PgGrid: uma implementação de fragmentação de dados para o pgCluster

Gustavo A. Tonini, Frank Siqueira

Departamento de Informática e Estatística – Universidade Federal de Santa Catarina
Florianópolis – SC – Brasil

gustavotonini@gmail.com, frank@inf.ufsc.br

Abstract. *As an organization grows, it needs to store great amounts of data and to organize them in such a way that favors its recovery increases. The proposal of this project is to offer an extension to the PostgreSQL DBMS that allows the fragmentation of data, so that they are distributed in the most convenient form among sites that compose the distributed database system, and to manage the replication of such data. For this it was necessary to modify the "pgcluster" tool, to manage data location and to optimize queries. Morevoer, an extension to the DDL was proposed, for the definition of the data distribution parameters and the sites that compose the distributed database system.*

Resumo. *À medida que as organizações crescem, também cresce a necessidade de armazenar grandes massas de dados e organizá-los de uma forma que favoreça sua recuperação. A proposta deste trabalho é oferecer uma extensão ao SGBD PostgreSQL que permita a fragmentação dos dados para que os mesmos sejam distribuídos da forma mais conveniente nos servidores de banco de dados que compõem o conjunto, além de gerenciar sua replicação. Para isso foi necessário modificar a ferramenta "pgcluster" para gerenciar a localização dos dados no sistema distribuído e otimizar as consultas. Além disso, foi implementada uma extensão à linguagem DDL para a definição dos parâmetros da distribuição dos dados e dos "sítios" que formam o sistema de banco de dados distribuído.*

1. Introdução

Nos primórdios da computação, todo processamento era realizado de forma centralizada. Com o passar dos anos foi-se notando o potencial que as características dos sistemas distribuídos proporcionam. Sistemas de banco de dados distribuídos são sistemas distribuídos que armazenam, manipulam e organizam dados.

Muitos estudos já foram realizados nesta área, principalmente no que diz respeito ao gerenciamento de transações e integridade dos dados em sistemas deste tipo. No entanto, existem poucas implementações disponíveis para uso.

De acordo com Ozsu (1999), as principais promessas e características dos sistemas de banco de dados distribuídos são:

- Administração transparente dos dados fragmentados nas máquinas que compõem o sistema;

- Confiabilidade nas transações distribuídas;
- Melhoria no desempenho das consultas, através da paralelização das mesmas e outras estratégias de otimização; e
- Fácil expansão e boa escalabilidade do sistema.

O presente artigo apresenta a implementação de uma extensão para o SGBD PostgreSQL que tem como intuito adicionar suporte à fragmentação e replicação de dados. Adicionalmente, a linguagem DDL do SGBD foi estendida de forma a suportar em sua sintaxe comandos que permitam definir a forma como os dados serão fragmentados e/ou replicados dentre os servidores integrantes do esquema de dados distribuído.

O artigo está organizado da seguinte maneira: a seção 2 descreve o PostgreSQL e o suporte para replicação existente; a seção 3 descreve as alterações efetuadas no SGBD de modo a suportar replicação e fragmentação; a seção 4 compara a implementação realizada com outros SGBDs distribuídos; a seção 5 demonstra, através de um caso de uso, a utilização do SGBD distribuído; por fim, a seção 6 apresenta as conclusões dos autores e perspectivas de evolução do trabalho.

2. PostgreSQL e pgCluster

PostgreSQL é um sistema gerenciador de banco de dados de código-fonte aberto poderoso, multiplataforma, com mais de 15 anos de desenvolvimento. Atende os padrões SQL99 e seu gerenciador de transações implementa as propriedades ACID [PostgreSQL 2009].

O *pgCluster* é um sistema de replicação síncrono baseado no PostgreSQL. O sistema possui duas funções principais:

- **Balanceamento de carga**: Um módulo recepciona os comandos de consulta e transações e os distribui da forma mais conveniente, levando em consideração a carga (capacidade de processamento comprometida) dos servidores.
- **Alta disponibilidade**: É garantida através da replicação total dos dados; assim, se um servidor falha, os demais podem responder em seu lugar, desde que pelo menos um servidor de replicação continue no ar.

