

BANCO DE DADOS

Trabalho – Relatório

Curso:	Análise e Desenvolvimento de Sistemas
Aluno(a):	Alexandre Monteiro Lima Cardoso
RU:	4518317

1. 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.

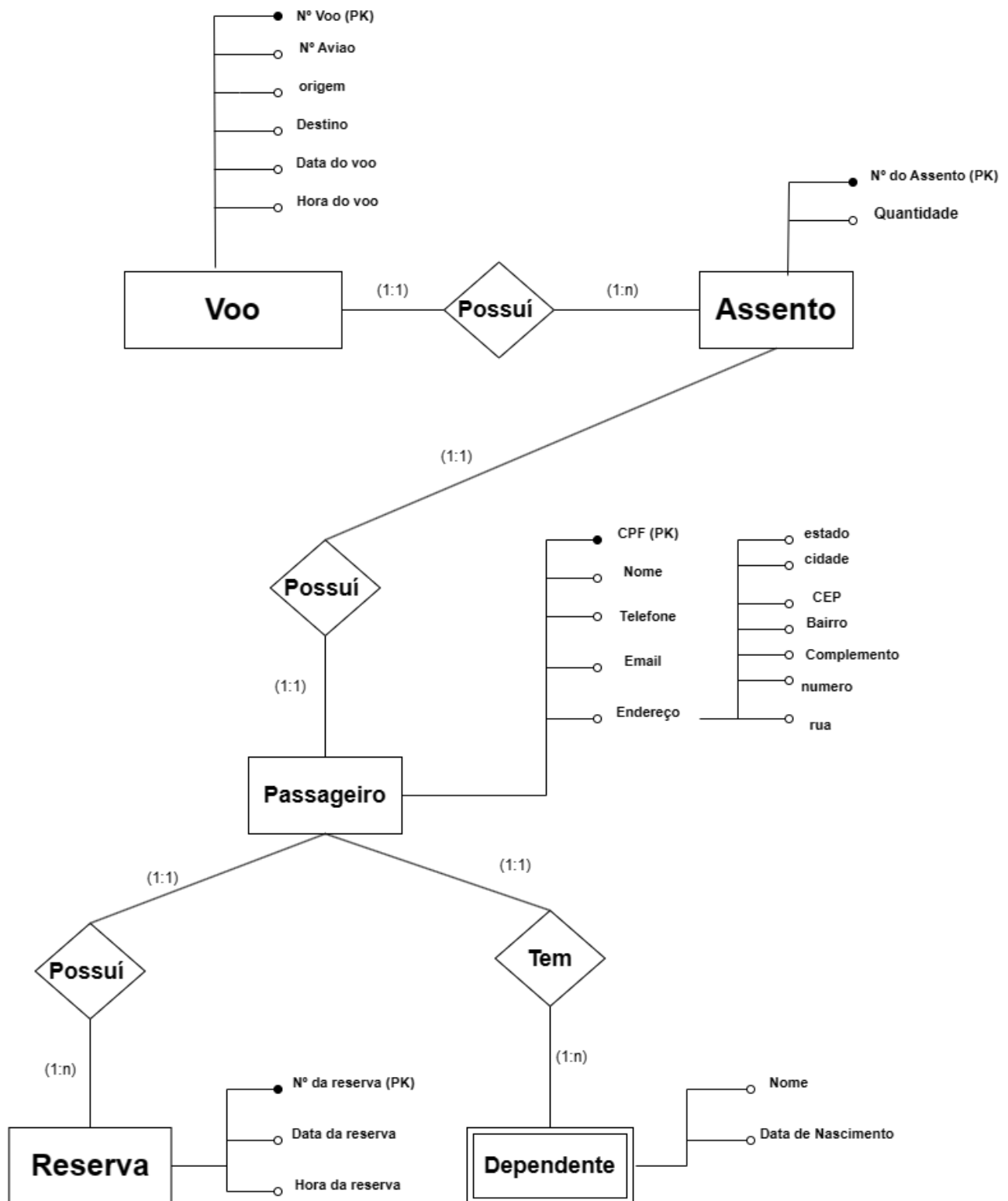
Uma companhia aérea necessita controlar os dados de seus voos. Para isso, contratou um profissional de Banco de Dados, a fim de modelar o Banco de Dados que armazenará os dados dos voos.

