SALÃO DO CONHECIMENTO UNIJUÍ 2014
ciência - tecnologia - desenvolvimento social
Câmpus Ijuí, Santa Rosa, Panambi e Três Passos

Modalidade do trabalho: Relatório técnico-científico
Evento: XXII Seminário de Iniciação Científica

COMPARATIVO DE DESEMPENHO ENTRE POSTGRESQL E DB4O EM UM SISTEMA DE MONITORAMENTO DE SUBESTAÇÕES[1]

Guilherme Fabrin Franco[2], Geisa Ramborger Da Silva[3], Vinícius Maran[4].

[1] Desenvolvimento e Implantação de um Lote Pioneiro de um Sistema de Monitoramento e Supervisão de Unidades Transformadoras e Subestações
[2] Aluno de Ciência da Computação da Unijui, bolsista de iniciação Científica P&D.
[3] Aluna de Ciencia da Computação da Unijui
[4] Professor de Ciência da Computação Unijui

1. INTRODUÇÃO

A necessidade continua de aumentar a eficiência no gerenciamento de informações torna-se gradativamente maior ao longo dos últimos anos, paralelamente a isto o volume e a quantidade de dados coletados e posteriormente armazenados tem sido incrementado sistematicamente o que tem tem gerado a necessidade do estudo e definições de novas metodologias para acessar, de forma rápida, as bases de dados. Muitos dos Bancos de dados utilizados atualmente surgiram na década de 70 onde as informações eram armazenadas em arquivos simples e de difícil acesso, onde apenas especialistas em Sistema de Gerência de Banco de Dados (SGBD) conseguiam manuseá-los, a partir da década de 80 foram desenvolvidos os SGBD relacionais baseado em álgebra relacional (ELMASRI; NAVATHE, 2000). Até os dias de hoje, os SGBD Relacionais, são os mais utilizados, eles são baseados na utilização e manipulação de tabelas. Por outro lado, o desenvolvimento de software e sistemas utiliza o paradigma, na sua concepção, da orientação a objetos. Neste sentido, originou-se uma grande duvida para os desenvolveres de sistemas, é correto realizar o desenvolvimento de um sistema, baseado na orientação a objeto, sobre uma base de dados relacional ou ainda seria mais interessante a utilização de uma base de dados que siga a mesma metodologia, ou seja um banco de dados orientado a objetos. Neste contexto, o presente artigo objetiva realizar uma analise comparativa entre de dois bancos de dados, um relacional e um orientado a objetos, a partir de um sistema de monitoramento de subestações de energia elétrica. A analise será restrita as questões de inserção e posterior pesquisa de dados na base de dados.

METODOLOGIA

Para realizar a análise da velocidade de inserção e seleção de dados, foi desenvolvido um ambiente de teste que foi idealizado a partir da utilização do conceito de virtualização. Foi criada uma maquina virtual que utiliza um processador I5 com 4 GB de memória e 1 TB de espaço em disco, esta maquina utiliza o sistema operacional Linux (Ubuntu Server 13.04). Nesta maquina virtual foram instalados um banco de dados relacional o PostgreSQL e um banco de dados

Modalidade do trabalho: Relatório técnico-científico
Evento: XXII Seminário de Iniciação Científica

orientado a objetos o DB4O. Também foi desenvolvida uma aplicação, em C# .Net., para simular a aquisição de dados a partir da utilização de Sockets. A opção em utilizar o PostgreSQL e o DB4O reside no fato de ambos serem bancos de dados que adotam o conceito open source (sem custo de licença) e de serem amplamente utilizados em vários ambientes de produção. Já a linguagem escolhida para o programa de aquisição de dados foi a linguagem C# .Net pela rica biblioteca de funções existente para manipulação de bases de dados e por fim foi utilizado a linguagem Java para ser utilizada nos programas que realizam a simulação de operação dos Terminais Java TC65i por ser a linguagem nativa de desenvolvimento destes equipamentos. As ferramentas utilizadas para desenvolver os programas foram as IDEs MonoDevelop v.4.0.12 para desenvolvimento dos programas em C#, e Netbeans 7.3 para desenvolvimento dos programas em Java. A seguir são apresentados os esquemas de banco de dados criados para os testes.
Banco relacional, statement da tabela de dados:
CREATE TABLE ceee_substation (substation_data_hora timestamp without time zone, substation_id serial NOT NULL, substation_dados character varing, PRIMARY KEY (substation_id)
);Banco OO, classe C#:
class Dado { public byte[] bytes {get;set;} public DateTime data_hora{get;set;} }

Após a definição e criação da estrutura dos bancos de dados, foi desenvolvido um conjunto de instruções para realizar os procedimentos de inserção e consultas as bases de dados. Os códigos são apresentados a seguir.
Queries usadas para pesquisa e inserção
Para Inserção no Postgres foi utilizada a query: INSERT INTO ceee_substation (substation_data_hora, substation_dados) values (<data>,”0x00 … 0x039”); Para pesquisa no mesmo utilizou-se a query: SELECT substation_dados from ceee_substation where substation between <data_init> and <data_end>; Onde <data>, <data_init> e <data_end> são referencias para as datas requisitadas.
Script usado para pesquisa e inserção no banco de dados orientado a objetos
Para inserção no banco de dados orientado a objetos foi utilizado o script: using (IObjectContainer db = Db4oEmbedded.OpenFile(“File Name”)) { for (int i = 0; i < list.Count; i++) { byte[] temp = list[i]; byte[] buff = new byte[8]; Array.Copy(temp, 48 - 12, buff, 0, 8); long l = BitConverter.ToInt64(buff, 0); DateTime dt = l.ToDateTime(); Dado t = new Dado(dt, temp); db.Store(t); } } Para seleção foi utilizado o script baseado em um exemplo pré-definido: using (IObjectContainer db = Db4oEmbedded.OpenFile(NomeDB4O)) { IList<Dado> results = db.Query<Dado>(delegate(Dado dado) { DateTime init = new DateTime(2014, 02, 20, 8, 00, 00); DateTime end = new DateTime(2014, 02, 20, 9, 00, 00); return dado.dt > init && dado.dt < end; }); foreach(Dado dado

