FACULDADE DE TECNOLOGIA DE AMERICANA

Curso Superior Bacharelado em Análise de Sistemas e Tecnologia da Informação

Guilherme Laterza Pereira Lopes

Otimização de consultas SQL: Uma análise de Desempenho

Americana, SP

2013

FACULDADE DE TECNOLOGIA DE AMERICANA

Curso Superior de Bacharelado em Análise de Sistemas e Tecnologia da Informação

Guilherme Laterza Pereira Lopes

Otimização de consultas SQL: Uma análise de desempenho

Trabalho monográfico, desenvolvido em cumprimento à exigência curricular do Curso Superior de Bacharelado em Análise de Sistemas e Tecnologia da Informação da Fatec Americana, sob orientação do Prof. Me. José Alberto Rodrigues Filho.

Área de concentração: Banco de Dados.

Americana, SP

2013

Guilherme Laterza Pereira Lopes

Otimização de consultas SQL: Uma análise de desempenho

Trabalho de conclusão de curso apresentado à Faculdade de Tecnologia de Americana como parte dos requisitos para obtenção do título de Bacharel em Análise de Sistemas e Tecnologia da Informação.

Área de concentração: Banco de Dados.

Americana, 03 de Dezembro de 2013

Banca Examinadora:

_____
José Alberto Rodrigues Filho (Presidente)
Mestre
FATEC Americana

_____
Gabriel de Souza Fedel (Membro)
Mestre
FATEC Americana

_____
Rossano Pablo Pinto (Membro)
Mestre
FATEC Americana

AGRADECIMENTOS

Em primeiro lugar gostaria de agradecer ao corpo docente da FATEC Americana, pois foram eles que me proporcionaram a oportunidade de chegar até aqui, todos tiveram uma parcela de participação na minha formação acadêmica e humanitária.

Queria também agradecer a todos meus amigos que acreditaram em mim, me dando forças para perseguir meus objetivos sem me deixar abater pelos obstáculos que surgiram, em especial, Rafael, Samuel, Gabriel e Daiane.

E por fim, agradecer ao meu orientador, que me guiou durante a execução desta monografia, mostrando disponibilidade e vontade de ajudar.

DEDICATÓRIA

A minha família, que representa tudo que sou hoje e gostaria de ser um dia, além de serem a base da minha educação, são também modelos do que eu acredito ser mais justo e correto.

RESUMO

Atualmente a otimização de consultas SQL em um Banco de Dados tem sido motivo de discussão e análise por parte de especialistas em Banco de Dados, entretanto segue sendo um assunto pouco explorado pelos profissionais da área seja pela falta de experiência ou pela falta de conhecimento no assunto, que pouco é difundido e exigido no mercado de trabalho, dessa maneira muitas vezes punindo quem aplica essa metodologia e cobra a mais por ela. Cabe a este trabalho dissecar um pouco mais sobre a teoria deste assunto, entender o que é e como é possível aplicar essa otimização, mas sobretudo aplicar a teoria na prática, e assim descobrir através de resultados concretos, obtidos por um estudo de caso, qual é a verdadeira vantagem de aplicar o conceito de otimização de consultas SQL na construção de um Banco de Dados.

Palavras-chave: Qualidade. Desempenho. Teste; Banco de Dados.

ABSTRACT

Nowadays the SQL query tuning or optimization in a Database has been a reason for discussion and analysis among the Database specialists. However it remains an unexplored subject among the professionals from this field, due to lack of experience or even lack of knowledge in the matter, which is poorly spread and demanded by business, leading most of the time to impair those who apply this methodology and charges more for it. This paper has the responsibility to dissect a little more about the theory of this subject, understanding what it is and how is it possible to apply this tuning, but mainly to apply the theory in a practical experience and find out through concrete results, obtained by a case study, what is the true advantage of applying this concept of SQL query optimization in the process of building the Database.

Keywords: *Quality. Performance. Testing; Database.*

LISTA DE ILUSTRAÇÕES

LISTA DE TABELAS

LISTA DE ABREVIATURAS E SIGLAS

BD: Banco de Dados

DDL: *Data Definition Language* ou Linguagem de Definição de Dados

DML: *Data Modification Language* ou Linguagem de Modificação de Dados.

SGBD: Sistema Gerenciador de Banco de Dados

SQL: *Structured Query Language* ou Linguagem de Consulta Estruturada

XML: *eXtensible Markup Language*

SUMÁRIO

1 INTRODUÇÃO

Segundo Heuser (2008) define-se Banco de Dados como um conjunto de dados integrados que tem por objetivo atender a uma comunidade de usuários. Os Bancos de Dados (BD) atualmente estão presentes em praticamente todas as empresas de todos os ramos, portanto é uma peça fundamental para a funcionalidade de qualquer negócio.

Como peça fundamental, é necessário que seja dada sua devida importância na implementação de algum Sistema de Informação, passando da escolha de um Sistema Gerenciador de Banco de Dados (SGBD), pela construção de tabelas e relacionamentos até a criação de consultas de Linguagem de Consulta Estruturada ou *Structured Query Language (*SQL) que permite que usuários possam fazer alterações no BD, ou consultas específicas formadas por diversos filtros. Essas consultas são muitas vezes tratadas como meros caminhos sem importância para se chegar ao objetivo final que é o resultado da mesma, porém o que fica cada vez mais evidente é que dependendo do caminho a se seguir, podemos chegar ao objetivo de maneira mais rápida e sem imprevistos. Mesmo um aplicativo bem projetado poderá experimentar problemas de desempenho se a frase SQL que usa for mal construída. Falhas no projeto do aplicativo e na construção do SQL causam a maioria dos problemas de desempenho em bancos de dados projetados adequadamente (VOLACO, 2004).

Portanto se vê a importância de consultas SQL bem pensadas, e para esse propósito foi criada a otimização de consultas também chamado de "tuning", que trata de maneiras de deixar as consultas mais eficientes, obtendo assim resultados mais rápidos. Esta monografia trata justamente desse assunto, exemplificando com testes as mudanças ocorridas entre consultas construídas de diferentes maneiras, buscando assim demonstrar o real ganho dessa otimização e também as dificuldades para atingi-la.

No capítulo 2 uma revisão bibliográfica sobre a área de BD explica um pouco da história e dos conceitos chaves necessários para uma melhor compreensão do trabalho. No capítulo 3 a linguagem de consulta de dados é o foco, esse capítulo

abrange desde detalhes da linguagem SQL até os conceitos de otimização de consultas SQL, na forma automática e manual. O capítulo 4 traz um estudo de caso que evidencia os resultados da otimização de consultas SQL, através da comparação de performance entre consultas normais e consultas otimizadas. Por fim, o capítulo 5 traz a conclusão do trabalho, ou seja, uma análise de tudo que foi estudado visando descobrir os reais efeitos da otimização de consultas SQL.

2 REVISÃO BIBLIOGRÁFICA

Desde os primórdios da civilização, existe Banco de Dados, seja ele em forma de manuscritos, pastas, armários, prateleiras até o armazenamento virtual que é aplicado hoje em dia. Com foco no armazenamento virtual da informação, segundo Heuser (2008) do mesmo jeito que se usam gerenciadores de comunicação para comunicar-se através de processos remotos, usam-se SGBD para manter grandes armazenamentos de dados compartilhados, ou seja, para manter um banco de dados.

> O primeiro Sistema Gerenciador de Banco de Dados (SGBD) comercial surgiu no final de 1960 com base nos primitivos sistemas de arquivos disponíveis na época, os quais não controlavam o acesso concorrente por vários usuários ou processos. Os SGDBs evoluíram desses sistemas de arquivos de armazenamento em disco criando novas estruturas de dados com o objetivo de armazenar informações. Com o tempo, os SGBDs passaram a utilizar diferentes formas de representação, ou modelos de dados para descrever a estrutura das informações contidas em seus bancos de dados (TAKAI; ITALIANO; FERREIRA, 2005, p.6).

Se no final de 1960 surgiu o primeiro SGBD, foi só no começo de 1970 que surgiram os primeiros SGBDs relacionais, como o System R criado pela IBM e o INGRES criado como projeto de pesquisa na Universidade da Califórnia, Berkeley. Foram esses os precursores do modelo que é até hoje o mais usado no mercado de trabalho. Com o modelo relacional surgiu também a ideia e necessidade de se criar uma linguagem padrão para manipulação de BD relacionais, nascendo assim o SQL.

Apesar de já existir, o SGBD relacional não foi muito utilizado na década de 70, predominando os SGBDs hierárquicos e em rede, foi só na década de 80 que o System R incentivou a criação do SQL/DS, que no futuro se tornaria o DB2, e com ele vieram também o Oracle, DEC Rdb que teriam um papel fundamental no desenvolvimento desse modelo de SGBD.

Segundo Silberschatz, Korth e Sudarshan (2006) a partir desse momento os SGBDs relacionais se tornaram competitivos em relação aos anteriores, porém com

sua facilidade de uso, realizando as tarefas de baixo nível automaticamente, ele logo dominou o mercado, já que os programadores não mais teriam que dedicar tanto tempo à programação em baixo nível, podendo focar assim todas suas atenções no nível lógico, acelerando a implementação de grandes sistemas.

No final da década de 80 e início da década de 90 surgiu um novo conceito de SGBD, que fora empurrado pela crescente utilização da programação orientada a objeto, nascia aí o SGBD orientado a objeto.

A última grande inovação da área de BD surgiu no final da década de 90 e início da década de 2000, com a crescente utilização da internet, se tornou necessário que os sistemas de banco de dados tivessem disponibilidade 24/7 e comportassem um processamento de transação muito intenso, além é claro de aceitar interfaces da Web para dados. Apareceram nesse período o *eXtensible Markup Language* (XML) e a linguagem de consulta Xquery, despontando como uma nova tecnologia de Banco de dados que pode ser usada de forma nativa (depende do XML), ou como suporte a XML (mapeia o XML para BD relacional) (SILBERSCHATZ, KORTH e SUDARSHAN, 2006).

Antes de avançar mais no conceito de SGDB, é preciso entender exatamente o que é um Banco de Dados para poder identificar onde ele se encaixa na vida de cada um, e assim compreender o tamanho de sua importância para o mundo atual. *"*Um banco de dados é uma coleção de dados relacionados. Com dados, queremos dizer fatos conhecidos que podem ser registrados e possuem significado implícito.*"* (ELMASRI; NAVATHE, 2011, p.3). Apesar de ser uma definição genérica do conceito de BD, já nos permite ter um ponto de partido para melhor compreender a plenitude de um Banco de Dados.

Ainda segundo Elmasri, Navathe (2011, p.3) para especificar o significado de Banco de Dados no contexto deste trabalho, pode-se dizer que ele representa algum aspecto da realidade, que é um aglomerado de dados interligados de maneira lógica e coerente com algum significado e por fim é construído para ser populado por esse aglomerado de dados, dando assim uma aplicação para determinado grupo de usuários.

Para atestar essa definição, podemos tomar alguns exemplos do nosso dia a dia em que o BD está ativamente presente, mesmo sem percebermos. Um bom exemplo é um caixa eletrônico. Sem perceber nós estamos acessando um BD que contém nossos dados pessoais (conta, agência, nome, cpf, saldo) e a partir desses dados podemos efetuar um saque de dinheiro, que então atualizará o nosso registro, debitando o saque do saldo da conta. Comparando agora com a definição, percebe-se que esse BD representa uma realidade (no caso o registro de cada um de nós), que ele é uma coleção de dados com um significado (todas as informações da conta pessoal de uma pessoa, juntas, identificam um indivíduo e seu registro dentro do banco) e por fim ele possui uma aplicação para seus usuários (nesse exemplo ele serve para que possamos retirar dinheiro).

