Projeto Integrador – Banco de Dados I – Linguagem de Programação II

Andrey H. L. Zeferino, Vitor Araújo Lins, Nicole Guerreiro Diniz

Ciências da Computação – Centro Universitário Campos de Andrade (Uniandrade)

Curitiba – PR – Brasil

andrey.zeferino@gmail.com

Resumo: *Foi solicitado a criação de um sistema de cadastro, edição, visualização e exclusão de dados de alunos registrados no Instituto Politécnico do Norte usando a linguagem Java e o banco de dados MySQL. Para o desenvolvimento desse sistema utilizamos a IDE Eclipse para programar em Java e MySQL Workbench para criar o Modelo de Entidade de Relacionamento do banco de dados do sistema, assim como para criar seu script SQL para gerar tabelas, colunas e banco de dados utilizados.*

Abstract: Has request the development of a register, edit, view and delete of diverses student data registered in the Instituto Politécnico Do Norte using the Java programming language and database MySQL. For the development of this system has used Eclipse IDE for Java and MySQL Workbench for create the entity–relationship model of system database, that has used too for creation of tables, columns and databases with a SQL script

1. Introdução

O Instituto Politécnico do Norte tem como uma de suas principais funções a gestão de várias escolas superiores. Cada escola por sua vez possui Diretor, Professor, Aluno, Departamentos e Turmas, onde cada uma dessas entidades é gerida pelo sistema através de tabelas no banco de dados. E para que o usuário possa manejar os registros de alunos, o banco de dados é consultado e possivelmente modificado para remover e inserir novos registros através de um sistema elaborado em Java.

2. Banco de dados

Para o armazenamento e consultas de dados dos alunos salvos foi usado o Sistema Gerenciador de Banco de Dados MySQL.

2.1 DER e Entidades

A entidade Escola se relaciona uma vez com a entidade Departamento, a entidade Escola é composta por: Primary Key idEscola do tipo INT, atributo diretor do tipo VARCHAR, atributo nomeEscola do tipo VARCHAR.

A entidade Departamento se relaciona diversas vezes com a entidade Escola e uma vez com as entidades Professor, Estudante e Disciplina, Além disso uma das tuplas de Professor será o professor chefe de departamento através de um relacionamento um-para-um. A entidade Departamento é composta por: Primary Key idDepartamento do tipo INT, atributo nomeDepartamento do tipo VARCHAR, Foreign Key ProfessorChefeDepartamento do tipo VARCHAR, Foreign Key idEscola do tipo INT.

A entidade Professor se relaciona diversas vezes com a entidade Departamento e Disciplina, também um dos professores é chefe do departamento através de um relacionamento um-para-um, Professor também se relaciona uma vez com estudante. A entidade professor é composta por: Primary Key idProfessor do tipo INT, Foreign Key idDepartamento do tipo INT, Foreign Key idDisciplina do tipo INT, atributo estaSobContradoDeInvestigacao do tipo Booleano, atributo Nome do tipo VARCHAR, atributo CPF do tipo VARCHAR.

A entidade Disciplina se relaciona diversas vezes com a entidade Departamento e Estudantes, se relaciona uma vez com a entidade Turma. A entidade disciplina é composta por: PrimaryKey idDisciplina do tipo INT, atributo nomeDisciplina do tipo VARCHAR, Foreign Key idDepartamento do tipo INT.

A entidade Estudante se relaciona diversas vezes com as entidades Turma, Departamento, Disciplina e Professor. A entidade Estudante é composta por: PrimaryKey idEstudante do tipo INT, Foreign Key idTurma do tipo INT, Foreign Key idDisciplina do tipo INT, Foreign Key idDepartamentoDisciplinaEspecial do tipo INT, Foreign Key idProfessor do tipo INT, atributo nome do tipo varchar, atributo cpf do tipo varchar, atributo dataNascimento do tipo DATE.

A entidade Turma se relaciona diversas vezes com a entidade Disciplina e uma vez com a entidade Estudante e Aula, e ela é composta por: Primary Key idTurma do tipo INT, Foreign Key idDisciplina do tipo INT, Foreign Key idProfessor do tipo INT.

A entidade Aula se relaciona um vez com turma. É composta por: Primary Key idAula do tipo INT, atributo sala do tipo varchar, atributo horario do tipo DATETIME, Foreign Key idTurma do tipo INT.

