

Curso de DBA - Trabalho Prático: Banco de Dados de Produções Artísticas

Visão Geral

Você está prestes a se tornar o administrador de banco de dados da **CineTech Studios**, uma empresa fictícia de entretenimento que gerencia um vasto banco de dados de produções artísticas incluindo filmes, séries de TV, videogames e muito mais.

Sua missão é projetar, implementar e gerenciar um banco de dados PostgreSQL que servirá como base para a plataforma de analytics de produção da CineTech. Este é um cenário real onde você aplicará todos os conceitos de DBA que aprendeu até agora.

Objetivos de Aprendizagem

- Projetar e implementar esquemas de banco de dados
 - Ingerir grandes conjuntos de dados de forma eficiente
 - Implementar controle de acesso adequado e gerenciamento de usuários
 - Criar views analíticas e realizar análise de dados
 - Trabalhar com Git para controle de versão e colaboração
 - Documentar operações e procedimentos de banco de dados
-

O Desafio dos Dados

A CineTech forneceu três arquivos de texto massivos contendo seus dados de produção:

Estrutura dos Arquivos de Dados

- **producao.txt** (34MB) - Contém informações de produção
 - Formato: producaoID##titulo##ano_producao##tipo_ID
 - Exemplo: 70625##Campanile d'oro, Il##1955##1
- **peessoa.txt** (47MB) - Contém informações de pessoas
 - Formato: pessoaID##nome
 - Exemplo: 2523##Accardi, A.J.
- **equipe.txt** (308MB) - Contém informações de equipe/elenco
 - Formato: pessoaID##producaoID##papel
 - Exemplo: 715190##37497##Nadine's Client

Tipos de Produção

O campo `tipo_ID` em `producao.txt` representa diferentes tipos de produções artísticas (parte do desafio é descobrir quais IDs representam quais tipos):

- Filmes
- Séries de TV
- Videogames
- Documentários
- Curtas-metragens
- Clipes de Música
- Produções Teatrais
- Séries Web
- Animações

Detalhamento das Tabelas

Temos como entrada três relações armazenadas:

- **Producao** (producaoID, titulo, ano_producao, tipo_ID)
- **Pessoa** (pessoaID, nome)
- **Equipe** (pessoaID, producaoID, papel)

A relação Producao contém o nome e o ano de produção de diversos tipos de produções artísticas como filmes, séries televisivas e videogames. O tipo específico de uma produção é definida pelo atributo tipo_ID. A chave primária desta relação é o atributo produçãoID. A relação Pessoa contém o nome de pessoas envolvidas em uma determinada produção como atores, atrizes, diretor e compositor. A chave primária desta relação é o atributo pessoaID. A relação Equipe representa o relacionamento M:N entre Producao e Pessoa (isto é, uma produção envolve muitas pessoas, uma pessoa pode participar em muitas produções). Para atores e atrizes de filmes, o atributo papel informa o nome do personagem. A chave primária desta relação é formada pelos atributos produçãoID e pessoaID, que são chaves estrangeiras referenciando Producao e Pessoa, respectivamente.

Exemplo de Consulta

Para produções do tipo filme – tipoID=1 – selecione o título do filme, ano de produção, nome e papel de todas as pessoas envolvidas no filme. Ordene os resultados por ID do filme.

Expressão em SQL

```
SELECT p.titulo, p.ano_produção, ps.nome, e.papel
FROM Producao as p, Pessoa as ps, Equipe as e
WHERE p.tipoID = 1
      AND p.producaoID = e.producaoID
      AND ps.pessoaID = e.pessoaID
ORDER BY p.producaoID
```

Exemplo de saída

```
Matrix, The | 1999 | Reeves, Keanu | Neo
Matrix, The | 1999 | Moss, Carrie-Anne | Trinity
Matrix, The | 1999 | Fishburne, Laurence | Morpheus
Matrix, The | 1999 | Weaving, Hugo | Agent Smith
```

📦 Fase 1: Projeto e Implementação do Banco de Dados

Passo 1: Criar o Banco de Dados Principal

```
-- Criar o banco de dados principal
CREATE DATABASE cinetech_productions;
```

Passo 2: Projetar o Esquema de Dados Brutos

Criar um esquema chamado raw_data com três tabelas:

1. **producao** - Armazenar todos os dados de produção
2. **pessoa** - Armazenar todos os dados de pessoas
3. **equipe** - Armazenar todos os relacionamentos de equipe/elenco

Requisitos:

- Usar tipos de dados apropriados
- Implementar chaves primárias adequadas
- Adicionar restrições de chave estrangeira
- Considerar estratégias de indexação para performance

Passo 3: Ingestão de Dados

Criar um script Python ou SQL para ingerir os dados dos arquivos de texto em suas tabelas.

