Projeto de banco de dados relacional em SQL Server

Objetivo de negócio do projeto: Atender as demandas de um sistema de gerenciamento de dados de uma escola fictícia de cursos online, possibilitando, assim, o armazenamento seguro, eficiente e confiável de todos os dados necessários para o funcionamento do negócio.

Objetivo técnico do projeto: Utilizar técnicas da linguagem SQL para criar o banco de dados e realizar as mais diversas inserções, alterações e consultas aos dados.

Sumário:

1. Modelagem de Dados

O primeiro passo é montar, caso já não tenha disponível, um diagrama conceitual e um diagrama entidade-relacionamento para solucionar a demanda em alto nível. Na Figura 1, é possível observar uma solução inicial para a demanda a qual aborda um modelo de negócio específico. No caso, a escola vende cursos de formação para empresas parceiras, criando-se, assim, turmas com respectivos cursos, os quais contém disciplinas e professores. Além disso, as disciplinas se relacionam com notas que são associadas a matrículas específicas, por sua vez com alunos e turmas.

Figura 1 - Diagrama conceitual do banco de dados relacional.

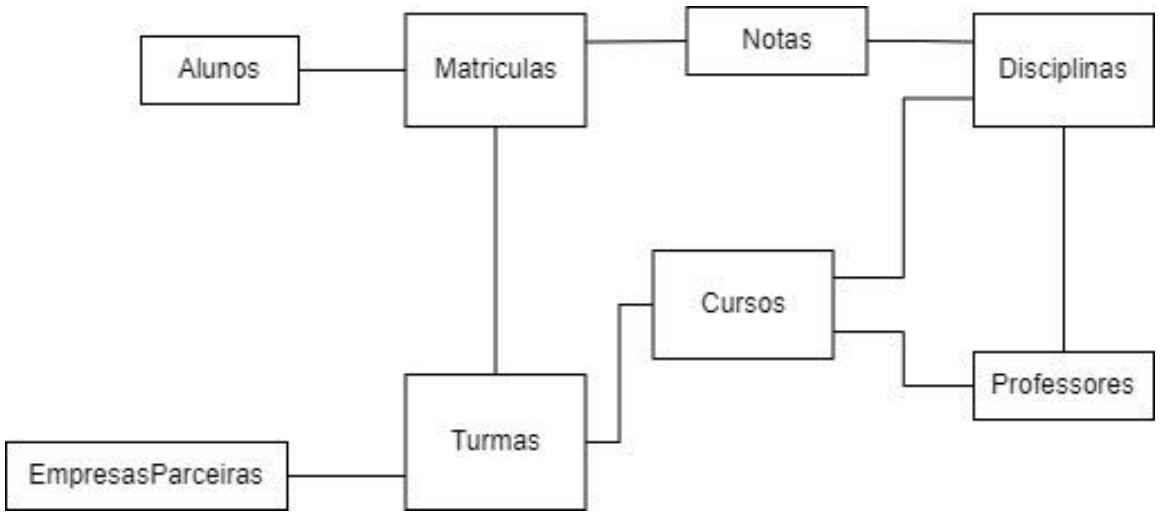
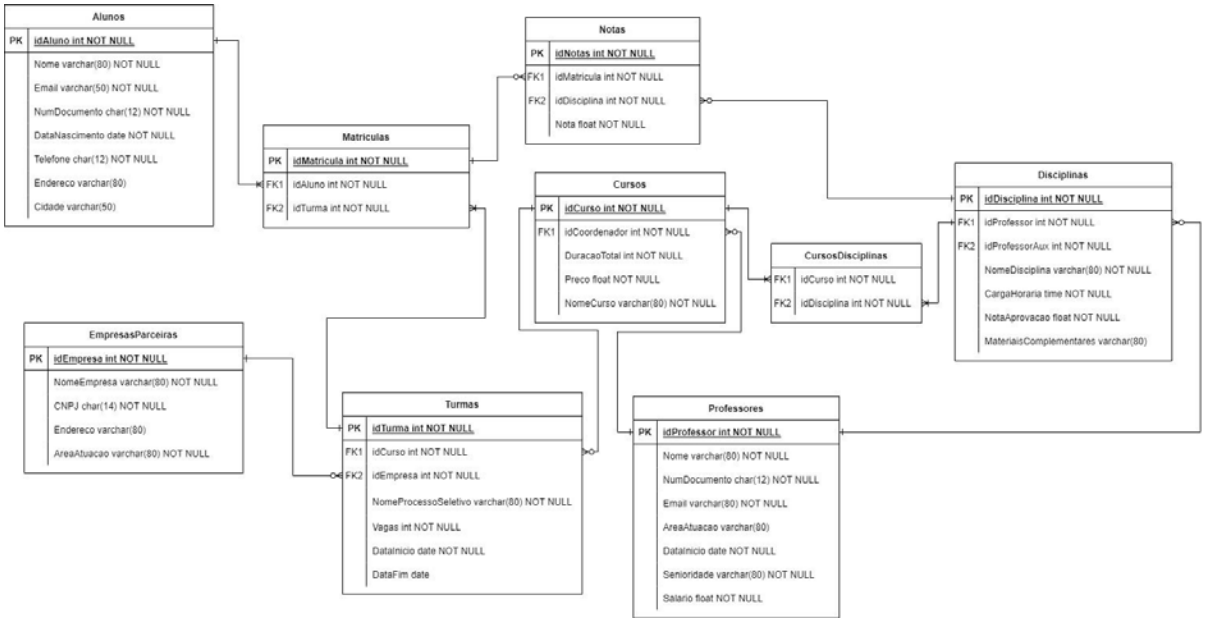