A entidade escola representa as escolas superiores (ex: Escola Superior de TI), o Departamento representa os departamentos que cada escola superior pode ter, disponibilizando disciplinas, a Disciplina é as disciplinas que o departamento disponibiliza, Professor é os professores, onde cada professor é de um departamento especifico e pode dar aula de suas respectivas disciplinas, também pode ser impedido de dar aula através do atributo “estaSobContratoInvestigacao”, a entidade Turma representa uma turma de estudantes aonde uma turma pode ter sua Aula em um momento especifico. A tabela Estudante contém todos os dados necessarios para o cadastro e para o funciomento do CRUD para estudantes, este criado através da IDE Eclipse e a linguagem de programação Java

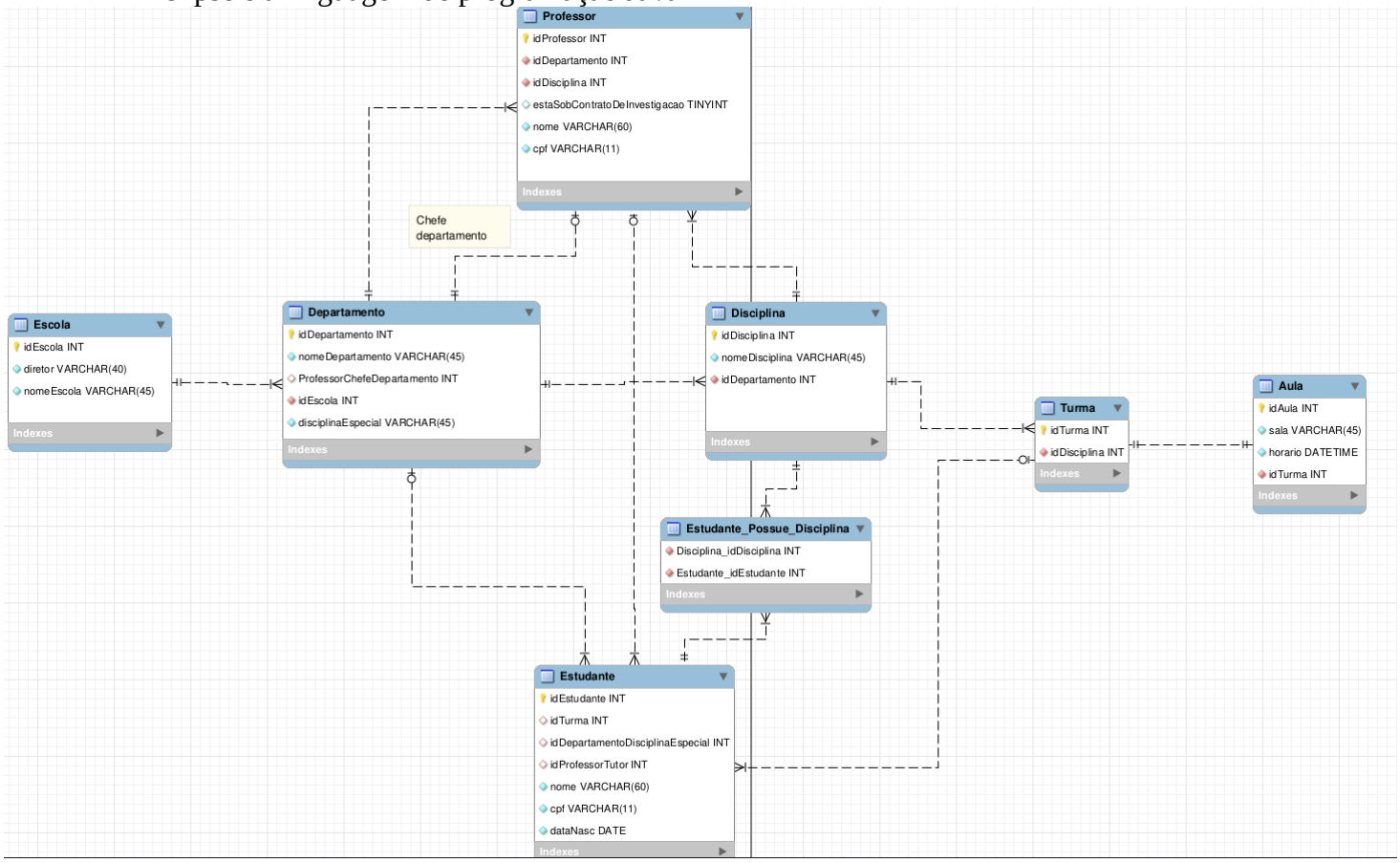

Figura 1. Diagrama de Entidade e Relacionamento (DER)

