# TUNING PARA SISTEMA GERENCIADOR DE BANCO DE DADOS POSTGRESQL

TOMASINI, Fabiano[2]
[1] Artigo da disciplina de Trabalho Multidisciplinar I
[2] Acadêmico do curso de Engenharia da Computação – Univates, Lajeado, RS, ftomasini.rs@gmail.com

**RESUMO**

Este artigo apresenta um estudo sobre o SGBD (Sistema Gerenciador de banco de dados) PostgreSQL, facilitando a análise e identificação de possíveis gargalos de performance e ressaltando a importância de otimizar esse serviço através de técnicas de tuning com a finalidade de melhorar o desempenho do SGBD.

**PALAVRAS-CHAVE:** PostgreSQL, banco de dados, análise de desempenho e otimização.

## 1. INTRODUÇÃO

Com o desenvolvimento tecnológico, os softwares exigem mais dos computadores, ou seja, maior poder de processamento de modo que a extração de informações importantes sejam obtidas de maneira fácil, rápida e que possam ajudar em decisões importantes.

O banco de dados é considerado uma coleção de dados relacionados que são fornecidos para as aplicações de uma forma de fácil entendimento ao usuário final. Se as informações armazenadas no banco de dados não forem obtidas com facilidade e rapidez podem causar desconforto aos usuários do sistema além de perderem muito tempo com tarefas que deveriam ser rápidas.

O presente artigo visa apresentar soluções para a manutenção de grandes bancos de dados com uma ênfase especial na performance das aplicações que dependem desse serviço.

A sessão 2 explicará conceitos de um banco de dados e a importância da otimização e seus tipos. Já a sessão 3 abordará análise de performance e a sessão 4 irá esclarecer os processos de otimização.

## 2. BANCO DE DADOS

Segundo Korth(1994), um banco de dados “é uma coleção de dados relacionados”, em outras palavras sempre que tenho informações que se relacionam e tratam do mesmo assunto pode ser considerado um banco de dados.

Um banco de dados pode ser exemplificado com situações clássicas do nosso dia a dia como uma lista telefônica ou uma lista de preços.

Já um SGBD é um sistema com recursos para manipular as informações do banco de dados e interagir com o usuário. O principal objetivo de um sistema gerenciador de banco de dados é isolar o usuário de detalhes internos do banco de dados.

### 2.1. Importância de otimizar

Geralmente a performance de um sistema influência diretamente no seu custo, quando a otimização de um SGBD é feita faz com que o sistema aproveite melhor os recursos disponíveis, diminuindo a necessidade de investimentos em hardware. Além disso, também aumenta a produtividade de quem trabalha com o sistema.

### 2.2. Tipos de otimização

A otimização de um SGBD pode ser classificada em diversas categorias, entre elas:

- Otimização dos parâmetros do SGBD.
- Otimização das consultas.
- Distribuição de carga.
- Hardware mais potente.

A otimização dos parâmetros do SGBD, pode ser considerada a parte simples do processo de otimização, e deve ser empregada em todos os casos, pois assumimos que para otimizar outros fatores do banco o hardware deve estar sendo usado de forma adequada (LOUREIRO, 2005).

A otimização das consultas é refatorar comandos SQL afim de facilitar o trabalho do otimizador e obter resultados com menor custo de execução. A otimização de consultas as vezes requer pequenas alterações na estrutura da base

de dados pois nem sempre o modelo ER é feito tendo em vista a performance da aplicação (LOUREIRO, 2005).

A distribuição de carga consiste em colocar partes do trabalho do SGBD em máquinas distintas, de forma que cada um execute uma quantidade reduzida de consultas. Isso pode ser feito de varias formas, replicação, *cluster* de servidores (LOUREIRO, 2005).

