BANCO DE DADOS

Trabalho – Relatório

Curso:	Tecnologia em banco De Dados
Aluno(a):	Emerson Pereira de Souza
RU:	418525

• 1ª Etapa – Modelagem

Pontuação: 25 pontos.

Dado o estudo de caso abaixo, elabore o Modelo Entidade-Relacionamento (MER), isto é, o modelo conceitual.

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

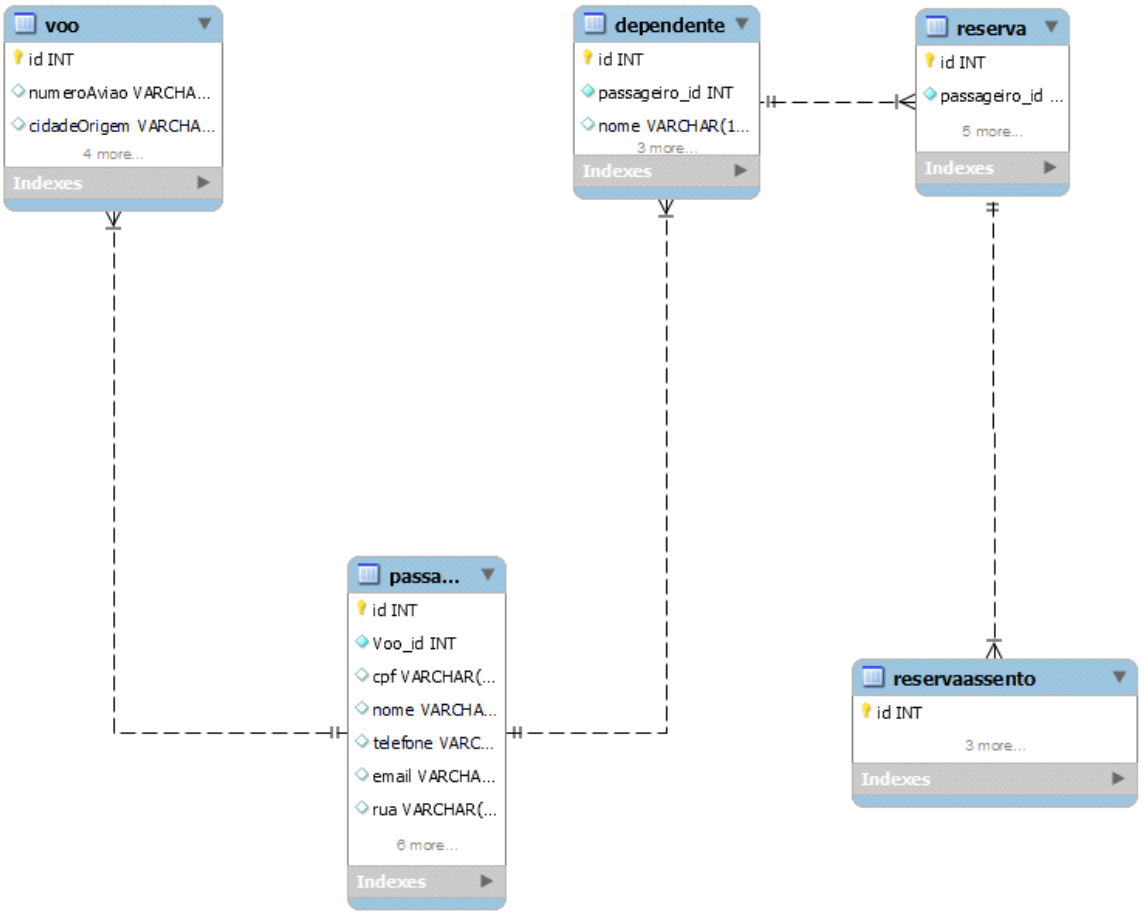
- Entidades;
- Atributos;
- Relacionamentos;
- Cardinalidades.

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:

