

BANCO DE DADOS

Trabalho – Relatório

Curso:	Engenharia de Software
Aluno(a):	Yan Lucas Bezerra Silva
RU:	4351640

- **1ª Etapa – Modelagem**

Pontuação: 25 pontos.

Dadas as regras de negócio abaixo listadas, referentes ao estudo de caso de uma companhia aérea, elabore o Modelo Entidade-Relacionamento (MER), isto é, o modelo conceitual.

O Modelo Entidade-Relacionamento (MER) deve contemplar os seguintes itens:

- Entidades;
- Atributos;
- Relacionamentos;
- Cardinalidades;
- Chaves primárias;
- Chaves estrangeiras.

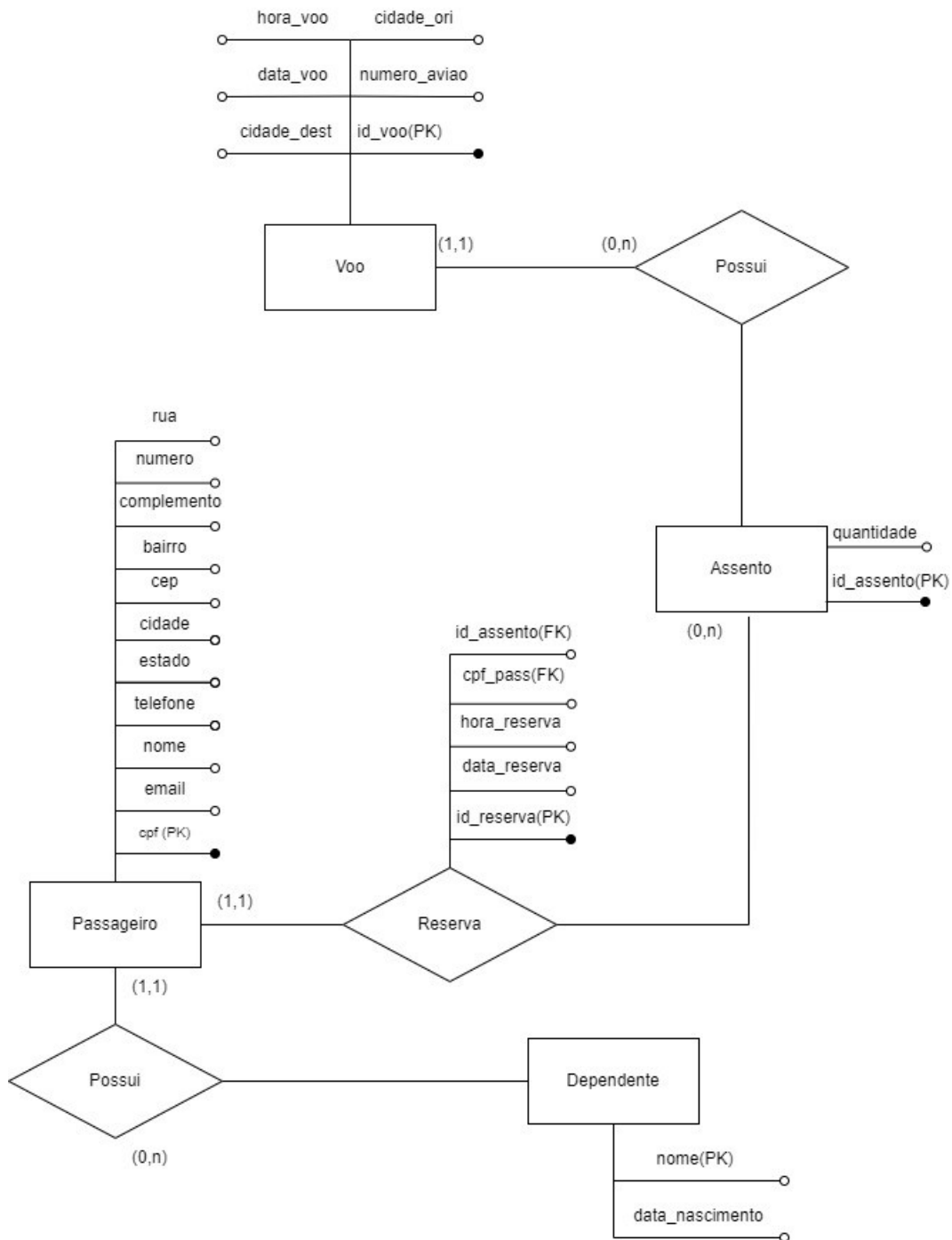
Uma companhia aérea necessita controlar os dados de seus voos. Para isso, contratou um profissional de Banco de Dados, a fim de modelar o Banco de Dados que armazenará os dados dos voos.

