SQLAlchemy, ORM I

Kaip susikurti objekto klasę, iš kurios bus sukurta lentelė:

Tam, kad sukurtume reikiamos struktūros duomenų bazę ir ja naudotumės, užtenka sukurti SQLAlchemy klasę ir ją paleisti.

```
import datetime
from sqlalchemy import Column, Integer, String, Float, DateTime, create_engine
from sqlalchemy.ext.declarative import declarative_base

engine = create_engine('sqlite:///projektai.db')
Base = declarative_base()

class Projektas(Base):
    __tablename__ = 'Projektas'
    id = Column(Integer, primary_key=True)
    name = Column("Pavadinimas", String)
    price = Column("Kaina", Float)
    created_date = Column("Sukūrimo data", DateTime,
default=datetime.datetime.utcnow)

    def __init__(self, name, price):
        self.name = name
        self.price = price

    def __repr__(self):
        return f"{self.id} {self.name} - {self.price}: {self.created_date}"

Base.metadata.create_all(engine)
```

Kaip įrašyti, nuskaityti, atnaujinti, ištrinti duomenys SQLAlchemy lentelėje

(CRUD – create, read, update, delete)

Kaip sukurti ryšį su sukurta DB kitame faile:

```
from sqlalchemy import create_engine
from sqlalchemy.orm import sessionmaker
from projektas import Projektas

engine = create_engine('sqlite:///projektai.db')
Session = sessionmaker(bind=engine)
session = Session()
```

Kaip įrašyti duomenis į lentelę:

(**C**rud)

```
projektas1 = Projektas("Naujas pr.", 20000)
session.add(projektas1)
session.commit()

projektas2 = Projektas("2 projektas", 55000)
session.add(projektas2)
session.commit()
```

Kaip gauti duomenis iš lentelės (cRud):

```
projektas1 = session.query(Projektas).get(1)

print(projektas1.name)
# Naujas pr.

projektas2 = session.query(Projektas).filter_by(name="2 projektas").one()

projektai = session.query(Projektas).all()

for projektas in projektai:
    print(projektas.name, projektas.price)
# Naujas pr. 20000.0
# 2 projektas 55000.0
```

Kaip ieškoti duomenų pagal sąlygą ar šabloną:

```
search = session.query(Projektas).filter(Projektas.name.ilike("2%"))
search2 = session.query(Projektas).filter(Projektas.price > 1000)
search3 = session.query(Projektas).filter(
    Projektas.price > 1000,
    Projektas.name.ilike("2%"))

print([i for i in search])
print([i for i in search2])
print([i for i in search3])

# [2 2 projektas - 55000.0: 2021-02-03 14:29:33.477232]
# [1 Naujas pr. - 20000.0: 2021-02-03 14:29:33.437231, 2 2 projektas - 55000.0:
2021-02-03 14:29:33.477232]
# [2 2 projektas - 55000.0: 2021-02-03 14:29:33.477232]
```

Kaip pakeisti duomenis lentelėje (crUd):

```
projektas1 = session.query(Projektas).get(1)
projektas1.price = 22000
session.commit()

projektas2 = session.query(Projektas).filter_by(name="2 projektas").one()
projektas2.name = "2 projektas tikrai"
session.commit()
```

Kaip ištrinti duomenis lentelėje (cruD):

```
projektas1 = session.query(Projektas).filter_by(name="Naujas pr.").one()

session.delete(projektas1)
session.commit()
```

Programos su duomenų baze (konsolėje) pavyzdys:

```
from models import engine, Projektas
from sqlalchemy.orm import sessionmaker

engine = create_engine('sqlite:///projektai.db')
Session = sessionmaker(bind=engine)
session = Session()

while True:
```

```
    pasirinkimas = int(input("Pasirinkite veiksmą: \n1 - atvaizduoti projektus \n2
- sukurti projektą \n3 - pakeisti projektą \n4 - ištrinti projektą\n"))

    if pasirinkimas == 1:
        projektai = session.query(Projektas).all()
        print("--------------------")
        for projektas in projektai:
            print(projektas)
        print("--------------------")

    if pasirinkimas == 2:
        name = input("Įveskite projekto pavadinimą")
        price = float(input("Įveskite projekto kainą"))
        projektas = Projektas(name, price)
        session.add(projektas)
        session.commit()

    if pasirinkimas == 3:
        projektai = session.query(Projektas).all()
        print("--------------------")
        for projektas in projektai:
            print(projektas)
        print("--------------------")
        keiciamo_id = int(input("Pasirinkite norimo pakeisti projekto ID"))
        keiciamas_projektas = session.query(Projektas).get(keiciamo_id)
        pakeitimas = int(input("Ką norite pakeisti: 1 - pavadinimą, 2 - kainą"))
        if pakeitimas == 1:
            keiciamas_projektas.name = input("Įveskite projekto pavadinimą")
        if pakeitimas == 2:
            keiciamas_projektas.price = float(input("Įveskite projekto kainą"))
        session.commit()

    if pasirinkimas == 4:
        projektai = session.query(Projektas).all()
        print("--------------------")
        for projektas in projektai:
            print(projektas)
        print("--------------------")
        keiciamo_id = int(input("Pasirinkite norimo ištrinti projekto ID"))
        trinamas_projektas = session.query(Projektas).get(keiciamo_id)
        session.delete(trinamas_projektas)
        session.commit()
```