Documentos

Novembro, 2007 71

ISSN 1677-9274

Migração de Banco de Dados Oracle para PostgreSQL

Embrapa

ISSN 1677-9274
Novembro, 2007

Empresa Brasileira de Pesquisa Agropecuária
Embrapa Informática Agropecuária
Ministério da Agricultura, Pecuária e Abastecimento

Documentos 71

Migração de Banco de Dados Oracle para PostgreSQL

Evandro Porto de Souza

Embrapa Informática Agropecuária
Campinas, SP
2007

Embrapa Informática Agropecuária
Área de Comunicação e Negócios (ACN)
Av. André Tosello, 209
Cidade Universitária "Zeferino Vaz" – Barão Geraldo
Caixa Postal 6041
13083-970 – Campinas, SP
Telefone (19) 3789-5743 – Fax (19) 3289-9594
URL: http://www.cnptia.embrapa.br
e-mail: sac@cnptia.embrapa.br

Supervisão editorial: *Ivanilde Dispato*
Normalização bibliográfica: *Maria Goretti Gurgel Praxedes*
Editoração eletrônica: *Área de Comunicação e Negócios (ACN)*

1ª. edição on-line - 2007

Souza, Evandro Porto de.
Migração de banco de dados Oracle para PostgreSQL / Evandro Porto de Souza. – Campinas: Embrapa Informática Agropecuária, 2007.
37 p. : il. – (Documentos / Embrapa Informática Agropecuária ; 71).

ISSN 1677-9274

1. Oracle. 2. PostgreSQL. 3 Database. 4. SGBD. 5. Migração.

CDD – 21st 005.7565

Autor

Evandro Porto de Souza
Bacharel em Ciência da Computação,
Analista da Embrapa Informática Agropecuária
Caixa Postal 6041 - Barão Geraldo
13083-970 - Campinas, SP
Telefone: (19) 3789-5772
e-mail: evandro@cnptia.embrapa.br

Apresentação

Este documento tem o propósito de apresentar uma solução viável para a migração de banco de dados Oracle para PostgreSQL, bem como avaliar o desempenho desta nova plataforma, a partir da adoção de um banco de dados como projeto-piloto.

O trabalho a ser investigado consiste no levantamento de informações sobre as plataformas de Sistema de Gerenciamento de Banco de Dados (SGBD) envolvidas, os métodos passíveis de implementação e as ferramentas específicas para a execução deste objetivo.

Durante o desenvolvimento dos trabalhos, serão apresentados os procedimentos utilizados, os testes realizados, bem como, os resultados obtidos e considerações finais, positivas e negativas, envolvendo todo o processo.

É esperado que, ao final deste trabalho, além de atingir a meta proposta, isto é, realizar a migração propriamente dita, deverá ser elaborado um guia que facilite, contribua e oriente outros usuários em procedimentos similares.

Eduardo Delgado Assad
Chefe-Geral

Sumário

Migração de Banco de Dados Oracle para PostgreSQL

Evandro Porto de Souza

Preparação do Ambiente Operacional

Este trabalho consiste no estudo, análise e identificação de uma solução voltada para a conversão de bancos de dados implementados na plataforma Oracle para o PostgreSQL, visando atender as diretrizes da Empresa quanto à adoção de software livre.

Além disso, este plano propõe-se também a diagnosticar e avaliar o desempenho do PostgreSQL com o intuito de se aferir a real viabilidade na sua implantação.

Banco de dados

Para fins dos estudos, implementação e testes foi selecionado para o projeto-piloto o Banco de Dados “Agroclima” que, por se tratar de uma base com um considerável quantitativo de tabelas e um grande volume de dados, proporciona uma configuração ideal para a busca de uma solução na qual este trabalho se propõe. Segue relação das tabelas existentes no banco de dados Agroclima, bem como os quantitativos de registros em cada uma (Tabela 1 - posição de junho/2007).

Tabela 1. Lista e volume das tabelas do DBAgroclima.

Nº	TABELA	Quantidade Registros	Nº	TABELA	Quantidade Registros
1	alerta	-	29	estacoes	1.254
2	animais	5.558	30	estacoesvirtuais	9.925
3	av_alerta	1	31	estvizinha	14.441
4	av_alertaboi	-	32	estvizinhavirtuais	62.758
5	av_area	2	33	gta	24
6	av_area_bov	5	34	instfonte	-
7	av_atividade	3	35	instituicao	37
8	av_atuacaoestabelecimento	-	36	localextra	-
9	av_caracteristicaadicional	32	37	monitoramento_cultivar	245
10	av_classificacao	15	38	monitoramento_mun	5.527
11	av_estabelecimento	1.833	39	municipio	647
12	av_motivo	5	40	municipios	5.574
13	av_motivo_bov	16	41	municipiovilas	9.473
14	av_status	2	42	previsao	3.465.987
15	av_tipopessoa	2	43	probabilidade	9.125
16	av_tipopropriedade	2	44	producao	455.671
17	av_tipotecnico	2	45	propriedades	20
18	av_usuario	1	46	radiacao	71
19	boletim	30.344	47	regiao	-
20	cliente	-	48	regiaoestacoes	-
21	climadia	3.674.709	49	simulado	-
22	climadiatemp	-	50	simulados	191.063
23	climadiavirtuais	8.848.741	51	temp	153.690
24	climaper	1.025.458	52	teste	1
25	climapertemp	-	53	usuario	111
26	climatemp	-	54	usuarios	5
27	cultivargd	3	55	zoneamento	1.653.894
28	estacao	110	56	zoneamento0405	2.276.099

Plataforma de SGBD atual

Atualmente, a base de dados “Agroclima” é implementada na plataforma de grande porte Oracle versão 9i, que se trata do mais completo e estável Sistema Gerenciador de Banco de Dados (SGBD) para grandes volumes de dados, o que lhe rende o título de melhor e mais utilizado SGBD em todo mundo, detendo cerca de 42% do mercado mundial (ORACLE CORPORATION, 2005).

Política de software livre na Empresa

A adesão ao software livre tem crescido a cada dia, haja vista a liberdade e flexibilidade quanto ao desenvolvimento e otimização que este proporciona aos seus usuários, assim como a racionalização de custos com aquisição de licenças.

Em virtude disso, o governo federal tem investido e apoiado o desenvolvimento e utilização do software livre em todo o país, em especial nos órgãos e entidades públicos. A Embrapa, na condição de empresa estatal, também tem se adequado a essa política de governo, por meio da elaboração de diretrizes que incentivam sua adesão.

Plataforma de SGBD a ser implementada

Ribamar (2006) o PostgreSQL é um SGBD relacional de médio e grande porte com algumas implementações e recursos orientados a objetos, e como já mencionado anteriormente é um software de livre distribuição e *open source*, ou seja, possui código-fonte disponível para modificação, sob a licença BSD (Berkeley Software Distribution). Atualmente, o PostgreSQL é equiparado aos melhores SGBD comerciais, devido à infinidade de recursos, a alta performance, a estabilidade e a capacidade de armazenamento de grandes volumes de dados. Essa comparação do PostgreSQL às melhores plataformas de banco de dados é reforçada pelo fato deste ser adotado por grandes empresas de todo o mundo como: Apple, Sun, RadHat, Basf, Fujtsu, SRA, Afilias (registros de domínio .org), entre outras.

O PostgreSQL possui muitas funcionalidades e recursos avançados, além de suportar as principais características contidas nos mais populares SGBD, tais como:

a) Padrão SQL: é implementada no padrão ANSI SQL (89, 92 e 98), além de suportar grande parte do SQL:2003;
b) Controle de concorrência multiversão (MVCC): trata-se de um mecanismo para controle de manipulação simultânea por parte de usuários para os mesmos registros;
c) Tipos definidos pelo usuário: o usuário pode criar tipos (de dados, operadores, métodos de índices, funções, funções de agregação e Store Procedures) visando atender às suas necessidades;

d) Triggers (Gatilhos): consiste em procedimentos definidos pelo usuário para serem executados na ocorrência de alguma transação de manipulação de dados em tabelas, visando assim, manter a integridade referencial do banco de dados;
e) Stored Procedures: são funções armazenadas, que podem ser executadas dentro do servidor, evitando tráfego na rede, aumentando a velocidade de processamento e a consistência de dados, podendo estas ser escritas em várias linguagens de programação (PL/PgSQL, Perl, Python, Ruby, e outras);
f) Tablespaces: permite criar áreas de armazenamento ou tabelas temporárias durante uma conexão;
g) Transações: oferece suporte a múltiplas transações concorrentes entre usuários;
h) Log de Transação: é possível realizar o arquivamento e a restauração de bancos de dados a partir de logs de transação;
i) Backup: dispõe de recurso para gerar cópias de segurança sem a necessidade de desconectar o banco de dados;
j) Outros recursos: esquemas (schemas), integridade referencial, extensões para dados geoespaciais, indexação de textos, xml e várias outras, conexões SSL, pontos de salvamento (savepoints), commit em duas fases, entre outros.

