

Curso de Ciência da Computação

Banco de Dados 2
Aula 06

Prof. Dr. rer. nat. Eros Comunello

Segurança e Recuperação, Data Lake

Segurança e Recuperação em BD Relacionais

O que é Data Lake?

Governança de dados

Elementos de um Data lake

Segurança e recuperação em BD Relacionais

- ❑ É imprescindível que se tenha um plano de manutenção e backup, garantia de que este esteja sempre disponível.
- ❑ No PostgreSQL, os tipos de backup mais utilizados são:
 - ❑ Lógico (que se dá com a geração de arquivo por meio do utilitário “`pg_dump`” e de forma on-line, ou seja, com o serviço do SGBD em execução); e
 - ❑ Físico (que consiste na cópia do diretório de dados “`pgdata`” e de maneira off-line, ou seja, com o serviço do SGBD parado).
- ❑ Há, ainda, a modalidade de backup **incremental** que oferece uma maior precisão quanto ao instante de recuperação dos dados.

Segurança e recuperação em BD Relacionais

- ❑ **pg_dump:** é possível apenas recuperar os dados pertinentes ao instante em que a cópia de segurança foi realizada, com o banco sendo restaurado exatamente como o mesmo se encontrava;
 - ❑ Essa modalidade é extremamente rápida e fácil de se manusear;
 - ❑ Não permite recuperar dados até um determinado ponto na dimensão tempo;
 - ❑ Ex.:
 - ❑ Backup realizado com o “pg_dump” às 3:00, se às 16:30 houvesse uma exclusão indesejável de uma determinada tabela, somente seria possível a recuperação do banco de dados com a situação das 3:00;
 - ❑ Ou seja, todas as inclusões, alterações e/ou exclusões corretas, ocorridas durante o dia, seriam perdidas.

Point-in-Time Recovery (PITR)

- ❑ No PostgreSQL, o backup incremental, utilizando o mecanismo Point-in-Time Recovery;
 - ❑ Oferece como grande diferencial em relação aos demais modelos, uma solução para o problema descrito anteriormente;
 - ❑ Ou seja, a possibilidade de se recuperar dados, retrocedendo até um momento específico na linha do tempo.
- ❑ Toda e qualquer transação de alteração de dados no PostgreSQL, antes de efetivada a gravação nos arquivos de dados do servidor, é registrada em uma área específica denominada de Registro Prévio da Escrita ou, do inglês WAL (Write Ahead Log).

Point-in-Time Recovery (PITR)

- ❑ A função da área de WAL é:
 - ❑ Garantir a consistência e segurança quanto à gravação dos dados em disco, principalmente em situações adversas, como queda de eletricidade;
 - ❑ Propiciar um aumento de desempenho por meio da redução significativa do número de escritas em disco, já que esta representa uma operação de alto custo e extremamente onerosa ao sistema operacional.
- ❑ O mecanismo PITR consiste na geração de uma cópia física completa dos dados e no arquivamento de log das transações realizadas no BD após a geração do backup inicial, para sua posterior restauração.

Point-in-Time Recovery (PITR)

- ❑ O mecanismo PITR apenas garantirá a possibilidade de recuperação de dados, a partir do momento em que as três operações cruciais sejam executadas, são elas:
 - ❑ habilitar o arquivamento de log das transações (WAL);
 - ❑ realizar a cópia de segurança com identificador de base inicial; e
 - ❑ armazenar os arquivos de log das transações realizadas (WAL) após o backup inicial.
- ❑ É imprescindível a ocorrência de tais ações para o funcionamento desse recurso.

Segurança e recuperação em BD Relacionais

Configuração do arquivo “postgresql.conf”

- ❑ Efetuar login com o usuário “postgres” (*sudo su – postgres*)
- ❑ Criar o diretório onde serão armazenados os arquivos de log(WAL)
 - ❑ Preferencialmente .../postgres/12/backup/wal
- ❑ Abrir arquivo de configurações “postgresql.conf”
 - ❑ ex.: /Library/PostgreSQL/12/share/postgresql/
- ❑ “archive_command”, deve-se inserir neste, o comando do sistema operacional (aqui, será utilizado o comando “cp”) o qual realizará as cópias dos arquivos de log;
- ❑ Também deve-se excluir o caractere de identificação de comentário “#” dessa linha.

