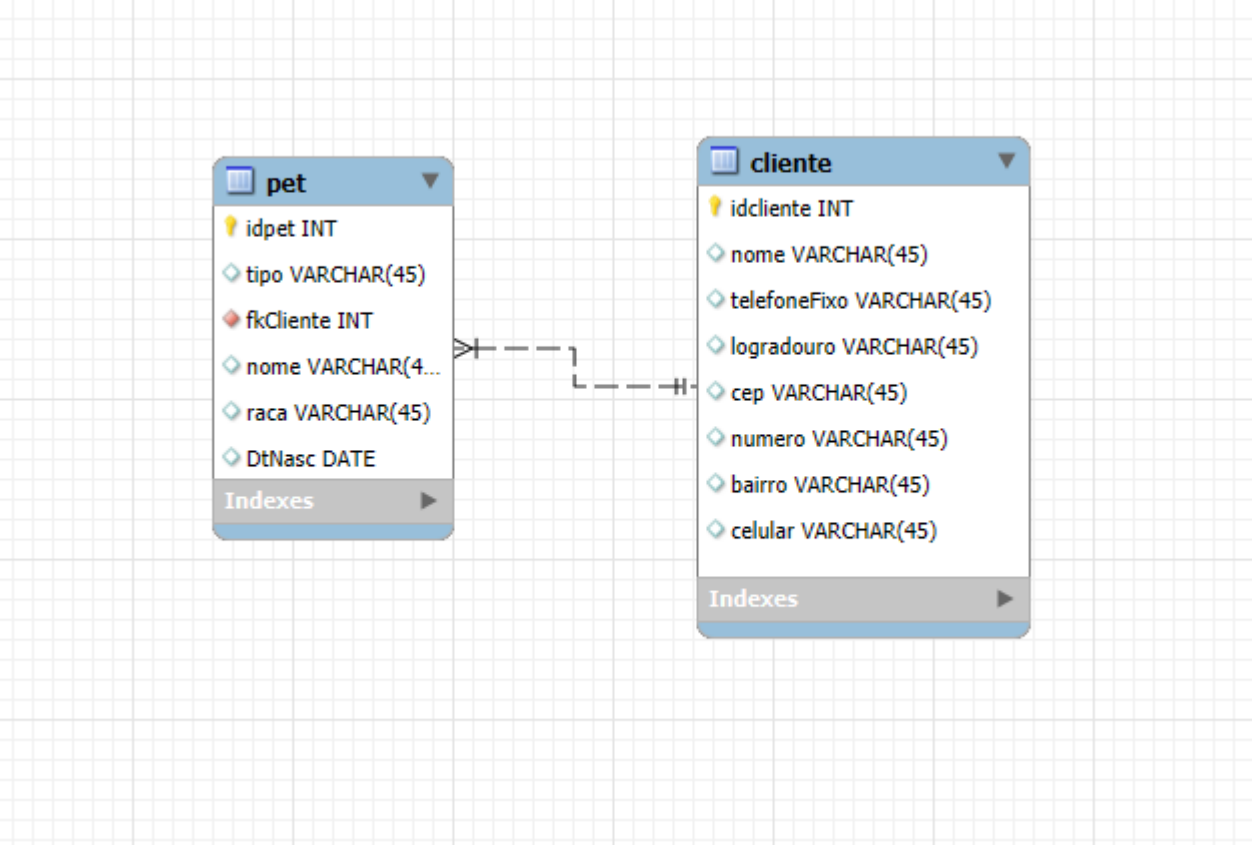
NOME	TURMA
Izael Reis de Oliveira Junior	1ADSB

EXERCÍCIOS PRÁTICA 05

EXERCÍCIO 1



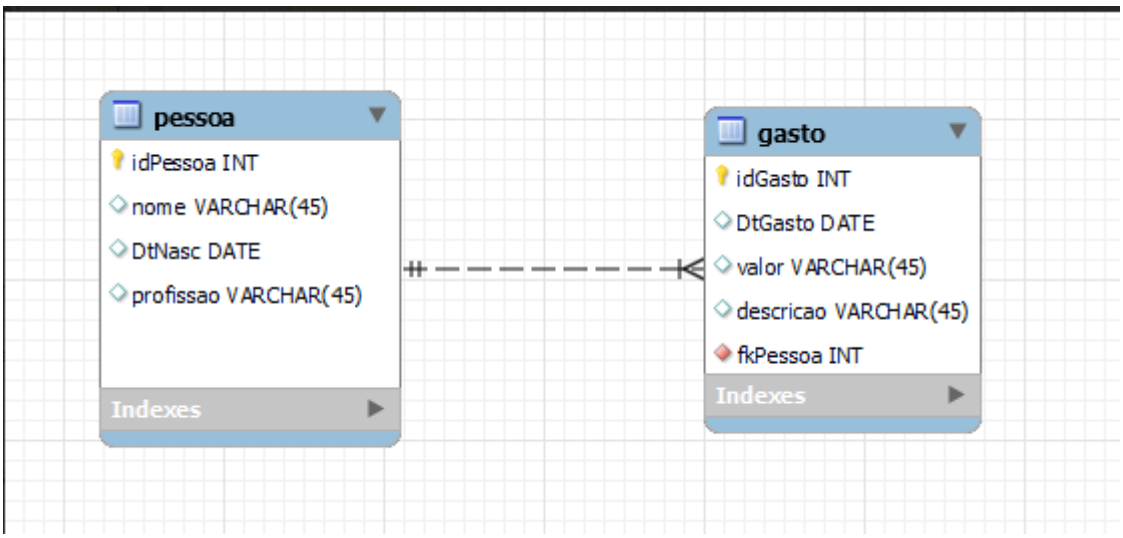

```
-- Criar um banco de dados Pet no MySQL, selecionar esse banco de dados e
-- executar as instruções relacionadas a seguir.
CREATE DATABASE Pet;
USE Pet;

-- Criar as tabelas equivalentes à modelagem.
CREATE TABLE Pet (
idpet int primary key auto_increment,
tipo varchar(45),
raca varchar(45),
nome varchar(45),
DtNasc date,
fkCliente int
) auto_increment = 101;

CREATE TABLE Cliente (
idCliente int primary key auto_increment,
```

```
nome varchar(45),
telefoneFixo varchar(20),
logradouro varchar(45),
cep varchar(45),
numero varchar(20),
bairro varchar(45),
celular varchar(20)
) auto_increment = 1;

-- Inserir dados nas tabelas, de forma que exista mais de um tipo de animal diferente,
-- e que exista algum cliente com mais de um pet cadastrado. Procure inserir pelo
-- menos 2 clientes diferentes que tenham o mesmo sobrenome.