Etapas

Principais fases na busca e implementação de uma solução para este processo de migração de SGBD, a seguir:

a) Estudo sobre PostgreSQL e Oracle: o primeiro passo na iniciação deste trabalho foi um levantamento minucioso de informações técnicas sobre os SGBD PostgreSQL e Oracle, tais como: instalação, configuração, recursos, funcionalidades, sintaxe de comandos, entre outros
b) Implantação do PostgreSQL: obtidos os subsídios necessários, foi realizada a instalação da ferramenta, além da configuração imprescindível para o adequado funcionamento do PostgreSQL, visando a preparação para abrigar a base de dados Agroclima;
c) Implantação do Oracle: com a preparação do servidor PostgreSQL, os trabalhos foram direcionados para a plataforma atual. Com a intenção de se obter um ambiente de testes, foi instalado o SGBD Oracle 9i e realizado a configuração mínima indispensável para a replicação do Banco de Dados Agroclima para fins de testes;

d) Identificação e teste com possíveis soluções: com a infra-estrutura computacional preparada, o passo seguinte foi buscar possíveis soluções por intermédio de ferramentas que facilitassem o processo de migração e avaliá-las;

e) Implementação da melhor solução: por fim, selecionada uma dentre as soluções aferidas, aquela que apresentou maior sucesso na efetivação da migração foi implementada;

f) Testes de desempenho: esta é sem dúvida uma das etapas mais importantes, pois tem como objeto disponibilizar informações e resultados obtidos em testes, que mostrarão o desempenho do banco de dados Agroclima na plataforma PostgreSQL frente à plataforma Oracle;

g) Otimização e Tuning: essa fase consiste na busca por um melhor desempenho do PostgreSQL, por meio da configuração e ajuste de parâmetros do SGBD que afetam o processamento e gerenciamento de memória do banco de dados;

h) Plano de Backup: segurança é um item imprescindível quando o assunto é informação, logo, é de vital importância a elaboração de uma rotina de backup para o banco de dados, o que será apresentada no item Criação e restauração de cópias de segurança.

Processo de Migração do SGBD

Dando início aos trabalhos foco deste projeto, serão detalhadas todas as etapas percorridas ao longo do processo de conversão do Banco de Dados Agroclima da plataforma Oracle para o PostgreSQL.

Hardware utilizado

Para desenvolvimento desta aplicação foi utilizado um computador com processador AMD Turion 64 Mobile de 2 GHz e memória RAM de 1,37 GB.

Sistema operacional utilizado

O computador utilizado como servidor tinha como sistema operacional o Microsoft Windows XP Professional versão 2002 Service Pack 2.

Restauração do BD Agroclima no Oracle

No intuito de se dispor de um servidor a ser utilizado como ambiente de testes, foi instalado o SGBD Oracle 9i Enterprise Manager Release 9.2.0.1.0, de maneira tal, que se pudesse realizar a replicação do Banco de Dados

Agroclima. Esta replicação foi realizada a partir do arquivo Dump binário "agroclima.dmp" gerado por meio da ferramenta Export do Oracle e restaurado via Import deste mesmo:

a) codificação do SGBD: é extremamente importante que na criação do banco de dados de destino, se tenha o cuidado em manter a mesma codificação de caractere do banco de dados de origem, o qual se deseja restaurar, evitando incompatibilidade de linguagem. Logo, atendendo a esse requisito, o Banco de Dados Agroclima foi ajustado para a seguinte Codificação: WE8ISO8859P1;

b) criação do banco de dados: o esquema do banco de dados a ser restaurado será armazenado no Database inicial do Oracle, criado no ato da configuração do SGBD com o Alias[1] e SID[2] fornecidos durante o processo de instalação do Oracle, no nosso caso, o Database foi nomeado de DBAgro;

c) criação de usuário: deve ser criado também o usuário proprietário do banco de dados, preferencialmente antes de se iniciar o processo de restauração propriamente dito. Dessa forma, foi criado, por meio do Enterprise Manager Console do Oracle, o usuário "agroclima" com senha "agroclima", sendo atribuído a este, permissões de "DBA" para criar database e usuários;

d) restauração via Import: o Oracle Import consiste em um utilitário para leitura e restauração de bancos de dados ou esquemas, completos ou parciais a partir de arquivos originados pelo Oracle Export. Para visualizar todos os parâmetros do Oracle Import, ou para maior detalhamento sobre os parâmetros, consultar a documentação de utilitários Oracle (ORACLE CORPORATION, 2001). A seguir apresenta-se a linha de comando utilizada para a importação do Banco de Dados Agroclima, executada a partir de uma chamada do "Prompt de Comandos" (Fig. 1).

```
IMP agroclima/agroclima FULL=Y FILE=E:\Oracle\Dump\agroclima.dmp SHOW=Y
```

Fig. 1. Comando Import para restaurar BD Agroclima.

[1] Alias - Rótulo de identificação para os bancos de dados no Oracle.

[2] SID - Código de identificação agregado aos bancos de dados no Oracle.

Preparação do PostgreSQL

Este item trata-se dos procedimentos necessários para a configuração e preparação do SGBD PostgreSQL para receber o Banco de Dados Agroclima. Descreve-se a seguir as configurações exigidas e demais orientações pertinentes:

a) codificação do SGBD: assim como ocorreu na restauração do BD na plataforma Oracle, deve-se atentar para manter o mesmo padrão de codificação de caractere do banco de dados de origem, preservando assim a compatibilidade de linguagem. Deste modo, e em conformidade com a documentação do PostgreSQL, foi adotado como padrão a codificação LATIN1 que é equivalente à WE8ISO8859P1 do Oracle. Este ajuste é realizado durante o processo de instalação, porém, pode também ser efetuado por meio do: menu "Tools\ServerConfiguration\postgresql.conf" no PgAdmin[3], devendo necessariamente, após alterar e salvar o parâmetro, recarregar a configuração do PostgreSQL; na criação do banco de dados; ou em tempo de execução no terminal psql ou no próprio PgAdmin III, através dos comandos SQL (Fig. 2).

```
SET client_encoding TO 'LATIN1';
SHOW client_encoding;
```

Fig. 2. Script para configurar e visualizar codificação do cliente.

b) criação de usuário: o usuário proprietário do BD deve ser criado, de preferência, antes do banco de dados e poderá ser gerado via o programa cliente PgAdmin III ou através do terminal interativo *psql*. Segue o script de comandos SQL via PSQL, para criação do usuário "agroclima", ao qual são atribuídas permissões para criar bancos de dados e usuários (Fig. 3).

```
CREATE USER agroclima
  WITH PASSWORD 'agroclima'
     CREATEDB
     CREATEUSER;
```

Fig. 3. Script de criação do usuário Agroclima.

