Aplicativo de persistência em Banco de Dados

Angelo Victor Kraemer Foletto[1]

[1]Departamento de Ciência da Computação – Instituto Federal Catarinense – Unidade Urbana (IFC)
CEP 89160-202 – Rio do Sul – SC – Brasil

angelovkfoletto@gmail.com

Abstract. *Information is a primary resource for decision making, and it is with it that a solution can be found. It can be stored by various means, always in the form of data, digital spreadsheets, paper drafts, notepad on mobile devices, etc., or in a database. In this way, it is possible to guarantee scalability, transparency, performance, availability, among other "services" that a database can offer. And through a DBMS (database management system), the goal is to provide a pleasant experience to the user when the goal is to manipulate their data, transforming it into information. Through this, we have a simple DBMS conceived through the Java programming language, which is integrated - through a jdbc (java database connectivity) library - to a Structured Query Language (SQL) database.*

Resumo. *A informação é um recurso primordial para a tomada de decisão, e é com ela que se pode solução. Esta pode ser armazenada por diversos meios, sempre em forma de dados, planilhas digitais, rascunhos em papel, bloco de notas em dispositivos móveis, etc, ou em um banco de dados. Desta maneira, é possível garantir escalabilidade, transparência, performance, disponibilidade, dentre outros "serviços" que um banco de dados pode oferecer. E por meio de um SGBD (sistema de gerenciamento de banco de dados), objetiva-se proporcionar uma experiência prazerosa ao usuário quando o objetivo é manipular seus dados, transformando-os em informação. Por meio deste, tem-se a confecção de um simples SGBD, concebido por meio da linguagem de programação Java, a qual está integrada – por meio de uma biblioteca jdbc (*java database connectivity*) – a um banco de dados SQL (*Structured Query Language*).*

1. Introdução

A palavra informação não aparece de forma explícita, mas implicitamente fica evidenciada a sua necessidade, pois o ato verificar significa comparar as informações do que ocorreu com as informações do que foi, estabelecido [Cavalcanti, 1995].

A informação tem tornado-se importante e demandada com o passar dos anos e diferentes segmentos sociais dependem cada vez mais das várias fontes disponíveis destas para a realização de suas atividades [Bochner *et al*, 2011].

Não só no espaço global a informação torna-se um recurso importante, mas também no espaço local, onde atores e agentes estão mais voltados aos problemas da prática cotidiana (SENRA, 1999). Segundo Maria, Rosane e Claudio (2011), esta perspectiva é fundamental para o campo da saúde coletiva onde o território é o espaço privilegiado, tanto das leituras da saúde quanto do adoecimento.

De forma a garantir estes dados se faz necessários persisti-los em algum ambiente confiável e que esteja disponível à acesso quando necessário for. Ao referir-se a persistência, diversas maneiras de praticá-la veem em mente, sendo ela as mais comuns como papel, documento de planilha física, almoxarifados enormes, livros,

dentre outros. Mas a tecnologia está, a cada meia década, nos proporcionando métodos e formas eficazes de realizar a proteção e perpetuação dessas informações (por meio de dados).

Uma destas propostas, é o mantenimento deste dados em plataformas (*cloud*) disponibilizadas *online*, que possam ser acessadas de qualquer lugar no mundo, desde que exista acesso a rede mundial de computadores. Nestas plataformas, existem banco de dados capazes de realizar a persistência destas informações, de forma altamente escalável, dinâmica, transparente, segura, atômica, disponível, dentre outros.

Ao confeccionar uma aplicação através da linguagem de programação Java[1] capaz de por meio de uma classe pré-definida, persistir dados em um banco de dados SQL (Figura 1).

Figura 1: Esquema básico de comunicação entre um Banco de Dados e usuários.

Ao criar esta classe "genérica" é possível realizar operações sobre estes dados em um banco de dados (aqueles que estiverem disponíveis na aplicação). A aplicação converterá as variáveis presentes na classe "genérica" - previamente criada - e persistirá estas informações no BD (base de dados). Considera-se que o *database* não existir poderá ser criado pelo usuário manualmente acessando o banco, não por meio da aplicação; se as tabelas não existirem deverão ser criadas pelo usuário; se os dados existirem, poderão ser alterados, excluídos e consultados.