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;

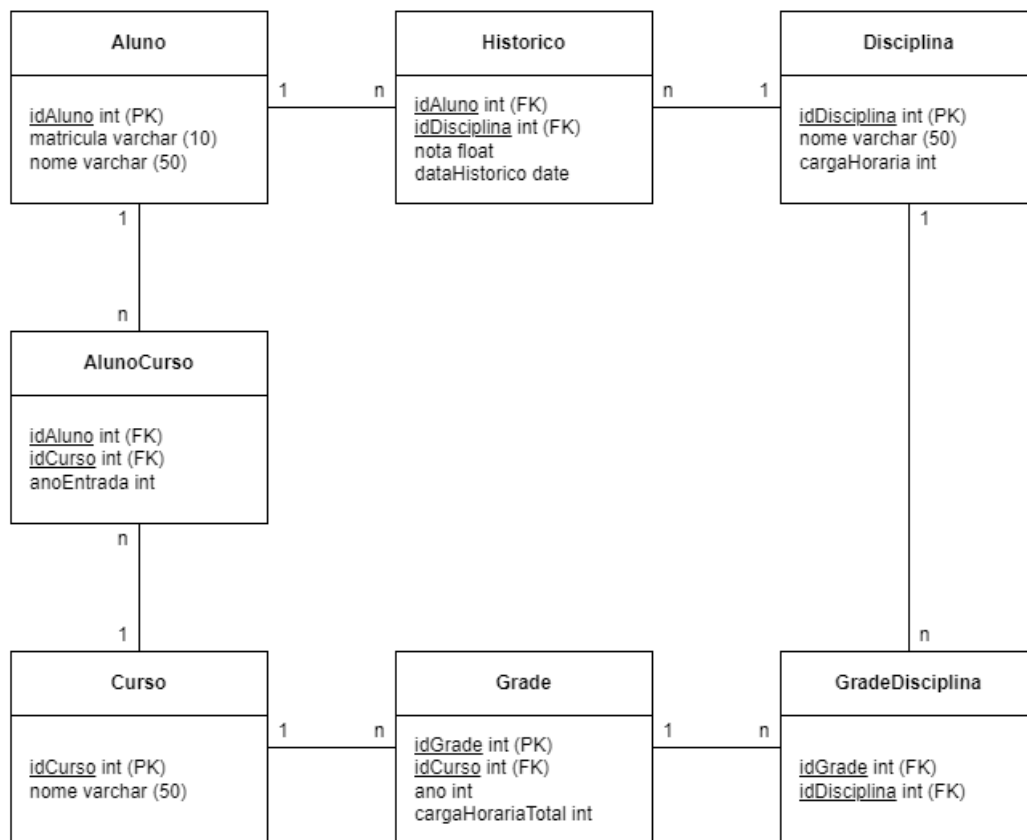
- Assento – Deverão ser armazenados os seguintes dados: identificação do assento e quantidade;
- Passageiro – Deverão ser armazenados os seguintes dados: CPF, nome, telefone, e-mail e endereço (rua, número, complemento, bairro, CEP, cidade e estado);
- Dependente – Deverão ser armazenados os seguintes 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.

Cole o Modelo Entidade-Relacionamento (MER) aqui.



2. 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.

1. 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*).

-- Isso cria o banco de dados

```
create database faculdade;
```

-- Usa o banco de dados faculdade

```
use faculdade;
```

-- criando as tabelas

-- cria tabela alunos e seus adornos

```
create table aluno (
```

```
    idAluno int not null auto_increment, matricula varchar(10) not null, nome varchar(50) not null, primary key (idAluno));
```

-- insere os dados

```
insert into Aluno (idAluno, matricula, nome) values
```

```
( 1, 'ADS001', 'Alice de Souza'),
( 2, 'BDS001', 'Ana Luiza de Paula'),
( 3, 'CDS001', 'Maria Helena Mantovani'),
( 4, 'DSM001', 'Marta da Silva'),
( 5, 'ENC001', 'Viviane Chaves Filha'),
( 6, 'ENS001', 'Paula Roberta Vitorino'),
( 7, 'GTI001', 'Miriam Miranda'),
( 8, 'JDS001', 'Beatriz Leopoldina'),
( 9, 'RCS001', 'Nicole Amanda de Jesus'),
(10, 'RCS002', 'Vitor Martins'),
(11, 'JDS002', 'João Augusto de Moura'),
(12, 'GTI002', 'Matheus Murilo de Souza'),
(13, 'ENS002', 'Mario Vicente'),
(14, 'ENC002', 'Antônio Cozer'),
(15, 'DSM002', 'Luciano Tucolo'),
```

```
(16, 'CDS002', 'Guilherme Koeriche'),  
(17, 'BDS002', 'Lucas Cochuelo'),  
(18, 'ADS002', 'Diogo Furlan'),  
(19, 'ADS003', 'Marcelo Luis dos Santos');
```

