Trabalho Final (2013/01)

Disciplina de Técnicas de Programação

A) Data de entrega: 02/07/2013
Apresentações: 02/07/2013

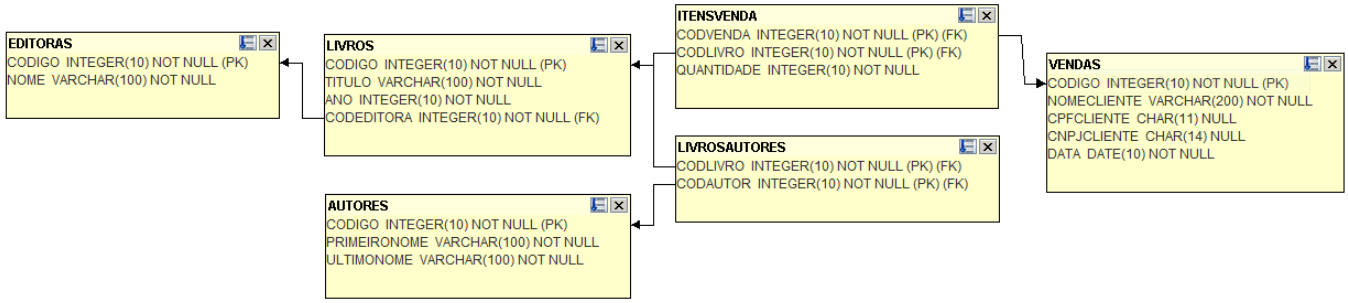
B) Objetivo:

O objetivo é consolidar o conhecimento sobre conceitos e construção de sistemas empresariais orientados a objetos em arquiteturas multicamadas através da exploração dos tópicos discutidos na disciplina de Técnicas de Programação.

C) Enunciado do problema:

Estamos interessados em um sistema de informação que mantenha um cadastro de editoras, com seus respectivos livros e autores, vendas realizadas e controle do pagamento de direitos autorais.

Os dados, disponibilizados em um banco de dados relacional, possuem as seguintes características (conforme a figura a seguir).



- Editoras – possui código (int, autogerado) e nome (string); possui diversos livros;
- Livros – possui código (int, autogerado), título (string) e ano (int); é associado a somente uma única editora; pode possuir vários autores;
- Autores – possui código (int, autogerado) e nome composto por primeiro nome (string) e útimo nome (string); pode ser autor de diversos livros;
- LivrosAutores – tabela associativa entre livros e autores;
- Vendas – possui código (int, autogerado), nome (string), CPF (string) ou CNPJ (string) do cliente, data da venda (date) e pode possuir vários itens de venda;
- ItensVenda – associa um livro a uma venda com a respectiva quantidade (int) vendida.

O código SQL para a criação das tabelas (de acordo com a sintaxe do Derby/JavaDB) é apresentado a seguir.

```
CREATE TABLE AUTORES(
CODIGO int NOT NULL GENERATED ALWAYS AS IDENTITY(START WITH 1, INCREMENT BY 1),
PRIMEIRONOME varchar(100) NOT NULL,
ULTIMONOME varchar(100) NOT NULL,
CONSTRAINT PK_AUTORES PRIMARY KEY (CODIGO))

CREATE TABLE EDITORAS(
CODIGO int NOT NULL GENERATED ALWAYS AS IDENTITY(START WITH 1, INCREMENT BY 1),
NOME varchar(100) NOT NULL,
```

```
CONSTRAINT PK_EDITORAS PRIMARY KEY (CODIGO))

CREATE TABLE LIVROS(
CODIGO int NOT NULL GENERATED ALWAYS AS IDENTITY(START WITH 1, INCREMENT BY 1),
TITULO varchar(100) NOT NULL,
ANO int NOT NULL,
CODEDITORA int NOT NULL,
CONSTRAINT PK_LIVROS PRIMARY KEY (CODIGO),
CONSTRAINT FK_LIVROS_EDITORAS FOREIGN KEY (CODEDITORA) REFERENCES EDITORAS(CODIGO))

CREATE TABLE LIVROSAUTORES(
CODLIVRO int NOT NULL,
CODAUTOR int NOT NULL,
CONSTRAINT PK_LIVROSAUTORES PRIMARY KEY (CODLIVRO,CODAUTOR),
CONSTRAINT FK_LIVROSAUTORES_LIVROS FOREIGN KEY (CODLIVRO) REFERENCES LIVROS(CODIGO),
CONSTRAINT FK_LIVROSAUTORES_AUTORES FOREIGN KEY (CODAUTOR) REFERENCES AUTORES(CODIGO))

CREATE TABLE VENDAS(
CODIGO int NOT NULL GENERATED ALWAYS AS IDENTITY(START WITH 1, INCREMENT BY 1),
NOMECLIENTE varchar(200) NOT NULL,
CPFCLIENTE char(11),
CNPJCLIENTE char(14),
DATA date NOT NULL,
CONSTRAINT PK_VENDAS PRIMARY KEY (CODIGO))

CREATE TABLE ITENSVENDA(
CODVENDA int NOT NULL,
CODLIVRO int NOT NULL,
QUANTIDADE int NOT NULL,
CONSTRAINT PK_ITENSVENDA PRIMARY KEY (CODVENDA,CODLIVRO),
CONSTRAINT FK_ITENSVENDA_LIVROS FOREIGN KEY (CODLIVRO) REFERENCES LIVROS(CODIGO),
CONSTRAINT FK_ITENSVENDA_VENDAS FOREIGN KEY (CODVENDA) REFERENCES VENDAS(CODIGO))
```

O cálculo do pagamento de direitos de autores de livros utilizado pode variar para uma determinada editora, contratos específicos ou mesmo ao longo do tempo. A cada mês, a editora contabiliza os valores a serem pagos como direitos aos autores de livros que pertencem ao seu catálogo.

