

## BANCO DE DADOS

### Trabalho – Relatório

<b>Curso:</b>	CST ANÁLISE E DESENVOLVIMENTO DE SISTEMAS - DISTÂNCIA
<b>Aluno(a):</b>	Matheus Belarmino Pignata
<b>RU:</b>	4525875

#### 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:

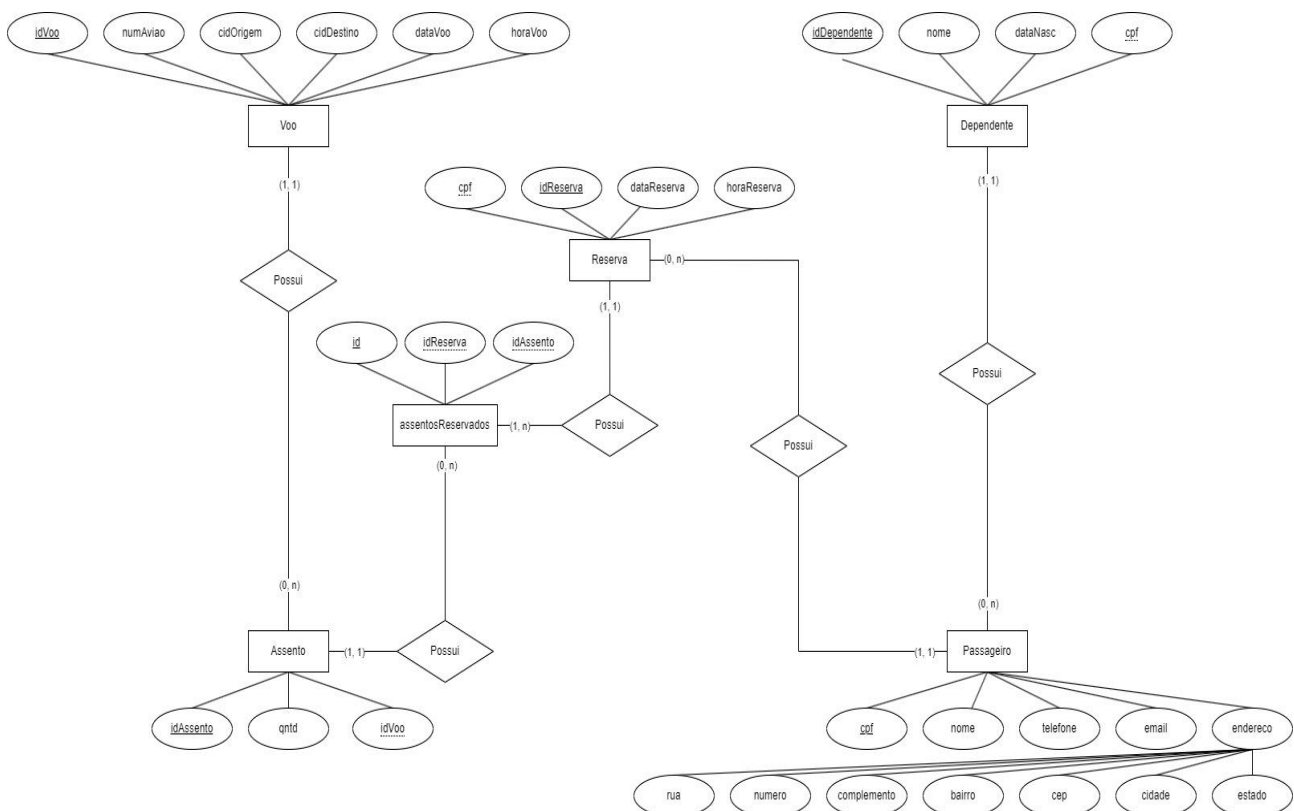
- Entidades;
- Atributos;
- Relacionamentos;
- Cardinalidades;
- Chaves primárias;
- Chaves estrangeiras.