-- mostra a tabela aluno:

```
select * from aluno;
```

-- cria a tabela disciplina:

```
create table disciplina (  
    idDisciplina int not null auto_increment, nome varchar(50) not null, cargaHoraria int  
not null, primary key (idDisciplina));
```

-- insere os dados da tabela disciplina:

```
insert into Disciplina (idDisciplina, nome, cargaHoraria) values
```

```
( 1, 'Análise de Sistemas', 60),  
( 2, 'Arquitetura de Computadores', 60),  
( 3, 'Atividade Extensionista I', 40),  
( 4, 'Atividade Extensionista II', 40),  
( 5, 'Banco de Dados', 60),  
( 6, 'Empreendedorismo', 40),  
( 7, 'Engenharia de Software', 60),  
( 8, 'Fundamentos de Sistemas de Informação', 60),  
( 9, 'Gestão de Projetos de Software', 60),  
(10, 'Lógica de Programação e Algoritmos', 80),  
(11, 'Matemática Computacional', 40),  
(12, 'Programação de Computadores', 80),  
(13, 'Programação Orientada a Objetos', 80),  
(14, 'Sistema Gerenciador de Banco de Dados', 60),  
(15, 'Sistemas Operacionais', 60);
```

-- Mostra a tabela disciplina:

```
select * from disciplina;
```

-- mostra somente a parte de nomes das disciplinas na tabela disciplina:

```
select nome from Disciplina;
```

-- insere o nome de todos os cursos e seus respectivos alunos em ordem decrescente:

```
select curso.nome, aluno.nome from curso join aluno_curso on curso.idCurso =  
aluno_curso.idCurso join aluno on aluno_curso.idAluno = aluno.idAluno order by  
curso.nome desc;
```

-- implementa uma consulta para listar o nome das disciplinas e a média das notas das disciplinas em todos os cursos:

```
select disciplina.nome, avg(historico.nota) as media from historico join disciplina on  
historico.idDisciplina = disciplina.idDisciplina group by disciplina.nome;
```

-- cria a tabela curso:

```
create table curso (  
idCurso int not null auto_increment, nome varchar(50) not null, primary key (idCurso));
```

-- insere dados pra tabela curso:

```
insert into curso (idCurso, nome) values  
  (1, 'Análise e Desenvolvimento de Sistemas'),  
  (2, 'Banco de Dados'),  
  (3, 'Ciência de Dados'),  
  (4, 'Desenvolvimento Mobile'),  
  (5, 'Engenharia da Computação'),  
  (6, 'Engenharia de Software'),  
  (7, 'Gestão da Tecnologia da Informação'),  
  (8, 'Jogos Digitais'),  
  (9, 'Redes de Computadores');
```

-- Mostra a tabela curso:


```
select * from curso;
```

-- mostra o total de cursos:

```
select count(*) total_cursos from curso;
```

-- Implementa uma consulta para listar o nome de todos os cursos e a quantidade de alunos em cada curso:

```
select curso.nome, count(aluno_curso.idAluno) as quantidade from curso join  
aluno_curso on curso.idCurso = aluno_curso.idCurso group by curso.nome;
```

-- Cria a tabela historico:

```
create table historico (  
    idHistorico int not null auto_increment, idAluno int not null, idDisciplina int not null, nota  
float not null, dataHistorico date not null,  
    primary key (idHistorico), foreign key (idAluno) references aluno (idAluno) on delete  
cascade, foreign key (idDisciplina) references disciplina (idDisciplina) on delete cascade);
```

-- Anotação: on delete serve pra deletar todas os dados especifico selecionado d uma tabela pai.

