**PONTIFÍCIA UNIVERSIDADE CATÓLICA DO PARANÁ**

**CIÊNCIA DA COMPUTAÇÃO**

ERICK MAESTRI DE SOUZA

**Desenvolvimento de um Sistema de Cadastro e Gerenciamento de Informações em Clínicas Veterinárias utilizando SQLAlchemy e SQLite**

CURITIBA

2024

ERICK MAESTRI DE SOUZA

**Desenvolvimento de um Sistema de Cadastro e Gerenciamento de Informações em Clínicas Veterinárias utilizando SQLAlchemy e SQLite**

Trabalho Discente Efetivo realizado ao curso de Ciência da Computação da Pontifícia Universidade Católica do Paraná como requisito parcial da disciplina de Arquitetura de Banco de Dados.

Orientador: Prof. Ms. Rodrigo da Silva do Nascimento

CURITIBA

2024

SUMÁRIO

1. CAPA1
2. FOLHA DE ROSTO2
3. SUMÁRIO3
4. LISTA DE FIGURAS4
5. RESUMO5
6. ABSTRACT6
7. INTRODUÇÃO7
8. DESENVOLVIMENTO DO SISTEMA8

8.1 Modelo Textual8

8.2 Modelo Conceitual9

8.3 Descrição do Sistema10

8.4 Implementação do Banco de Dados10

8.5 Funcionalidades e CRUD11

1. ANÁLISE DOS RESULTADOS12
2. CONCLUSÂO14
3. REFERÊNCIAS15
4. APÊNDICES16

APÊNDICE A - Código Fonte17

**LISTA DE FIGURAS**

Figura 1 – Modelo textual utilizado para projetar o modelo conceitual8

Figura 2 – Modelo conceitual no qual o sistema foi baseado para operar 9

**RESUMO**

O trabalho apresenta o desenvolvimento de um sistema de gerenciamento para uma clínica veterinária, focado inicialmente na administração de dados de donos, com aplicação específica a pessoas físicas. Esse sistema é resultado de um estudo aplicado que visa criar uma solução prática para a realização de operações CRUD (criar, ler, atualizar e deletar) sobre a entidade "dono". O objetivo principal do projeto é fornecer uma ferramenta que facilite o controle de informações importantes, como nome, endereço e números de contato dos proprietários de animais que frequentam a clínica. A metodologia utilizada inclui a implementação de um banco de dados SQLite, integrado ao sistema por meio de um mapeamento objeto-relacional (ORM) utilizando a biblioteca SQLAlchemy. Como resultado, foi desenvolvido um sistema funcional capaz de gerenciar eficientemente os dados dos donos, permitindo o cadastro de novos registros, a consulta por CPF, a atualização de informações e a exclusão de registros. Embora o CRUD atualmente esteja limitado à entidade "dono", há um plano de expansão para incluir animais, veterinários e espécies. Conclui-se que o sistema oferece uma base robusta e prática, com potencial para aprimorar a gestão de dados na clínica veterinária.

**Palavras-chave:** Gerenciamento Veterinário. CRUD. SQLite. SQLAlchemy.

**ABSTRACT**

The project presents the development of a management system for a veterinary clinic, initially focused on the administration of owner data, specifically for individuals. This system is the result of an applied study aimed at creating a practical solution for performing CRUD operations (create, read, update, and delete) on the "owner" entity. The main objective of the project is to provide a tool that facilitates the management of important information, such as the names, addresses, and contact numbers of animal owners who visit the clinic. The methodology used includes the implementation of an SQLite database, integrated into the system through object-relational mapping (ORM) using the SQLAlchemy library. As a result, a functional system was developed that efficiently manages owner data, allowing for the registration of new records, CPF-based queries, information updates, and record deletions. Although CRUD is currently limited to the "owner" entity, there is a plan to expand to include animals, veterinarians, and species. It is concluded that the system provides a robust and practical foundation with the potential to enhance data management in the veterinary clinic.

**Keywords:** Veterinary Management. CRUD. SQLite. SQLAlchemy.

**INTRODUÇÃO**

O presente trabalho tem como finalidade desenvolver um sistema de gerenciamento para uma clínica veterinária, com foco inicial na administração dos dados dos donos dos animais atendidos. A importância desse sistema consiste na necessidade de uma solução prática e eficiente para a gestão das informações dos clientes, que inclui dados pessoais e de contato. A clínica veterinária, como entidade que lida com um grande volume de dados de clientes e seus animais, requer um sistema que permita o gerenciamento eficaz e organizado desses dados.