Requisitos:

- Lidar adequadamente com o delimitador ##
- Tratar possíveis problemas de qualidade dos dados
- Implementar tratamento de erros
- Fornecer feedback de progresso para arquivos grandes
- Documentar seu processo de ingestão

Fase 2: Projeto do Esquema Analítico

Passo 4: Criar o Esquema de Analytics

Criar um novo esquema chamado analytics com tabelas especializadas para cada tipo de produção:

Tabelas:

- `movies` - Todas as produções de filmes
- `tv_shows` - Todas as produções de séries de TV
- `video_games` - Todas as produções de videogames
- `documentaries` - Todas as produções de documentários
- `short_films` - Todas as produções de curtas-metragens
- `music_videos` - Todas as produções de clipes de música
- `theater_productions` - Todas as produções teatrais
- `web_series` - Todas as produções de séries web
- `animations` - Todas as produções de animação

Requisitos:

- Cada tabela deve conter apenas produções daquele tipo específico
- Incluir todos os campos relevantes da tabela original `producao`
- Adicionar índices apropriados para padrões de consulta comuns
- Considerar estratégias de particionamento para tabelas grandes

Fase 3: Gerenciamento de Usuários e Controle de Acesso

Passo 5: Criar Usuários Especializados

Criar os seguintes usuários com permissões apropriadas:

1. `analyst_movies` - Pode acessar apenas dados relacionados a filmes
2. `analyst_tv` - Pode acessar apenas dados relacionados a séries de TV
3. `analyst_games` - Pode acessar apenas dados relacionados a videogames
4. `analyst_docs` - Pode acessar apenas dados relacionados a documentários
5. `analyst_all` - Pode acessar todos os dados analíticos (somente leitura)
6. `data_scientist` - Pode acessar todos os dados com permissões de escrita no esquema `analytics`

Requisitos:

- Implementar controle de acesso
- Usar declarações `GRANT/REVOKE` apropriadas
- Documentar a matriz de permissões
- Testar o acesso para cada usuário

Fase 4: Analytics e Relatórios

Passo 6: Criar Views Analíticas

Criar as seguintes views no esquema `analytics`:

1. `production_summary` - Estatísticas resumidas por tipo de produção
2. `top_actors_by_type` - Atores mais ativos por tipo de produção
3. `yearly_production_trends` - Tendências de produção ao longo do tempo
4. `crew_analysis` - Análise de papéis e participação da equipe

Passo 7: Realizar Análise de Dados

Escrever consultas SQL para responder estas questões de negócio:

1. Qual tipo de produção tem mais produções?
2. Quais são os 10 atores mais ativos em todos os tipos de produção?
3. Como o número de produções mudou nos últimos 50 anos?
4. Quais anos tiveram a maior produção?
5. Qual porcentagem de produções tem membros da equipe com papéis específicos?

Fase 5: Performance e Otimização (Extra, envolve pesquisa)

Passo 8: Otimização de Performance

- Analisar performance de consultas usando `EXPLAIN ANALYZE`
- Criar índices apropriados para suas consultas mais comuns
- Implementar particionamento de tabelas se benéfico
- Documentar suas estratégias de otimização

Passo 9: Estratégia de Backup e Recuperação

- Criar uma estratégia de backup para seu banco de dados
 - Documentar procedimentos de recuperação
 - Testar seu processo de backup e recuperação
-

Fase 6: Documentação e Submissão

Passo 10: Criar Documentação Abrangente

Criar a seguinte documentação:

1. Documentação do Esquema do Banco de Dados

- DER (Diagrama Entidade-Relacionamento)
- Definições de tabelas com restrições
- Estratégias de indexação

2. Documentação de Acesso de Usuários

- Matriz de permissões
- Scripts de criação de usuários
- Procedimentos de teste de acesso

3. Documentação de Ingestão de Dados

- Scripts de ingestão
- Verificações de qualidade dos dados
- Procedimentos de tratamento de erros

4. Documentação de Desempenho

- Estratégias de otimização de consultas
- Recomendações de índices
- Benchmarks de performance

Passo 11: Fluxo de Trabalho Git

Requisitos de Submissão:

1. Criar uma nova branch para seu trabalho:

```
git checkout -b homework/seu-nome
```

2. Organizar seus arquivos:

```
homework/  
├── sql/  
│   ├── 01_criacao_banco.sql  
│   ├── 02_projeto_esquema.sql  
│   ├── 03_ingestao_dados.sql  
│   ├── 04_gerencia_usuario.sql  
│   ├── 05_views_analytics.sql  
│   └── 06_consultas_analise.sql  
├── python/  
│   └── ingestao_dados.py  
├── docs/  
│   ├── documentacao_esquema.md  
│   ├── guia_acesso_usuarios.md  
│   ├── analise_performance.md  
│   └── guia_backup_recuperacao.md  
└── README.md
```

3. Fazer commit do seu trabalho com mensagens significativas:

```
git add .  
git commit -m "feat: implementar esquema de banco e ingestão de dados"  
git push origin homework/seu-nome
```

Critérios de Avaliação

Seu trabalho será avaliado com base em:

Critério

Projeto e Implementação do Banco de Dados
Ingestão e Qualidade dos Dados
Gerenciamento de Usuários e Segurança
Analytics e Performance
Documentação e Fluxo de Trabalho Git

Bônus (Opcional)

- Implementar testes automatizados para seu banco de dados
- Criar um dashboard de monitoramento
- Implementar procedimentos de validação de dados
- Adicionar estratégias de arquivamento de dados
- Criar um plano de recuperação de desastres

Boa Sorte!

Esta é sua chance de demonstrar suas habilidades de DBA em um cenário real. Tome seu tempo, planeje cuidadosamente e crie algo do qual você se orgulhe. Lembre-se, boa administração de banco de dados é sobre mais do que apenas escrever SQL - é sobre projetar sistemas que são seguros, performáticos e sustentáveis.

Que suas consultas sejam rápidas e seus dados sejam limpos!