Figura 2 - Modelo lógico do banco de dados segundo notação James Martin.



2. Criação da Base de Dados e das Tabelas - SQL

De posso do DER, pode-se criar a base de dados em um Sistema Gerenciador de Banco de Dados (SGBD), que neste presente exemplo é utilizado o MSSQL. Os comandos a seguir mostram como a criação do banco de dados em si, das entidades e seus respectivos atributos.

```
CREATE DATABASE db_lufalufa;

USE db_lufalufa;

CREATE TABLE Alunos(
        idAluno int IDENTITY(1,1) PRIMARY KEY,
        Nome varchar(80) NOT NULL,
        Email varchar(50) NOT NULL,
        NumDocumento char(12) NOT NULL,
        DataNascimento date NOT NULL,
        Telefone char(11) NOT NULL,
        Endereco varchar(80),
        Cidade varchar(50)
);

CREATE TABLE EmpresasParceiras(
        idEmpresa int IDENTITY(1,1) PRIMARY KEY,
        NomeEmpresa varchar(80) NOT NULL,
        CNPJ char(14) NOT NULL,
        Endereco varchar(80),
        AreaAtuacao varchar(80) NOT NULL
);

CREATE TABLE Professores(
        idProfessor int IDENTITY(1,1) PRIMARY KEY,
        Nome varchar(80) NOT NULL,
        NumDocumento CHAR(12) NOT NULL,
        Email varchar(80) NOT NULL,
        AreaAtuacao varchar(80),
        DataInicio date NOT NULL,
        Senioridade varchar(80) NOT NULL,
        Salario float NOT NULL
);

CREATE TABLE Disciplinas(
        idDisciplina int IDENTITY(1,1) PRIMARY KEY,
        idProfessor int NOT NULL FOREIGN KEY REFERENCES Professores(idProfessor),
        idProfessorAux int NOT NULL FOREIGN KEY REFERENCES Professores(idProfessor),
        NomeDisciplina varchar(80) NOT NULL,
        CargaHoraria int NOT NULL,
```

```
        NotaAprovacao float NOT NULL,
        MateriaisComplementares varchar(80)
);

CREATE TABLE Cursos(
        idCurso int IDENTITY(1,1)  PRIMARY KEY,
        idCoordenador int NOT NULL FOREIGN KEY REFERENCES Professores(idProfessor),
        DuracaoTotal int NOT NULL,
        Preco float NOT NULL,
        NomeCurso varchar(80) NOT NULL
);

CREATE TABLE CursosDisciplinas(
        idCurso int NOT NULL FOREIGN KEY REFERENCES Cursos(idCurso),
        idDisciplina int NOT NULL FOREIGN KEY REFERENCES Disciplinas(idDisciplina)
);

CREATE TABLE Turmas(
        idTurma int IDENTITY(1,1)  PRIMARY KEY,
        idCurso int NOT NULL FOREIGN KEY REFERENCES Cursos(idCurso),
        idEmpresa int NOT NULL FOREIGN KEY REFERENCES EmpresasParceiras(idEmpresa),
        NomeProcessoSeletivo varchar(80) NOT NULL,
        Vagas int NOT NULL,
        DataInicio date NOT NULL,
        DataFim date
);

CREATE TABLE Matriculas(
        idMatricula int IDENTITY(1,1) PRIMARY KEY,
        idAluno int NOT NULL FOREIGN KEY REFERENCES Alunos(idAluno),
        idTurma int NOT NULL FOREIGN KEY REFERENCES Turmas(idTurma)
);

CREATE TABLE Notas(
        idNotas int IDENTITY(1,1) PRIMARY KEY,
        idMatricula int NOT NULL FOREIGN KEY REFERENCES Matriculas(idMatricula),
        idDisciplina int NOT NULL FOREIGN KEY REFERENCES Disciplinas(idDisciplina),
        Nota float NOT NULL
);
```

