**Desenvolvimento de Sistema de Banco de Dados – Biblioteca – SQL Server - Java**

Este projeto aplica assuntos estudados nas disciplinas Técnicas de Programação e Bancos de Dados. Deve ser desenvolvido em duplas, de preferência as mesmas já estabelecidas nos projetos de Técnicas de Programação.

1. **Descrição da situação-problema**

Uma prefeitura municipal possui diversas bibliotecas públicas distribuídas pelo município. Cada biblioteca possui um número e um nome que a identifica. A prefeitura está passando por um processo de automatização de suas atividades e, portanto, deseja que um sistema de banco de dados seja desenvolvido, para ser executado em uma rede local da prefeitura (intranet), sendo que esse sistema será acessível pelas diversas bibliotecas, para operação das mesmas.

Como a operação será realizada localmente, optou-se por um sistema em modo desktop com um servidor de banco de dados relacional centralizado.

Cada uma das bibliotecas do município possui um acervo de livros que podem ser emprestados a leitores cadastrados como usuários do sistema de bibliotecas do município.

Cada livro tem uma área à qual está associado. Por exemplo: Literatura Brasileira, Literatura Estrangeira, Matemática, Geografia, Computação. Cada área, portanto, deve ser cadastrada e cada livro deve ser associado a uma das áreas (apenas uma).

A biblioteca mantém um fichário de autores dos livros, com informações como: seus nomes, país de origem, país de residência, data de nascimento. Um livro possui um autor principal, a ele associado, e um autor pode ter escrito vários livros.

Cada livro tem um código internacional padrão, de 13 caracteres, chamado ISBN. No entanto, no sistema municipal de bibliotecas, cada título (livro) é identificado por um código interno, de 6 caracteres, compartilhado por todas as bibliotecas.

Um livro pode ter vários exemplares, numerados de 1 a n, que representam, cada um, um livro físico do acervo. Os leitores tomam emprestados os **exemplares**, portanto. Mas cada exemplar pertence **fisicamente** a uma das bibliotecas da cidade, de maneira que, mesmo que o código do livro e do exemplar possam se repetir pelas diversas bibliotecas, é necessário saber em qual biblioteca o exemplar **físico está localizado**.

Um leitor é uma pessoa física cujos dados cadastrais atualmente são mantidos em fichários de papel. Cada ficha possui um número sequencial, que é usado para identificar o leitor rapidamente e associá-lo ao empréstimo de um exemplar.

Um leitor pode ter consigo, simultaneamente, até 5 livros emprestados, no máximo. Esse controle de empréstimos é feito em uma ficha de papel que fica armazenada em cada exemplar dos livros, informando a data de empréstimo, o número da ficha do leitor, a data prevista de devolução e o número de dias em atraso na devolução do livro.

A devolução de um exemplar de um livro emprestado só pode ocorrer na **mesma biblioteca física** onde o empréstimo teve origem. Caso seja devolvido com atraso em relação ao prazo de devolução previsto, deve-se registrar o número de dias em atraso e enviar um e-mail ao leitor, informando o valor da multa que deverá ser paga antes que o leitor possa tomar outros livros emprestados. Ou seja, o atraso na devolução de um livro torna o leitor suspenso.

1. **Sistema de Biblioteca**



/\* Logico\_Biblioteca: \*/

--Create Schema SisBib

CREATE TABLE SisBib.Biblioteca (

idBiblioteca int PRIMARY KEY Identity,

nome varchar(50)

);

CREATE TABLE SisBib.Livro (

codLivro varchar(6) PRIMARY KEY,

titulo varchar(100),

idAutor int,

idArea int

);

CREATE TABLE SisBib.Leitor (

idLeitor int PRIMARY KEY Identity,

nome varchar(50),

estaSuspenso char(1)

);

CREATE TABLE SisBib.Emprestimo (

idEmprestimo int PRIMARY KEY Identity,

idLeitor int,

idExemplar int,

dataEmprestimo Date,

devolucaoEfetiva Date,

devolucaoPrevista Date

);

CREATE TABLE SisBib.Autor (

idAutor int PRIMARY KEY Identity,

nome varchar(50)

);

CREATE TABLE SisBib.Area (

idArea int PRIMARY KEY,

nome varchar(50)

);

CREATE TABLE SisBib.Exemplar (

idExemplar int PRIMARY KEY Identity,

idBiblioteca int,

codLivro varchar(6),

numeroExemplar int

);

ALTER TABLE SisBib.Livro ADD CONSTRAINT FK\_Livro\_2

FOREIGN KEY (idAutor)

REFERENCES SisBib.Autor (idAutor)

ON DELETE CASCADE;

ALTER TABLE SisBib.Livro ADD CONSTRAINT FK\_Livro\_3

FOREIGN KEY (idArea)

REFERENCES SisBib.Area (idArea)

ON DELETE CASCADE;

ALTER TABLE SisBib.Emprestimo ADD CONSTRAINT FK\_Emprestimo\_1

FOREIGN KEY (idLeitor)

