Treinamento PostgreSQL - Aula 09

Eduardo Ferreira dos Santos

SparkGroup
Treinamento e Capacitação em Tecnologia
eduardo.edusantos@gmail.com
eduardosan.com

10 de Junho de 2013

Cronograma

Semana 1: 27 de Maio a 3 de Junho Administração de Dados

Semana 2: 4-11 de Junho Administração de Banco de Dados

Semana 3: 13-18 de Junho Alta disponibilidade

Semana 4: 19-24 de Junho Performance Tuning

Tipos de backup

SQL Dump Utilização da ferramenta pg_dump para gerar diferentes tipos de backup:

- Comandos SQL para recriação da base
  - Utiliando COPY
  - Utilizando INSERT
- Formato binário padrão do PostgreSQL (pg_dump -FC)

Backup do sistema de arquivos Compressão e empacotamento do diretório PGDATA

```
tar -czvf backup.tar.gz /usr/local/pgsql/data
```

PITR Arquivamento contínuo e *Point-in-Time Recovery*

SQL Dump

```
# Backup com COPY
pg_dump dbname > outfile

# Usando INSERT
pg_dump --inserts dbname > outfile

# Restore
psql dbname < infile

# Backup de todo o cluster
pg_dumpall > outfile
psql -f infile postgres

# Compressao enquanto gera o backup
pg_dump dbname | gzip > filename.gz
gunzip -c filename.gz | psql dbname

# Dividindo em pequenos arquivos de 1M
pg_dump dbname | split -b 1m - filename
cat filename* | psql dbname

# Formato binario do PostgreSQL
pg_dump -Fc dbname > filename
pg_restore -d dbname filename
```

Backup do sistema de arquivos

```
tar -czvf backup.tar.gz /usr/local/pgsql/data
```

- Qualquer comando para cópia do diretório PGDATA pode ser utilizado:
- Duas restrições importantes que tornam a cópia física praticamente inutilizável:
  1. É obrigatório parar o servidor para obter um backup reutilizável.
  2. Fazer backup de certos arquivos e diretórios somente não funciona. Os dados das tabelas são inúteis sem referenciar as informações de transação na tabela pg_clog/, que contém o status de todas as transações.

Resumo

Não faça backup somente utilizando o sistema de arquivos. Salvo raríssimas exceções, não vai funcionar!!!

Tempo de recuperação pg_dump[1]

- Tempo de recuperação do backup estimado?

1GB	Uns 30min?
10GB	Umas 3-4h?
100GB	1 dia inteiro?
500GB	Vários dias?

[1] Óbvio que o tempo é apenas um chute. O tempo para recuperação do backup depende de tantos fatores que não vale nem a pena citar

pg_dump não é backup [Telles, 2010]

- Por que fazemos backup?

pg_dump não é backup [Telles, 2010]

- Por que fazemos backup? Recuperação em caso de desastres!

pg_dump não é backup [Telles, 2010]

- Por que fazemos backup? Recuperação em caso de desastres!
- Para os tipos de desastre elencados, pg_dump resolve? Em quanto tempo?
  - Desastre natural.

pg_dump não é backup [Telles, 2010]

- Por que fazemos backup? Recuperação em caso de desastres!
- Para os tipos de desastre elencados, pg_dump resolve? Em quanto tempo?
  - Desastre natural.
  - Falha de hardware.

pg_dump não é backup [Telles, 2010]

- Por que fazemos backup? Recuperação em caso de desastres!
- Para os tipos de desastre elencados, pg_dump resolve? Em quanto tempo?
  - Desastre natural.
  - Falha de hardware.
  - Falha nos discos.

pg_dump não é backup [Telles, 2010]

- Por que fazemos backup? Recuperação em caso de desastres!
- Para os tipos de desastre elencados, pg_dump resolve? Em quanto tempo?
  - Desastre natural.
  - Falha de hardware.
  - Falha nos discos.
  - Falha no SO.

pg_dump não é backup [Telles, 2010]

- Por que fazemos backup? Recuperação em caso de desastres!
- Para os tipos de desastre elencados, pg_dump resolve? Em quanto tempo?
  - Desastre natural.
  - Falha de hardware.
  - Falha nos discos.
  - Falha no SO.
  - Sistema de arquivos corrompido.

pg_dump não é backup [Telles, 2010]

- Por que fazemos backup? Recuperação em caso de desastres!
- Para os tipos de desastre elencados, pg_dump resolve? Em quanto tempo?
  - Desastre natural.
  - Falha de hardware.
  - Falha nos discos.
  - Falha no SO.
  - Sistema de arquivos corrompido.
  - Dados removidos e/ou alterados.