As regras de negócio são:

- Voo – Deverão ser armazenados os seguintes dados: identificação do voo, número do avião, cidade de origem, cidade de destino, data do voo e hora do voo;
- Assento – Deverão ser armazenados os seguintes dados: identificação do assento e

quantidade;

- Passageiro – Deverão ser armazenados os seguintes dados: CPF, nome, telefone, e-mail e endereço (rua, número, complemento, bairro, CEP, cidade e estado);
- Dependente – Deverão ser armazenados os seguintes dados: nome e data de nascimento;
- Um voo pode ter zero ou vários assentos, assim como zero ou vários assentos pertencem a um voo;
- Um passageiro pode ter zero ou várias reservas de assentos, assim como zero ou várias reservas de assentos pertencem a um passageiro;
- Um passageiro pode ter zero ou vários dependentes, assim como zero ou vários dependentes são de um passageiro;
- Da reserva deverão ser armazenados os seguintes dados: data da reserva e hora da reserva.



• 2ª Etapa – Implementação

Considere o seguinte Modelo Relacional (modelo lógico), referente ao estudo de caso de uma faculdade:

Com base no Modelo Relacional dado e utilizando a *Structured Query Language* (SQL), no MySQL Workbench, implemente o que se pede.

Observação: Para testar o Banco de Dados após a implementação, utilize os comandos contidos no arquivo “Trabalho – Populando o Banco de Dados” para popular as tabelas. Tal arquivo contém todos os comandos de inserção dos dados (fictícios) necessários para a realização dos testes.

Pontuação: 25 pontos.

- Implemente um Banco de Dados chamado “Faculdade”. Após, implemente as tabelas, conforme o Modelo Relacional dado, observando as chaves primárias e as chaves estrangeiras. Todos os campos, de todas as tabelas, não podem ser nulos (*not null*).

```
CREATE DATABASE Faculdade;
```

```
USE Faculdade;
```

```
CREATE TABLE Aluno(  
    idAluno INT NOT NULL,  
    matricula VARCHAR(10) NOT NULL,  
    nome VARCHAR(100) NOT NULL,  
    PRIMARY KEY (idAluno)  
);
```

```
CREATE TABLE Disciplina(  
    idDisciplina INT NOT NULL,  
    nome VARCHAR(100) NOT NULL,  
    cargaHoraria INT NOT NULL,  
    PRIMARY KEY (idDisciplina)  
);
```

```
CREATE TABLE Curso(  
    idCurso INT NOT NULL,  
    nome VARCHAR(100) NOT NULL,  
    PRIMARY KEY (idCurso)  
);
```

```
CREATE TABLE Historico(  
    idHistorico INT AUTO_INCREMENT PRIMARY KEY,
```

```

idAluno INT NOT NULL,
idDisciplina INT NOT NULL,
nota DECIMAL(5, 2) NOT NULL,
dataHistorico DATE NOT NULL,
FOREIGN KEY (idAluno) REFERENCES Aluno(idAluno),
FOREIGN KEY (idDisciplina) REFERENCES Disciplina(idDisciplina)
);

CREATE TABLE AlunoCurso(
    idAlunoCurso INT AUTO_INCREMENT PRIMARY KEY,
    idAluno INT NOT NULL,
    idCurso INT NOT NULL,
    anoEntrada INT NOT NULL,
    FOREIGN KEY (idAluno) REFERENCES Aluno(idAluno),
    FOREIGN KEY (idCurso) REFERENCES Curso(idCurso)
);

CREATE TABLE Grade(
    idGrade INT AUTO_INCREMENT PRIMARY KEY,
    idCurso INT NOT NULL,
    ano INT NOT NULL,
    cargaHorariaTotal INT NOT NULL,
    FOREIGN KEY (idCurso) REFERENCES Curso(idCurso)
);

CREATE TABLE GradeDisciplina(
    idGrade INT NOT NULL,
    idDisciplina INT NOT NULL,
    FOREIGN KEY (idGrade) REFERENCES Grade(idGrade),
    FOREIGN KEY (idDisciplina) REFERENCES Disciplina(idDisciplina)
);

```

Pontuação: 10 pontos.

- Implemente uma consulta para listar o quantitativo de cursos existentes.

```

SELECT          count(idCurso)          FROM          faculdade.curso;

```

```
1 • SELECT count(idCurso) FROM faculdade.curso;
```

Result Grid | | Filter Rows: | Export: | Wrap Cell Content:

	count(idCurso)
▶	9

Pontuação: 10 pontos.

- Implemente uma consulta para listar o nome das disciplinas existentes.

SELECT **nome** **FROM** **faculdade.curso;**

```
1 • SELECT nome FROM faculdade.curso;
```

Result Grid | | Filter Rows: | Export: | Wrap Cell Content:

	nome
▶	Análise e Desenvolvimento de Sistemas
	Banco de Dados
	Ciência de Dados
	Desenvolvimento Mobile
	Engenharia da Computação
	Engenharia de Software
	Gestão da Tecnologia da Informação
	Jogos Digitais
	Redes de Computadores

Pontuação: 10 pontos.