Segurança e recuperação em BD Relacionais

Configuração do arquivo “postgresql.conf”

```
# - Archiving -  
  
#archive_mode = off          # enables archiving; off, on, or always  
#                                # (change requires restart)  
#archive_command = ''        # command to use to archive a logfile segment  
#                                # placeholders: %p = path of file to archive  
#                                #                 %f = file name only  
#                                # e.g. 'test ! -f /mnt/server/archivedir/%f && cp %p /mnt/server/archivedir/%f'  
#archive_timeout = 0          # force a logfile segment switch after this  
#                                # number of seconds; 0 disables
```

- ❑ Caracteres “%p” presentes na linha de comando serão substituídos no interpretador pelo caminho do arquivo cuja cópia de segurança será feita;
- ❑ Caracteres “%f” serão substituídos no interpretador pelo nome do arquivo a ser copiado.

Point-in-Time Recovery (PITR)

- ❑ Sem essa configuração, o servidor PostgreSQL irá descartar os arquivos de log do WAL sempre que o segmento for completado e as transações forem efetivadas em disco;
- ❑ Sem isso, torna impossível a recuperação de dados ou o retorno à situação de um determinado momento do banco de dados.
- ❑ Por fim, lembre-se de reiniciar seu servidor postgresql.

Backup incremental

Criação de Backup-base

- ❑ Os passos do item anterior asseguram o funcionamento da tarefa de arquivamento de log WAL e armazenamento dos arquivos no local pré-definido;
- ❑ Entretanto, para permitir a possibilidade de recuperação de dados com base nos arquivos de log armazenados é necessário que seja gerada uma cópia de segurança inicial do banco de dados,
 - ❑ Será utilizada como base para uma futura recuperação de dados utilizando os arquivos de log..

Backup incremental

Criação de Backup-base

- ❑ Acesse o terminal iterativo “psql” do PostgreSQL, com o superusuário “postgres”
 - ❑ psql -U postgres
- ❑ #SELECT pg_start_backup('rotulo_backup_base');
 - ❑ Irá gerar um arquivo denominado de “backup_label” que contém informações a respeito do backup a ser realizado;
 - ❑ Permitirá que durante uma possível recuperação de dados, identifique o ponto exato de onde teve início a cópia de segurança
 - ❑ Poderá garantir a total compatibilidade e continuidade entre o backup-base e os arquivos de log das transações realizadas posteriormente.

Backup incremental

Criação de Backup-base

- ❑ Pode iniciar a cópia de segurança do diretório de dados do PostgreSQL (“pgdata”) e compacta-lo utilizando, por exemplo o TAR e o GZIP (“pgdata.tar.gz”);
- ❑ Função para identificar fim do backup:
 - ❑ #SELECT pg_stop_backup();
- ❑ Reinicie o PostgreSQL.
- ❑ De posse do backup-base já criado e dos arquivos de log WAL que serão criados em decorrência das transações que venham a alterar a base de dados, tem se a ideia de “Backup incremental” no PostgreSQL.

Backup incremental

Criação de Backup-base

- ❑ Esse recurso associado ao Mecanismo PITR permite a recuperação total ou a recuperação cronológica de dados.
- ❑ Em bancos de dados com grande volume de transações de manipulação de dados (insert, update e delete), os arquivos de log WAL podem crescer consideravelmente.
 - ❑ Atente-se ao espaço de armazenamento disponível e, periodicamente, caso necessário, exclua os arquivos de log WAL e refaça os procedimentos do item “Criação de Backup-base”.

Backup incremental

Criação de Backup-base

- ❑ O backup incremental é um recurso de segurança mais voltado para periodicidades menores(semanal ou mensal);
- ❑ Exigência por maior espaço de armazenamento, bem como o fato de não ser muito usual a necessidade de se retroceder a situação do BD maior do que um mês.
- ❑ Para fins de cópias de segurança de maior periodicidade (acima de um mês) é recomendado o uso paralelo de métodos mais eficientes como o backup físico tradicional (cópia do diretório de dados) ou, principalmente, o backup lógico (usando o utilitário “`pg_dump`”).

