PgSimilar: Uma ferramenta *open source* para suporte a consultas por similaridade no PostgreSQL

Eduardo N. Borges[*1], Carina F. Dorneles[2]

[1]Instituto de Informática – Universidade Federal do Rio Grande do Sul (UFRGS)
Caixa Postal 15.064 – 91.501-970 – Porto Alegre – RS – Brasil

[2]Ciência da Computação - Instituto de Ciências Exatas e Geociências – ICEG
Universidade de Passo Fundo – UPF

enborges@inf.ufrgs.br, dorneles@upf.br

Abstract. *Currently, the commercial database management systems (DBMS) do not support approximate queries directly. There is a need for technologies on DBMS to support this type of queries. This paper presents a tool which was developed to attach some functionality to the PostgreSQL DBMS. A similarity open source module named PgSimilar was developed to implement some similarity functions that handle similarity between strings. This module was attached to the DBMS to provide support for approximate queries. Furthermore, operators for similarity comparison for metric domains have also been developed.*

Resumo. *Atualmente, os sistemas gerenciadores de bancos de dados (SGBD) comerciais não suportam diretamente consultas aproximadas. Faz-se necessário um SGBD que implemente este tipo de pesquisa. O presente trabalho apresenta uma ferramenta desenvolvida no intuito de anexar funcionalidades ao SGBD PostgreSQL. Foi desenvolvido um módulo de similaridade open source denominado PgSimilar que implementa funções e procedimentos que tratam vários tipos de similaridade entre strings. Este módulo foi anexado ao SGBD para dar suporte às consultas aproximadas. Além disso, operadores de comparação por similaridade para domínios métricos foram desenvolvidos.*

1. Introdução

Variações na representação das informações permitem diversas instâncias de um mesmo objeto do mundo real. Esta variação deve-se ao fato de que os dados armazenados são digitados por diferentes usuários ou gerados por diferentes aplicações. Os mecanismos tradicionais de pesquisa em banco de dados utilizam uma busca por comparação exata.

---

[*] Este trabalho foi realizado na Fundação Universidade Federal do Rio Grande (FURG) durante o desenvolvimento do Trabalho de Conclusão de Curso do autor.

Em uma base de dados de cidades do Rio Grande do Sul, por exemplo, o nome de uma cidade pode estar armazenado de diversas formas como na Figura 1: `Rio Grande`, `R. Grande`, ou ainda `Rio Graande`. Todas as representações são consideradas objetos diferentes, mas fazem referência a uma mesma cidade. A recuperação da informação por meio de consultas exatas pode ser ineficiente neste caso. Portanto, torna-se necessária a utilização de um mecanismo de pesquisa mais elaborado com suporte a consultas imprecisas ou aproximadas, também conhecidas como consultas por similaridade [Jagadish 1995].

"Selecione as cidades que sejam, no mínimo, 70% similares a Rio Grande"

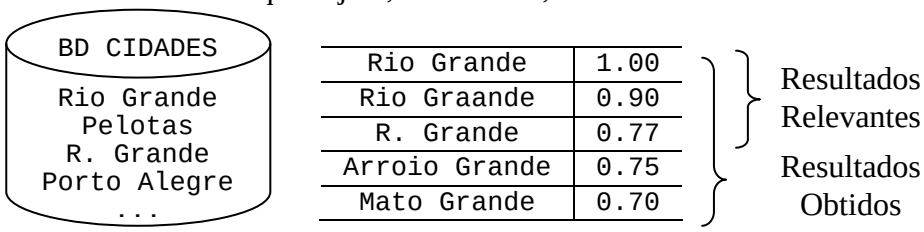

Figura 1. Consulta por similaridade aplicada a uma base de cidades.

Algoritmos de pesquisa baseados em funções de similaridade são aplicados nas bases de dados onde a consulta deve ser precisa o suficiente para que possa retornar resultados satisfatórios (os objetos que representam a cidade `Rio Grande`). Ao mesmo tempo a consulta deve ser flexível o bastante a fim de retornar todos, ou a maior parte, dos resultados relevantes (`R.Grande` neste exemplo).

Os resultados obtidos por meio da consulta representada na Figura 1 são formados por uma lista de cinco cidades, mas apenas três delas são resultados relevantes, os quais representam a cidade de `Rio Grande`. Nota-se que a consulta abrange mais cidades do que deveria. O limiar de 70% de similaridade adotado neste caso faz com que mais objetos façam parte do raio de busca da consulta. Definições adequadas de valores de limiar são discutidas na literatura [Stasiu et al. 2005] e não fazem parte do escopo do trabalho apresentado neste artigo.

