CENTRO UNIVERSITÁRIO UNIVATES
CURSO DE ENGENHARIA DA COMPUTAÇÃO

FERRAMENTA DE COLETA E ANÁLISE DE ESTATÍSTICAS PARA SISTEMA GERENCIADOR DE BANCO DE DADOS POSTGRESQL

Fabiano Tomasini

Lajeado, junho de 2015.

Fabiano Tomasini

FERRAMENTA DE COLETA E ANÁLISE DE ESTATÍSTICAS PARA SISTEMA GERENCIADOR DE BANCO DE DADOS POSTGRESQL

Trabalho de Conclusão de Curso I apresentado ao curso de Engenharia da computação do Centro Universitário Univates, para obtenção parcial do título de Bacharel em Engenharia da Computação.

Orientador: Prof. Evandro Franzen

Lajeado, junho de 2015.

RESUMO

O crescimento das empresas e organizações tem evidenciado a importância da informação para a tomada de decisões, o volume cada vez maior de dados têm exigido atenção com a velocidade com que as informações são obtidas e preocupações quanto a forma com que os dados são armazenados. É fundamental garantir a integridade das informações e um bom desempenho em seu acesso. Algumas dessas preocupações podem ser amenizadas com algumas ações que podem ser tomadas em relação ao SGBD (Sistema de Gerenciamento de Banco de Dados) escolhido. Este trabalho tem o objetivo de desenvolver uma ferramenta que possibilita identificar, de forma visual, possíveis problemas de desempenho em relação ao SGBD e sugerir possíveis formas de resolver ou contornar estes problemas. Existem várias técnicas que possibilitam o resultado esperado, ou seja, ter a informação desejada de forma rápida e correta. Nesse projeto serão abordados gargalos de performance que estejam relacionados a administração de banco de dados que serão obtidas através das estatísticas extraídas do PostgreSQL. Possibilitando, monitoramento em tempo real da performance, sugerindo criação de índices, alterações em configurações do sistema utilizado e criação de alarmísticas avisando de possíveis problemas. Todas com a finalidade de melhorar o desempenho do software. A ferramenta será desenvolvida para facilitar a otimização e o monitoramento do PostgreSQL, porém, possibilitando que seja implementado o monitoramento de outros SGBDs.

Palavras-chave: PostgreSQL, banco de dados, desempenho e otimização.

ABSTRACT

The growth of enterprises and organizations has shown the importance of information for decision making, the increasing volume of data has required attention to the speed with which information is obtained and concerns about the way the data is stored and is essential to ensure the integrity of information and a good performance in access. Some of these concerns can be alleviated with some actions that can be taken in relation to the DBMS (Database Management System) chosen. This work aims to develop a tool that helps identify, visually, potential performance issues in relation to the DBMS and suggest possible ways to solve or circumvent these problems. There are several techniques that allow the expected result, namely to have the desired information quickly and correctly. This project will address performance bottlenecks that are related to database administration to be obtained through the statistics extracted from PostgreSQL. Enabling real-time monitoring of performance, suggesting creating indexes, changes to system settings used and creating alarmísticas warning of possible problems, all with the purpose of improving the performance of the software. The tool will be developed to facilitate the optimization and monitoring of PostgreSQL, however, enabling the monitoring of other DBMSs to be implemented.

key words: PostgreSQL, database, performance and optimization.

LISTA DE FIGURAS

LISTA DE TABELAS

LISTA DE ABREVIATURAS

HTML:	HyperText Markup Language
HTTP:	Hypertext Transfer Protocol
PHP:	PHP Hypertext Preprocessor
SGBD:	Sistema de Gerenciamento de Banco de Dados
SQL:	Structured Query Language
BSD:	Berkeley Software Distribution
DBA:	Database Adminstrator
SaaS:	Software as a Service
GNU:	General Public License

SUMÁRIO

1 INTRODUÇÃO

Com o desenvolvimento tecnológico os softwares exigem mais dos computadores, ou seja, maior poder de processamento de modo que informações importantes sejam obtidas de maneira fácil e rápida. Sendo essas informações a chave para uma boa tomada de decisão podemos considerar que a mesma garante a sobrevivência de uma organização no mercado global, motivando diversos estudos na área de Banco de dados (ROB; CORONEL, 2011).

Segundo Silberschatz, Korth e Sudarshan (2012) um banco de dados é considerado uma coleção de dados relacionados que são fornecidos ao usuário final de maneira conveniente, eficiente e segura. Visto que a manipulação de informações é tão importante para a gestão de uma organização, cientistas da computação têm desenvolvido um grande conjunto de conceitos e técnicas de gerenciamento de dados visando a otimização da obtenção de dados.

Para garantir o armazenamento correto e manter a integridade dos dados é indispensável monitorar o uso do SGBD, verificando por exemplo a ocorrências de impasses (*deadlocks*) que podem afetar a disponibilidade do software, além disso é fundamental analisar a performance das consultas realizadas e isso requer o conhecimento das estatísticas do banco de dados, tais como, número de consultas por tabela que utilizaram ou não utilizaram índices, percentual de consultas por tabela que utilizaram ou não utilizaram cache.

O presente trabalho,visa apresentar uma solução para a manutenção de grandes bancos de dados com uma ênfase especial na performance das aplicações que dependem desse serviço. Para obter o resultado esperado, serão abordados gargalos de performance relacionados a administração de banco de dados através de estatísticas extraídas do SGBD. As

informações coletadas servirão para medir a performance do banco e alertando o DBA (*Database administrator*) de possíveis pontos melhorias e possíveis problemas.

A ferramenta contará com módulos coletores de estatísticas que deverão ter acesso ao banco de dados que estará sendo analisado. Nesse trabalho, será desenvolvido o módulo coletor para o SGBD PostgreSQL, porém, possibilitando que seja implementado o monitoramento de outros SGBDs em trabalhos futuros.

1.1 Motivação

Com base na importância do monitoramento do SGBD para manter a disponibilidade do serviço, e da análise de estatísticas para garantir a otimização de consultas, é evidenciada a necessidade de se ter ferramentas que auxiliam na execução desse trabalho.

Atualmente, existem varias ferramentas que tem como objetivo gerenciar tabelas, acessos, porém, quando se deseja monitorar estatísticas do banco de dados, são encontrados poucos softwares que fornecem essas informações de maneira gráfica. Algumas ferramentas atingem o objetivo, mas acabam sendo descartadas por serem pagas, descontinuadas, não suportar mais de um SGBD ou simplesmente não proporcionam o que se deseja de maneira clara.

Tendo como base essa realidade, o presente trabalho, visa proporcionar ao DBA (Database Adminstrator) uma ferramenta que possibilita o monitoramento e análise de estatísticas de maneira gráfica e de fácil manuseio.

1.2 Objetivos

O objetivo geral do presente trabalho é disponibilizar uma ferramenta para auxiliar na administração, monitoramento e otimização de um banco de dados que utilize qualquer SGBD e que tenha um grande volume de informação armazenada. Inicialmente, o software terá um enfoque maior no sistema PostgreSQL, porém, possibilitará que seja implementado o monitoramento de outros SGBDs em trabalhos futuros.

1.2.1 Objetivos específicos

Para atingir o objetivo principal definido, os seguintes objetivos específicos também deverão ser cumpridos:

a) Compreender os princípios de coleta e utilização de estatísticas de um banco de dados;

b) Permitir a coleta de estatísticas através de rotinas automatizadas para vários SGBDs;

c) Contribuir para uma melhor administração de dados através da ferramenta proposta.

1.3 Organização do trabalho

A fim de melhorar a compreensão do presente trabalho, os capítulos serão apresentados na seguinte ordem.

O capítulo 2 explicará os conceitos necessários para o desenvolvimento da proposta, tais como, conceitos sobre bancos de dados relacionais, administração de bancos de dados, consultas, concorrência, transação etc.

No capítulo 3 será feito um estudo comparativo de ferramentas já existentes para monitoramento e otimização de SGBDs.

O capítulo 4 apresenta a proposta do presente trabalho onde serão abordadas a metodologia utilizada para a realização do trabalho, as etapas de desenvolvimento do projeto e as funcionalidades da ferramenta.

Por fim, o capítulo 5 apresenta as considerações parciais do presente trabalho.

2 REFERENCIAL TEÓRICO

Neste capítulo, serão abordados os principais conceitos necessários para o desenvolvimento da proposta. Foram realizados estudos em artigos, livros e documentos digitais. Dentre as informações levantadas, destacam-se, conceitos sobre bancos de dados, administração de sistema de gerenciamento de bancos de dados, processamento de consultas e controle de concorrência.

2.1 Banco de dados

Segundo Silberschatz, Korth e Sudarshan (2012), um banco de dados, é uma coleção de dados relacionados, ou seja, sempre que tenho informações que se relacionam e que tratam do mesmo assunto, pode ser considerado um banco de dados.

Um banco de dados de uma universidade, por exemplo, poderia conter informações sobre os alunos, professores e suas turmas, e, poderiam estar relacionados as matriculas dos alunos nos cursos ministrados pelos professores e ao uso de salas por curso. (RAMAKRISHNAN; GEHRKE, 2008).

O banco de dados tem como principal objetivo, armazenar e permitir aos usuários buscar e atualizar informações quando necessário. Essas informações podem ser qualquer dado que tenha algum significado para um usuário ou organização, em outras palavras, as informações são dados necessários para auxiliar no processo geral de tomada de decisões (DATE, 2004).

2.2 Sistema de gerenciamento de banco de dados

Segundo Ramakrishnan, Gehrke(2008) um sistema de gerenciamento de banco de dados, ou SGBD, é um software que foi desenvolvido para auxiliar na manipulação e organização de vastos conjuntos de dados. A necessidade de um software com esse fim tem crescido rapidamente, pois antes de surgirem, os dados eram armazenados em arquivos e era necessário escrever códigos específicos na própria aplicação para gerenciá-los.

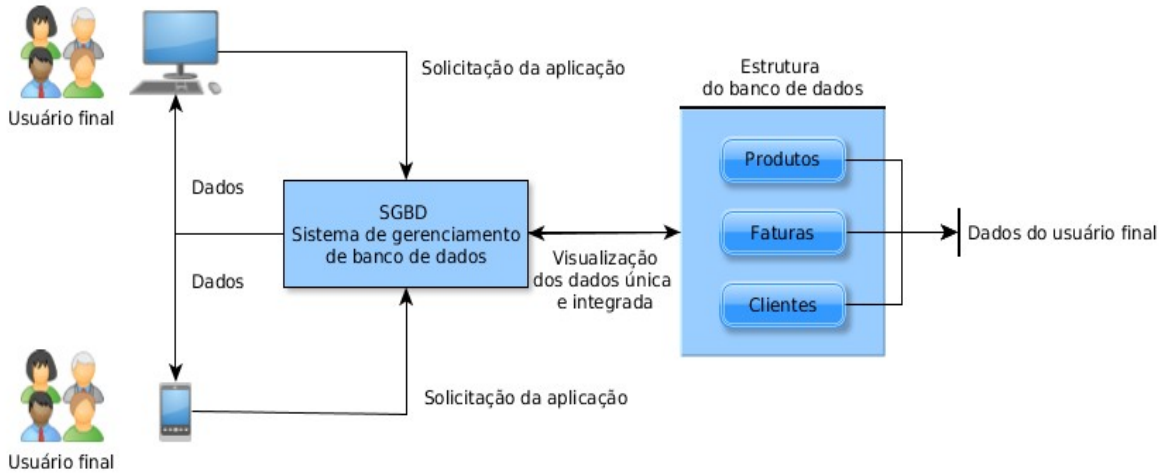
Para Siberschartz, Korth, Sudarshan (2012) um sistema de gerenciamento de banco de dados (SGBD) é considerado uma coleção de dados inter-relacionados e um compilado de programas para manipular esses dados. A coleção de dados, mais conhecida como banco de dados, armazenam informações importantes para uma empresa. O principal propósito de um SGBD é possibilitar uma forma de armazenar e obter informações de um banco de dados de maneira conveniente e eficiente.