INSERT INTO Cliente VALUES
(default, 'João Silva', '(11)1234-5678', 'Rua das Flores', '02430-598', '123', 'Vila Maria',
'(11)91234-5678'),
(default, 'Maria Silva', '(21)2345-6789', 'Av. Brasil', '42458-325', '456', 'Vila Maria', '(21)92345-
6789'),
(default, 'Carlos Dias', '(31)3456-7890', 'Rua Verde', '12582-369', '789', 'Vila Maria', '(31)93456-
7890'),
(default, 'Ana Dias', '(41)4567-8901', 'Rua das Palmeiras', '25874-654', '321', 'Vila Maria',
'(41)94567-8901');

INSERT INTO Pet VALUES
(default, 'Cachorro', 'Labrador', 'Rex', '2018-05-12', 1),
(default, 'Gato', 'Nina', 'Felix', '2019-08-23', 1),
(default, 'Cachorro', 'Poodle', 'Bobby', '2020-02-17', 2),
(default, 'Gato', 'Persa', 'Luna', '2021-10-05', 3),
(default, 'Cachorro', 'Bulldog Francês', 'Max', '2017-11-11', 4),
(default, 'Gato', 'Angorá', 'Nina', '2019-03-30', 4);

-- Exibir todos os dados de cada tabela criada, separadamente.
SELECT * FROM Cliente;

SELECT * FROM Pet;

-- Fazer os acertos da chave estrangeira, caso não tenha feito no momento da
-- criação.
