softwares (com suas respectivas versões) e hardware foram utilizados, além de tratar de algumas métricas preestabelecidas juntamente com as configurações especificas dos bancos de dados estudados. Já na seção 4 serão mostrados os resultados obtidos no teste de carga, para que se possa concluir qual o ganho percentual dessas otimizações.

1. FUNDAMENTAÇÃO TEÓRICA

Nesta seção são abordados conceitos básicos necessários para a compreensão dos principais temas abordados nesta pesquisa. De uma forma geral serão explanados os conceitos fundamentais em torno de Sistemas de Gerenciamento de Banco de Dados, os comandos SQL e suas particularidades. Os conceitos em torno dos testes necessários para avaliação de desempenho dos dois Banco de Dados em estudo e testes estatísticos utilizados na análise dos resultados.

* 1. DADOS
  2. INFORMAÇÃO
  3. BANCO DE DADOS
  4. SISTEMA DE GERENCIAMENTO DE BANCO DE DADOS

Originalmente da sigla em inglês DBMS (Data Base Management System), ou Sistema de Gerenciamento de banco de dados (SGBD), é um pacote de software projetado cuja função é gerenciar uma base dados. As vantagens do seu uso são [fonte do livro]:

* Independência de Dados: Os programas aplicativos não devem, idealmente, ser expostos aos detalhes de representação e armazenamento de dados.
* Acesso Eficiente aos Dados: Um SGBD utiliza uma variedade de técnicas sofisticadas para armazenar e recuperar dados eficientemente.
* Integridade e Segurança dos Dados: Se os dados são sempre acessados através do SGBD, ele pode forçar restrições de integridade.
* Administração de Dados: Quando diversos usuários compartilham dados, centralizar a administração dos dados pode oferecer melhorias significativas.
* Acesso Concorrente e Recuperação de Falha: Um SGBD planeja o acesso concorrente aos dados de maneira tal que os usuários podem achar que os dados estão sendo acessados por apenas um único usuário de cada vez. Além disso, o SGBD protege os usuários dos efeitos de falhas de sistema.
* Tempo Reduzido de Desenvolvimento de Softwares: O SGBD suporta funções importantes que são comuns a várias linguagens de programação que acessam os dados no SGBD.

É importante ressaltar (Ramakrishnan & Gehrke, 2009) que algumas vezes, não se torna viável o uso de um SGBD. Isso se aplica já que se trata de um software complexo para executar determinadas aplicações, como por exemplo, responder a consultas complexas ou tratar várias requisições concorrentes. Por isso, seu desempenho pode não ser adequado para determinados aplicações especificas. Entretanto, na maioria das situações em que é necessário gerenciamento de dados em grande escala, os SGBDs têm se tornado uma ferramenta indispensável.

* 1. BANCO DE DADOS MySQL

O MySQL é um sistema de gerenciamento de banco de dados (SGBD), que utiliza a linguagem SQL (Linguagem de Consulta Estruturada, do inglês Structured Query Language) como interface. É atualmente um dos sistemas de gerenciamento de bancos de dados mais populares da Oracle Corporation, com mais de 10 milhões de instalações pelo mundo.

Possui as seguintes características:

* É um Software Livre com base na GPL;
* Alta portabilidade já que suporta praticamente qualquer plataforma atual;
* Boa Compatibilidade com linguagens de programação pois existe drivers ODBC, JDBC e .NET e módulos de interface para diversas linguagens de programação, como Delphi, Java, C/C++, C#, Visual Basic, Python, Perl, PHP, ASP e Ruby);
* Excelente desempenho e estabilidade;
* Pouco exigente quanto a recursos de novos hardwares;
* Facilidade no manuseio;
* Contempla a utilização de vários Storage Engines como MyISAM, InnoDB, Falcon, BDB, Archive, Federated, CSV, Solid;
* Suporta controle transacional;
* Suporta Triggers;
* Suporta Cursors (Non-Scrollable e Non-Updatable);
* Suporta Stored Procedures e Functions;
* Replicação facilmente configurável;
  + 1. ÍNDICES

Os índices são usados ​​para localizar linhas com valores de colunas específicas mais rapidamente. Sem um índice, o MySQL deve começar com a primeira linha e, em seguida, leia toda a tabela para encontrar o registro procurado (MySQL 8.0, 2020). Quanto maior a tabela, o mais isso custa. Se a tabela tiver um índice para as colunas em questão, o MySQL pode determinar rapidamente a posição para procurar no meio do arquivo de dados sem ter que olhar para todos os dados. Isso é muito mais rápido do que ler cada linha sequencialmente. A maioria dos índices utulizados no MySQL são (Primary Key, Unique, Index e Fulltext).

Paginah 1614

A documentação oficial da linguagem indica para que haja um melhor desempenho do banco, os parâmetros que devem ser observados são:

Para encontrar as linhas que correspondem a uma cláusula WHERE rapidamente.