Um SGBD nada mais é que um conjunto de programas que gerenciam a estrutura do banco de dados e controlam os acessos aos dados armazenados (ROB;CORONEL, 2011).

Os SGBDs proporcionam produtividade na manipulação de informações, ou seja, obter de maneira rápida os dados desejados para uma decisão. Utilizando uma linguagem de alto nível, os SGBDs permitem que seus usuários escrevam consultas de maneira simples sem definir detalhes relacionados ao seu processamento, tarefa a qual é atribuída ao próprio SGBD que irá escolher através de um processo de planejamento e otimização, a forma mais eficaz de obter os dados desejados (DATE 2004).

Segundo Rob, Coronel (2011) um SGBD serve como um intermediário entre o banco de dados e o usuário final. Em sua estrutura interna os dados são armazenados em um conjunto de arquivos e a única forma de acessar as informações é por meio do SGBD. A figura 1 ilustra que são mostrados para o usuário final uma visualização única e integrada das informações armazenadas. O SGBD recebe diversos comandos enviados a partir de aplicações e os traduz em uma linguagem compreensiva para atendê-los. O SGBD oculta dos usuários boa parte da complexidade interna do banco de dados.

Figura 1 - O SGBD gerencia a interação entre o usuário final e o banco de dados



Fonte: Modificado de Rob, Coronel (2011,p. 7)

2.2.1 Modelo relacional

Segundo Ramakrishnan, Gehrke(2008), um modelo de dados é considerado uma seleção de construtores de alto nível que ocultam detalhes de baixo nível do armazenamento de informações. O SGBD permite que o usuário final defina as informações a serem armazenadas em relação ao modelo de dados. Grande parde dos SGBDs atuais baseia-se no modelo relacional.

O modelo relacional devido sua compreensibilidade é considerado o principal modelo de dados utilizado atualmente, facilitando o trabalho dos desenvolvedores de software, diferentemente dos modelos mais antigos como o modelo de rede ou o modelo hierárquico (SILBERSCHATZ; KORTH; SUDARSHAN, 2012).

Segundo Siberschartz, Korth, Sudarshan (2012) um banco de dados que utiliza o modelo relacional contém um conjunto de tabelas (conjunto de entidades), onde cada tabela recebe uma denominação única que irá representá-la. Cada tabela contém campos (atributos) que também recebem uma denominação única e um tipo especifico de dados.

2.2.2 Entidades

De acordo com Siberschartz, Korth, Sudarshan (1999) uma entidade é uma “coisa” ou um “objeto” do mundo real que pode ser determinada de forma homogênea em relação a todos os outros objetos. Por exemplo, em uma empresa, cada pessoa é uma entidade. Uma entidade é composta por uma ou mais propriedades, sendo que algumas dessas propriedades devem ser únicas. Uma entidade pode ser concreta, como uma pessoa ou um livro, ou pode ser abstrata como um empréstimo.

2.2.3 Atributos

Os atributos são características descritivas de cada membro de um conjunto de entidades (tabelas). A escolha de um atributo para uma tabela mostra que o banco de dados mantém informações equivalentes de cada entidade, entretanto cada entidade pode ter seu próprio valor em cada atributo (SILBERSCHATZ; KORTH; SUDARSHAN, 1999).

2.2.4 Chaves

Segundo Siberschartz, Korth, Sudarshan (1999) uma chave em um banco de dados tem o objetivo de identificar e estabelecer ligações entre entidades. Em um banco de dados relacional existem dois tipos de chave, chave primaria e chave estrangeira.

2.2.4.1 Chave primária

A chave primaria de uma determinada entidade é representada por uma coluna ou uma combinação de colunas, onde os valores irão diferenciar de forma única uma entidade das demais dentro de uma tabela (SILBERSCHATZ; KORTH; SUDARSHAN, 1999).

2.2.4.2 Chave estrangeira

Segundo Siberschartz, Korth, Sudarshan (1999) uma chave estrangeira de uma tabela é um atributo ou um conjunto de atributos que referenciam uma chave primaria, ou seja, os atributos de uma chave estrangeira possui o mesmo valor de um atributo chave primaria de uma outra tabela. A chave estrangeira permite relacionamento entre os conjuntos de entidades contidas em um banco de dados relacional.

2.2.5 Administração de um banco de dados

Segundo Milani (2008) a administração de um banco de dados é uma tarefa que, dependendo das ferramentas utilizadas e da complexidade dos bancos de dados em questão e pode ser fácil ou extremamente difícil.

De acordo com Siberschartz, Korth, Sudarshan (1999) a tarefa de administração de um banco de dados é executada por uma pessoa denominada DBA (Database Adminstrator) que centraliza o controle do sistema. Dentre as funções do DBA podem ser destacadas as seguintes;

a) Definição da estrutura de dados e método de acesso: O DBA é responsável por criar estruturas de dados e métodos de acessos definidos através de um conjunto de instruções, as quais são traduzidas pelo compilador do SGBD (SILBERSCHATZ; KORTH; SUDARSHAN, 1999);

b) Esquema e modificação na organização física: O DBA é responsável por alterações no esquema do banco de dados por meio de um conjunto de definições que serão utilizadas pelo SGBD gerando modificações nas tabelas apropriadas (SILBERSCHATZ; KORTH; SUDARSHAN, 1999);

c) Fornecer autorização de acesso ao sistema: O DBA é responsável por regular o acesso de diferentes usuários às diferentes partes do sistema. Os dados referente aos acesos são armazenados em uma estrutura especial do SGBD e é consultada sempre que o acesso a determinado dado for solicitado. (SILBERSCHATZ; KORTH; SUDARSHAN, 1999);

d) Especificação de regra de integridade: O DBA é responsável por definir regras que devem garantir a integridade dos dados armazenados. As regras são tratadas por uma estrutura especial do SGBD e é consultada sempre que uma atualização está em curso no sistema. (SILBERSCHATZ; KORTH; SUDARSHAN, 1999);

e) Definição de esquema: O DBA é responsável pela criação do esquema do banco de dados, escrevendo um conjunto de definições que são transformadas pelo compilador do SBGD em um conjunto de tabelas armazenadas permanentemente no dicionário de dados. (SILBERSCHATZ; KORTH; SUDARSHAN, 1999).

2.2.6 Catálogo de um SGBD

Segundo Elmasri, Ramez (2011) um SGBD é um sistema genérico projetado para atender diversas aplicações de banco de dados, desta maneira, sempre que necessário conhecer a estrutura de um banco de dados específico deve se recorrer ao catálogo, pois é lá que o SGBD armazena os metadados do esquema, tais como informações sobre tabelas, colunas e informações de controle interno.

No catálogo, além de informações sobre a estrutura do banco de dados, são armazenadas também informações estatísticas que são usadas pelo SGBD para otimizar o plano de execução de uma consulta, além disso as estatísticas podem ser usadas pelo administrador de banco de dados para identificar problemas de performance, tais como índices não utilizados e a ocorrência de *deadlocks* (SILBERSCHATZ; KORTH; SUDARSHAN, 1999).

Uma consulta convencional retorna informações existentes em tabelas, já uma consulta no catálogo retorna informações sobre os bancos, os objetos dos bancos, os campos de tabelas, seus tipos de dados, seus atributos etc. Na figura 2 podemos visualizar um consulta que obtém do catálogo do PostgreSQL informações como o nome e o schema de dados de todas as tabelas de um banco de dados (POSTGRESQL;2015).

Figura 2 - Consulta no catálogo do PostgreSQL

```
SELECT schemaname AS esquema, tablename AS tabela
  FROM pg_catalog.pg_tables
 ORDER BY schemaname, tablename;
```

Fonte: Elaborado pelo autor.

2.2.7 Processamento de consultas

Segundo Siberschartz, Korth, Sudarshan (1999) o processamento de consultas refere-se ao conjunto de tarefas incluídas na extração de informações de um banco de dados. As tarefas incluem, tradução das consultas em uma linguagem que seja entendida pelo SGBD e que permitem ser usadas no nível físico do sistema de arquivos e uma série de transformações de otimização das consultas.

Antes de qualquer consulta iniciar o processamento, o SGBD precisa transcrever a consulta para uma linguagem interna. A linguagem de banco de dados SQL utilizada na maioria dos SGBDs relacionais, é ideal para a ação humana, porém essa representação não é a ideal para o sistema interno de consultas. A representação interna adequada é baseada na álgebra relacional. (SILBERSCHATZ; KORTH; SUDARSHAN, 1999).

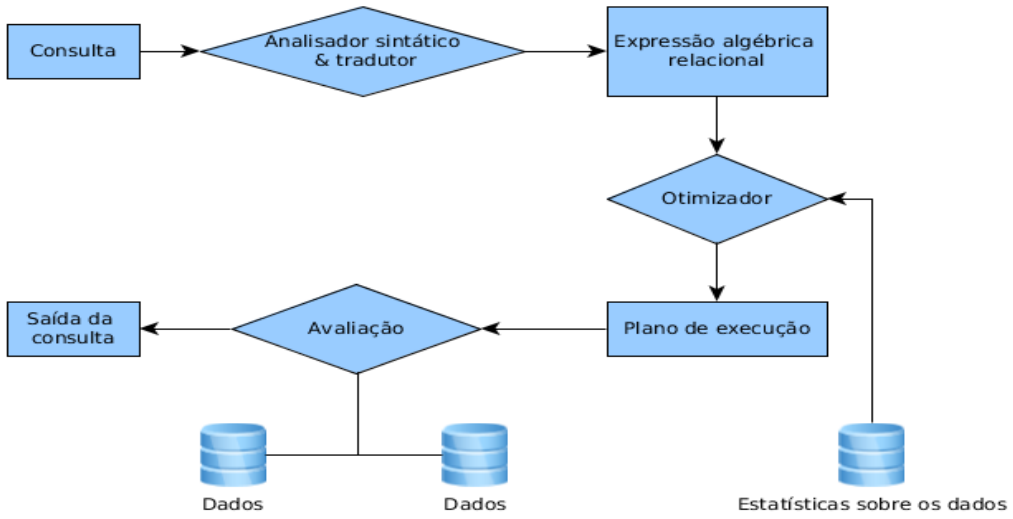
Na Figura 3 podemos visualizar as etapas envolvidas no processamento de uma consulta, onde podemos destacar as seguintes:

a) **Analisador sintático e tradutor:** Convertem a consulta para interpretação interna do SGBD. (SILBERSCHATZ; KORTH; SUDARSHAN, 1999);

b) **Otimizador:** Transforma a expressão em um modelo de álgebra relacional. (SILBERSCHATZ; KORTH; SUDARSHAN, 1999);

c) **Avaliação:** É o mecanismo responsável pela execução da consulta, executar o plano e retorna a resposta. (SILBERSCHATZ; KORTH; SUDARSHAN, 1999).

Figura 3 - Etapas no processamento de consultas no banco de dados



Fonte: Adaptado de Silberschatz, Korth, Sudarshan(1999).

Segundo Blumm, Fornari (2006) no processamento de consultas o otimizador desempenha o papel principal. Há dois tipos de otimizações:

a) **Otimização baseada em regras heurísticas:** Estas regras estão inclusas ao SGBD Oracle e produzem bons resultados, porém não há comprovação que garanta sua correção em todas as consultas (BLUMM; FORNARI, 2006);

b) **Otimização baseada em estatísticas:** Nesta opção o otimizador de consultas aplica algumas fórmulas para calcular o custo (tempo de processamento + tempo de acesso aos dados em disco) de várias alternativas possíveis e opta pela que apresentar o menor custo estimado. Os dados estatísticos como o número de linhas de uma determinada tabela são mantidos no dicionário de dados do SGBD (BLUMM; FORNARI, 2006). Para estimar os custos de diversas estratégias de execução, o SGBD registra qualquer informação necessárias para as funções de custo, essas informações são acessadas pelo otimizador sempre que necessário (ELMASRI; RAMEZ , 2011).

Segundo Blumm, Fornari (2006) alguns SGBDs utilizam apenas um dos métodos, apresentados e outros permitem a escolha do método que apresentar melhores resultados.