Com um melhor entendimento de BD é possível compreender o propósito de um SGBD, que é definido por Elmasri e Navathe, (2011, p.3,4) *"*O SGDB é um sistema de software de uso geral que facilita o processo de definição, construção, manipulação e compartilhamento de banco de dados entre diversos usuários e aplicações.", ou seja, é ele quem gerencia todas as informações existentes dentro de um BD, facilitando assim a interação desses dados com os usuários.

Existem muitas vantagens em se usar um SGBD, porém como esse não é o foco da pesquisa, segue apenas algumas das destacadas por Elmasri, Navathe (2011, p. 11 – 15), que são:

- Capacidade de controlar redundância: evita que sejam armazenados dados iguais em diferentes lugares, acabando com os dados redundantes, possíveis discrepâncias em informações buscadas em lugares diferentes e ainda diminuí o desperdício de esforço;

- Controle de acesso: evita o acesso de usuários não autorizados a determinadas informações;

- Backup e recuperação de dados: oferece recursos para recuperar os dados em casos de falha de software ou hardware, ou seja, além dele fazer o backup dos

dados atuais, ele consegue recuperar o sistema todo baseado nesses dados caso haja alguma falha;

- Múltiplas interfaces de usuário: oferece diferentes interfaces para diferentes tipos de usuários, com isso podem existir interfaces mais básicas para usuários que irão fazer simples consultas aos dados armazenados, ou então interfaces mais robustas e complexas para os programadores de aplicação poderem fazer alterações no sistema;

- Restrição de Integridade: existem vários tipos de restrições, do mais básico, que seria restringir um campo a aceitar apenas números inteiros (por exemplo, uma nota de prova, ou salário), a casos mais complexos que envolvem verificações programadas para avaliar diferentes circunstâncias, ou ainda, restrição para atualizações simultâneas, impedindo que diferentes usuários atualizem a mesma informação.

Existem ainda muitas outras vantagens específicas para cada tipo de SGBD adotado, esta pesquisa será conduzida com o modelo de dados relacional, portanto a seguir este modelo é visto de forma mais detalhada.

Segundo Silberschatz, Korth e Sudarshan (2006) o modelo de dados relacional é o mais usado comercialmente devido a sua simplicidade, que facilita o trabalho do programador, mas como conceituar um modelo de dados e mais especificamente o modelo de dados relacional?

“Um modelo de dados é uma coleção de ferramentas conceituais para descrever dados, relações de dados, semântica de dados e restrições de consistência [...]. O modelo relacional usa um conjunto de tabelas para representar tanto os dados quanto as relações entre eles.” (SILBERSCHATZ; KORTH; SUDARSHAN, 2006, p. 23).

Esse modelo tem como base um conjunto de tabelas para representar os dados, e através de duas linguagens específicas podemos atribuir relações entre essas tabelas, são elas: linguagem de manipulação de dados ou *Data Manipulation*

Language (DML) e linguagem de definição de dados ou *Data Definition Language* (DDL). Cada tabela é subdividida em colunas que são denominadas atributos e para cada conjunto de atributos é possível identificar um registro único, ou seja, tomando como exemplo uma tabela de clientes, ela será subdividida em vários atributos desse cliente (nome, CPF, data de nascimento, endereço, entre outros), que separados poderiam indicar diferentes informações, mas como um conjunto identificará um cliente específico.

Cada tabela necessita de um atributo (ou conjunto de atributos) que possa diferenciar cada um dos registros como algo único, no caso da tabela clientes, pode-se usar o CPF (pois não existem dois CPF iguais no Brasil) para diferenciar cada registro, ou então muitas vezes cria-se um atributo novo que será um número que se auto-incrementa a cada registro criado. Além disso, cada atributo terá sua própria restrição, como explicado anteriormente, no item “Restrição de Integridade”.

A linguagem de manipulação de dados permite que os usuários acessem e manipulem os dados contidos nas tabelas, podendo assim executar quatro diferentes tipos de ações: recuperação de informações contidas no BD, inserção de novas informações, exclusão de informações e modificação de informações existentes. Com isso é possível adicionar um novo cliente no sistema, excluir um cliente que faleceu, consultar dentro de um registro de cliente sua data de nascimento, ou ainda, modificar o endereço de um cliente que se mudou.

Já a linguagem de definição de dados constrói a estrutura de um BD e projeta suas funcionalidades, ou seja, cria as tabelas, os atributos, define as restrições de cada atributo, as autorizações dos usuários, entre outras funcionalidades. Existem diferentes linguagens que utilizam a DDL e DML de maneira conjunta (muitas vezes chamadas de linguagens de consulta), facilitando assim o uso dos usuários e programadores, estão entre elas: QBE, Datalog, PL/SQL e, a mais usada comercialmente, SQL.

3 LINGUAGEM DE CONSULTAS DE DADOS

Essa linguagem tem papel fundamental em um SGBD, pois é através dela que serão feitas a maioria das programações, definições, consultas e configurações de um Banco de Dados. Para a maioria dos usuários finais de um sistema de informação informatizado, a função mais utilizada em um Banco de Dados é a consulta de informações, sendo que na maioria das vezes o usuário nem sabe de que forma essa consulta está sendo feita, ou ainda de que uma consulta ao BD está ocorrendo. O que é então uma consulta e quando ela é utilizada pelos usuários?

Uma consulta é a busca por informações contidas dentro de um Banco de Dados. Um exemplo, toda vez que é feito uma busca por um produto em um site de vendas online, uma consulta do extrato bancário ou uma verificação de estoque, existe uma consulta sendo formulada para retornar o valor exato que o usuário busca, e dependendo do modo que essa consulta é formulada e executada, o sistema pode demorar mais ou menos para encontrar a informação mais pertinente.

Como já dito, a linguagem mais usada comercialmente pelo modelo relacional é a SQL, portanto é necessário apresentá-la de forma mais detalhada, para entender melhor o seu funcionamento.

3.1. Linguagem SQL

A SQL trabalha com os termos tabela, linha e coluna, se adequando assim ao modelo relacional. Na Tabela 1 fica mais fácil compreender o significado de cada um desses termos:

Tabela 1 - Tabela de BD chamada Cliente

Nome	CPF (c)	Endereco	Cidade	Bairro	Telefone
Guilherme Lopes	395076764-43	R. José Paiva, 665	Campinas	Pq. Taquaral	99898-4354
Daiane Rossi	296745908-33	R. Oito, 43	Campinas	Jd. Oliveira	99934-3243
Rafael Barbosa	295908123-55	Av. São Luis, 888	Americana	Botafogo	98843-0974
Samuel Vieira	343012945-44	R. Otávio Magalhães	Americana	Centro	99016-3399

Fonte: Tabela de autoria própria

A tabela em questão está armazenando dados dos clientes de uma empresa, portanto temos o nome da tabela “Cliente”, algumas colunas que identificam as informações que estarão contidas nessa tabela, e as linhas que contém os registros para cada cliente.

Outro fator importante é a identificação da chave primária, que nada mais é que o atributo único da tabela, que impede que dois registros possam ser gravados para uma mesma pessoa. Nesse caso a coluna CPF está marcada como chave primária, identificado pelo “(c)” à frente de CPF, já que todo cidadão brasileiro tem um CPF único.

A seguir segue uma demonstração de como fica a sintaxe da criação dessa tabela em linguagem SQL:

```
CREATE TABLE Cliente
        (Nome varchar(20),
        CPF varchar(12),
        Endereco varchar (25),
        Cidade varchar(15),
        Bairro varchar(15),
        Telefone varchar (12),
        PRIMARY KEY (CPF)
        )
```

A definição “varchar” acompanhada de um número entre parênteses se refere ao tipo de dado que será preenchido nesse campo, nesse caso todos os campos aceitarão valores do tipo texto, para permitir tanto números como letras, pontos e traços.

Após as tabelas serem criadas e populadas com as informações necessárias, é preciso existir uma forma de pesquisar e resgatar essas informações, nesse contexto que temos a consulta, que em SQL é feita com o comando SELECT, como visto a seguir:

```
SELECT CPF
FROM Cliente
 WHERE Nome='Guilherme Lopes'
```

Essa instrução trará o CPF de todos os clientes que tiverem o nome de Guilherme Lopes, nesse exemplo foi usada uma consulta simples, porém elas podem ser bem mais complexas e quanto mais complexas, mais tempo demoram a ser executadas. Existem outros fatores que influenciam no tempo de execução de uma consulta, como a quantidade de registros presentes na tabela a ser consultada, a velocidade do computador que está executando a consulta e até a maneira que a consulta é codificada, esse último fator que será focado no próximo tópico.

3.2. Otimização de consultas SQL

Existem diferentes sintaxes para se construir uma consulta que traga os mesmos resultados, cada uma dessas sintaxes pode ser mais demorada ou consumir maior processamento dependendo da situação em que ela é usada, por isso é necessário identificar qual é a forma com melhor custo benefício para cada situação, essa avaliação em busca de uma consulta é chamada de otimização de consulta.

"Otimização da consulta é o processo de selecionar o plano de avaliação de consulta mais eficiente dentre as muitas estratégias normalmente possíveis para o processamento de determinada consulta, especialmente se esta for complexa." *(*SILBERSCHATZ, KORTH e SUDARSHAN, 2006, p.383).

O princípio da otimização é analisar as consultas de forma algébrica relacional, portanto não é uma tarefa simples, fazendo com que a grande maioria da otimização seja feita pelo próprio SGBD, que transformará a consulta de linguagem SQL em uma expressão algébrica e assim poderá encontrar uma nova expressão

equivalente que tenha uma execução mais eficiente (SILBERSCHATZ, KORTH e SUDARSHAN, 2006).

Para entender melhor esse conceito, pode-se ver abaixo um exemplo de uma consulta SQL transformada em expressão algébrica:

```
SELECT Unome, Pnome
FROM Funcionario
WHERE Salario > (SELECT MAX(Salario)
                 FROM Funcionario
                 WHERE Dnr=5)
```

Esta consulta retornará o nome dos funcionários da empresa que ganham mais que o maior salário do departamento 5. Agora transformando isso em uma expressão de álgebra relacional, temos duas expressões:

$$\mathcal{F}_{MAX\ Salario}\ (\sigma_{Dnr=5}\ (FUNCIONARIO)) = c$$

$$\pi_{Unome,Pnome}\ (\sigma_{Salario>c}\ (FUNCIONARIO))$$

A primeira representa o SELECT interno, atribuindo c para seu resultado, que será o maior salário do departamento 5.Já a segunda expressão representa o SELECT externo, que retornará o resultado esperado dessa consulta SQL, que é o nome dos funcionários que tem um salário maior que c.

Agora que a consulta SQL foi traduzida para uma expressão de álgebra relacional, o otimizador irá escolher um plano de execução com o melhor desempenho (ELMASRI; NAVATHE, 2011).

Apesar da maior parte da otimização ser feita pelo próprio SGBD, existem maneiras do programador ajudar para que a otimização ocorra de maneira correta e sem maiores problemas. Essas maneiras podem variar desde o início quando o BD está sendo construído, através de configurações específicas, até no momento final quando as consultas são criadas.

3.3. Otimização manual

Apesar do SGBD possuir uma otimização automática, é necessário colaborar com uma ajuda manual para que essa otimização seja efetiva, já que a construção de algumas consultas e a configuração do BD em si podem acabar impedindo que a otimização ocorra. Além disso, é importante destacar que existem dicas e melhores práticas para serem aplicadas na hora da criação de consultas, antes mesmo delas serem otimizadas, fazendo com que o ganho de desempenho seja maior ainda.

Tendo em vista a visão dos autores Volaco (2006) e Couto (2006), primeiro será detalhado as modificações nas configurações do BD seguido pelas consultas, dessa forma fica mais fácil compreender a importância do planejamento de um BD em todas as suas etapas, do início ao fim.