-- inserção de dados da tabela historico:

```
insert into historico (idAluno, idDisciplina, nota, dataHistorico) values  
    ( 3, 1, 90, '2022-12-09'),  
    ( 3, 3, 75, '2022-12-09'),  
    ( 3, 5, 85, '2022-12-09'),  
    ( 9, 1, 80, '2022-12-16'),  
    ( 9, 9, 75, '2022-12-16'),  
    ( 9, 11, 70, '2022-12-16'),  
    (13, 12, 70, '2022-12-09'),  
    (13, 13, 70, '2022-12-09'),  
    (13, 14, 82, '2022-12-09'),  
    (15, 2, 76, '2022-12-16'),
```

```
(15, 4, 80, '2022-12-16'),
```

```
(15, 6, 89, '2022-12-16');
```

```
-- Mostra a tabela historico:
```

```
select * from historico;
```

```
-- Cria a tabela aluno curso:
```

```
create table aluno_curso (
```

```
    idAluno int not null, idCurso int not null, anoEntrada int not null, primary key (idAluno,  
idcurso), foreign key (idAluno) references aluno (idAluno) on delete cascade,  
    foreign key (idCurso) references curso (idCurso) on delete cascade);
```

```
-- insere os dados da tabela:
```

```
insert into aluno_curso (idAluno, idCurso, anoEntrada) values
```

```
    ( 1, 1, 2023),
```

```
    ( 2, 2, 2023),
```

```
    ( 3, 3, 2022),
```

```
    ( 4, 4, 2023),
```

```
    ( 5, 5, 2023),
```

```
    ( 6, 6, 2023),
```

```
    ( 7, 7, 2023),
```

```
    ( 8, 8, 2023),
```

```
    ( 9, 9, 2022),
```

```
    (10, 9, 2023),
```

```
    (11, 8, 2023),
```

```
    (12, 7, 2023),
```

```
    (13, 6, 2022),
```

```
    (14, 5, 2023),
```

```
    (15, 4, 2022),
```

```
    (16, 3, 2023),
```

```
    (17, 2, 2023),
```

```
    (18, 1, 2023),
```

(19, 1, 2023);

-- mostra a tabela curso:

```
select * from aluno_curso;
```

-- cria a tabela grade:

```
create table grade (
```

```
    idGrade int not null auto_increment, idCurso int not null, ano int not null,  
    cargaHorariaTotal int not null, primary key (idGrade), foreign key (idCurso) references  
    curso (idCurso) on delete cascade);
```

-- injeção de dados da tabela grade:

```
insert into grade (idGrade, idCurso, ano, cargaHorariaTotal) values
```

```
( 1, 1, 2021, 880),
```

```
( 2, 2, 2022, 880),
```

```
( 3, 3, 2022, 880),
```

```
( 4, 4, 2022, 880),
```

```
( 5, 5, 2019, 880),
```

```
( 6, 6, 2022, 880),
```

```
( 7, 7, 2022, 880),
```

```
( 8, 8, 2022, 880),
```

```
( 9, 9, 2019, 880),
```

```
(10, 1, 2023, 880),
```

```
(11, 5, 2023, 880),
```

```
(12, 9, 2023, 880);
```

-- mostra a tabela grade:

```
select * from grade;
```

-- cria a tabela grade disciplina:

```
create table grade_disciplina (
```

idGrade int not null, idDisciplina int not null, primary key (idGrade, idDisciplina), foreign key (idGrade) references grade (idGrade) on delete cascade,

foreign key (idDisciplina) references disciplina (idDisciplina) on delete cascade);

-- insercao de dados da tabela grade disciplina:

insert into grade_disciplina (idGrade, idDisciplina) values

(1, 1), (1, 2), (1, 3), (1, 4), (1, 5), (1, 6), (1, 7), (1, 8), (1, 9), (1, 10), (1, 11), (1, 12), (1, 13), (1, 14), (1, 15),
(2, 1), (2, 2), (2, 3), (2, 4), (2, 5), (2, 6), (2, 7), (2, 8), (2, 9), (2, 10), (2, 11), (2, 12), (2, 13), (2, 14), (2, 15),
(3, 1), (3, 2), (3, 3), (3, 4), (3, 5), (3, 6), (3, 7), (3, 8), (3, 9), (3, 10), (3, 11), (3, 12), (3, 13), (3, 14), (3, 15),
(4, 1), (4, 2), (4, 3), (4, 4), (4, 5), (4, 6), (4, 7), (4, 8), (4, 9), (4, 10), (4, 11), (4, 12), (4, 13), (4, 14), (4, 15),
(5, 1), (5, 2), (5, 3), (5, 4), (5, 5), (5, 6), (5, 7), (5, 8), (5, 9), (5, 10), (5, 11), (5, 12), (5, 13), (5, 14), (5, 15),
(6, 1), (6, 2), (6, 3), (6, 4), (6, 5), (6, 6), (6, 7), (6, 8), (6, 9), (6, 10), (6, 11), (6, 12), (6, 13), (6, 14), (6, 15),
(7, 1), (7, 2), (7, 3), (7, 4), (7, 5), (7, 6), (7, 7), (7, 8), (7, 9), (7, 10), (7, 11), (7, 12), (7, 13), (7, 14), (7, 15),
(8, 1), (8, 2), (8, 3), (8, 4), (8, 5), (8, 6), (8, 7), (8, 8), (8, 9), (8, 10), (8, 11), (8, 12), (8, 13), (8, 14), (8, 15),
(9, 1), (9, 2), (9, 3), (9, 4), (9, 5), (9, 6), (9, 7), (9, 8), (9, 9), (9, 10), (9, 11), (9, 12), (9, 13), (9, 14), (9, 15),
(10, 1), (10, 2), (10, 3), (10, 4), (10, 5), (10, 6), (10, 7), (10, 8), (10, 9), (10, 10), (10, 11), (10, 12), (10, 13), (10, 14), (10, 15),
(11, 1), (11, 2), (11, 3), (11, 4), (11, 5), (11, 6), (11, 7), (11, 8), (11, 9), (11, 10), (11, 11), (11, 12), (11, 13), (11, 14), (11, 15),
(12, 1), (12, 2), (12, 3), (12, 4), (12, 5), (12, 6), (12, 7), (12, 8), (12, 9), (12, 10), (12, 11), (12, 12), (12, 13), (12, 14), (12, 15);