2.2.7.1 Álgebra relacional

Segundo Borello, Kneipp (2008) a álgebra Relacional é considerada uma linguagem de consulta procedural, onde o usuário especifica as etapas a serem executadas através de um conjunto de operações. Esta sequencia de operações forma uma expressão em álgebra relacional, do qual o resultado originará também uma consulta.

A álgebra relacional faz parte da manipulação de dados do modelo relacional e consiste em um conjunto de operações que fornece uma nova ligação a partir de uma ou mais relações existentes no banco de dados, dessa maneira proporciona os procedimentos essenciais para a execução de uma sequência de operações, de tal forma a adquirir o resultado desejado independente da forma de tratamento do banco de dados. (SILBERSCHATZ; KORTH; SUDARSHAN, 1999)

As operações da álgebra relacional pode ser decomposto em dois grupos:

a) **Conjunto de operações da teoria dos conjuntos:** UNION (união), INTERSECTION (interseção), DIFFERENCE (diferença) e CARTESIAN PRODUCT (produto cartesiano) (DATE,2004);

b) **Conjunto de operações projetadas para banco de dados relacionais:** SELECT (seleção), PROJECT (projeção) e JOIN (junção) (DATE,2004);

2.2.7.2 Heurística na otimização

Segundo Siberschartz, Korth, Sudarshan (1999) a heurística trabalha diretamente com a álgebra relacional com objetivo de converter as consultas para realizar as operações de seleção o mais breve possível.

O otimizador baseado em heurística realizar seleções antecedendo a projeção, pois nesse caso há grandes chances de obter-se a redução das relações (SILBERSCHATZ; KORTH; SUDARSHAN, 1999).

Em resumo, as heurísticas reestruturam uma representação preliminar de uma árvore de consulta, dessa forma as operações que reduzem o tanho dos resultados intermediários são executadas primeiro (SILBERSCHATZ; KORTH; SUDARSHAN, 1999).

2.2.8 Transações

Segundo Ramakrishnan, Gehrke (2008) uma transação é vista pelo SGBD como uma série ou lista de ações. As ações que podem ser executadas por uma transação incluem leituras e gravações de objetos de banco de dados.

Cada transação além de ler e gravar deve especificar como ação final ou a efetivação (incluir com sucesso) ou cancelamento (desfazer todas as ações executadas até o momento). As transações interagem uma com as outras apenas por meio de operações de leitura e gravação do banco de dados; por exemplo, elas não podem trocar mensagens. (RAMAKRISHNAN; GEHRKE, 2008).

De acordo com Siberschartz, Korth, Sudarshan (1999) um conjunto de várias operações no banco de dados é vista pelo usuário como uma única operação. Por exemplo, a transferência de fundos de uma conta corrente para uma poupança é uma operação única sob o ponto de vista do cliente, porém internamente no banco de dados ela envolve várias operações.

Uma transação atualiza vários itens de dados e geralmente é o resultado da execução de um programa de usuário escrito em uma linguagem de programação. Cada transação é delimitada por declarações que marcam seu início e fim, as operações que serão executados nessa transação são todas as invocadas ali entre o começo e o fim (SILBERSCHATZ; KORTH; SUDARSHAN, 1999).

De acordo com Siberschartz, Korth, Sudarshan (1999) para assegurar a integridade dos dados, é exigido que o sistema de banco de dados mantenha as seguintes propriedades das transações:

a) Atomicidade: Ou todas as operações da transação são executadas com sucesso no banco de dados ou nenhuma o será executada;

b) Consistência: A execução de uma transação isolada, ou seja, sem a execução concorrente de uma outra transação, preserva a consistência do banco de dados;

c) Isolamento: Sabendo que diversas transações podem ser executadas de forma concorrente o sistema deve garantir que, para cada par de transações T_i e T_j, T_j tenha terminado sua execução antes de T_i começar, ou que T_j tenha começado sua execução após T_i terminar. Assim, cada transação não toma conhecimento de outras transações concorrentes no sistema;

d) Durabilidade: Depois da transação ser finalizada com sucesso, as mudanças que ela fez no banco de dados persistem até mesmo se houver falhas no sistema.

2.2.9 Controle de concorrência

Uma das propriedades fundamentais de uma transação é o isolamento. Entretanto, quando várias transações são executadas concorrentemente no banco de dados, essa propriedade pode não ser mantida. Para garantir que essa propriedade seja preservada o SGBD precisa controlar a interação entre as transações concorrentes; esse controle é atingido por uma série de procedimentos chamados de esquemas de controle de concorrência. (SILBERSCHATZ; KORTH; SUDARSHAN, 2012)

De acordo com Siberschartz, Korth, Sudarshan (2012) existem uma variedade de esquemas (protocolos) de controle de concorrência entre eles estão, protocolo baseados em bloqueios, protocolo de bloqueios baseados em duas fases e protocolo baseado em grafo.

2.2.9.1 Protocolo baseados em bloqueio

Nesse esquema é exigido que os dados sejam acessados de maneira mutuamente exclusiva, ou seja, enquanto uma transação está acessando uma informação ou dado, nenhuma outra transação pode modificar esse dado. O método mais comum para atender esse requisito é permitir que uma transação acesse esse dado somente se estiver atualmente mantendo um bloqueio sobre o mesmo (SILBERSCHATZ; KORTH; SUDARSHAN, 2012).

2.2.9.2 Protocolo de bloqueio em duas fases

Segundo Siberschartz, Korth, Sudarshan (2012) esse esquema requer que cada transação emita solicitações de bloqueio e desbloqueio em duas fases:

a) Fase de crescimento: Uma transação pode obter bloqueios, mas não pode liberar qualquer bloqueio;

b) Fase de encolhimento: Uma transação pode liberar bloqueios mas não pode liberar novos bloqueios.

Inicialmente, uma transação encontra-se na fase de crescimento e adquire bloqueios conforme a necessidade. Quando a transação libera um bloqueio ela entra na fase de encolhimento e não pode emitir mais solicitações de bloqueio. (SILBERSCHATZ; KORTH; SUDARSHAN, 2012)

O bloqueio em duas fases não garante o surgimento de impasse (*deadlock*)

2.2.9.3 Protocolo baseado em grafo

Nesse protocolo precisamos de informações adicionais sobre como cada transação acessará o banco de dados. Existem vários modelos que nos dão as informações adicionais. O modelo mais simples exige conhecimento prévio sobre a ordem em que os itens do banco de dados serão acessados. (SILBERSCHATZ; KORTH; SUDARSHAN, 2012)

No protocolo baseado em grafo, a única instrução de bloqueio permitida é o bloqueio de modo exclusivo. Cada transação T_i pode bloquear um item de dados no máximo uma vez e precisa observar as seguintes regras:

a) O primeiro bloqueio por T_i pode ser sobre qualquer item de dados;

b) Subsequentemente, um item de dados A pode ser bloqueado por T_i somente se o pai de A estiver atualmente bloqueado por T_i;

c) Os itens de dados podem ser desbloqueados a qualquer momento;

d) Um item de dados que foi bloqueado e desbloqueado por T_i não pode ser bloqueado novamente por T_i;

2.2.10 Impasses

Segundo Elmasri, Ramez (2011) um impasse (*deadlock*) ocorre quando cada transação em um conjunto de duas ou mais transações está esperando por algum item que está bloqueado por alguma outra transação.

Considere o exemplo a seguir. A transação T_i define um bloqueio exclusivo sobre o dado A, T_j define um bloqueio exclusivo sobre B, T_i requisita um bloqueio exclusivo sobre B e é enfileirada, e T_j requisita um bloqueio exclusivo sobre A e é enfileirada. Agora, T_i está esperando que T_j libere seu bloqueio e T_j está esperando que T_i libere seu bloqueio. Esse ciclo de transações esperando que os bloqueios sejam liberados é chamado de impasse (*deadlock*). Podemos deduzir que essas duas transações não terão nenhum progresso. O SGBD deve evitar ou detectar as situações de impasse. (RAMAKRISHNAN; GEHRKE, 2008).

Se um impasse não for identificado pode ocorrer o travamento do SGBD onde uma transação fica esperando pela outra eternamente, até o serviço ficar indisponível. Os impasses podem também serem camuflados por um *timeout* de transação que pode ser configurado na maioria dos SGBDs, nesse caso não ocorre o travamento porém nenhuma das transações concorrentes são executadas e podem passar despercebidas, caso não haja um monitoramento do banco de dados. (ELMASRI; RAMEZ , 2011)

2.2.11 Índices

Segundo Rob, Coronel, (2011) um índice pode ser considerado uma estrutura ou arquivo associado a uma tabela, e tem como objetivo melhorar o desempenho de acesso à dados de uma ou mais linhas de uma tabela, dessa maneira o índice cria ponteiros para os dados armazenados nas colunas que devem ser especificas em sua criação.

Os índices podem ser utilizados de diferentes maneiras e levar a planos de execução mais rápidos do que qualquer outro que não utiliza índices (RAMAKRISHNAN; GEHRKE 2008).

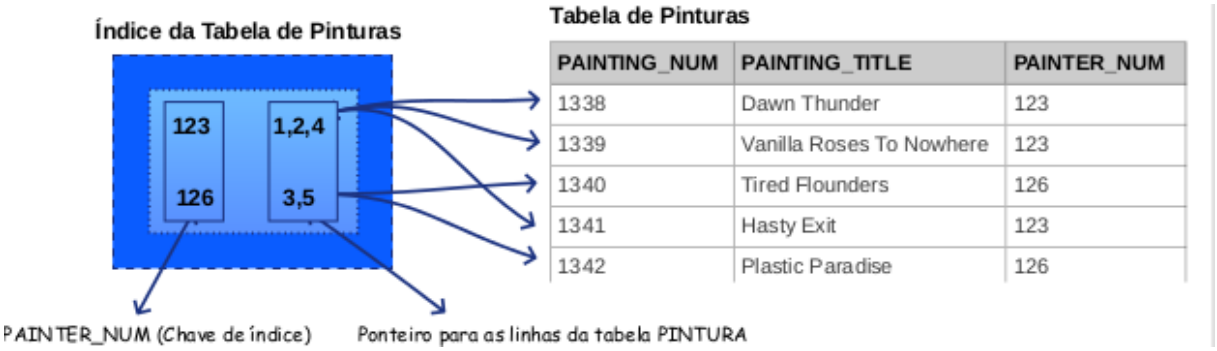
Um índice é uma disposição ordenada utilizada para acessar logicamente uma tabela. Considerando de um ponto de vista conceitual, é formado de uma chave de índice e de um conjunto de ponteiros, onde, a chave de índice é o ponto de referência do mesmo. Cada chave aponta para a localização dos dados identificados por ela (ROB; CORONEL, 2011).

Os índices executam um papel importante na estrutura de um SGBD, pois ao definir a chave primaria de uma tabela, o SGBD cria automaticamente um índice exclusivo para a(s) coluna(s) dessa chave (ROB; CORONEL, 2011).

Uma tabela pode ter vários índices, porém, cada um deles está associado a apenas uma tabela (ROB; CORONEL, 2011).

Por exemplo suponha que você queira procurar todas as pinturas criadas por um determinado pintor no banco de dados. Sem um índice, é necessário ler todas as linhas da tabela pintura e ver se o atributo correspondente ao código do pintor é do pintor solicitado. No entanto, se for criado um índice utilizando-se o código do pintor, basta procurar o valor adequado desse atributo no índice e encontrar os ponteiros correspondentes. Em termos conceituais, o índice se assemelha à representação ilustrada na figura 4 (ROB; CORONEL, 2011).

Figura 4 - Etapas no processamento de consultas no banco de dados



Fonte: Modificado de Rob, Coronel (2011).

2.2.12 PostgreSQL

Nessa seção serão abordados os principais conceitos envolvendo o PostgreSQL que será o sistema gerenciador de banco de dados inicialmente monitorado pela ferramenta proposta. Serão abordadas as limitações da ferramenta, características e informações sobre a coleta e visualização de estatísticas.

O PostgreSQL é um poderoso SGBD relacional *open source* com mais de 15 anos de desenvolvimento. É extremamente robusto e confiável, além de ser extremamente flexível e rico em recursos (BIAZUS, 2003).