pg_dump não é backup [Telles, 2010]

- Por que fazemos backup? Recuperação em caso de desastres!
- Para os tipos de desastre elencados, pg_dump resolve? Em quanto tempo?
  - Desastre natural.
  - Falha de hardware.
  - Falha nos discos.
  - Falha no SO.
  - Sistema de arquivos corrompido.
  - Dados removidos e/ou alterados.
  - Falha do próprio PostgreSQL.

pg_dump não é backup [Telles, 2010]

- Por que fazemos backup? Recuperação em caso de desastres!
- Para os tipos de desastre elencados, pg_dump resolve? Em quanto tempo?
  - Desastre natural.
  - Falha de hardware.
  - Falha nos discos.
  - Falha no SO.
  - Sistema de arquivos corrompido.
  - Dados removidos e/ou alterados.
  - Falha do próprio PostgreSQL.

Conceitos

- O log de transações (pg_xlog) contém o registro de todas as alterações realizadas no sistema.
- Existe para recuperação em caso de desastres: caso o sistema falhe, as transações podem ser executadas novamente.
- Ideia: utilizar um backup do sistema de arquivos junto com o backup do WAL.

Vantagens

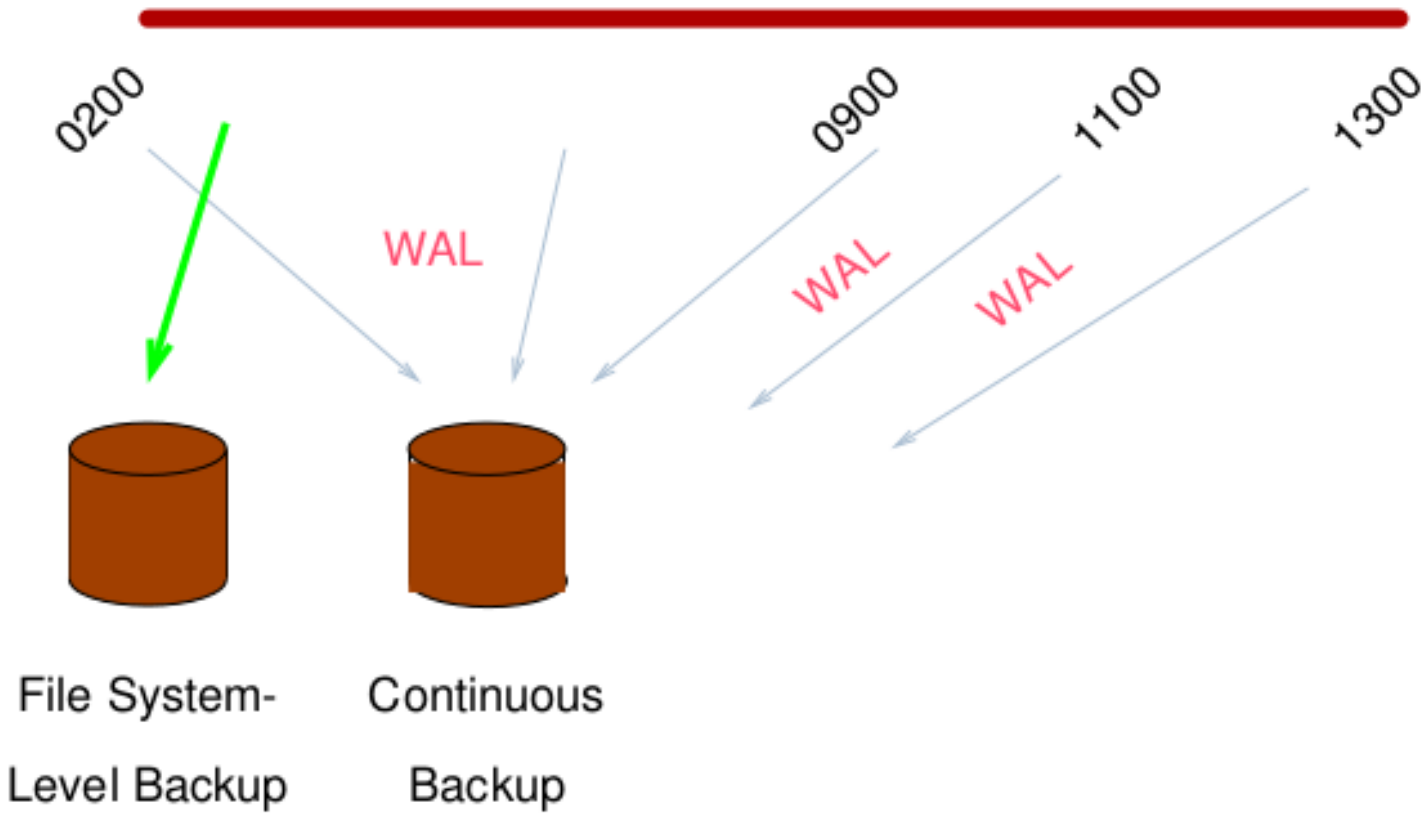
- O backup não precisa estar 100% consistente para começar: se houver qualquer problema as transações do WAL são executadas novamente;
- O tamanho do WAL é indefinidamente grande, então uma fila de transações a executar pode ser criada;
- Nem todas as transações precisam ser executadas novamente. É possível parar em determinado momento e ter um backup consistente naquele ponto;
- Enviar os arquivos do WAL para outra máquina que contém um backup da estrutura do banco de dados gera um sistema de Warm Stand By, que a qualquer momento pode substituir o sistema principal.