Modalidade do trabalho: Relatório técnico-científico
Evento: XXII Seminário de Iniciação Científica

in results){ Helper.DumpToConsole(dado.bytes); } count = results.Count; }

Esquemas da segunda bateria de testes

Para simular um ambiente mais próximo de um ambiente real foram criadas tabelas e objetos que seguem o padrão adotado no sistema original que monitora uma subestação de energia elétrica com todas as grandezas típicas deste ambiente. Neste sentido, para as próximas análises foram utilizado os esquemas:

Banco relacional, statement da tabela de dados

CREATE TABLE ceee_substation_n (
substation_temperatura real,
substation_corrente_primario real, substation_corrente_secundario real,
substation_tensao_secundario real, substation_data_hora timestamp without time zone,
substation_ventilador boolean, substation_ventilador_protecao boolean, PRIMARY KEY (substation_data_hora));

4.1.2 Banco OO, classe C#:

class Dado { public DateTime substation_data_hora { get; set; } public double substation_temperatura { get; set; } public double substation_corrente_primario { get; set; }
public double substation_corrente_secundario { get; set; } public double substation_tensao_secundario { get; set; } public bool substation_ventilador { get; set; }
public bool substation_ventilador_protecao { get; set; } public Dado(byte[] buffer) {
//Métodos de conversão dos bytes referentes para cada atributo //do objeto.
} }

Utilizando as mesmas formas de inserção e seleções descritas anteriormente, foi realizada uma nova bateria de testes com os esquemas baseados na realidade do sistema atual. Foi verificado, após a analise dos testes, que a conexão a base de dados demora em torno de 230 a 270 ms, neste sentido pode-se deduzir que quando mais otimizado seja o processo de conexão, minimizando o número de conexões, mais rápido será o sistema como um todo.

#	Conectar e Armazenar (ms)	Pesquisar e Mostrar (ms)	Tamanho por Dado (bytes/dado)	Tamanho em um ano e uma subestação (terabyte)	Tamanho em um ano e 160 subestações (terabyte)
Db4o	2927	4810	120	0,0003	0,0531
PostgreSQL	11107	458	256,3886	0,0007	0,116

Tabela 1. Resultados da primeira bateria de testes com 8563 dados.

#	Conectar e Armazenar (ms)	Pesquisar e Mostrar (ms)	Tamanho por Dado (bytes/dado)	Tamanho em um ano e uma subestação (terabyte)	Tamanho em um ano e 160 subestações (terabyte)
Db4o	1249	829	177	0,0005	0,08

Modalidade do trabalho: Relatório técnico-científico
Evento: XXII Seminário de Iniciação Científica

PostgreSQL	3668	18	234,9049	0,0006	0,10632

Tabela 2. Resultados da segundo bateria de testes com 6626 dados.

CONCLUSÃO

Percebe-se na primeira bateria de testes que o DB4O se destaca no processo de inserção com uma vantagem de aproximadamente 8 segundos em relação ao ProstgreSQL, sendo ainda que utiliza 50 % do espaço em disco para realizar o mesmo número de inserções. Já o PostgreSQL apresenta um desempenho 10 vezes superior no processo de seleção quando comparado ao DB4O. Na segunda bateria de testes que é obtido um resultado muito próximo da primeira bateria de testes, entretanto com a versão simulada do ambiente real percebemos que DB4O ainda é muito inferior no processo de seleção das informações. A partir dos testes realizados pode-se afirmar que o DB4O tem desempenho superior no processo de inserção e o PostgreSQL no processo de pesquisa/seleção. Outro ponto favorável ao DB4O é no quisito armazenamento, o DB4O se destaca por armazenar em um menor numero de bytes a mesma informação com uma precisão maior, pois no objeto é usado double e no banco relacional é usado real em partes de situações. Ou seja, no DB4O é possível gravar toda uma estrutura predefinida para cada dado, métodos ou conversões. Como trabalhos futuros pretende-se realizar ajustes no método de pesquisa para, talvez possibilitar ao DB4O reduzir o tempo e consequentemente aumentar seu desempenho tornando-se neste caso uma melhor opção de uso quando comparado com o PostgreSQL. Palavras-chave: database, postgres, db4o

Referências

ELMASRI e NAVATHE "Fundamentals of Database Systems", Benjamin-Cummings, 3a. edição, 2000. Kim W. "Modern Database Systems", ACM Press, 1995 Cattell R.G. "Object Data Management", Addison-Wesley, 2a.edição, 1994. Koshafian S. "Object-Oriented Database Systems" Morgan Kaufmann, 1993. O2 Technology, "The O2 User's Manual" Version 4.5, março 1995. Ozsu e Valduriez "Principles of Distributed Database Systems", Prentice Hall, 1999.