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:

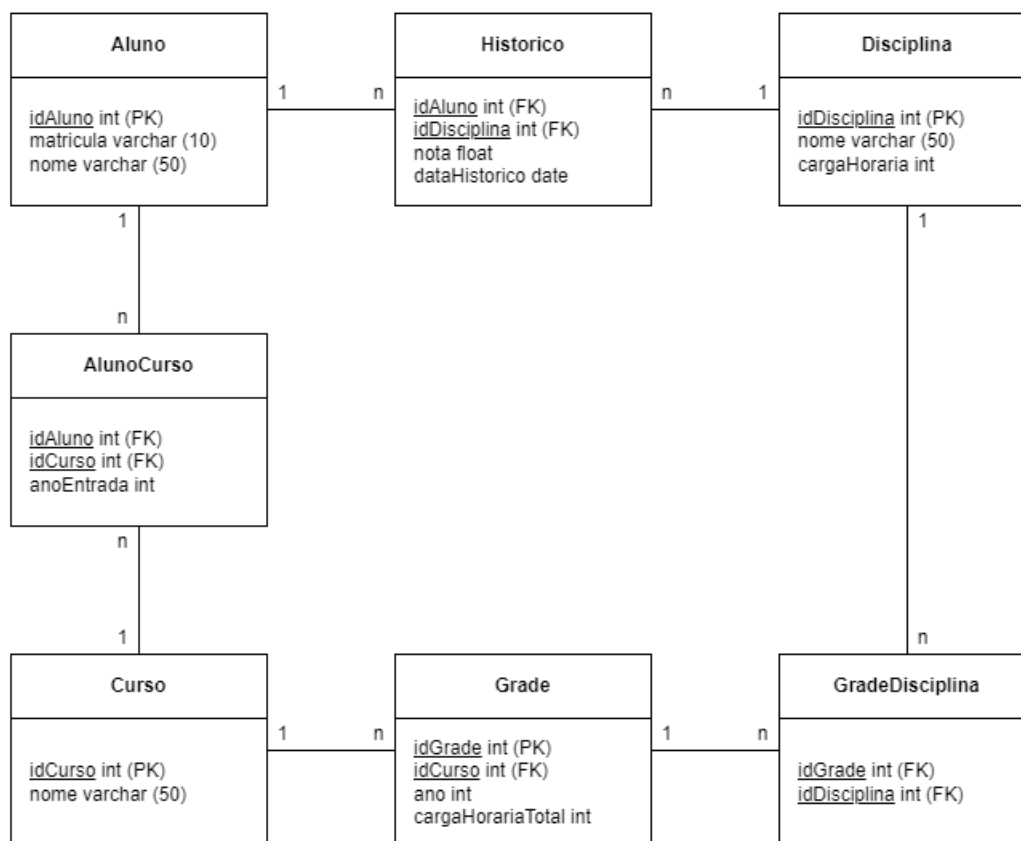
- Voo – Deverão ser armazenados os seguintes dados: identificação do voo, número do avião, cidade de origem, cidade de destino, data do voo e hora do voo;
- Assento – Deverão ser armazenados os seguintes dados: identificação do assento e quantidade;

- 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.



## 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*).

**CREATE DATABASE faculdade;**

**USE faculdade;**

**CREATE TABLE Aluno (  
idAluno INT PRIMARY KEY NOT NULL,  
matricula VARCHAR(10) NOT NULL,  
nome VARCHAR(50) NOT NULL  
);**

**CREATE TABLE Disciplina (  
idDisciplina INT PRIMARY KEY NOT NULL,  
nome VARCHAR(50) NOT NULL,  
cargaHoraria INT NOT NULL  
);**

**CREATE TABLE Curso (  
idCurso INT PRIMARY KEY NOT NULL,  
nome VARCHAR(50) NOT NULL  
);**

**CREATE TABLE Historico (  
idHistorico INT PRIMARY KEY AUTO\_INCREMENT NOT NULL,  
idAluno INT NOT NULL,  
idDisciplina INT NOT NULL,  
nota FLOAT NOT NULL,  
dataHistorico DATE NOT NULL,  
FOREIGN KEY (idAluno) REFERENCES Aluno(idAluno),**