Backup incremental

Criação de Backup-base

- ❑ Portanto, o backup incremental deve ser utilizado em paralelo a outro método de cópia de segurança e não em substituição a este, já que seu propósito principal aqui é possibilitar a recuperação cronológica de dados, preferencialmente em um espaço de tempo recente.

Recuperação cronológica de dados

- ❑ Essa operação tem custo operacional e daí a necessidade de terem horários programados para sua execução, geralmente uma vez ao dia;
- ❑ Métodos tradicionais de backup (físico ou lógico): são extremamente eficazes, mas estão sempre com defasagem de atualização;
- ❑ Mecanismo PITR: esse recurso permite que se recupere todos os dados (total) em um determinado intervalo ou somente até determinado instante (parcial);
 - ❑ Isso é possível porque todas as transações SQL de DDL4, DML5 e DCL6 são, antes de efetivadas em disco, gravadas nos arquivos de log WAL, podendo essas serem recuperadas a partir do incremento em um backup-base gerado previamente.

Recuperação cronológica de dados

- ❑ Assim, com o backup-base criado e o Mecanismo PITR habilitado, o usuário passa a ter em mãos uma poderosa ferramenta de recuperação de dados total ou parcial;
- ❑ Desta forma, o usuário poderá restaurar ou recuperar sua base de dados a qualquer instante sempre que houver uma situação que implique na perda dos dados;
- ❑ Quando ativado, o mecanismo PITR fará a restauração completa do banco de dados, ou seja, efetuará o incremento de todos os arquivos de log WAL armazenados junto ao backup-base, restaurando assim, a base de dados ao instante que precedeu essa operação.

Recuperação cronológica de dados

- ❑ Essa opção é recomendada para casos em que o banco de dados foi excluído acidentalmente ou tenha corrompido seus arquivos;
- ❑ Caso o problema seja de inclusão, alteração ou exclusão não desejada de objetos do banco de dados (por exemplo, tabelas, colunas, registros, chaves, restrições, entre outros), será necessário **informar o “id” da transação ou a data e horário** do instante em que aconteceu a operação não desejada, para que o PITR recupere os dados até aquele determinado instante.

Restauração do diretório de dados

- ❑ Para iniciar os procedimentos de restauração propriamente dito, é preciso parar o serviço do PostgreSQL;
- ❑ Mantenha o diretório de dados atual guardado, renomeando-o ou movendo para outro diretório;
- ❑ O passo a seguir é copiar o backup-base “pgdata.tar.gz” para o local onde é alocado o diretório de dados do PostgreSQL e descompacte o arquivo;
- ❑ Acesse o novo diretório de dados e exclua o arquivo identificador do processo “postmaster.pid”;

Restauração do diretório de dados

- ❑ Dentro do diretório de dados, acesse o subdiretório “pg_xlog” e remova todo o seu conteúdo (arquivos de segmentos WAL antigos);
- ❑ Subdiretório “archive_status” que deverá ser mantido;
- ❑ Para garantir que as últimas transações realizadas no banco de dados, antes de sua parada ou do problema ocorrido, também estejam disponíveis, faça a cópia dos arquivos de segmento WAL do diretório de dados antigo “pgdata.old” para o diretório atual “pgdata”.

Restauração do diretório de dados

Configuração do arquivo “recovery.conf”

- ❑ A última etapa para o processo de Point-in-Time Recovery é a configuração do arquivo “recovery.conf”, que possui as definições necessárias para a operação de recuperação dos dados;
- ❑ Renomeie o arquivo “recovery.conf.sample” para “recovery.conf” e mova para o diretório “pgdata”;
- ❑ No documento “recovery.conf” procure pelo item “restore_command”, que se trata do parâmetro obrigatório para funcionamento do mecanismo PITR.