[3] PgAdmin III - Ferramenta gráfica para gerenciamento do banco de dados PostgreSQL.

c) criação do banco de dados: a criação do banco de dados se faz necessária pelo fato de não ser possível a geração deste no ato da restauração, pois é preciso que o Database de destino já exista para que se inicie o processo de migração. Segue o comando SQL para criação do Banco de Dados, que por concepção será nomeado de "dbagroclima" simplesmente para evitar conflito no entendimento com a nomenclatura do Esquema (Fig. 4).

```
CREATE DATABASE dbagroclima
  WITH OWNER = agroclima
     ENCODING = 'LATIN1';
```

Fig. 4. Script de criação do BDAgroclima.

d) criação do esquema: finalizando a preparação do PostgreSQL para início da migração é preciso ainda criar um novo esquema padrão específico para nossa base de dados, uma vez que o esquema genérico "public" é o padrão, logo, todo e qualquer objeto importado para o nosso Database BDAgroclima seria automaticamente inserido neste. Para solucionar essa ambigüidade, será criado um novo esquema denominado Agroclima – por isso o banco de dados ter sido nomeado de BDAgroclima, visando dar maior clareza ao entendimento – e estabelece-se esse novo esquema como padrão em substituição ao esquema "public", que será excluído. As Fig. 5 e 6 apresentam os comandos SQL para criação e exclusão dos respectivos esquemas mencionados anteriormente. Para tornar o esquema Agroclima padrão é necessário acessar o menu "Tools\ServerConfiguration\postgresql.conf" no PgAdmin, devendo necessariamente, após alterar e salvar o parâmetro, recarregar a configuração do servidor PostgreSQL (Fig. 5).

```
DROP SCHEMA public CASCADE;
```

Fig. 5. Script de criação do esquema Agroclima.

```
CREATE SCHEMA agroclima AUTHORIZATION agroclima;
```

Fig. 6. Script de exclusão do esquema Public.

Ferramentas avaliadas

Na busca por uma solução mais prática, ágil e viável foram testadas algumas ferramentas que possuíam propósitos e funcionalidades que pudessem atender a essa demanda. Segue o detalhamento e a análise quanto à utilização de tais ferramentas.

a) Ora2Pg: trata-se de um módulo *open source* orientado a objeto, um interpretador Perl, com a finalidade de converter bancos de dados e esquemas de Oracle para PostgreSQL. É mencionada na comunidade do PostgreSQL, entretanto, a carência de documentação e de informações sobre a ferramenta, dificultam a sua implementação. Foram realizados alguns testes, contudo não houve progresso na sua instalação devido à falta de informação disponível;

b) DBTools Manager Professional versão Freeware: é uma ferramenta freeware para gerenciamento de banco de dados com muitas funcionalidades e com suporte aos principais SGBD conhecidos, tais como, Oracle, DB2, MySQL, SyBase, SQL Server, além do próprio PostgreSQL. Nos testes realizados com o DBTools Manager, todo o esquema Agroclima foi importado, porém, não foi recuperado os dados das tabelas com volume superior a 1.000.000 de registros. Logo, essa ferramenta não pode ser utilizada como solução única.

c) Data Transformation Service (DTS): é um utilitário do Microsoft SQL Server[4] com suporte a diversos tipos de armazenamento como os SGBD Oracle e PostgreSQL, Driver ODBC, Arquivos dbf ou textos, entre outros. O DTS possui as ferramentas Export Data e Import Data para exportação, importação e transformação de dados. Através do DTS Export Data, o processo de migração foi bem sucedido, assim, essa foi a ferramenta adotada como solução para migrar o banco de dados Agroclima do Oracle para PostgreSQL. No item a seguir, apresenta-se os procedimentos realizados para obter os resultados esperados.

Solução encontrada para migração em plataforma Windows

Uma vez concluída a análise e procedimentos iniciais, será discutido a partir daqui, a fórmula encontrada para a conclusão dos trabalhos deste projeto-piloto. A solução adotada baseia-se na importação automática das tabelas menores via utilitário DTS, na importação das tabelas maiores via comando SQL COPY dos arquivos CSV[5] também gerados com DTS, e por fim nos ajustes necessários na DDL[6] do banco de dados BDAgroclima. Seguem os

[4] Foi utilizado nos testes a versão trial (licença de uso para 30 dias) do Microsoft SQL Server 2000 Evaluation.
[5] CSV - ou Comma Separated Value é uma extensão para arquivo tipo texto com valores separados por vírgula.
[6] DDL - ou Data Definition Language consiste na linguagem SQL de definição de dados e de criação do banco.

Gerar script DDL

É recomendado que se tenha disponível o script DDL com todos os comandos SQL de criação do banco de dados, visando facilitar eventuais ajustes ou incrementar elementos ou regras não contemplados pela ferramenta utilizada na conversão de plataforma. Esse roteiro é possível ser extraído a partir de uma das ferramentas descritas a seguir, observando-se a necessidade de se fazer ajustes de compatibilidade da linguagem SQL, já que cada SGBD possui alguns comandos específicos e particularidades próprias:

a) Oracle: o SGBD Oracle 9i oferece recursos por meio do Enterprise Manager Console para gerar script da DDL dos elementos do banco de dados. Contudo, deve-se atentar para as alterações necessárias para tornar o script compatível ao PostgreSQL, tais como tipos de dados, caractere finalizador de comando e o uso de aspas em identificadores[7] para diferenciação de caracteres maiúsculos e minúsculos.

b) DBTools Manager Professional versão Freeware: apesar de não concluir com a migração de todo o banco de dados, esta ferramenta dispõe de bons recursos de administração de banco de dados, que possibilitam a geração de script DDL com os comandos SQL de criação de todo o banco, inclusive seus objetos, e com a vantagem deste roteiro já estar totalmente compatível para o PostgreSQL.

c) DTS: é possível também através da ferramenta DTS, extrair comandos SQL de criação de tabelas, devendo-se atentar às alterações de compatibilidade como tipos de dados e o uso de aspas em identificadores. Entretanto, não conterá os comandos de todos os elementos do banco de dados. Logo, não deve ser tratada como a melhor opção para esta finalidade.

Importação direta ao PostgreSQL via DTS

Neste item descreve-se o início ao processo de migração de fato, mostrando o roteiro para a utilização da ferramenta. Nesta etapa, serão criadas todas as tabelas no PostgreSQL. Todavia, no que diz respeito à importação dos dados, haverá uma limitação na transferência dos dados para tabelas com mais de 1.000.000 de registros. Assim, as tabelas com volume superior a este quantitativo serão importadas por outro recurso que será apresentado a seguir. Para iniciar, deve-se acionar o DTS Import/Export localizado no diretório ou no menu do SQL Server:

[7] Nomes usados para referenciar tabelas, colunas ou outros objetos de banco de dados.

a) origem dos dados: o primeiro passo no DTS é selecionar o Driver ODBC para Oracle, e informar o DSN[8] do Database que se deseja exportar, juntamente com o usuário e senha do respectivo banco de dados (Fig. 7).

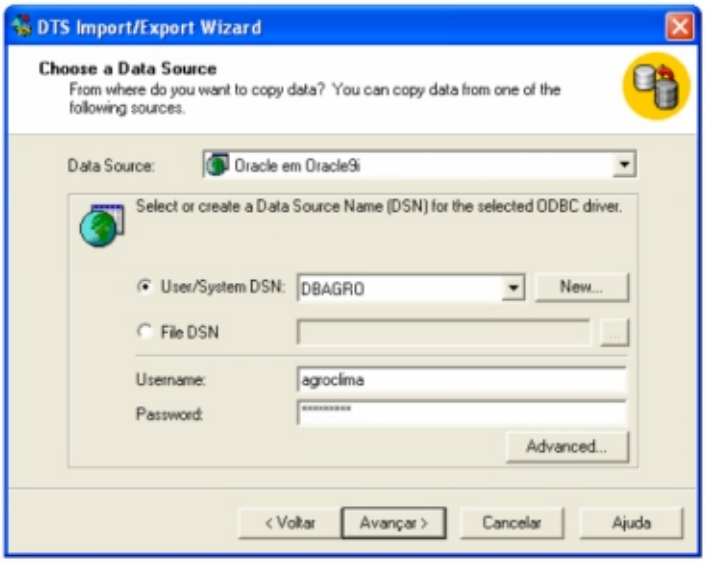

Fig. 7. Seleção do BD de origem no DTS.

b) driver ODBC: para conexão com o PostgreSQL é preciso criar um DSN de conexão ao banco de dados proposto pelo usuário. Para isso, deve-se acessar o Administrador de fonte de dados ODBC, adicionar uma fonte de dados com o driver PostgreSQL ANSI e fornecer os parâmetros de conexão à base, como mostrado na Fig. 8;

c) destino dos dados: finalizado a configuração do driver ODBC, a próxima etapa do DTS é exatamente a seleção do DSN DBAgroclima para o driver PostgreSQL ANSI (Fig. 9). Para este DSN, não é necessário informar usuário e senha, uma vez que este já detém as credenciais de autenticação para conexão ao banco de dados;

[8] DSN - ou Data Source Name é o ponteiro que armazena informação sobre a conexão com uma base de dados requerida pelo driver ODBC, tais como: tipo de servidor, localização, porta, autenticação, entre outros.