O PostgreSQL derivou do projeto POSTGRES da universidade de Berkley, cuja última versão foi a 4.2, foi originalmente patrocinado pelo DARPA (Agência de Projetos de Pesquisa Avançada para Defesa), ARO (Departamento de Pesquisa Militar), NSF (Fundação Científica Nacional) (BIAZUS, 2003).

O desenvolvimento do projeto POSTGRES teve inicio em 1986 e em 1987 já estava operacional. A primeira versão lançada para o público externo foi em 1989 (BIAZUS, 2003).

Segundo Biazus (2003) o PostgreSQL pode ser considerado um SGBD objeto-relacional, por conter algumas características de orientação a objetos, como herança e tipos personalizados.

O PostgreSQL é compatível com diversos sistemas operacionais entre eles podemos citar: Windows, Linux, MacOS, Solaris. Além disso o SGBD fornece suporte a diversas plataformas e linguagens de programação como: Java, C, Python, PHP e Ruby (MILANI, 2008)

Segundo Milani (2008) o PostgreSQL encontra-se em uma versão estável e confiável, e disponibiliza os principais recursos existentes nos sistemas gerenciadores de banco de dados pagos do mercado e com capacidade para suprir pequenas, médias e grandes aplicações.

2.2.12.1 Características do PostgreSQL

A seguir, algumas características existentes no PostgreSQL;

a) Suporte a transações: O PostgreSQL possui suporte a operações ACID (Atomicidade, Consistência, Isolamento e Durabilidade). Cada uma destas propriedades garante a qualidade dos serviços disponibilizados pelo SGBD (MILANI, 2008);

b) Alta disponibilidade: Visando expansão da capacidade de processamento o PostgreSQL possibilita que atue como um *cluster* de informações (MILANI, 2008);

c) Multithreads: O PostgreSQL possibilita mais de uma conexão com o banco de dados, por meio de recurso de *multithereads*, ou seja, permite que mais de uma pessoa possa acessar a mesma informação simultaneamente sem ocasionar atrasos ou filas de acesso (MILANI, 2008);

d) Segurança e criptografia: O PostgreSQL possui suporte nativo a SSL, possibilitando conexões seguras através destes canais para trafegar informações consideradas sigilosas (MILANI, 2008);

e) Sql: Adota os padrões ANSI SQL na implementação de suas funcionalidades (MILANI, 2008);

f) Incorporável em aplicações gratuitamente: Por utilizar a licença BSD (*Berkeley Software Distribution*), pode ser livremente incorporado em aplicações pessoais e/ou comerciais (MILANI, 2008);

g) Capacidade de armazenamento: O PostgreSQL suporta de maneira eficiente e confiável grandes tamanhos de informações não tendo limite máximo para um banco de dados.

2.2.12.2 Limitações

Segundo Smanioto (2007) o PostgreSQL é bem flexível no que diz respeito a processador e plataforma. O SGBD pode ser instalado de forma instável em qualquer tipo de hardware com diversos tipos de sistemas operacionais entre eles Linux, UNIX (AIX, BSD, HP-UX, SGI ARIX, MAC OS X, Solaris, Tru64), e Windows. A tabela 1 apresenta algumas limitações.

Tabela 1 - Limitações do PostgreSQL.

Limite	Valor
Tamanho máximo do banco de dados	Ilimitado
Tamanho máximo de uma tabela	32 TB
Tamanho máximo de uma linha de tabela	1.6 TB
Tamanho máximo de um registro	1 GB
Quantidade de linhas por tabela	Ilimitado
Quantidade de colunas por tabela	250 à 1600. Depende do tipo de coluna
Quantidade de index por tabela	Ilimitado

Fonte: Modificado de Smanioto (2007).

2.2.12.3 Catálogo do PostgreSQL

Os catálogos do SGBD PostgreSQL são tabelas como qualquer outra do banco de dados. Estas tabelas podem ser removidas e recriadas, podem ser adicionadas novas colunas, podem ser inseridos e atualizados valores, porém nada disso é recomendável pois normalmente esses dados são inseridos e atualizados automaticamente pelo SGBD. (POSTGRESQL, 2015)

2.2.12.4 Coletor de estatísticas do PostgreSQL

O coletor de estatísticas do PostgreSQL é um subsistema do SGBD que coleta informações sobre as atividades do servidor que estejam relacionadas aos dados armazenados no banco de dados. Com as estatísticas coletadas podemos analisar por exemplo acessos em tabelas e índices (POSTGRESQL, 2015).

2.2.12.4.1 Configuração para coleta de estatísticas

A coleta de estatísticas no PostgreSQL pode deixar o servidor sobrecarregado, dessa forma o SGBD pode ser configurado para coletar ou não coletar informações. Isto é controlado por parâmetros de configuração que são normalmente definidos no arquivo postgresql.conf (POSTGRESQL, 2015).

Conforme PostgreSQL (2015), segue alguns parâmetros de configuração que controlam a coleta de estatísticas:

a) *track_activities:* Este parâmetro quando ativado permite o monitoramento do comando atual que está sendo executado por qualquer processo no banco de dados;

b) *track_counts:* Este parâmetro quando ativado coleta estatísticas sobre acessos em tabelas e índices;

c) *track_functions:* Este parâmetro quando ativado permite o acompanhamento do uso de funções definidas pelo usuário;

d) *track_io_timing:* Este parâmetro quando ativado permite o monitoramento de tempo de leitura e escrita em um bloco de transação.

Geralmente, esses parâmetros são definidos no arquivo postgresql.conf para que tenham efeito em todos os processos do servidor, mas é possível ativá-los ou desativá-los em sessões individuais usando o comando *SET,* por motivos de segurança somente usuários com privilégios administrativos podem alterar essas configurações utilizando o comando *SET* (POSTGRESQL, 2015).

2.2.12.4.2 Visualizando as estatísticas coletadas

Segundo PostgreSQL (2015) existem várias visões que permitem o resultado das estatísticas coletadas, dessa forma pode-se construir visualizações customizadas usando os recursos nativos do SGBD PostgreSQL.

Conforme PostgreSQL para construir visualizações customizadas podem ser utilizadas as visões nativas listadas na figura 5 abaixo:

Figura 5 - Quadro de visões de estatísticas nativas do PostgreSQL.

Nome	**Descrição**
pg_stat_activity	Retorna uma linha por processo que está sendo executado no servidor, mostrando informações relacionadas a atividade atual desse processo.
pg_stat_bgwriter	Retorna uma linha única, mostrando estatísticas sobre a atividade do processo em *background* de escrita.
pg_stat_database	Retorna uma linha por banco de dados, mostrando as estatísticas de todo o banco de dados.
pg_stat_all_tables	Retorna uma linha para cada tabela do banco de dados atual, mostrando estatísticas sobre acessos a essa tabela em específico.
pg_stat_sys_tables	Retorna o mesmo que a *pg_stat_all_tables*, exceto que somente são mostradas as tabelas do sistema.
pg_stat_user_tables	Retorna o mesmo que a *pg_stat_all_tables*, exceto que somente são mostradas as tabelas de usuário.
pg_stat_xact_all_tables	Tem um resultado semelhantes a *pg_stat_all_tables*, mas conta as medidas tomadas até agora dentro da transação corrente (que ainda não estão incluídas no *pg_stat_all_tables*).
pg_stat_xact_sys_tables	Tem o mesmo resultado que a *pg_stat_xact_all_tables*, exceto que somente são mostradas as tabelas do sistema.
pg_stat_xact_user_tables	Tem o mesmo resultado que a *pg_stat_xact_all_tables*, exceto que somente são mostradas as tabelas de usuário.
pg_stat_all_indexes	Retorna uma linha para cada índice do banco de dados atual, mostrando estatísticas sobre acessos a esse índice em específico.
pg_stat_sys_indexes	Tem o mesmo resultado que a *pg_stat_all_indexes*, exceto que somente os índices das tabelas do sistema são mostradas.
pg_stat_user_indexes	Tem o mesmo resultado que a *pg_stat_all_indexes*, exceto que somente índices das tabelas do usuário são mostrados.
pg_statio_all_tables	Retorna uma linha para cada tabela no banco de dados atual, mostrando estatísticas sobre I / O da tabela específica.
pg_statio_sys_tables	Tem o mesmo resultado que a *pg_statio_all_tables*, exceto que somente são mostradas as tabelas do sistema.
pg_statio_user_tables	Tem o mesmo resultado que a *pg_statio_all_tables*, exceto que somente são mostradas as tabelas de usuário.
pg_statio_all_indexes	Retorna uma linha para cada índice no banco de dados atual, mostrando estatísticas sobre I / O do índice em específico.
pg_statio_sys_indexes	Tem o mesmo resultado que *pg_statio_all_indexes*, exceto que somente os índices das tabelas do sistema são mostradas.
pg_statio_user_indexes	Tem o mesmo resultado que *pg_statio_all_indexes*, exceto

	que somente índices das tabelas do usuário são mostrados.
pg_statio_all_sequences	Retorna uma linha para cada sequência no banco de dados atual, mostrando estatísticas sobre I/O da sequência em específico.
pg_statio_sys_sequences	Retorna o mesmo que *pg_statio_all_sequences*, exceto que somente sequências de sistema são mostradas.
pg_statio_user_sequences	Retorna o mesmo que *pg_statio_all_sequences*, exceto que somente sequências de usuários são mostrados.
pg_stat_user_functions	Retorna uma linha pra cada função, mostrando estatísticas sobre suas execuções.
pg_stat_xact_user_functions	Resultado semelhante a *pg_stat_user_functions*, mas conta apenas chamadas enquanto a transação etá em andamento.
pg_stat_replication	Retorna uma linha por processo remetente WAL, mostrando estatísticas sobre a replicação para o servidor de espera.
pg_stat_database_conflicts	Retorna linha por banco de dados, mostrando as estatísticas de todo o banco de dados sobre as consultas cancelas devido à conflitos.

Fonte:Modificado de PostgreSQL (2015).

Se forem usadas as estatísticas para monitorar os processos que estão sendo executados, é importante perceber que os dados não são atualizadas instantaneamente, as informações estatísticas são atualizadas após o término das transações, dessa forma as transações que ainda estão em processo não afetam os totais exibidos. Quando um processo do banco de dados solicita as estatísticas coletadas, primeiramente ele busca o relatório mais recente emitido pelo coletor e em seguida será utilizado até o final da transação corrente, este recurso permite que você execute várias consultas sobre as estatísticas sem se preocupar com números mudando continuamente com transações que ainda não terminaram (POSTGRESQL, 2015).

A transação também tem suas próprias estatísticas e podem ser visualizadas utilizando as visões *pg_stat_xact_all_tables*, *pg_stat_xact_sys_tables*, *pg_stat_xact_user_tables* e *pg_stat_xact_user_functions* (POSTGRESQL, 2015).

2.2.12.5 O Comando EXPLAIN

O comando EXPLAIN mostra o plano de execução gerado pelo planejador do PostgreSQL para o comando fornecido. O plano de execução mostra como as tabelas referenciadas pelo comando serão varridas, por uma varredura sequencial simples ou uma

varredura por um índice, etc. Do que é mostrado, a parte mais importante é o custo estimado de execução do comando, que é a estimativa feita pelo planejador de quanto tempo vai demorar para executar o comando (POSTGRESQL, 2015).

Para visualizar o plano que o planejador cria para a consulta pode ser utilizado o comando EXPLAIN + consulta (POSTGRESQL, 2015).

2.2.12.6 O comando VACUUM

O comando VACUUM recupera o armazenamento ocupado por tuplas mortas. Em condições normais o PostgreSQL mantém tuplas obsoletas para ter mais agilidade na execução de um comando que atualizam ou excluem informações do banco, essas tuplas só são fisicamente removidas quando o comando VACCUM é executado. Portanto o mesmo deve ser periodicamente executado, especialmente em tabelas frequentemente atualizadas (POSTGRESQL, 2015).