Restauração do diretório de dados

Configuração do arquivo “recovery.conf”

- ❑ Exclua o caractere “#” do “restore_command”;
- ❑ Esse procedimento é o suficiente para, assim que o serviço do PostgreSQL for novamente iniciado, garantir a realização da operação de restauração do cluster de banco de dados;
- ❑ O mecanismo PITR fará a restauração completa do cluster de BD;
- ❑ Reinicie o PostgreSQL para executar o recovery!

Restauração do diretório de dados

Configuração do arquivo “recovery.conf”

restore_command = ‘cp /home/postgres/12/backup/wal/%f %p’

```
# - Archive Recovery -
# These are only used in recovery mode.

#restore_command = ''          # command to use to restore an archived logfile segment
# placeholders: %p = path of file to restore
#                 %f = file name only
# e.g. 'cp /mnt/server/archivedir/%f %p'
# (change requires restart)
```

Restauração do diretório de dados

Configuração do arquivo “recovery.conf”

- ❑ Caso deseje realizar uma recuperação cronológica de dados até determinado instante, é necessário localizar a sessão “OPTIONAL PARAMETERS” e alterar, no mínimo, um dos parâmetros;

```
# - Recovery Target -  
  
# Set these only when performing a targeted recovery.  
  
#recovery_target = ''          # 'immediate' to end recovery as soon as a  
#                             # consistent state is reached  
#                             # (change requires restart)  
#recovery_target_name = ''     # the named restore point to which recovery will proceed  
#                             # (change requires restart)  
#recovery_target_time = ''     # the time stamp up to which recovery will proceed  
#                             # (change requires restart)  
#recovery_target_xid = ''      # the transaction ID up to which recovery will proceed  
#                             # (change requires restart)  
#recovery_target_lsn = ''      # the WAL LSN up to which recovery will proceed  
#                             # (change requires restart)  
#recovery_target_inclusive = on # Specifies whether to stop:  
#                             # just after the specified recovery target (on)  
#                             # just before the recovery target (off)  
#                             # (change requires restart)  
#recovery_target_timeline = 'latest' # 'current', 'latest', or timeline ID  
#                             # (change requires restart)  
#recovery_target_action = 'pause'   # 'pause', 'promote', 'shutdown'  
#                             # (change requires restart)
```

DATA LAKE

O que é Data Lake?

- ❑ Um **Data Lake** é uma abordagem de arquitetura que permite armazenar grandes quantidades de dados em um local central para que esteja disponível para ser **categorizado, processado, analisado e consumido** por diversos grupos dentro de sua organização;
- ❑ É uma plataforma de armazenamento baseada em nuvem, segura e durável que permite você inserir e armazenar tanto dados estruturados quanto não estruturados;
- ❑ Os dados podem ser armazenados em seu formato original!
 - ❑ Não precisa convertê-los em um esquema predefinido e também não precisa necessariamente saber de antemão quais perguntas (consultas) vai fazer sobre seus dados;

O que é Data Lake?

- ❑ Pode usar um portfólio completo de ferramentas de exploração de dados, relatórios, análises, *machine learning* (aprendizado de máquina) e visualização nos dados;
- ❑ Exemplos de data lakes podem ser encontrados na AWS, Azure, Google Cloud, dentre outros.

Características de um Data Lake

- ❑ Centralizar todos os dados da organização num único local;
- ❑ Aceita dados estruturados, semi-estruturados e não-estruturados;
- ❑ Alta performance em escrita (*ingestão*) e em acesso (*consumption*);
- ❑ Baixo custo de armazenamento;
- ❑ Suporta regras de segurança e proteção de dados;
- ❑ Desacopla o armazenamento do processamento (permitindo alta performance e alta escala).

Quando usar?

- ❑ Geralmente, usa-se de ***Data Lakes*** para um volume de *petabytes* ou *exabytes* de dados;
- ❑ Útil quando o volume é grande o suficiente para não poder ser processado num único servidor num **tempo aceitável**, e que não poderia ser executado num **SBDG convencional**, por isso, precisa de **processamento paralelo** e de **compactação de dados**;
- ❑ Poucas fontes de dados, pouco volume de dados, formatos padronizados de dados, e tudo pode ser facilmente analisado num único banco de dados relacional, criar um *Data Lake* seria uma abordagem exagerada;

Quando usar?