ALTER TABLE pet
ADD CONSTRAINT fkClientePet FOREIGN KEY (fkCliente)
REFERENCES Cliente(idCliente);

-- Altere o tamanho da coluna nome do cliente.
ALTER TABLE Cliente
MODIFY COLUMN nome varchar(30);
```

```
-- Exibir os dados de todos os pets que são de um determinado tipo (por exemplo:
-- cachorro).
SELECT * FROM Pet
WHERE tipo = 'cachorro';

-- Exibir apenas os nomes e as datas de nascimento dos pets.
SELECT nome, dtNasc FROM Pet;

-- Exibir os dados dos pets ordenados em ordem crescente pelo nome.
SELECT * FROM Pet ORDER BY nome;

-- Exibir os dados dos clientes ordenados em ordem decrescente pelo bairro.
SELECT * FROM Cliente ORDER BY bairro DESC;

-- Exibir os dados dos pets cujo nome comece com uma determinada letra.
SELECT * FROM Pet
WHERE nome LIKE 'R%';

-- Exibir os dados dos clientes que têm o mesmo sobrenome.
SELECT * FROM Cliente
WHERE nome LIKE '% Silva';

-- Alterar o telefone de um determinado cliente.
UPDATE Cliente
SET telefoneFixo = '(11)9060-3245'
WHERE idCliente = 1;

-- Exibir os dados dos clientes para verificar se alterou.
SELECT * FROM Cliente;

-- Exibir os dados dos pets e dos seus respectivos donos.
SELECT p.idPet, p.nome as NomePet, p.raca as Raça, c.idCliente, c.nome as NomeDono,
c.celular, c.cep
FROM Pet as p JOIN Cliente as c ON p.fkCliente = c.idCliente;

-- Exibir os dados dos pets e dos seus respectivos donos, mas somente de um
-- determinado cliente.
SELECT p.idPet, p.nome as NomePet, p.raca as Raça, c.idCliente, c.nome as NomeDono,
c.celular, c.cep
FROM Pet as p JOIN Cliente as c ON p.fkCliente = c.idCliente
WHERE p.fkCliente = 1;

-- Excluir algum pet.
DELETE FROM Pet
```

```
WHERE idpet = 101;

-- Exibir os dados dos pets para verificar se excluiu.
SELECT * FROM Pet;

-- Excluir as tabelas.
DROP TABLE Pet;
DROP TABLE Cliente;
```

EXERCÍCIO 2

