PONTIFÍCIA UNIVERSIDADE CATÓLICA DO PARANÁ

PROJETO DE BANCO DE DADOS

2ª PARTE

UniVerse

Equipe:

Adriano Vale

Ary Farah

Caroline Assis

Ícaro Kuchanovicz

Curitiba

Novembro de 2023

Sumário

1 Domínio de Aplicação para o Banco de Dados

1.1 Identificação do Projeto

UniVerse

1.2 Tema do Projeto

Portal de alunos, onde ele poderá acessar suas notas, frequência, matérias, atividades complementares, com o diferencial de um chat de interação entre todos os integrantes da faculdade (funcionários, alunos, professores), um sistema de caronas e de estágio.

1.3 Usuários do sistema

Professores: Disponibilizar notas e conteúdos.

Alunos: Acessar seus dados e informações gerais, participar do programa de caronas, buscar estágio.

Funcionários: Ver informações gerais e eventos disponíveis.

Empresas: Procurar alunos para estagiar.

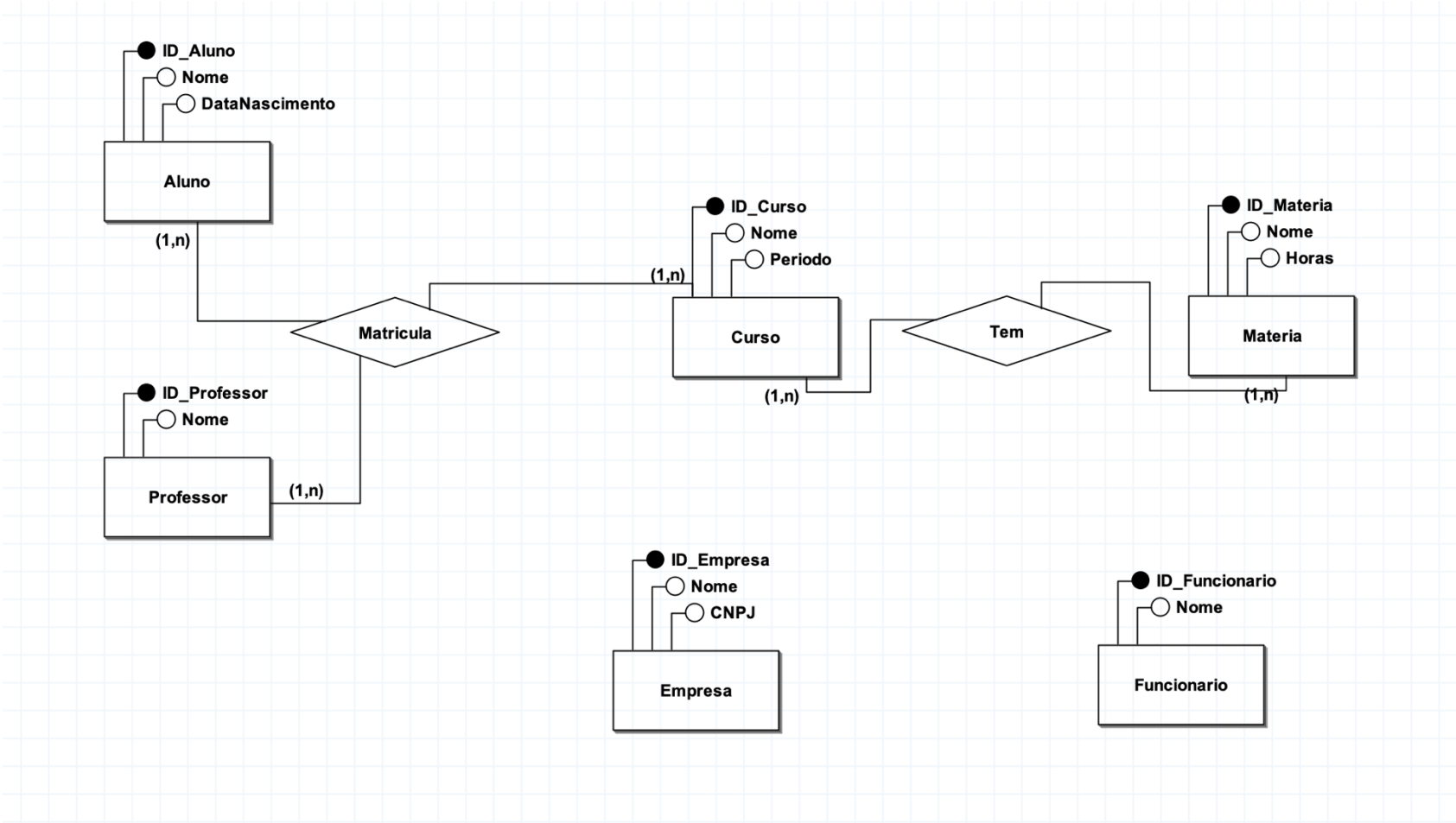
1.4 Funcionalidade 1 do Projeto

Realizar login

1.5 Funcionalidade 2 do Projeto

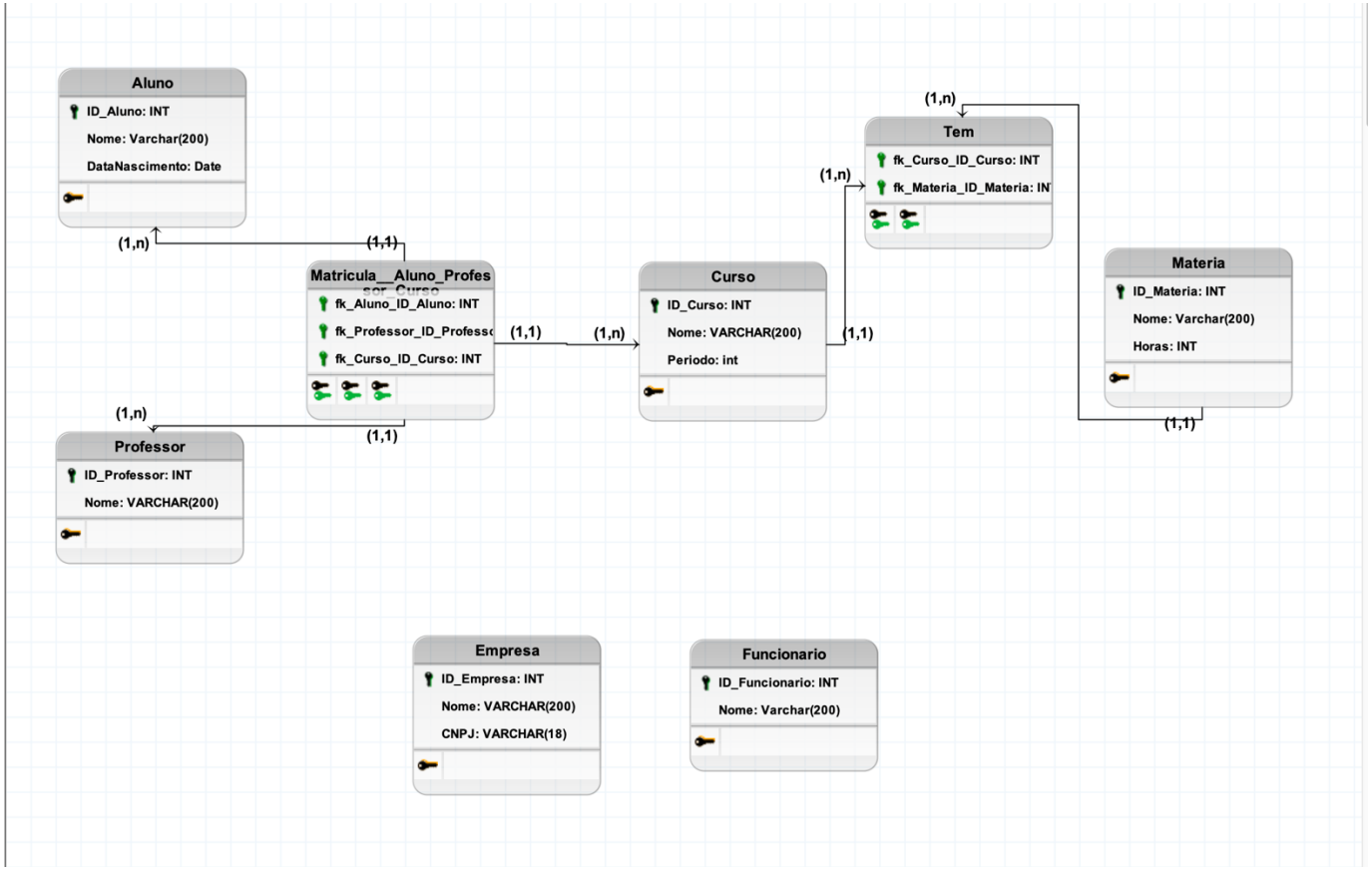
2 Modelo Conceitual

Imagem de boa resolução do Modelo Conceitual (**Modelo Entidade-Relacionamento - MER**) na **3FN**