É comum observar que muitos Bancos de Dados são configurados de maneira parecida, que costuma ser o padrão fornecido pelo SGBD, isso demonstra um planejamento pouco minucioso por parte dos programadores, pois cada BD deve ser configurado de acordo com a necessidade do cliente e de seu negócio, portanto o primeiro passo para atingir um desempenho satisfatório de um BD é planeja-lo antes de construí-lo. Abaixo estão alguns tópicos que são importantes de se levar em consideração no planejamento:

- Modelagem do BD: Independente da técnica usada para a modelagem, é necessário fazê-la com muita atenção, e preocupar-se em especial com a normalização das tabelas.

- Buffer do BD: Armazenando objetos muito utilizados na memória, se obtém uma maior velocidade de acesso.

- Ambiente físico: Tão importante quanto o BD é onde ele está hospedado, portanto manter sempre configurações de ambiente adequadas para o uso do BD (espaço em disco, processador, memória, velocidade do HD).

- Tipo de dado das colunas: Usar o tipo de dado que melhor se encaixa à necessidade do cliente, exemplificando, evitar o uso de varchar para campos que só utilizam números, saber quando é possível usar smallint em vez de int e dimensionar o tamanho do varchar com sua real necessidade.

- Análise de negócio: Procure descobrir mais sobre o negócio do cliente, assim será mais fácil avaliar quantos registros serão cadastrados por tabela, quais consultas serão mais utilizadas, entre outras informações que irão auxiliar no dimensionamento do BD.

- Indexação: A utilização de colunas com índices faz com que o SGBD busque os dados através de uma pesquisa indexada (pesquisa que utiliza um índice sobre a coluna definida para reduzir o tempo de resposta) ao invés de uma pesquisa completa (pesquisa que percorre linha a linha todos os registros de uma tabela).

Após os ajustes anteriores e do BD estar construído, é hora de voltar o foco às consultas e como podemos tomar certos cuidados na hora de escrever sua sintaxe para que o SGBD consiga utilizar da melhor maneira seu otimizador.

- Utilizar campos indexados: De nada adianta implementar a indexação nas colunas, se durante a construção da sintaxe de uma consulta não for usado as colunas indexadas para realizar a pesquisa. Ex:

```
SELECT Endereco, Cidade, Bairro
FROM Cliente
WHERE Nome=’Daiane’
```

Quando criamos uma coluna como chave primária, ela é automaticamente criada com um índice, portanto caso uma tabela não tenha outras colunas com índice, procure sempre que possível utilizar a chave primária para realizar consultas (nesse caso CPF é a chave primária, portanto podemos buscar um cliente através de seu CPF ao invés de seu nome).

```
SELECT Endereco, Cidade, Bairro
FROM Cliente
WHERE CPF='296745908-33'
```

- Evitar o parâmetro "*": Muitas vezes consultas são construídas para retornar todos os campos de uma tabela, quando na verdade só seria necessário retornar alguns atributos da tabela. Como exemplo, o objetivo das duas consultas abaixo é descobrir o número de telefone dos clientes de Campinas para verificar se o digito 9 já foi adicionado em seu cadastro:

```
SELECT *
FROM Cliente
WHERE Cidade='Campinas'
```

Como o único objetivo dessa consulta é verificar o número do telefone, só é necessário retornar o próprio número:

```
SELECT Telefone
FROM Cliente
WHERE Cidade='Campinas'
```

- Conversão de dados: Converter dados de uma coluna, por exemplo, utilizar um valor int (número inteiro) entre aspas como um varchar, fará com que o índice dessa coluna não seja usado, portanto inutilizando sua função de agilizar a realização da consulta. Ex:

```
SELECT Nome
FROM Funcionario
WHERE Salario='1500'
```

Salário nesse caso é um int, portanto não é necessário usar aspas para comparar dois números.

```
SELECT Nome
FROM Funcionario
WHERE Salario=1500
```

- Operadores aritméticos: Modificar dados de uma coluna com operações aritméticas anulará o uso do índice na realização da consulta.

```
SELECT Nome
FROM Funcionario
WHERE Salario*12>15000
```

Para descobrir se um salário anual atingiu uma determinada meta, é possível também dividir essa meta por 12 ao invés de multiplicar o salário por 12, desse jeito não modificamos no dado da coluna Salario:

```
SELECT Nome
FROM Funcionario
WHERE Salario>15000/12
```

- Padronizar os comandos SQL: Uma dica fácil e útil é manter sempre o mesmo padrão de sintaxe na construção das consultas. Ex:

```
Select CPF FROM Cliente
WHERE Nome='Guilherme'
```

Durante todos os capítulos foi utilizado um único padrão de sintaxe SQL, enquanto a sintaxe acima não está com o "Select" em caixa alta e o "FROM" está na primeira linha.

```
SELECT CPF
FROM Cliente
WHERE Nome='Guilherme'
```

Agora que ficou mais claro como manualmente aumentar o desempenho das consultas do BD, seja nas primeiras etapas com o planejamento do BD ou nas últimas com as mudanças na sintaxe das consultas, é necessário entender melhor como funciona o otimizador automático do SGBD.

3.4. Otimizador automático

Entender completamente o funcionamento do otimizador automático é uma tarefa difícil por envolver conceitos complexos, portanto este tópico será tratado apenas de forma superficial para atingir o objetivo de entender o que é o otimizador.

Conforme foi dito na seção 3.2 o otimizador traduz as consultas para a álgebra relacional e somente depois disso que ele trabalhará para otimizar essas consultas,

o que não foi dito ainda é de que maneira o otimizador faz essa otimização e com qual propósito.

De acordo com Elmasri, Navathe (2011), para realizar a otimização, o SGBD utiliza algoritmos específicos que tratam as diferentes partes de uma consulta, e apesar de cada um ter uma particularidade nesse processo, todos utilizam algoritmos específicos para cada parte, que são:

- Algoritmos para ordenação externa: Este algoritmo trata da intercalação da consulta, que é usado quando a cláusula ORDER BY está presente na consulta.

- Algoritmos para operação de seleção: Este algoritmo trata da operação de pesquisa para localizar certos registros, que é a função da cláusula SELECT.

- Algoritmo para operação de junção: Este algoritmo trata da junção de informações de diferentes tabelas em um único resultado, e é possível ver esse comportamento na utilização da cláusula JOIN ou NATURAL JOIN.

- Algoritmo para operações de projeção e conjunto: Este algoritmo é responsável por filtrar as colunas de uma tabela, eliminando assim as duplicatas. A única cláusula SQL que utiliza essa operação é o DISTINCT.

- Algoritmo para operação de agregação: Este algoritmo é responsável pelos operadores de agregação que no SQL são (MIN, MAX, COUNT, AVERAGE, SUM).

- Algoritmo para operação de junção externa: Diferente do algoritmo para junção,ele trata somente as junções externas, ou seja, LEFT OUTER JOIN, RIGHT OUTER JOIN e FULL OUTER JOIN.

Após demonstrar de que maneira as otimizações são feitas, é necessário também entender um pouco do propósito dessas otimizações, ou seja, o que o SGBD busca. Existem duas principais estratégias de otimização, são elas:

- Otimização heurística: Tem como estratégia transformar a árvore de consulta inicial (primeira tradução da consulta SQL para expressão da álgebra relacional) na final otimizada equivalente, através de diferentes regras que garantem a equivalências das expressões.

- Otimização baseada em custo: Tem como estratégia buscar a solução com a estimativa de custo mais baixa. Para calcular a estimativa de custo de uma consulta, são levados em conta os seguintes custos:

    - Custo de acesso ao armazenamento secundário
    - Custo de armazenamento em disco
    - Custo de computação
    - Custo de uso da memória
    - Custo de comunicação

4 EXEMPLO DE APLICAÇÃO

Este capítulo apresenta um estudo de caso para comparar os resultados obtidos de diferentes consultas realizadas em um mesmo Banco de Dados, a fim de compreender a real diferença entre consultas otimizadas, com as técnicas expostas anteriormente, e consultas não otimizadas. O objetivo desse estudo de caso é descobrir se há uma diferença de desempenho no BD aplicando técnicas de otimização de consultas SQL, e então mensurar quão significativa é essa diferença para sistemas de pequeno, médio e grande porte.

Foi escolhido o MySQL para conduzir este estudo de caso. Essa escolha foi feita devido ao fato do software ser bastante difundido no mercado de trabalho e existir versões grátis, desse jeito procurando simular da maneira mais verossímil, um cenário comum no ambiente de trabalho.

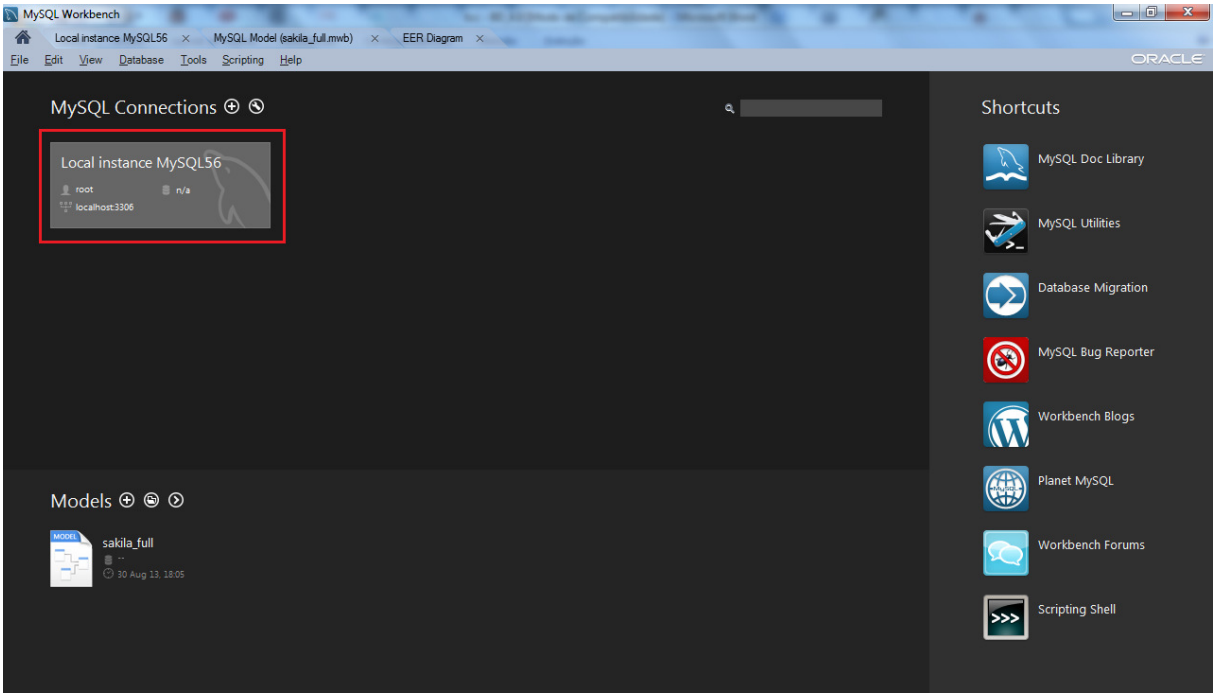
O segundo passo foi definir a estrutura do BD. Como o foco não é a construção das tabelas e sim a construção das consultas, será utilizado uma estrutura de BD exemplo que já vem configurada no MySQL. Com um planejamento traçado foi necessário reproduzir o ambiente em que este BD está hospedado, será detalhado a seguir a instalação do MySQL em uma máquina local e a construção das tabelas do BD.

4.1. Construção do Banco de Dados

Para esse estudo de caso usaremos a versão de desenvolvimento, que não tem custo e está disponibilizada no próprio site do software com o nome de MySQL Community Server 5.6.14: http://dev.mysql.com/downloads/mysql/.

Com o SGBD MySQL instalado com sucesso, o próximo passo é estabelecer uma conexão com a instância MySQL56 criada, para isso basta abrir o MySQL Workbench 6.0 CE e clicar em “Local Instance” como indicado na Figura 1.