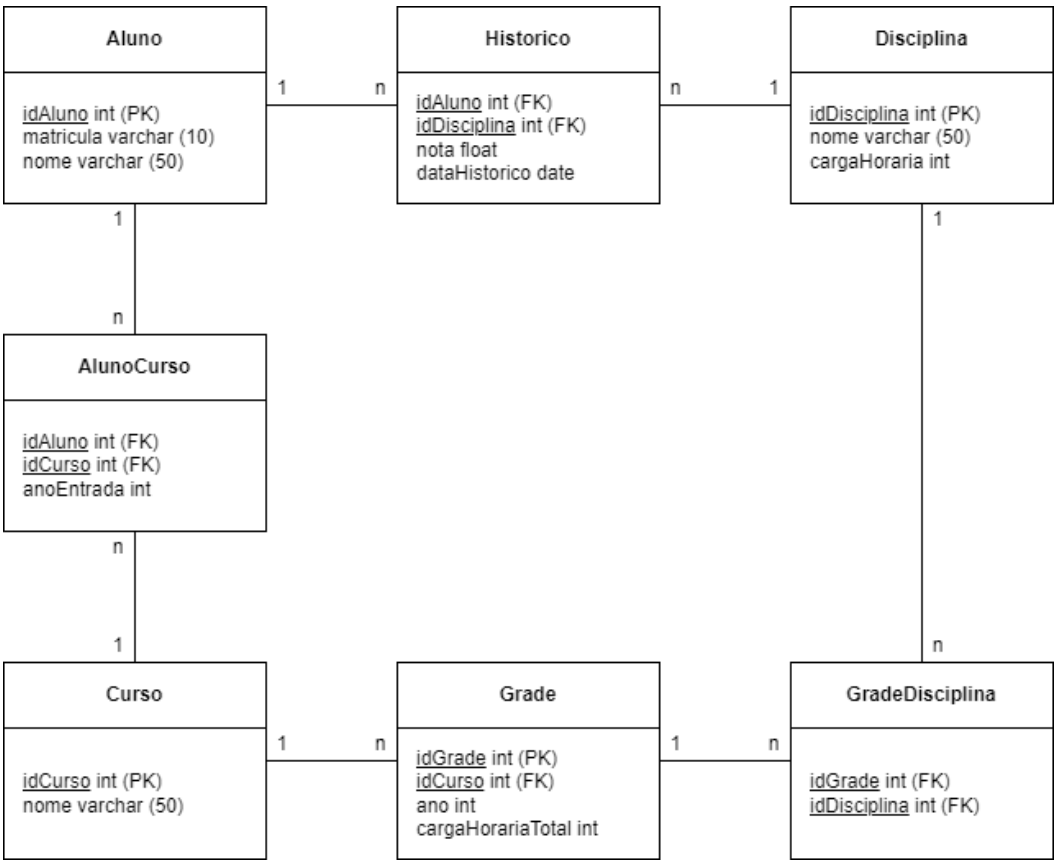
- Voo – Deverão ser armazenados os seguintes dados: identificação do voo, número do avião, cidade de origem, cidade destino, data do voo e hora do voo;
- Assentos – Deverão ser armazenados os seguintes dados: identificação do assento e quantidade;

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


voo
id INT
numeroAviao VARCHA...
cidadeOrigem VARCHA...
4 more...
Indexes
dependente
id INT
passageiro_id INT
nome VARCHAR(1...
3 more...
Indexes
reserva
id INT
passageiro_id ...
5 more...
Indexes
passa...
id INT
Voo_id INT
cpf VARCHAR(...
nome VARCHA...
telefone VARC...
email VARCHA...
rua VARCHAR(...
6 more...
Indexes
reservaassento
id INT
3 more...
Indexes

- ## 2ª Etapa – Implementação

Considere o seguinte Modelo Relacional (lógico):



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 criação, utilize os comandos contidos no arquivo “Trabalho – Populando o Banco de Dados”, o qual contém todos os comandos de inserção de dados (fictícios) necessários para a realização dos testes.

Pontuação: 25 pontos.

- Implemente um Banco de Dados chamado “Faculdade”. Após, crie 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.
- use faculdade;
- show databases;
-
- CREATE TABLE Alunos (
- id INTEGER UNSIGNED NOT NULL AUTO_INCREMENT,
- matricula VARCHAR(10) NULL,
- nome VARCHAR(50) NULL,
- PRIMARY KEY(id)
-);
-
- insert into Alunos 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');

                    select *from alunos;

CREATE TABLE Disciplina (
  id INTEGER UNSIGNED NOT NULL AUTO_INCREMENT,
  nome VARCHAR(50) NULL,
  cargahoraria INT NULL,
  PRIMARY KEY(id)
);


insert into Disciplina 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);

                    select * from disciplina;