O método de colocar um hardware mais potente muitas vezes é o mais utilizado pois não demanda conhecimento da base de dados e nem de como SGBD funciona, porém é uma solução cara e pode ser eficiente em um período curto de tempo pois o volume de dados vai aumentando e consequentemente o sistema vai voltar a ficar lento (LOUREIRO, 2005).

### 2.3. PostgreSQL

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 SGBD PostgreSQL pode ser considerado objeto-relacional por implementar, além das características de um SGBD relacional, algumas características de orientação a objetos, como herança e tipos personalizados (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 Cinetífica 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).

## 3. ANÁLISE DE PERFORMANCE

O primeiro passo para analisar a performance é descobrir se o SGBD realmente é a fonte de lentidão, existem várias técnicas que identificam isso, por exemplo em uma aplicação cliente servidor, temos três componentes para analisar,

camada de aplicação, camada de rede ou camada do SGBD.

Podemos eliminar a possibilidade de gargalos na aplicação se soubermos que ela se comporta bem com outros bancos e ambientes semelhantes.

Uma maneira de analisar se o problema é na camada de rede é fazer uma transferência de arquivo de uma máquina para a outra ou usar um software tipo sniffer para verificar o trafego no servidor.

Caso nenhum dos métodos apresentados tenha confirmado problemas podemos assumir que existe um gargalo no SGBD.

## 4. OTIMIZAÇÃO

### 4.1. Alterando parâmetros do SGBD

Uma das maneiras de melhorar o desempenho do SGBD PostgreSQL é a alteração de parâmetros de configuração, quando instalamos o PostgreSQL esses parâmetros são atribuídos com valores mínimos para a utilização do software porém quando o servidor tem uma boa capacidade de processamento esses parâmetros podem ser alterados resultando em um melhor desempenho do SGBD.

Entre os parâmetros que podem melhorar o desempenho estão:

- *shared_buffer*: Memória compartilhada entre backends para ler e gravar páginas de dados;
- *work_mem*: Memória alocada dentro de cada backend, usada para ordenamentos e agregações;
- *maintenance_wok_mem*: Memória individual para operações de criação de índices e vacuum.

### 4.2. Analisando e otimizando consultas

Entre todas as técnicas de otimização a considerada mais eficaz é a otimização das consultas feitas no banco de dados.

Muitas vezes os programadores quando estão desenvolvendo consultas utilizadas em suas aplicações não se preocupam com a performance do comando SQL podendo assim exigir muito do SGBD para fazer operações que supostamente seriam simples.

Existem várias formas de testar se o comando desenvolvido pelo

programador é eficiente ou não, uma delas é utilizar o comando EXPLAIN que 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 varredura sequencial simples, varredura pelo índice etc.

## 5. CONCLUSÃO

A utilização de boas práticas em banco de dados desde o inicio de um projeto de desenvolvimento de software traz muitos benefícios na utilização dos sistemas e pode evitar problemas futuros com performance, as técnicas de tuning amenizam esses problemas mas se a base de dados não foi bem estruturada o problema pode voltar a acontecer.

A melhor forma de evitar problemas de performance é prever na confecção do modelo do banco de dados e no desenvolvimento de consultas como o sistema se comportaria com um grande volume de dados.

Como trabalhos futuros poderia-se demonstrar boas praticas no planejamento de um modelo de banco de dados.

## REFERÊNCIAS


BIAZUS, D.O. **PostgreSQL**, 2003. Disponível em:<https://wiki.postgresql.org/wiki/Introdu%C3%A7%C3%A3o_e_Hist%C3%B3rico>. Acesso em: 01 nov. 2013.

SILBERSCHATZ, A. ; KORTH, H.F. **Sistemas de Bancos de Dados**, Makron Books, 2a. edição revisada, 1994.

LOUREIRO, R. D. A. **Administração de SGBD: um estudo de caso**, 2005. Monografia – Curso de Graduação em Ciências da Computação, UNIVERSIDADE FEDERAL DE PERNAMBUCO, Recife, 2005.

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