Figura 1 - Tela de conexão do BD

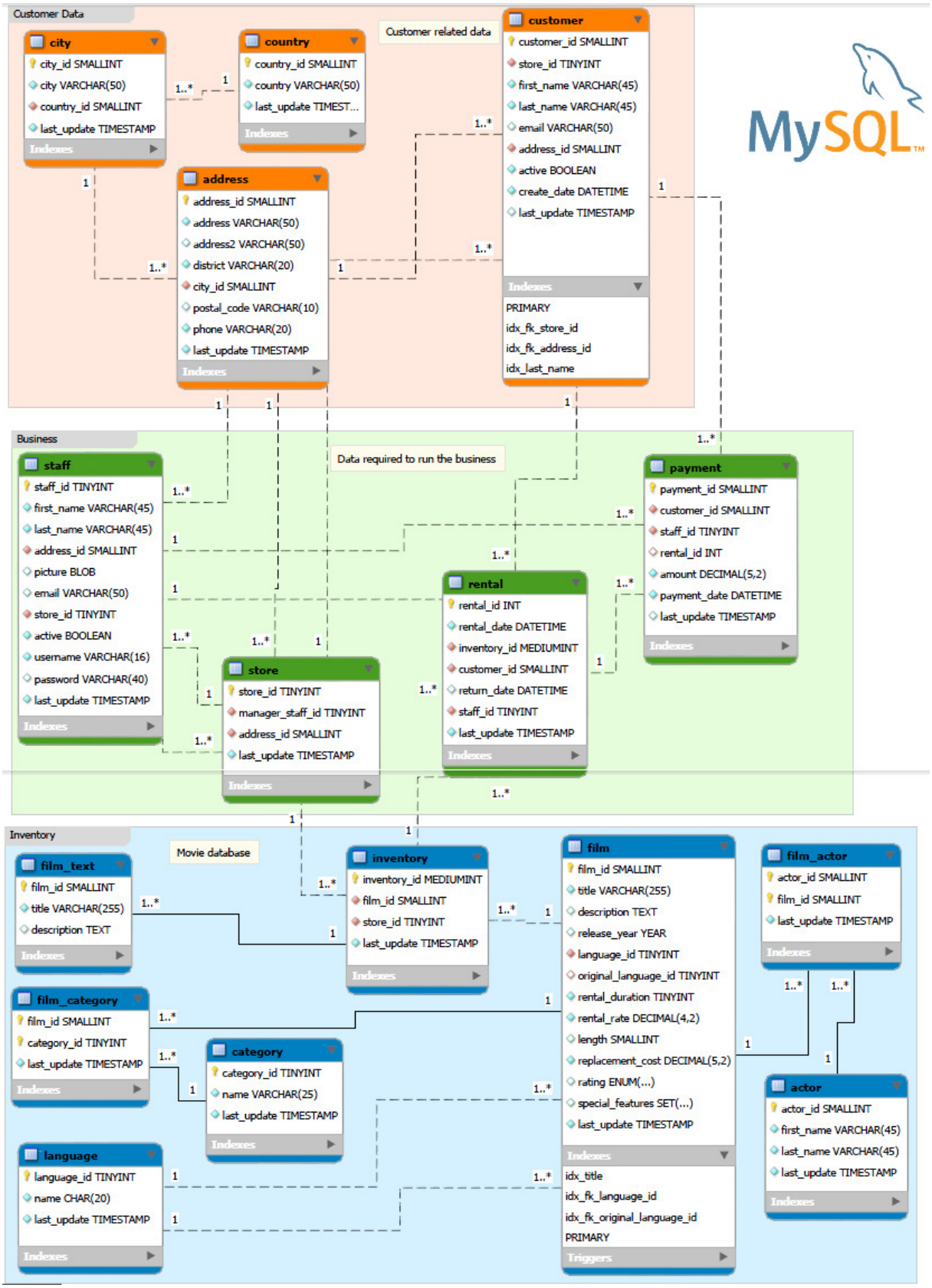


Fonte: Figura de autoria própria

Com o ambiente configurado, nota-se que já existe um BD completo construído com o nome de “sakila”, ele é uma simulação de um BD para uma empresa locadora de filmes.

Através da figura 2, é possível notar que as tabelas estão divididas em três grupos, dados relacionados ao cliente, dados de filmes e informação requerida para funcionamento do negócio, e os traços que ligam uma tabela a outra representam a relação existente entre elas, sendo que o “n” está representado pelo valor “1..*”. Outro aspecto importante são os símbolos a frente de cada linha, neste diagrama a chave primária está representada pelo símbolo amarelo, a chave estrangeira está representada pelo símbolo vermelho, um campo que não aceita valor nulo (NOT NULL) está representada pelo símbolo azul claro e por fim um campo que tem como valor padrão o vazio (NULL) está representado pelo símbolo branco.

Figura 2 - Modelo relacional do BD



Fonte: MySQL Workbench 6.0 CE

Apesar do BD estar finalizado, ainda falta um passo crucial para seguir adiante nos testes, que é popular as tabelas com registros suficientes para simular uma verdadeira empresa. Foram feitos três etapas de inserção de registros, cada uma delas representa o tamanho do negócio a ser simulado, primeiro foi inserido seiscentos clientes para simular uma microempresa, depois dez mil e seiscentos clientes para simular uma empresa pequena e por fim cem mil e seiscentos clientes para simular uma empresa média, em cada uma dessas etapas foram feitos testes para comparar as diferenças de desempenho, obtendo assim uma conclusão mais específica para cada nicho.

Para facilitar o preenchimento de todos os registros de clientes de forma que os dados fossem diferentes, foi usado um programa gerador de informação hospedado no site http://www.generatedata.com/, com ele foi possível gerar os sessenta mil registros sem maiores esforços em questões de alguns minutos.

4.2. Análise e interpretação dos dados

Duas particularidades foram levadas em conta para fazer as comparações entre os diferentes tipos de consultas, a primeira é o tamanho do BD, que foi separado em três testes, com seiscentos clientes registrados, dez mil e seiscentos e por fim cem mil e seiscentos clientes. A segunda é a técnica utilizada para manualmente otimizar a consulta sem alterar o propósito da mesma, portanto foram comparados consultas feitas no mesmo tamanho de BD e que apesar de terem uma sintaxe diferente, ambas tragam o resultado esperado pelo usuário.

Cada resultado foi obtido através de dez testes, e foi utilizado o valor que mais apareceu nesses dez testes (em média oito a cada dez testes davam resultados iguais), para evitar assim qualquer oscilação de uso de memória e cpu do sistema no momento da consulta. Além disso, foi necessário parar o SGBD e depois inicia-lo novamente a cada novo teste, evitando assim que dados fossem salvos na memória e depois reutilizados de maneira mais rápida. Outra medida tomada foi realizar os testes no mesmo computador e sem nenhum outro programa rodando, pois para as comparações terem credibilidade, é necessário que as mesmas condições sejam aplicadas para todos os testes.

Após demonstrar os resultados obtidos de todas as consultas, foi feito uma análise comparada com uma projeção para grandes empresas que tem BD com centenas de milhares ou até milhões de registros, desse jeito obtendo um embasamento suficiente para tirar melhores conclusões.

4.3. Resultado de consultas não otimizadas

Cada consulta traz uma explicação prévia do objetivo do usuário, para garantir que o resultado informado seja o resultado procurado, além disso, foram utilizadas consultas com sintaxes que não foram modificadas através dos conceitos de otimização.

Os resultados foram divididos em três tabelas, em ordem crescente de tamanho do BD, com duas colunas representando respectivamente as consultas elaboradas para cada questão imposta e o tempo de execução das consultas / linhas retornadas.

1) Quais são os nomes dos clientes que estão cadastrados na loja de código 1, que está localizada no endereço 47 MySakila Drive?

```
select c.first_name, c.last_name
from customer as c, address as a, store as s
where c.store_id=s.store_id and s.address_id=a.address_id and a.address=”47 MySakila Drive”
```

2) Qual é o nome do cliente com customer_id 430?

```
select first_name last_name
from customer
where customer_id='430'
```

3) Liste o nome dos clientes que começam com a letra “C”.

```
select *
from customer
where SUBSTR(first_name,1,1) = "C"
```

4) Descobrir quais clientes gastarão 1200 reais, caso gastem todo ano o total já gasto até hj, pelos próximos 12 anos.

```
select first_name, last_name
from
(select first_name, last_name, SUM(p.amount) as total_gasto
from payment as p, customer as c
where c.customer_id=p.customer_id
group by c.customer_id) as soma
where total_gasto*12>1200
```

5) Quais clientes moram nos Estados Unidos (United States)?

```
SelecT  first_name,  last_name
FROM customer as c,
country as y,
city as t, address as a

where  c.address_id=a.address_id  AND  a.city_id=t.city_id  aNd
t.country_id=y.country_id and country="United States"
```

A tabela 2 demonstra as consultas realizadas no BD de 600 registros:

Tabela 2 - Consultas normais BD micro

Consultas	Tempo / Linhas retornadas
Consulta 1	0.171 seg./326 rows
Consulta 2	0.016 seg./1 row
Consulta 3	0.000 seg./33 row
Consulta 4	0.109 seg./395 rows
Consulta 5	0.078 seg./36 rows

Fonte: Tabela de autoria própria

A tabela 3 demonstra as consultas realizadas no BD de 10.600 registros:

Tabela 3 - Consultas normais BD pequeno

Consultas	Tempo / Linhas retornadas
Consulta 1	0.225 seg./5255 rows
Consulta 2	0.047 seg./1 row
Consulta 3	0.062 seg./957 rows
Consulta 4	0.312 seg./395 rows
Consulta 5	0.140 seg./783 rows

Fonte: Tabela de autoria própria

A tabela 4 demonstra as consultas realizadas no BD de 100.600 registros:

Tabela 4 - Consultas normais BD médio

Consultas	Tempo / Linhas retornadas
Consulta 1	0.307 seg./50300 rows
Consulta 2	0.078 seg./1 row
Consulta 3	0.094 seg./9052 rows
Consulta 4	0.266 seg./395 rows
Consulta 5	0.764 seg./7241 rows

Fonte: Tabela de autoria própria

4.4. Resultados de consultas otimizadas

Seguindo os mesmos moldes das consultas não otimizadas, os resultados foram divididos em três tabelas, em ordem crescente de tamanho do BD, com duas colunas representando respectivamente as consultas elaboradas para cada questão imposta e o tempo de execução das consultas / linhas retornadas. Para facilitar a disposição da tabela, a seguir estão as questão impostas e a sintaxe das consultas otimizadas, cada uma elaborada de maneira que teste um método de otimização manual.

Na primeira questão foram utilizados os índices naturais das tabelas (chave primária e chave estrangeira) para a cláusula de condição “where”, em vez de utilizar campos não indexados:

1) Quais são os nomes dos clientes que estão cadastrados na loja de código 1, que está localizada no endereço 47 MySakila Drive?

```
select first_name, last_name
from customer
where store_id=1
```

Na segunda questão foi evitada a conversão de dados para a cláusula de condição “where”, portanto como o campo “customer_id” é do tipo inteiro, não será

usado o número 430 entre aspas, evitando que o campo tenha que ser convertido de string para inteiro:

2) Qual é o nome do cliente com customer_id 430?

```
select first_name last_name
from customer
where customer_id=430
```

Na Terceira questão foram apresentada três alternativas de otimização, a primeira prioriza a utilização do “LIKE” para evitar funções dentro da consulta, a segunda retorna somente os valores necessários para o cliente visando abolir o uso do “*”, por fim a terceira utiliza a indexação do campo “first_name” antes da consulta.

3) Liste o nome dos clientes que começam com a letra “C”.

