Desafio 1 - Gerenciamento de Banco de Dados Estudo de Caso e Implementação Prática de Gerenciamento de Banco de Dados para uma Loja Online 01

Data: 17 de agosto de 2025

Equipe H: Rafael Claumann Bernardes, Guilherme Claumann Silva

Repositório: GitHub

Análise e Planejamento: Analise a situação da E-Shop e identifique os principais problemas e desafios relacionados ao gerenciamento do banco de dados e como resolvê-los.

> É preciso definir políticas no backend para armazenar as senhas encriptadas com SHA1.
> No momento, as senhas são salvas como Plain Text.
>
> O endereço estava armazenado em um campo da tabela Clientes, o que dificultava a realização de análises e busca por informações pertinentes.
>
> O banco de dados da E-shop ainda não tem roles e perfis necessários, nós vamos resolver isso criando perfis e adicionando permissões específicas a cada perfil.

Backup e Restauração: Descreva um procedimento de backup e restauração para o banco de dados da E-Shop, detalhando as etapas e o cronograma para a realização destas tarefas.

Vamos usar um script shell que combina backups completos e incrementais de forma automatizada usando `xtrabackup`.

Quando executa um backup completo, o script cria uma cópia de todos os dados do banco, que serve como base para os backups incrementais subsequentes. Esse backup completo é registrado para que o script saiba sempre qual é a referência atual.

Nos backups incrementais, o script copia apenas as alterações ocorridas desde o último backup completo, garantindo que cada incremental possa ser aplicado corretamente para restaurar o estado do banco até um ponto específico no tempo.

Caso não exista um backup completo, o script cria automaticamente um, assegurando que sempre haja uma base confiável. Cada backup é armazenado em uma pasta separada com timestamp, preservando o histórico e evitando a sobrescrita de dados.

```
#!/bin/bash

#exec >> /proc/1/fd/1 2>&1
CORRELATION_ID=$(date +%s%N)

MYSQL_HOST=db
MYSQL_USER=root
MYSQL_PASSWORD=password
MYSQL_DB=loja

BACKUP_DIR=/backups
MYSQL_DATA=/var/lib/mysql
TYPE=$1
DATE=$(date +"%Y-%m-%d_%H-%M-%S.%3N")
```

```
LAST_FULL="$BACKUP_DIR/last_full.txt"

echo "$CORRELATION_ID - [$(date +"%Y-%m-%d_%H-%M-%S.%3N")][S
TART $TYPE BACKUP]"

if [ "$TYPE" = "full" ]; then
    TARGET="$BACKUP_DIR/full_$DATE"
    echo "$CORRELATION_ID - [POC] Criando FULL backup em $TARGET"

   xtrabackup --backup \
        --datadir=$MYSQL_DATA \
        --target-dir=$TARGET \
        --host=$MYSQL_HOST \
        --user=$MYSQL_USER \
        --password=$MYSQL_PASSWORD \
        >/dev/null 2>&1

    echo "$TARGET" > $LAST_FULL

elif [ "$TYPE" = "incremental" ]; then
    if [ ! -f "$LAST_FULL" ]; then
        echo "$CORRELATION_ID - [POC] Nenhum full encontrado, criando fu
ll primeiro."
        $0 full
        exit 0
    fi

    BASE=$(cat $LAST_FULL)
    TARGET="$BACKUP_DIR/inc_$DATE"
    echo "$CORRELATION_ID - [POC] Criando INCREMENTAL baseado em
$BASE → $TARGET"
    xtrabackup  --backup \
        --datadir=$MYSQL_DATA \
        --incremental-basedir=$BASE \
        --target-dir=$TARGET \
```

```
        --host=$MYSQL_HOST \
        --user=$MYSQL_USER \
        --password=$MYSQL_PASSWORD \
        >/dev/null 2>&1

else
    echo "Uso: $0 [full|incremental]"
    exit 1
fi

echo "$CORRELATION_ID - [$(date +"%Y-%m-%d_%H-%M-%S.%3N")][F
INISH $TYPE BACKUP]"
echo
```

O script será executado através de cron no linux. Nesta prova de conceito nós realizamos backups completos a cada 10 minutos e backups incrementais a cada 1 minuto.

```
  backup:
    container_name: mysql_backup
    image: ubuntu:22.04
    volumes:
      - mysql_data:/var/lib/mysql
      - ./backup-scripts:/scripts
      - ./backup-scripts/log.txt:/var/log/backup.log
    entrypoint:
      - /bin/bash
      - -c
      - |
        # Atualiza e instala pacotes básicos
        apt update
        apt install -y curl gnupg2 lsb-release cron

        # Baixa e instala o repositório Percona
```

```
    curl -O https://repo.percona.com/apt/percona-release_latest.generic_
all.deb
    apt install -y ./percona-release_latest.generic_all.deb

    # Configura e instala o XtraBackup
    percona-release setup pxb-80
    apt install -y percona-xtrabackup-80
    xtrabackup --version

    mkdir backups


        # backups incrementais a cada 1 minuto
        # obs: tempo reduzido apenas para testes
    (crontab -l 2>/dev/null; echo "* * * * * /scripts/backup.sh incremental
>> /scripts/log.txt 2>&1") | crontab -

        # backups completos a cada 5 minutos
        # obs: tempo reduzido apenas para testes
    (crontab -l 2>/dev/null; echo "*/5 * * * * /scripts/backup.sh full >> /sc
ripts/log.txt 2>&1") | crontab -

    cron -f
```

Recuperação de Dados: Crie uma situação hipotética em que alguns dados foram perdidos ou corrompidos.