A necessidade de um sistema de gerenciamento eficiente é evidente, dado o crescimento constante das clínicas veterinárias e a complexidade na administração dos dados dos clientes e dos animais. O projeto busca preencher essa lacuna ao oferecer uma ferramenta que facilite operações essenciais como a criação, leitura, atualização e exclusão de registros de donos, especificamente pessoas físicas, através de uma interface simples e funcional.

O objetivo geral do projeto é implementar um sistema que permita a administração eficiente dos dados dos donos dos animais, com a possibilidade de realizar operações CRUD (criar, ler, atualizar e deletar). Especificamente, o sistema se concentrará na implementação de um banco de dados SQLite integrado com SQLAlchemy para gerenciar as informações dos donos e garantir a integridade e acessibilidade dos dados.

O documento está estruturado da seguinte forma: na seção de Desenvolvimento do Sistema, estão descritos os detalhes técnicos e a implementação do banco de dados. A Análise dos Resultados discutirá o desempenho e a eficácia do sistema desenvolvido. Finalmente, a Conclusão apresentará as principais conclusões e possíveis direções para futuras melhorias.

### **DESENVOLVIMENTO DO SISTEMA**

**8.1 Modelo Textual**

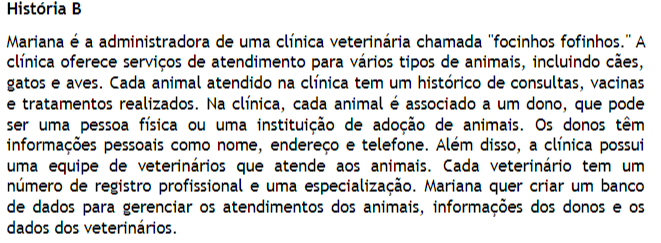


Figura 1 - Modelo textual na íntegra

O modelo textual utilizado para a construção do sistema de gestão da clínica veterinária "Focinhos Fofinhos" foi baseado em um cenário fictício, que descreve as necessidades e interações entre os elementos fundamentais da clínica. A clínica, administrada por Mariana, oferece diversos serviços para animais de diferentes espécies, como cães, gatos e aves.

Cada animal que recebe atendimento na clínica possui um histórico detalhado de consultas, vacinas e tratamentos realizados. Esses animais estão vinculados a seus respectivos donos, que podem ser tanto pessoas físicas quanto instituições de adoção de animais. As informações pessoais dos donos, como nome, endereço e telefone, são registradas no sistema.

A clínica conta com uma equipe de veterinários, cada um com seu número de registro profissional e especialização, responsáveis por realizar os atendimentos. O objetivo de Mariana, como administradora, é criar um banco de dados que permita o gerenciamento eficiente dessas informações, contemplando o controle de atendimentos, dados dos donos e veterinários.

Esse modelo textual orienta a estrutura do sistema a ser desenvolvido, garantindo que as principais relações e atributos necessários estejam claramente definidos para a criação de um banco de dados robusto e funcional.

