

**TDE:**  
**ARQUITETURA DE BANCO DE**  
**DADOS**

**PUCPR**

**Felipe Fernandes Bortolino**

# **TDE NO FORMATO DE APLICAÇÃO DE PETSHOP**

# Resumo

Resumo: Essa aplicação é fruto de conhecimento, pesquisa e vários testes, além disso foi desenvolvido uma aplicação em python com ajuda do “Xampp” e o mysql (phpmyadmin) para desenvolver, nesse app é possível usar as funcionalidades básicas do CRUD (Create, Read, Update, Delete), tudo isso por meio da biblioteca “sqlalchemy” do python, essa plataforma nos dá uma grande diversidade de funções que permitiram o desenvolvimento e funcionamento do app, o app funciona por meio do de “inputs” que identificam as teclas digitadas e gravam as informações do “phpmyadmin” (mysql), o tutorial completo de como foi implementado para que possa ser reproduzido em outras máquinas será implementado ao longo da documentação.

# Sumário

Introdução .....
Instalando plugins .....
Configurando o Xampp .....
Conectando ao mysql .....
Criando banco de dados .....
Conectando banco de dados ao código .....
Criando classes .....
Criando funções .....
Verificações if, elif e else .....
Testando a aplicação .....
Corrigindo problemas .....
Usando CRUD .....
Referências .....

# Introdução

Nesse TDE aprendi algumas funções básicas e usei meus conhecimentos em python para implementar uma aplicação de petshop, usando alguns sites e vídeos no youtube foi possível implementar de forma que funcione em conjunto com mysql, o sistema em si faz tudo por meio de inputs, ou seja, não possui interface gráfica, o app funciona via terminal, seja no pycharm ou no visual studio code, para esse TDE funcionar em outras máquinas serão necessárias algumas bibliotecas e plugins, ao longo da documentação mostrarei passo a passo de como foi implementado.

## Resultado

Para começar vamos importar e instalar alguns plugins, na imagem abaixo mostrarei o comando e como achar no visual studio code:

```
sh-5.2$ pip install sqlalchemy
```

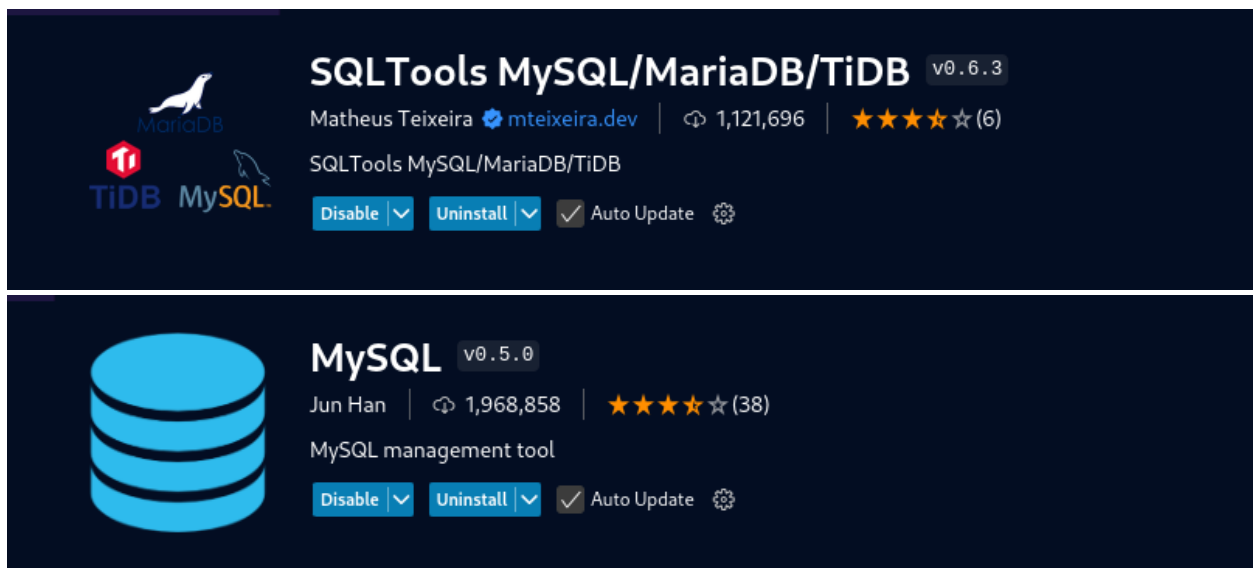
```
sh-5.2$ pip install pymysql
```

Após instalar os plugins via pip install vamos importa-los:

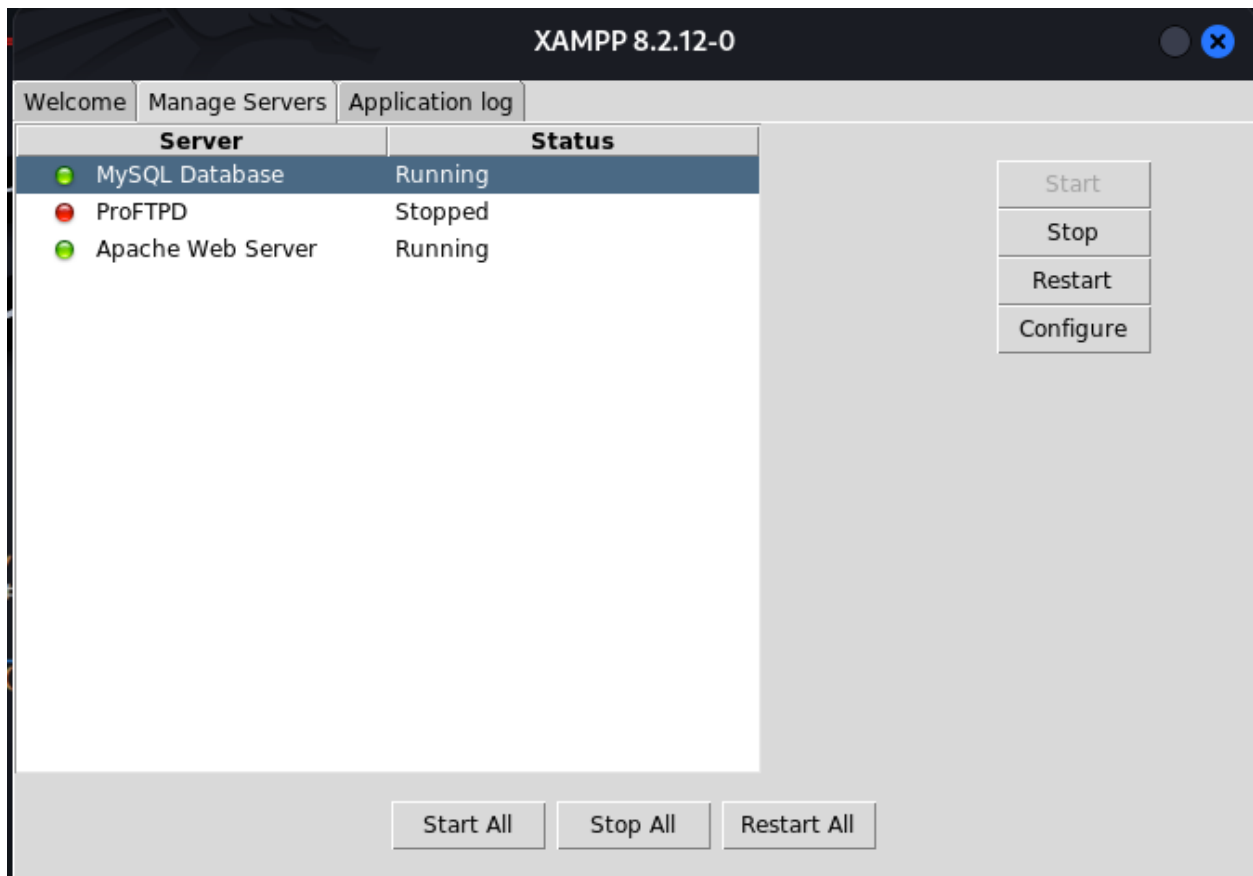
```
import os
import time
from sqlalchemy import create_engine, Column, Integer, String, Text, Numeric, ForeignKey
from sqlalchemy.ext.declarative import declarative_base
from sqlalchemy.orm import sessionmaker, relationship
```

Os imports OS e Time são apenas para design, não sendo necessários.