Restrições

- Administrar um sistema para o *Warm Stand By* envolve administrar um pequeno cluster. Não é uma tarefa simples[2].
- Só é possível restaurar todo o cluster, e não um pequeno subconjunto de dados.
- Para funcionar, a sequência de arquivos do WAL deve ser contínua. Assim, ajuste o cluster antes de executar o primeiro backup.

[2] Vale sempre ressaltar que Banco de dados não é para amadores

Estrutura



Backup [Momjian, 2010]

Etapas

1. Execute o backup físico
2. Ajuste o tamanho dos "pedaços" do WAL (WAL segments). Normalmente 16MB.
3. Altere os seguintes parâmetros no postgresql.conf:
   - wal_level = 'archive'
   - archive_mode = 'on'
   - archive_command

```
archive_command = 'test ! -f /mnt/server/archivedir/%f && cp %p /mnt/
    server/archivedir/%f'  # Unix
archive_command = 'copy "%p" "C:\\server\\archivedir\\%f"'  # Windows
```

4. Execute o backup.

Observações importantes

- O usuário do banco de dados deve possuir permissão de escrita no diretório (normalmente postgres)
- Teste o comando antes de executar
- O comando pode falhar por várias razões:
  - Falha de rede;
  - Falta de espaço em disco;
  - Falha no sistema de arquivos;
  - Etc...
- O comando deve returnar status zero somente se não houver nenhuma falha
- Sempre considere o fator humano e tente encontrar as falhas o mais rápido possível. Normalmente o PostgreSQL vai executar um PANIC shutdown se houver alguma inconsistência no archive.
- Todas as outras observações possíveis. O seu backup não pode falhar.
- Mantenha o pg_dump. Vai que...

Passos

1. Certifique-se de que o archive está configurado e funcionando;
2. Entre como superusário no banco e execute o comando:

```
SELECT pg_start_backup('label');
```

   Ou o outro:

```
SELECT pg_start_backup('label', true);
```

3. Copie o sistema de arquivos do banco de dados para outra máquina.
   Não é necessário parar o banco
4. Pare o backup:

```
SELECT pg_stop_backup();
```

5. Após a finalização da cópia, o backup está funcional.

Observações

- Teste! Teste! Teste!
- Lembre-se de incluir todos os tablespaces no comando de cópia do archive
- O tempo entre a execução do pg_start_backup e do pg_stop_backup não é tão importante
- Verifique o tamanho dos arquivos de backup e garanta que o diretório tem espaço suficiente

Recuperando o backup

1. Pare o servidor, se estiver no ar
2. Se houver recursos disponíveis, copie todo o diretório PGDATA e os Tablespaces para outro servidor
3. Restaure todos os arquivos do backup físico realizado
4. Apague todos os arquivos do diretorio pg_xlog/
5. Crie um arquivo de recuperação recovery.conf no diretório do Cluster para impedir conexões até que você esteja certo de que o backup está funcionando
6. Inicie o servidor
7. Verifique se o arquivo recovery.done foi criado na pasta PGDATA
8. Verifique o conteúdo do banco de dados