2.2 Script SQL

No script SQL foi feito o modelo físico planejado no Diagrama de Entidade e Relacionamento, com a criação de índices sobre o atributo nome das entidades Estudante, Professor, Disciplina e Departamento, além de ter sido inserido alguns valores fixos e constantes (Valores que não podem ser alterados pelo CRUD de Estudantes)

3. Programação

Para o desenvolvimento do CRUD de estudantes foi utiliado a linguagem Java, juntamente com seu paradigma de orientação a objetos, a seguir será apresentado o funcionamento do código junto com uma explicação de suas classes

3.1 Classes

Classe pessoa é composta por CPF do tipo String e nome do tipo String. E ela trabalha com 4 métodos: getNome() e getCpf(), ambos retornam uma String e são utilizados para receber o valor, setNome(String) e setCpf(String) são utilizados para aplicar o valor a um objeto

Classe Disciplina é composta por nome do tipo String e ID do tipo Integer. E ela trabalha com 2 métodos: getNome() que retorna uma String e getId() retornando um Integer, são utilizados para receber o valor de seus respectivos atributos.

Classe Professor é composta por ID que é um Integer e um método getID() que retorna um Integer, o construtor desta classe é responsável por preencher tanto o nome quanto o id do professor, os métodos e atributos de pessoas são herdados por Professor.

Classe Aluno é composta por dataNascimento do tipo String, disciplinas que é uma lista de no máximo 6 objetos da classe Disciplina, tutor um objeto da classe Professor, idEstudante do tipo Integer, departamento um objeto da classe Departamento e ela trabalha com 11 métodos: getDataNascimento() retorna String, getDisciplinas() retorna uma lista de disciplinas, getIdEstudante() retorna Integer, getDepartamento() retorna objeto Departamento, getTutor() retorna objeto Professor são utilizados para para receber os valores e setDataNasc(String), setDisciplina(Disciplina), setIdEstudante(Integer), setDepartamento(Departamento), setTutor(Professor) são utilizados para aplicar o valor a um objeto e PrintarInformacoes() que é responsável por exibir os dados do aluno de uma forma limpa e clara ao usuário. A classe Aluno herda os métodos e atributos da classe Pessoa

Classe Escola é a responsavel por todo o CRUD de estudantes, tendo ligação com a classe AlunoDAO que controla a gere o acesso, consultas, atualizações e novos cadastros no banco de dados, ela é composta por alunos que é uma lista de objetos Alunos, alunodao que é um objeto de AlunoDAO e teclado que é um objeto Scanner. Seu construtor é responsavel por inicializar o teclado e o alunodao, e também por preencher inicialmente a lista de alunos através do método listar() de alunodao. Escola é composta por 5 métodos públicos: criarAluno() que é responsavel pela criação e cadastro de um novo Aluno; ListarAlunos() que é responsavel por exibir os dados de todos os alunos salvos na lista alunos; PesquisarAluno() que pesquisa um aluno na lista e exibe todos seus dados; atualizarAluno() que é responsável pela atualização do cadastro de um aluno através do ID; removerAluno() que remove um aluno através do ID. Também tem 5 métodos privados para auxiliar os publicos, sendo eles: PerguntarInformacoesAluno(Aluno) que retorna um objeto Aluno, este método é responsável por pedir para o usuário todas as informações dos alunos que será cadastrado no sistema, este método auxilia os métodos public criarAluno() e atualizarAluno(); AtribuirIdentificadorAoAluno() este método auxilia o criarAluno() e é responsável pela atribuição de ID aos alunos; registrarDisciplinas() este método retorna uma lista de disciplina e é responsável pelo resgistro de disciplinas ao aluno, auxiliando então o método PerguntarInformacoesAluno(Aluno), este método possui um menu próprio , além de ter um laço do..while que pode ser repetido até no máximo 6 vezes, limitando assim as disciplinas cadastradas de todos os alunos para 6; BuscarAlunoPorID(int) método que auxilia os métodos removerAluno() e atualizarAluno(), ele retorna um objeto Aluno com base em seu ID; BuscarAlunoPorNome(String) este auxilia os métodos de pesquisa por nome, retorna um objeto Aluno com base em seu nome