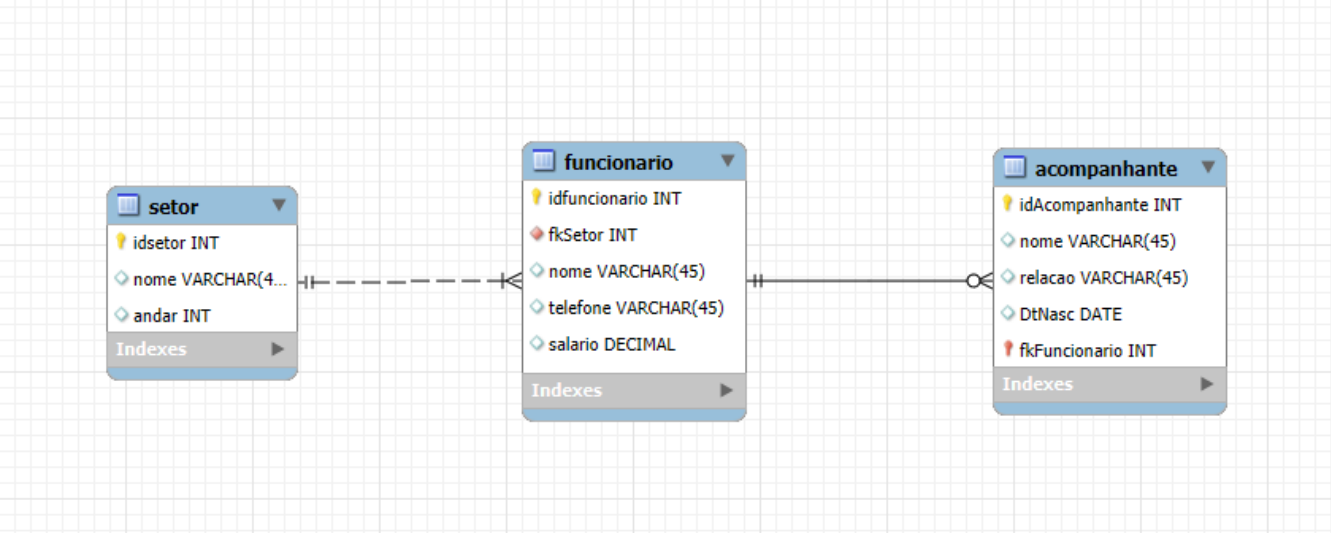



```
CREATE DATABASE economia;
USE economia;
CREATE TABLE pessoa (
idPessoa int primary key auto_increment,
nome varchar(30),
DtNasc date,
profissao varchar(30)
);


CREATE TABLE gastos (
idGastos int primary key auto_increment,
DtGasto date,
valor float,
descricao varchar(45),
fkPessoa int,
CONSTRAINT fkPessoaGastos FOREIGN KEY (fkPessoa)
REFERENCES pessoa(idPessoa)
);


-- Insira dados nas tabelas.
```

```
INSERT INTO pessoa VALUES
(default, 'João Marcos', '2000-10-10', 'Engenheiro'),
(default, 'Maria Costa', '2005-05-02', 'Médico'),
(default, 'Fabio Assunção', '1990-03-15', 'Advogado');


INSERT INTO gastos VALUES
(default, '2023-10-10', 300.50, 'Controle do video game', 1),
(default, '2024-02-20', 10, 'Biscoitos', 2),
(default, '2020-08-27', 130.80, 'Compra da semana', 3);


-- Exiba os dados de cada tabela individualmente.
SELECT * FROM pessoa;
SELECT * FROM gastos;


-- Exiba somente os dados de cada tabela, mas filtrando por algum dado da
-- tabela (por exemplo, as pessoas de alguma profissão, etc).
SELECT * FROM pessoa
WHERE profissao = 'Advogado';


SELECT * FROM gastos
WHERE DtGasto > '2024-01-01';


-- Exiba os dados das pessoas e dos seus gastos correspondentes.
SELECT p.idPessoa, p.nome, p.profissao, g.idGastos, g.valor, g.descricao, g.DtGasto
FROM pessoa as p JOIN gastos as g ON g.fkPessoa = p.idPessoa;


-- Exiba os dados de uma determinada pessoa e dos seus gastos
-- correspondentes.
SELECT p.idPessoa, p.nome, p.profissao, g.idGastos, g.valor, g.descricao, g.DtGasto
FROM pessoa as p JOIN gastos as g ON g.fkPessoa = p.idPessoa
WHERE p.idPessoa = 1;


-- Atualize valores já inseridos na tabela.
UPDATE pessoa
SET profissao = 'Atleta'
WHERE idPessoa = 1;


UPDATE gastos
```

```
SET valor = 400
WHERE idGastos = 2;


-- Exclua um ou mais registros de alguma tabela.