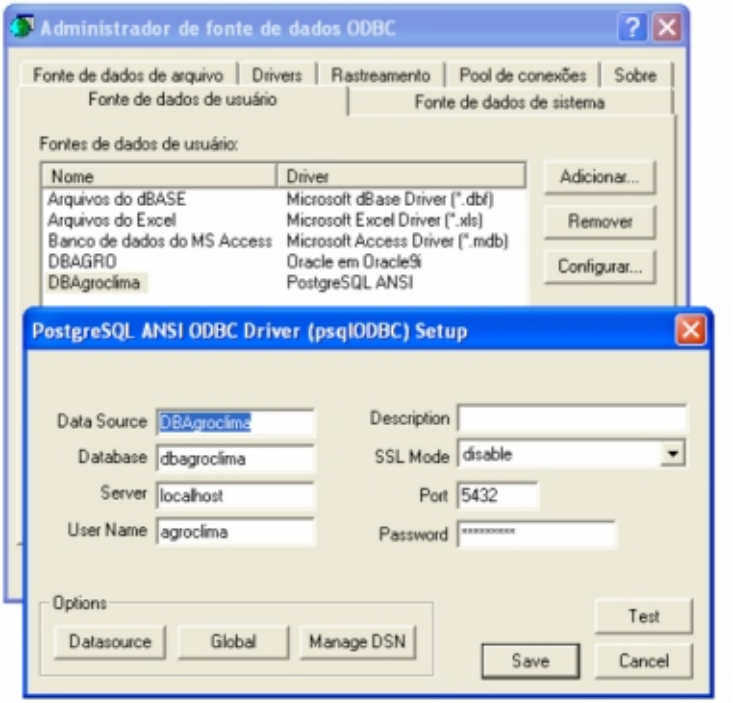

Fig. 8. Criação de DSN para BDAgroclima na ODBC.

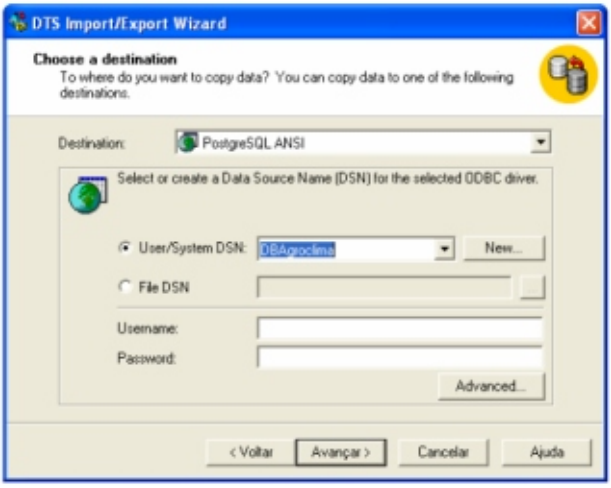

Fig. 9. Seleção do BD de destino no DTS.

d) especificação do tipo de cópia: aqui pode-se especificar ao DTS o que se deseja transportar para o seu destino. A opção “Copy Table...” irá criar a(s) tabela(s) selecionadas, com todos os seus campos e em seguida carregará todos os seus dados, de acordo com a limitação do quantitativo de registros já mencionada. Enquanto que com a opção “Use a query...” o usuário poderá selecionar os campos desejados e aplicar filtros, porém esta alternativa é recomendada apenas para um número pequeno de tabelas, já que deve ser feito para cada tabela alvo por vez. Como se pretende importar todo o banco de dados, selecione a opção “Copy Table...”, como ilustrado na Fig. 10.

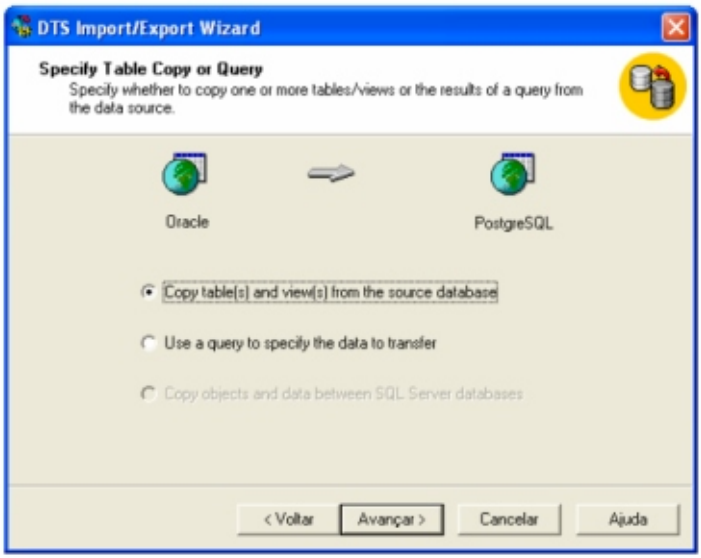

Fig. 10. Seleção do BD de destino no DTS.

e) seleção de tabelas: e para finalizar esta primeira etapa de importação, a última etapa do DTS que se trata da seleção e transformação das tabelas desejadas. A transformação consiste nas alterações quanto à estrutura da(s) tabela(s), e está disponível com um clique na coluna “Transform”, onde poderá ser modificado o script DDL de criação da tabela. Em virtude de o SGBD PostgreSQL possuir linguagem *case sensitive*, ou seja, faz diferenciação entre caracteres maiúsculos e minúsculos, é importante se ater a este detalhe. Deste modo, foi adotada neste trabalho a remoção das aspas duplas (“”) e a alteração dos nomes de identificadores de maiúsculo para minúsculo, preservando assim, maior compatibilidade com o padrão SQL.

Após este passo (Fig. 11), dar-se-á início a tela de progresso da importação, na qual será possível acompanhar o andamento e verificar notificações de sucesso ou de erro, caso ocorra algum.

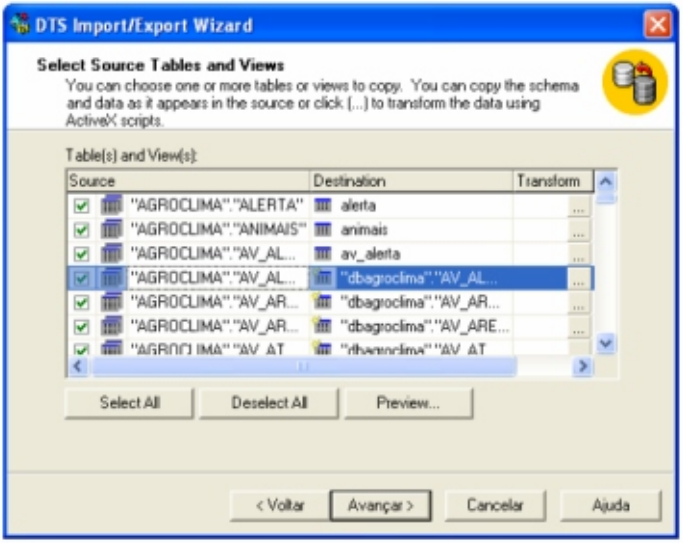

Fig. 11. Seleção e transformação de tabelas no DTS.

Geração de arquivos CSV via DTS

Dando continuidade ao processo de migração, iniciará a solução para importação das tabelas com grande volume de dados nas quais, por problemas de limitação, não puderam ser importadas diretamente para a plataforma PostgreSQL. A solução encontrada foi importar os dados de tais tabelas para arquivos com valores separados por vírgula (csv), e posteriormente importá-los ao PostgreSQL via comando SQL Copy. A seguir serão mencionados os passos para gerar os arquivos no formato csv:

a) origem dos dados: repetir os passos mencionados no subitem "Importação direta ao PostgreSQL via DTS", alínea a;

b) destino dos dados: selecionar como destinação dos dados a opção “Text File” e informar o nome do arquivo a ser gerado (com a extensão “.csv”) com o diretório onde será gravado (Fig. 12);

c) especificação do tipo de cópia: repetir os passos mencionados no subitem “Importação direta ao PostgreSQL via DTS”, alínea d;

d) configuração do arquivo CSV: para concluir a geração do arquivo csv, deve-se informar a tabela que se deseja importar e os parâmetros de configuração. Neste trabalho, adotou-se o uso do delimitador de coluna *semicolon* ou ponto-e-vírgula (;) e o qualificador de textos, *Double quote* ou aspas duplas (“”), Fig. 13.