Classe Main é composta por objeto escola que recebe a classe Escola. Ela trabalha com 2 métodos: main() é utilizada para chamar a função main, menu() é utilizado para chamar o menu do sistema, o menu tem 6 possibilidades: cadastrar que cadastra um novo aluno, invocando o método CriarAluno da escola; Listar que exibe uma lista de alunos com seus respectivos dados através do

método ListarAlunos da escola; Pesquisar que pesquisa um aluno invocando o método PesquisarAluno da escola; Atualizar que atualiza o cadastro de um aluno invocando o método AtlualizarAluno da escola; Remover que remove o cadastro de um aluno invocando o método RemoverAluno da escola; sair que fecha o software, caso digite um valor invalido será exibido na tela uma mensagem “Opção inválida”.

Classe ConnectionFactory é composta por connection um objeto de Connection responsável por toda a conexão ao MySQL, e as constantes Strings DRIVER_CLASS que carrega o driver de conexão ao banco de dados, DATABASE_URL que é a URL já com o banco de dados para ser conectado, USERNAME que é o usuário que será logado no banco de dados e PASSWORD que é a senha do usuário. Esta classe possuí o construtor privado que é responsável pelo carregamento do driver de conexão, possui 3 métodos dos quais 2 são statics: método privado createConnection() retorna um objeto Connection, este é responsável por gerar a conexão com o banco de dados através do DATABASE_URL, USERNAME e PASSWORD, método público static closeConnection() que é responsável por fechar e finalizar a conexão ao banco de dados, e método público static getConnection que retorna um objeto Connection, esta sendo o método principal que leva a conexão com banco de dados ao AlunoDAO

Classe AlunoDAO, está classe é a responsável por todo o CRUD de aluno com o banco de dados, armazenando e consultando os dados no MySQL, é composta por um atributo connection que é um objeto de Connection e 9 mensagens, sendo elas: FALHA_CONEXÃO, FALHA_OPERACAO, CADASTRO_SUCESSO, CADASTRO_FALHA, CONSULTA_VAZIA, ATUALIZACAO_SUCESSO, ATUALIZACAO_FALHA, EXCLUSAO_SUCESSO e EXCLUSAO_FALHA, possuí 5 métodos públicos: inserir(Aluno) que insere na tabela Estudantes os dados do objeto Aluno recebido como parâmetro, este método realiza 2 query SQL, uma para inserir e salvar o cadastro do aluno e outra para inserir e salvar o cadastro das disciplinas deste estudante se a inserção for um sucesso retorna true, caso contrário retorna false; atualizar(int, Aluno) responsável pela atualização do aluno no banco de dados, ele atualiza a linha na tabela Estudantes se essa linha ter o idEstudante que deseja atualizar, este método realiza 2 query SQL, uma para inserir e salvar o cadastro atualizado do id do aluno e outra para inserir e salvar o cadastro atualizados das disciplinas deste estudante se a atualização for um sucesso retorna true, caso contrário retorna false; Remover(int id) este método é responsável pela remoção de um cadastro de Estudante através de seu id, este método realizar 2 query SQL, uma para deletar as disciplinas do cadastro do aluno e outra para remover o cadastro do aluno se a remoção for um sucesso retorna true, caso contrário retorna false; listar() este método é responsável por trazer todos os alunos cadastrados no banco de dados através de uma lista de objetos do tipo Aluno, que após a finalização do método é retornado para quem chamou o método listar, caso a fila que será retornada esteja vazia será enviado uma mensagem de CONSULTA_VAZIA. Está classe também possuí 3 métodos privados auxiliares que em geral acessam outras tabelas que não seja a de Estudante: getDisciplina(int) este método é responsável por pegar as disciplinas do aluno através do relacionamento muitos-para-muitos entre as tabelas Estudante e Disciplina, retornando uma lista de todas as disciplinas que o estudante do id em parâmetro possuí em seu cadastro; getProfessor(int) este método é responsável por pegar o Professor atráves do relacionamento entre Estudante e Professor, o professor extraído através deste método é o professor tutor do aluno cujo o id está em parâmetro, retorna um objeto Professor; getDepartamento(int) este método acessa a tabela Departamento através do relacionamento entre Departamento e Estudante, retorna um objeto Departamento que está vinculado ao aluno cujo id está em parãmetro deste método.

