Nome: Adriano Moura da Silva

RA: 23814

Para desenvolver um sistema de base de dados que atenda às necessidades de uma faculdade, é necessário considerar diversos aspectos. Vamos explorar as respostas para as perguntas fornecidas e delinear uma estrutura de base de dados adequada para armazenar e manipular as informações de alunos, professores, cursos, matérias, turmas e notas.

1. **Principais necessidades do cliente:**
Os proprietários de universidades devem:
Armazene informações sobre alunos, professores, cursos, disciplinas e notas. Aulas de controle, incluindo alunos e professores. Registre as notas dos alunos por disciplina e turma. Obtenha relatórios ou análises do desempenho dos alunos.

2. **Que informações devem ser armazenadas:**

Alunos: dados pessoais e acadêmicos. Professores: dados pessoais e disciplinas ministradas. Cursos: informações sobre os diversos cursos oferecidos. Disciplinas (Disciplina): cada disciplina pertencente a um determinado curso. Tema: relações entre professores, alunos e disciplinas. Qualificações: qualificações atribuídas a cada aluno de acordo com as disciplinas.

3. Quais dados devem ser armazenados:

Alunos: nome, matrícula, data de nascimento, contato, curso. Professores: nome, matrícula, especialidade, disciplinas que ministram. Curso: nome do curso, duração, assuntos relevantes. Disciplinas (Disciplina): nome, código, carga horária, curso a que pertencem. Turmas: identificação da turma, disciplina, professor, alunos matriculados. Notas: nota final de cada aluno em cada disciplina.

4. **O que será feito com os dados posteriormente:**

Consultas sobre o desempenho dos alunos em cada disciplina. Acompanhamento do curso pelo docente. Controle de matrícula de alunos em turmas especiais. Geração de notas e relatórios de frequência. Análise dos sujeitos reprovados ou aprovados.

5. **Tabelas que precisam ser criadas**:
Aqui estão as principais tabelas necessárias para armazenar todas as informações:

Tabela de alunos

Tabela dos professores
Tabela de cursos
Tabela de disciplinas (disciplinas)
Tabela de turmas
Tabela de notas

6. Atributos de cada tabela

- `id_aluno` (chave primária, inteiro): Identificador único do aluno.
- `nome` (texto): Nome do aluno.
- `matricula` (texto): Número de matrícula.
- `data_nascimento` (data): Data de nascimento.
- `email` (texto): E-mail de contato.
  Tabela de professores

- `id_curso` (chave estrangeira, inteiro): Curso ao qual o aluno está matriculado.
- `id_professor` (chave primária, inteiro): Identificador único do professor.
- `nome` (texto): Nome do professor.
- `especialidade` (texto): Área de especialização.
- `matricula` (texto): Número de matrícula do professor.
- `email` (texto): E-mail de contato.
  Tabela:cursos

- `id_curso` (chave primária, inteiro): Identificador único do curso.
- `nome_curso` (texto): Nome do curso.
- `duracao` (inteiro): Duração do curso em semestres.
  Tabela: disciplinas

`id_disciplina` (chave primária, inteiro): Identificador único da disciplina.

- `nome_disciplina` (texto): Nome da disciplina.
- `carga_horaria` (inteiro): Carga horária da disciplina.
- `id_curso` (chave estrangeira, inteiro): Curso ao qual a disciplina pertence.
- **Tabela:Turmas**

- `id_turma` (chave primária, inteiro): Identificador único da turma.
- `id_disciplina` (chave estrangeira, inteiro): Disciplina oferecida nesta turma.
- `id_professor` (chave estrangeira, inteiro): Professor responsável pela turma.
- ano (inteiro): Ano da turma.
- `semestre` (inteiro): Semestre da turma.

Tabela: Notas

- `id_nota` (chave primária, inteiro): Identificador único da nota.
- `id_aluno` (chave estrangeira, inteiro): Aluno a quem a nota pertence.
- `id_turma` (chave estrangeira, inteiro): Turma na qual a nota foi atribuída.
- `nota_final` (decimal): Nota final do aluno na disciplina.