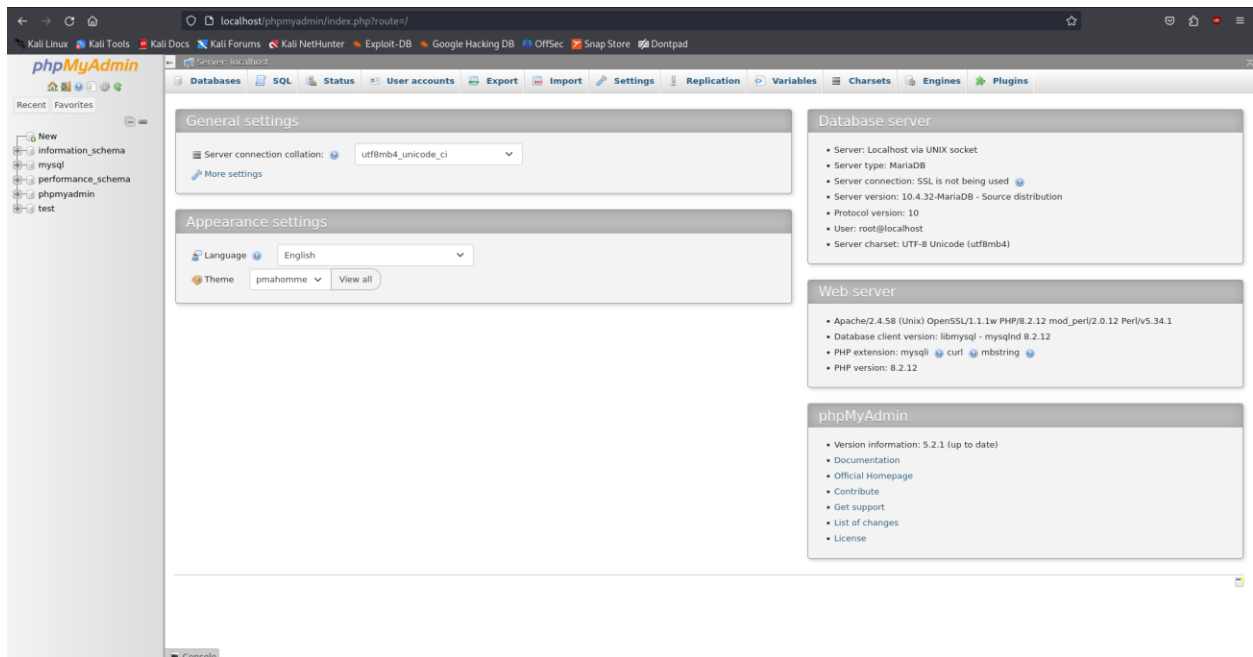
Agora vamos para os plugins do visual studio code:



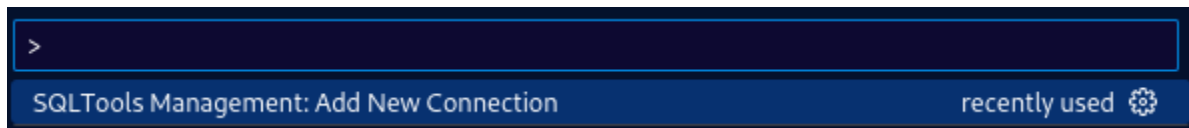
Em seguida vamos para o Xampp para criar um servidor local:



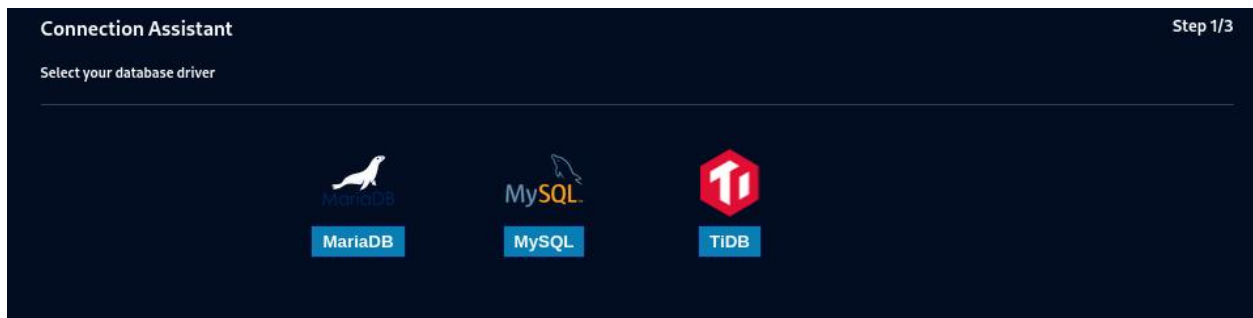
Abrindo com o Xampp rodando o localhost/phpmyadmin



Com CTRL + Shift + P, pesquisando SQLTools Management: Add New Connection





Vamos adicionar uma conexão via MySQL:



Antes de continuar no vscode vamos criar o banco de



# Databases

 **Create database** 

**Create**

O banco vai se chamar “tde”

Agora vamos continuar no vscode:

**Connection Assistant** < Step 2/3

**Connection Settings**

**Connection name\***

tde

**Connection group**

**Connect using\***

Server and Port

**Server Address\***

localhost

**Port\***

3306

**Database\***

tde

**Username\***

root

**Password mode**

Use empty password

**MySQL driver specific options**

**Authentication Protocol**

default

Try to switch protocols in case you have problems.