**FOREIGN KEY (idDisciplina) REFERENCES Disciplina(idDisciplina)**  
**);**

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

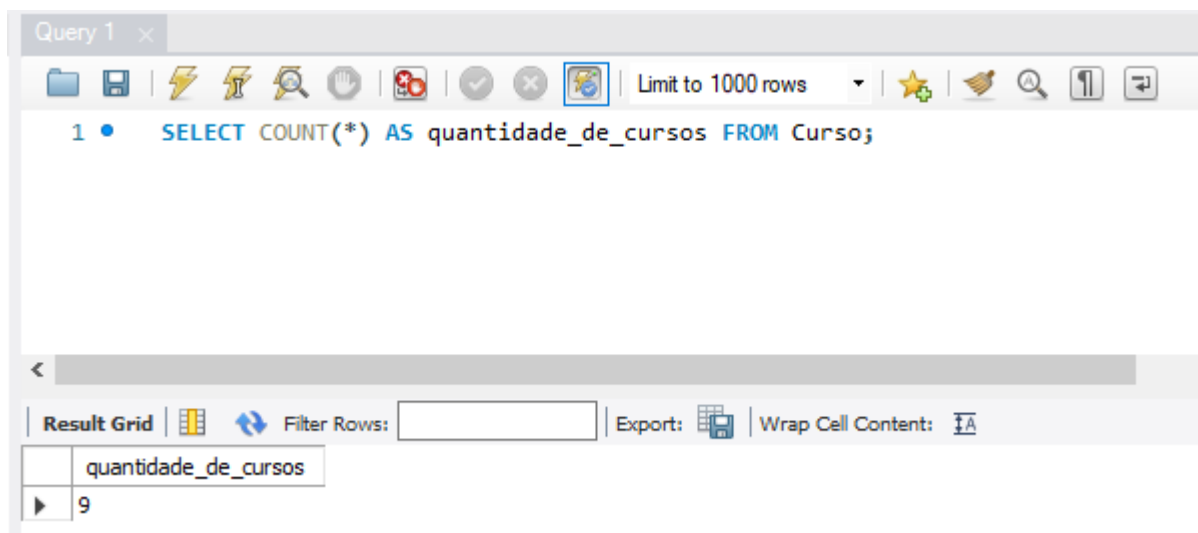
**CREATE TABLE Grade (**  
**idGrade INT PRIMARY KEY NOT NULL,**  
**idCurso INT NOT NULL,**  
**ano INT NOT NULL,**  
**cargaHorariaTotal INT NOT NULL,**  
**FOREIGN KEY (idCurso) REFERENCES Curso(idCurso)**  
**);**

**CREATE TABLE GradeDisciplina (**  
**idGradeDisciplina INT PRIMARY KEY AUTO\_INCREMENT NOT NULL,**  
**idGrade INT NOT NULL,**  
**idDisciplina INT NOT NULL,**  
**FOREIGN KEY (idGrade) REFERENCES Grade(idGrade),**  
**FOREIGN KEY (idDisciplina) REFERENCES Disciplina(idDisciplina)**  
**);**