Conforme se pode facilmente perceber, o maior problema do pgCluster é a falta das funcionalidades de fragmentação, que exigem que o usuário replique totalmente as bases de dados para que o sistema funcione, degradando o desempenho e também desperdiçando recursos do sistema.

Dentre uma gama enorme de SGBD relacionais e objeto-relacionais disponíveis no mercado, o PostgreSQL se sobressai devido à sua aceitação pela comunidade de software livre mundial. Além das qualidades citadas acima, o PostgreSQL possui grande usabilidade em ambientes distribuídos homogêneos, possuindo grande parte das funções de replicação já implementadas e testadas.

3. Alterações no pgCluster

Fragmentar os dados é uma necessidade básica na maioria dos ambientes distribuídos. Com este propósito, o pgGrid adiciona funções de fragmentação para o pgCluster e fornece comandos, como uma extensão à SQL, para definição do ambiente de cluster.

A configuração da replicação no pgCluster exige a manipulação de arquivos XML. Desta forma, torna-se complexo para o administrador definir as novas configurações do pgGrid, como a definição dos fragmentos e a alocação dos mesmos. Para resolver este problema foram adicionados comandos específicos para a definição destes parâmetros que serão gravados nos catálogos do sistema.

Três grupos de instruções especiais foram adicionados à gramática:

- Adição e exclusão de servidores no sistema (CREATE SERVER);
- Definição de fragmentos (CREATE FRAGMENT);
- Alocação de fragmentos nos sites (PLACE);

A Figura 1 apresenta alguns exemplos de uso dos comandos adicionados.

```
CREATE SERVER site1 HOST inf.ufsc.br PORT 5432 RECOVERY PORT 7000;
CREATE FRAGMENT cliente_pessoa_fisica ON CLIENTE where tipo_pessoa = 'F';
CREATE FRAGMENT cliente_filial1 ON CLIENTE where cod_filial = 1;
PLACE cliente_pessoa_fisica ON site1;
```

Figura 1. Exemplo de uso dos comandos de definição do cluster

As informações definidas nos comandos supracitados ficam armazenadas nos catálogos do sistema. Quatro catálogos foram adicionados com este propósito:

- pg_grid_site;
- pg_grid_fragment;
- pg_grid_fragment_attribute;
- pg_grid_allocation;

Depois que os parâmetros foram definidos, o pgGrid mantém os servidores sincronizados de forma a garantir as regras de fragmentação explicitadas pelo administrador.

Sites e fragmentos podem ser definidos e alocados a qualquer momento, sem a necessidade de interrupção do funcionamento do sistema. Tal característica provê a escalabilidade necessária para aplicações reais.

3.1. Processamento dos comandos de manipulação de dados

Toda vez que um comando INSERT, UPDATE ou DELETE é submetido ao servidor, o mesmo é analisado e enviado para um módulo denominado *executor*. O executor foi alterado para consultar os catálogos do pgGrid e verificar a necessidade de replicação da solicitação.

Quando o sistema identifica que um comando destes deve alterar dados de um ou mais sítios, é iniciada uma transação distribuída onde o comando em questão é

replicado para todos os sites envolvidos. A execução do comando somente é dada como concluída quando todos os sites confirmarem o sucesso da operação para o site que recebeu a solicitação.

3.2. Processamento das consultas distribuídas

Toda vez que um comando SELECT é submetido ao servidor, o mesmo é analisado, passa por otimizações para melhorar seu desempenho e depois é enviado para o mesmo módulo *executor* dos demais comandos. O executor das consultas foi alterado para consultar os catálogos do pgGrid e verificar a necessidade de solicitação de dados para outros sites do cluster.

Quando o sistema identifica que uma consulta necessita de dados externos (através de reduções [Ozsu 1999]), uma solicitação é enviada para cada site que contém dados usados pela consulta. Cada site processa as operações possíveis no seu conjunto de dados (provendo paralelização) e envia o resultado para o servidor que recebeu a solicitação, este é responsável por "juntar" os dados (através de operações de união e junção) e apresentar o resultado final ao usuário.