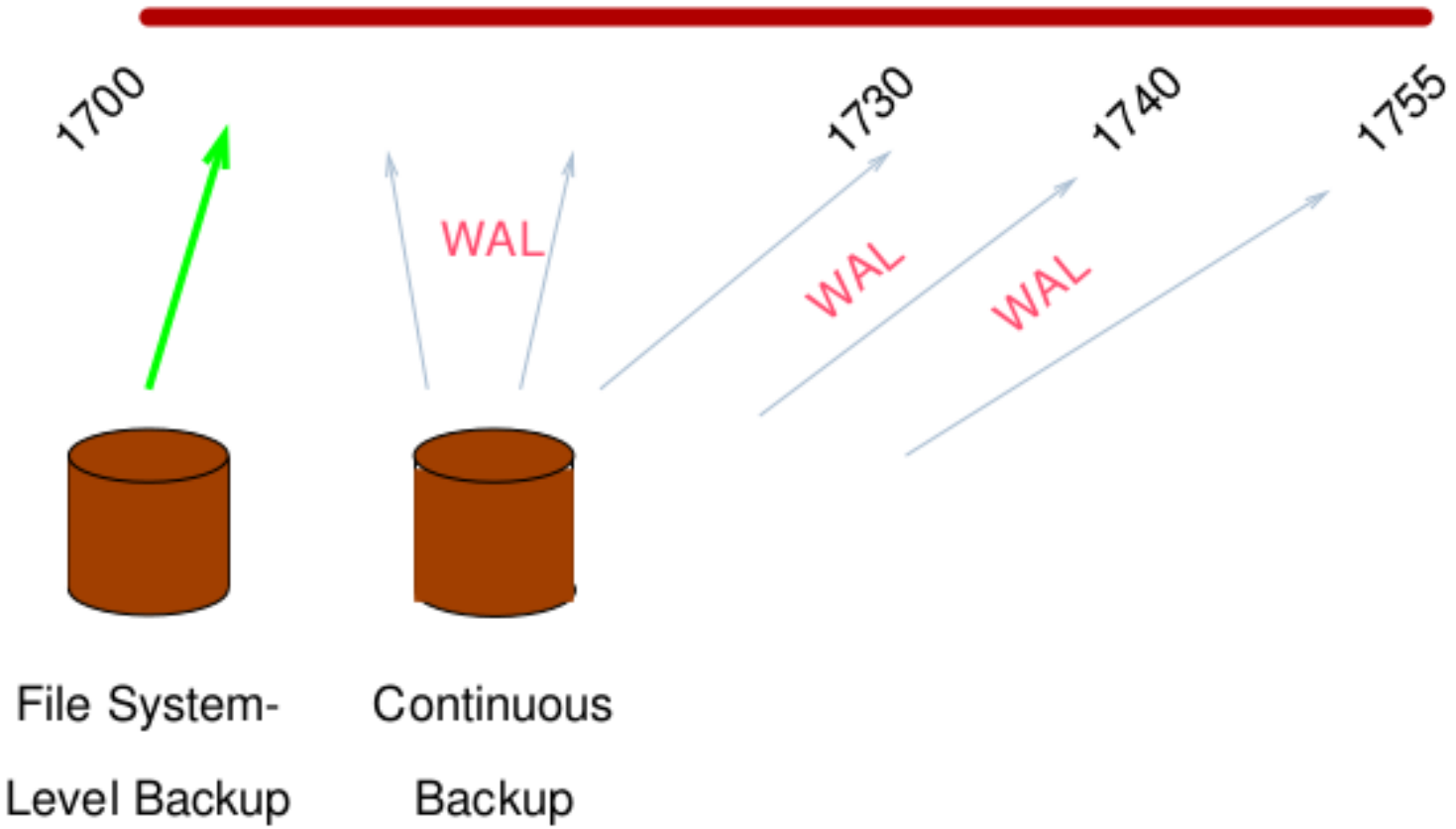
Recuperação

- Chave: criar um arquivo de configuração que diga claramente o que deve ser recuperado e quando deve.

```
restore_command = 'cp /mnt/server/archivedir/%f %p'      # e.g. 'cp /mnt/
    server/archivedir/%f %p'
```

- Perceba que o arquivo vem com todas as opções em branco. Montar cluster não é seguir receita de bolo.

Estrutura para recuperação



Point-In-Time Recovery [Momjian, 2010]

Momjian, B. (2010).
Mastering postgresql administration.
http://momjian.us/main/writings/pgsql/administration.pdf Acessado em 10/11/2009.

PostgreSQL, C. (2013).
Backup.
http://www.postgresql.org/docs/9.1/interactive/backup.html Acessado em 10/06/2013.

Telles, F. (2010).
Dump não é backup.
http://savepoint.blog.br/dump-nao-e-backup/ Acessado em 10/06/2013.

Contato

Eduardo Ferreira dos Santos
Sparkgroup
Lightbase Consultoria em Software Público

eduardo.santos@lightbase.com.br
eduardo.edusantos@gmail.com

www.postgresql.org.br
www.eduardosan.com

+55 61 3347-1949