(a)
```
Select *
From customer
Where first)name LIKE “C%”
```

(b)
```
Select first_name, last_name
From customer
Where first_name LIKE “C%”
```

(c)
```
create index idx_first_name on customer(first_name);

Select first_name, last_name
From customer
Where first_name LIKE “C%”
```

Na quarta questão foi modificada a última cláusula de condição “where” para evitar que o campo da coluna salário seja modificado por uma operação aritmética:

4) Descobrir quais clientes gastarão mais que 1200 reais, caso gastem todo ano o total já gasto até hoje, pelos próximos 12 anos.

```
select first_name, last_name
from
(select first_name, last_name, SUM(p.amount) as total_gasto
from payment as p, customer as c
where c.customer_id=p.customer_id
group by c.customer_id) as soma
```

```
where total_gasto>(1200/12)
```

Por fim, na última questão foi testada a padronização da sintaxe da consulta:

5) Quais clientes moram nos Estados Unidos (United States)?

```
select first_name, last_name
from customer as c, country as y, city as t, address as a
where c.address_id=a.address_id and a.city_id=t.city_id and t.country_id=y.country_id
and country="United States"
```

A tabela 5 demonstra as consultas otimizadas realizadas no BD de 600 registros:

Tabela 5 - Consultas otimizadas BD micro

Consultas	Tempo / Linhas retornadas
Consulta 1	0.078 seg./326 rows
Consulta 2	0.015 seg./1 row
Consulta 3.a	0.000 seg./33 rows
Consulta 3.b	0.000 seg./33 rows
Consulta 3.c	0.000 seg./33 rows
Consulta 4	0.094 seg./395 rows
Consulta 5	0.078 seg./36 rows

Fonte: Tabela de autoria própria

A tabela 6 demonstra as consultas otimizadas realizadas no BD de 10.600 registros:

Tabela 6 - Consultas otimizadas BD pequeno

Consultas	Tempo / Linhas retornadas
Consulta 1	0.125 seg./5255 rows
Consulta 2	0.015 seg./1 row
Consulta 3.a	0.062 seg./957 rows
Consulta 3.b	0.047 seg./957 rows
Consulta 3.c	0.078 seg./957 rows

Consulta 4	0.249 seg./395 rows
Consulta 5	0.140 seg./783 rows

Fonte: Tabela de autoria própria

A tabela 7 demonstra as consultas otimizadas realizadas no BD de 100.600 registros:

Tabela 7 - Consultas otimizadas BD médio

Consultas	**Tempo / Linhas retornadas**
Consulta 1	0.140 seg./50300 rows
Consulta 2	0.015 seg./1 row
Consulta 3.a	0.094 seg./9052 rows
Consulta 3.b	0.120 seg./9052 rows
Consulta 3.c	0.187 seg./9052 rows
Consulta 4	0.202 seg./95 rows
Consulta 5	0.764 seg./7241 rows

Fonte: Tabela de autoria própria

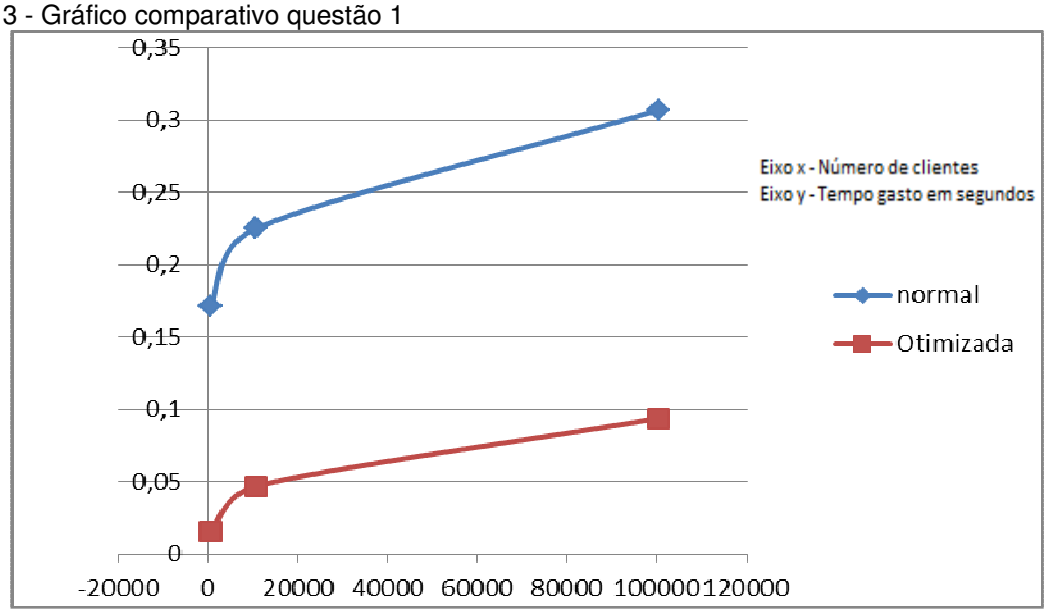
4.5. Análise comparada

Com os dados expostos de maneira separada, é difícil tirar qualquer conclusão ou informação relevante deles, portanto para facilitar a análise dos resultados segue uma análise comparativa dos resultados obtidos nas seções 4.3 e 4.4.

Os próximos gráficos mostram duas linhas, uma para o resultado das consultas otimizadas e outra para as normais, ambas da mesma questão, essas linhas foram feitas a partir de três pontos, cada ponto representa um tempo de execução de uma determinada consulta em um determinado tamanho de BD.

A figura 3 representa a comparação das consultas realizadas referente à questão 1.

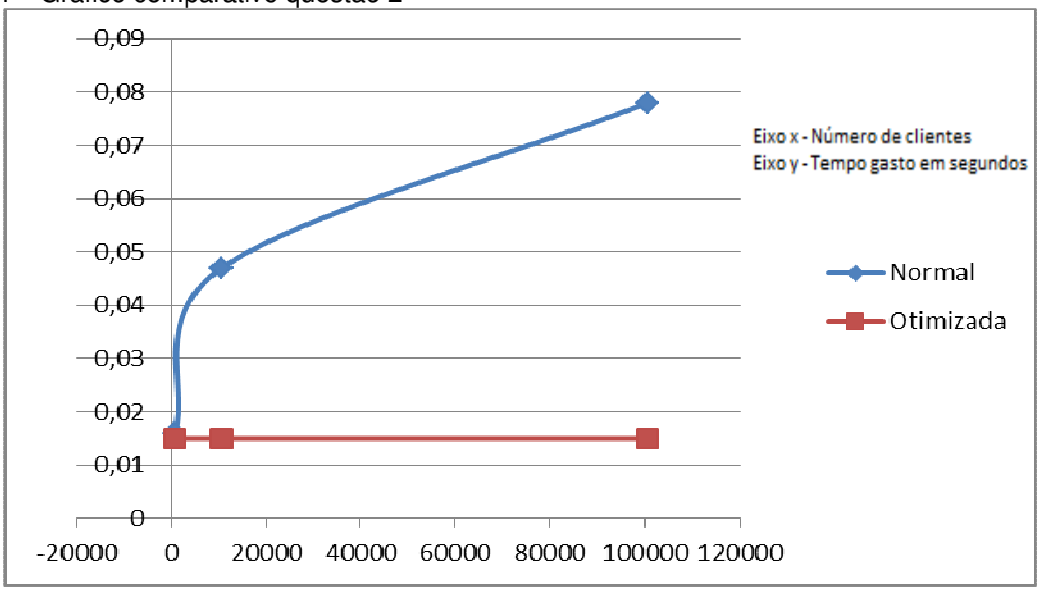
Figura 3 - Gráfico comparativo questão 1



Fonte: Figura de autoria própria

A figura 4 representa a comparação das consultas realizadas referente à questão 2.

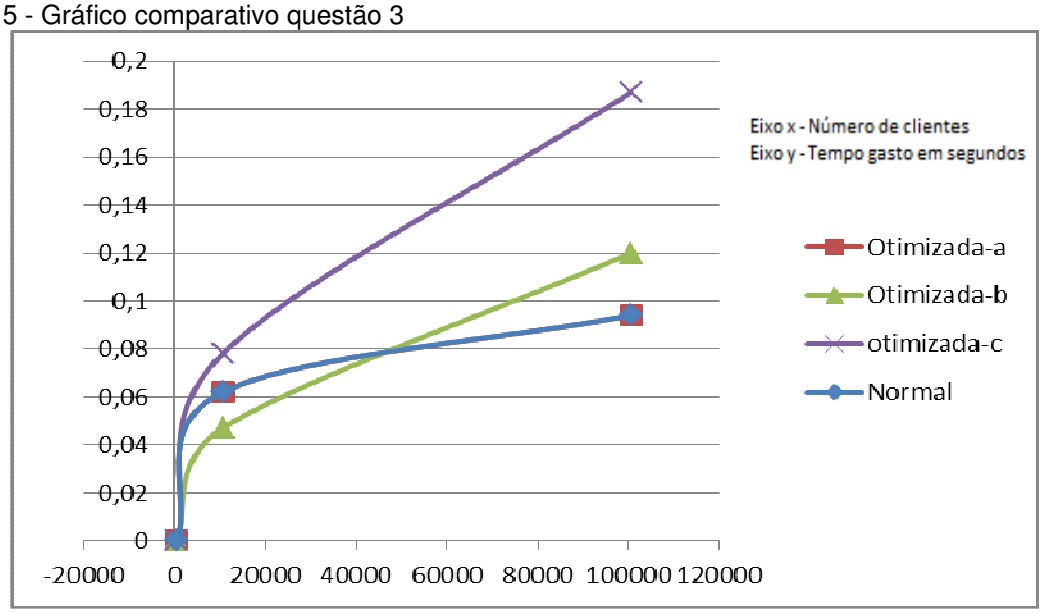
Figura 4 – Gráfico comparativo questão 2



Fonte: Figura de autoria própria

A figura 5 representa a comparação das consultas realizadas referente à questão 3.

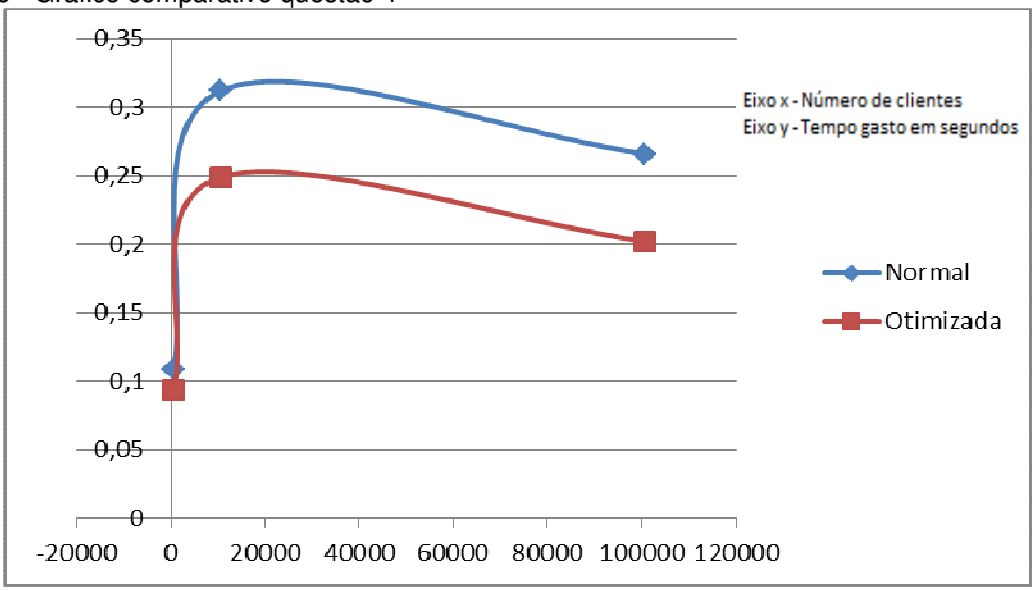
Figura 5 - Gráfico comparativo questão 3



Fonte: Figura de autoria própria

A figura 6 representa a comparação das consultas realizadas referente à questão 4.

