

Teste de JAVA com banco de dados

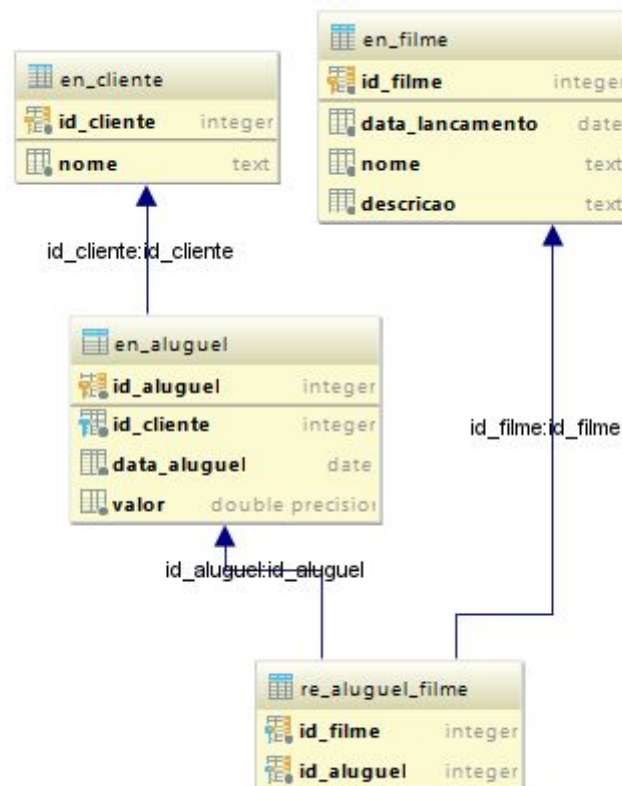
Considerando as tabelas especificadas abaixo, crie duas classes que implementam as interfaces “FilmeDAO” e “AluguelDAO” demonstrando o seu funcionamento na classe “Main”.

A aplicação tem um único objetivo: Realizar o registro de aluguéis de filmes, desconsiderando o status do aluguel do filme.

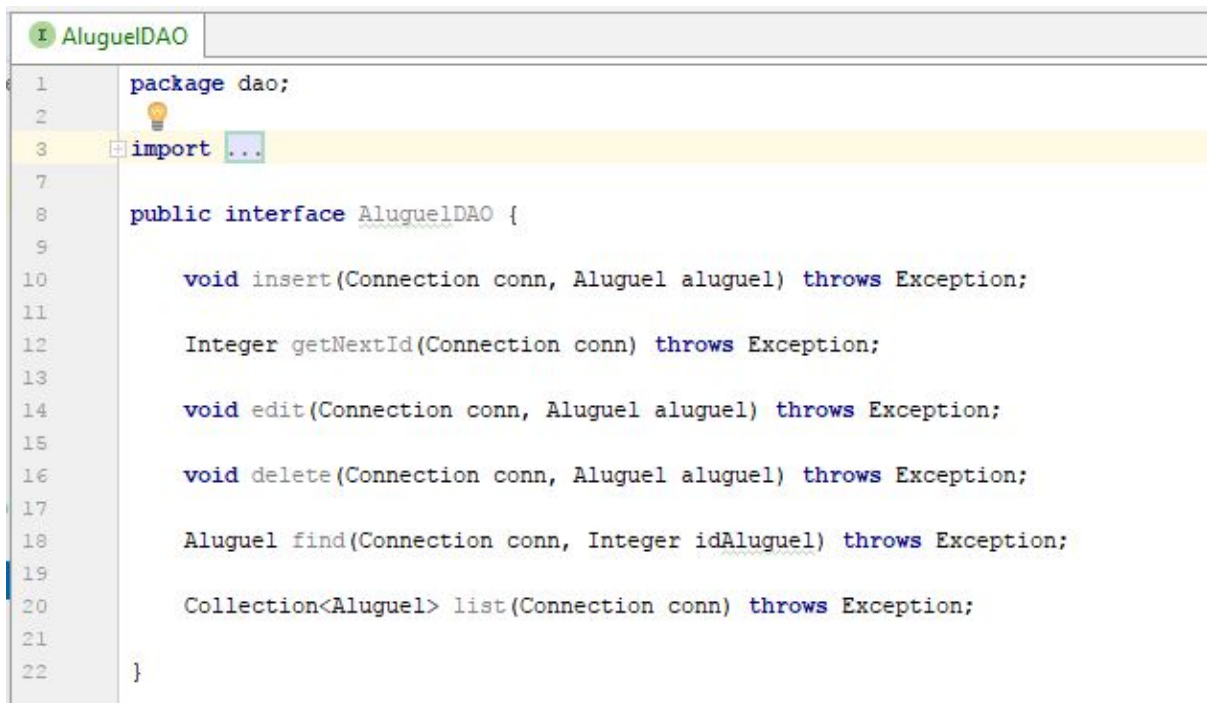
No final do teste, a aplicação deve possibilitar o CRUD (cadastro, listagem, alteração e deleção) de clientes, filmes e aluguéis.

Após realizar o desenvolvimento, utilize o arquivo decisoes-de-projeto.txt, presente junto ao código fonte, para explicar sobre suas decisões durante a implementação da solução.

Observação: Caso esteja realizando esse teste **remotamente**, execute os comandos SQL contidos no arquivo “app.sql” no banco de dados. Estes comandos irão criar as tabelas e popular com dados iniciais.