ID_Aluno
Nome
DataNascimento
Aluno
(1,n)
Matricula
(1,n)
ID_Curso
Nome
Periodo
Curso
(1,n)
Tem
ID_Materia
Nome
Horas
Materia
(1,n)
ID_Professor
Nome
Professor
(1,n)
ID_Empresa
Nome
CNPJ
Empresa
ID_Funcionario
Nome
Funcionario

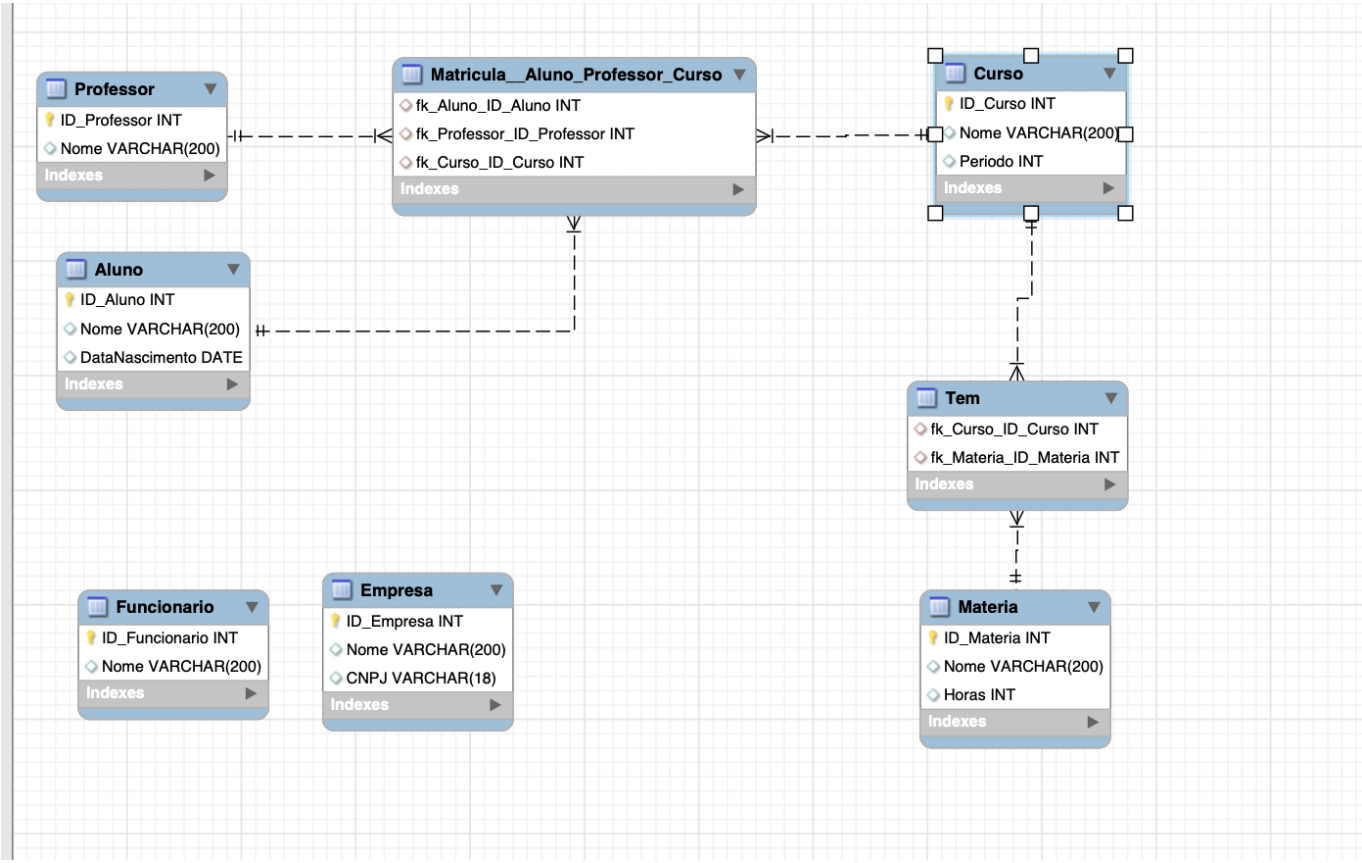
3 Modelo Lógico (3FN)

Imagem de boa resolução do Modelo Lógico (**Modelo Relacional - MR**) na **3FN**, feito com o **brModelo**.



4 Modelo Lógico (3FN)

Imagem de boa resolução do Modelo Lógico (**Modelo Relacional - MR**) na **3FN**, feito com o **ENGENHARIA REVERSA do MySQL Workbench**.