3. Inserção de Dados - SQL

Essa seção contém, majoritariamente, os comandos para popular as entidades com dados. Além disso, para solucionar um problema específico encontrado entre as entidades “Cursos”, “CursosDisciplinas” e “Disciplinas” e os respectivos atributos de “DuracaoTotal” e

“CargaHoraria”, foram criadas duas soluções dentro do próprio SQL. A primeira foi criar uma view da duração total do curso, sendo possível, então, a exclusão do próprio atributo duração total. Outra solução foi criar um trigger na entidade de “CursosDisciplinas” que é ativado quando um insert, update ou delete é realizado na mesma e a ação feita é de alterar o atributo “DuracaoTotal”.

Outro problema levantado foi na hora de cadastrar uma nota, que necessita de uma disciplina e matrícula associada, logo é necessário ocorrer uma validação entre a matrícula e a turma que aquela nota está atrelada. Para solucionar isto, foi criado uma view que une (Join) todas as entidades para ressaltar quais disciplinas estão associadas a uma matrícula. Em seguida, criou-se um procedure para inserir a nota atrelada a disciplina e matrícula caso aquelas informações estejam contidas dentro da view criada, caso contrário é apresentado um erro. Ressalta-se que estas questões poderiam ser solucionadas em um ambiente externo ao SQL Server como em uma rotina Python.

```
CREATE TRIGGER UpdateDuracao
ON dbo.CursosDisciplinas
FOR INSERT, UPDATE, DELETE
AS
BEGIN
	DECLARE @Action as char(1);
	SET @Action = (CASE WHEN EXISTS(SELECT * FROM INSERTED)
		 AND EXISTS(SELECT * FROM DELETED)
		THEN 'U'  -- Set Action to Updated.
		WHEN EXISTS(SELECT * FROM INSERTED)
		THEN 'I'  -- Set Action to Insert.
		WHEN EXISTS(SELECT * FROM DELETED)
		THEN 'D'  -- Set Action to Deleted.
		ELSE NULL -- Skip. It may have been a "failed delete".
	    END)

	DECLARE
	@idCurso int,
	@idDisciplina int,
	@CargaHoraria int,
	@idCurso2 int,
	@idDisciplina2 int,
	@CargaHoraria2 int

	IF @Action = 'I'
		BEGIN
			SELECT @idCurso = idCurso, @idDisciplina = idDisciplina FROM
INSERTED
```

```
                SELECT @CargaHoraria = CargaHoraria FROM dbo.Disciplinas
                        WHERE idDisciplina = @idDisciplina
                UPDATE dbo.Cursos SET DuracaoTotal = DuracaoTotal + @CargaHoraria
                        WHERE idCurso = @idCurso
        END
    ELSE IF @Action = 'D'
        BEGIN
                SELECT @idCurso = idCurso, @idDisciplina = idDisciplina FROM
DELETED
                SELECT @CargaHoraria = CargaHoraria FROM dbo.Disciplinas
                        WHERE idDisciplina = @idDisciplina
                UPDATE dbo.Cursos SET DuracaoTotal = DuracaoTotal - @CargaHoraria
                        WHERE idCurso = @idCurso
        END
    ELSE IF @Action = 'U'
        BEGIN
                SELECT @idCurso = idCurso, @idDisciplina = idDisciplina FROM
INSERTED
                SELECT @idCurso2 = idCurso, @idDisciplina2 = idDisciplina FROM
DELETED
                SELECT @CargaHoraria = CargaHoraria FROM dbo.Disciplinas
                        WHERE idDisciplina = @idDisciplina
                SELECT @CargaHoraria2 = CargaHoraria FROM dbo.Disciplinas
                        WHERE idDisciplina = @idDisciplina2
                UPDATE dbo.Cursos SET DuracaoTotal = DuracaoTotal + @CargaHoraria
                        WHERE idCurso = @idCurso
                UPDATE dbo.Cursos SET DuracaoTotal = DuracaoTotal - @CargaHoraria2
                        WHERE idCurso = @idCurso2
        END
END
GO

CREATE VIEW ViewID AS
SELECT Matriculas.idAluno,
        Matriculas.idMatricula,
        Turmas.idTurma,
        Cursos.idCurso,
        CursosDisciplinas.idDisciplina,
        Turmas.idEmpresa,
        Turmas.NomeProcessoSeletivo,
        Cursos.NomeCurso
FROM Matriculas
LEFT JOIN Turmas
ON Matriculas.idTurma = Turmas.idTurma
LEFT JOIN Cursos
ON Cursos.idCurso = Turmas.idCurso
LEFT JOIN CursosDisciplinas
ON Cursos.idCurso=CursosDisciplinas.idCurso;
```

```
GO

CREATE VIEW ViewDuracaoCurso AS
SELECT Cursos.NomeCurso,
        SUM(Disciplinas.CargaHoraria) AS DuracaoTotal
FROM Cursos
INNER JOIN CursosDisciplinas ON Cursos.idCurso = CursosDisciplinas.idCurso
INNER JOIN Disciplinas ON Disciplinas.idDisciplina = CursosDisciplinas.idDisciplina
GROUP BY Cursos.NomeCurso;
GO


CREATE PROCEDURE InserirNotas
@idMatricula int,
@idDisciplina int,
@Nota float
AS
BEGIN
        IF @idMatricula NOT IN (SELECT idMatricula FROM ViewID WHERE idDisciplina =
@idDisciplina)
                BEGIN
                        RAISERROR('Disciplina %i não pertence à Matrícula
%i',16,0,@idDisciplina, @idMatricula)
                END
        ELSE
                INSERT INTO Notas (idMatricula,idDisciplina,Nota)