**SSL**

Disabled

**Connection Timeout**

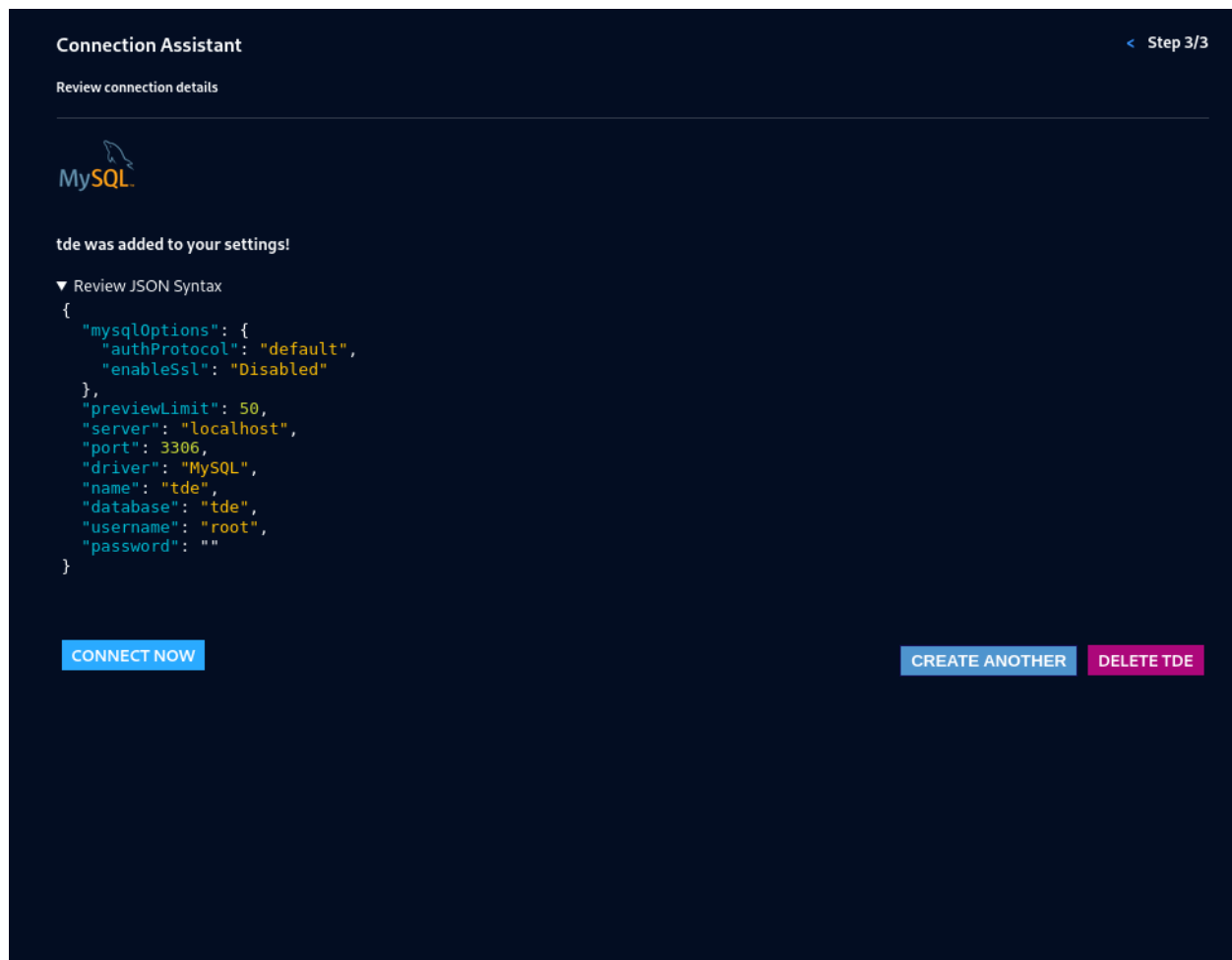
**Show records default limit**

50

Successfully connected!

