Первое знакомство с ORM SQLAlchemy

Никита Нестеренко
Руководитель проекта и главный инженер по разработке в СБЕР

Skillbox

ORM: что это и зачем нужно?

ORM-определение

ORM — это технология программирования, которая связывает базы данных с концепциями объектно-ориентированных языков программирования.

Запрос сущности из БД напрямую

```python
if __name__ == '__main__':

    import sqlite3

    sqlite_connection = sqlite3.connect('sqlite_python.db')


    create_table_query = '''CREATE TABLE players (
                                id INTEGER PRIMARY KEY,
                                name TEXT NOT NULL,
                                email text NOT NULL UNIQUE,
                                joining_date datetime,
                                salary REAL NOT NULL);'''

    cursor = sqlite_connection.cursor()
    cursor.execute(create_table_query)
    sqlite_connection.commit()

    select_players_query = '''
    SELECT * FROM  players WHERE name = "Nikita";
    '''

    cursor = sqlite_connection.cursor()
    students = cursor.execute(select_players_query)
    cursor.close()
```

Запрос с использованием ORM

```python
if __name__ == '__main__':

    from sqlalchemy import Column, Integer, String, DateTime, Float
    from sqlalchemy.orm import sessionmaker, declarative_base
    from sqlalchemy import create_engine

    engine = create_engine('sqlite:///sqlite_python.db')
    Base = declarative_base()
    Session = sessionmaker(bind=engine)
    session = Session()

    class PLayer(Base):
        __tablename__ = 'players'

        id = Column(Integer, primary_key=True)
        name = Column(String, nullable=False)
        email = Column(String, nullable=False, unique=True)
        joining_date = Column(DateTime)
        salary = Column(Float, nullable=False)

    Base.metadata.create_all(engine)
    players = session.query(PLayer).filter_by(name="Никита")
```

Плюсы при работе с ORM

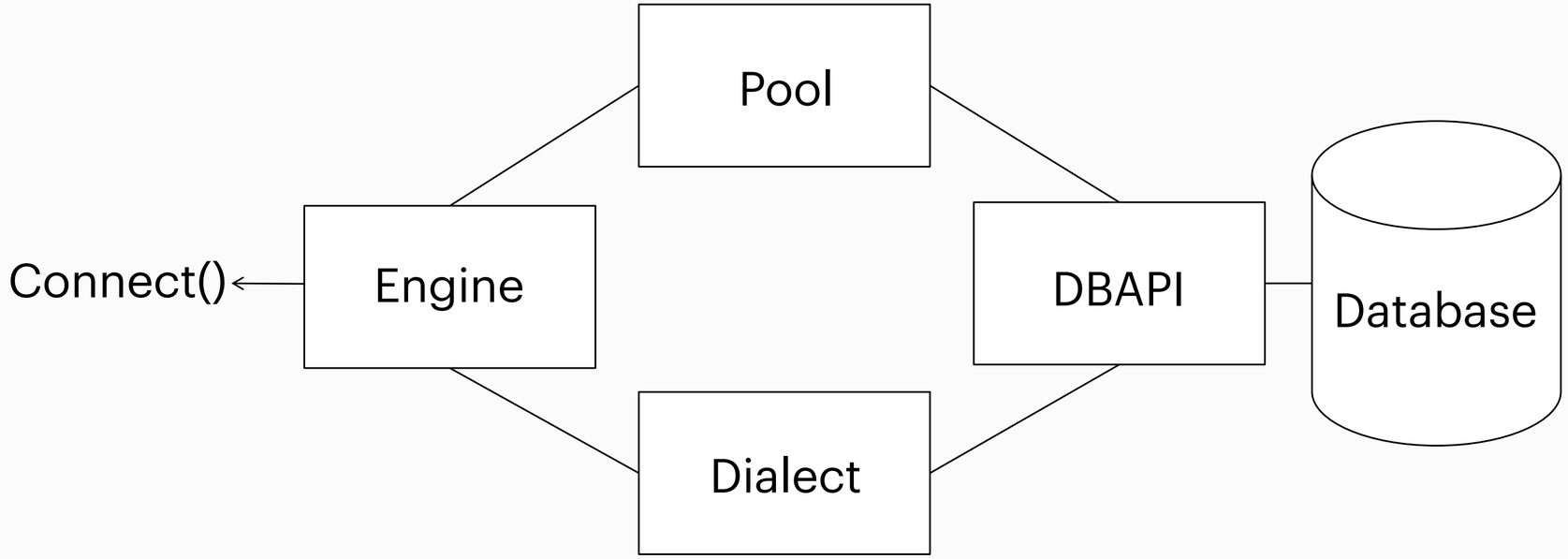
1. Познакомиться с принципами обратной связи, её видами
2. Не нужно писать SQL-запросы
3. Не нужно самому создавать таблицы
4. Такой код легко сопровождать

Минусы при работе с ORM

1. Снижение скорости работы с базой
2. Затраченное время на освоение ORM
3. Потеря контроля над SQL-запросами

Способы подключения ORM к приложению

Структура взаимодействия ORM с БД



Engine-определение

engine =
create_engine("dialect[+driver]://login:password@host/db_n
ame[?key=value]"), где:
dialect — это название СУБД (mssql, postgres, mysql)
driver — это название DBAPI (psycopg2, pyodbc).

Пример:

engine =
create_engine("mssql+pyodbc://supermegaadmin:123QWEa
sd@localhost/my_db")

Модели базы данных

CRUD методы ORM

Итоги модуля

- Ознакомились с концепцией ORM на примере модуля SQLAlchemy
- Научились работать с базовыми конструкциями запросов
- Описали таблицы с помощью ORM-моделей
- Интегрировали ORM в приложение