3.2 Funcionamento

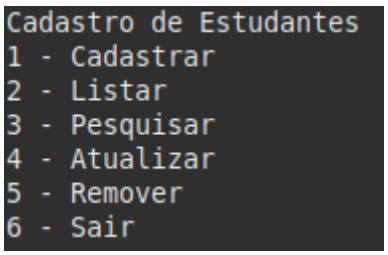

Figura 2. Menu Principal

Ao executar o código a primeira coisa que será vista é o menu principal mostrado acima, com cada uma das 6 opções disponibilizadas pela classe Main, ao escolher a opção cadastrar ou a opção atualizar será pedida os seguintes dados: nome do estudante, data do estudante (coloque no formato YYYY-MM-DD), CPF do estudante, o ID do professor tutor, o ID do departamento, e por fim vai ir para um menu secundario que é o menu de cadastro de disciplinas para o aluno, após cadastrar 6 disciplinas ou escolher a opção “0” o cadastro será finalizado com sucesso, caso seja uma atualização de cadastro, a primeira informação pedida é o ID que do cadastro que deseja atualizar. Na parte de cadastro, os únicos erros possíveis são os erros de pegar disciplinas, departamento ou professor não existentes no sistema, nesse caso o software pode acusar erro de valor inválido, além de erro de data, caso o valor inserido não seja no formato certo, por exemplo, “2000-24-12”.

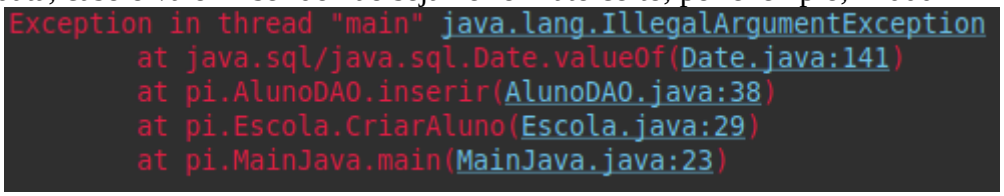

Figura 3. Erro de data em formato inválido

As opções Pesquisar e Listar exibem na tela as informações de todos os alunos, no caso de listar ou a de um aluno em especifico, no caso de pesquisar. A opção de pesquisa também necessita do nome do aluno. Ambas exibem na tela os dados dos alunos de uma forma clara e limpa.

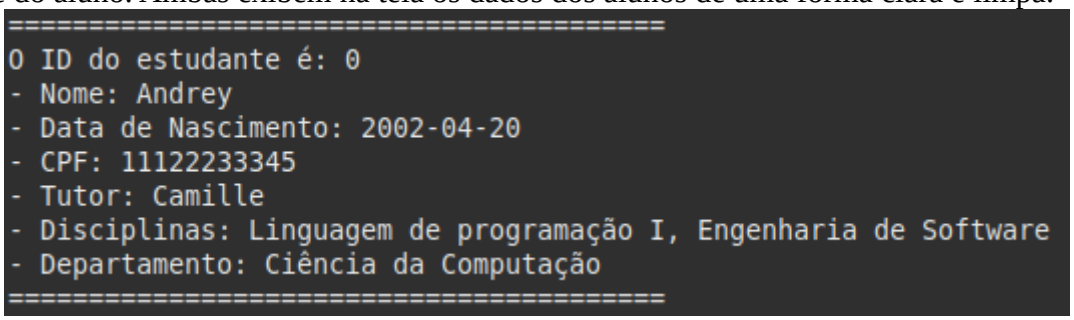

Figura 4. Exemplo de exibição de dados de aluno

A opção remover apenas remove um aluno e imprime uma mensagem de sucesso ou falha e a opção sair apenas finaliza o programa.

4. Conclusão

Após finalizar o projeto foi possível realizar cadastro,consultas e atualizações de dados de estudantes no banco de dados MySQL através de um código Java sem bugs, apenas algumas limitações como por exemplo o formato padrão de data do MySQL.

Referencias

DEITEL, Paul J.; DEITEL, Harvey M. (2010) Java: como programar. 8. ed. São Paulo: Ed. Pearson Prentice Hall.

ELMASRI, Ramez; NAVATHE, Shamkant B. (2005) Sistemas de Banco de Dados. 4. ed. São Paulo: Ed. Pearson Addison Wesley.

Christopher J. Date (1975), Introdução a Sistemas de Bancos de Dados, Elsevier 8th edition

Joshua Bloch (2001), Effective Java, ALTA BOOKS 3th edition