**SAVE CONNECTION**

**TEST CONNECTION**



Depois de conectado, vamos conectar ao código:

```
5
7 DATABASE_URL = "mysql+pymysql://root:@localhost/tde"
8
9 engine = create_engine(DATABASE_URL)
9 Base = declarative_base()
1 Base.metadata.create_all(engine)
2
3 Session = sessionmaker(bind=engine)
4 session = Session()
5
```

Na linha `DATABASE_URL = "mysql+pymysql://root:@localhost/tde"` é onde foi configurado o nome, usuário, senha e host .

Após isso comecei meu programa conectando as tabelas e colunas para criar alterações pelo próprio programa sem necessidade de abrir o servidor "phpmyadmin" para alterar.

You, 2 days ago | 1 author (You)

```
class Cliente(Base):
    __tablename__ = 'clientes'
    id = Column(Integer, primary_key=True, autoincrement=True)
    nome = Column(String(255), nullable=False)
    telefone = Column(String(255), nullable=False)
    email = Column(String(255), unique=True, nullable=False)
    pets = relationship('Pet', back_populates='dono')
```

You, 2 days ago | 1 author (You)

```
class Pet(Base):
    __tablename__ = 'pets'
    id = Column(Integer, primary_key=True, autoincrement=True)
    nome = Column(String(255), nullable=False)
    especie = Column(String(255), nullable=False)
    raca = Column(String(255), default=None)
    idade = Column(Integer, default=None)
    id_dono = Column(Integer, ForeignKey('clientes.id'))
    dono = relationship('Cliente', back_populates='pets')
```

You, 2 days ago | 1 author (You)

```
class Servico(Base):
    __tablename__ = 'servicos'
    id = Column(Integer, primary_key=True, autoincrement=True)
    nome = Column(String(255), nullable=False)
    descricao = Column(Text, default=None)
    preco = Column(Numeric(10, 2), nullable=False)
```

You, 2 days ago | 1 author (You)

```
class Taxidog(Base):
    __tablename__ = 'taxidog'
    id = Column(Integer, primary_key=True, autoincrement=True)
    preco = Column(Numeric(10, 2), nullable=False)
    tempo_ida = Column(Integer, nullable=False)
    tempo_volta = Column(Integer, nullable=False)
```

Já podemos construir as funções para Adicionar, Ler, Atualizar e Deletar:

```
def adicionar_cliente(nome, telefone, email):
    novo_cliente = Cliente(nome=nome, telefone=telefone, email=email)
    session.add(novo_cliente)
    session.commit()
    print(f"Cliente '{nome}' adicionado com sucesso!")

def ler_cliente(id_cliente):
    cliente = session.query(Cliente).filter(Cliente.id == id_cliente).first()
    if cliente:
        return cliente
    print(f"Cliente com ID {id_cliente} não encontrado.")
    return None

def atualizar_cliente(id_cliente, nome=None, telefone=None, email=None):
    cliente = session.query(Cliente).filter(Cliente.id == id_cliente).first()
    if cliente:
        if nome:
            cliente.nome = nome
        if telefone:
            cliente.telefone = telefone
        if email:
            cliente.email = email
        session.commit()
        print(f"Cliente '{id_cliente}' atualizado com sucesso!")
    else:
        print(f"Cliente com ID {id_cliente} não encontrado.")

def deletar_cliente(id_cliente):
    cliente = session.query(Cliente).filter(Cliente.id == id_cliente).first()
    if cliente:
        session.delete(cliente)
        session.commit()
        print(f"Cliente '{id_cliente}' deletado com sucesso!")
    else:
        print(f"Cliente com ID {id_cliente} não encontrado.")
```

Faremos isso em todas as entidades.