**Pontuação:** 10 pontos.

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

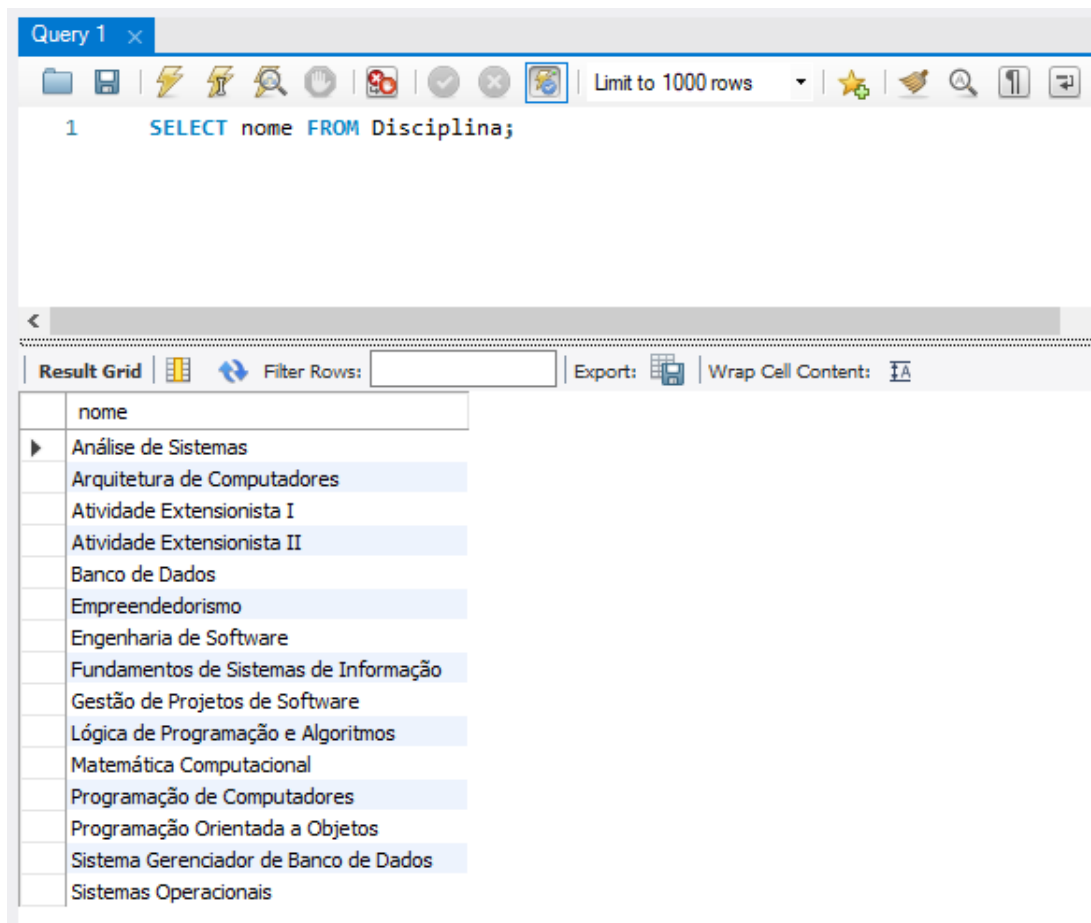
**SELECT COUNT(\*) AS quantidade\_de\_cursos FROM Curso;**