                VALUES (@idMatricula,@idDisciplina,@Nota)

END
GO


INSERT INTO db_lufalufa.dbo.EmpresasParceiras
(NomeEmpresa, CNPJ, Endereco, AreaAtuacao)
VALUES('Empresa1', '58160789000128', 'Av Paulista', 'Finanças'),
('Empresa2', '60701190000106', 'Rua Consolação', 'Finanças'),
('Empresa3', '76535764000143', 'Rua dos Telefones', 'Telecomunicações'),
('Empresa4', '09346601000125', 'Rua das Acoes', 'Finanças'),
('Empresa5', '61590410000124', 'Rua da Saude', 'Saude'),
('Empresa6', '06990590000123', 'Rua do Buscador', 'Tecnologia');

INSERT INTO db_lufalufa.dbo.Professores
(Nome, NumDocumento, Email, AreaAtuacao, DataInicio, Senioridade, Salario)
VALUES('Brian', '963595317541', 'brian@gmail.com', 'Estudante', '2021-04-02', 'Junior', 1700),
('Rafa', '735454338714', 'rafael@gmail.com', 'Professor', '2020-08-21', 'Pleno', 2100),
('Livia', '674595141657', 'livia@gmail.com', 'Banco de Dados', '2019-01-01', 'Senior', 2400),
('Romero', '684083121474', 'romero@gmail.com', 'Cientista de Dados', '2022-04-03', 'Pleno', 2000),
('Bruna', '256579287646', 'bruna@gmail.com', 'DEVOPS', '2022-06-03', 'Pleno', 2050);
```

```
INSERT INTO db_lufalufa.dbo.Cursos
(idCoordenador, DuracaoTotal, Preco, NomeCurso)
VALUES(5, 0, 14000, 'DEVOPS'),
(2, 0, 15000, 'Data Science'),
(1, 0, 21000, 'Full Stack');


INSERT INTO db_lufalufa.dbo.Disciplinas
(idProfessor, idProfessorAux, NomeDisciplina, CargaHoraria, NotaAprovacao)
VALUES(5, 1, 'Python', 20, 5),
(3, 4, 'JavaScript', 24, 5),
(2, 5, 'CSS', 18, 5),
(3, 2, 'MSSQL',20, 5),
(1, 5, 'Estatistica',22, 5),
(4, 2, 'Machine Learning', 18, 5),
(2, 5, 'Docker', 20, 5),
(3, 1, 'Kubernetes', 26, 5),
(5, 4, 'AWS', 16, 5);

INSERT INTO db_lufalufa.dbo.CursosDisciplinas
(idCurso, idDisciplina)
VALUES(2, 1);
INSERT INTO db_lufalufa.dbo.CursosDisciplinas
(idCurso, idDisciplina)
VALUES(3, 1);
INSERT INTO db_lufalufa.dbo.CursosDisciplinas
(idCurso, idDisciplina)
VALUES(3, 2);
INSERT INTO db_lufalufa.dbo.CursosDisciplinas
(idCurso, idDisciplina)
VALUES(1, 2);
INSERT INTO db_lufalufa.dbo.CursosDisciplinas
(idCurso, idDisciplina)
VALUES(3, 3);
INSERT INTO db_lufalufa.dbo.CursosDisciplinas
(idCurso, idDisciplina)
VALUES(2, 4);
INSERT INTO db_lufalufa.dbo.CursosDisciplinas
(idCurso, idDisciplina)
VALUES(1, 4);
INSERT INTO db_lufalufa.dbo.CursosDisciplinas
(idCurso, idDisciplina)
VALUES(2, 5);
INSERT INTO db_lufalufa.dbo.CursosDisciplinas
(idCurso, idDisciplina)
VALUES(2, 6);
INSERT INTO db_lufalufa.dbo.CursosDisciplinas
```

```
(idCurso, idDisciplina)
VALUES(3, 6);
INSERT INTO db_lufalufa.dbo.CursosDisciplinas
(idCurso, idDisciplina)
VALUES(1, 7);
INSERT INTO db_lufalufa.dbo.CursosDisciplinas
(idCurso, idDisciplina)
VALUES(3, 7);
INSERT INTO db_lufalufa.dbo.CursosDisciplinas
(idCurso, idDisciplina)
VALUES(1, 8);
INSERT INTO db_lufalufa.dbo.CursosDisciplinas
(idCurso, idDisciplina)
VALUES(2, 8);
INSERT INTO db_lufalufa.dbo.CursosDisciplinas
(idCurso, idDisciplina)
VALUES(1, 9);

INSERT INTO db_lufalufa.dbo.Turmas
(idCurso, idEmpresa, NomeProcessoSeletivo, Vagas, DataInicio, DataFim)
VALUES (2, 1, 'TopCoders', 25, '2022-05-12', '2022-11-23'),
        (1, 1, 'TopCoders', 50, '2022-05-12', '2022-11-23'),
        (1, 2, 'Bootcamp Itau Devs', 40, '2022-05-16', '2023-01-25'),
        (1, 3, 'Oi Devs', 15, '2022-09-21', '2023-02-25'),
        (1, 5, 'Sirio Libanes Tech', 10, '2022-08-24', '2023-02-08'),
        (3, 4, 'Programa <Dev>a', 20, '2022-09-14', '2023-03-14'),
        (1, 4, 'Programa <Dev>a', 20, '2022-09-14', '2023-03-14'),
        (2, 6, 'Prep Tech _afro', 30, '2022-08-01', '2022-12-12');

INSERT INTO db_lufalufa.dbo.Alunos
(Nome, Email, NumDocumento, DataNascimento, Telefone, Endereco, Cidade)
VALUES('Gustavo P', 'gustavo@gmail.com', '12345678900', '1990-08-01', '16998124567', 'Rua Sem
Nome, 456', 'Sao Paulo'),
('Bruno B', 'brunobe@gmail.com', '46327939066', '1998-01-07', '11869915959', 'Avenida Jorge, 654',
'Santos'),
('Rodrigo S', 'rodrigo@gmail.com', '25138816055', '1997-10-18', '8932894954', 'Rua Leste, 69',
'Recife'),
('Fernanda D', 'fernanda@gmail.com', '20604797036', '1995-02-05', '6238536862', 'Rua Agile, 87',
'Sao Paulo'),
('Daniel G', 'daniel@gmail.com', '66785791006', '1996-11-03', '9222880881', 'Rua Ferrari, 645', 'Sao
Paulo'),
('Guilherme F', 'guilherme@gmail.com', '21158345003', '1995-12-25', '7523344423', 'Avenida Golf,
87', 'Campinas'),
('Joao R', 'joao@gmail.com', '21471223000', '1997-11-01', '9522254771', 'Rua Honda, 64', 'Recife'),
('Bruno S', 'bruno@gmail.com', '40445343060', '1990-02-22', '9635805160', 'Rua Lambo, 32',
'Salvador'),
('Guilherme H', 'guilhermeh@gmail.com', '14104148091', '1989-03-26', '9229334466', 'Rua R35, 645',
'Orlando'),
```