O servidor sofreu uma queda de energia durante uma atualização de pedidos e, após reiniciar, foi detectado que:

- O banco ainda sobe, mas a tabela `Pedidos` está **corrompida**.
- O diretório de backups (`/backups`) contém:

    - Um **backup full** feito na madrugada (2h).
    - Três **backups incrementais** feitos ao longo do dia.
- O último incremental foi às 18h, e a falha ocorreu às 18h30.

Dentro da pasta de backup, verificar quais diretórios existem:

```
ls -lh /backups
# full_2025-08-17_02-00-00
# incr_2025-08-17_10-00-00
# incr_2025-08-17_14-00-00
# incr_2025-08-17_18-00-00
```

Aplicar --prepare no backup full para deixá-lo consistente:

```
xtrabackup --prepare --target-dir=/backups/full_2025-08-17_02-00-00
```

Aplicar cada backup incremental na ordem em que foram feitos:

```
xtrabackup --prepare --apply-log-only \
  --target-dir=/backups/full_2025-08-17_02-00-00 \
  --incremental-dir=/backups/incr_2025-08-17_10-00-00

xtrabackup --prepare --apply-log-only \
  --target-dir=/backups/full_2025-08-17_02-00-00 \
  --incremental-dir=/backups/incr_2025-08-17_14-00-00

xtrabackup --prepare --apply-log-only \
  --target-dir=/backups/full_2025-08-17_02-00-00 \
  --incremental-dir=/backups/incr_2025-08-17_18-00-00

xtrabackup --prepare \
  --target-dir=/backups/full_2025-08-17_02-00-00
```

Parar o MySQL, mover os dados corrompidos para outra pasta e restaurar o backup:

```
systemctl stop mysql

mv /var/lib/mysql /var/lib/mysql_bkp_corrompido

xtrabackup --copy-back --target-dir=/backups/full_2025-08-17_02-00-00

chown -R mysql:mysql /var/lib/mysql
```

Subir o MySQL e validar

```
systemctl start mysql
mysqlcheck -u root -p --all-databases
```

Controle de Permissões: Defina um esquema de controle de permissões para os diferentes usuários do sistema, como administradores, funcionários e clientes. Justifique suas escolhas.

A ideia é que o acesso ao banco de dados seja segregado com diferentes perfis de acesso para garantir que o banco mantenha-se consistente.

Usuários Administrativos

1. `dba_user`
    - **Grants:** `ALL PRIVILEGES ON *.* WITH GRANT OPTION`
    - **Descrição:** Usuário responsável pela administração completa do banco.
    - **Justificativa:** Precisa de acesso irrestrito para criar, alterar e remover tabelas, índices, usuários, permissões e manter a integridade geral do sistema. O `GRANT OPTION` permite delegar permissões a outros usuários.
2. `infra_user`
    - **Grants:** `PROCESS, RELOAD, REPLICATION CLIENT, SHOW DATABASES`

- **Descrição:** Usuário de infraestrutura/DevOps que monitora a instância e executa tarefas de manutenção.
- **Justificativa:** Deve visualizar processos, verificar status de replicação e reinicializar cache sem ter acesso direto aos dados (segurança).

3. `auditor_user`
   - **Grants:** `SELECT ON loja.*`
   - **Descrição:** Usuário de auditoria.
   - **Justificativa:** Permite leitura completa dos dados para fins de compliance e auditoria, mas sem poder alterar informações.

Usuários Operacionais

1. `rh_user`
   - **Grants:** `SELECT, INSERT, UPDATE ON loja.Funcionarios`
   - **Descrição:** Usuário do sistema de RH.
   - **Justificativa:** Precisa consultar dados de funcionários e atualizar informações de cargos/salários. Não possui acesso a outras tabelas para evitar exposição indevida de dados de clientes e vendas.
2. `ecom_user` (E-commerce – vendas online)
   - **Grants:**
     - `SELECT, INSERT, UPDATE ON loja.Pedidos`
     - `SELECT, INSERT, UPDATE ON loja.ItensPedidos`
     - `SELECT, INSERT, UPDATE ON loja.Produtos`
     - `SELECT, INSERT, UPDATE ON loja.Clientes`
     - `SELECT, INSERT, UPDATE ON loja.Enderecos`
   - **Descrição:** Usuário utilizado pela plataforma de vendas online.
   - **Justificativa:**
     - Precisa criar e atualizar pedidos e itens.