**Pontuação:** 10 pontos.

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

**SELECT nome FROM Disciplina;**



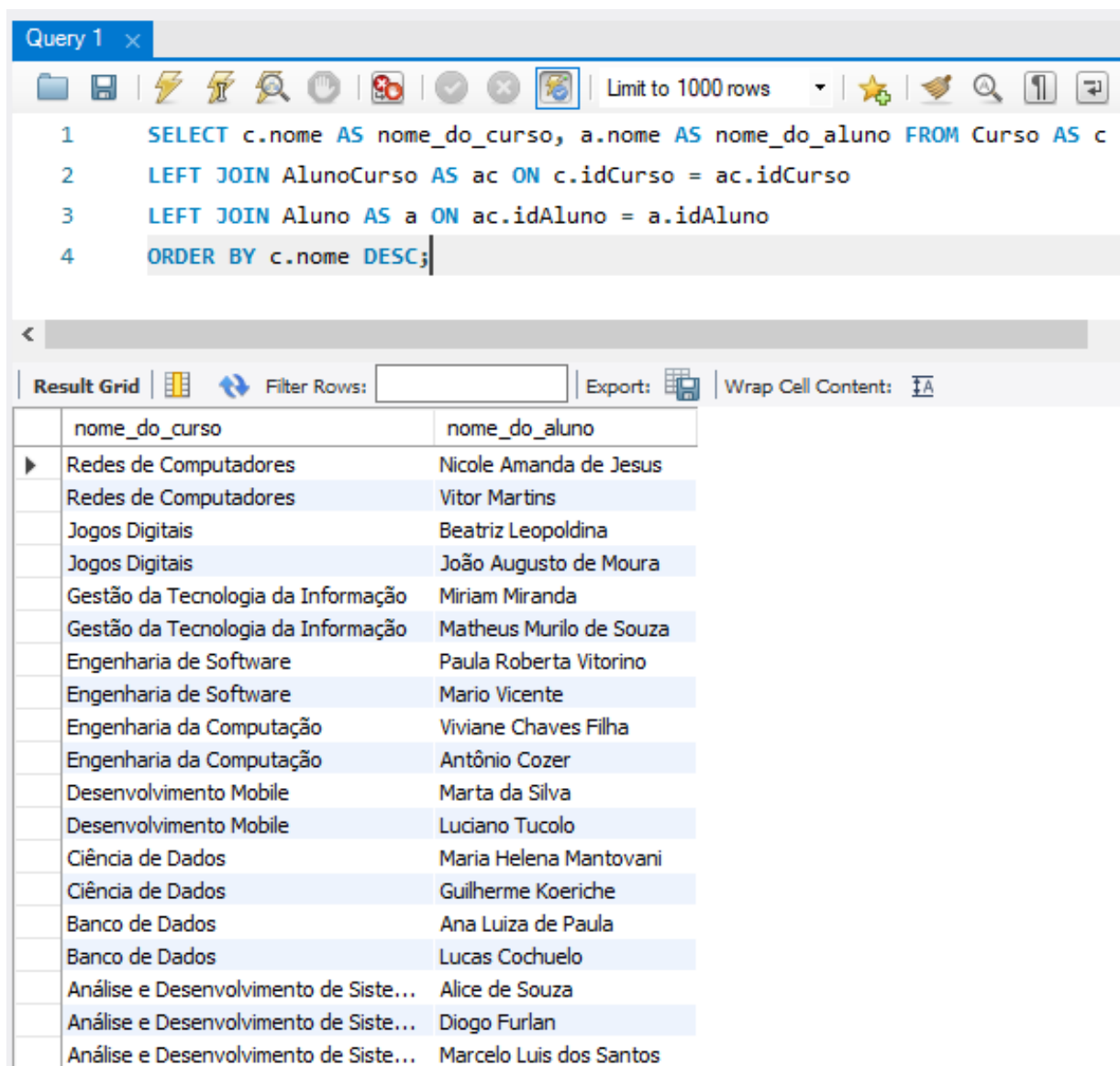
The screenshot shows a database query tool interface. At the top, there is a toolbar with various icons for file operations, execution, and settings. Below the toolbar, the query editor displays the SQL statement: `1 SELECT nome FROM Disciplina;`. The results are shown in a table below the query editor. The table has a single column named 'nome' and lists 18 different disciplines. The interface also includes a 'Filter Rows' section and an 'Export' button.

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.

```
SELECT c.nome AS nome_do_curso, a.nome AS nome_do_aluno
FROM Curso AS c
LEFT JOIN AlunoCurso AS ac ON c.idCurso = ac.idCurso
LEFT JOIN Aluno AS a ON ac.idAluno = a.idAluno
ORDER BY c.nome DESC;
```



Query 1

Limit to 1000 rows

```
1 SELECT c.nome AS nome_do_curso, a.nome AS nome_do_aluno FROM Curso AS c
2 LEFT JOIN AlunoCurso AS ac ON c.idCurso = ac.idCurso
3 LEFT JOIN Aluno AS a ON ac.idAluno = a.idAluno
4 ORDER BY c.nome DESC;
```