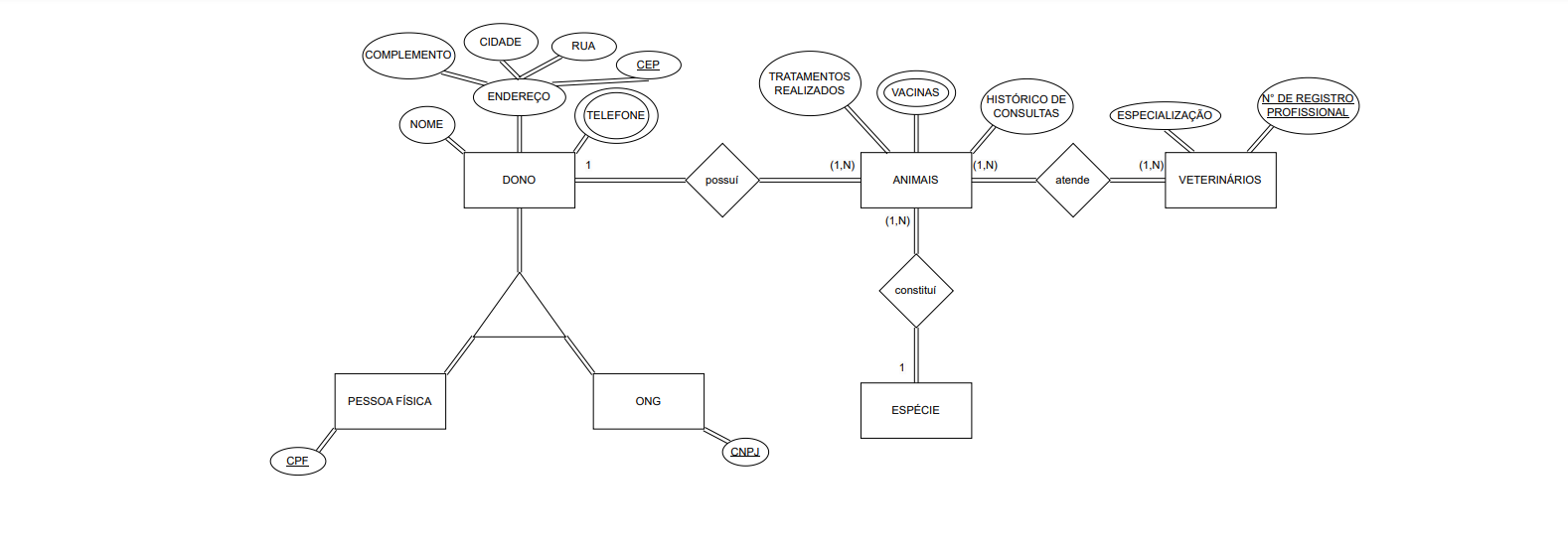
**8.2** **Modelo Conceitual**

Figura 2 - Modelo conceitual

O modelo conceitual desenvolvido para a clínica veterinária "Focinhos Fofinhos" baseia-se na definição das principais entidades e suas interações. As entidades centrais identificadas são Donos, Animais, Espécie e Veterinários. Esse modelo visa organizar as informações relevantes de forma estruturada, facilitando a implementação e o gerenciamento dos dados no sistema.

A entidade Donos possui duas especializações: ONG e Pessoa Física, permitindo a distinção entre instituições de adoção e indivíduos que possuem animais. Cada dono é responsável por um ou mais animais, e suas informações incluem dados pessoais, como nome, endereço e telefone.

A entidade Animais está relacionada diretamente a um dono e a uma Espécie específica, categorizando os tipos de animais atendidos, como cães, gatos e aves. Os dados dos animais também incluem um histórico detalhado de atendimentos, consultas e tratamentos realizados.

A entidade Veterinários armazena as informações dos profissionais responsáveis pelo atendimento aos animais, contendo atributos como o número de registro profissional e a especialização de cada veterinário. Esses profissionais podem realizar diversas consultas, e suas especializações garantem o atendimento adequado conforme a necessidade de cada espécie.

O modelo conceitual foi representado utilizando diagramas de entidade-relacionamento (ER), destacando as interações entre as entidades e garantindo que os relacionamentos estejam devidamente mapeados, facilitando a transição para o banco de dados e a implementação do sistema.

#### **8.3 Descrição do Sistema**

O sistema foi desenvolvido com o objetivo de gerenciar os dados de uma clínica veterinária, facilitando o cadastro, consulta e atualização de informações relacionadas a proprietários de animais e seus respectivos pets. O foco principal é o cadastro de **Pessoas Físicas**, que herdam da entidade "Dono" os atributos básicos, como nome e endereço. Além disso, o sistema permite o armazenamento de dados como o CPF e múltiplos números de telefone de cada dono, sendo os números representados por uma tabela associada, configurada como um atributo multivalorado.

O sistema também é capaz de gerenciar as informações dos animais pertencentes a esses donos, incluindo detalhes como histórico de consultas, tratamentos realizados e as vacinas aplicadas. Essas informações são armazenadas em tabelas específicas, como a tabela "Vacinas", que representa um atributo multivalorado, permitindo a associação de várias vacinas a um único animal.

Embora o objetivo do sistema não seja especificamente acompanhar os históricos de consultas e tratamentos realizados nos animais, ele foi estruturado para registrar e manter esses dados, além de possibilitar a inclusão e gerenciamento das vacinas administradas. Todas essas informações são associadas aos veterinários responsáveis, criando um elo entre os animais tratados e os profissionais da clínica.

#### **8.4 Implementação do Banco de Dados**

O banco de dados foi implementado utilizando a biblioteca **SQLAlchemy** para realizar o mapeamento objeto-relacional (ORM), facilitando a interação entre o código Python e o banco de dados relacional, permitindo a manipulação de dados como se fossem objetos Python. Além disso, a biblioteca **os**, foi utilizada para garantir que o banco de dados fosse limpo sempre que o sistema fosse iniciado.

A estrutura do banco de dados é composta por várias tabelas que representam as entidades principais do sistema. A tabela "Dono" é uma das mais importantes, pois armazena as informações básicas sobre os proprietários dos animais. Além disso, ela serve como uma classe pai para "Pessoa Física" e "ONG", entidades que herdam seus atributos e adicionam campos específicos, como CPF (para pessoa física) e CNPJ (para ONGs).