- ❑ O custo de armazenamento de **Data Lakes** é geralmente bem menor do que de um *Data Warehouse*;
- ❑ Chega a ser de **20x a 50x mais barato** armazenar, gerenciar e analisar dados em comparação com as tecnologias tradicionais de *data warehouse*;
- ❑ Quando usar Data Lake?
 - ❑ Preciso tratar *streaming* de dados (IoT, *Click Streams*, etc.);
 - ❑ Meus dados tem diferentes fontes de origem;
 - ❑ Os formatos desses dados são variados;
 - ❑ O volume de dados é muito grande (petabytes, exabytes).

Exemplo

Como funciona um Data Lake na AWS?

- ❑ A AWS oferece uma série de serviços e tecnologias que permite que faça *upload* de **volumes muito grandes de dados** nos seus mais diferentes formatos como CSV, JSON e Parquet em Buckets do S3;
- ❑ Depois catalogue esses dados com o **AWS Glue** para finalmente usando o **Athena** fazer consultas nesses dados que estão armazenados em seu formato original;
- ❑ Resposta é similar a consultas em um banco de dados relacional através com SQL.

Como funciona um Data Lake na AWS?

- ❑ Criar soluções de **Data Lake** de maneira econômica usando o Amazon S3 e AWS Glue para fazer o seguinte:
 - ❑ Ingerir e armazenar dados de uma ampla variedade de fontes em uma plataforma centralizada;
 - ❑ Armazenar, pagando apenas pelo que você usar;
 - ❑ Criar um catálogo de dados para localizar e usar os dados armazenados no *Data Lake*;
 - ❑ Proteger os dados armazenados no *Data Lake*;
 - ❑ Usar ferramentas e políticas para monitorar, analisar e otimizar a infraestrutura e os dados;
 - ❑ ...

Como funciona um Data Lake na AWS?

- ❑ Transformar dados brutos em formatos otimizados (ex: parquet);
- ❑ Consultar dados;
- ❑ Usar ferramentas de análise de dados, ciência de dados, *machine learning* e visualização de dados (dataviz);
- ❑ Integrar rapidamente as ferramentas de processamento de dados;
- ❑ Compartilhar dados e resultados processados facilmente.

Ingestão de Dados no Data Lake

- ❑ Um dos principais recursos de uma arquitetura de ***data lake*** é a capacidade de ingerir vários tipos de dados de maneira rápida e fácil, isso é, gravar dados de diferentes fontes no seu *data lake*;
- ❑ Algumas fontes de dados comuns são, por exemplo:
 - ❑ dados *streams* em tempo real (dispositivos, IoT, *click streams*);
 - ❑ dados de plataformas *on premisses*;
 - ❑ dados de sistemas legados;
 - ❑ dados de *mainframes*;
 - ❑ dados de *data warehouses*.

Exemplo: Ingestão de Dados no Data Lake

Ferramentas úteis para ingestão de dados que a AWS :

- ❑ **Amazon Kinesis Firehose:** Um serviço gerenciado para fornecer dados de streaming em tempo real diretamente para o Amazon S3;
 - ❑ O Kinesis Firehose é dimensionado automaticamente para atender o volume e taxa de transferência de dados;
 - ❑ Pode ser configurado para transformar dados, compactar (GZIP, ZIP e SNAPPY.), criptografar e chamar funções do Lambda.

Exemplo: Ingestão de Dados no Data Lake

Ferramentas úteis para ingestão de dados que a AWS :

- ❑ **AWS Snowball:** São dispositivos físicos usados para migrar grandes quantidades de dados *on-premises* para a nuvem da AWS;
 - ❑ Solicitado pelo Console da AWS, um dispositivo *Snowball* é enviado para você copiar seus dados com criptografia AES de 256 bits;
 - ❑ Segurança: As chaves de criptografia nunca são enviadas com o dispositivo *Snowball*;
 - ❑ Você então manda o dispositivo de volta para a AWS. Após o recebimento na AWS, seus dados são transferidos do dispositivo *Snowball* para um *bucket* do S3 e armazenados como objetos S3 no formato original.