Função principal:

```
def principal():
    while True:
        limpar_terminal()
        print("\nMenu do Petshop")
        print("1. Adicionar Cliente")
        print("2. Ler Cliente")
        print("3. Atualizar Cliente")
        print("4. Deletar Cliente")
        print("5. Adicionar Pet")
        print("6. Ler Pet")
        print("7. Atualizar Pet")
        print("8. Deletar Pet")
        print("9. Adicionar Serviço")
        print("10. Ler Serviço")
        print("11. Atualizar Serviço")
        print("12. Deletar Serviço")
        print("13. Adicionar Taxa de Transporte")
        print("14. Ler Taxa de Transporte")
        print("15. Atualizar Taxa de Transporte")
        print("16. Deletar Taxa de Transporte")
        print("0. Sair")

        escolha = input("Escolha uma opção: ")
```

```

if escolha == "0":
    break

if escolha == "1":
    nome = input("Nome do cliente: ")
    telefone = input("Telefone do cliente: ")
    email = input("Email do cliente: ")
    adicionar_cliente(nome, telefone, email)

elif escolha == "2":
    id_cliente = int(input("ID do cliente: "))
    cliente = ler_cliente(id_cliente)
    if cliente:
        print(f"ID: {cliente.id}, Nome: {cliente.nome}, Telefone: {cliente.telefone}, Email: {cliente.email}")

elif escolha == "3":
    id_cliente = int(input("ID do cliente: "))
    nome = input("Novo nome (deixe em branco para não alterar): ")
    telefone = input("Novo telefone (deixe em branco para não alterar): ")
    email = input("Novo email (deixe em branco para não alterar): ")
    atualizar_cliente(id_cliente, nome if nome else None, telefone if telefone else None, email if email else None)

elif escolha == "4":
    id_cliente = int(input("ID do cliente: "))
    deletar_cliente(id_cliente)

elif escolha == "5":
    nome = input("Nome do pet: ")
    especie = input("Espécie do pet: ")
    raca = input("Raça do pet: ")
    idade = int(input("Idade do pet: "))
    id_dono = int(input("ID do dono (cliente): "))
    adicionar_pet(nome, especie, raca, idade, id_dono)

elif escolha == "6":
    id_pet = int(input("ID do pet: "))
    pet = ler_pet(id_pet)
    if pet:
        print(f"ID: {pet.id}, Nome: {pet.nome}, Espécie: {pet.especie}, Raça: {pet.raca}, Idade: {pet.idade}, ID do Dono: {pet.id_dono}")

```

```

elif escolha == "7":
    id_pet = int(input("ID do pet: "))
    nome = input("Novo nome (deixe em branco para não alterar): ")
    especie = input("Nova espécie (deixe em branco para não alterar): ")
    raca = input("Nova raça (deixe em branco para não alterar): ")
    idade = input("Nova idade (deixe em branco para não alterar): ")
    id_dono = input("Novo ID do dono (deixe em branco para não alterar): ")
    atualizar_pet(id_pet, nome if nome else None, especie if especie else None, raca if raca else None, int(idade) if idade else None, int(id_dono) if id_dono else None)

elif escolha == "8":
    id_pet = int(input("ID do pet: "))
    deletar_pet(id_pet)

elif escolha == "9":
    nome = input("Nome do serviço: ")
    descricao = input("Descrição do serviço: ")
    preco = float(input("Preço do serviço: "))
    adicionar_servico(nome, descricao, preco)

elif escolha == "10":
    id_servico = int(input("ID do serviço: "))
    servico = ler_servico(id_servico)
    if servico:
        print(f"ID: {servico.id}, Nome: {servico.nome}, Descrição: {servico.descricao}, Preço: {servico.preco}")

elif escolha == "11":
    id_servico = int(input("ID do serviço: "))
    nome = input("Novo nome (deixe em branco para não alterar): ")
    descricao = input("Nova descrição (deixe em branco para não alterar): ")
    preco = input("Novo preço (deixe em branco para não alterar): ")
    atualizar_servico(id_servico, nome if nome else None, descricao if descricao else None, float(preco) if preco else None)

elif escolha == "12":
    id_servico = int(input("ID do serviço: "))
    deletar_servico(id_servico)

elif escolha == "13":
    preco = float(input("Preço da taxa de transporte: "))
    tempo_ida = int(input("Tempo de ida (em minutos): "))
    tempo_volta = int(input("Tempo de volta (em minutos): "))
    adicionar_taxidog(preco, tempo_ida, tempo_volta)

```

```

elif escolha == "13":
    preco = float(input("Preço da taxa de transporte: "))
    tempo_ida = int(input("Tempo de ida (em minutos): "))
    tempo_volta = int(input("Tempo de volta (em minutos): "))
    adicionar_taxidog(preco, tempo_ida, tempo_volta)

elif escolha == "14":
    id_taxa = int(input("ID da taxa de transporte: "))
    taxa = ler_taxidog(id_taxa)
    if taxa:
        print(f"ID: {taxa.id}, Preço: {taxa.preco}, Tempo de Ida: {taxa.tempo_ida}, Tempo de Volta: {taxa.tempo_volta}")

elif escolha == "15":
    id_taxa = int(input("ID da taxa de transporte: "))
    preco = input("Novo preço (deixe em branco para não alterar): ")
    tempo_ida = input("Novo tempo de ida (deixe em branco para não alterar): ")
    tempo_volta = input("Novo tempo de volta (deixe em branco para não alterar): ")
    atualizar_taxidog(id_taxa, float(preco) if preco else None, int(tempo_ida) if tempo_ida else None, int(tempo_volta) if tempo_volta else None)

elif escolha == "16":
    id_taxa = int(input("ID da taxa de transporte: "))
    deletar_taxidog(id_taxa)

if __name__ == "__main__":
    principal()

```

Esses "if", "elif" e "else" são verificação para o input, e o if `__name__ == "__main__"`:

É para que o programa comece a ser executado por ele!