Segundo PostgreSQL (2015) o comando VACUUM pode ser executado agrupado com alguns parâmetros, que serão listado abaixo:

a) *FULL:* Faz uma limpeza completa em tuplas mortas e pode recuperar mais espaço, mas leva muito mais tempo e bloqueia as tabelas. Esse método também requer espaço em disco extra, uma vez que ele faz uma cópia das tabelas e só libera a cópia antiga quando a operação estiver concluída;

b) *VERBOSE* : Imprime um relatório detalhado da atividade de limpeza de cada tabela;

c) *ANALYZE:* Atualiza as estatísticas utilizadas pelo planejador para determinar a maneira mais eficiente de executar uma consulta.

2.2.12.7 Parâmetros de configuração que podem melhoram a performance

Segundo Berkus (2005), alguns parâmetros do SGBD PostgreSQL se ajustados de maneira correta podem refletir em aumento de performance. Segue na figura 6 alguns desses parâmetros:

Figura 6 - Quadro de parâmetros que podem melhorar a performance do PostgreSQL.

Nome	**Descrição**
shared_buffers	Bloco de memória dedicado ao PostgreSQL utilizado para as operações ativas.
work_mem	Bloco de memória não compartilhada, sendo alocada para cada operação, esta configuração coloca um teto na quantidade de memória que uma única operação ocupar antes de ser forçada para o disco.
maintenance_work_mem	Bloco de memória utilizada pelo PostgreSQL para executar os comandos VACUUM, ANALYZE, CREATE INDEX, e adição de chaves estrangeiras.
checkpoint_segments	Bloco de memória que define o tamanho do cache do log de transações para operações de escrita.
effective_cache_size	Bloco de memória que define ao planejador de consultas o maior objeto do banco de dados que pode se esperar ser cacheado.

Fonte: Modificado de Berkus (2005).

3 Ferramentas de análise e monitoramento de estatísticas

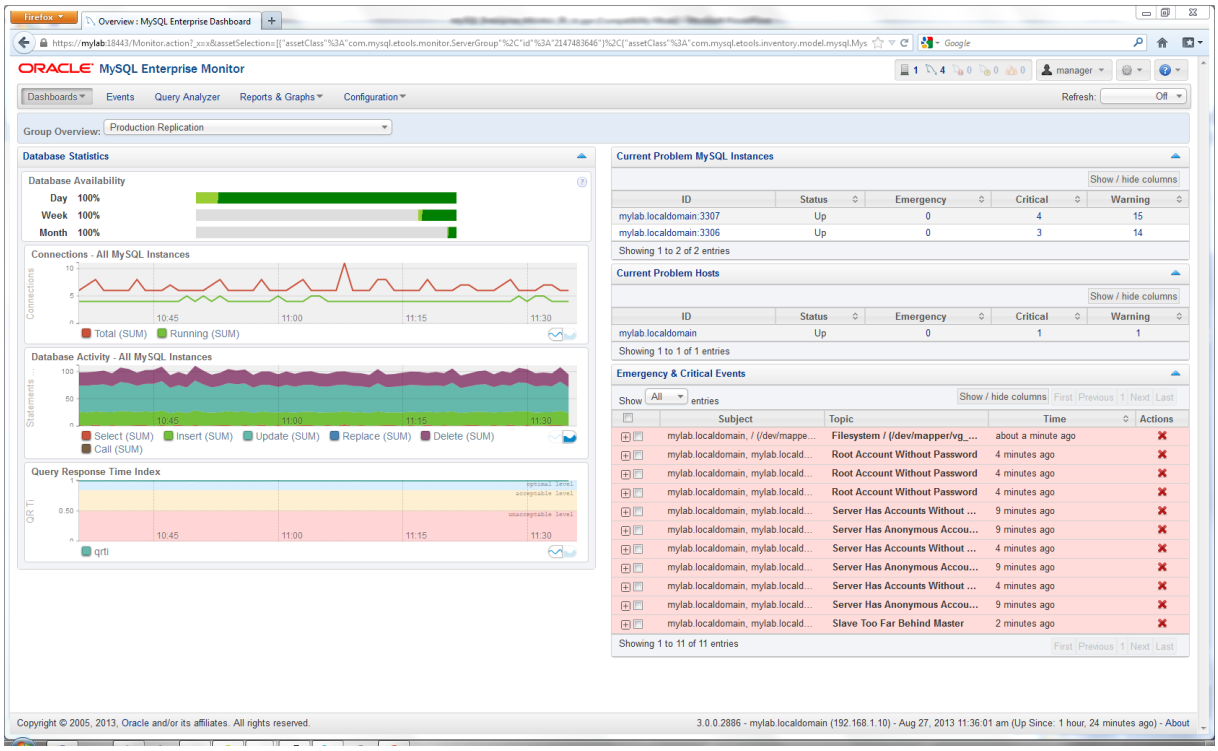
Nessa sessão serão apresentadas algumas ferramentas já existentes para análise e monitoramento de estatísticas para bases de dados que utilizam SGBDs.

3.1 MySQL Enterprise Monitor

Segundo Oracle (2015) o MySQL Enterprise Monitor é um sistema de monitoramento para base de dados que utilizam o SGBD MySQL, o sistema notifica sobre problemas e aconselha como corrigi-los.

O MySQL Enterprise Monitor monitora continuamente consultas e métricas de desempenho de uma ou mais base de dados que utiliza o SGBD MySQL, o sistema alerta o DBA sobre desvios significativo das métricas e recomenda alterações em configurações para manter o desempenho ORACLE (2015). Segue na figura 7 um *screenshot* da ferramenta MySQL Enterprise Monitor.

Figura 7 - Screenshot da ferramenta MySQL Enterprise Monitor.



Fonte: Oracle (2015).

3.2 Licença

O programa MySQL possui licença dupla. Os usuários podem optar por usar o software MySQL como um produto *open source* sob os termos da GNU (General Public License) ou podem comprar uma licença comercial padrão da Oracle.

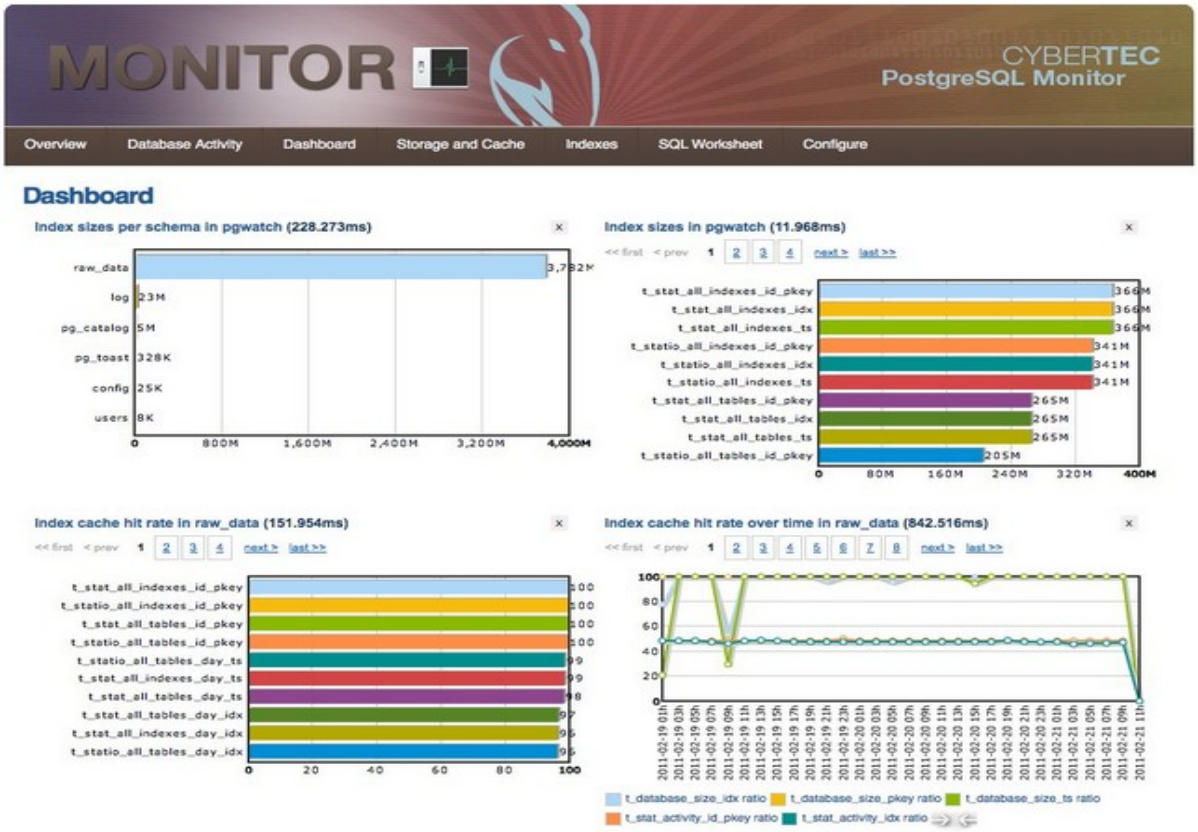
3.3 Pgwatch

O Pgwatch é uma ferramenta para monitorar bancos de dados PostgreSQL que à partir de sua versão 2.0 se tornou uma ferramenta *open source*. A versão 2.0 encontra-se em desenvolvimento e pode ser adquirida através do site da mantenedora. (CYBERTEC, 2015)

A ferramenta contém um *daemon* que coleta periodicamente estatísticas do sistema de banco de dados, as estatísticas são armazenadas em uma base de dados central do Pgwatch para uma posterior análise (CYBERTEC, 2015).

Atualmente o Pgwatch em sua versão de desenvolvimento suporta as versões 9.0 e 9.1 do PostgreSQL, não contemplando as versões mais atuais como a 9.3. (CYBERTEC, 2015). Segue na figura 8 um *screenshot* da ferramenta Pgwatch.

Figura 8 - Screenshot da ferramenta Pgwatch



Fonte: Cybertec (2015)

3.3.1 Licença

O Pgwatch está licenciado sob a Licença Creative Commons e pode ser baixado gratuitamente. (CYBERTEC, 2015)

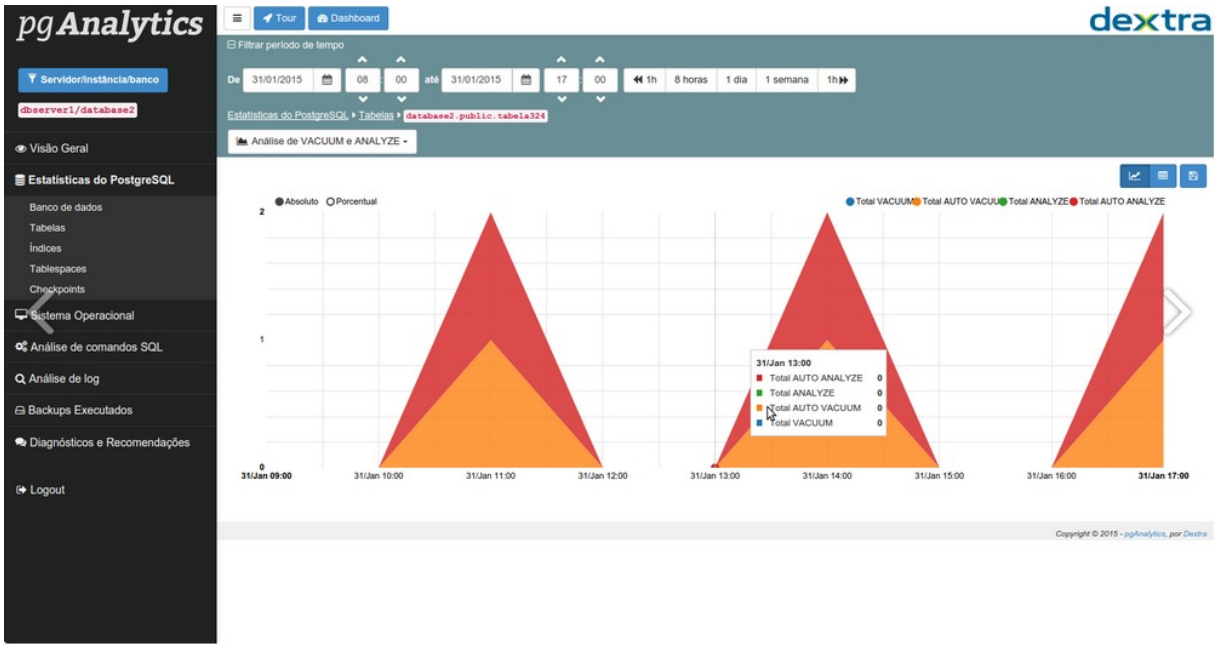
3.4 Pganalytics

O Pganalytics é uma ferramenta SaaS (Software as a Service) desenvolvida pela Dextra com o objetivo de coleta e exibição de dados estatísticos de bases de dados que utilizam o SGBD PostgreSQL (DEXTRA, 2015).