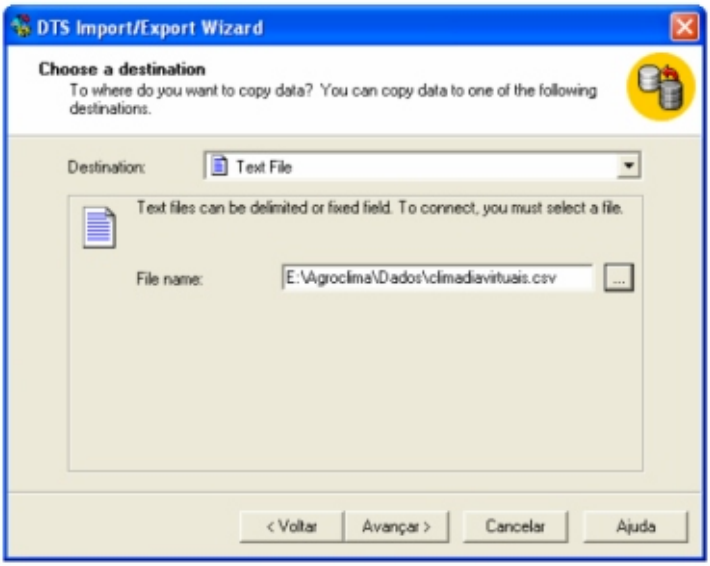

Fig. 12. Criação de arquivo CSV no DTS.

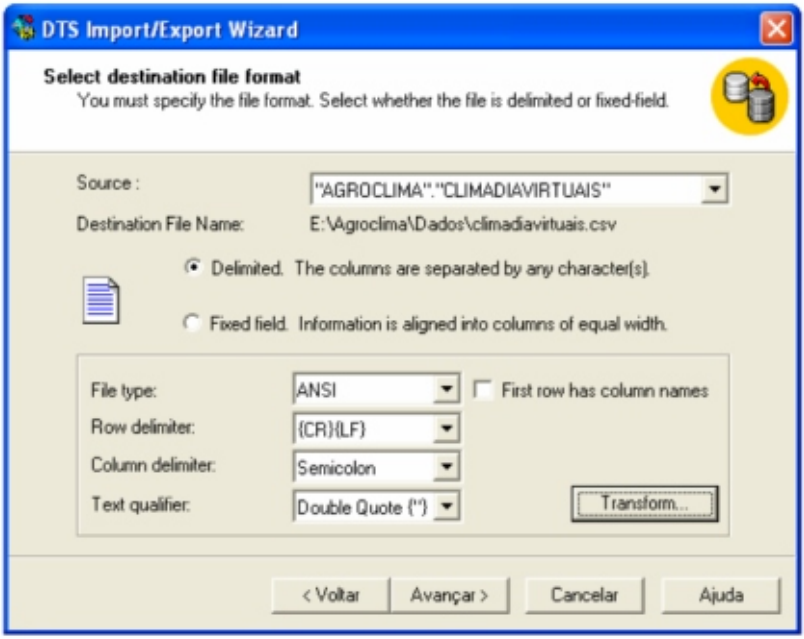

Fig. 13. Configuração de arquivo CSV no DTS.

Ao avançar este passo, dar-se-á início a tela de progresso da importação, onde será possível acompanhar o andamento e verificar notificações de sucesso ou de erro, caso ocorra algum. Deve-se repetir estas etapas para todas as tabelas com grandes volumes de dados, que se deseja importar.

Importação de arquivos CSV via SQL

Uma vez gerados, os arquivos csv serão importados através do comando SQL Copy no terminal psql ou no PgAdmin. É aconselhado limpar a tabela antes da execução do comando copy, uma vez que poderiam existir registros importados durante o processo inicial, o que poderia ocasionar erro em virtude da duplicação de chave primária. Além disso, recomenda-se ainda configurar o parâmetro “client_encoding” como “LATIN1”, alterando assim a codificação do cliente de maneira a preparar os dados de entrada com a mesma codificação do banco. A Fig. 14 apresenta uma operação de importação, utilizando o comando Copy com a configuração dos parâmetros necessários para a leitura do arquivo csv: ponto-e-vírgula como delimitador e aspas duplas como qualificador de texto. Esta operação deverá ser repetida para todos os arquivos csv gerados.

```
DELETE FROM CLIMADIAVIRTUAIS;

SET CLIENT_ENCODING TO 'LATIN1';

COPY CLIMADIAVIRTUAIS FROM 'E:/Agroclima/Dados/climadiavirtuais.csv'
  DELIMITER AS ';'
  NULL AS ''
  CSV QUOTE AS '"';
```

Fig. 14. Importação de tabela via comando Copy.

Restauração dos demais elementos do BD via SQL

A conclusão do processo de migração do banco de dados para PostgreSQL depende do script do BD gerado no subsubitem “Gerar script DDL”. Este último procedimento se deve ao fato do utilitário DTS carregar apenas a estrutura simples das tabelas e os seus dados, não contemplando assim, índices, chaves primárias e estrangeiras. Logo, tais elementos serão criados via comandos SQL, executados a partir do terminal psql ou do gerenciador gráfico PgAdmin. Assim, para finalizar o banco BDAgroclima é preciso abrir o arquivo contendo o script DDL de todo o banco de dados, preferencialmente no PgAdmin, e executar as linhas de comando referentes às chaves primárias, aos índices, às chaves estrangeiras e por fim, às visões.

Solução alternativa ao uso do SQL Server em plataformas Windows e Linux

Na busca por maior rapidez na conclusão dos trabalhos, foi adotada a utilização de software proprietário, o DTS Export/Import, que é um utilitário integrante do pacote de SGBD Microsoft SQL Server. Entretanto, foi também analisada outra solução alternativa (podendo ser aplicada tanto em plataformas Windows, quanto em Linux), que se baseia: na montagem do esquema do banco de dados a partir da execução dos comandos SQL do script DDL, já mencionado anteriormente, deixando apenas a criação de chaves estrangeiras para após a importação dos dados, visando evitar conflitos de relacionamentos; e na geração dos arquivos de texto com campos separados por vírgula (.csv) por meio do Editor de conteúdo no Console do Oracle. A geração de arquivos csv deverá ocorrer repetidamente para cada tabela existente que se deseja exportar, o que torna esta solução um tanto trabalhosa a depender do quantitativo de tabelas.

E para finalizar esta solução, os arquivos csv foram importados com a utilização do comando COPY do PostgreSQL já apresentado anteriormente.

Otimização de Desempenho do PostgreSQL

O desempenho de um banco de dados depende de vários fatores e não unicamente da escolha do SGBD. É primordial que o banco de dados, antes de tudo, tenha uma boa modelagem, e que este tenha passado por uma normalização mínima (1ª, 2ª e 3ª Forma Normal), pois um BD mal modelado está sujeito a sérios problemas de integridade, segurança e desempenho. Mas partindo da premissa que se tenha um Database bem desenhado, é preciso ter cuidado na criação de consultas Select, ou até mesmo de visões, pois quando não montadas da maneira ideal, poderão ocasionar uma grande perda de desempenho, apesar do resultado da consulta ser o esperado. Logo, recomenda-se uma atenção especial com o uso de comandos Select encadeados, junções, filtros em campos não-indexados, entre outros.

Abandonando o campo do SGBD, é necessário também se ater ao sistema operacional, uma vez que algumas configurações e parâmetros deste possuem impacto direto no desempenho do banco de dados, tais como: funcionamento da CPU, alocação de memória, gerenciamento de dispositivos I/O, controle de processos concorrentes, etc.

Não menos importante que os anteriores, e embora o requisito mínimo para instalação do SGBD PostgreSQL seja de apenas 8 MB de memória RAM disponível, 100 MB de espaço em disco e sistema operacional Windows/Linux/Unix/BSD, é imprescindível contar com um bom alicerce de hardware, com uma boa capacidade de processamento e principalmente uma boa envergadura de armazenamento e memória.