('Tania R', 'tania@gmail.com', '85305187052', '1986-04-25', '7735427288', 'Avenida Silvia, 678', 'Barueri'),
('Rodolfo J', 'rodolfo@gmail.com', '56526770070', '1985-05-24', '8421570788', 'Rua Lancer, 543', 'Osasco'),
('Brenda T', 'brenda@gmail.com', '22162850052', '1981-06-29', '9425224813', 'Rua Evo, 786', 'Rio de Janeiro'),
('Carol E', 'carol@gmail.com', '48813002025', '1998-05-12', '9622766494', 'Rua Fiat, 88', 'Sao Paulo'),
('Renato J', 'renato@gmail.com', '90689598084', '1998-08-08', '9622665102', 'Avenida Audi, 687', 'Porto Alegre'),
('Luiz G', 'luiz@gmail.com', '39553954057', '1997-09-25', '7935671335', 'Rua Sandero, 645', 'Sao Paulo'),
('Gabriel I', 'gabriel@gmail.com', '54939601099', '1996-08-12', '1538173994', 'Rua Nissan, 687', 'Sao Paulo'),
('Lucas U', 'lucas@gmail.com', '77256542089', '1995-07-11', '672693-8347', 'Avenida GTR, 845', 'Porto Alegre'),
('Ariel T', 'ariel@gmail.com', '90811145069', '1998-02-05', '7736177334', 'Rua Senna, 68', 'Rio de Janeiro'),
('Jessica L', 'jessica@gmail.com', '37172177045', '1999-01-03', '8536845927', 'Rua Inter, 356', 'Rio de Janeiro'),
('Joyce Q', 'joyce@gmail.com', '74238597028', '1998-03-22', '5429599111', 'Avenida Breno, 154', 'Sao Paulo');