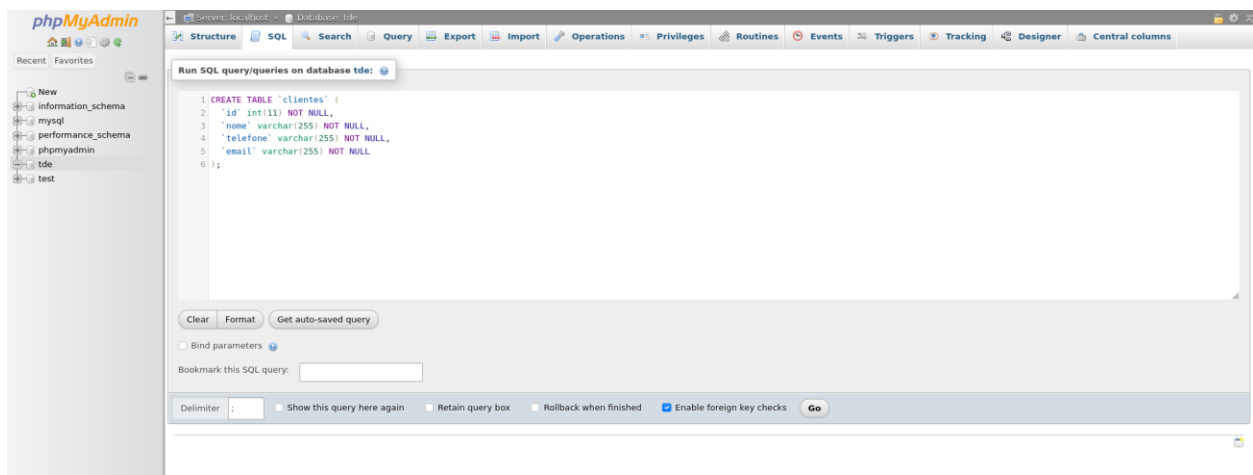
Iniciando o programa:

```

Menu do Petshop
1. Adicionar Cliente
2. Ler Cliente
3. Atualizar Cliente
4. Deletar Cliente
5. Adicionar Pet
6. Ler Pet
7. Atualizar Pet
8. Deletar Pet
9. Adicionar Serviço
10. Ler Serviço
11. Atualizar Serviço
12. Deletar Serviço
13. Adicionar Taxa de Transporte
14. Ler Taxa de Transporte
15. Atualizar Taxa de Transporte
16. Deletar Taxa de Transporte
0. Sair
Escolha uma opção:

```

Antes de testar vamos criar as tabelas e colunas dentro do phpmyadmin:



Faremos isso para todos.

Me deparei com alguns erros ao longo desse processo, então fui perguntar em alguns foruns e canais do discord, cheguei a algumas resoluções:

```
ALTER TABLE `clientes`  
ADD PRIMARY KEY (`id`),  
ADD UNIQUE KEY `email` (`email`);
```

```
ALTER TABLE `pets`  
ADD PRIMARY KEY (`id`),  
ADD KEY `id_dono` (`id_dono`);
```

```
ALTER TABLE `servicos`  
ADD PRIMARY KEY (`id`);
```

```
ALTER TABLE `taxidog`  
ADD PRIMARY KEY (`id`);
```

```
ALTER TABLE `clientes`  
MODIFY `id` int(11) NOT NULL AUTO_INCREMENT, AUTO_INCREMENT=3;
```

```
ALTER TABLE `pets`  
MODIFY `id` int(11) NOT NULL AUTO_INCREMENT, AUTO_INCREMENT=2;
```



```
ALTER TABLE `servicos`  
MODIFY `id` int(11) NOT NULL AUTO_INCREMENT;  
ALTER TABLE `taxidog`  
MODIFY `id` int(11) NOT NULL AUTO_INCREMENT;  
  
ALTER TABLE `pets`  
ADD CONSTRAINT `pets_ibfk_1` FOREIGN KEY (`id_dono`) REFERENCES `clientes` (`id`);
```

Copiado do tde.sql, por isso o formato e cores!