A ferramenta disponibiliza diagnósticos e recomendações que auxiliam na administração do banco de dados, a ferramenta conta também com com um recurso baseado em alertas avisando o usuário sobre incidentes que podem afetar o funcionamento do SGBD. (DEXTRA, 2015)

O Pganalytics disponibiliza para seus clientes um agente não intrusivo que deve ser instalado no servidor de banco de dados que pode ser Linux ou Windows. O agente coleta estatísticas e sincroniza as informações com um servidor mantido pela Dextra. Para utilizar a ferramenta é necessário criar uma conta e pagar uma mensalidade que varia de acordo com o plano de monitoramento escolhido. (DEXTRA, 2015) Segue na figura 9 um *screenshot* da ferramenta Pganalytics.

Figura 9 - Screenshot da ferramenta Pganalytics



Fonte: Dextra (2015)

3.5 Cedrus

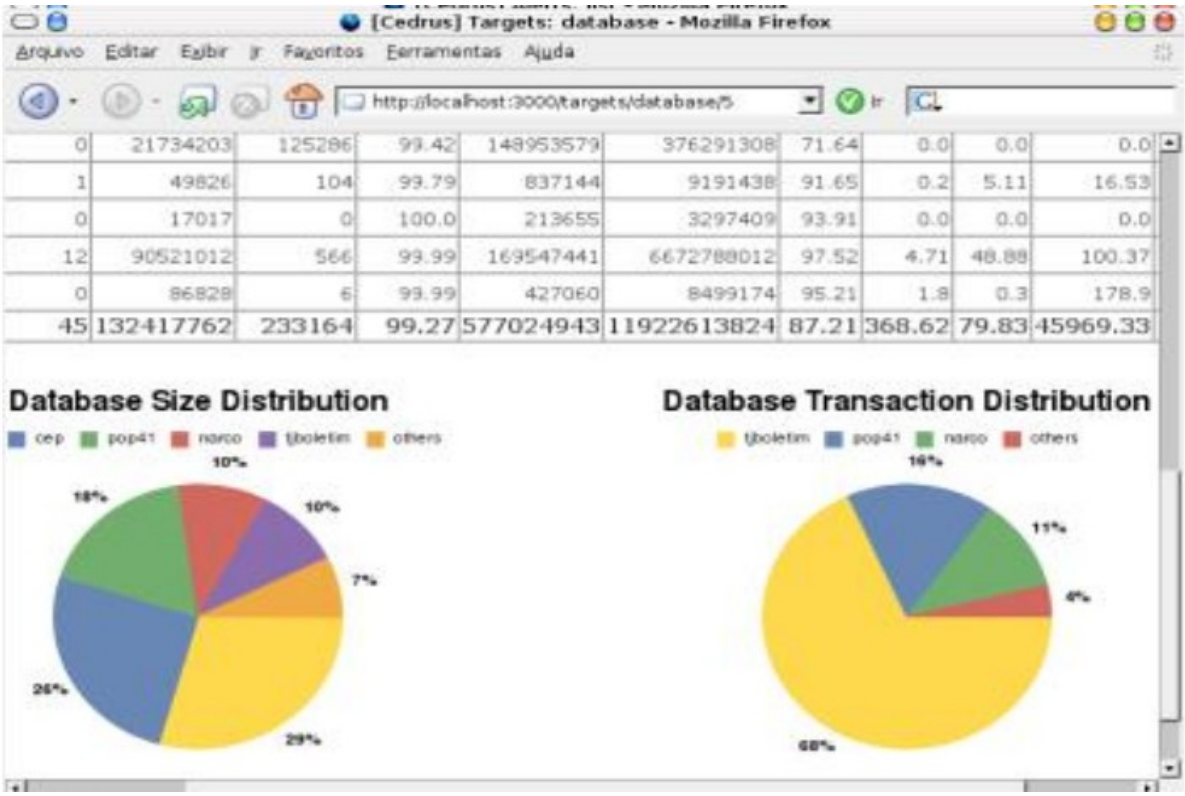
O Cedrus é uma ferramenta de administração e monitoramento gráfica para PostgreSQL desenvolvida por Rodrigo Hjort. Inspirado no Enterprise Manager da Oracle. A ferramenta coleta informações úteis sobre uma ou mais instâncias do PostgreSQL e do sistema operacional onde a base de dados está instalada. A coleta de dados é feita através de um agente instalado no servidor de banco de dados, o coletor envia para a base de dados do

Cedrus todas as informações obtidas, sendo assim é possível ter uma base histórica de todas as informações coletadas (CEDRUS, 2006).

Através da interface gráfica do sistema é possível emitir relatórios e visualizar gráficos sobre informações do sistema operacional e estatísticas coletadas do banco de dados (CEDRUS, 2006) .

O Cedrus foi desenvolvido por Rodrigo Hjort em 2006 e não teve continuidade, não suportando versões mais atuais do PostgreSQL (CEDRUS, 2006). Segue na figura 10 um *screenshot* da ferramenta Cedrus.

Figura 10 - Screenshot da ferramenta Cedrus



Fonte: Cedrus (2006)

3.5.1 Licença

GNU General Public License version 2.0 (GPLv2). (CEDRUS, 2006)

3.6 Comparativo entre as ferramentas estudadas

A figura 11 apresenta um comparativo entre as ferramentas estudadas, levando em consideração algumas características consideradas relevantes como:

a) Administração de múltiplos servidores: Identifica se a ferramenta possibilita monitorar mais de um servidor de banco de dados;

b) Open source: Identifica se a ferramenta é de livre distribuição;

c) Compatibilidade com versões atuais de SGBDs: Identifica se a ferramenta pode ser utilizada com as versões mais atuais de SGBDs;

d) Alarmísticas: Identifica se a ferramenta possuí funcionalidades que avisam ao usuário de possíveis problemas com o SGBD;

e) Gráficos pre definidos: Identifica se a ferramenta já possui algum gráfico de monitoramento pré definido;

f) Multiplataforma: Identifica se a ferramenta pode ser utilizada quando o SGBD está instalado em diferentes sistemas operacionais;

g) Geração de relatórios: Informa se a ferramenta possibilita a geração de relatórios das informações coletadas.

Figura 11 - Quadro comparativo entre ferramentas de análise e monitoramento para PostgreSQL.

Características	**MySQL Enterprise Monitor**	**Pgwatch**	**Pganalytics**	**Cedrus**
Administração de múltiplos servidores	Sim	Sim	Sim	Sim
Open source	Sim	Sim	Não	Sim
Compatibilidade com versões atuais de SGBDs	Sim	Não	Sim	Não
Alarmísticas	Sim	Não	Sim	Sim
Gráficos pre definidos	Sim	Sim	Sim	Sim
Multiplataforma	Sim	Sim	Sim	Não
Geração de relatórios	Sim	Sim	Sim	Sim

Fonte: Elaborado pelo autor.

4 DESCRIÇÃO DA PROPOSTA

Este capítulo tem como objetivo apresentar os artefatos, documentos e técnicas que serão utilizados para o desenvolvimento da ferramenta proposta.

4.1 Metodologia

Nesta seção será apresentado o enquadramento metodológico utilizado neste estudo, com finalidade de atender aos objetivos propostos.

De acordo com Gil (2002) a pesquisa tem um caráter pragmático, é um "processo formal e sistemático de desenvolvimento do método científico. O objetivo fundamental da pesquisa é descobrir para os problemas utilizando procedimentos científicos".

Uma pesquisa exploratória, tem como objetivo aumentar o conhecimento do pesquisador sobre um determinado problema. Ela pode ser composta por uma revisão de literatura, entrevistas, entre outros (GIL, 2002).

O presente trabalho, buscará construir uma ferramenta web de coleta e análise de estatísticas para o sistema gerenciador de banco de dados PostgreSQL. Para isso, serão realizados estudos sobre os temas relacionados, com a finalidade de desenvolver uma ferramenta em conformidade com os conceitos já existentes. Desta forma, este trabalho se caracteriza conforme Santos (1999), de acordo com o seu objetivo, como pesquisa exploratória.

4.2 Visão geral

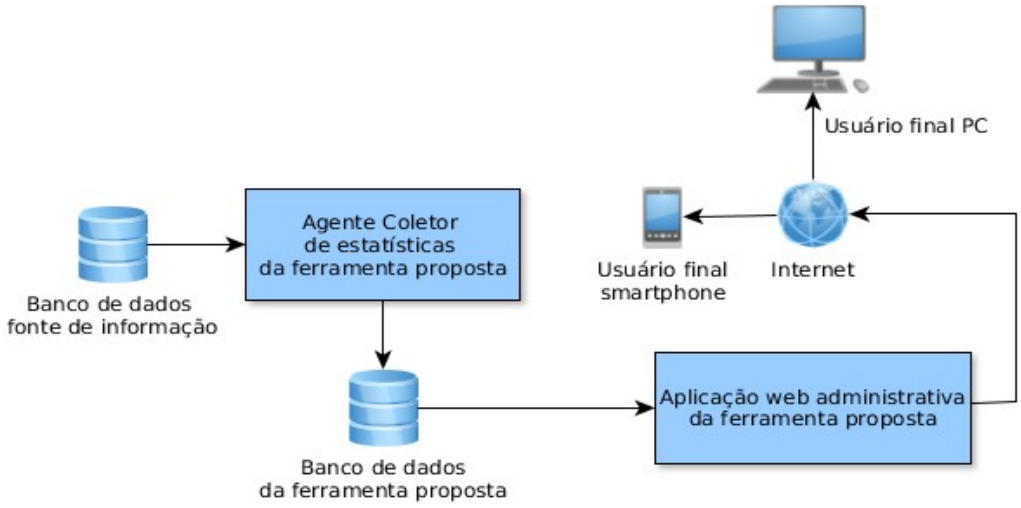
A proposta tem como objetivo a criação de uma plataforma web, *open source* que possibilite ao usuário DBA extrair e visualizar estatísticas do software PostgreSQL e futuramente de qualquer SGBD, possibilitando o monitoramento e a tomada de ações para garantir a performance da base de dados, sugerindo a criação de índices, alterações em configurações do SGBD e criação de alarmísticas avisando de possíveis problemas.

O projeto, possibilitará apenas o monitoramento do SGBD PostgreSQL como um módulo da ferramenta, dessa maneira, possibilita a expansão para atender outros SGBDs em trabalhos futuros.

Na figura 12 pode ser visualizada a arquitetura da ferramenta que está sendo proposta.

A ferramenta contará com um agente coletor que irá se comunicar com a base de dados a ser monitorada e o banco de dados da ferramenta que será desenvolvida. O usuário irá configurar o sistema e alterar configurações do agente através de uma interface web que poderá ser disponibilizada na internet e acessada por um computador ou um *smartphone.*

Figura 12 - Arquitetura da ferramenta proposta



Fonte: Elaborado pelo autor.