O sistema de indexação de citações CiteSeer [Guiles et al. 1998] é um exemplo de base de dados cujas informações são originadas por meio da integração de várias outras bases. O sistema indexa literatura acadêmica no formato eletrônico (arquivos *Postscript* na *Web*) e identifica citações do mesmo artigo em diferentes formatos. Desta forma, a base de dados contém informações repetidas, mas que são diferentes na maneira como são representadas.

Atualmente, os sistemas gerenciadores de bancos de dados (SGBD) comerciais não suportam diretamente consultas aproximadas [Gravano et al. 2001]. Para que informações similares (dados representados de maneira diferente, mas com o mesmo significado) possam ser recuperadas através de uma consulta simples é necessário que o SGBD possua esta funcionalidade implementada em seu *engine* de consulta.

Visando preencher uma lacuna não abordada pelos SGBD no que tange ao tratamento de consultas aproximadas, o presente trabalho apresenta uma contribuição para um SGBD que permite solucionar problemáticas referentes à ineficiência das consultas por comparação exata. Esta contribuição é uma extensão para o SGBD comercial PostgreSQL [2005] denominada **PgSimilar**. Esta ferramenta é constituída de

um módulo de pesquisa por similaridade composto de um conjunto de funções de similaridade entre *strings*, operadores SQL e procedimentos para o suporte a consultas aproximadas.

Este artigo está dividido da seguinte forma: a seção 1 descreve a motivação e os objetivos da ferramenta. O funcionamento, o pacote de extensão do PostgreSQL e as funções de similaridade implementadas são apresentados na seção 2. E por fim, na seção 3, são descritas as considerações finais, conclusões e trabalhos futuros.

2. PgSimilar

O módulo **PgSimilar** foi projetado com o objetivo de possibilitar ao usuário a realização de consultas utilizando as funções de forma transparente, pois estas são executadas internamente no SGBD. Após a análise realizada pelo `Analisador Léxico e Sintático` do PostgreSQL, o `Sistema de Reescrita` analisa semanticamente a consulta. Nesta etapa, o SGBD traduz as chamadas das funções do módulo **PgSimilar** e reescreve a árvore de análise. Após, são verificadas as regras no catálogo do sistema e é elaborado um plano de execução otimizado. Os dados são buscados no banco através do `Processador de Consultas` e exibidos ao usuário através da interface de conexão.

Na primeira vez que uma função definida no **PgSimilar** é chamada em uma sessão, o carregador dinâmico carrega o arquivo objeto na memória para que a função possa ser chamada. O arquivo objeto carregado dinamicamente é mantido em memória após a primeira carga, ou seja, a primeira utilização de uma das funções contidas no código objeto. Chamadas posteriores das funções somente envolvem acessos à tabela de símbolos. Isto ocasiona um ganho no desempenho pelo fato das funções já estarem contidas na memória. A Figura 2 exibe os passos da execução de uma consulta aproximada utilizando funções do **PgSimilar**.

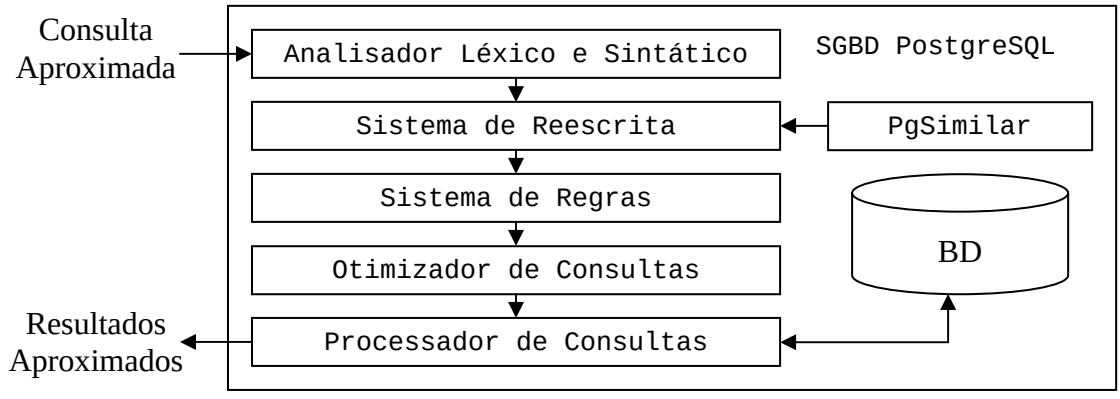

Figura 2. Execução de uma consulta utilizando as funções do PgSimilar.

2.1. Pacote

A ferramenta desenvolvida consiste em um módulo de extensão composto por vários arquivos organizados através da estrutura representada na Figura 3. A seguir serão descritas as funções de cada arquivo do pacote.