Já corrigido, podemos testar:

Adicionando Cliente:

```
Menu do Petshop  
1. Adicionar Cliente  
2. Ler Cliente  
3. Atualizar Cliente  
4. Deletar Cliente  
5. Adicionar Pet  
6. Ler Pet  
7. Atualizar Pet  
8. Deletar Pet  
9. Adicionar Serviço  
10. Ler Serviço  
11. Atualizar Serviço  
12. Deletar Serviço  
13. Adicionar Taxa de Transporte  
14. Ler Taxa de Transporte  
15. Atualizar Taxa de Transporte  
16. Deletar Taxa de Transporte  
0. Sair  
Escolha uma opção: 1  
Nome do cliente: Teste  
Telefone do cliente: 41999999999  
Email do cliente: teste@teste.com
```

Após adicionado vamos olhar como está no phpmyadmin:



#### Menu do Petshop

1. Adicionar Cliente
2. Ler Cliente
3. Atualizar Cliente
4. Deletar Cliente
5. Adicionar Pet
6. Ler Pet
7. Atualizar Pet
8. Deletar Pet
9. Adicionar Serviço
10. Ler Serviço
11. Atualizar Serviço
12. Deletar Serviço
13. Adicionar Taxa de Transporte
14. Ler Taxa de Transporte
15. Atualizar Taxa de Transporte
16. Deletar Taxa de Transporte
0. Sair

Escolha uma opção: 2

Atualizando cliente:

#### Menu do Petshop

1. Adicionar Cliente
2. Ler Cliente
3. Atualizar Cliente
4. Deletar Cliente
5. Adicionar Pet
6. Ler Pet
7. Atualizar Pet
8. Deletar Pet
9. Adicionar Serviço
10. Ler Serviço
11. Atualizar Serviço
12. Deletar Serviço
13. Adicionar Taxa de Transporte
14. Ler Taxa de Transporte
15. Atualizar Taxa de Transporte
16. Deletar Taxa de Transporte
0. Sair

Escolha uma opção: 3

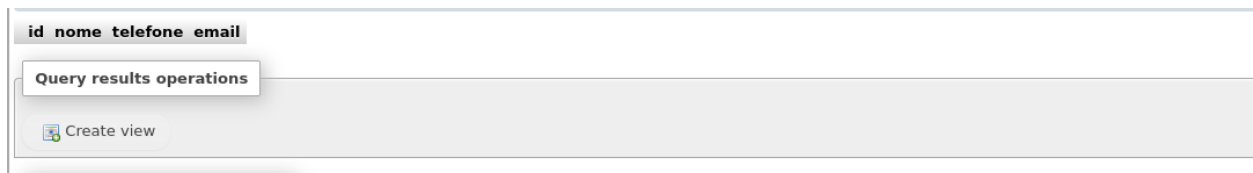
ID do cliente: 3

Novo nome (deixe em branco para não alterar):

Deletando Cliente:

```
Menu do Petshop
1. Adicionar Cliente
2. Ler Cliente
3. Atualizar Cliente
4. Deletar Cliente
5. Adicionar Pet
6. Ler Pet
7. Atualizar Pet
8. Deletar Pet
9. Adicionar Serviço
10. Ler Serviço
11. Atualizar Serviço
12. Deletar Serviço
13. Adicionar Taxa de Transporte
14. Ler Taxa de Transporte
15. Atualizar Taxa de Transporte
16. Deletar Taxa de Transporte
0. Sair
Escolha uma opção: 4
ID do cliente: 3
```

Vamos olhar no phpmyadmin:



Apagado com sucesso.

Tudo o que foi solicitado pelo AVA para o TDE foi implementado

# Referências

<https://discord.gg/rocketseat>

<https://discord.gg/butecodosdevs>

<https://discord.gg/RFrpgJd7Gq>

<https://pt.stackoverflow.com/questions/tagged/banco-de-dados>

[https://www.youtube.com/watch?v=\\_q3j25ACmQ4&pp=ygUiY3JpYW5kbyBwcm9qZXRvIGNvbSBweXRob24gZSBteXNxbA%3D%3D](https://www.youtube.com/watch?v=_q3j25ACmQ4&pp=ygUiY3JpYW5kbyBwcm9qZXRvIGNvbSBweXRob24gZSBteXNxbA%3D%3D)

<https://youtube.com/playlist?list=PLwsAoT89dh3rkdg2MYH4jOhzQbw1eKYU7&si=cpCKIyrqLFfVI5o>

<https://www.youtube.com/watch?v=Mdg1D-Kdmrw&pp=ygUiY3JpYW5kbyBwcm9qZXRvIGNvbSBweXRob24gZSBteXNxbA%3D%3D>