Roteiro

Considere um banco de dados MongoDB com duas collection cujas interfaces são:

```
interface User {
   _id: ObjectId;
   name: string;
   address: {
       street: string;
       number: number;
       city: string;
       state: string;
       country: string;
   };
   birthDate: Date;
   pets: string[];
}

interface Pet {
   _id: ObjectId;
   name: string;
   bride: string;
   birthDate: Date;
   type: 'dog' | 'cat' | 'parrot';
   weight: number;
}
```

1. Usando Mongo Query Language (MQL), encontre todos os usuários que vivem no pais "**Brasil**", utilizando a função **find**.
2. Usando MQL, retorne uma lista de "**User**" que contenha apenas os campos "**Nome**", "**Cidade**" e "**Pets**". *Dica: use a função **aggregate** e o estágio **project**.*
3. Usando MQL, retorne uma lista de pets incluindo os campos: **Nome do pet**, **Raça** e **Nome do dono**. *Dica: use a função **aggregate** e os estágios **lookup** e **project**.*
4. Usando Python e a biblioteca pymongo, liste todos os gatos utilizando a função **find**, e retorne uma lista de dicionários que possue os campos "**Nome**", "**Raça**" e "**Peso**", sendo que peso deverá incluir o sufixo "kg" (ex.: "10 kg").

Considere um banco de dados PostgreSQL com duas tables cujas interfaces são:

```
interface User {
   id: UUID; @primary
   name: String;
   street: String;
   number: Int;
   city: String;
   state: String;
```

```
   country: String;
   birthDate: Date;
}

interface UserPet {
   userId: UUID;
   petId: UUID;

   @primary([userId, petId])
}

interface Pet {
   id: UUID; @primary
   name: String;
   bride: String;
   birthDate: Date;
   type: Enum('dog', 'cat', 'parrot');
   weight: Int;
}
```

5. Usando Structured Query Language (SQL) liste todos os Cachorros.
6. Usando SQL, Delete todos os animais do tipo Papagaio do usuário cujo id é **"1571fbb8-50dc-467f-9c90-1687b2ed1aa6"**
7. Usando SQL, Encontre todos os animais cujo dono tenha o id **"1571fbb8-50dc-467f-9c90-1687b2ed1aa6"**