Universidade Tecnológica Federal do Paraná - UTFPR

Relatório Projeto Final - ICSB30

Discentes:
João André Lima Lanzarini
Pedro Henrique de Freitas Neves.

Conteúdo

1 Introdução e Requisitos

O tabalho final da disciplina de Introdução A Banco de Dados tem como objetivo a implementação de uma ferramenta "semi-gráfica"que permite a visualização de bancos de dados MySQL e PostgreSQL. A ferramenta, chamada de *dbMonitor*, foi desenvolvida em Java e deve atender os seguintes requisitos:

- Ser capaz de se conectar a um banco de dados PostgreSQL e MySQL;
- Gravar dados de conexão;
- Mostrar graficamente (em árvore) as tabelas e views do baco de dados;
- Mostrar os campos de cada tabela e seus tipos e tamanho;

  Apontar as chaves primárias de cada tabela.
- Consultar dados da tabela;
- Dados mostrados em estutura de tabelas;
- Mostrar todos os dados (limitado a 1000 registros - configurável);
- Permitir consultas gerais (digitando o SQL diretamente);
- Exportação de dados das tabelas e das consultas em CSV ou JSON;

Seguindo esses requisitos, o grupo optou por desenvolver a ferramente em Java, utilizando a biblioteca *JDBC* para a conexão com os bancos de dados, como foi demonstrado em aula, e utilizando o terminal como interface gráfica.

2 Dependências

Para a execução do projeto, é necessário ter instalado o Java Development Kit (JDK) 22 , o MySQL e o PostgreSQL. Além disso, é necessário adicionar o driver JDBC do MySQL e do PostgreSQL ao projeto. Para isso, basta baixar os arquivos *.jar* dos drivers e adicioná-los ao projeto.

3 Implementação

Para implementar o projeto, foram criadas três classes, sendo elas as seguintes:

- **Main:** Classe principal do projeto, responsável por instanciar as demais classes e gerenciar a interação com o usuário. Ela é responsável por exibir o menu principal e chamar os métodos das outras classes.

- **DbConnection:** Classe responsável por gerenciar a conexão com o banco de dados, tanto MySQL quanto PostgreSQL, a partir da biblioteca JDBC.
- **Table:** Classe responsável por implementar uma estrutura de dados que armazena os elementos das tabelas e views do banco e imprime as linhas e colunas de maneira compatível no terminal

3.1 Main

A classe Main é a classe principal do projeto, responsável por instanciar as outras classes, gerenciar a interação com o usuário e executar as consultas do banco de dados. Os seus principais atributos são:

```
private static DbConnection connection;
private static boolean queryMode;
private static boolean isRunning;
```

Na função principal, existe um loop que utiliza o booleano *isRunning* para manter o programa em execução. Dentro desse loop, é exibido um menu com as opções disponíveis para o usuário, e a opção escolhida é passada para um switch que chama os métodos de acordo com a opção escolhida.

Além disso, o atributo *connection* é um objeto da classe DbConnection utilizado para criar uma conexão com o banco de dados e executar queries, passando o conjunto de dados de resultado para o método que utiliza os dados.

Por fim, o atributo *queryMode* é um booleano que indica se o usuário está em modo de consulta ou não. Se estiver, o programa exibe um prompt para o usuário digitar a query, e se não estiver, o programa exibe o menu principal.

3.2 DbConnection

A classe DbConnection é responsável por gerenciar a conexão com o banco de dados, a partir da biblioteca JDBC. Esses são os principais atributos da classe:

```
private Connection con;
private Statement stm;
private boolean con_status;
```

O atributo *con* é um objeto da classe Connection, que representa a conexão com o banco de dados. O atributo *stm* é um objeto da classe Statement, que é utilizado para executar queries no banco de dados. Por fim, o atributo *con_status* verifica se a conexão foi estabelecida corretamente. Todos esses atributos são utilizados na

classe Main para executar as queries, exibir resultados e para saber se a conexão foi estabelecida ou não.

3.3 Table

A classe Table é responsável por implementar a estrutura de dados de uma tabela gerada a partir do conjunto de resultados de uma query. Esses são os principais atributos da classe:

```
private List<String> data;
private int nColumns;
```

O atributo *data* é uma lista de strings que armazena os dados da tabela, e o atributo *nColumns* é um inteiro que armazena o número de colunas da tabela. Dessa forma, para criar uma tabela a partir de uma lista de dados, precisamos armazenar o número de colunas para poder separar as linhas corretamente.

Essa classe também possui métodos para imprimir a tabela no terminal, de acordo com o número de colunas e o tamanho dos dados.

4 Funcionalidades

A ferramenta implementada cumpre com os requisitos propostos a partir dos comandos que o usuário pode executar. A lista de comandos, obtidas ao digitar *help* no terminal, é a seguinte:

- **help** – Mostra a lista de comandos.
- **tables** – Mostra as tabelas do banco de dados.
- **info** [**table_name**] – Mostra os dados da tabela.
- **query** – Entra em modo de consulta.
- **tree** – Mostra as tabelas e views do banco de dados em árvore.
- **use** [**database_name**] – Muda o banco de dados.
- **max** [**number**] – Muda o número máximo de registros mostrados.
- **exit** – Sai do programa.

A maioria dos comandos são autoexplicativos e cumprem com os requisitos propostos, no entanto, o comando *query* é um pouco mais complicado. Ao digitar *query* na linha de comandos, o programa entra em modo de consulta, que sobrescreve a entrada do usuário para apenas consultas até que seja digitado *"end;"*. Nesse modo,

o usuário pode digitar diversas queries, terminadas por *";"*, e assim que sair do modo query, elas são executadas sequencialmente, com os resultados sendo exibidos no terminal.

Além disso, quando uma query é executada com sucesso, o programa da a opção para que o usuário salve o resultado em um arquivo CSV, com nome escolhido pelo usuário, como pedido por um dos requisitos.

5 Conclusão

O projeto foi implementado com sucesso, cumprindo com os requisitos propostos e com as funcionalidades esperadas. A utilização da biblioteca JDBC foi simples e efetiva, considerando que já havíamos utilizado ela em aulas anteriores, e a implementação da interface gráfica no terminal foi suave mas interessante, visto que foi necessário criar uma estrutura de dados para representar corretamente as tabelas do SQL.

Os aquivos fonte do projeto estão em anexo no .zip enviado junto com este relatório, e disponíveis no repositório acessado em Github.