Para eliminar linhas de consideração. Se houver uma escolha entre vários índices, o MySQL normalmente usa o índice que encontra o menor número de linhas (o índice mais seletivo).

Se a tabela tiver um índice de várias colunas, qualquer prefixo mais à esquerda do índice pode ser usado pelo otimizador

para procurar linhas. Por exemplo, se você tem um índice de três colunas em (col1, col2, col3), você tem

Otimização de chave primária

recursos de pesquisa indexados em (col1), (col1, col2) e (col1, col2, col3). Para mais

informações, consulte a Seção 8.3.6, “Índices de múltiplas colunas”.

Quando se deseja uma melhor performance para se encontrar linhas de outras tabelas ao realizar junções. MySQL pode usar índices em colunas mais de forma eficiente se forem declarados como do mesmo tipo e tamanho. Neste contexto, VARCHAR e CHAR são considerados iguais se forem declarados do mesmo tamanho. Por exemplo, VARCHAR (10) e CHAR (10) são do mesmo tamanho, mas VARCHAR (10) e CHAR (15) não são (MySQL 8.0, 2020).

Para comparações entre colunas de string não binárias, ambas as colunas devem usar o mesmo conjunto de caracteres. Por exemplo, comparar uma coluna utf8 com uma coluna latin1 impede o uso de um índice.

• Para encontrar o valor MIN () ou MAX () para uma coluna indexada específica key\_col. Isso é otimizado por um

pré-processador que verifica se você está usando WHERE key\_part\_N = constante em todas as partes-chave

que ocorrem antes de key\_col no índice. Neste caso, o MySQL faz uma única pesquisa de chave para cada MIN ()

ou MAX () expressão e a substitui por uma constante. Se todas as expressões forem substituídas por constantes, o

consulta retorna de uma vez. Por exemplo:

SELECT MIN (key\_part2), MAX (key\_part2)

FROM nome\_tabela WHERE parte\_chave1 = 10;

• Para classificar ou agrupar uma tabela se a classificação ou agrupamento for feito em um prefixo mais à esquerda de um índice utilizável (para exemplo, ORDER BY key\_part1, key\_part2). Se todas as partes-chave forem seguidas por DESC, a chave é lido na ordem inversa. (Ou, se o índice for um índice decrescente, a chave é lida em ordem progressiva.)

Consulte a Seção 8.2.1.16, “Otimização ORDER BY”, Seção 8.2.1.17, “Otimização GROUP BY”, e

Seção 8.3.13, “Índices descendentes”.

• Em alguns casos, uma consulta pode ser otimizada para recuperar valores sem consultar as linhas de dados. (Um índice que fornece todos os resultados necessários para uma consulta é chamado de índice de cobertura.) Se uma consulta usa de uma tabela apenas colunas que estão incluídas em algum índice, os valores selecionados podem ser recuperados da árvore de índice

para maior velocidade:

SELECT key\_part3 FROM tbl\_name

WHERE key\_part1 = 1

Os índices são menos importantes para consultas em tabelas pequenas ou grandes tabelas onde as consultas de relatório são mais processadas ou

todas as linhas. Quando uma consulta precisa acessar a maioria das linhas, ler sequencialmente é mais rápido do que trabalhar

por meio de um índice. As leituras sequenciais minimizam as buscas no disco, mesmo que nem todas as linhas sejam necessárias para a consulta.

Consulte a Seção 8.2.1.23, “Evitando varreduras de tabelas completas” para obter os detalhes.

* + 1. PARTICIONAMENTO DE TABELAS
  1. TPC-H
  2. COMANDOS SQL

Structured Query Language, ou em tradução livre, Linguagem Estruturada de Consulta é a linguagem usada nos SGBDs por padrão, no entanto, cada um tem suas particularidades dentro da própria linguagem, tendo implementações diferentes (Atanazio, 2019). A linguagem SQL tem algumas divisões, que facilitam o entendimento da mesma, categorizando seus comandos. Sendo que as mais conhecidas, que serão explicadas a seguir, são: DDL, DML e DCL.

* + 1. DDL DATA DEFINITION LANGUAGE

Linguagem de Definição de Dados, é a parte da Linguagem SQL que trata, como o próprio nome diz, da definição da estrutura dos dados, cujos efeitos se dão sobre objetos. Esses comandos são utilizados para a criação de bancos de dados, tabelas, views, triggers. Exemplos de comandos: CREATE (criação), ALTER (alteração), DROP (remoção), etc.

* + 1. DML DATA MANIPULATION LANGUAGE

Linguagem de Manipulação de Dados, é a parte da Linguagem SQL que não altera a estrutura e sim os registros de uma base de dados, cujos efeitos se dão sobre registros (Atanazio, 2019). São comandos que fazem consultas, inserem, alteram ou apagam registros. Exemplos de comandos: SELECT (consulta), INSERT (inserção), UPDATE (alteração), DELETE (remoção), etc.