4.3 Tecnologias que serão utilizadas

Para o desenvolvimento da ferramenta proposta, serão utilizadas diversas tecnologias são elas:

a) PHP: (*Hypertext Preprocessor*) é uma linguagem de script *open source* de uso geral, muito utilizada, e adequada para o desenvolvimento web. O php[1] será utilizado para desenvolver o agente coletor e os processamentos relacionados a ferramenta web; (PHP, 2015)

b) Apache HTTP Server: É um software de código aberto que tem como objetivo disponibilizar páginas HTML à web. O Apache[2] será utilizado como servidor web para os usuários terem acesso a ferramenta proposta; (Apache, 2015)

c) PostgreSQL: É um SGBD (Sistema Gerenciador de Banco de Dados) objeto-relacional de código aberto extremamente robusto e confiável, O PostgreSQL[3] será utilizado para armazenar as estatísticas coletadas, usuários que irão acessar a ferramenta e configurações do sistema; (PostgreSQL, 2015)

d) HTML: (*HyperText Markup Language*) é uma linguagem de marcação da internet, utilizada para desenvolver as páginas na web. O HTML[4] será utilizado para criar a interface da plataforma; (HTML, 2015)

e) JavaScript: É uma linguagem de programação do lado cliente processada pelo próprio navegador. O JavaScript[5] será utilizado para criar efeitos nas interfaces da ferramenta para proporcionar uma maior interatividade; (JAVASCRIPT, 2015)

f) Bootstrap: É um *framework* para facilitar a criação de sites ou páginas web (responsivo) O Bootstrap[6] será utilizado para deixar a ferramenta mais intuitiva e utilizavel em qualquer dispositivo que tenha acessoa a web. (BOOTSTRAP, 2015);

[1]http://www.php.net
[2]http://httpd.apache.org/
[3]http://www.postgresql.org/
[4]http://www.w3.org/html/
[5]https://developer.mozilla.org/en-US/docs/Web/JavaScript
[6]http://getbootstrap.com/

g) Morris: É uma biblioteca leve que usa jQuery e para tornar fácil desenhar gráficos simples. O Morris[7] será utilizado para representar graficamente as estatísticas coletadas.

4.4 Levantamento de requisitos

Para que a ferramenta proposta atinja os objetivos desejados é necessário que atendam determinados requisitos. Os requisitos são divididos em funcionais e não funcionais e podem ser visualizados nas sessões seguintes.

4.4.1 Requisitos funcionais

A seguir, serão listados os principais requisitos funcionais que devem ser atendidos pela ferramenta proposta:

a) Configurar a ferramenta: Através desse requisito o usuário DBA poderá configurar a ferramenta que está sendo proposta, deverá ser possível alterar informações como a conexão com a base de dados da ferramenta e a frequência com que os processos que estão programados para o agente serão executados.

a) Gerenciar usuários: A ferramenta deve disponibilizar uma interface que permita o gerenciamento de usuários, ou seja, inserir, editar e excluir usuários que poderão utilizar a ferramenta. No gerenciamento de usuários deverá ser possível também informar o e-mail do usuário para que possa receber alertas que estejam configurados.

b) Visualizar estatísticas: Através desse requisito o o usuário DBA poderá visualizar todas as estatísticas coletadas pelo agente de forma gráfica, permitindo uma melhor análise e facilitando a decisão da ação que será tomada para melhorar o desempenho.

c) Gerenciar estatísticas: Através desse requisito o usuário DBA poderá configurar a frequência com que as estatísticas serão coletadas, ativá-la ou desativá-la e dar permissão para que seja exibida no painel principal.

[7]http://morrisjs.github.io/morris.js/

d) Agendar tarefas: Através desse requisito o usuário DBA poderá agendar tarefas para serem executadas em horários que o servidor tem menos acesso. Essa funcionalidade será útil quando for preciso executar manutenções no banco de dados que envolvem um grande volume de dados.

e) Gerenciar alertas: Através desse requisito o usuário DBA poderá configurar alertas baseando-se nas estatísticas coletadas, ou seja, se o resultado da estatística não for o esperado pelo DBA um e-mail será enviado para ele informando do ocorrido para que tome as devidas providencias.

f) Monitorar processos em execução: Através desse requisito o usuário DBA poderá monitorar os processos que estão em execução no banco de dados, a partir desses processos poderá verificar o plano de execução de consultas e estatísticas das tabelas envolvidas.

4.4.2 Requisitos não funcionais

A seguir, serão listados os principais requisitos não funcionais que devem ser atendidos pela ferramenta proposta:

a) Ser desenvolvido na linguagem PHP: Este requisito define qual a linguagem a ser utilizada na escrita do programa. Como a ferramenta que será desenvolvida será web foi optado por utilizar PHP por ser um linguagem de desenvolvimento web *open source*;

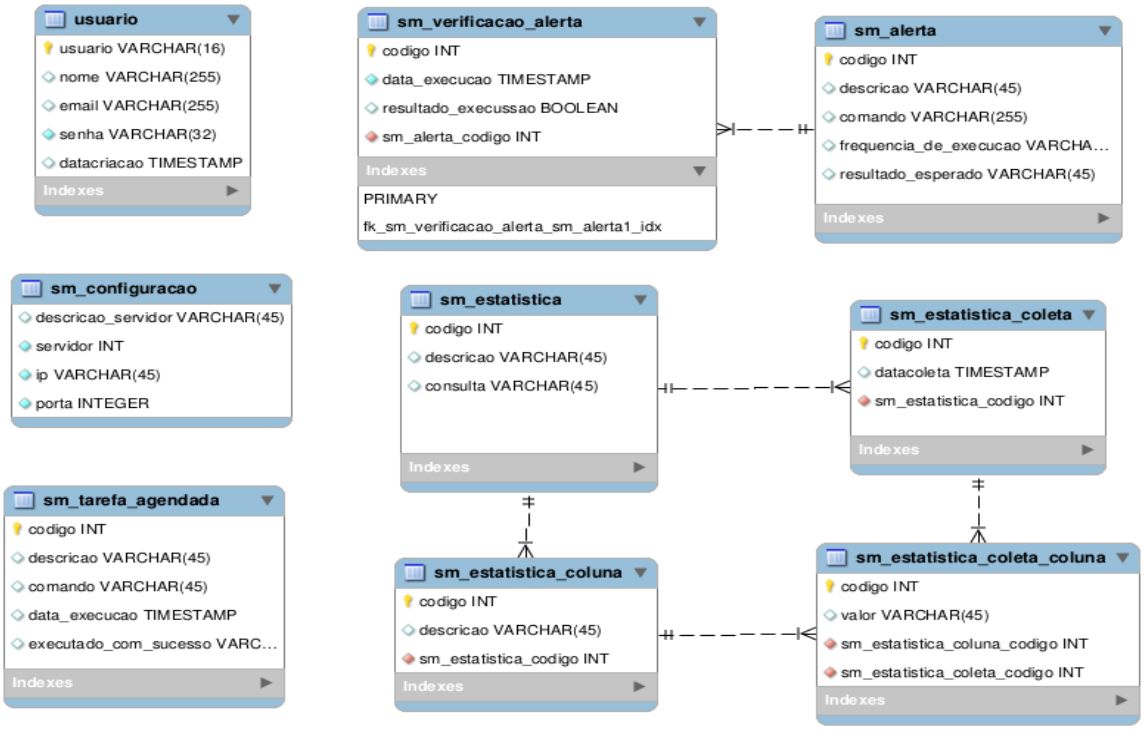
b) Utilizar SGBD PostgreSQL: Este requisito define o sistema gerenciador de banco de dados que será utilizado para armazenar os dados de configurações, usuários e estatísticas coletadas. Será utilizado o PostgreSQL por ser um SGBD robusto *open source*;

c) Estar disponível 24h: Através deste requisito, é desejável que a ferramenta seja acessível a qualquer momento.

4.5 Modelagem do banco de dados

Através dos requisitos levantados foram identificados os elementos necessários para definir o modelo do banco de dados da ferramenta proposta. Este modelo é apresentado na figura 13 e tem como objetivo permitir o armazenamento de forma genérica das estatísticas de qualquer SGBD.

Figura 13 - Modelo do banco de dados



Fonte: Elaborado pelo autor.

4.6 Agente

O agente será a parte da ferramenta responsável pela coleta das estatísticas da base de dados, execução de tarefas agendadas, captura de comandos que estão sendo executados e registro de alertas que devem ser enviados para o DBA.

Segue abaixo algumas estatísticas que serão coletadas e suas respectivas consultas.

a) Tabelas com poucas pesquisas por índices: Utilizando essa estatística o DBA poderá identificar através de um fator percentual, tabelas que tiveram poucas consultas que utilizaram algum índice, isso pode caracterizar a necessidade de criar novos índices

para a tabela. Na figura 14 pode ser visualizada a consulta utilizada para obter essa estatística no SGBD PostgreSQL.

Figura 14 - Consulta que obtém as tabelas com poucas pesquisas por índice

```
  SELECT relname,
         idx_scan::numeric / (seq_scan + idx_scan)::numeric * 100
as idx_percent
    FROM pg_stat_user_tables
   WHERE seq_scan + idx_scan > 0
ORDER BY idx_percent, seq_scan DESC;
```

Fonte: Elaborado pelo autor.

b) Tamanho de tabelas em disco: Através dessa estatística coletada do catálogo do PostgreSQL o usuário DBA poderá identificar o tamanho de cada tabela da base de dados, dessa maneira pode identificar a necessidade de armazená-la em um outro disco ou considerar uma mudança de estrutura na base de dados. Na figura 15 pode ser visualizada a consulta utilizada para obter essa estatística no SGBD PostgreSQL.

Figura 15 - Consulta que obtém o tamanho de tabelas em disco

```
SELECT pg_size_pretty(pg_relation_size('nome_tabela'));
```

Fonte: Elaborado pelo autor.

c) Tamanho da tabela em disco levando em consideração seus índices: Por meio dessa estatística o DBA poderá identificar o espaço total ocupado por uma tabela no banco de dados levando em consideração seus índices. Caso a tabela esteja ocupando muito espaço o DBA pode considerar a possibilidade de deletar índices não utilizados da tabela que está sendo monitorada. Na figura 16 pode ser visualizada a consulta utilizada para obter essa estatística no SGBD PostgreSQL.

Figura 16 - Consulta que obtém o tamanho de tabela levando em consideração seus índices

```
SELECT pg_size_pretty(pg_total_relation_size('nome_tabela'));
```

Fonte: Elaborado pelo autor.

d) Tamanho da base de dados: Através dessa estatística o DBA poderá controlar o volume total de dados que está sendo armazenado. Caso a base de dados esteja ocupando muito espaço o administrador pode considerar a possibilidade de aumentar o espaço disponível. Na figura 17 pode ser visualizada a consulta utilizada para obter essa estatística no SGBD PostgreSQL.

Figura 17 - Consulta que obtém o tamanho de uma base de dados

```
SELECT pg_size_pretty(pg_database_size('nome_banco_de_dados'));
```

Fonte: Elaborado pelo autor.

e) Aproveitamento do cache por tabela: Por meio dessa estatística extraída do SGBD o DBA poderá monitorar o aproveitamento do uso do cache do PostgreSQL para cada tabela, com essa informação o administrador poderá regular o parâmetro *shared_buffers* que define a memória dedicada ao PostgreSQL. Na figura 18 pode ser visualizada a consulta utilizada para obter essa estatística no SGBD PostgreSQL.

Figura 18 - Consulta que obtém o aproveitamento do cache por tabela

```
SELECT relname,
       round(
       (heap_blks_hit + COALESCE(toast_blks_hit, 0))::numeric /
       (heap_blks_read + COALESCE(toast_blks_read, 0) +
        heap_blks_hit + COALESCE(toast_blks_hit,0))::numeric *
100, 3) as hit_ratio
  FROM pg_statio_user_tables
 WHERE (heap_blks_read + COALESCE(toast_blks_read, 0)) +
heap_blks_hit + COALESCE(toast_blks_hit) > 0
 ORDER BY 2 DESC ;
```

Fonte: Elaborado pelo autor.

f) Aproveitamento do cache do banco de dados: Utilizando essa estatística o DBA poderá monitorar o aproveitamento do uso do cache de toda a base de dados. Da mesma forma como a estatística anteriormente apresentada, com essa informação o administrador poderá regular o parâmetro *shared_buffers* que define a memória dedicada ao PostgreSQL. Na figura 19 pode ser visualizada a consulta utilizada para obter essa estatística no SGBD PostgreSQL.