REFERENCES SisBib.Leitor (idLeitor)

ON DELETE SET NULL;

ALTER TABLE SisBib.Emprestimo ADD CONSTRAINT FK\_Emprestimo\_3

FOREIGN KEY (idExemplar)

REFERENCES SisBib.Exemplar (idExemplar);

ALTER TABLE SisBib.Exemplar ADD CONSTRAINT FK\_Exemplar\_1

FOREIGN KEY (idBiblioteca)

REFERENCES SisBib.Biblioteca (idBiblioteca)

ON DELETE CASCADE;

ALTER TABLE SisBib.Exemplar ADD CONSTRAINT FK\_Exemplar\_2

FOREIGN KEY (codLivro)

REFERENCES SisBib.Livro (codLivro)

ON DELETE SET NULL;

Desenvolver uma aplicação em Java usando Swing e JDBC para acessar o banco de dados acima e efetuar as operações de uma biblioteca com ele.

As tabelas **Biblioteca**, **Area**, **Autor** e **Leitor** devem ser preenchidas diretamente no SSMS (ou outro) por meio de comandos Insert, sem que seja necessário codificar um programa de manutenção dessas tabelas. Atenção para o fato de que as chaves primárias são autoincremento, de forma que cada registro dessas tabelas terá um campo Identity gerado automaticamente no Insert e os valores desses campos terão de ser usados nas chaves estrangeiras das demais tabelas.

Formulários:

1. Frame Principal - Login no Banco de Dados, que deverá ler as credenciais do usuário no Servidor de BD e efetuar a conexão do programa com o BD dos alunos. O usuário informará qual biblioteca estará mantendo. Deve ter menu que chama os demais frames. Pesquisem como criar um frame principal que usa um menu para chamar os demais frames que serão criados abaixo.

|  |  |  |
| --- | --- | --- |
| duke | Esse é o mascote da linguagem Java, seu nome é Duke. | Flickr Sistema de Biblioteca | | |
| Livros... Exemplares... Empréstimos... Devoluções... | | |
|  | | |
| Servidor: |  | 26 Date Picker (Release 8) |
| Banco de Dados: |  |
| Usuário: |  |
| Senha: |  |
|  | Conectar |
| Biblioteca: |  |  |

Após conectar ao banco de dados, buscar os registros de biblioteca (idBiblioteca e nome) para exibir no JCombobox que permite selecionar qual biblioteca será usada e o habilite para uso [ cbxBiblioteca.setEnabled(true); ].

Nesse momento, habilitar o uso do menu superior [ menuSuperior.setEnabled(true) ], instanciar e exibir o frame correspondente à opção de menu selecionada pelo usuário. Passe para o frame escolhido o id da biblioteca selecionada logo acima. Para isso, cada frame deverá ter um **atributo idBibliotecaEscolhida** inteiro, que receberá o valor da PK da biblioteca escolhida, para que esse valor possa ser usado nos selects do frame para buscar apenas registros da biblioteca escolhida. frmLivros.setIdBiblioteca(cbxBiblioteca.itemselecionado) e dai frmLivros.setVisible(true).

1. Manutenção de Livros – usando o programa de manutenção de Departamentos como modelo, efetuar as operações de navegação, busca, inclusão, exclusão, alteração de livros do acervo da prefeitura.
2. Manutenção de exemplares dos livros da biblioteca informada no Login – usando o programa de manutenção de Departamentos como modelo, fazer manutenção dos exemplares dos livros da biblioteca selecionada. Somente essa terá seus exemplares mantidos durante a execução desse formulário.
3. Empréstimos de exemplares de livros da biblioteca informada no Login – relacionar leitor com exemplar de livro da biblioteca selecionada, informando as datas de realização do empréstimo e a data prevista para devolução. No momento do empréstimo, o campo devolucaoEfetiva ficará nulo, pois o livro ainda não foi devolvido. Mostrar os livros em atraso numa guia de um TabControl usando a view do item 6. Usar a trigger do item 7 para capturar exceções e impedir o empréstimo de ser feito, avisando o usuário.
4. Devoluções de exemplares de livros da biblioteca informada no Login – solicitar o idLeitor, o idLivro a ser devolvido e seu número de exemplar. Verificar se existe esse exemplar na biblioteca selecionada e, caso exista, registrar no campo devolucaoEfetiva da tabela de empréstimo (para esse exemplar e esse leitor ) a data em que foi devolvido. Caso haja atraso na devolução, registrar na tabela Leitor que o leitor ficou suspenso. Usar a stored procedure do item 8 para mudar a situação do leitor para suspenso.
5. Criar view para listar os livros em atraso e o valor de multa a ser cobrada de cada leitor. Cada dia de atraso corresponde a R$5,00.
6. Criar trigger para impedir que um exemplar já emprestado e ainda não devolvido seja emprestado novamente.
7. Criar stored procedure para atualizar a situação do leitor para suspenso.