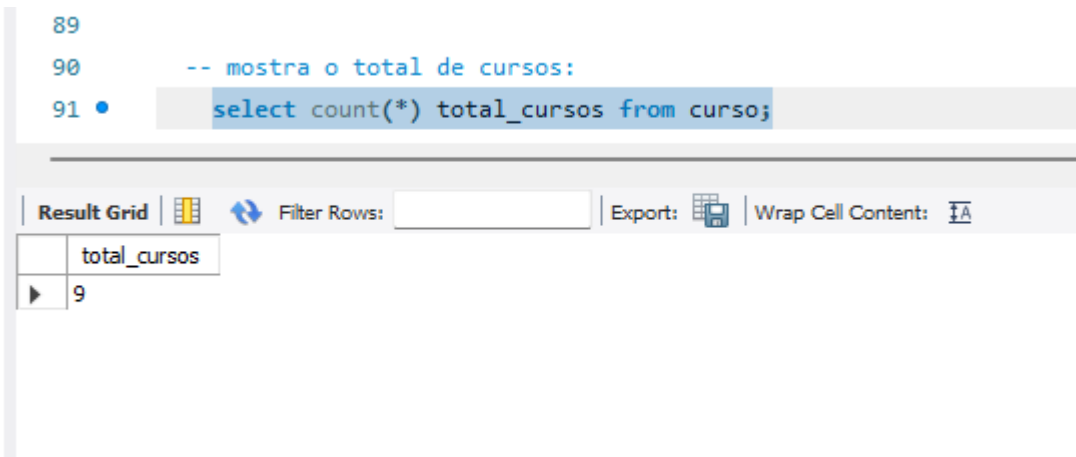
-- mostra a tabela grade disciplina:

```
select * from grade_disciplina;
```

Pontuação: 10 pontos.

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

```
89
90      -- mostra o total de cursos:
91 •    select count(*) total_cursos from curso;
```