Figura 19 - Consulta que obtém o total do uso de cache em uma base de dados

```
SELECT round(
       AVG(
       (heap_blks_hit + COALESCE(toast_blks_hit, 0))::numeric /
       (heap_blks_read + COALESCE(toast_blks_read, 0) +
        heap_blks_hit + COALESCE(toast_blks_hit,0))::numeric *
100), 3) as hit_ratio
  FROM pg_statio_user_tables
 WHERE (heap_blks_read + COALESCE(toast_blks_read, 0)) +
heap_blks_hit + COALESCE(toast_blks_hit) > 0;
```

Fonte: Elaborado pelo autor.

g) Índices obsoletos: Através dessa estatística o DBA poderá verificar todos os índices que não estão sendo utilizados na base de dados que está sendo monitorada, como os índices podem ocupar uma grande fatia de espaço, o administrador pode decidir por deletar os índices obsoletos. Na figura 20 pode ser visualizada a consulta utilizada para obter essa estatística no SGBD PostgreSQL.

Figura 20 - Consulta que obtém os índices que estão obsoletos

```
SELECT schemaname,
       relname,
       indexrelname
  FROM pg_stat_user_indexes
 WHERE idx_scan = 0;
```

Fonte: Elaborado pelo autor.

h) Processos em execução: Por meio dessa estatística o DBA poderá monitorar as tarefas que estão sendo executadas na base de dados. Com base nessa informação, o administrador pode identificar tarefas que estão sendo executadas a muito tempo e precisam ser interrompidas para não comprometer a disponibilidade do serviço. Na figura 21 pode ser visualizada a consulta utilizada para obter essa estatística no SGBD PostgreSQL.

Figura 21 - Consulta que obtém os processos que estão em execução na base de dados

```
SELECT pg_stat_get_backend_pid(s.backendid) AS procpid,
       pg_stat_get_backend_activity(s.backendid) AS
current_query
  FROM (SELECT pg_stat_get_backend_idset() AS backendid) AS s;
```

Fonte: Elaborado pelo autor.

4.7 Protótipo

Nesta seção será apresentado um esboço da interface da ferramenta proposta. Na figura 22 é apresentado um protótipo da tela principal do sistema, através dessa tela o usuário final poderá acessar as principais funcionalidades do sistema, entre elas estão:

a) Menu principal: A partir dessa interface o DBA poderá visualizar as estatísticas que estão sendo monitoradas;

b) Menu configurações: A partir dessa interface o DBA poderá configurar a base de dados que será monitorada;

c) Menu processos em execução: Interface que possibilita gerenciar todas as tarefas que estão sendo executadas no banco de dados;

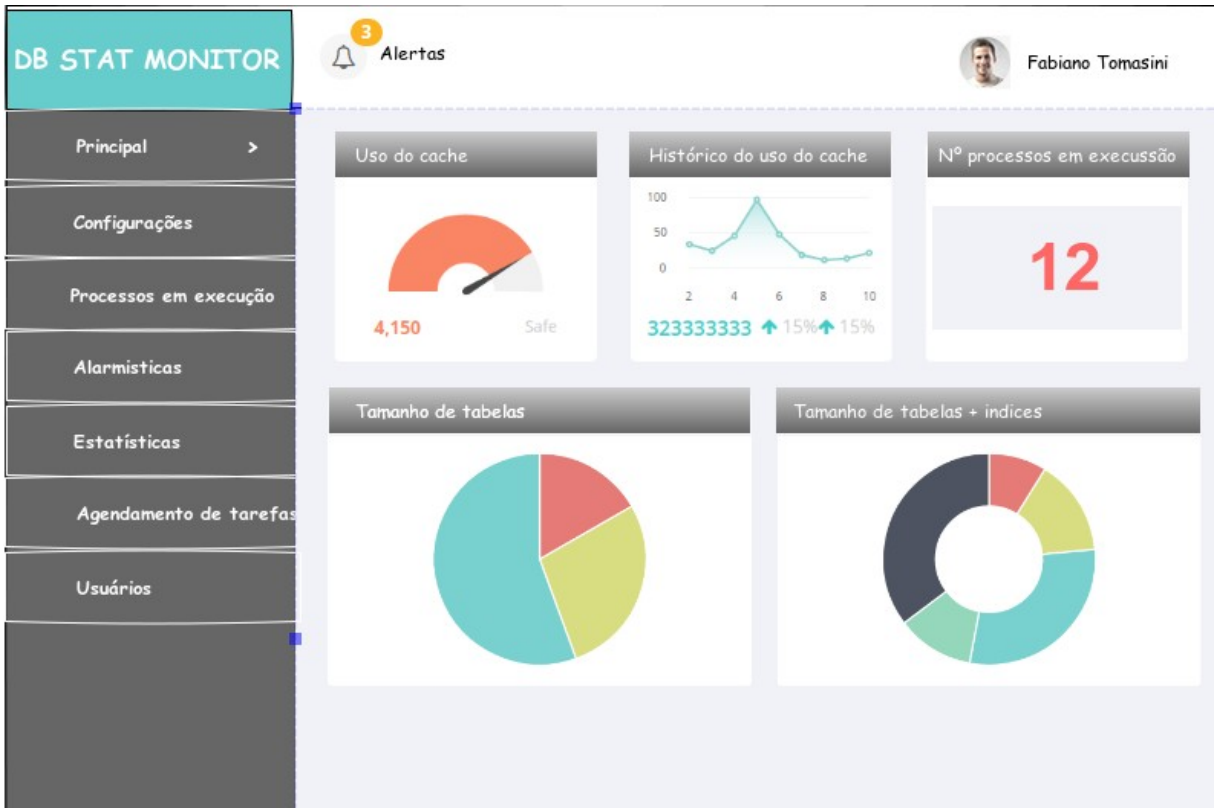
d) Menu alarmísticas: Interface que permite gerenciar os avisos que serão feitos ao DBA sobre possíveis problemas;

e) Menu estatísticas: Interface que permite gerenciar as estatísticas coletadas e configurá-las para aparecer no painel principal;

f) Menu agendamento de tarefas: Através dessa interface o DBA poderá agendar processos para serem executados em um período ocioso do banco de dados;

g) Menu usuários: A partir dessa interface o DBA poderá gerenciar os usuários que poderão acessar a ferramenta.

Figura 22 - Protótipo da ferramenta proposta



Fonte: Elaborado pelo autor.

4.8 Cronograma

Tabela 2 - Cronograma para desenvolver e concluir a proposta. Ano base, 2015

Atividades / Mês	Jan		Fev		Mar		Abr		Mai		Jun	
Coleta de material bibliográfico												
Redação inicial do projeto Tema / Justificativa / Objetivos / Metodologia												
Elaboração de métodos para melhoria de performance do SGBD												
Validação de métodos levantados												
Comparação dos métodos levantados com os existentes em outras ferramentas												
Modelagem da ferramenta proposta												
Revisão da redação e das normas técnicas												
Entrega da versão definitiva												

Fonte: Elaborado pelo autor.

Segue abaixo uma breve explicação de cada etapa do cronograma:

a) Coleta de material bibliográfico: Nesta etapa, será coletado boa parte do material bibliográfico que será utilizado para desenvolver o projeto;

b) Redação inicial do projeto Tema / Justificativa / Objetivos / Metodologia: Nesta etapa será elaborada a redação inicial com o objetivo de deixar claro a abrangência do projeto;

c) Elaboração de métodos para melhoria de performance do SGBD: Nesta etapa será elaborado alguns métodos e algumas técnicas que serão disponibilizadas na ferramentas com finalidade de melhorar o desempenho do SGBD;

d) Validação de métodos levantados: Após a elaboração dos métodos que serão disponibilizados na ferramente precisamos validá-los. Nesta etapa serão feitos alguns testes identificado a viabilidade de aplicação de cada método na ferramenta proposta;

e) Comparação dos métodos levantados com os existentes em outras ferramentas: Nessa etapa será feito um comparativo entre as ferramentas de análise monitoramento já existente com a ferramenta proposta;

f) Modelagem da ferramenta proposta: Após os estudos realizados será feita a modelagem da ferramenta proposta, onde serão levantados todos os requisitos necessários para execução do projeto;

g) Revisão da redação e das normas técnicas: Nessa etapa será revisada toda a documentação gerada antes da entrega da versão definitiva;

h) Entrega da versão definitiva: Entrega do documento revisado;

5 CONSIDERAÇÕES PARCIAIS

Através dos estudos realizados, foi possível verificar a importância que o monitoramento e a análise das estatísticas disponibilizadas pelo SGBD empregam no funcionamento ideal e na performance de um banco de dados. Quando as ações que devem ser tomadas em cima dessas informações não são efetuadas, podem causar transtorno e insatisfação para quem necessita dos dados armazenados para a tomada de decisão, botando em risco o futuro de uma organização.

Até o presente momento, foram feitos estudos bibliográficos com a finalidade de contextualizar o tema abordado e identificar os requisitos da ferramenta proposta, com isso garantir a aprovação dessa etapa e permitir a continuidade do projeto, visando a implementação dos requisitos restantes.

A próxima etapa do presente projeto, tem como principal objetivo, o desenvolvimento da ferramenta proposta, garantindo a implementação de todos os requisitos levantados e efetivação de ajustes na redação do documento final.

REFERÊNCIAS

BERKUS, J. Checklist de Performance do PostgreSQL 8.0. **PostgreSQL WIKI,** 2005. Disponível em: <https://wiki.postgresql.org/wiki/Checklist_de_Performance_do_PostgreSQL_8.0>. Acesso em. Acesso em: 06 mai. 2015.

BIAZUS, D.O. PostgreSQL. **PostgreSQL WIKI,** 2003. Disponível em: <https://wiki.postgresql.org/wiki/Introdu%C3%A7%C3%A3o_e_Hist%C3%B3rico>. Acesso em: 03 mai. 2015.

BLUMM, C; FORNARI, M. R. Dúvidas frequentes sobre Banco de Dados. **Revista SQL Magazine,** 28. ed., 2006.

BORELLO, F. ; KNEIPP, R. E. Álgebra Relacional. **Revista SQL Magazine**, p. 1 - 1, 2008.

CEDRUS. Cedrus: PostgreSQL Manager. Disponível em: <http://sourceforge.net/projects/cedrus/> Acesso em: 19 mai. 2015.

CYBERTEC. Pgwatch Cybertec Enterprise PostgreSQL Monitor. Disponível em: <http://www.cybertec.at/postgresql_produkte/pgwatch-cybertec-enterprise-postgresql-monitor/> Acesso em: 19 mai. 2015.

DATE, C. J. **Introdução a sistemas de bancos de dados.** 8.ed. Rio de Janeiro: Campus, 2004.

ELMASRI, R; NAVATHE, S. B. **Sistemas de banco de dados.** 6. ed. Pearson, 2011.

GIL, A. C. **Como elaborar projetos de pesquisa.** 4. ed. São Paulo: Atlas, 2002.

MILANI, A. **PostgreSQL:** Guia do Programador. São Paulo: Novatec Editora, 2008.

ORACLE. MySQL Enterprise Monitor. Disponível em: <https://www.mysql.com/products/enterprise/monitor.html> Acesso em 19 maio. 2015.

POSTGRESQL. Documentation. Disponível em <http://www.postgresql.org/docs/manuals/archive/ > Acesso em: 18 abr. 2015.

SANTOS, A. R. **Metodologia científica: a construção do conhecimento.** 2. ed. Rio de Janeiro: DP&A editora, 1999.

SILBERSCHATZ, A.; KORTH, H. F.; SUDARSHAN, S. **Sistemas de Bancos de Dados.** 3. ed. Makron Books, 1999.

SILBERSCHATZ, A.; KORTH, H.F.; SUDARSHAN, S; **Sistemas de Bancos de Dados.** 6. ed. Makron Books, 2012.

SMANIOTO, C. E. PostgreSQL. **Revista SQL Magazine,** 37. ed., 2007.

RAMAKRISHNAN, R.; GEHRKE, J. **Sistemas de Gerenciamento de Bando de Dados.** 3. ed. Mc Graw Hill, 2008.

ROB, P.; CORONEL, C. **Sistemas de Bando de Dados**: Projeto, implementação e administração. 8. ed. CENGATE Learning, 2011.