Professor
ID_Professor INT
Nome VARCHAR(200)
Indexes
Matricula__Aluno_Professor_Curso
fk_Aluno_ID_Aluno INT
fk_Professor_ID_Professor INT
fk_Curso_ID_Curso INT
Indexes
Curso
ID_Curso INT
Nome VARCHAR(200)
Periodo INT
Indexes
Aluno
ID_Aluno INT
Nome VARCHAR(200)
DataNascimento DATE
Indexes
Tem
fk_Curso_ID_Curso INT
fk_Materia_ID_Materia INT
Indexes
Funcionario
ID_Funcionario INT
Nome VARCHAR(200)
Indexes
Empresa
ID_Empresa INT
Nome VARCHAR(200)
CNPJ VARCHAR(18)
Indexes
Materia
ID_Materia INT
Nome VARCHAR(200)
Horas INT
Indexes

5 Modelo Físico (3FN)

5.1 SQL para criação de tabelas e restrições

```sql
CREATE TABLE Aluno (
    ID_Aluno INT PRIMARY KEY,
    Nome VARCHAR(200),
    DataNascimento DATE
);

CREATE TABLE Professor (
    ID_Professor INT PRIMARY KEY,
    Nome VARCHAR(200)
);

CREATE TABLE Curso (
    ID_Curso INT PRIMARY KEY,
    Nome VARCHAR(200),
    Periodo INT
);

CREATE TABLE Materia (
    ID_Materia INT PRIMARY KEY,
    Nome VARCHAR(200),
    Horas INT
);

CREATE TABLE Empresa (
    ID_Empresa INT PRIMARY KEY,
    Nome VARCHAR(200),
    CNPJ VARCHAR(18)
);

CREATE TABLE Funcionario (
    ID_Funcionario INT PRIMARY KEY,
    Nome VARCHAR(200)
);

CREATE TABLE Matricula_Aluno_Professor_Curso (
    fk_Aluno_ID INT,
    fk_Professor_ID INT,
    fk_Curso_ID INT,
    PRIMARY KEY (fk_Aluno_ID, fk_Professor_ID, fk_Curso_ID),
    FOREIGN KEY (fk_Aluno_ID) REFERENCES Aluno(ID_Aluno),
    FOREIGN KEY (fk_Professor_ID) REFERENCES Professor(ID_Professor),
    FOREIGN KEY (fk_Curso_ID) REFERENCES Curso(ID_Curso)
);
```

5.2 SQL para inserção de pelo menos 15 registros para cada tabela

Importante: Indicar em imagens de boa resolução, o resultado de cada comando de INSERT.

```sql
INSERT INTO Aluno (ID_Aluno, Nome, DataNascimento) VALUES
(1, 'Alice Silva', '2003-02-10'),
(2, 'Bruno Gomes', '2003-04-15'),
(3, 'Carla Souza', '2003-06-20'),
(4, 'Daniel Rocha', '2003-08-25'),
(5, 'Eduardo Costa', '2003-10-30'),
(6, 'Fernanda Dias', '2003-12-05'),
(7, 'Gabriel Martins', '2004-01-10'),
(8, 'Helena Santos', '2004-03-15'),
(9, 'Ígor Oliveira', '2004-05-20'),
(10, 'Joana Almeida', '2004-07-25'),
(11, 'Kaique Barros', '2004-09-30'),
(12, 'Larissa Teixeira', '2004-11-05'),
(13, 'Mário Andrade', '2004-12-10'),
(14, 'Natália Lima', '2005-01-15'),
(15, 'Otávio Pereira', '2005-03-20');
```

ID_Aluno	Nome	DataNascimento
1	Alice Silva	2003-02-10
2	Bruno Gomes	2003-04-15
3	Carla Souza	2003-06-20
4	Daniel Rocha	2003-08-25
5	Eduardo Costa	2003-10-30
6	Fernanda Dias	2003-12-05
7	Gabriel Martins	2004-01-10
8	Helena Santos	2004-03-15
9	Ígor Oliveira	2004-05-20
10	Joana Almeida	2004-07-25
11	Kaique Barros	2004-09-30
12	Larissa Teixeira	2004-11-05
13	Mário Andrade	2004-12-10
14	Natália Lima	2005-01-15
15	Otávio Pereira	2005-03-20

```sql
INSERT INTO Professor (ID_Professor, Nome) VALUES
(1, 'Prof. Ana Beatriz'),
(2, 'Prof. Bruno César'),
(3, 'Prof. Carla Dantas'),
(4, 'Prof. Daniel Esteves'),
(5, 'Prof. Elisa Figueiredo'),
(6, 'Prof. Fernando Gomes'),
(7, 'Prof. Gabriela Horta'),
(8, 'Prof. Henrique Ito'),
(9, 'Prof. Isabela Jardim'),
(10, 'Prof. João Kleber'),
(11, 'Prof. Larissa Moraes'),
(12, 'Prof. Márcio Nogueira'),
(13, 'Prof. Nívea Oliveira'),
(14, 'Prof. Oscar Pires'),
(15, 'Prof. Patrícia Queiroz');
```