INSERT INTO db_lufalufa.dbo.Matriculas
(idAluno, idTurma)
VALUES (1, 1),
(2, 2),
(3, 2),
(4, 3),
(5, 1),
(6, 3),
(7, 4),
(8, 3),
(9, 2),
(10, 5),
(11, 5),
(12, 6),
(13, 2),
(14, 3),
(15, 4),
(16, 3),
(17, 2),
(18, 1),
(19, 3),
(20, 1);

INSERT INTO db_lufalufa.dbo.Notas
(idMatricula, idDisciplina, Nota)
VALUES (1,1,7.8),

```
(1, 4, 6.7),
(1, 5, 9.2),
(1, 6, 8.0),
(2, 7, 9.4),
(13, 8, 7.1),
(2, 9, 7.7),
(3, 7, 6.5),
(15, 8, 8.3),
(4, 7, 6.2),
(6, 8, 6.6),
(11, 9, 9.7),
(12, 9, 5.9),
(5, 1, 8.8),
(5, 4, 7.7),
(8, 9, 7.8),
(9, 9, 7.9),
(14, 7, 9.7),
(16, 9, 10.0),
(19, 8, 6.2);

EXECUTE InserirNotas
@idMatricula = 20,
@idDisciplina = 4,
@Nota = 5.4
```