DELETE FROM pessoa
WHERE idPessoa = 1;
```

EXERCÍCIO 3

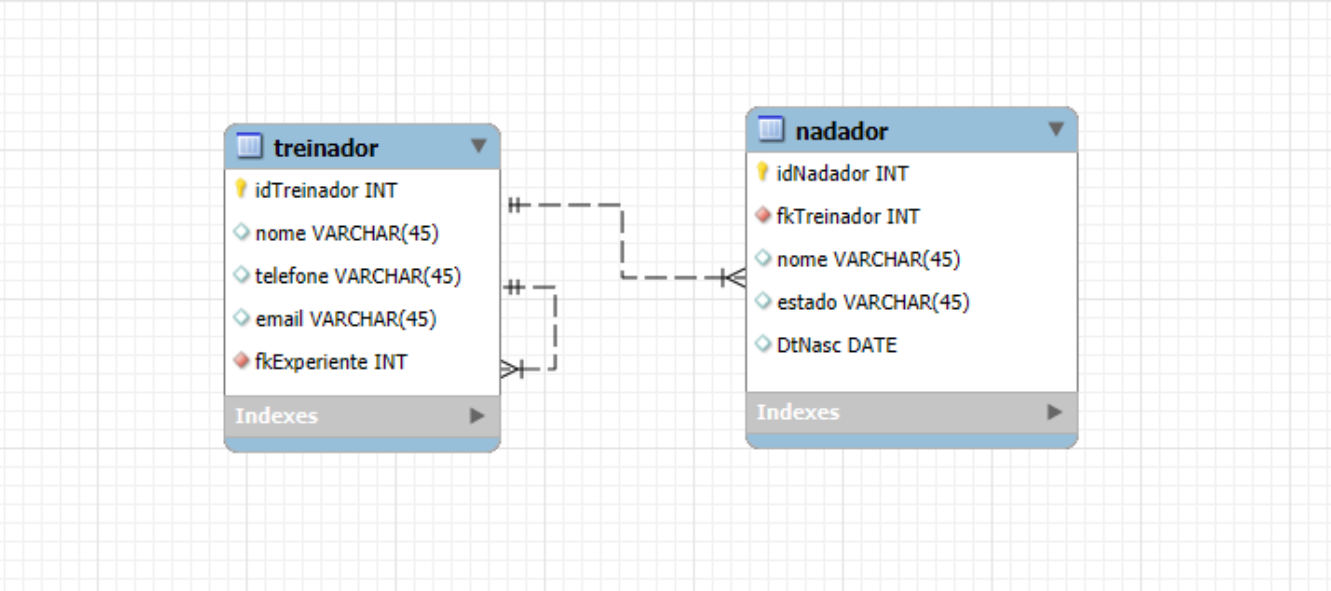



```
-- Criar um banco de dados chamado PraticaFuncionario.
CREATE DATABASE PraticaFuncionario;

-- Selecionar esse banco de dados.
USE PraticaFuncionario;

-- Criar as tabelas correspondentes à sua modelagem.
CREATE TABLE setor (
idSetor int primary key auto_increment,
nome varchar(45),
andar int
);

CREATE TABLE funcionario (
idFunc int primary key auto_increment,
nome varchar(45),
telefone varchar(20),
salario decimal,
fkSetor int
);
ALTER TABLE funcionario
ADD CONSTRAINT chkSalario CHECK (salario > 0);
```

```
CREATE TABLE acompanhante (
idAcomp int,
fkFunc int,
nome varchar(45),
relacao varchar(45),
DtNasc date,
primary key(idAcomp, fkFunc)
);

-- Inserir dados nas tabelas, de forma que exista mais de um funcionário em cada
-- setor cadastrado.
INSERT INTO setor VALUES
(default, 'Financeiro', 1),
(default, 'Recursos Humanos', 2),
(default, 'Tecnologia da Informação', 3);

INSERT INTO funcionario VALUES
(default, 'João Silva', '5555-1234', 3500, 1),
(default, 'Maria Oliveira', '5555-5678', 4200.00, 1),
(default, 'Carlos Pereira', '5555-9101', 3800.00, 2),
(default, 'Ana Souza', '5555-1122', 4100.00, 2),
(default, 'Luís Costa', '5555-3344', 5000.00, 3),
(default, 'Fernanda Lima', '5555-5566', 5300.00, 3);

INSERT INTO acompanhante VALUES
(1, 1, 'Pedro Silva', 'Filho', '2012-05-10'),
(2, 2, 'Laura Oliveira', 'Esposa', '1985-08-15'),
(3, 3, 'Clara Pereira', 'Filha', '2010-11-20'),
(4, 4, 'Carlos Souza', 'Marido', '1983-02-18'),
(5, 5, 'Sofia Costa', 'Esposa', '1990-04-25'),
(6, 6, 'Marcos Lima', 'Filho', '2015-09-12');

-- Exibir todos os dados de cada tabela criada, separadamente.