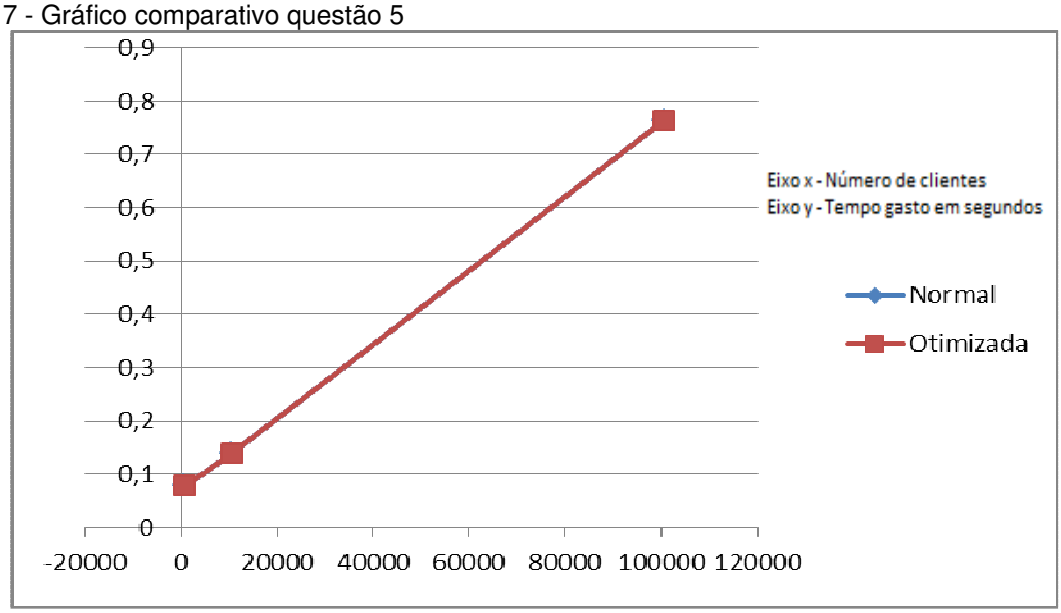
Figura 6 - Gráfico comparativo questão 4



Fonte: Figura de autoria própria

A figura 7 representa a comparação das consultas realizadas referente à questão 5.

Figura 7 - Gráfico comparativo questão 5



Fonte: Figura de autoria própria

A partir destes gráficos, é possível observar alguns comportamentos que permitirão a formulação de uma ou mais conclusões.

A primera observação feita é de que em quase todos os gráficos as consultas otimizadas são executadas de maneira igual ou mais rápida que as normais, com exceção à consulta três, portanto uma análise mais profunda nos resultados dessa consulta torna-se indispensável.

Nota-se que as consultas otimizadas “a” e “b” não demonstram resultado muito diferente da consulta normal, entretanto a consulta otimizada “c” demonstra uma evidente perda de desempenho em relação às outras. O fator que levou a esse resultado foi a indexação, que apesar de ser uma das técnicas mais utilizadas para a otimização de consultas, pode acabar gerando resultados inversos do esperado, pois quando utilizado em BD de tamanho insuficiente ele acaba por prejudicar o desempenho em vez de melhora-lo, o grande desafio então é descobrir o tamanho certo para que a indexação seja benéfica em vez de prejudicial.

É possível observar também que o gráfico da consulta cinco não traz nenhuma diferença, portanto o método de otimização utilizado, neste caso a padronização da sintaxe, não surtiu efeito. Isso pode ter acontecido devido ao software utilizado, no caso o MySQL conseguiu identificar a sintaxe independente

dos espaços utilizados e das letras minúsculas ou maiúsculas, de forma transparente para o programador.

Por fim, os gráficos 1, 2 e 4 apresentaram uma diferença significativa na performance quando comparadas as consultas otimizadas com as normais, portanto é evidente que nesses casos os métodos utilizados fizeram de fato a diferença. O constante tempo de execução da consulta otimizada no gráfico 2 pode ser explicado pela velocidade da consulta, durante os testes ficou claro que quando as consultas são muito rápidas, seu tempo de execução fica entre 0 e 0,016 segundos, qualquer valor nesse intervalo de tempo pode ser considerado nulo, pois existe um pequeno tempo gasto para abrir as tabelas usadas nas consultas, portanto esse resultado específico não reflete o tempo gasto nas consultas, que é o foco do estudo de caso.

5 CONCLUSÃO

Otimização de consultas SQL é um assunto cada vez mais discutido, analisado e praticado dentre os administradores e programadores de BD, porém é também um assunto muito complexo que envolve além da inteligência humana, o suporte do software, portanto tem de haver um conhecimento sobre o SGBD utilizado e como é possível tirar dele o melhor proveito através de técnicas específicas.

Dito isso, ficou claro através da análise comparada que não são todas as técnicas de otimização que funcionam para todas as situações e todos os SGBD. Exigi-se assim um cuidado a mais na construção da sintáxe das consultas SQL, que muitas vezes administradores e programadores não tem, e são nesses casos que o BD pode demonstrar declínio de desempenho, ou seja, demora na obtenção de resultados das consultas.

Foi possível observar também que os computadores dos dias atuais tem uma capacidade de processamento muito alta, portanto o tempo de resposta das consultas é muito baixo quando se trata de um BD de micro a médio porte, provavelmente será observada uma diferença mais significante se tratando de BD gigantescos (como os sites de buscas que varrem grande parte da internet).

Com esse pensamento, é interessante propor estudos futuros voltados a BD com esse tipo de volume, pois assim deve ser possível obter diferença na casa dos segundos ou até minutos, facilitando a avaliação do real benefício financeiro que uma otimização trará.

Com relação aos BD de micro a pequeno porte, conclui-se que a otimização de consultas pode fazer alguma diferença, mas não será significativa para obter um ganho de custo ou de satisfação do cliente, portanto caso a empresa não tenha projeções de crescimento exponencial, a otimização não deve ser tratada com um nível de urgência e prioridade alto.

No caso dos BD de médio porte, a otimização já começa a dar sinais de uma melhoria mais acentuada, se juntado com o tráfego intenso de usuários essa melhoria pode gerar frutos ainda maiores, alcançando assim uma percepção do usuário final, e como consequência uma satisfação maior do cliente.

REFERÊNCIAS

ASSOCIAÇÃO BRASILEIRA DE NORMAS TÉCNICAS. **Citação**: NBR-10520/ago - 2002. Rio de Janeiro: ABNT, 2002.

_____. **Referências**: NBR-6023/ago. 2002. Rio de Janeiro: ABNT, 2002.

COUTO, E. Aumentando a performance da aplicação através da otimização de SQL. **Revista Imasters**. Maio 2013. Ano 02. Edição 06. São Paulo: ISSOO. 2006 Disponível em: <http://imasters.com.br/artigo/4055/bancodedados/ aumentando_a_performance_da_aplicacao_atraves_da_otimizacao_de_sql/>. Acesso em: 20 Maio 2012.

DATE, C. J. **Introdução a sistemas de banco de dados**. Tradução da 8ª Edição Americana. Rio de Janeiro: Elsevier, 2004.

Heuser, C. A. **Projeto de banco de dados**. 3. Ed. Porto Alegre - RS: Artmed Editora S.A, 2008. p.14.

MAYER, R.C. **Otimizando a performance de bancos de dados relacionais**. Rio de Janeiro: Editora Axcel Books, 2001.

SILBERSCHATZ, A.; KORTH, H. F.; SUDARSHAN S. **Sistema de banco de dados**. 5ª Ed. Rio de Janeiro: Elsevier, 2006.

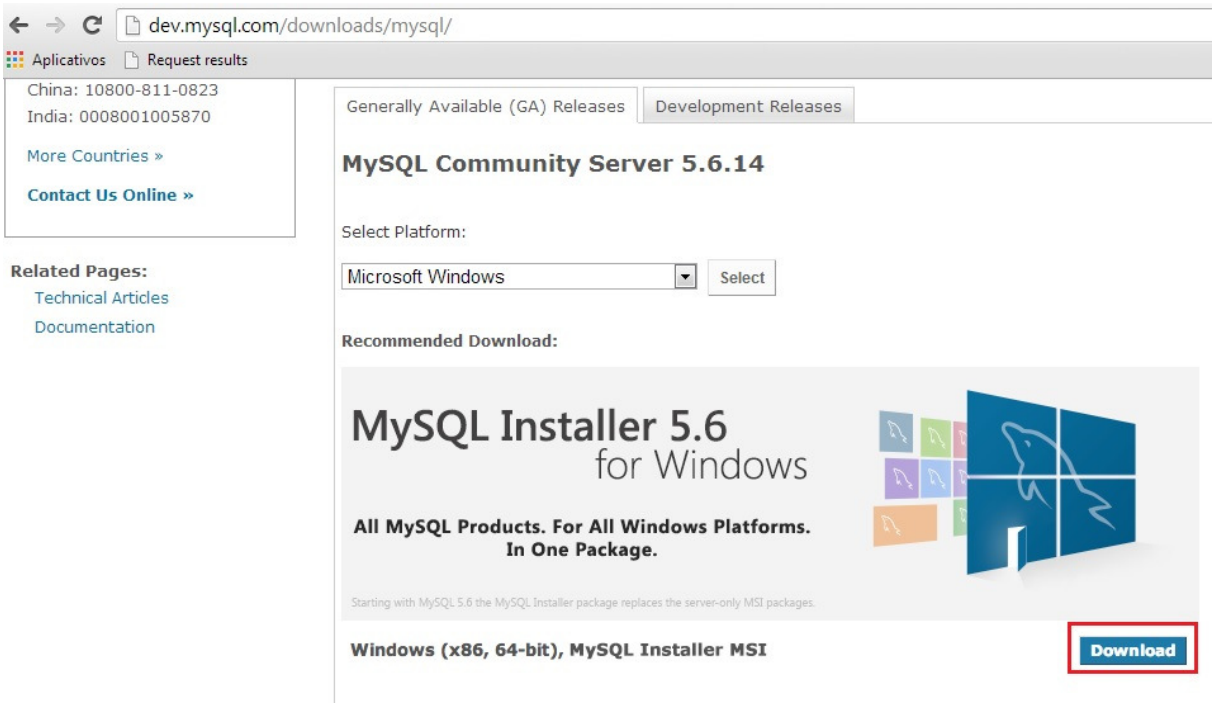
TAKAI, O.K.; ITALIANO I.C.; FERREIAR J.E. **Introdução a banco de dados**. 1ª Ed. São Paulo: DCC-IME-USP, 2005.

VOLACO, E. B. Otimização de comandos SQL. **Revista SQL Magazine**. Out 2007. Ano 01. Edição 01. São Paulo: DevMedia. 2006. Disponível em: <http://www. devmedia.com.br/post-6926-Artigo-SQL-Magazine-1-Otimizacao-de-Comandos-SQL.html>. Acesso em: 10 Abr 2012.