- Implemente uma consulta para listar o nome de todos os cursos e o nome de seus respectivos alunos. A listagem deve ser mostrada em ordem decrescente pelo nome dos cursos.

```
SELECT curso.nome AS nome_curso, al.nome AS nome_aluno FROM Curso AS curso LEFT JOIN AlunoCurso AS ac ON curso.idCurso = ac.idCurso LEFT JOIN Aluno AS al ON ac.idAluno = al.idAluno ORDER BY curso.nome DESC;
```

The screenshot shows a database management tool interface. On the left, a 'SCHEMAS' pane displays a tree view of the database structure, including tables like 'aluno', 'alunocurso', 'curso', 'disciplina', 'grade', 'historico', 'test', 'views', 'stored procedures', 'functions', and 'phpmyadmin'. The main area displays a SQL query in a text editor, which is the same query provided in the text block. Below the query, a 'Result Grid' shows the results of the query. The grid has two columns: 'nome_curso' and 'nome_aluno'. The results are sorted in descending order by course name. The table structure for 'curso' is also visible in the bottom left pane.

nome_curso	nome_aluno
Redes de Computadores	Nicole Amanda de Jesus
Redes de Computadores	Vitor Martins
Jogos Digitais	Beatriz Leopoldina
Jogos Digitais	João Augusto de Moura
Gestão da Tecnologia da Informação	Marian Miranda
Gestão da Tecnologia da Informação	Matheus Murilo de Souza
Engenharia de Software	Paula Roberta Vitorino
Engenharia de Software	Mario Vicente
Engenharia de Computação	Viviane Chaves Filha
Engenharia de Computação	Antônio Cozer
Desenvolvimento Mobile	Marta da Silva
Desenvolvimento Mobile	Luciano Tucolo
Ciência de Dados	Maria Helena Mantovani
Ciência de Dados	Guilherme Koerich
Banco de Dados	Ana Luiza de Paula
Banco de Dados	Lucas Cochuelo
Análise e Desenvolvimento de Sistemas	Alice de Souza
Análise e Desenvolvimento de Sistemas	Diogo Furlan
Análise e Desenvolvimento de Sistemas	Marcelo Luis dos Santos

Pontuação: 10 pontos.

- Implemente uma consulta para listar o nome das disciplinas e a média das notas das disciplinas em todos os cursos. Para isso, utilize o comando *group by*.

```
SELECT disciplina.nome AS nome_disciplina, AVG(hist.nota) AS media FROM Disciplina AS disciplina LEFT JOIN Historico AS hist ON disciplina.idDisciplina = hist.idDisciplina GROUP BY disciplina.nome;
```

Navigator: teste* curso

Limit to 1000 rows

```

1 SELECT disciplina.nome AS nome_disciplina, AVG(hist.nota) AS media FROM Disciplina AS disciplina LEFT
2 JOIN Historico AS hist ON disciplina.idDisciplina = hist.idDisciplina GROUP BY disciplina.nome;
3

```

Result Grid

nome_disciplina	media
Análise de Sistemas	85.000000
Arquitetura de Computadores	76.000000
Atividade Extensionista I	75.000000
Atividade Extensionista II	80.000000
Banco de Dados	85.000000
Empreendedorismo	89.000000
Engenharia de Software	70.000000
Fundamentos de Sistemas de Informação	75.000000
Gestão de Projetos de Software	70.000000
Lógica de Programação e Algoritmos	70.000000
Matemática Computacional	70.000000
Programação de Computadores	70.000000
Programação Orientada a Objetos	70.000000
Sistema Gerenciador de Banco de Dados	82.000000
Sistemas Operacionais	70.000000

Table: curso

Columns:

Pontuação: 10 pontos.

- Implemente uma consulta para listar o nome de todos os cursos e a quantidade de alunos em cada curso. Para isso, utilize os comandos *join* e *group by*.

```

SELECT curso.nome AS nome_curso, COUNT(alunoc.idAluno) AS total_alunos
FROM Curso AS curso LEFT
JOIN AlunoCurso AS alunoc ON curso.idCurso = alunoc.idCurso GROUP BY
curso.nome;

```

Navigator: teste* curso

Limit to 1000 rows

```

1 SELECT curso.nome AS nome_curso, COUNT(alunoc.idAluno) AS total_alunos FROM Curso AS curso LEFT
2 JOIN AlunoCurso AS alunoc ON curso.idCurso = alunoc.idCurso GROUP BY curso.nome;
3

```

Result Grid

nome_curso	total_alunos
Análise e Desenvolvimento de Sistemas	3
Banco de Dados	2
Ciência de Dados	2
Desenvolvimento Mobile	2
Engenharia da Computação	2
Engenharia de Software	2
Gestão da Tecnologia da Informação	2
Jogos Digitais	2
Redes de Computadores	2

Table: curso

Columns:

idCurso int(11) PK

nome varchar(100)