Result Grid

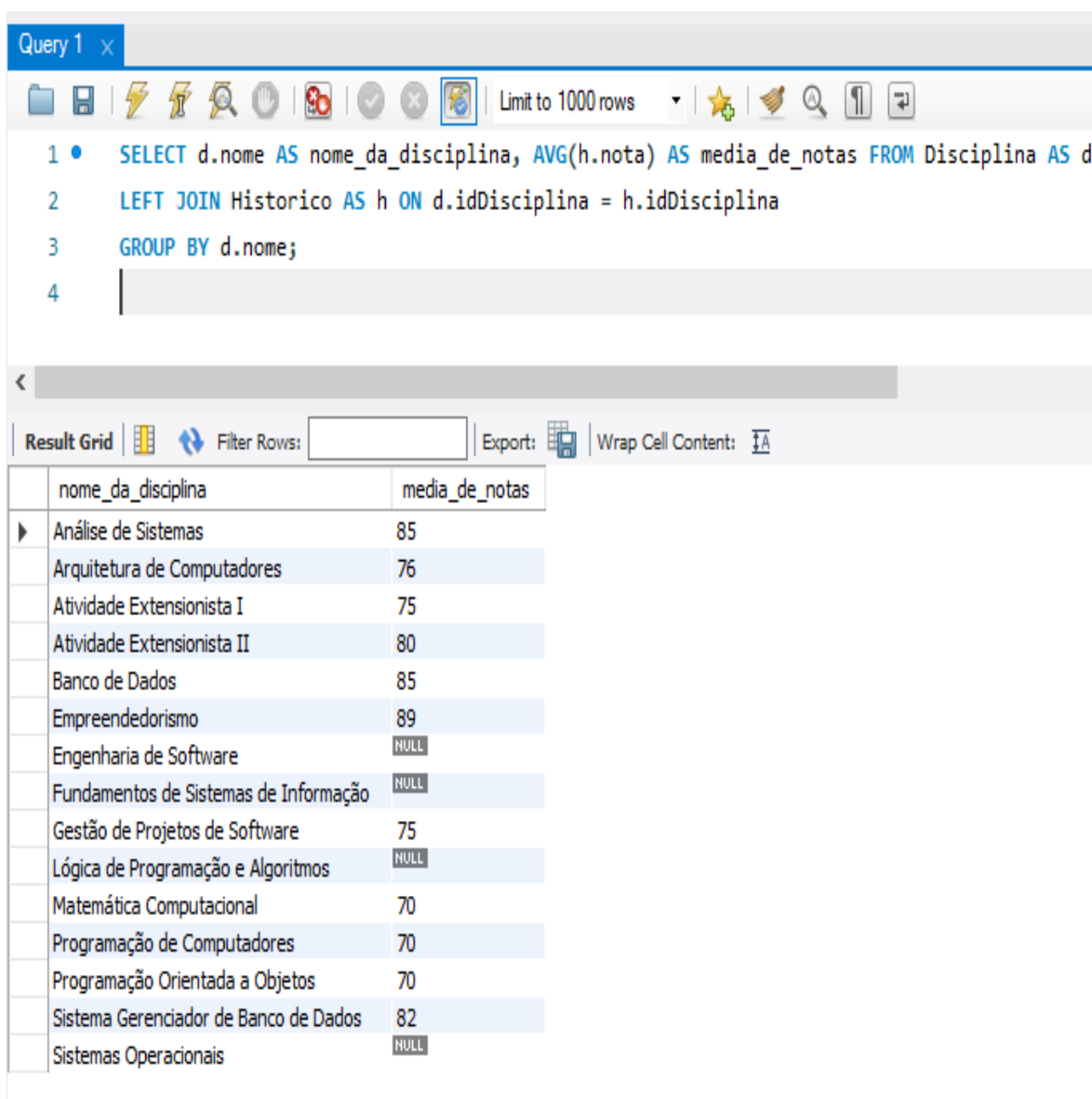
	nome_do_curso	nome_do_aluno
▶	Redes de Computadores	Nicole Amanda de Jesus
	Redes de Computadores	Vitor Martins
	Jogos Digitais	Beatriz Leopoldina
	Jogos Digitais	João Augusto de Moura
	Gestão da Tecnologia da Informação	Miriam Miranda
	Gestão da Tecnologia da Informação	Matheus Murilo de Souza
	Engenharia 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 Koeriche
	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*.

```
SELECT d.nome AS nome_da_disciplina, AVG(h.nota) AS media_de_notas
FROM Disciplina AS d
LEFT JOIN Historico AS h ON d.idDisciplina = h.idDisciplina
GROUP BY d.nome;
```



Query 1 x

Limit to 1000 rows

```
1 • SELECT d.nome AS nome_da_disciplina, AVG(h.nota) AS media_de_notas FROM Disciplina AS d
2 LEFT JOIN Historico AS h ON d.idDisciplina = h.idDisciplina
3 GROUP BY d.nome;
4
```

