Exercício 3

Integrante: Wesley Bernardes (020321)

1) Defina os seguintes termos: esquema de relação, domínio, instância de relação, cardinalidade da relação e grau da relação.

- **Esquema de relação:** é a estrutura que define a organização dos dados em uma relação (tabela). Especifica os atributos (colunas) e seus tipos de dados.
- **Domínio:** é o conjunto de valores possíveis que um atributo pode assumir. Exemplo: um atributo `idade` pode ter um domínio de valores entre 0 e 120.
- **Instância de relação:** é o conjunto de tuplas (linhas) existentes em uma relação (tabela) em um determinado momento. A instância muda conforme os dados são inseridos, modificados ou removidos.
- **Cardinalidade da relação:** é o número de tuplas (linhas) presentes em uma relação (tabela) em um determinado momento.
- **Grau da relação:** é o número de atributos (colunas) que compõem a relação.

2) Qual é a diferença entre uma chave candidata e a chave primária de determinada relação? O que é uma superchave?

- **Chave candidata:** é o conjunto mínimo de atributos que identifica unicamente cada tupla (linha) em uma relação. Pode haver mais de uma chave candidata.
- **Chave primária:** é a chave escolhida entre as chaves candidatas para ser a principal identificação da tupla. Apenas uma chave primária é definida por tabela.
- **Superchave:** é qualquer conjunto de atributos que identifica unicamente uma tupla, podendo incluir atributos

redundantes.

Exemplo:

```
CREATE TABLE alunos (
  matricula INT PRIMARKEY IDENTITY,
  cpf VARCHAR(11) NOT NULL UNIQUE,
  nome VARCHAR(100)
);
```

- `matricula` e `cpf` são **chaves candidatas**.
- `matricula` foi escolhida como **chave primária**.
- `{matricula, cpf, nome}` é uma **superchave** (pois inclui atributos redundantes).

3) Dê um exemplo de atributo (ou conjunto de atributos) que você pode deduzir que não é uma chave candidata considerando uma relação hipotética Professores.

Na relação professores:

```
CREATE TABLE professores (
  id INT PRIMARY KEY,
  nome VARCHAR(100),
  departamento VARCHAR(50)
);
```

- O atributo `nome` não pode ser uma chave candidata porque pode haver professores com nomes repetidos.
- O atributo `departamento` também não pode ser uma chave candidata, pois vários professores podem pertencer ao mesmo departamento.

4) Há algum exemplo de atributo (ou conjunto de atributos) que você pode deduzir que é uma chave candidata (Professor)?

Na relação professores, um exemplo de chave candidata seria:

```
CREATE TABLE Professores (
  id INT NOT NULL UNIQUE,
  cpf VARCHAR(11) NOT NULL UNIQUE,
  nome VARCHAR(100),
  PRIMARY KEY (id)
);
```

- `id` e `cpf` são chaves candidatas, pois identificam unicamente cada professor.
- `id` foi escolhida como chave primária.

5) O que é uma restrição de chave estrangeira? Por que tais restrições são importantes? O que é integridade referencial?

- **Chave estrangeira:** é um atributo (ou conjunto de atributos) em uma tabela que faz referência à chave primária de outra tabela. Garante a relação entre os dados.
- **Importância:** evita dados inconsistentes e garante que os valores de referência existam na tabela associada.
- **Integridade referencial:** conjunto de regras que assegura que a chave estrangeira sempre aponta para uma chave primária válida.

Exemplo:

```
CREATE TABLE departamentos (
  id INT PRIMARY KEY,
  nome VARCHAR(50)
);

CREATE TABLE professores (
  id INT PRIMARY KEY,
  nome VARCHAR(100),
  departamento_id INT,
  FOREIGN KEY (departamento_id) REFERENCES Departamentos(id)
);
```

A chave estrangeira `departamento_id` garante que um professor só pode estar associado a um departamento existente.

6) Qual construção da SQL permite a definição de uma relação?

A construção `CREATE TABLE` permite definir uma relação (tabela).

Exemplo:

```
CREATE TABLE Alunos (
  id INT PRIMARY KEY,
  nome VARCHAR(100),
  email VARCHAR(100) UNIQUE
);
```

7) Quais construções permitem a modificação de instâncias de relação?