4. Outras implementações de grid

Como parte do trabalho, foi realizada a avaliação de várias implementações de banco de dados distribuídos disponíveis no mercado.

Os gerenciadores avaliados foram: MySQL Cluster, Oracle RAC e IBM DB2. A avaliação foi realizada segundo alguns critérios relevantes para a manutenção de um sistema de banco de dados distribuído, de uma forma comparativa com o pgGrid.

A Tabela 1 lista os critérios escolhidos e a avaliação de cada produto quanto à compatibilidade com o mesmo.

Característica	**MySQL**	**Oracle**	**IBM DB2**	**pgGrid**
Consultas distribuídas	Sim	Sim	Sim	Sim
Tecnologia multi-mestre	Sim	Sim	Sim	Sim
Replicação parcial	Não	Sim	Sim	Sim
Replicação síncrona	Sim	Sim	Sim	Sim
Replicação assíncrona	Não	Sim	Não	Não
Fragmentação	Sim	Sim	Sim	Sim
Integridade referencial entre os sites	Não	Não	Não	Sim
Protocolo de confiabilidade distribuída	2PC	2PC	2PC	2PC

Tabela 1. Comparativo entre as tecnologias de BDD

5. Caso de uso

Para demonstrar as funcionalidades do pgGrid, um esquema de BDD foi montado. O esquema armazena dados sobre produtos produzidos no estado de Santa Catarina, conforme ilustrado no diagrama entidade-relacionamento da Figura 2.

Cinco sites pertencem ao sistema, localizados nas principais cidades do estado: Florianópolis (FLN), Joinville (JVL), Blumenau (BLU), Criciúma (CRI) e Chapecó (XAP). As regras de armazenamento são simples: cada cidade armazena seu cadastro e os produtos produzidos em sua região. A capital (FLN), além dos dados de seus produtos, armazena o cadastro de todas as cidades do estado. Os produtos possuem como atributos: código, nome e cidade de origem. As cidades possuem código, nome e a distância (em quilômetros) até a capital.