Há ainda um outro imponente fator de impacto na performance de um SGBD, que é o chamado *Tuning*, que na verdade contempla alguns dos elementos mencionados anteriormente. O processo de Tuning consiste em um refinamento na configuração de todo o ambiente do SGBD, que abrange a revisão das consultas SQL, a configuração adequada do sistema operacional e o inventário de parâmetros disponibilizados pelo próprio SGBD PostgreSQL, que visam otimizar o desempenho do banco de dados. Como por exemplo, o parâmetro “max_connections” que limita a quantidade máxima de conexões concorrentes ao servidor PostgreSQL, o valor padrão é 100, mas pode ser ajustado (para mais ou menos) de acordo com a quantidade de memória disponível.

A seguir será apresentado alguns destes principais parâmetros oferecidos pelo PostgreSQL. Tais parâmetros encontram-se (habilitados ou desabilitados) no arquivo de configurações “postgresql.conf” disponível no diretório de dados do PostgreSQL ou no menu “Tools\ServerConfiguration” do PgAdmin III.

Otimização de memória

a) **shared_buffers:** *"Define o número de buffers de memória compartilhada, utilizados pelo servidor de banco de dados. O valor típico é 1000, mas pode ser menor se a configuração do núcleo do sistema operacional não tiver capacidade para suportar este valor (conforme determinado durante o initdb). Cada buffer possui 8192 bytes, a menos que seja escolhido um valor diferente para o tamanho do buffer ao construir o servidor. O valor definido deve ser pelo menos igual a 16, assim como pelo menos duas vezes o valor de max_connection. Entretanto, normalmente é necessário definir um valor bem maior que o mínimo para obter um bom desempenho. São recomendados valores de alguns poucos milhares para instalações de produção. Somente pode ser definido na inicialização do servidor. O aumento deste parâmetro pode fazer com que o PostgreSQL requisite mais memória compartilhada System V que o permitido pela configuração padrão do sistema operacional"* (POSTGRESQL GLOBAL DEVELOPMENT GROUP, 2005).

b) **work_mem:** *"Especifica a quantidade de memória utilizada pelas operações internas de classificação e tabelas de dispersão (hash tables), antes de alternar para arquivos temporários em disco. O valor é especificado em kilobytes e o valor padrão é 1024 kilobytes (1 MB). Deve ser observado que, em uma consulta complexa, podem ser executadas várias classificações ou operações de hash em paralelo; cada uma pode utilizar tanta memória quanto especificado por este parâmetro, antes de colocar os dados em arquivos temporários. Além disso, diversas sessões em execução podem estar fazendo operações de classificação simultaneamente. Portanto, a memória total utilizada pode ser várias vezes o valor de work_mem; é necessário ter este fato em mente ao escolher o valor. As operações de classificação são utilizadas por ORDER BY, DISTINCT e junções por mesclagem (merge joins). As tabelas de dispersão (hash) são utilizadas em junções de dispersão (hash joins), agregações baseadas em dispersão (hash-based tablesaggregation), e no processamento baseado em dispersão (hash-based processing) de subconsultas IN"* (POSTGRESQL GLOBAL DEVELOPMENT GROUP, 2005).

c) **maintenance_work_mem:** *"Especifica a quantidade máxima de memória que pode ser utilizada pelas operações de manutenção, como VACUUM, CREATE INDEX e ALTER TABLE ADD FOREIGN KEY. O valor é especificado em kilobytes, e o valor padrão é 16384 kilobytes (16 MB). Uma vez que somente pode ser executada uma destas operações*

por vez em uma mesma sessão de banco de dados, e o servidor normalmente não possui muitas delas acontecendo ao mesmo tempo. É seguro definir este valor significativamente maior que work_mem. Definições maiores podem melhorar o desempenho do comando VACUUM e da restauração de cópias de segurança" (POSTGRESQL GLOBAL DEVELOPMENT GROUP, 2005).

Otimização de disco e de WAL (Write Ahead Log)

a) **max_fsm_pages:** *"Define o número máximo de páginas em disco para as quais o espaço livre será acompanhado no mapa de espaço livre compartilhado. São consumidos seis bytes de memória compartilhada para cada encaixe de página. O valor definido deve ser maior que 16 * max_fsm_relations. O valor padrão é 20000. Somente pode ser definido na inicialização do servidor"* (POSTGRESQL GLOBAL DEVELOPMENT GROUP, 2005).

b) **wal_buffers:** *"Número de buffers de página em disco alocados na memória compartilhada para os dados do WAL. O valor padrão é 8. Esta definição somente precisa ser grande o suficiente para conter a quantidade de dados do WAL gerados por uma transação típica. Somente pode ser definido na inicialização do servidor"* (POSTGRESQL GLOBAL DEVELOPMENT GROUP, 2005).

c) **effective_cache_size:** *"Define o tamanho efetivo presumido pelo planejador acerca do cache de disco disponível para uma única varredura de índice. É um dos elementos usados na estimativa do custo de utilização de um índice; um valor maior torna mais provável a utilização de uma varredura de índice, enquanto um valor menor torna mais provável a utilização de uma varredura seqüencial. Ao definir este valor devem ser considerados os buffers de memória compartilhada do PostgreSQL, e a parcela do cache de disco do sistema operacional que será utilizada pelos arquivos de dado do PostgreSQL. Também deve ser levado em consideração o número esperado de comandos simultâneos que utilizam índices diferentes, uma vez que estes têm de compartilhar o espaço disponível. Este parâmetro não tem efeito sobre o tamanho da memória compartilhada alocada pelo PostgreSQL, nem reserva cache de disco do sistema operacional; é usado apenas para finalidades de estimativa. O valor é medido em páginas de disco, normalmente com 8192 bytes cada. O valor padrão é 1000*" (POSTGRESQL GLOBAL DEVELOPMENT GROUP, 2005).

d) Random_page_cost: *"Define a estimativa do planejador de comandos do custo da busca não seqüencial de uma página de disco. É medido como um múltiplo do custo da busca seqüencial de uma página. Um valor maior torna mais provável a utilização de uma varredura seqüencial, enquanto um valor menor torna mais provável a utilização de uma varredura de índice. O valor padrão é 4*" (POSTGRESQL GLOBAL DEVELOPMENT GROUP, 2005).

Configurações utilizadas para o BDAgroclima

A Tabela 2, apresenta as alterações adotadas no PostgreSQL, visando extrair um maior desempenho do banco de dado Agroclima para essa plataforma. É importante ressaltar que após as alterações pertinentes, é necessário salvá-las e recarregar (ou seja, parar e iniciar o processo mestre) o servidor PostgreSQL.

Tabela 2. Configurações de Tuning para BDAgroclima.

PARÂMETRO	UNIDADE	VALOR PADRÃO	VALOR NOVO
shared_buffers	8KB	2048 (=16MB)	8192 (=64MB)
work_mem	KB	1024 (=1MB)	2048 (=2MB)
maintenance_work_mem	KB	16384 (=16MB)	65536 (=64MB)
max_fsm_pages	páginas	20000	120000
wal_buffers	páginas	8	64
effective_cache_size	páginas	1000	32768
random_page_cost	custo	4	2

Criação e Restauração de Cópias de Segurança

Quando o assunto é dados, é indispensável ter um bom plano de segurança para protegê-los. E o PostgreSQL dispõe de ferramentas que facilitam a realização de backup e sua posterior restauração.

Cópias de segurança

O utilitário pg_dump é uma rotina de backup que gera um arquivo texto ou binário contendo todos os comandos SQL capazes de restaurar um banco de dados total ou parcialmente. O arquivo dump gerado pelo pg_dump corresponde à imagem exata do banco de dados no momento da execução do comando e contém sintaxe SQL de criação de tabelas, índices, chaves primárias e estrangeiras, visões, funções, seqüências e todos os demais

elementos que compõem o banco, inclusive todos os dados existentes. A Fig. 15 mostra script utilizado para gerar arquivo dump do banco de dados BDAgroclima.

```
pg_dump -f
C:\PostgreSQL\Agroclima\Backup\dbagroclima_full_20070727.dmp -F c -v -h
localhost -p 5432 -u dbagroclima
```

Fig. 15. Gerar arquivo dump de backup de um banco de dado.