ID_Professor	Nome
1	Prof. Ana Beatriz
2	Prof. Bruno César
3	Prof. Carla Dantas
4	Prof. Daniel Esteves
5	Prof. Elisa Figueiredo
6	Prof. Fernando Gomes
7	Prof. Gabriela Horta
8	Prof. Henrique Ito
9	Prof. Isabela Jardim
10	Prof. João Kleber
11	Prof. Larissa Moraes
12	Prof. Márcio Nogueira
13	Prof. Nívea Oliveira
14	Prof. Oscar Pires
15	Prof. Patrícia Queiroz

```sql
INSERT INTO Empresa (ID_Empresa, Nome, CNPJ) VALUES
(1, 'Empresa A', '12.345.678/0001-91'),
(2, 'Empresa B', '23.456.789/0001-82'),
(3, 'Empresa C', '34.567.890/0001-73'),
(4, 'Empresa D', '45.678.901/0001-64'),
(5, 'Empresa E', '56.789.012/0001-55'),
(6, 'Empresa F', '67.890.123/0001-46'),
(7, 'Empresa G', '78.901.234/0001-37'),
(8, 'Empresa H', '89.012.345/0001-28'),
(9, 'Empresa I', '90.123.456/0001-19'),
(10, 'Empresa J', '01.234.567/0001-01'),
(11, 'Empresa K', '11.235.678/0002-92'),
(12, 'Empresa L', '21.346.789/0002-83'),
(13, 'Empresa M', '31.457.890/0002-74'),
(14, 'Empresa N', '41.568.901/0002-65'),
(15, 'Empresa O', '51.679.012/0002-56');
```

ID_Empresa	Nome	CNPJ
1	Empresa A	12.345.678/0001-91
2	Empresa B	23.456.789/0001-82
3	Empresa C	34.567.890/0001-73
4	Empresa D	45.678.901/0001-64
5	Empresa E	56.789.012/0001-55
6	Empresa F	67.890.123/0001-46
7	Empresa G	78.901.234/0001-37
8	Empresa H	89.012.345/0001-28
9	Empresa I	90.123.456/0001-19
10	Empresa J	01.234.567/0001-01
11	Empresa K	11.235.678/0002-92
12	Empresa L	21.346.789/0002-83
13	Empresa M	31.457.890/0002-74
14	Empresa N	41.568.901/0002-65
15	Empresa O	51.679.012/0002-56

```sql
INSERT INTO Funcionario (ID_Funcionario, Nome) VALUES
(1, 'Lucas Marques'),
(2, 'Mariana Ribeiro'),
(3, 'Rafael Almeida'),
(4, 'Juliana Campos'),
(5, 'Paulo Soares'),
(6, 'Renata Martins'),
(7, 'Thiago Fernandes'),
(8, 'Patricia Lima'),
(9, 'Eduardo Ramos'),
(10, 'Sofia Gonçalves'),
(11, 'Felipe Araújo'),
(12, 'Isabelly Costa'),
(13, 'Mateus Carvalho'),
(14, 'Giovanna Rodrigues'),
(15, 'Gustavo Barbosa');
```

ID_Funcionario	Nome
1	Lucas Marques
2	Mariana Ribeiro
3	Rafael Almeida
4	Juliana Campos
5	Paulo Soares
6	Renata Martins
7	Thiago Fernandes
8	Patricia Lima
9	Eduardo Ramos
10	Sofia Gonçalves
11	Felipe Araújo
12	Isabelly Costa
13	Mateus Carvalho
14	Giovanna Rodrig…
15	Gustavo Barbosa

```sql
INSERT INTO Materia (ID_Materia, Nome, Horas) VALUES
(1, 'Matemática', 60),
(2, 'Português', 60),
(3, 'História', 50),
(4, 'Geografia', 50),
(5, 'Biologia', 55),
(6, 'Química', 55),
(7, 'Física', 55),
(8, 'Inglês', 40),
(9, 'Espanhol', 40),
(10, 'Artes', 30),
(11, 'Educação Física', 30),
(12, 'Filosofia', 45),
(13, 'Sociologia', 45),
(14, 'Informática', 60),
(15, 'Economia', 50);
```