3.1) Consulta simples a cada entidade

```
SELECT * FROM Alunos;
SELECT * FROM Cursos;
SELECT * FROM CursosDisciplinas;
SELECT * FROM Disciplinas;
SELECT * FROM EmpresasParceiras;
SELECT * FROM Matriculas;
SELECT * FROM Notas;
SELECT * FROM Professores;
SELECT * FROM Turmas;
```

4. Consultas aos Bancos - Perguntas de Negócios

Esta seção apresenta perguntas de negócios que poderiam ser formuladas para responder alguma dúvida, curiosidade ou problema da empresa com o intuito de ilustrar algumas consultas.

4.1) Quais e quantos alunos têm notas maiores ou iguais a 8.5?

```
SELECT Matriculas.idMatricula, Nome, Nota
```

```
FROM Alunos
INNER JOIN Matriculas on Matriculas.idAluno = Alunos.idAluno
INNER JOIN NOTAS ON Notas.idMatricula = Matriculas.idMatricula
WHERE Nota >= 8.5 ORDER BY Nota DESC
```

Resultado:

idMatricula	Nome	Nota
16	Gabriel I	10
14	Renato J	9,7
11	Rodolfo J	9,7
2	Bruno B	9,4
1	Gustavo P	9,2
5	Daniel G	8,8

Essa lista de alunos seria de interesse da escola para poder premiar aqueles com médias elevadas; ter uma seleção de alunos para indicar para empresas que estão contratando, etc.

4.2) Quais são as empresas parceiras que efetivamente possuem turmas?

```
SELECT DISTINCT(NomeEmpresa) AS EmpresasComTurmas
FROM EmpresasParceiras INNER JOIN Turmas ON EmpresasParceiras.idEmpresa =
Turmas.idEmpresa;
```

Resultado:

EmpresasComTurmas
B3
Google
Itau
Oi
Safra
Sirio-Libanes

Essa pergunta poderia respondida verificando os números de turmas por empresa registrada:

```
SELECT DISTINCT(EmpresasParceiras.NomeEmpresa) AS Empresa, count(Turmas.idEmpresa) AS
Qtde_Turmas_total
FROM EmpresasParceiras FULL OUTER JOIN Turmas ON EmpresasParceiras.idEmpresa =
Turmas.idEmpresa
GROUP BY EmpresasParceiras.NomeEmpresa;
```

Resultado:

Empresa	Qtde_Turmas_total
B3	2
Google’	1
Itau	1
Oi	1
Safra	2
Sirio-Libanes	1

Essa pergunta serviria para saber a porcentagem de efetivação de contratação das empresas registradas/prospectadas.

4.3) Quais disciplinas cada professor já lecionou?

```
SELECT Nome, NomeDisciplina, 'Professor Titular' AS Tipo
FROM Disciplinas
INNER JOIN Professores ON Disciplinas.idProfessor = Professores.idProfessor
UNION ALL
SELECT Nome, NomeDisciplina, 'Professor Auxiliar' AS Tipo
FROM Disciplinas
INNER JOIN Professores ON Disciplinas.idProfessorAux = Professores.idProfessor
ORDER BY Nome;
```

Resultado:

Nome	NomeDisciplina	Tipo
Brian	Estatistica	Professor Titular
Brian	Python	Professor Auxiliar
Brian	Kubernetes	Professor Auxiliar
Bruna	Estatistica	Professor Auxiliar
Bruna	CSS	Professor Auxiliar
Bruna	Python	Professor Titular
Bruna	AWS	Professor Titular
Bruna	Docker	Professor Auxiliar
Livia	Kubernetes	Professor Titular
Livia	MSSQL	Professor Titular
Livia	JavaScript	Professor Titular
Rafa	CSS	Professor Titular
Rafa	Docker	Professor Titular
Rafa	MSSQL	Professor Auxiliar
Rafa	Machine Learning	Professor Auxiliar
Romero	AWS	Professor Auxiliar
Romero	JavaScript	Professor Auxiliar
Romero	Machine Learning	Professor Titular

Essa pergunta serve para saber em qual disciplina tal professor já atuou na escola para facilitar atribuições futuras.