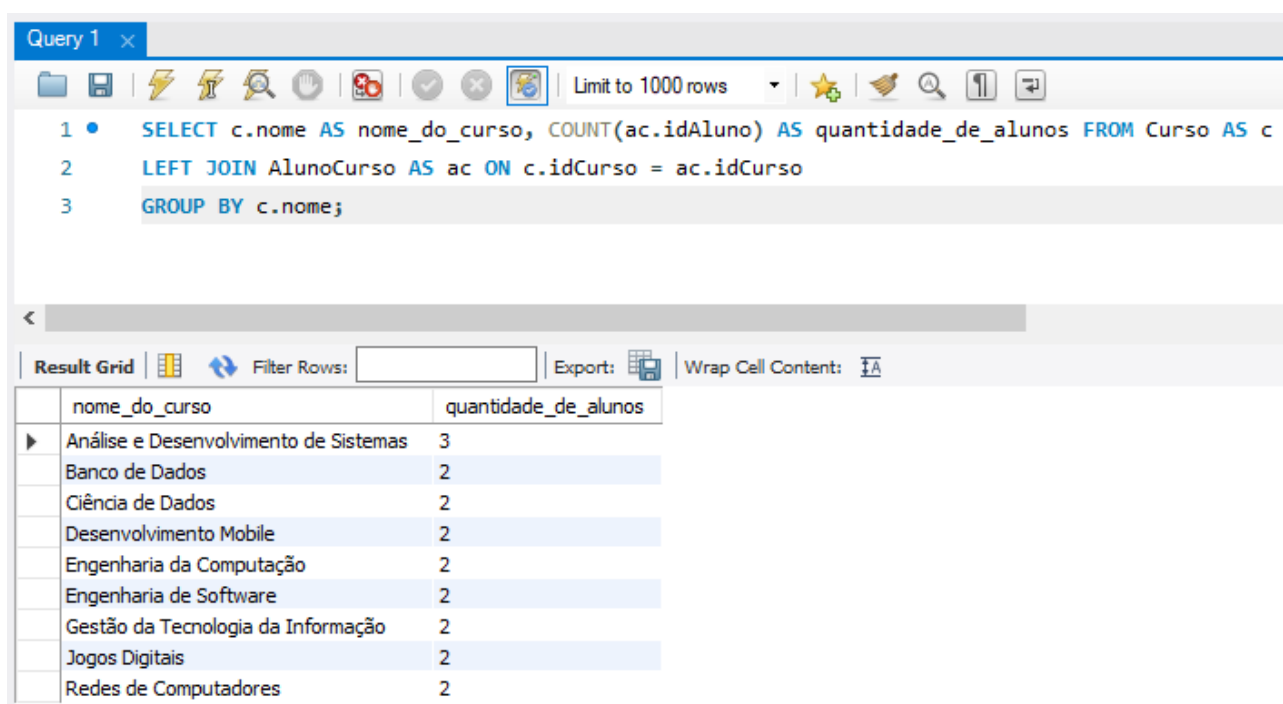
Result Grid

nome_da_disciplina	media_de_notas
Análise de Sistemas	85
Arquitetura de Computadores	76
Atividade Extensionista I	75
Atividade Extensionista II	80
Banco de Dados	85
Empreendedorismo	89
Engenharia de Software	NULL
Fundamentos de Sistemas de Informação	NULL
Gestão de Projetos de Software	75
Lógica de Programação e Algoritmos	NULL
Matemática Computacional	70
Programação de Computadores	70
Programação Orientada a Objetos	70
Sistema Gerenciador de Banco de Dados	82
Sistemas Operacionais	NULL

**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*.

```
SELECT    c.nome    AS    nome_do_curso,    COUNT(ac.idAluno)    AS  
quantidade_de_alunos FROM Curso AS c  
LEFT JOIN AlunoCurso AS ac ON c.idCurso = ac.idCurso  
GROUP BY c.nome;
```



The screenshot shows a database query editor with a toolbar and a query window. The query is as follows:

```
1 • SELECT c.nome AS nome_do_curso, COUNT(ac.idAluno) AS quantidade_de_alunos FROM Curso AS c  
2 LEFT JOIN AlunoCurso AS ac ON c.idCurso = ac.idCurso  
3 GROUP BY c.nome;
```

Below the query window, the results are displayed in a grid. The grid has two columns: **nome\_do\_curso** and **quantidade\_de\_alunos**. The results are as follows:

nome_do_curso	quantidade_de_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