ID_Materia	Nome	Horas
1	Matemática	60
2	Português	60
3	História	50
4	Geografia	50
5	Biologia	55
6	Química	55
7	Física	55
8	Inglês	40
9	Espanhol	40
10	Artes	30
11	Educação…	30
12	Filosofia	45
13	Sociologia	45
14	Informática	60
15	Economia	50

```sql
INSERT INTO Curso (ID_Curso, Nome, Periodo) VALUES
(1, 'Engenharia Civil', 10),
(2, 'Medicina', 12),
(3, 'Direito', 10),
(4, 'Administração', 8),
(5, 'Ciência da Computação', 8),
(6, 'Psicologia', 10),
(7, 'Arquitetura', 10),
(8, 'Engenharia Mecânica', 10),
(9, 'Enfermagem', 10),
(10, 'Educação Física', 8),
(11, 'Contabilidade', 8),
(12, 'Marketing', 8),
(13, 'Odontologia', 10),
(14, 'Engenharia de Produção', 10),
(15, 'Farmácia', 10);
```

ID_Curso	Nome	Periodo
1	Engenharia Civil	10
2	Medicina	12
3	Direito	10
4	Administração	8
5	Ciência da Computação	8
6	Psicologia	10
7	Arquitetura	10
8	Engenharia Mecânica	10
9	Enfermagem	10
10	Educação Física	8
11	Contabilidade	8
12	Marketing	8
13	Odontologia	10
14	Engenharia de Produção	10
15	Farmácia	10

```sql
INSERT INTO Matricula_Aluno_Professor_Curso (fk_Aluno_ID, fk_Professor_ID, fk_Curso_ID) VALUES
(1, 1, 1),
(2, 2, 1),
(3, 3, 1),
(4, 4, 1),
(5, 5, 2),
(6, 6, 2),
(7, 7, 2),
(8, 8, 2),
(9, 9, 3),
(10, 10, 3),
(11, 11, 3),
(12, 12, 3),
(13, 13, 4),
(14, 14, 4),
(15, 15, 4);
```

fk_Aluno_ID	fk_Professor_ID	fk_Curso_ID
1	1	1
2	2	1
3	3	1
4	4	1
5	5	2
6	6	2
7	7	2
8	8	2
9	9	3
10	10	3
11	11	3
12	12	3
13	13	4
14	14	4
15	15	4

6 Consultas e Programação

Para consultas com referências entre tabelas, usar apenas JOIN.

6.1 SQL para 3 consultas com AGREGAÇÃO de recuperação de dados

Consulta 1: Número médio de alunos por curso

a) Descreva por qual razão a consulta é importante para o usuário.
Para a instituição administrar e entender a média de cada curso, auxiliando em planejamentos futuros, como abrir novas turmar

b) Código SQL

```sql
SELECT c.Nome AS Curso,
       CAST(AVG(AlunoCount) AS DECIMAL(10,2)) AS Média_Alunos
FROM Curso c
LEFT JOIN (
    SELECT fk_Curso_ID, COUNT(DISTINCT fk_Aluno_ID) AS AlunoCount
    FROM Matricula_Aluno_Professor_Curso
    GROUP BY fk_Curso_ID
) AS CursoAlunos ON CursoAlunos.fk_Curso_ID = c.ID_Curso
GROUP BY c.Nome;
```

c) Resultado da consulta.

Curso	Média_Alunos
Engenharia Civil	4.00
Medicina	4.00
Direito	4.00
Administração	3.00
Ciência da Computação	NULL
Psicologia	NULL
Arquitetura	NULL
Engenharia Mecânica	NULL
Enfermagem	NULL
Educação Física	NULL
Contabilidade	NULL
Marketing	NULL
Odontologia	NULL
Engenharia de Produção	NULL
Farmácia	NULL

Consulta 2: Total de horas de cada professor

a) Descreva por qual razão a consulta é importante para o usuário.

A instituição acompanha a carga de trabalho de cada professor

b) Código SQL.

```sql
SELECT c.Nome AS Curso, c.Periodo, COUNT(t.fk_Materia_ID) AS Quantidade_Materias
FROM Curso c
JOIN Tem t ON c.ID_Curso = t.fk_Curso_ID
JOIN Materia m ON t.fk_Materia_ID = m.ID_Materia
GROUP BY c.Nome, c.Periodo;
```

c) Resultado da consulta.