SELECT * FROM setor;
SELECT * FROM funcionario;
SELECT * FROM acompanhante;

-- Fazer os acertos da chave estrangeira, caso não tenha feito no momento da
-- criação.
ALTER TABLE funcionario
ADD CONSTRAINT fkSetorFunc FOREIGN KEY (fkSetor)
REFERENCES setor(idSetor);

ALTER TABLE acompanhante
```

```
ADD CONSTRAINT fkFuncAcomp FOREIGN KEY (fkFunc)
REFERENCES funcionario(idFunc);

-- Exibir os dados dos setores e dos seus respectivos funcionários.
SELECT s.idSetor, s.nome as NomeSetor, s.andar, f.idFunc, f.nome as NomeFunc, f.telefone,
f.salario
FROM setor as s JOIN funcionario as f ON f.fkSetor = s.idSetor;

-- Exibir os dados de um determinado setor (informar o nome do setor na
-- consulta) e dos seus respectivos funcionários.
SELECT s.idSetor, s.nome as NomeSetor, s.andar, f.idFunc, f.nome as NomeFunc, f.telefone,
f.salario
FROM setor as s JOIN funcionario as f ON f.fkSetor = s.idSetor
WHERE s.nome = 'Recursos Humanos';

-- Exibir os dados dos funcionários e de seus acompanhantes.
SELECT f.idFunc, f.nome as NomeFunc, f.telefone, f.salario, a.idAcomp, a.nome as
NomeAcomp, a.relacao, a.DtNasc
FROM funcionario as f JOIN acompanhante as a ON a.fkFunc = f.idFunc;

-- Exibir os dados de apenas um funcionário (informar o nome do funcionário) e
-- os dados de seus acompanhantes.
SELECT f.idFunc, f.nome as NomeFunc, f.telefone, f.salario, a.idAcomp, a.nome as
NomeAcomp, a.relacao, a.DtNasc
FROM funcionario as f JOIN acompanhante as a ON a.fkFunc = f.idFunc
WHERE f.nome = 'João Silva';

-- Exibir os dados dos funcionários, dos setores em que trabalham e dos seus
-- acompanhantes
SELECT s.idSetor, s.nome as NomeSetor, s.andar, f.idFunc, f.nome as NomeFunc, f.telefone,