- `INSERT INTO` (para inserir dados)
- `UPDATE` (para modificar dados)
- `DELETE` (para remover dados)

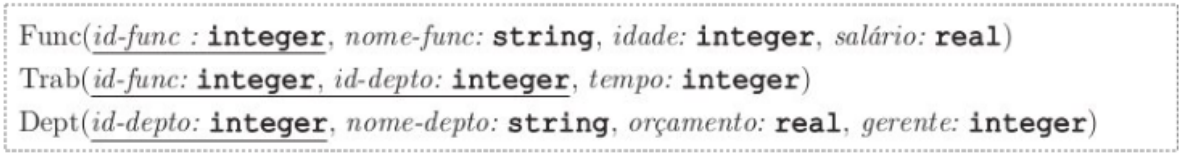
Exemplo:

```
INSERT INTO Alunos (id, nome, email) VALUES (1, 'John Doe', 'john-doe@email.com');

UPDATE Alunos SET nome = 'John Doe' WHERE id = 1;

DELETE FROM Alunos WHERE id = 1;
```

8) Responda sucintamente cada uma das perguntas a seguir. As perguntas são baseadas no seguinte esquema relacional:



a) Dê um exemplo de restrição de chave estrangeira que envolva a relação Dept. Quais são as opções para garantir essas restrições quando um usuário tentar excluir uma tupla de Dept?

```
FOREIGN KEY (id-depto) REFERENCES Dept(id-depto) ON DELETE CASCADE;
```

- `ON DELETE CASCADE`: se um departamento for excluído, todas as tuplas de `Trab` que fazem referência a esse departamento serão automaticamente excluídas.
- `ON DELETE SET NULL`: se um departamento for excluído, o campo `id-depto` em `Trab` será definido como `NULL`.
- `ON DELETE RESTRICT` (ou `NO ACTION`): a exclusão do departamento será impedida caso existam registros em `Trab` referenciando-o.

b) Escreva as instruções SQL exigidas para criar as relações anteriores, incluindo as versões apropriadas de todas as restrições de integridade de chave primária e estrangeira.

```
CREATE TABLE Func (
    id-func INTEGER PRIMARY KEY,
    nome-func VARCHAR(100) NOT NULL,
    idade INTEGER NOT NULL,
    salario REAL NOT NULL
);

CREATE TABLE Dept (
    id-depto INTEGER PRIMARY KEY,
    nome-depto VARCHAR(100) NOT NULL,
    orçamento REAL NOT NULL,
    gerente INTEGER NOT NULL,
    FOREIGN KEY (gerente) REFERENCES Func(id-func) ON DELETE SET NULL
);

CREATE TABLE Trab (
    id-func INTEGER,
    id-depto INTEGER,
    tempo INTEGER NOT NULL,
    PRIMARY KEY (id-func, id-depto),
```

```
    FOREIGN KEY (id-func) REFERENCES Func(id-func) ON DELETE CASCADE,
    FOREIGN KEY (id-depto) REFERENCES Dept(id-depto) ON DELETE CASCAD
);
```

- `Func`: `id-func` é a **chave primária**.
- `Dept`: `id-depto` é a **chave primária**, `gerente` é uma **chave estrangeira** referenciando `Func(id-func)`, garantindo que todo departamento tenha um gerente.
- `Trab`: `id-func` e `id-depto` formam a **chave primária composta** e são **chaves estrangeiras** referenciando `Func` e `Dept`.

c) Defina a relação Dept em SQL, de modo que seja garantido que todo departamento tenha um gerente.

```
CREATE TABLE Dept (
    id-depto INTEGER PRIMARY KEY,
    nome-depto VARCHAR(100) NOT NULL,
    orçamento REAL NOT NULL,
    gerente INTEGER NOT NULL,
    FOREIGN KEY (gerente) REFERENCES Func(id-func) ON DELETE SET NULL
);
```

d) Escreva uma instrução SQL para adicionar João Ferreira como funcionário, com id-func = 101, idade = 32 e salário = 5000,00.

```
INSERT INTO Func (id-func, nome-func, idade, salario)
VALUES (101, 'João Ferreira', 32, 5000.00);
```

e) Escreva uma instrução SQL para dar a cada funcionário um aumento de 10%.

```
UPDATE Func SET salario = salario * 1.10;
```

f) Escreva uma instrução SQL para excluir o departamento Brinquedos. Dadas as restrições de integridade referencial que você escolheu para esse esquema, explique o que acontece quando essa instrução é executada.

```
DELETE FROM Dept WHERE nome-depto = 'Brinquedos';
```

O comportamento depende da restrição `ON DELETE` definida para a chave estrangeira `id-depto` na tabela `Trab`:

- **Se estiver definido** `ON DELETE CASCADE`, todas as tuplas em `Trab` relacionadas ao departamento "Brinquedos" também serão excluídas.
- **Se estiver** `ON DELETE SET NULL`, o `id-depto` nas tuplas de `Trab` associadas será definido como `NULL`.
- **Se estiver** `ON DELETE RESTRICT`, a exclusão será impedida caso existam registros em `Trab` referenciando esse departamento.