CREATE TABLE Curso (
  id INTEGER UNSIGNED NOT NULL AUTO_INCREMENT,
  nome VARCHAR(50) NULL,
  PRIMARY KEY(id)
);

insert into Curso 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');

                    select * from curso;


```

```
CREATE TABLE Historico (
  id INTEGER UNSIGNED NOT NULL AUTO_INCREMENT,
  Disciplina_id INTEGER UNSIGNED NOT NULL,
  Alunos_id INTEGER UNSIGNED NOT NULL,
  nota FLOAT NULL,
  dataHistorico DATE NULL,
  PRIMARY KEY(id),
  INDEX Historico_FKIndex1(Alunos_id),
  INDEX Historico_FKIndex2(Disciplina_id)
);

insert into Historico values (1, 3,  1, 90,  '2022-12-09'),
                              (2, 3,  3, 75,  '2022-12-09'),
                        (4, 3,  5, 85,  '2022-12-09'),
                              (5, 9,  8, 80,  '2022-12-16'),
                        (6, 9,  9, 75,  '2022-12-16'),
                        (7, 9, 11, 70,  '2022-12-16'),
                              (8, 13, 12, 70, '2022-12-09'),
                        (9, 13, 13, 70, '2022-12-09'),
                        (10, 13, 14, 82, '2022-12-09'),
                        (11, 15,  2, 76, '2022-12-16'),
                        (12, 15,  4, 80, '2022-12-16'),
                        (13, 15,  6, 89, '2022-12-16'),
                        (14, 14, 7, 88,  '2022-12-16'),
                        (15, 15, 15, 90, '2022-12-17'),
                                                (16, 13, 16, 91, '2022-12-08'),
                        (17, 12, 17, 93, '2022-12-09'),
                        (18, 11, 18, 92, '2022-12-10'),
                        (19, 10, 19, 95, '2022-12-15');
                          select *from historico;
```

```
•
•
•  CREATE TABLE alunoCurso (
•   id INTEGER UNSIGNED NOT NULL AUTO_INCREMENT,
•   Alunos_id INTEGER UNSIGNED NOT NULL,
•   Curso_id INTEGER UNSIGNED NOT NULL,
•   anoentrada INT NULL,
•   PRIMARY KEY(id),
•   INDEX alunoCurso_FKIndex1(Curso_id),
•   INDEX alunoCurso_FKIndex2(Alunos_id)
• );
•
•
• insert into AlunoCurso values ( 1, 1, 1, 2023),
•                               ( 2, 2, 2, 2023),
•                               ( 3, 3, 3, 2022),
•                               ( 4, 4, 4, 2023),
•                               ( 5, 5, 5, 2023),
•                               ( 6, 6, 6, 2023),
•                                       ( 7, 7, 7, 2023),
•                               ( 8, 8, 8, 2023),
•                               ( 9, 9, 9, 2022),
•                               (10, 10, 9, 2023),
•                               (11, 11, 8, 2023),
•                               (12, 12, 7, 2023),
•                               (13, 13, 6, 2022),
•                               (14, 14, 5, 2023),
•                               (15, 15, 4, 2022),
•                               (16, 16, 3, 2023),
•                               (17, 17, 2, 2023),
•                               (18, 18, 1, 2023),
```

```
                    (19, 19, 1, 2023);
                select * from AlunoCurso;


CREATE TABLE Grade (
  id INTEGER UNSIGNED NOT NULL AUTO_INCREMENT,
  Curso_id INTEGER UNSIGNED NOT NULL,
  ano INT NULL,
  cargaHorariatotal INT NULL,
  PRIMARY KEY(id),
  INDEX Grade_FKIndex1(Curso_id)
);

insert into Grade 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);
          select * from Grade;