[Figura 1 - Modelo do banco de dados]

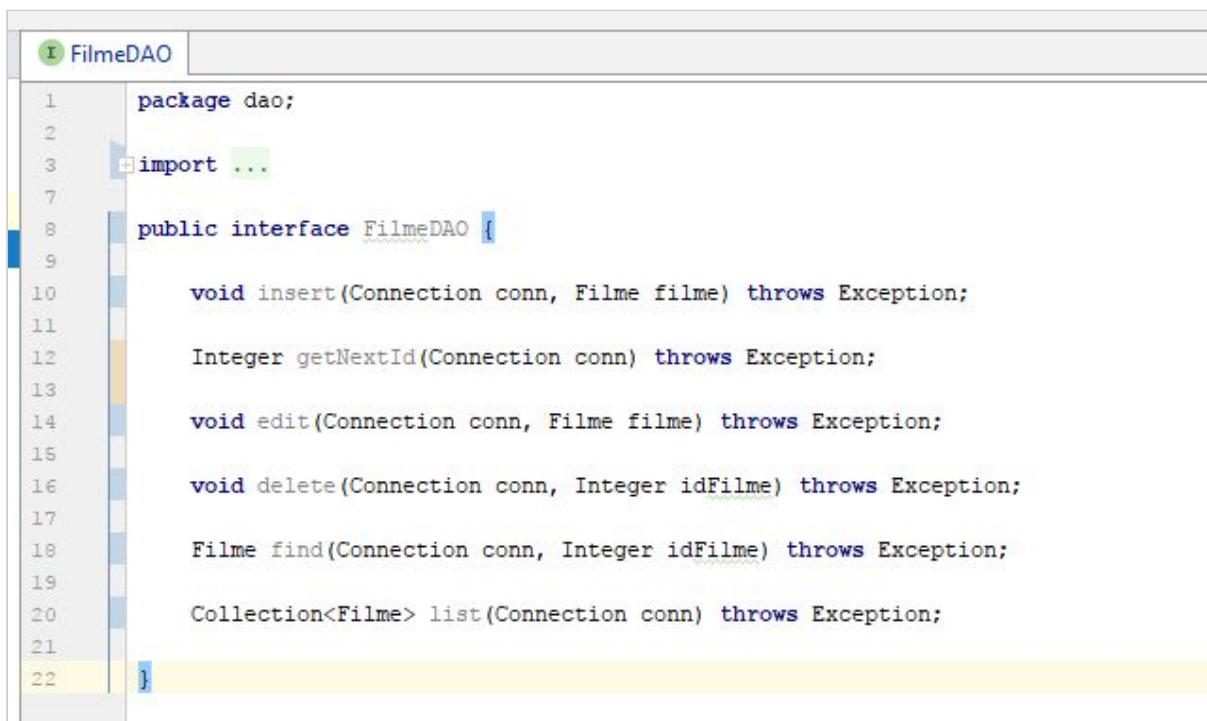


```

1 package dao;
2
3 import ...
4
5
6
7
8 public interface AluguelDAO {
9
10     void insert(Connection conn, Aluguel aluguel) throws Exception;
11
12     Integer getNextId(Connection conn) throws Exception;
13
14     void edit(Connection conn, Aluguel aluguel) throws Exception;
15
16     void delete(Connection conn, Aluguel aluguel) throws Exception;
17
18     Aluguel find(Connection conn, Integer idAluguel) throws Exception;
19
20     Collection<Aluguel> list(Connection conn) throws Exception;
21
22 }

```

[Figura 2 - Interface “AluguelDAO”]



```

1 package dao;
2
3 import ...
4
5
6
7
8 public interface FilmeDAO {
9
10     void insert(Connection conn, Filme filme) throws Exception;
11
12     Integer getNextId(Connection conn) throws Exception;
13
14     void edit(Connection conn, Filme filme) throws Exception;
15
16     void delete(Connection conn, Integer idFilme) throws Exception;
17
18     Filme find(Connection conn, Integer idFilme) throws Exception;
19
20     Collection<Filme> list(Connection conn) throws Exception;
21
22 }

```

[Figura 3 - Interface “FilmeDAO”]

1. Quando for referenciar a instância de uma classe que implementa uma interface, lembre-se de atribuir esta instância a uma variável do tipo interface.
2. Na implementação do método de **insert**, lembre-se que o sistema deve gerar o ID daquele novo registro. Nesta atividade estamos utilizando **Sequences**, uma solução do **Postgres** que gera o ID. Cada tabela tem uma **sequence** diferente, verifique o **pgadmin** para encontrar a **sequence** que você deve utilizar para o Filme e Aluguel.
3. A entidade **Filme** tem um relacionamento do tipo **N-N** com **Aluguel**, ou seja, um aluguel está relacionado a muitos filmes e um filme está relacionado a muitos aluguéis. Lembre-se disto quando estiver desenvolvendo.
4. A classe Date do java.util.Date é diferente do java.sql.Date então quando for manipular objetos no banco de dados será necessário transformar os valores.
5. O banco de dados não permite transações que insiram um estado inconsistente ao banco. Quando for implementar a exclusão de Filme, lembre-se também de excluir o seu relacionamento com Aluguéis.
6. Para implementar a edição de aluguéis, pode ser mais fácil limpar o relacionamento com filmes e inserir novamente os dados corretos.
7. Para imprimir na tela os dados de uma classe, lembre-se de implementar o método **toString**. Isto deixa o código mais fácil de se entender.
8. Foque primeiro na implementação da integração com o banco de dados. Deixe para o final a demonstração da sua solução.