Curso	Periodo	Quantidade_Materi...
Engenharia Civil	10	1
Medicina	12	1
Direito	10	1
Administração	8	1
Ciência da Computação	8	1
Psicologia	10	1
Arquitetura	10	1
Engenharia Mecânica	10	1
Enfermagem	10	1
Educação Física	8	1
Contabilidade	8	1
Marketing	8	1
Odontologia	10	1
Engenharia de Produção	10	1
Farmácia	10	1

Consulta 3: Quantidades de matérias por período de um curso

a) Descreva por qual razão a consulta é importante para o usuário.
Fornece informações sobre a distribuição de disciplinas ao longo dos períodos de cada curso

b) Código SQL.

```sql
SELECT p.Nome AS Professor, SUM(m.Horas) AS Total_Horas_Lecionadas
FROM Professor p
JOIN Matricula_Aluno_Professor_Curso mapc ON p.ID_Professor = mapc.fk_Professor_ID
JOIN Tem t ON mapc.fk_Curso_ID = t.fk_Curso_ID
JOIN Materia m ON t.fk_Materia_ID = m.ID_Materia
GROUP BY p.Nome;
```

c) Resultado da consulta.

Professor	Total_Horas_Lecionad...
Prof. Ana Beatriz	60
Prof. Bruno César	60
Prof. Carla Dantas	60
Prof. Daniel Esteves	60
Prof. Elisa Figueiredo	60
Prof. Fernando Gomes	60
Prof. Gabriela Horta	60
Prof. Henrique Ito	60
Prof. Isabela Jardim	50
Prof. João Kleber	50
Prof. Larissa Moraes	50
Prof. Márcio Nogueira	50
Prof. Nívea Oliveira	50
Prof. Oscar Pires	50
Prof. Patrícia Queiroz	50

6.2 SQL para 3 consultas com IR (PK + FK) de recuperação de dados

Consulta 1: Detalhes do curso e professor

a) Descreva por qual razão a consulta é importante para o usuário.
Permite que os alunos e a administração vejam quais professores estão em cada curso

b) Código SQL.

```sql
SELECT c.Nome AS Curso, c.Periodo, COUNT(t.fk_Materia_ID) AS Quantidade_Materias
FROM Curso c
JOIN Tem t ON c.ID_Curso = t.fk_Curso_ID
JOIN Materia m ON t.fk_Materia_ID = m.ID_Materia
GROUP BY c.Nome, c.Periodo;
```

c) Resultado da consulta.

Curso	Periodo	Quantidade_Materi...
Engenharia Civil	10	1
Medicina	12	1
Direito	10	1
Administração	8	1
Ciência da Computação	8	1
Psicologia	10	1
Arquitetura	10	1
Engenharia Mecânica	10	1
Enfermagem	10	1
Educação Física	8	1
Contabilidade	8	1
Marketing	8	1
Odontologia	10	1
Engenharia de Produção	10	1
Farmácia	10	1

Consulta 2: Alunos matriculados em cada curso

a) Descreva por qual razão a consulta é importante para o usuário.

Para que a instituição saiba exatamente quais alunos estão matriculados em cada curso

b) Código SQL.

```sql
SELECT c.Nome AS Curso, a.Nome AS Aluno
FROM Curso c
JOIN Matricula_Aluno_Professor_Curso mapc ON c.ID_Curso = mapc.fk_Curso_ID
JOIN Aluno a ON mapc.fk_Aluno_ID = a.ID_Aluno
GROUP BY c.Nome, a.Nome;
```

c) Resultado da consulta.

Curso	Aluno
Engenharia Civil	Alice Silva
Engenharia Civil	Bruno Gomes
Engenharia Civil	Carla Souza
Engenharia Civil	Daniel Rocha
Medicina	Eduardo Costa
Medicina	Fernanda Dias
Medicina	Gabriel Martins
Medicina	Helena Santos
Direito	Ígor Oliveira
Direito	Joana Almeida
Direito	Kaique Barros
Direito	Larissa Teixeira
Administração	Mário Andrade
Administração	Natália Lima
Administração	Otávio Pereira

Consulta 3: Relação de matérias com seus cursos

a) Descreva por qual razão a consulta é importante para o usuário.
Para entender como as matérias estão distribuídas entre os cursos

b) Código SQL.

```
SELECT c.Nome AS Curso, m.Nome AS Materia
FROM Curso c
JOIN Tem t ON c.ID_Curso = t.fk_Curso_ID
JOIN Materia m ON t.fk_Materia_ID = m.ID_Materia;
```

c) Resultado da consulta.