A tabela "Telefone" está associada à tabela "Dono" e foi criada para armazenar múltiplos números de telefone, representando um atributo multivalorado. De forma similar, a tabela "Vacinas" está associada à tabela "Animal", permitindo que várias vacinas sejam registradas para um único animal.

Outras tabelas incluem "Animal", que armazena dados como espécie e dono do animal, e "Veterinario", que registra os profissionais da clínica. A relação entre veterinários e animais é tratada por meio de uma tabela de associação chamada "veterinario\_Animais", que possibilita uma relação muitos-para-muitos entre essas duas entidades.

#### **8.5 Funcionalidades e CRUD**

O sistema implementa operações básicas de CRUD (Create, Read, Update, Delete) apenas para a entidade "Pessoa Física". As principais funcionalidades desenvolvidas são:

* Create: Permite o cadastro de uma nova pessoa física, inserindo dados como nome, CPF, endereço e números de telefone. Cada telefone é armazenado como um registro separado na tabela "Telefone", garantindo que uma pessoa possa ter múltiplos números associados.
* Read: O sistema oferece a funcionalidade de consulta de pessoas físicas cadastradas com base no CPF. Durante a consulta, o sistema exibe todas as informações associadas à pessoa, incluindo seus números de telefone, se houver.
* Update: O sistema permite a atualização das informações de uma pessoa física, incluindo nome, endereço e números de telefone. Caso um número de telefone precise ser alterado, o usuário pode selecionar qual telefone modificar.
* Delete: Também é possível excluir registros de pessoas físicas, removendo todos os dados associados, incluindo os números de telefone registrados.

**ANÁLISE DE RESULTADOS**

Durante o desenvolvimento do sistema para gerenciamento de dados de uma clínica veterinária, foi possível observar como a implementação do modelo relacional contribuiu para organizar e manipular informações de forma eficiente. O uso do SQLAlchemy como ORM simplificou o mapeamento das entidades e suas relações, permitindo uma interface mais natural para interagir com o banco de dados. A modelagem correta das entidades, como "Dono", "Pessoa Física", "Animal" e "Veterinário", facilitou o processo de CRUD, especialmente no que se refere ao gerenciamento de dados de **Pessoa Física**.

#### **Relacionamentos Multivalorados**

Uma das principais vantagens observadas foi a flexibilidade proporcionada pelos atributos multivalorados, como os números de telefone e vacinas. A tabela "Telefone", associada à entidade "Dono", permitiu que múltiplos números fossem atribuídos a cada proprietário, o que reflete melhor a realidade, onde uma pessoa pode possuir mais de um telefone. Da mesma forma, a tabela "Vacinas", associada à entidade "Animal", facilitou o armazenamento de múltiplas vacinas administradas, sem que fosse necessário criar campos redundantes na tabela principal.

Essa abordagem multivalorada possibilitou a expansão do sistema para novas funcionalidades, como o gerenciamento de vacinas de maneira mais robusta e eficaz. Mesmo que o foco inicial do sistema não fosse acompanhar detalhadamente históricos de consultas e tratamentos, a estrutura de dados criada permite facilmente incorporar esses aspectos no futuro, caso necessário.

#### **CRUD para Pessoa Física**

Os resultados obtidos no CRUD de **Pessoa Física** foram satisfatórios, pois o sistema demonstrou um bom desempenho ao realizar operações de criação, leitura, atualização e exclusão de registros no banco de dados. A funcionalidade de inserção de novos registros com múltiplos telefones funcionou de maneira esperada, possibilitando que as informações fossem corretamente armazenadas nas tabelas associadas.

A operação de consulta foi um dos pontos altos, permitindo recuperar dados completos de uma pessoa física, incluindo seus números de telefone e detalhes de endereço. Isso mostrou como o relacionamento entre as entidades foi bem projetado, facilitando a consulta de dados complexos com um simples comando. A opção de atualização também demonstrou ser flexível, permitindo que o usuário escolha campos específicos a serem alterados, sem necessidade de regravação total do registro.

#### **Integração entre as Entidades**

A integração entre as entidades "Dono", "Animal" e "Veterinário" mostrou-se eficaz no que diz respeito ao relacionamento entre veterinários e os animais que eles tratam. A tabela de associação "veterinario\_Animais" foi bem implementada, permitindo que um veterinário estivesse vinculado a vários animais e vice-versa, sem comprometer a integridade dos dados. Essa abordagem reforça a robustez da modelagem do sistema, que facilita o gerenciamento de informações em um contexto de relacionamentos complexos.