* + 1. DCL DATA CONTROL LANGUAGE

Linguagem de Controle de Dados, é a parte da linguagem SQL referente ao controle de acesso a objetos por usuários e seus respectivos privilégios. Os principais comandos são:

* GRANT: Garante (permite) acesso dado a um usuário;
* REVOKE: Revoga (retira) direitos dados a um usuário. Os direitos dados a um usuário podem ser: ALL, CREATE, EXECUTE, REFERENCES, SELECT, TRIGGER, USAGE, CONNECT, DELETE, INSERT, RULE, TEMPORARY, UPDATE, etc.

1. PERFORMANCE

Em tecnologia, refere-se ao desempenho da aplicação, o tempo de resposta para executar uma operação de forma satisfatória, garantindo também que a aplicação seja escalável (IZAC, 2018).

* 1. TESTE DE CARGA OU VOLUME

É utilizado para identificar o limite da aplicação, qual o máximo de processamento que pode suportar trabalhando com cargas acima do normal, seja para quantidade de acessos simultâneos ou quantidade de operações\transações processadas com grande volume.

Para isso, antes dos testes começarem, define-se quais métricas serão coletadas e analisadas. A mais comum é o throughput, que é a taxa de transferência do processamento. Por exemplo, quantos registros foram processados por segundo, conceito usado bastante para redes, mas existe também o tempo de resposta, consumo de CPU e memória. É preciso conhecer essas métricas em condições normais para efeitos de comparação e análise das métricas quando o processamento validado for acima do normal.

* 1. TESTE DE CAPACIDADE

O teste de capacidade medirá as questões listadas nos testes de carga citado acima, porém considera um intervalo de tempo pré-determinado para essas medições.

1. PROCEDIMENTOS METODOLÓGICOS
   1. METODOLOGIA

A metodologia de elaboração deste trabalho está dividida nas seguintes etapas:

* Etapa 1: Elaboração do Modelo Entidade Relacionamento (MER);
* Etapa 2: Instalação e configuração dos bancos de dados;
* Etapa 3: Criação e população das tabelas;
* Etapa 4: Pesquisa de metodologias utilizadas em de trabalhos científicos relacionados ao estudo de performance de banco de dados.
* Etapa 5: Aplicados os testes.
* Etapa 6: Coleta e análise de resultados.

Para que seja viabilizado o estudo, deve-se tomar como início pelos seguintes parâmetros no ambiente virtualizado:

* Sistema operacional utilizado;
* Versão do banco utilizado;
* Qual mecanismo de criação de tabelas (engine) utilizado;
  1. TIPO DE PESQUISA
  2. AMBIENTE DE TESTES

Para os testes foi utilizado o sistema com as seguintes características:

* OS: Debian GNU/Linux 10 (Buster) x86\_64;
* Kernel: 4.19.0-10-amd64;
* Packages: 1426 (dpkg);
* Shell: bash 5.0.3;
* DE: Xfce4;
* CPU: AMD Ryzen 3 2200G (4) @ 3.500GHz;
* GPU: AMD ATI Radeon Vega Series / Radeon Vega Mobile Seri;
* Memory: 1795MiB / 15026MiB;
* MySQL: 8.0.21 for Linux on x86\_64 (MySQL Community Server - GPL).
  1. COLETA DE DADOS

1. ANÁLISE DOS DADOS DA PESQUISA (OBTIDOS)
2. CONSIDERAÇÕES FINAIS
3. REFERÊNCIAS

Ramakrishnan Raghu e Gehrke Johannes. Sistemas de Gerenciamento Sistemas de Gerenciamento de Banco de Dados [Livro]. - São PAULO: McGraw-Hill, 2009.

Schwartz Baron, Zaitsev Peter e Tkachenko Vadim. High Performance MySQL: Optimization, Backups, and Replication. 2ª. ed. Massachusetts: O’Reilly, 2012.

DB-Engines. DB-Engines Ranking. 2020. Disponível em: <https://db-engines.com/en/ranking>. Acesso em: 14 de Junho 2020.

Izac, Ariane. Introdução a Testes de Performance. 2018. Disponível em: < http://www.matera.com/blog/post/introducao-a-testes-de-performance>. Acessado em: 15 de Junho 2020

wikipedia. David DeWitt. 2020. Disponível em: <https://en.wikipedia.org/wiki/David\_DeWitt>. Acessado em: 15 de Junho de 2020.

Moran, Brian. The Devil's in the DeWitt Clause**.** 2003. Disponível em: https://www.itprotoday.com/sql-server/devils-dewitt-clause. Acessado em: 16 de Junho de 2020