Curso	Materia
Engenharia Civil	Matemática
Medicina	Português
Direito	História
Administração	Geografia
Ciência da Computação	Biologia
Psicologia	Química
Arquitetura	Física
Engenharia Mecânica	Inglês
Enfermagem	Espanhol
Educação Física	Educação Física
Contabilidade	Economia
Marketing	Informática
Odontologia	Filosofia
Engenharia de Produção	Sociologia
Farmácia	Artes

6.3 1 Stored procedure

SP 1:

a) Descreva o que a *Stored Produceure* (SP) faz (valores de INPUT e de OUTPUT).
• É um conjunto pré-definido de comando sql que são armazenados no banco de dados. A Sp pode aceitar valores de entrada e executar operações no banco de dados e retornando uma saída. Podem ser consultas de seleções, inserções, atualizações e exclusões, podendo conter logica de laços e de condicionais.

b) Descreva por qual razão a SP é importante para o usuário.
• Executa operações complexas, fazendo a performance ser melhor por ser mais rápido do que enviar múltiplos comandos para o cliente, pode ser reutilizada em diferentes aplicações, ajuda a prevenir inserções sql.

c) Código SQL

```
DELIMITER //

CREATE PROCEDURE InserirMatricula(
    IN aluno_id INT,
    IN professor_id INT,
    IN curso_id INT
)
BEGIN
    DECLARE alunoExiste, professorExiste, cursoExiste, jaMatriculado INT;

    SELECT COUNT(*) INTO alunoExiste FROM Aluno WHERE ID_Aluno = aluno_id;
    SELECT COUNT(*) INTO professorExiste FROM Professor WHERE ID_Professor = professor_id;
    SELECT COUNT(*) INTO cursoExiste FROM Curso WHERE ID_Curso = curso_id;
    SELECT COUNT(*) INTO jaMatriculado FROM Matricula_Aluno_Professor_Curso WHERE fk_Aluno_ID = aluno_id AND fk_Professor_ID = professor_id AND fk_Curs

    IF alunoExiste > 0 AND professorExiste > 0 AND cursoExiste > 0 AND jaMatriculado = 0 THEN
        INSERT INTO Matricula_Aluno_Professor_Curso (fk_Aluno_ID, fk_Professor_ID, fk_Curso_ID)
        VALUES (aluno_id, professor_id, curso_id);
    ELSEIF jaMatriculado > 0 THEN
        SIGNAL SQLSTATE '45000'
        SET MESSAGE_TEXT = 'Erro: Matrícula já existente.';
    ELSE
        SIGNAL SQLSTATE '45000'
        SET MESSAGE_TEXT = 'Erro: Aluno, professor ou curso não encontrado.';
    END IF;
END //

DELIMITER ;

CALL InserirMatricula(2, 1, 1);
CALL InserirMatricula(1, 16, 1);
CALL InserirMatricula(5, 6, 2);
```

d) Resultado da execução da SP

✖	19	17:14:25	CALL InserirMatricula(1, 1, 1)	Error Code: 1644. Erro: Matrícula já existente.
✖	20	17:17:23	CALL InserirMatricula(1, 16, 1)	Error Code: 1644. Erro: Aluno, professor ou curso não...
✔	21	17:17:50	CALL InserirMatricula(5, 6, 2)	1 row(s) affected

Três casos, um testando cada saída.

6.4 1 Trigger

Trigger 1:

e) Descreva o que o *Trigger* faz para as operações de INSERT / UPDATE / DELETE.
• É um tipo de Stored Producer que é automaticamente executado pelo sistema de banco de dados em resposta a eventos como Insert, Update, Delete de uma tabela.

f) Descreva por qual razão o *Trigger* é importante para o usuário.

• Assegura que as alterações no banco de dados sigam as regras de negócio, fazendo com que haja integridade de dados, simplifica a manutenção e operação dos sistemas, já que as tarefas são executadas de forma automática.

g) Código SQL

```
DELIMITER //

CREATE TRIGGER IncrementarContadorMatricula
AFTER INSERT ON Matricula_Aluno_Professor_Curso
FOR EACH ROW
BEGIN
    UPDATE Curso SET Periodo = Periodo + 1 WHERE ID_Curso = NEW.fk_Curso_ID;
END;

//
DELIMITER ;

INSERT INTO Matricula_Aluno_Professor_Curso (fk_Aluno_ID, fk_Professor_ID, fk_Curso_ID) VALUES (2, 2, 3);
```

h) Resultado da execução do *Trigger*