    - Pode cadastrar e atualizar dados de clientes, endereços e produtos.

3. `bi_user` (Business Intelligence / Relatórios)
    - **Grants:** `SELECT ON loja.*`
    - **Descrição:** Usuário responsável por relatórios e dashboards.
    - **Justificativa:** Necessita acesso de leitura a todas as tabelas para análise, mas não pode modificar dados.

```
-- ============================
-- Usuários Administrativos
-- ============================

-- DBA: controle total
CREATE USER 'dba_user'@'%' IDENTIFIED BY 'senhaSegura#1';
GRANT ALL PRIVILEGES ON *.* TO 'dba_user'@'%' WITH GRANT OPTION;
-- Justificativa: acesso irrestrito para administração completa do banco.

-- Infra: monitoramento e manutenção
CREATE USER 'infra_user'@'%' IDENTIFIED BY 'senhaSegura#2';
GRANT PROCESS, RELOAD, REPLICATION CLIENT, SHOW DATABASES ON
*.* TO 'infra_user'@'%';
-- Justificativa: monitorar performance, replicação e reinicializar cache, se
m acesso direto aos dados.

-- Auditor: leitura global
CREATE USER 'auditor_user'@'%' IDENTIFIED BY 'senhaSegura#3';
GRANT SELECT ON loja.* TO 'auditor_user'@'%';
-- Justificativa: leitura completa para compliance/auditoria, sem alterar na
da.

-- ============================
-- Usuários Operacionais
```

```
-- ========================

-- RH: gestão de funcionários
CREATE USER 'rh_user'@'%' IDENTIFIED BY 'senhaSegura#4';
GRANT SELECT, INSERT, UPDATE ON loja.Funcionarios TO 'rh_user'@'%';
-- Justificativa: gerencia dados de funcionários, sem acesso a clientes/pe
didos.

-- E-commerce: vendas online
CREATE USER 'ecom_user'@'%' IDENTIFIED BY 'senhaSegura#6';
GRANT SELECT, INSERT, UPDATE ON loja.Pedidos TO 'ecom_user'@'%';
GRANT SELECT, INSERT, UPDATE ON loja.ItensPedidos TO 'ecom_use
r'@'%';
GRANT SELECT, INSERT, UPDATE ON loja.Produtos TO 'ecom_user'@'%';
GRANT SELECT, INSERT, UPDATE ON loja.Clientes TO 'ecom_user'@'%';
GRANT SELECT, INSERT, UPDATE ON loja.Enderecos TO 'ecom_use
r'@'%';
-- Justificativa: pode registrar pedidos online, gerenciar dados de clientes
e endereços, mas não alterar produtos.

-- BI: relatórios e dashboards
CREATE USER 'bi_user'@'%' IDENTIFIED BY 'senhaSegura#7';
GRANT SELECT ON loja.* TO 'bi_user'@'%';
-- Justificativa: acesso apenas leitura para relatórios, sem risco de modific
ação.

-- ========================
-- Aplicar as permissões
-- ========================
FLUSH PRIVILEGES;
```

Integridade e Consistência de Dados: Verifique a integridade e a consistência dos dados no banco de dados atual e proponha melhorias. Implemente as

mudanças sugeridas, utilizando constraints e outros recursos do SGBD.

Encontramos duas oportunidades claras de melhoria.

1. A primeira oportunidade estava na tabela Clientes que armazenava o endereço do cliente como uma String sem validações ou restrições.

   Nós criamos a tabela Enderecos para normalizar o campo da tabela Clientes e facilitar buscar futuras já que agora os dados estão padronizados por colunas e tipo.

```
CREATE TABLE Enderecos (
    id INT PRIMARY KEY,
    id_cliente INT NOT NULL,
    logradouro VARCHAR(155) NOT NULL,
    numero INT NOT NULL,
    complemento VARCHAR(50),
    bairro VARCHAR(50) NOT NULL,
    cidade VARCHAR(30) NOT NULL,
    estado VARCHAR(2) NOT NULL,
    cep VARCHAR(9) NOT NULL,
    FOREIGN KEY (id_cliente) REFERENCES Clientes (id)
);
```

2. A segunda oportunidade envolve o campo status da tabela Pedidos.
   Este campo também era armazenado como uma string simples e permitia diferentes valores sem qualquer tipo de validação.

   Criamos a tabela Status para padronizar os possíveis status dos pedidos.

```
CREATE TABLE Status (
  id INT PRIMARY KEY,
  nome ENUM('CARRINHO', 'PAGAMENTO', 'ANDAMENTO', 'ENTREGU
E') NOT NULL UNIQUE
);
```

Schema atualizado