O pg_dumpall é similar ao comando pg_dump, contudo, esse utilitário efetuará o backup de todos os bancos de dados existentes no servidor PostgreSQL. A Fig. 16 mostra as formas de linha de comando para execução do pg_dumpall.

```
pg_dumpall > arquivo_de_saída
ou
pg_dumpall parâmetros
```

Fig. 16. Gerar arquivo dump de backup todos os bancos.

Para automatizar a tarefa de backup, basta utilizar o agendador de tarefas do sistema operacional e configurar conforme a necessidade.

Restauração de cópias de segurança

A restauração de bancos de dados por meio de arquivo dump gerado por pg_dump ou pg_dumpall é extremamente simples. Entretanto é necessário a criação do(s) banco(s) através de comando SQL “CREATE DATABASE” no terminal psql ou do utilitário createdb no prompt, uma vez que o comando somente restaura os elementos internos do banco de dados. Feito isso, basta apenas passar os parâmetros ao utilitário pg_restore do que se deseja restaurar, conforme Fig. 17.

```
psql nome_do_banco_de_dados < arquivo_de_entrada
```

Fig. 17. Restaurar banco de dado a partir de arquivo dump.

Avaliação e Resultados

Os resultados obtidos e avaliados nesse processo de migração serão apresentados deslumbrando os pontos fortes e fracos, dificuldades encontradas e oportunidades de melhoria.

Diagnóstico de performance

A Tabela 3 apresenta um cenário da execução de algumas consultas SQL no Oracle e no PostgreSQL com e sem ajustes de tuning, de maneira que se possa confrontá-los e comparar o desempenho do segundo em relação ao primeiro. Foram aplicadas múltiplas consultas simulando as mais distintas situações e utilizando diversos comandos e operadores, no intuito de se verificar o comportamento de cada um em rotinas operacionais.

Por convenção, adotou-se o uso de amostragem para medir o tempo de execução das consultas, sendo realizado então três medições de tempo (T1, T2 e T3) para cada SGBD, a fim de que se evitasse a ocorrência de desvio-padrão muito acima da margem de erro aceitável, e considerando, dentre essas, o maior dos tempos obtidos (Tmax) de cada um, para então obter o fator de desempenho (FD), que expressa o tempo do PostgreSQL em relação ao tempo do Oracle. Os resultados podem ser conferidos na Tabela 3.

Os dados obtidos com o diagnóstico foram bastante satisfatórios, visto que o PostgreSQL apresentou desempenho bem próximo ao Oracle, obtendo inclusive, resultados superiores em algumas consultas. Observe então algumas considerações sobre os resultados obtidos no diagnóstico de desempenho:

a) em geral, para bases de dados históricas o SGBD Oracle é infinitamente superior, haja vista a sua maior utilização e melhor gerenciamento do armazenamento em cache, o que lhe concede um expressivo ganho em tempo. Entretanto, para bases de dados com alterações freqüentes e que exijam uma atualização real dos dados é necessário observar minuciosamente os tipos de consultas, cláusulas e operadores com maior índice de utilização, pois há um significativo ganho ora para um, ora para o outro, a depender do tipo de consulta empregada;
b) para consultas em tabelas menores (na casa de milhares de registros) utilizando operador "=" o SGBD PostgreSQL mostrou desempenho bem superior ao Oracle, enquanto que, para tabelas com grandes volumes (na casa de milhões de registros), ambos tiveram desempenho similar;

c) em consultas com utilização do operador LIKE, o PostgreSQL apresentou superioridade para tabelas com pequeno volume de dados e com o uso do LIKE para início de campo em tabelas com grandes volumes, enquanto o Oracle mostrou desempenho imensamente maior para uso do LIKE no meio e fim de campos;

d) o PostgreSQL obteve melhor desempenho na ordenação de dados, e para agrupamento, houve empate técnico, com ligeira vantagem para o Oracle;

e) com relação às junções, o PostgreSQL apresentou leve ganho com Subselect e imensa superioridade em LEFT JOIN. Entretanto, foi extremamente inferior no uso de INNER JOIN e RIGHT JOIN.

Tabela 3. Comparativo de desempenho de Oracle x PostgreSQL.

CONSULTA			TEMPO DE EXECUÇÃO (segundos)								
			Oracle				PostgreSQL				
Nº	QUERY	DESCRIÇÃO	T1	T2	T3	TM	T1	T2	T3	TM	FD (%)
1	SELECT * FROM estacoes WHERE idestacoes LIKE 'U%';	Consulta em campo índice usando operador LIKE (Tabela com aprox. 1000 registros)	1,078	0,218	0,203	**1,078**	0,062	0,031	0,031	**0,062**	**94,2%**
2	SELECT * FROM estacoes WHERE idestacoes = 'UNESP.002';	Consulta em campo índice usando operador "=" (Tabela com aprox. 1000 registros)	0,125	0,141	0,032	**0,141**	0,031	0,015	0,015	**0,031**	**78,0%**
3	SELECT * FROM estacoes WHERE upper(nome) = 'FORTALEZA';	Consulta em campo não índice (Tabela com aprox. 1000 registros)	0,984	2,125	1,594	**2,125**	0,031	0,016	0,016	**0,031**	**98,5%**
4	SELECT * FROM climadia WHERE idclimadia = '.CDVale.1.SP.01/01/2007';	Consulta em campo índice usando operador "=" (Tabela com aprox. 3,7 milhões registros)	0,078	0,063	0,047	**0,078**	0,094	0,015	0,015	**0,094**	-20,5%
5	SELECT * FROM climadia WHERE idclimadia LIKE '.CDVale.1.SP.01/11/%';	Consulta em campo índice usando operador LIKE no FIM (Tabela com aprox. 3,7 milhões registros)	0,500	0,469	0,453	**0,500**	51,937	51,828	53,687	**53,687**	-10637,4%
6	SELECT * FROM climadia WHERE idclimadia LIKE '.CDVale.%.01/11/2006';	Consulta em campo índice usando operador LIKE no MEIO (Tabela com aprox. 3,7 milhões registros)	2,500	1,968	1,468	**2,500**	51,609	53,250	53,031	**53,250**	-2030,0%
7	SELECT * FROM climadia WHERE idclimadia LIKE '.%.1.SP.01/11/2006';	Consulta em campo índice usando operador LIKE no INICIO (Tabela com aprox. 3,7 milhões registros)	382,360	371,453	374,313	**382,360**	53,469	53,031	52,812	**53,469**	**86,0%**
8	SELECT * FROM climadia WHERE data BETWEEN to_date('2001-03-19', 'YYYY-MM-DD') AND to_date('2001-03-21', 'YYYY-MM-DD');	Consulta em campo índice usando operador BETWEEN (Tabela com aprox. 3,7 milhões registros)	4,547	1,437	0,125	**4,547**	4,579	0,969	0,969	**4,579**	-0,7%
9	SELECT idestacoes, data, tmax FROM climadia WHERE tmax > 40;	Consulta em campo não índice utilizando operador ">" (Tabela com aprox. 3,7 milhões registros)	63,516	26,406	30,047	**63,516**	52,703	52,922	52,484	**52,922**	**16,7%**
10	SELECT idestacoes, data, tmax FROM climadia WHERE tmax > 40 ORDER BY tmax DESC;	Consulta em campo não índice utilizando operador ">" e cláusula ORDER BY (Tabela com aprox. 3,7 milhões registros)	169,797	172,625	167,719	**172,625**	53,906	52,484	53,141	**53,906**	**68,8%**
11	SELECT idestacoes, avg(tmin), avg(tmax) FROM climadia GROUP BY idestacoes;	Consulta utilizando cláusulas GROUP BY e função AVG (Tabela com aprox. 3,7 milhões registros)	49,343	43,797	44,781	**49,343**	52,594	51,610	52,812	**52,812**	-7,0%
12	SELECT idestacoes, avg(tmin), avg(tmax) FROM climadia GROUP BY idestacoes ORDER BY idestacoes;	Consulta utilizando cláusulas GROUP BY e ORDER BY e função AVG (Tabela com aprox. 3,7 milhões registros)	161,828	144,937	141,610	**161,828**	51,938	52,297	52,719	**52,719**	**67,4%**
13	SELECT climadia.idclimadia, estacoes.idestacoes FROM climadia, estacoes WHERE climadia.idestacoes = estacoes.idestacoes AND estacoes.idestacoes = '.CEMIG.00027';	Consulta simples em campo índice (Tabela com aprox. 3,7 milhões registros)	1,407	1,406	0,016	**1,407**	2,937	0,125	0,125	**2,937**	-108,7%
14	SELECT idclimadia, idestacoes, tmax, tmin, data FROM climadia WHERE climadia.idestacoes IN (SELECT idestacoes FROM estacoes WHERE idinstituicao = 'CPAC');	Consulta usando SubSelect (Tabela com aprox. 3,7 milhões registros)	2,187	0,594	0,422	**2,187**	2,172	2,172	2,172	**2,172**	**0,7%**
15	SELECT * FROM climadiavirtuais WHERE idclimadiavirtuais = '.AGRITEMPO.2190.GO.05/12/2006';	Consulta em campo índice usando operador "=" (Tabela com aprox. 9 milhões registros)	0,125	0,063	0,047	**0,125**	0,422	0,016	0,016	**0,422**	**-237,6%**