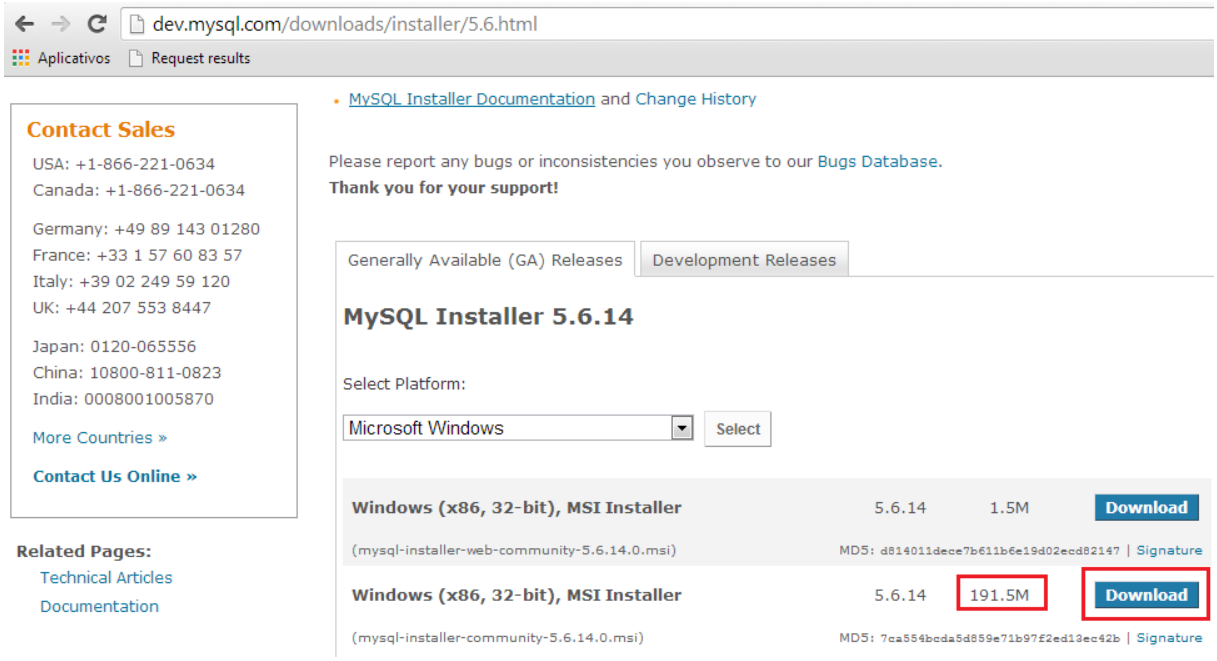
APÊNDICE A – INSTALAÇÃO MySQL

A fim de facilitar a reprodução dos testes comparativos executados nessa monografia e demonstrar a facilidade de instalação de um SGBD, segue um passo a passo de como instalar e configurar o MySQL desde o momento de ser download.

É possível encontrar o instalador no próprio site do software com o nome de MySQL Community Server 5.6.14: http://dev.mysql.com/downloads/mysql/.



Após clicar em “Download”, você será redirecionado para outra página, onde irá selecionar o arquivo com maior tamanho:



Depois de baixado, basta executar o instalador “mysql-installer-community-5.6.14.0”, que ele dará início ao processo de instalação do SGBD. A primeira tela do instalador dará três opções, selecione “Install MySQL Products”, a partir desse momento o processo de instalação é dividido em sete etapas:

1ª etapa: Serão listados os termos de licença, aceite os termos e selecione “Next”.

2ª etapa: O instalador fará uma busca por novas atualizações do produto, caso encontre irá instalar, em seguida selecione “Execute”.

3ª etapa: Escolher o tipo de configuração, como será desenvolvido um BD, selecione “Developer Default” e em seguida “Next”.

4ª etapa: Verifique requisitos, o próprio instalador fará uma varredura no seu sistema e indicará se os requisitos necessários estão instalados e caso não estejam ele instalará. Selecione “Next”.

5ª etapa: Progresso da instalação, o instalador dará um status de como a instalação está prosseguindo e ao final indicará se houve sucesso ou falha na instalação. Selecione “Execute”.

6ª etapa: Configuração, essa etapa é dividida em quatro partes:

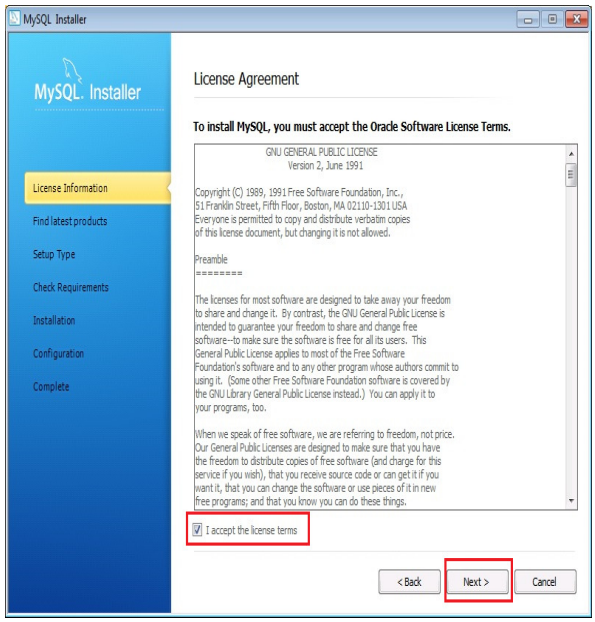
- Progresso da configuração: Mostrará um status da configuração, selecione “Next”;

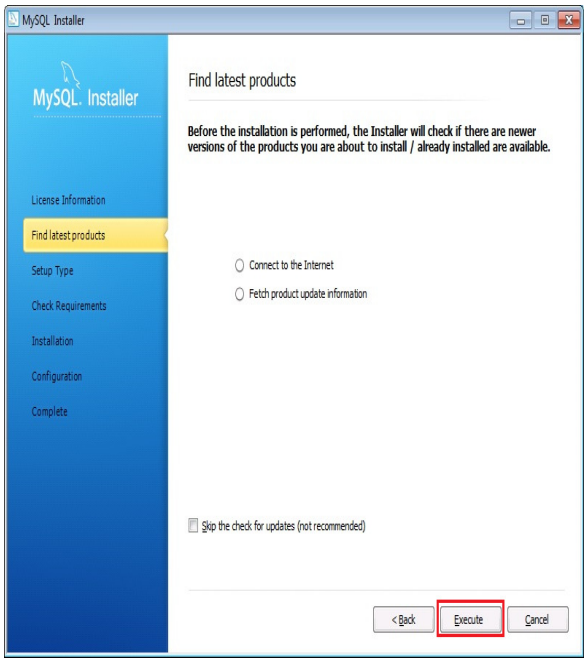
- Configuração do Server MySQL parte 1: Fornecerá opções para configurar o servidor, neste caso será usado as configurações padrão, portanto selecione “Next”;

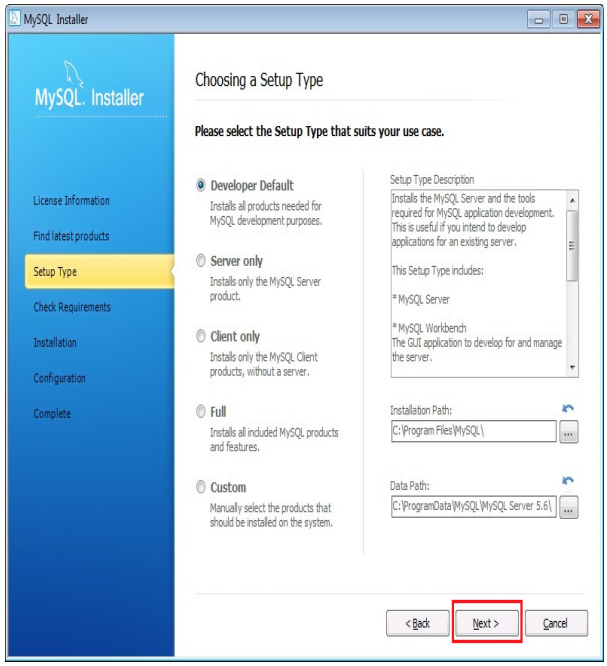
- Configuração do Server MySQL parte 2: Pede para que se defina a senha do usuário raiz. Digite uma senha e selecione “Next”;

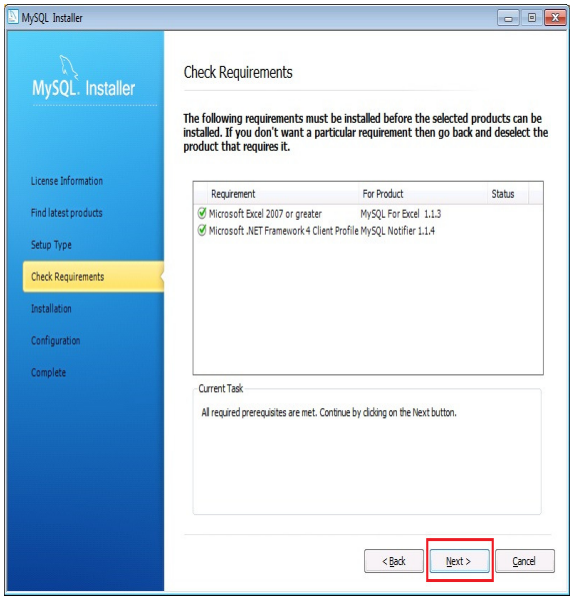
- Configuração do Server MySQL parte 2: Define o nome do serviço que o MySQL usará, por padrão será MySQL56. Selecione “Next”;

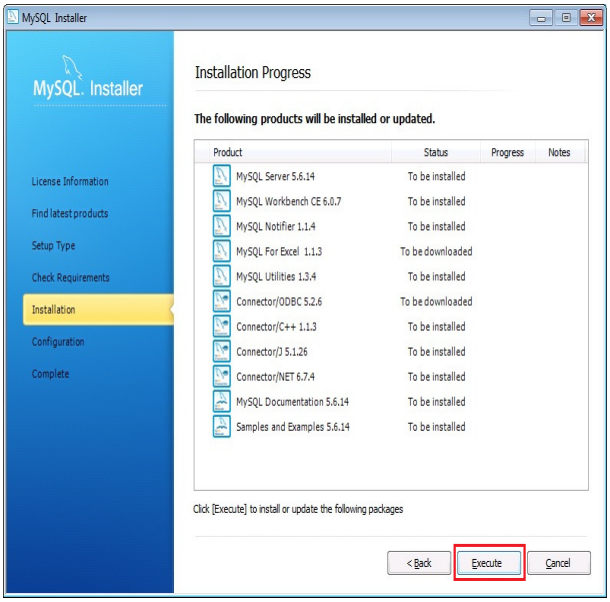
7ª etapa: Conclusão da instalação, o instalador informará que a instalação foi concluída. Selecione “Finish”.