De maneira geral, a realização do CRUD (create, *read*, *update*, *delete*) será toda elaborada para disponibilizar uma plataforma de "gerenciamento" do seu banco de dados (ou uma simples aplicação SGBD), a fim de usuários (leigo) despreocupadas e não especialistas possam manipular dados de maneira simples e segura.

Sempre será necessário que as classes estejam condizentes com aquelas pré existentes no banco de dados que será manipulado. Objetiva-se por expandir esta, tornando-a dinâmica e também para a integração de uma maior gama banco de dados relacionas.

2. Métodos e Funções

Realizar CRUD em um banco de dados, por meio de uma aplicação construída por meio da linguagem de programação Java, que possibilita, através de uma API (*aplication program interface)* de integração JDBC[2] conectar um banco de dados e realizar

1 Disponível em: <https://www.java.com/>. Acesso em: 05 de dezembro de 2019.
2 Disponível em: <https://docs.oracle.com/javase/tutorial/jdbc/basics/index.html>. Acesso em: 05 de dezembro de 2019.

operações sobre o mesmo.

Por meio desta aplicação torna-se possível o gerenciamento básico de um banco de dados. Isso é realizado através de uma GUI (interface de integração com o usuário) desenvolvida por meio de FXML. Uma linguagem de marcação de interface do usuário baseada em XML criada pela Oracle Corporation para definir a interface do usuário de um aplicativo JavaFX [Brown, 2011].

2.1. Limitações

Por meio de uma classe modelada sobre a estrutura de tabelas e colunas de um banco de dados, assim vice-versa.

De maneira geral, se o banco de dados já possuir modelagem deve-se respeitá-la, modelando a classe que irá receber estes dados na aplicação. Mas se o mesmo não existir, pode-se criar uma classe com suas regras e modelar o BD de maneira a seguir as regras implementadas nesta.

Limita-se o uso deste aplicativo para funções básicas, ou CRUD. Sendo utilizado apenas de maneira didática para implementação e aplicação de conceitos firmados e desenvolvidos ao decorrer de um período disciplinar universitário.

2.2. Diagrama de Classe

O MVP – Mínimo Produto Viável – conta com um diagrama de classe simples (figura 2), com variáveis e métodos não definidos. As especificações do mesmo virá com a definição do escopo de aplicação do protótipo após aprovação.