4.4) Quem são os alunos vinculados a determinada empresa parceira?

Exemplo: Empresa Parceira Safra

```
SELECT Nome
FROM dbo.EmpresasParceiras
INNER JOIN dbo.Turmas ON dbo.EmpresasParceiras.idEmpresa = dbo.Turmas.idEmpresa
INNER JOIN dbo.Matriculas ON dbo.Turmas.idTurma = dbo.Matriculas.idTurma
INNER JOIN dbo.Alunos ON dbo.Alunos.idAluno = dbo.Matriculas.idAluno
WHERE NomeEmpresa = 'Empresa2';
```

Resultado:

Nome
Gustavo P
Bruno B
Rodrigo S
Daniel G
Guilherme H
Carol E
Lucas U
Ariel T
Joyce Q

Essa informação é de interesse da empresa parceira/cliente a fins de registro/acompanhamento.

4.5) Quais são as matérias que apresentam pelo menos uma reprovação?

```
SELECT DISTINCT NomeDisciplina
FROM Notas INNER JOIN Disciplinas ON Notas.idDisciplina = Disciplinas.idDisciplina
WHERE Notas.Nota < 7.0;
```

Resultado:

NomeDisciplina
AWS
Docker
Kubernetes
MSSQL

Essa pergunta pode servir como um indicador de quais disciplinas a ementa, processo de ensino e etc deve ser melhorada.

4.6) Qual o custo gasto com formações por empresa parceira?

Exemplo: Empresa Parceira Google

```
SELECT SUM(Preco*Vagas) AS CustoFormacao
FROM dbo.EmpresasParceiras
INNER JOIN dbo.Turmas ON dbo.EmpresasParceiras.idEmpresa = dbo.Turmas.idEmpresa
INNER JOIN dbo.Cursos ON dbo.Cursos.idCurso = dbo.Turmas.idCurso
WHERE NomeEmpresa = 'Empresa3';
```

Resultado:

CustoFormacao
450000

Essa pergunta é de interesse da escola para saber o quanto cada empresa parceira está disposta a gastar com os cursos.

4.7) Houve retorno de algum cliente (empresa parceira)?

```
SELECT EmpresasParceiras.NomeEmpresa AS Empresa, Turmas.DataInicio
FROM EmpresasParceiras FULL OUTER JOIN Turmas ON EmpresasParceiras.idEmpresa =
Turmas.idEmpresa
GROUP BY EmpresasParceiras.NomeEmpresa, Turmas.DataInicio;
```

Resultado:

Empresa	**DataInicio**
B3	2022-09-14
Google	2022-08-01
Itau	2022-05-16
Oi	2022-09-21
Safra	2022-05-12
Sirio-Libanes	2022-08-24

Caso haja um nome de empresa repetido, quer dizer que há turmas com datas de início diferentes, indicando que ela voltou a contratar alguma turma com a escola. Isso indicaria a satisfação do cliente com a formação dada pela escola e o valor que tal empresa atribui a isso.

5. Inserção de dados no banco com arquivo csv e python

O código a seguir ilustra um modo simples de ler um arquivo .csv que contenha os dados de cadastro a serem inseridos no banco de dados. Para tanto, é utilizado a biblioteca sqlalchemy.

```
from sqlalchemy import create_engine, text
import csv

db_url = "mssql+pymssql://localhost:NÚMERO_DA_PORTA_TCPIP/db_lufalufa"
engine = create_engine(db_url, pool_size=5, pool_recycle=3600)

import csv
with open('nomes.csv') as file:
    planilha = list(csv.reader(file, delimiter=','))

string_final = ''
for elemento in planilha:
    string = '('
    for elementos in elemento:
        string = string + "'" + elementos + "'"
    string = (string + ')').replace(';','\',\'')
    string_final = string_final + ',' + string

string_final = string_final[1::]

conn = engine.connect()
sql_text = text(f'INSERT INTO dbo.Alunos VALUES {string_final}')
result = conn.execute(sql_text)
```