#### **Limitações e Expansões Futuras**

Apesar dos resultados positivos, algumas limitações foram observadas. O sistema atualmente realiza operações de CRUD apenas para a entidade **Pessoa Física**, deixando outras entidades, como **Veterinário, Animal**, e **ONG**, sem essas funcionalidades implementadas. No entanto, a arquitetura do sistema está preparada para que essas funcionalidades sejam facilmente adicionadas no futuro. Outro aspecto importante a ser considerado é a validação de dados, que atualmente é básica e pode ser aprimorada para garantir a consistência das informações inseridas.

A análise dos resultados demonstra que o sistema atende adequadamente às funcionalidades propostas, mas ainda há espaço para expandir o conjunto de operações de CRUD e a manipulação de dados para outras entidades além da **Pessoa Física**.

**CONCLUSÃO**

O sistema desenvolvido para a gestão de uma clínica veterinária demonstrou ser uma solução eficiente para o gerenciamento de dados relacionados a **pessoas físicas, animais, vacinas** e **veterinários**. A aplicação do SQLAlchemy como ferramenta ORM facilitou a implementação das operações de CRUD, o mapeamento objeto-relacional e o estabelecimento de relações complexas entre as entidades.

A escolha por atributos multivalorados para **telefones** e **vacinas** agregou flexibilidade ao sistema, permitindo uma estrutura expansível que reflete as necessidades reais de uma clínica veterinária. Embora o objetivo inicial não fosse gerenciar o histórico completo de consultas e tratamentos, o sistema foi projetado de forma que permita futuras expansões, incorporando novas funcionalidades sem a necessidade de grandes alterações na base existente.

A implementação do CRUD para a entidade **Pessoa Física** foi bem-sucedida, permitindo uma gestão completa de dados, desde a criação até a exclusão de registros. A integração entre as entidades e o uso de relações de muitos para muitos, como a existente entre **veterinários** e **animais**, também se mostraram bem estruturadas.

No entanto, o sistema ainda pode evoluir com a implementação do CRUD para outras entidades e o aprimoramento da validação de dados inseridos. Com uma arquitetura sólida, o sistema está bem preparado para receber futuras expansões, como o acompanhamento mais detalhado de tratamentos e consultas. Em suma, o projeto alcançou seu objetivo de proporcionar um gerenciamento eficaz e flexível para os dados essenciais de uma clínica veterinária, com grande potencial para evoluções futuras.

**REFERÊNCIAS**

* SQLAlchemy. SQLAlchemy: Database Toolkit for Python. Disponível em: https://www.sqlalchemy.org/.
* VAN ROSSEM, H. Python: Programming Language. Disponível em: https://www.python.org/.
* SQLite. SQLite: Database Engine. Disponível em: https://www.sqlite.org/. Acesso em: 16 set. 2024.
* PYTHON SOFTWARE FOUNDATION. os — Miscellaneous operating system interfaces. Disponível em: https://docs.python.org/3/library/os.html.
* SQLAlchemy. SQLAlchemy ORM Documentation. Disponível em: https://docs.sqlalchemy.org/en/20/orm/.
* SQLAlchemy. SQLAlchemy Core: SQL Expression Language. Disponível em: https://docs.sqlalchemy.org/en/20/core/metadata.html.
* SQLAlchemy. SQLAlchemy ORM Documentation: Relationships and Loading Strategies. Disponível em: https://docs.sqlalchemy.org/en/20/orm/relationships.html.
* DUNN, J. J. CRUD Operations: The Basics of Database Interaction. Disponível em: https://www.davidkinnard.com/blog/crud-operations-basics/.
* **POPA, L.** Python e Banco de Dados Relacionais. Disponível em: https://realpython.com/python-sqlalchemy/.
* **ESSLAM, A. et al.** SQLAlchemy ORM documentation. SQLAlchemy Project. Disponível em: https://docs.sqlalchemy.org/en/14/orm/.
* UNESP – Universidade Estadual Paulista "Júlio de Mesquita Filho". **Listas Opcionais de Documentos**. Disponível em: <https://www.ict.unesp.br/Home/sobreoict/biblioteca/dn-listas-opcionais.pdf>.

**APÊNDICE A**

**Texto

Descrição gerada automaticamenteTexto

Descrição gerada automaticamenteTexto

Descrição gerada automaticamenteTexto

Descrição gerada automaticamenteTexto

Descrição gerada automaticamenteTexto

Descrição gerada automaticamenteTexto

Descrição gerada automaticamenteTexto

Descrição gerada automaticamenteTexto

Descrição gerada automaticamenteTexto

Descrição gerada automaticamente**