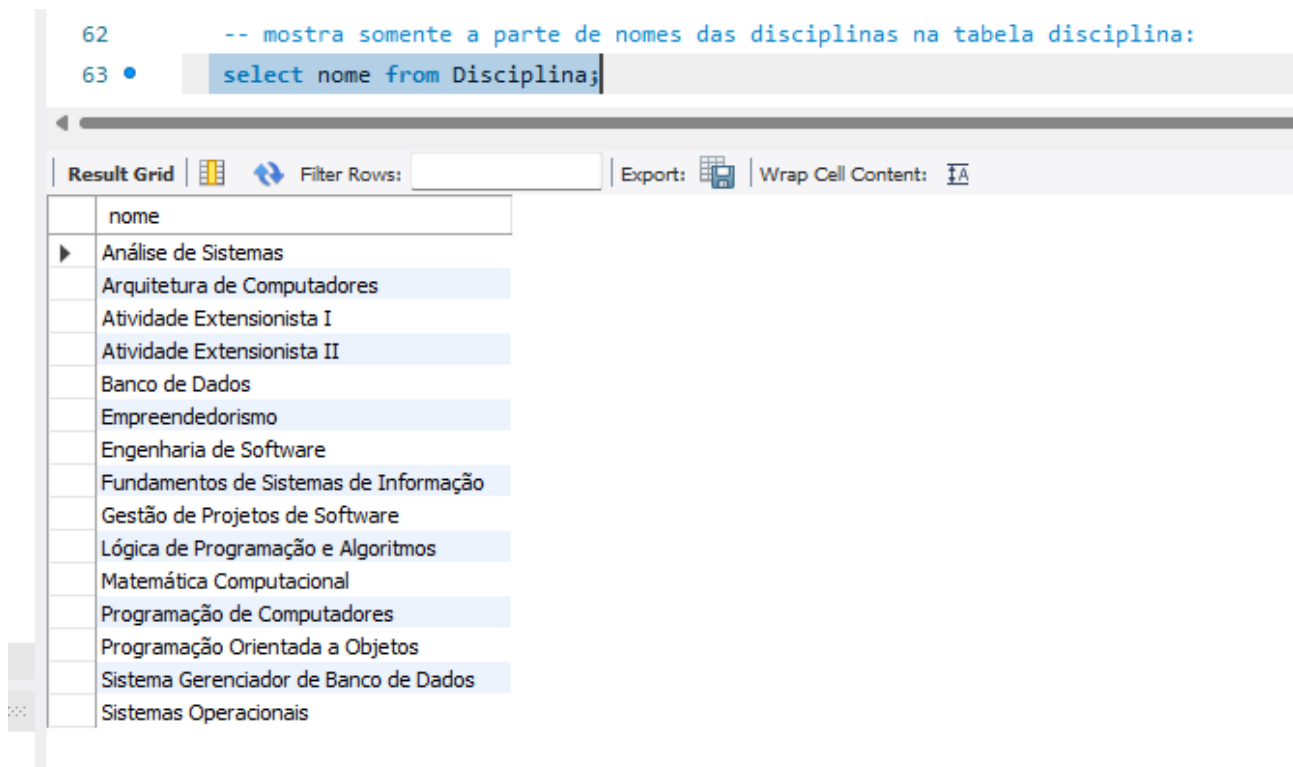


total_cursos
9

Pontuação: 10 pontos.

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

```
62      -- mostra somente a parte de nomes das disciplinas na tabela disciplina:
63 •    select nome from Disciplina;
```



nome
Análise de Sistemas
Arquitetura de Computadores
Atividade Extensionista I
Atividade Extensionista II
Banco de Dados
Empreendedorismo
Engenharia de Software
Fundamentos de Sistemas de Informação
Gestão de Projetos de Software
Lógica de Programação e Algoritmos
Matemática Computacional
Programação de Computadores
Programação Orientada a Objetos
Sistema Gerenciador de Banco de Dados
Sistemas Operacionais

Pontuação: 10 pontos.

4. 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.

```
65 -- insere o nome de todos os cursos e seus respectivos alunos em ordem decrescente:
66 • select curso.nome, aluno.nome from curso join aluno_curso on curso.idCurso = aluno_curso.idCurso join aluno on aluno_curso.idAluno = aluno.idAluno order by curso.nome desc;
```

nome	nome
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	Miriam Miranda
Gestão da Tecnologia da Informação	Matheus Murilo de Souza
Engenharia de Software	Paula Roberta Vitorino
Engenharia de Software	Mario Vicente
Engenharia da Computação	Viviane Chaves Filha
Engenharia da 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 Siste...	Alice de Souza
Análise e Desenvolvimento de Siste...	Diogo Furlan
Análise e Desenvolvimento de Siste...	Marcelo Luis dos Santos

Pontuação: 10 pontos.

5. 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*.

```
68 -- implementa uma consulta para listar o nome das disciplinas e a média das notas das disciplinas em todos os cursos:
69 • select disciplina.nome, avg(historico.nota) as media from historico join disciplina on historico.idDisciplina = disciplina.idDisciplina group by disciplina.nome;
```

nome	media
Análise de Sistemas	85
Atividade Extensionista I	75
Banco de Dados	85
Gestão de Projetos de Software	75
Matemática Computacional	70
Programação de Computadores	70
Programação Orientada a Objetos	70
Sistema Gerenciador de Banco de Dados	82
Arquitetura de Computadores	76
Atividade Extensionista II	80
Empreendedorismo	89

Pontuação: 10 pontos.

6. 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*.

```
93      -- Implementa uma consulta para listar o nome de todos os cursos e a quantidade de alunos em cada curso:  
94      select curso.nome, count(aluno_curso.idAluno) as quantidade from curso join aluno_curso on curso.idCurso = aluno_curso.idCurso group by curso.nome;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content: [IA](#)

	nome	quantidade
►	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