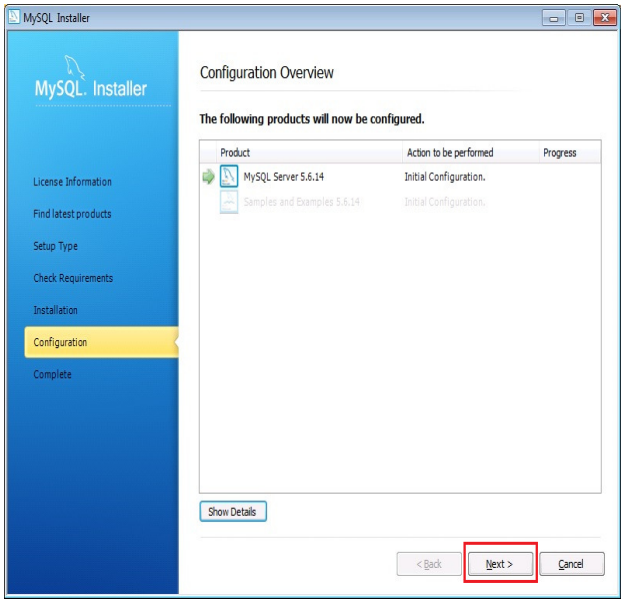


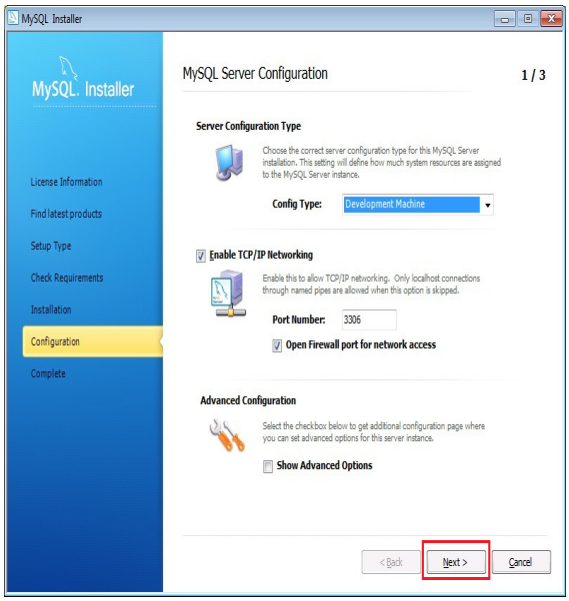

MySQL Installer
MySQL. Installer
Find latest products
Before the installation is performed, the Installer will check if there are newer versions of the products you are about to install / already installed are available.
License Information
Find latest products
Setup Type
Check Requirements
Installation
Configuration
Complete
Connect to the Internet
Fetch product update information
Skip the check for updates (not recommended)
< Back
Execute
Cancel

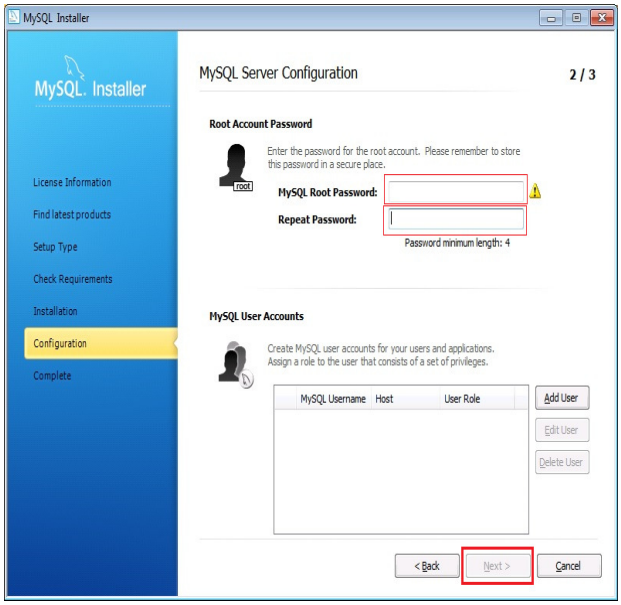

MySQL Installer
MySQL. Installer
Choosing a Setup Type
Please select the Setup Type that suits your use case.
License Information
Find latest products
Setup Type
Check Requirements
Installation
Configuration
Complete
Developer Default
Installs all products needed for MySQL development purposes.
Server only
Installs only the MySQL Server product.
Client only
Installs only the MySQL Client products, without a server.
Full
Installs all included MySQL products and features.
Custom
Manually select the products that should be installed on the system.
Setup Type Description
Installs the MySQL Server and the tools required for MySQL application development. This is useful if you intend to develop applications for an existing server.
This Setup Type includes:
* MySQL Server
* MySQL Workbench
The GUI application to develop for and manage the server.
Installation Path:
C:\Program Files\MySQL\
Data Path:
C:\ProgramData\MySQL\MySQL Server 5.6\
< Back
Next >
Cancel

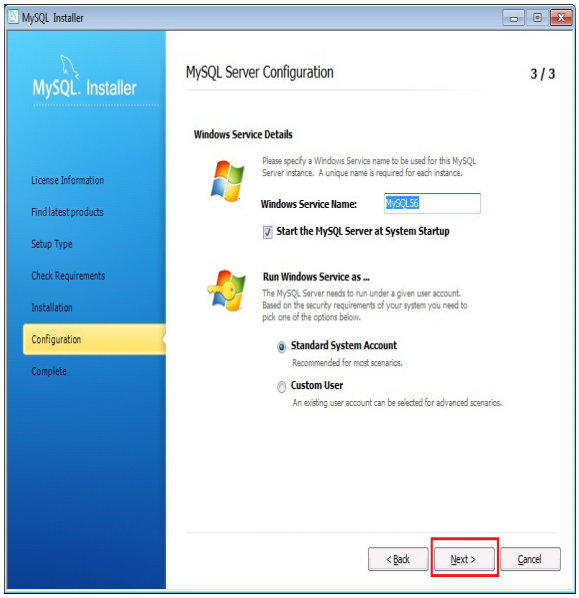

MySQL Installer
MySQL. Installer
Check Requirements
The following requirements must be installed before the selected products can be installed. If you don't want a particular requirement then go back and deselect the product that requires it.
License Information
Find latest products
Setup Type
Check Requirements
Installation
Configuration
Complete
Requirement
For Product
Status
Microsoft Excel 2007 or greater
MySQL For Excel 1.1.3
Microsoft .NET Framework 4 Client Profile
MySQL Notifier 1.1.4
Current Task
All required prerequisites are met. Continue by clicking on the Next button.
< Back
Next >
Cancel

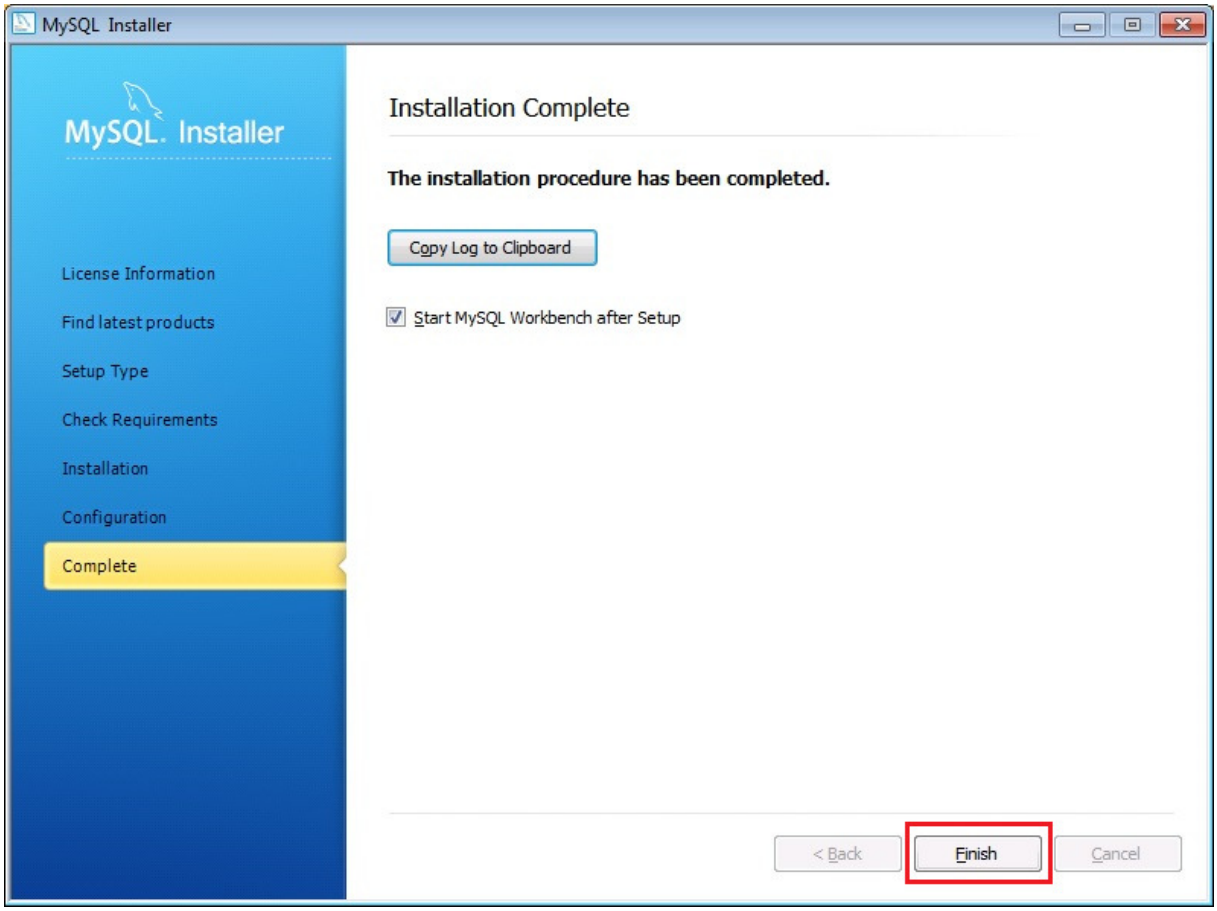

MySQL Installer
MySQL. Installer
Installation Progress
The following products will be installed or updated.
License Information
Find latest products
Setup Type
Check Requirements
Installation
Configuration
Complete
Product
Status
Progress
Notes
MySQL Server 5.6.14
To be installed
MySQL Workbench CE 6.0.7
To be installed
MySQL Notifier 1.1.4
To be installed
MySQL For Excel 1.1.3
To be downloaded
MySQL Utilities 1.3.4
To be installed
Connector/ODBC 5.2.6
To be downloaded
Connector/C++ 1.1.3
To be installed
Connector/J 5.1.26
To be installed
Connector/NET 6.7.4
To be installed
MySQL Documentation 5.6.14
To be installed
Samples and Examples 5.6.14
To be installed
Click [Execute] to install or update the following packages
< Back
Execute
Cancel


MySQL Installer
MySQL. Installer
Configuration Overview
The following products will now be configured.
Product
Action to be performed
Progress
MySQL Server 5.6.14
Initial Configuration.
Samples and Examples 5.6.14
Initial Configuration.
License Information
Find latest products
Setup Type
Check Requirements
Installation
Configuration
Complete
Show Details
< Back
Next >
Cancel


MySQL Installer
MySQL. Installer
MySQL Server Configuration
1 / 3
Server Configuration Type
Choose the correct server configuration type for this MySQL Server installation. This setting will define how much system resources are assigned to the MySQL Server instance.
Config Type:
Development Machine
Enable TCP/IP Networking
Enable this to allow TCP/IP networking. Only localhost connections through named pipes are allowed when this option is skipped.
Port Number:
3306
Open Firewall port for network access
Advanced Configuration
Select the checkbox below to get additional configuration page where you can set advanced options for this server instance.
Show Advanced Options
License Information
Find latest products
Setup Type
Check Requirements
Installation
Configuration
Complete
< Back
Next >
Cancel


MySQL Installer
MySQL. Installer
MySQL Server Configuration
2 / 3
Root Account Password
Enter the password for the root account. Please remember to store this password in a secure place.
MySQL Root Password:
Repeat Password:
Password minimum length: 4
MySQL User Accounts
Create MySQL user accounts for your users and applications. Assign a role to the user that consists of a set of privileges.
MySQL Username
Host
User Role
Add User
Edit User
Delete User
License Information
Find latest products
Setup Type
Check Requirements
Installation
Configuration
Complete
< Back
Next >
Cancel


MySQL Installer
MySQL. Installer
MySQL Server Configuration
3 / 3
Windows Service Details
Please specify a Windows Service name to be used for this MySQL Server instance. A unique name is required for each instance.
Windows Service Name:
MySQL56
Start the MySQL Server at System Startup
Run Windows Service as ...
The MySQL Server needs to run under a given user account. Based on the security requirements of your system you need to pick one of the options below.
Standard System Account
Recommended for most scenarios.
Custom User
An existing user account can be selected for advanced scenarios.
License Information
Find latest products
Setup Type
Check Requirements
Installation
Configuration
Complete
< Back
Next >
Cancel


MySQL Installer
MySQL. Installer
License Information
Find latest products
Setup Type
Check Requirements
Installation
Configuration
Complete
Installation Complete
The installation procedure has been completed.
Copy Log to Clipboard
Start MySQL Workbench after Setup
< Back
Finish
Cancel