CREATE TABLE GradeDisciplina (
  id INTEGER UNSIGNED NOT NULL AUTO_INCREMENT,
  Grade_id INTEGER UNSIGNED NOT NULL,
  Disciplina_id INTEGER UNSIGNED NOT NULL,
```

- id_2 INTEGER UNSIGNED NULL,
- PRIMARY KEY(id),
- INDEX GradeDisciplina_FKIndex1(Disciplina_id),
- INDEX GradeDisciplina_FKIndex2(Grade_id)
-);
-
- INSERT INTO GradeDisciplina (Grade_id, Disciplina_id, id_2)
- VALUES (1, 1, 0), (1, 2 ,0), (1, 3, 0), (1, 4, 0), (1, 5, 0), (1, 6, 0), (1, 7, 0), (1, 8, 0), (1, 9, 0), (1, 10, 0), (1, 11, 0), (1, 12, 0), (1, 13, 0), (1, 14, 0), (1, 15, 0),
- (2, 1, 0), (2, 2, 0), (2, 3, 0), (2, 4, 0), (2, 5, 0), (2, 6, 0), (2, 7, 0), (2, 8, 0), (2, 9, 0), (2, 10, 0), (2, 11, 0), (2, 12, 0), (2, 13, 0), (2, 14, 0), (2, 15, 0),
- (3, 1, 0), (3, 2, 0), (3, 3, 0), (3, 4, 0), (3, 5, 0), (3, 6, 0), (3, 7, 0), (3, 8, 0), (3, 9, 0), (3, 10, 0), (3, 11, 0), (3, 12, 0), (3, 13, 0), (3, 14, 0), (3, 15, 0),
- (4, 1, 0), (4, 2, 0), (4, 3, 0), (4, 4, 0), (4, 5, 0), (4, 6, 0), (4, 7, 0), (4, 8, 0), (4, 9, 0), (4, 10, 0), (4, 11, 0), (4, 12, 0), (4, 13, 0), (4, 14, 0), (4, 15, 0),
- (5, 1, 0),(5, 2, 0), (5, 3, 0), (5, 4, 0), (5, 5, 0), (5, 6, 0), (5, 7, 0), (5, 8, 0), (5, 9, 0), (5, 10, 0), (5, 11, 0), (5, 12, 0), (5, 13, 0), (5, 14, 0), (5, 15, 0),
- (6, 1, 0), (6, 2, 0), (6, 3, 0), (6, 4, 0), (6, 5, 0), (6, 6, 0), (6, 7, 0), (6, 8, 0), (6, 9, 0), (6, 10, 0), (6, 11, 0), (6, 12, 0), (6, 13, 0), (6, 14, 0), (6, 15, 0),
- (7, 1, 0), (7, 2,0), (7, 3, 0), (7, 4, 0), (7, 5, 0), (7, 6, 0), (7, 7, 0), (7, 8, 0), (7, 9, 0), (7, 10, 0), (7, 11, 0), (7, 12, 0), (7, 13, 0), (7, 14, 0), (7, 15, 0),
- (8, 1, 0), (8, 2, 0), (8, 3, 0), (8, 4, 0), (8, 5, 0), (8, 6, 0), (8, 7, 0), (8, 8, 0), (8, 9, 0), (8, 10, 0), (8, 11, 0), (8, 12, 0), (8, 13, 0), (8, 14, 0), (8, 15, 0),

- (9, 1, 0), (9, 2, 0), (9, 3, 0), (9, 4, 0), (9, 5, 0), (9, 6, 0), (9, 7, 0), (9, 8, 0), (9, 9, 0), (9, 10, 0), (9, 11, 0), (9, 12, 0), (9, 13, 0), (9, 14, 0), (9, 15, 0),
- (10, 1, 0), (10, 2, 0), (10, 3, 0), (10, 4, 0), (10, 5, 0), (10, 6, 0), (10, 7, 0), (10, 8, 0), (10, 9, 0), (10, 10, 0), (10, 11, 0), (10, 12, 0), (10, 13, 0), (10, 14, 0), (10, 15, 0),
- (11, 1, 0), (11, 2, 0), (11, 3, 0), (11, 4, 0), (11, 5, 0), (11, 6, 0), (11, 7, 0), (11, 8, 0), (11, 9, 0), (11, 10, 0), (11, 11, 0), (11, 12, 0), (11, 13, 0), (11, 14, 0), (11, 15,0),
- (12, 1, 0), (12, 2, 0), (12, 3, 0), (12, 4, 0), (12, 5, 0), (12, 6, 0), (12, 7, 0), (12, 8, 0), (12, 9, 0), (12, 10, 0), (12, 11, 0), (12, 12, 0), (12, 13, 0), (12, 14, 0), (12, 15, 0);

Pontuação: 10 pontos.

- Implemente uma consulta para listar o quantitativo de cursos existentes.
- SELECT COUNT(*) AS quantidade_cursos FROM Curso;

SELECT COUNT(*) AS quantidade_cursos FROM Curso;

Pontuação: 10 pontos.

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

SELECT nome FROM Disciplina;

-

SELECT nome FROM Disciplina;

Pontuação: 10 pontos.

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

```
SELECT c.nome AS curso, a.nome AS aluno
FROM Curso c
LEFT JOIN AlunoCurso ac ON c.id = ac.Curso_id
LEFT JOIN Alunos a ON ac.Alunos_id = a.id
ORDER BY c.nome DESC;
```

-

Pontuação: 10 pontos.

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

```
SELECT Curso.nome AS Curso, AVG(Historico.nota) AS Media
FROM Curso
JOIN AlunoCurso ON Curso.id = AlunoCurso.Curso_id
JOIN Alunos ON AlunoCurso.Alunos_id = Alunos.id
JOIN Historico ON Alunos.id = Historico.Alunos_id
GROUP BY Curso.id;
```

-

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, COUNT(AlunoCurso.id) AS quantidade_alunos
- FROM Curso
- JOIN AlunoCurso ON Curso.id = AlunoCurso.Curso_id
- GROUP BY Curso.nome;