7. Tipos de dados de cada atributo

Inteiro: Para atributos como IDs, duração e ano/semestre.
Texto (VARCHAR): Para nomes, matrículas, e-mails.
Decimal: Para armazenar notas.
Data: Para armazenar datas de nascimento.

8. Relacionamentos a serem criados entre as tabelas

Aluno-Curso: Relacionamento de um para muitos. Um curso pode ter vários alunos, mas um aluno pertence a um único curso.
Curso-Disciplina: Relacionamento de um para muitos. Um curso pode ter várias disciplinas.
Professor-Turma: Relacionamento de um para muitos. Um professor pode ter várias turmas.
Disciplina-Turma: Relacionamento de um para muitos. Uma disciplina pode ter várias turmas.
Turma-Nota: Relacionamento de um para muitos. Uma turma pode ter várias notas.
Aluno-Turma (através da tabela de Notas): Relacionamento de muitos para muitos (um aluno pode estar em várias turmas, e uma turma pode ter vários alunos).
Modelo de Diagrama Entidade-Relacionamento (DER)
O diagrama seria útil para ilustrar a estrutura da base de dados, mostrando os relacionamentos entre as tabelas. Seria algo como:

Aluno (1
) Curso
Curso (1
) Disciplina
Disciplina (1
) Turma

Professor (1
) Turma
Aluno (N
) Turma (com Notas ligando ambos)
Com este esqueleto, a base de dados será capaz de armazenar e gerenciar as informações necessárias para a faculdade.

9 Objetivos de Aprendizagem (Plano de Aprendizagem)

Os objetivos de aprendizagem deste projeto são abrangentes das várias etapas do ciclo de desenvolvimento de bases de dados, a qual é importante para garantir a funcionalidade conforme a proposta da solução.

Análise dos requisitos

Compreender as necessidades dos clientes (faculdade) e identificar quais informações são necessárias para o funcionamento do sistema.

Identificar quais tipos de dados devem, na verdade, ser armazenados, como alunos, professores, turmas, notas etc.

Modelagem Conceitual

Uma representação abstrata da estrutura dos dados e dos relacionamentos entre as entidades (alunos, professores, cursos etc.) a partir das necessidades levantadas.

Definir as entidades e de relacionamentos entre elas, sem preocupação ainda com a implementação técnica.

Modelagem Lógica

Realizar a transposição da modelagem conceitual para um modelo lógico mais próximo da implementação.

Modelagem física

Definir tabelas, atributos e relacionamentos com base no modelo relacional, com foco em chaves primárias, chaves estrangeiras e cardinalidade dos relacionamentos. Modelagem Física, por meio de modelagem lógica usando SQL.

Geração do código que será utilizado para caracterizar o esquema do banco de dados, incluindo a criação de tabelas, definições de tipos de dados e restrições, como chaves estrangeiras e chaves únicas.

10. Entregável

O projeto será respondido conforme os seguintes itens:

a) Texto contido na seção "projeto" deste documento

Este texto identifica as exigências para o sistema de armazenamento de informações da instituição de ensino, de acordo com a especificação de requisitos apresentada acima.

b) Modelo conceitual para a solução do problema

O modelo conceitual pode ser expresso por um diagrama de entidade-relação (DER), que descreve as entidades principais e as relações entre elas. Para o projeto da instituição de ensino, temos:

Entidades:

Aluno

Professor

Curso

Disciplina

Turma

Nota

Relacionamentos:

Um Curso pode conter muitas Disciplinas.

Um Aluno está em um Curso.

Um Professor leciona várias Turmas, e cada Turma é uma Disciplina.

Um Aluno possui Notas em várias Turmas.

Exemplo de diagrama conceitual (em texto):

Aluno (id_aluno, nome, matricula, data_nascimento, email)

Curso (id_curso, nome_curso, duracao)

Disciplina (id_disciplina, nome_disciplina, carga_horaria, id_curso)

Professor (id_professor, nome, especialidade, matricula, email)

Turma (id_turma, id_disciplina, id_professor, ano, semestre)

Nota (id_nota, id_aluno, id_turma, nota_final)

c) Modelo lógico para solução do problema

No **Modelo Lógico**, vamos detalhar as tabelas, com as chaves primárias (PK), estrangeiras (FK) e os tipos de relacionamentos entre elas.