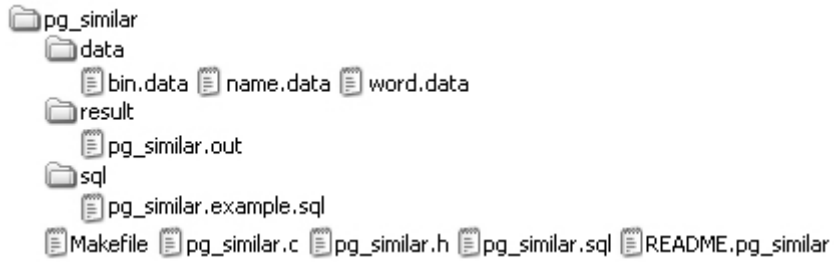

Figura 3. Estrutura de arquivos do pacote PgSimilar.

- `pg_similar.example.sql`: conjunto de testes que abrange todas as funções de similaridade do pacote. Neste *script*, as consultas fazem referência aos dados de teste contidos nos arquivos `bin.data`, `name.data` e `word.data`.
- `pg_similar.out`: contém a saída esperada para a execução dos testes.
- `pg_similar.h`: arquivo de biblioteca que contém os cabeçalhos das funções de similaridade e funções auxiliares.
- `pg_similar.c`: código-fonte do software que contém os procedimentos e funções de similaridade que tratam de vários tipos de similaridade entre *strings*.
- `pg_similar.sql`: *script* para instalar o pacote.
- `Makefile`: *script* para compilar o pacote. Faz uso do `Makefile.global` do código-fonte do SGBD PostgreSQL.
- `README.pg_similar`: arquivo de ajuda do software.

A instalação é simples. O pacote pode ser encontrado no endereço eletrônico do Instituto de Informática da UFRGS http://www.inf.ufrgs.br/~enborges/pg_similar.zip. Basta o usuário descompactar o pacote na pasta de contribuições do PostgreSQL, compilar o código-fonte e executar o *script* `pg_similar.sql` sobre uma base de dados `dbname`.

```
psql -d dbname -f /usr/local/pgsql/share/contrib/pg_similar.sql
```

No desenvolvimento das funções de similaridade anexadas ao PostgreSQL foi utilizada uma convenção de chamada para as funções em C denominada versão 1. Esta convenção faz uso de macros definidas nas bibliotecas do SGBD. As macros são utilizadas no intuito de otimizar a legibilidade, manutenibilidade e inteligibilidade do código fonte. Além disso, suprimem a maior parte da complexidade da passagem de argumentos e resultados.

```
PG_FUNCTION_INFO_V1(setlimit);
Datum setlimit(PG_FUNCTION_ARGS)
{
      float4 limit = PG_GETARG_FLOAT4(0);
      if (limit < 0 || limit > 1.0)
          elog(ERROR, "Should be between 0 and 1");
      pgsimlimit = limit;
      PG_RETURN_FLOAT4(pgsimlimit);
}
```

Figura 4. Uso de macros no desenvolvimento das funções.

Na Figura 4 é definida a função que altera o limiar de similaridade adotado nas consultas. O acesso aos argumentos é realizado utilizando a macro `PG_GETARG_X()` correspondente ao tipo de dado do argumento. A marco recebe como parâmetro o número do argumento da função, contado a partir de 0. O resultado é retornado utilizando a macro `PG_RETURN_X()` para o tipo de dado do parâmetro.

Tabela 1. Funções de gerenciamento de limiar.

Função	Descrição
`pg_similar_limit()`	`Retorna o limiar de similaridade atual`
`pg_similar_setlimit (real)`	`Atribui um valor ao limiar de similaridade adotado nas consultas subseqüentes`

2.2. Funções de Similaridade

A avaliação da similaridade entre objetos de um mesmo domínio (*strings*) é realizada por meio de funções baseadas na diferença entre palavras. Alguns exemplos destas funções são Hamming, Levenshtein [Levenshtein 1966], Jaro [Jaro 1989], Jaro-Winkler [Winkler 1990] e Carla [Mergen & Heuser 2005]. Conforme Cohen [et al. 2003], tais funções são utilizadas nas mais diversas áreas, como: checagem de significado, reconhecimento de pronúncia, análise de DNA, detecção de plágios, entre outras.

Tabela 2. Operadores de similaridade propostos.