Exemplo: Ingestão de Dados no Data Lake

Ferramentas úteis para ingestão de dados que a AWS :

- ❑ **AWS Storage Gateway:** esse serviço pode ser usado para integrar plataformas de processamento de dados *on-premises* com um **Data Lake** no S3;
 - ❑ É possível criar um compartilhamento de arquivos de rede através de uma conexão NFS;
 - ❑ Os arquivos salvos nessa partição (*mount*) então são convertidos em objetos e armazenados no Amazon S3 em seu formato original, sem nenhuma modificação.

Catalogar Dados no Data Lake

- ❑ É necessário **organizar** e **catalogar** esses dados para que você e sua equipe saibam exatamente o que está armazenado em seu Data Lake e então consigam encontrar e acessar facilmente as informações quando precisarem delas;
- ❑ Um **catálogo de dados** fornece uma interface de consulta de todos os recursos armazenados nos seu Data Lake (ex.: *buckets do S3*);
- ❑ O catálogo de dados foi projetado para fornecer uma única **fonte de verdade** sobre o conteúdo do Data Lake.

Catalogar Dados no Data Lake

- ❑ Há 2 tipos principais de catálogo de dados:
 - ❑ Abrangente (*Comprehensive Data Catalog*): contém informações sobre todos os ativos que foram ingeridos no Data Lake;
 - ❑ HCatalog (Hive Metastore Catalog): contém informações sobre ativos de dados que foram transformados e definições de tabela que são utilizáveis por ferramentas;
 - ❑ Por exemplo, o Athena, o Redshift, Spectrum e EMR.
- ❑ Esses 2 tipos de catálogos não são mutuamente exclusivos;
- ❑ O abrangente pode ser usado para pesquisar todos os ativos no Data Lake, e o HCatalog pode ser usado para descobrir e consultar esses dados.

Catalogar Dados no Data Lake

Ex. na AWS:

- ❑ O Catálogo **Abrangente** pode ser criado usando serviços como AWS Lambda, DynamoDB, ou Elasticsearch Service, por exemplo;
- ❑ Já para construir o seu **HCatalog**, recomendo utilizar o **AWS Glue**, que já é compatível com Hive e uma série de outras ferramentas que podem ser usadas posteriormente para explorar esses dados.

Consultando dados no Data Lake

- ❑ Um dos recursos mais importantes de um **Data Lake** é a capacidade de fazer ***In-Place Querying***;
- ❑ Pode consultar de dados sem precisar provisionar e gerenciar clusters de servidores;
- ❑ Em outras palavras você pode executar consultas analíticas sofisticadas diretamente no dados que estão no S3;
- ❑ **Ex. 1: Athena:**
 - ❑ Permite consultar dados do S3 como Athena pagando apenas pelas consultas executadas. O Athena é um serviço de consulta que facilita a análise de dados diretamente no Amazon S3 usando SQL padrão.

Consultando dados no Data Lake

❑ Ex. 2: Amazon Redshift Spectrum:

- ❑ O Redshift é um serviço de data warehouse gerenciado que pode ser usado em combinação com o S3 para consultar até exabytes de dados, tudo isso realizado com sintaxe SQL e permitindo que você cruze os dados do seu Data Lake no S3 com os dados no Redshift.

❑ Ex. 3: QuickSight:

- ❑ É um serviço para Visualização dos Dados (DataViz);
- ❑ Com ele você pode criar gráficos e *Dashboards* integrados ao athena que consultará os dados do seu Data Lake;
- ❑ É possível utilizar outras ferramentas similares como Tableau ou QlikView também.

Links complementares

- ❑ <https://blog.tecnospeed.com.br/backup-e-restore-postgresql/>
- ❑ <https://atendimento.tecnospeed.com.br/hc/pt-br/articles/360013719953-PostgreSQL-Recuperando-a-base-de-dados-sem-backup->
- ❑ <https://blog.andrefaria.com/data-lake-o-que-voce-precisa-saber-para-comecar>
- ❑ [The Enterprise Big Data Lake Gorelik, Alex. O'Reilly \(Publisher\)](#)