Avaliação técnica quanto ao uso do PostgreSQL

Os dados obtidos com o diagnóstico foram bastante satisfatórios, visto que o PostgreSQL apresentou desempenho bem próximo ao Oracle, obtendo até resultados superiores em algumas consultas. Durante o processo, o PostgreSQL mostrou-se uma ferramenta de boa capacidade de execução, muita flexibilidade e com muitos recursos e funcionalidades à disposição do usuário, seja em modo gráfico, seja via terminal de comando.

Aspectos positivos do processo

Alguns pontos fortes ou favoráveis obtidos durante a execução dos trabalhos:

- oportunidade de aprendizado sobre os SGBD de grande porte: Oracle 9i e PostgreSQL;
- estudo comparativo de desempenho das plataformas Oracle e PostgreSQL.

Dificuldades enfrentadas no processo

Enumera-se alguns pontos fracos, deficiências e dificuldades encontradas no decorrer do processo:

- sem dúvida, o fator dificultador nesse trabalho foi a importação de dados das tabelas com grande volume de dados, ou para ser exato, tabelas superiores a um milhão (1.000.000) de registros;
- houve também problemas com relação ao hardware disponível, devido à pequena capacidade de armazenamento, uma vez que era necessário se trabalhar com bancos replicados, exigindo grande volume de disco;
- a carência de um ponto de rede extra foi também um fator de retardo, já que se houvesse duas máquinas interligadas haveria um ganho de tempo com a alocação de tarefas paralelas;
- não houve sucesso na tentativa de se alterar o formato de data padrão do PostgreSQL de "ISO, MDY" para "SQL, DMY". Na saída obtém-se o formato desejado, porém a entrada é armazenada no padrão ISO.

Oportunidades de melhoria do processo

Algumas observações do que poderia ser melhorado, visando uma maior eficiência do processo tanto quanto aos recursos computacionais utilizados, como ao fator tempo:

- **Ora2Pg:** em virtude de a comunidade fazer menção a esta ferramenta, seria interessante estudá-la um pouco mais, apesar da carência de documentação e de poucas práticas de sucesso. Este módulo pode vir se tornar uma boa aplicação para migração de Oracle para PostgreSQL;
- **Aplicação Java:** outra prática que poderia tornar o processo de migração mais ágil seria o desenvolvimento de uma ferramenta em Java com conexão ao driver ODBC que pudesse assim, efetuar leitura em BD Oracle e reescrever em BD PostgreSQL diretamente sem envio de dados para buffer, não onerando assim o uso crítico de memória. A idéia da aplicação pode ser utilizada para plataformas Windows e Linux.

Conclusões

O processo de migração da plataforma Oracle para PostgreSQL, de um banco de dados com grande volume, é ainda atualmente, um processo um tanto árduo no mundo open source, haja vista a carência de ferramentas de grande porte para a execução de tal tarefa. Assim, é preciso reunir um conjunto de recursos para a dissolução deste processo, ou buscar soluções comerciais, ou até mesmo desenvolver rotinas e aprimorar práticas existentes como o Ora2pg. Contudo, apesar do nível de complexidade a ser enfrentado, a portabilidade para PostgreSQL é, sem margem de dúvida, uma iniciativa de sucesso, pois como mostra o próprio texto, a cada dia, empresas de todo o mundo aderem a esta plataforma.

Vale ressaltar que essa onda de crescimento na utilização do SGBD PostgreSQL, não se deve unicamente ao fato deste se tratar de um produto open source e sem custo de licença, mas principalmente pelo fato do PostgreSQL ser hoje um servidor de banco de dados de médio e grande porte, com alto desempenho, estabilidade e capacidade para suportar grandes volumes de dados, e com mais de 10 anos de desenvolvimento pleno, o que torna esse SGBD não mais uma promessa, e sim uma realidade – um produto com tanto potencial, recursos e além de tudo, gratuito e livre.

Referências

ORACLE CORPORATION. ***Oracle 9i:*** database utilies release 1 (9.0.1). Redwood City, 2001. 528 p.

ORACLE NEWS. **Desbancando a concorrência**. Disponível em: <http://www.oracle.com/partners/home/news/lad/portuguese/jun2005/desbancando.html> Acesso em: 19 jun. 2007.

POSTGRESQL GLOBAL DEVELOPMENT GROUP. **Documentação do PostgreSQL** 8.0.0. Tradução: Haley Pacheco de Oliveira. Rio de Janeiro, 2005. 1205 p.

RIBAMAR, F. S. **PostgreSQL prático***:* versão 8.1.4. São Paulo, 2006. 157 p.

Literatura Recomendada

ADAMI, T. **Otimizando bancos PostgreSQL**. Disponível em: <http://www.imasters.com.br/artigo/4406/postgresql/otimizando_bancos_postgresql_-_parte_01/> Acesso em: 01 jun. 2007.

ALMEIDA, R. **Dicas SQL*PLUS**. Disponível em: <http://www.imasters.com.br/artigo/1752/oracle/dicas_sqlplus_continuacao/> Acesso em: 01 jun. 2007.

COUTO, E. **Aumentando a performance da aplicação através da otimização de SQL**. Disponível em:<http://www.imasters.com.br/artigo/4055/bancodedados/aumentando_a_performance_da_aplicacao_atraves_da_otimizacao_de_sql/> Acesso em: 17 jul. 2007.

DBEXPERTS. **dbExperts PostgreSQL** – módulo I*:* apostila de Treinamento. São Paulo, jul. 2004. 83p.

DBEXPERTS. **dbExperts PostgreSQL** – módulo II*:* apostila de Treinamento. São Paulo, jul. 2004. 97p.

GUIMARÃES, C. C. **Fundamentos de bancos de dados**: Modelagem, projeto e linguagem SQL. Campinas: Ed. da Unicamp, 2003. 272 p.

LINUX NA REDE. **Instalando o PostgreSQL**. Disponível em: <http://www.linuxnarede.com.br/artigos/fullnews.php?id=298> Acesso em: 06 jun. 2007.

NIEDERAUER, J. **PostgreSQL***:* guia de consulta rápida. 2. ed. São Paulo: Novatec, 2004. 94 p.

ORACLE CORPORATION. **The World's Largest Enterprise Software Company**. Disponível em: <http://www.oracle.com/> Acesso em: 21 jun. 2007.

ORACLE CORPORATION. **Oracle 9i***:* SQL references – release 1 (9.0.1). Redwood City, 2001. 1492 p.

PEREIRA NETO, A. **PostgreSQL***:* técnicas avançadas: versões Open Source 7.x – soluções para desenvolvedores e administradores de bancos de dados. São Paulo: Érica, 2003. 284 p.

POSTGRESQL. **The world's most advanced open source database**. Disponível em: <http://www.postgresql.org/> Acesso em: 25 maio 2007.

POSTGRESQLBRASIL. **Sítio da comunidade PostgreSQL brasileira**. Disponível em: <http://www.postgresql.org.br/> Acesso em: 22 maio 2007.

TELLES, F. **Checklist de performance do PostgreSQL 8.0.** Disponível em: <http://www.devmedia.com.br/articles/viewcomp.asp?comp=5518> Acesso em: 01 jun. 2007.

Informática Agropecuária

Ministério da
Agricultura, Pecuária
e Abastecimento