f.salario, a.idAcomp, a.nome as NomeAcomp, a.relacao, a.DtNasc
FROM funcionario as f JOIN acompanhante as a ON a.fkFunc = f.idFunc JOIN setor as s ON
s.idSetor = f.fkSetor;
```

EXERCÍCIO 4

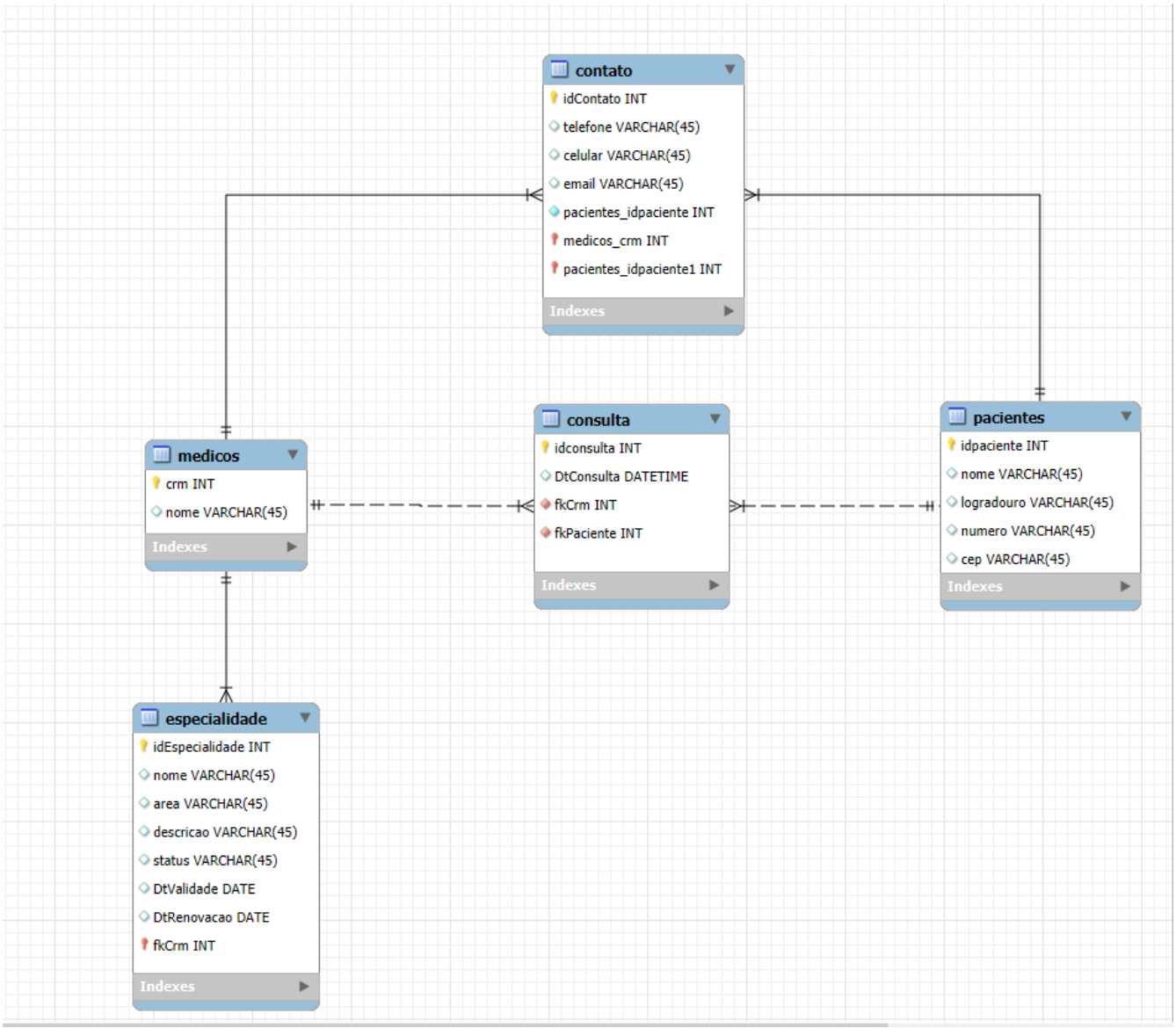



```
-- a) Criar um banco de dados chamado Treinador.
CREATE DATABASE Treinador;

-- b) Selecionar esse banco de dados.
USE Treinador;

-- c) Criar as tabelas correspondentes à sua modelagem.
CREATE TABLE professor (
idProfessor int primary key auto_increment,
nome varchar(45),
telefone varchar(45),
email varchar(50),
fkExperiente int
)auto_increment = 10;

CREATE TABLE nadador (
idNadador int primary key auto_increment,
nome varchar(45),
estado varchar(45),
DtNasc date,
fkProfessor int
)auto_increment = 100;

-- d) Inserir dados nas tabelas, de forma que exista mais de um nadador para algum
-- treinador, e mais de um treinador sendo orientado por algum treinador mais
-- experiente.
INSERT INTO professor VALUES
(default, 'Carlos Andrade', '555-1234', 'carlos.andrade@escola.com', NULL),
(default, 'Fernanda Silva', '555-5678', 'fernanda.silva@escola.com', 10),
(default, 'Paulo Souza', '555-9101', 'paulo.souza@escola.com', 10),
```

```
(default, 'Mariana Lima', '555-1122', 'mariana.lima@escola.com', 11);

INSERT INTO nadador VALUES
(default, 'Lucas Costa', 'SP', '2005-03-15', 11),
(default, 'Carla Santos', 'SP', '2007-06-20', 11),
(default, 'Pedro Souza', 'RJ', '2004-12-10', 12),
(default, 'Mariana Almeida', 'MG', '2006-09-25', 12),
(default, 'Ana Martins', 'RS', '2005-11-05', 13);

-- e) Exibir todos os dados de cada tabela criada, separadamente.