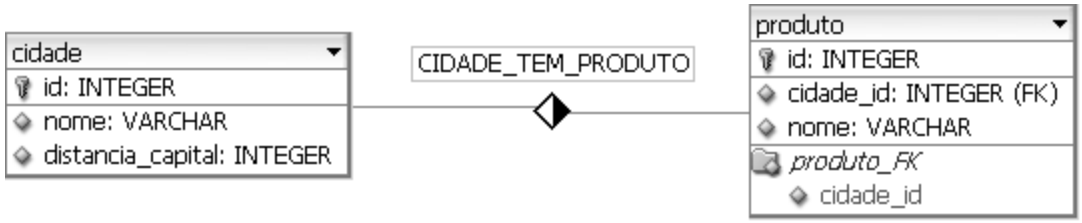

Figura 2. Diagrama entidade-relacionamento correspondente ao caso de uso

Depois que os cinco servidores foram configurados e colocados no ar, o modelo de dados e sua fragmentação foram definidos através dos comandos apresentados na Figura 3.

```
/*relação cidade*/
sc=# CREATE TABLE CIDADE (ID INT, NOME VARCHAR, DISTANCIA_CAPITAL INT);
sc=# ALTER TABLE CIDADE ADD CONSTRAINT PK_CIDADE PRIMARY KEY (ID);
/*relação produto*/
sc=# CREATE TABLE PRODUTO (ID INT, NOME VARCHAR, ID_CIDADE_ORIGEM INT);
sc=# ALTER TABLE PRODUTO ADD CONSTRAINT PK_PRODUTO PRIMARY KEY (ID);
sc=# ALTER TABLE PRODUTO ADD CONSTRAINT FK_PRODUTO_CIDADE FOREIGN
sc=# KEY (ID_CIDADE_ORIGEM) REFERENCES CIDADE (ID);
/*definição e alocação dos fragmentos dos produtos*/
sc=# CREATE FRAGMENT PRODUTO_FLN ON PRODUTO WHERE ID_CIDADE_ORIGEM=1;
sc=# PLACE PRODUTO_FLN ON FLN;
sc=# CREATE FRAGMENT PRODUTO_JVL ON PRODUTO WHERE ID_CIDADE_ORIGEM=2;
sc=# PLACE PRODUTO_JVL ON JVL;
sc=# CREATE FRAGMENT PRODUTO_BLU ON PRODUTO WHERE ID_CIDADE_ORIGEM=3;
sc=# PLACE PRODUTO_BLU ON BLU;
sc=# CREATE FRAGMENT PRODUTO_CRI ON PRODUTO WHERE ID_CIDADE_ORIGEM=4;
sc=# PLACE PRODUTO_CRI ON CRI;
sc=# CREATE FRAGMENT PRODUTO_XAP ON PRODUTO WHERE ID_CIDADE_ORIGEM=5;
sc=# PLACE PRODUTO_XAP ON XAP;
/*definição e alocação dos fragmentos das cidades*/
CREATE FRAGMENT CIDADE_FLN ON CIDADE;
PLACE CIDADE_FLN ON FLN;
CREATE FRAGMENT CIDADE_JVL ON CIDADE WHERE ID=2;
PLACE CIDADE_JVL ON JVL;
CREATE FRAGMENT CIDADE_BLU ON CIDADE WHERE ID=3;
PLACE CIDADE_BLU ON BLU;
CREATE FRAGMENT CIDADE_CRI ON CIDADE WHERE ID=4;
PLACE CIDADE_CRI ON CRI;
CREATE FRAGMENT CIDADE_XAP ON CIDADE WHERE ID=5;
PLACE CIDADE_XAP ON XAP;
```

Figura 3. Comandos utilizados na definição da fragmentação dos dados no caso de uso do pgGrid

A fragmentação foi definida através do identificador numérico de cada cidade. Este código representa a regra de alocação dos dados nos sites.

Foram realizados testes inserindo as cidades e vários produtos no sistema. Depois todos os sites foram reiniciados no modo centralizado (sem sincronia com o cluster). Cada site continha somente os dados da cidade que representava, atestando o funcionamento do pgGrid. O site que representava a capital listou todas as cidades do estado, conforme configurado.

6. Conclusões

Com a crescente necessidade de compartilhamento de informações, está surgindo uma demanda muito grande no armazenamento centralizado das mesmas. No entanto, com volumes muito grandes de dados, a centralização torna o processo de recuperação e carga dos dados ineficiente.

Para resolver tal problema, os sistemas distribuídos entram em jogo fornecendo soluções escaláveis e paralelizáveis. Este trabalho possui o intuito de propor uma solução deste tipo no contexto de banco de dados relacionais e objeto-relacionais.

No decorrer do trabalho, a escolha do PostgreSQL como base para a implementação se tornou muito vantajosa devido ao nível de estabilidade e amadurecimento de suas funcionalidades, assim como pela qualidade e legibilidade do seu código-fonte.

A proposta apresentada no início do trabalho mostrou-se viável no decorrer do desenvolvimento do mesmo. Tal viabilidade foi ilustrada pelo caso de uso. Existem muitas otimizações que devem ser implementadas e o software deve passar por testes mais rigorosos antes de ser colocado em produção em casos reais.

Como próximo passo planeja-se implementar as otimizações de consultas que o ambiente distribuído possibilita, tais como a paralelização de subconsultas. Também se pretende aperfeiçoar os servidores de modo a melhorar o protocolo de comunicação entre os mesmos e remover a necessidade de servidores de replicação.

Repositório do projeto

PgGrid é um software de código-fonte aberto, disponível pela licença GPL e está disponível para download no site http://pgfoundry.org/projects/pggrid/.

Referências

Ozsu, M. Tamer (1999) “Principles of distributed database systems”. New Jersey, United States, Prentice Hall, p. 7-10.

PostgreSQL Development Group (2009) “About PostgreSQL”, http://www.postgresql.org/, Janeiro.

Bedoya, Hernando et al. “Advanced Functions and Administration on DB2 Universal Database for iSeries”, http://www.redbooks.ibm.com/abstracts/sg244249.html, Janeiro.