Função de Similaridade	Operador Binário
`pg_similar_hamming (text, text)`	`Não Disponível`
`pg_similar_levenshtein (text, text)`	`~=`
`pg_similar_jaro (text, text)`	`Não Disponível`
`pg_similar_jarowinkler (text, text)`	`~~=`
`pg_similar_carla (text, text)`	`~==`

Foram implementadas as funções de similaridade entre *strings* citadas, funções de gerenciamento de limiar (*threshold* ou *limit*) e operadores de similaridade visando o suporte a consultas por abrangência (*Range Query*). A Tabela 1 descreve as funções de limiar. Já a Tabela 2 apresenta os operadores e as respectivas funções.

```
SELECT pg_similar_setlimit(0.7);
```

Consulta 1:

```
SELECT nome, pg_similar_levenshtein(nome, 'rio grande') as similaridade
FROM cidades
WHERE nome ~= 'rio grande'
ORDER BY similaridade DESC, nome;
```

Consulta 2:

```
SELECT nome, pg_similar_hamming(nome, 'rio grande') as similaridade
FROM cidades
WHERE pg_similar_hamming(nome,'rio grande') > pg_similar_limit()
ORDER BY similaridade DESC, nome;
```

Figura 5. Exemplos de consultas.

Na Figura 5 é exemplificado o uso de algumas das funções contidas no módulo **PgSimilar**. A primeira consulta realiza a busca apresentada na Figura 1. Já a segunda consulta mostra como é possível realizar a mesma busca utilizando a função Hamming, a qual opera somente palavras de tamanho igual. O operador correspondente não foi

implementado, pois dificilmente as palavras de uma coluna possuem a mesma largura. Já o operador correspondente à função Jaro não foi implementado porque a função Jaro-Winkler retorna resultados mais significativos por tratar-se de uma otimização deste algoritmo.

3. Conclusões e trabalhos futuros

Este trabalho visa preencher uma lacuna não abordada pelos SGBDs no que tange ao tratamento de consultas aproximadas. O desenvolvimento de uma contribuição para um SGBD permite demonstrar as facilidades e vantagens do uso da similaridade nestes sistemas e, além disso, solucionar problemáticas referentes à ineficiência das consultas por coincidência exata em certas bases de dados.

Novos tipos de dados e funções podem ser adicionados ao módulo de similaridade, aumentando a capacidade do sistema sem comprometer sua robustez e velocidade ou funcionalidade das consultas.

Métodos e índices de acessos adequados a cada uma das funções de similaridade devem futuramente ser acrescentados a esta proposta, pois o tempo de comparação entre dois objetos pode ser muito alto. Também deve ser anexado à ferramenta o suporte a consultas por similaridade aos k vizinhos mais próximos (kNNQ), tornando o **PgSimilar** uma ferramenta completa para a similaridade de texto em SGBD.

Referências

Cohen, W., Ravikumar, P., and Fienberg S. (2003): “A Comparison of String Distance Metrics for Name-Matching Tasks” in IIWeb 2003: 73-78.

Giles, C.L., Bollacker, K., and Lawrence, S. “CiteSeer: An Automatic Citation Indexing System”, Digital Libraries 98: Third ACM Conf. Digital Libraries, ACM Press, New York, 1998, pp. 89-98.

Gravano, L., Ipeirotis, P. G., Jagadish, H. V., Koudas, N., Muthukrishnan, S., Pietarinen, L., and Srivastava, D. (2001). “Using q-grams in a DBMS for Approximate String Processing”. IEEE Data Eng. Bull. 24(4): 28-34.

Jagadish, H. V., Mendelzon A. O., and Milo T. “Similarity-Based Queries”, Proceedings of the Fourteenth ACM SIGACT-SIGMOD-SIGART Symposium on Principles of Database Systems, May 22-25, 1995, San Jose, California. ACM Press. pages: 36-45.

Jaro, M. A. (1989). Advances in record-linkage methodology as applied to matching the 1985 census of Tampa, Florida. Journal of the American Statistical Association 84:414–420.

Levenshtein, I. V. (1966). Binary codes capable of correcting deletions, insertions and reversals. Cybernetics and Control Theory, 10(8):707–710.

Mergen, S. L. S., Heuser, C. A. “Carla: Uma técnica para comparação de cadeias de caracteres”. Escola Regional de Banco de Dados (ERBD), 2005, Porto Alegre. p. 55-60.

PostgreSQL (2005), “Sistema Gerenciador de Bancos de Dados PostgreSQL 8.0.3”, url: http://www.postgresql.org/download, acesso em outubro de 2005.

Stasiu, R.K., Heuser, C.A., and Silva, R.: Estimating Recall and Precision for Vague Queries in Databases. CAiSE 2005: 187-200.

Winkler, W. E. (1990). String Comparator Metrics and Enhanced Decision Rules in the Fellegi-Sunter Model of Record Linkage. Proceedings of the Section on Survey Research Methods, American Statistical Association. 354-359.