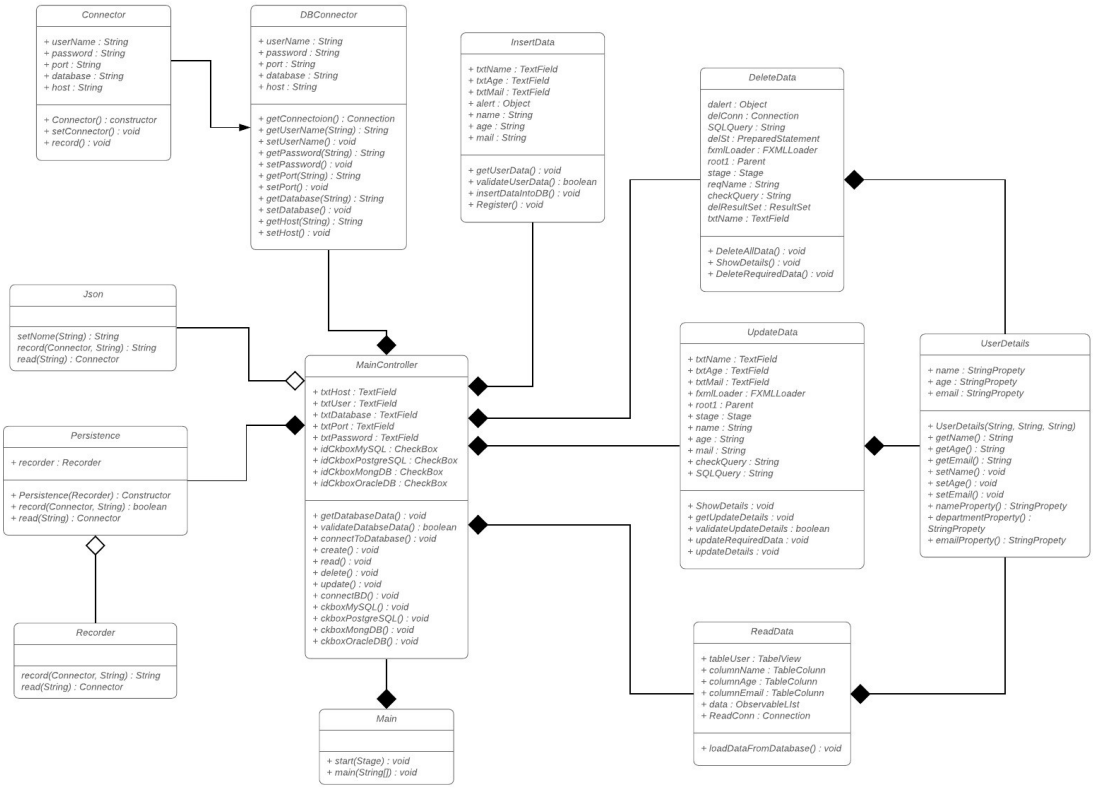

Figura 2: Diagrama de classe referente ao aplicativo.

De maneira geral, todas as classe comunicam-se com o MainController, sendo executado pelo Main.

Ao executar o aplicativo a classe Main chama o MainController (figura 3) exibindo então a tela inicial do mesmo. Após isto, exige-se como entrada os dados de conexão ao banco de dados para que todos os outros componentes possam operar de maneira correta.

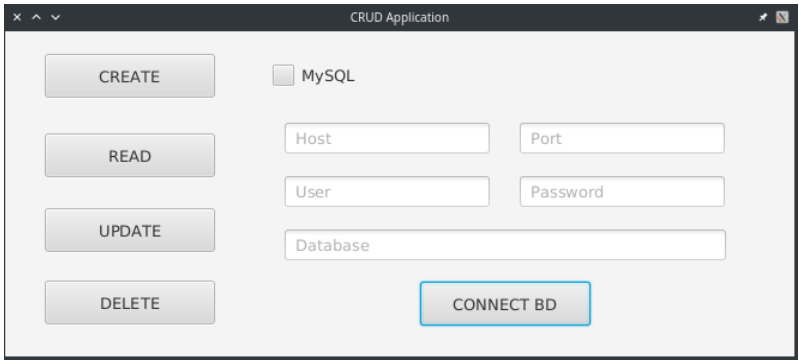

Figura 3: Tela principal do aplicativo.

Ao clicar no botão CONNECT BD, o aplicativo estará apto a executar as demais operações que o usuário desejar, listadas na figura 3 como sendo os botões: CREATE; READ; UPDATE; DELETE.

Cada tela tem como funcionalidade operar sua função descrita. De forma que o CRUD será realizado de maneira simples e intuitiva.

3. Conclusão

O emprego de SGBDs são extremamente importantes no dia a dia, seja o usuário classificado como leigo ou como profissional da área de Tecnologia da Informação. De maneira a facilitar estas funções, foi proposto uma solução por meio do desenvolvimento de um pequeno e simples gerenciador de banco de dados.

Quando emprega-se a utilização de um JDBC, facilita o gerenciamento da conexão entre a linguagem de programação – empregada neste – e o banco de dados aplicado. A interface gráfica, ou GUI, está simples e intuitiva.

Para trabalhos futuros, aconselha-se incluir novas funcionalidades e novas conexões com outros bancos de dados, como PostgreSQL[3], MongoDB[4] ou OracleDB[5].

Referências

Cavalcanti, Elmano Pontes. "Revolução da informação: algumas reflexões." Cadernos de Pesquisas em Administração-Programa de Pós-Graduação em Administração da FEA/USP 1.01 (1995): 40-46.

Bochner, Rosany, Maria Cristina Soares Guimarães, Rosane Abdala Lins de Santana, and Claudio Machado. "Qualidade da informação: a importância do dado primário, o princípio de tudo." (2011).

Greg Brown. Introducing FXML, A Markup Language for JavaFX. 2011. Disponível

3 Disponível em: <https://www.postgresql.org/>. Acesso em: 05 de dezembro de 2019.
4 Disponível em: <https://www.mongodb.com/>. Acesso em: 05 de dezembro de 2019.
5 Disponível em: <https://www.oracle.com/br/database/>. Acesso em: 05 de dezembro de 2019.

em: <http://fxexperience.com/wp-content/uploads/2011/08/Introducing-FXML.pdf>. Acesso em: dezembro de 2019.