Diferentes políticas podem ser utilizadas para o cálculo do valor dos direitos a serem recebidos pelos autores, entretanto, neste momento, duas políticas estão em uso (embora outras estejam em estudo):

Política 1:

- A cada mês, um valor fixo por livro é dividido de maneira igual entre todos os autores.
- Em função do volume de vendas do livro naquele mês, um adicional de porcentagem do total de vendas é pago (devendo ser dividido de maneira igual entre todos os autores).
- Por fim, um adicional fixo de exclusividade é pago a cada autor que não possui livros editados em outras editoras.

Política 2:

- O autor recebe um valor fixo, por livro, de acordo com o volume de vendas realizado no período:
    - Até 10 exemplares vendidos R$ 2,00 por exemplar;
    - Do 10º ao 20º exemplar vendido R$ 3,00 por exemplar;
    - A partir do 20º exemplar vendido R$ 4,00 por exemplar.

O sistema deve permitir:

- Realizar o cadastro de vendas de livros;
- Consultar a base de dados por diferentes critérios: editora, livro, autor, etc.;
- Permitir o cálculo dos valores mensais a serem pagos a cada autor de acordo com as regras apresentadas;
- Gerar relatórios de pagamentos de direitos autorais mensais (onde cada valor parcial que compõem o pagamento final a um autor deve aparecer de maneira individualizada além do somatório total) de acordo com uma determinada política selecionada.

D) Requisitos:

Os seguintes itens são obrigatórios na implementação do sistema:

- Arquitetura multicamada (pelo menos 3);
- Uso dos padrões de projeto explorados em sala de aula;
    - Uso de fachadas para isolar a camada de domínio da camada de apresentação;
    - Uso do padrão arquitetural "Domain model" na camada de domínio;
    - Uso obrigatório do padrão DAO/DTO na camada de persistência.
- Interface gráfica de usuário (desktop ou web serão aceitas, interface textual de console não será aceita);
- Persistência em banco de dados relacional, sendo obrigatório o uso do esquema de banco de dados apresentado na descrição do sistema;
- A camada de persistência deve ser implementada sem a utilização de frameworks mapeadores objeto/relacional (como JPA, Hibernate, etc);
- Tratamento correto do encapsulamento de exceções entre as camadas;
- O banco de dados deverá ter sido previamente populado (um arquivo contendo os scripts para geração do BD devem ser entregues juntamente com o código fonte) com, no mínimo:
    - 3 editoras
    - 10 autores
    - 15 livros
    - 50 vendas com pelo menos 200 itens de venda

E) Desenvolvimento, apresentação e avaliação do trabalho:

- O trabalho pode ser realizado individualmente ou em grupos de, no máximo, 3 alunos.
- O desenvolvimento do trabalho pode ser realizado em qualquer ambiente de programação e linguagem orientada a objetos.
- Os trabalhos serão apresentados no laboratório. Durante a apresentação, TODOS os alunos devem estar presentes e aptos a responder às perguntas. Respostas insatisfatórias por um aluno ou a sua ausência acarretarão descontos na nota final.
- A apresentação do trabalho é de inteira responsabilidade dos alunos (configuração da máquina, do ambiente de software, banco de dados, etc.) e o código-fonte utilizado deverá ser o mesmo entregue ao professor. É tarefa do grupo garantir que o sistema esteja apto a ser executado no dia da apresentação.
- Sistemas que não consigam ser executados ou apresentados no dia da apresentação receberão nota zero.
- Mensagens de erro apresentadas durante a execução do programa, mesmo que a aplicação não pare de executar, serão consideradas como erros de execução, e acarretarão descontos na nota do trabalho.
- Em caso de erro de sintaxe (compilação), o peso final do trabalho será valorado em zero.
- Em caso de erro de semântica (conteúdo), o peso final do trabalho sofrerá uma redução.
- Os trabalhos serão avaliados de acordo com critérios a serem estabelecidos pelo professor da disciplina, considerando o que é pedido no enunciado e o que foi realizado com sucesso pelo sistema. Também serão avaliadas a modelagem do sistema (correta criação das classes necessárias, com seus

atributos e métodos, encapsulamento, e correto estabelecimento de relações entre as classes) e sua implementação de acordo com os conceitos de orientação a objetos e arquitetura multicamada.
- A comprovação do uso de teste unitário, padrões de projeto e de programação por contratos será levada em conta na avaliação do trabalho.
- ***Trabalhos copiados resultarão em nota zero para todos os alunos envolvidos.***

F) Entrega do trabalho:

- Todos os arquivos necessários a execução do sistema, bem como os arquivos-fonte e os arquivos de documentação, deverão ser empacotados em um único arquivo (.zip) e submetidos através do sistema Moodle até a data de entrega.
- Devem fazer parte da documentação (documento texto) pelo menos:
    - Diagrama de classes do sistema. O diagrama de classes deverá ser entregue junto com documentação em texto salientando os pontos onde foram utilizados padrões de projeto na implementação da solução. É importante que as classes estejam agrupadas em pacotes de acordo com a arquitetura de camadas do sistema.
    - Os diagramas devem estar disponíveis em imagens com resolução suficiente e de fácil visualização. Não serão aceitos diagramas que estejam em formato original da ferramenta de desenho (como Visio, Astah, e outros).
- Não serão aceitos trabalhos enviados por correio eletrônico.
- Não serão aceitos trabalhos enviados fora do prazo estabelecido.