SELECT * FROM professor;
SELECT * FROM nadador;

-- f) Fazer os acertos da chave estrangeira, caso não tenha feito no momento da criação
-- das tabelas.
ALTER TABLE nadador
ADD CONSTRAINT fkProfNad FOREIGN KEY (fkProfessor)
REFERENCES professor(idProfessor);

ALTER TABLE professor
ADD CONSTRAINT fkExpProf FOREIGN KEY (fkExperiente)
REFERENCES professor(idProfessor);

-- g) Exibir os dados dos treinadores e os dados de seus respectivos nadadores.
SELECT n.idNadador, n.nome as NomeNadador, n.estado, n.DtNasc, p.idProfessor, p.nome as
NomeProfessor, p.telefone, p.email
FROM nadador as n JOIN professor as p ON n.fkProfessor = p.idProfessor;

-- h) Exibir os dados de um determinado treinador (informar o nome do treinador na
-- consulta) e os dados de seus respectivos nadadores.
SELECT n.idNadador, n.nome as NomeNadador, n.estado, n.DtNasc, p.idProfessor, p.nome as
NomeProfessor, p.telefone, p.email
FROM nadador as n JOIN professor as p ON n.fkProfessor = p.idProfessor
WHERE p.nome = 'Fernanda Silva';

-- i) Exibir os dados dos treinadores e os dados dos respectivos treinadores
-- orientadores.
SELECT p.idProfessor, p.nome as NomeProfessor, p.telefone, p.email, p.fkExperiente, e.nome
as NomeOrientador, e.telefone, e.email
FROM professor as p JOIN professor as e ON p.fkExperiente = e.idProfessor;

-- j) Exibir os dados dos treinadores e os dados dos respectivos treinadores
-- orientadores, porém somente de um determinado treinador orientador (informar o
-- nome do treinador na consulta).
```

```
SELECT p.idProfessor, p.nome as NomeProfessor, p.telefone, p.email, p.fkExperiente, e.nome
as NomeOrientador, e.telefone, e.email
FROM professor as p JOIN professor as e ON p.fkExperiente = e.idProfessor
WHERE e.nome = 'Fernanda Silva';

-- l) Exibir os dados dos treinadores, os dados dos respectivos nadadores e os dados
-- dos respectivos treinadores orientadores.
SELECT p.idProfessor, p.nome as NomeProfessor, p.telefone, p.email, p.fkExperiente, e.nome
as NomeOrientador, e.telefone, e.email, n.idNadador, n.nome as NomeNadador, n.estado,
n.DtNasc
FROM professor as p JOIN professor as e ON p.fkExperiente = e.idProfessor JOIN nadador as
n ON n.fkProfessor = p.idProfessor;

-- m) Exibir os dados de um treinador (informar o seu nome na consulta), os dados dos
-- respectivos nadadores e os dados do seu treinador orientador.
SELECT p.idProfessor, p.nome as NomeProfessor, p.telefone, p.email, p.fkExperiente, e.nome
as NomeOrientador, e.telefone, e.email, n.idNadador, n.nome as NomeNadador, n.estado,
n.DtNasc
FROM professor as p JOIN professor as e ON p.fkExperiente = e.idProfessor JOIN nadador as
n ON n.fkProfessor = p.idProfessor
WHERE p.nome = 'Paulo Souza';
```

EXERCÍCIO 5


contato
idContato INT
telefone VARCHAR(45)
celular VARCHAR(45)
email VARCHAR(45)
pacientes_idpaciente INT
medicos_crm INT
pacientes_idpaciente1 INT
Indexes
medicos
crm INT
nome VARCHAR(45)
Indexes
consulta
idconsulta INT
DtConsulta DATETIME
fkCrm INT
fkPaciente INT
Indexes
pacientes
idpaciente INT
nome VARCHAR(45)
logradouro VARCHAR(45)
numero VARCHAR(45)
cep VARCHAR(45)
Indexes
especialidade
idEspecialidade INT
nome VARCHAR(45)
area VARCHAR(45)
descricao VARCHAR(45)
status VARCHAR(45)
DtValidade DATE
DtRenovacao DATE
fkCrm INT
Indexes