- **Aluno** (PK: `id_aluno`): Relacionado a **Curso** (FK: `id_curso`).
- **Curso** (PK: `id_curso`): Relacionado a **Disciplina** (FK: `id_curso`).
- **Disciplina** (PK: `id_disciplina`): Relacionado a **Turma** (FK: `id_disciplina`).
- **Professor** (PK: `id_professor`): Relacionado a **Turma** (FK: `id_professor`).
- **Turma** (PK: `id_turma`): Relacionado a **Disciplina** (FK: `id_disciplina`) e **Professor** (FK: `id_professor`).
- **Nota** (PK: `id_nota`): Relacionado a **Aluno** (FK: `id_aluno`) e **Turma** (FK: `id_turma`).

Exemplo de **Modelo Lógico**

CREATE TABLE Aluno (id_aluno INT PRIMARY KEY,

nome VARCHAR(100),

matricula VARCHAR(20),

data_nascimento DATE,

```
 email VARCHAR(100),
 id_curso INT,
 FOREIGN KEY (id_curso) REFERENCES Curso(id_curso)
 );
CREATE TABLE Curso (
 id_curso INT PRIMARY KEY,
nome_curso VARCHAR(100),
 duracao INT
 );
 CREATE TABLE Disciplina (
id_disciplina INT PRIMARY KEY,
 nome_disciplina VARCHAR(100),
 carga_horaria INT,

 id_curso INT,
 FOREIGN KEY (id_curso) REFERENCES Curso(id_curso)
 );
 CREATE TABLE Professor (
 id_professor INT PRIMARY KEY,
 nome VARCHAR(100),
 especialidade VARCHAR(100),
matricula VARCHAR(20),
 email VARCHAR(100)
 );
CREATE TABLE Turma (
 id_turma INT PRIMARY KEY,
 id_disciplina INT,
 id_professor INT,
```

```
 ano INT,
semestre INT,
FOREIGN KEY (id_disciplina) REFERENCES Disciplina(id_disciplina),
 FOREIGN KEY (id_professor) REFERENCES Professor(id_professor)
 );
 CREATE TABLE Nota (
id_nota INT PRIMARY KEY,
 id_aluno INT,
 id_turma INT,
 nota_final DECIMAL(4,2),
 FOREIGN KEY (id_aluno) REFERENCES Aluno(id_aluno),
 FOREIGN KEY (id_turma) REFERENCES Turma(id_turma)
 );
```

d) Modelo físico (código SQL)

O modelo físico é implementação prática das tabelas SQL, conforme descrito no modelo logico. O exemplo de código SQL fornecido a cima corresponde a esta parte.

e) Link do GITHUB contendo o código SQL

Após criar o código SQL, será necessário subir o arquivo para um repositório no GITHUB e disponibilizar o link n final no PDF final.

AUTOAVALIAÇÃO

A autoavaliação deve contemplar a questão:

Se eu fosse o usuário do sistema, o banco de dados seria apropriado para meu uso?

Alguns tópicos para reflexão

Completude: O banco de dados cobre todas as necessidades levantadas? É fácil acessar as informações dos alunos, notas e disciplinas de forma eficaz!

Desempenho: O sistema está adequado em termos de consultas e armazenamento, existem índices criados, onde necessário!

Manutenção: O banco de dados é flexível para modificação caso seja preciso, se novos cursos ou disciplinas forem incluídos, o sistema poderá ser modificado com estrutura necessária!

Segurança: Existem mecanismos para garantir a integridade dos dados, como chaves estrangeiras e restrições adequadas!

Refletir sobre estes itens será útil para verificar se o projeto é adequado ao uso final.

Com esse projeto lidar com os processos diários se tornara mais fácil.