Python

16

Postgresql

sudo apt install libpq-dev python3-dev

pip install psycopg2

Sqlalchemy-utils

pip install sqlalchemy-utils

Подключение к postgresql через pycharm

View -> Tool Buttons -> Database -> + -> Data source -> PostgreSQL -> + -> OK

Host: localhost

User: postgres

Password: postgres

Port: 5432

Создание подключения к postgresql через sqlalchemy

```
from sqlalchemy_utils import create_database, database_exists

DB_USER = 'postgres'
DB_PASSWORD = 'postgres'
DB_NAME = 'test'
DB_ECHO = True
engine = create_engine(
    # "postgresql://postgres:postgres@localhost/test",
    f'postgresql://{DB_USER}:{DB_PASSWORD}@localhost/{DB_NAME}',
    echo=True,
)
if not database_exists(engine.url):
    create_database(engine.url)
```

Задание 16.01

Создать таблицу Учебной группы(Group) с помощью sqlalchemy. Группа характеризуется названием(name).

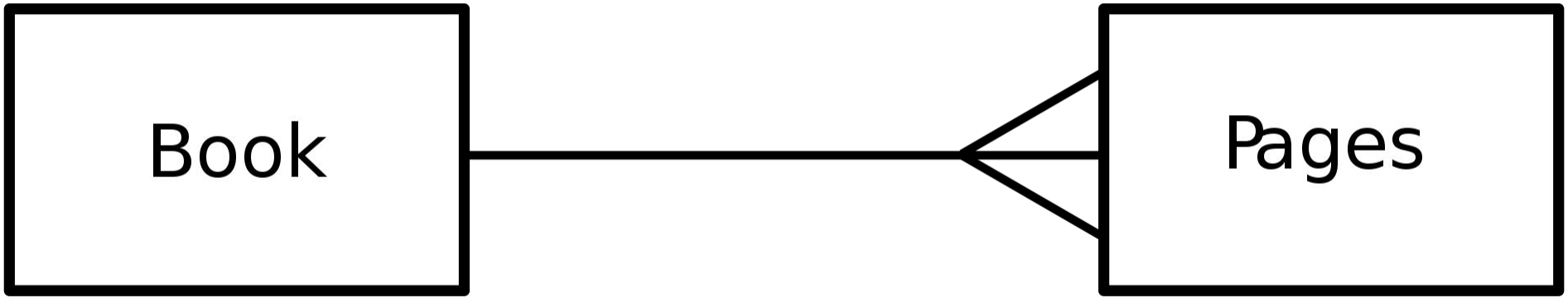
Типы связи

1) Многие-к-одному
2) Один-к-одному
3) Многие-ко-многим

Связь многие-к-одному(ForeignKey)

Основная задача: привязка несколько сущностей к одной общей.

Примеры: альбом и песни, футболисты и команда



Связь многие-к-одному(ForeignKey) SQL

```
CREATE TABLE artist(
 id INTEGER PRIMARY KEY,
 name TEXT
);
CREATE TABLE album(
   id     INTEGER,
   name   TEXT,
   artist_id INTEGER,
   FOREIGN KEY (artist_id) REFERENCES artist(id)
);
```

Объединение таблицы album и таблицы artist

Связь многие-к-одному(ForeignKey) sqlalchemy

```
class Artist(Base):
    __tablename__ = 'artist'
    id = Column(Integer, primary_key=True)
    name = Column(String)

class Album(Base):
    __tablename__ = 'album'
    id = Column(Integer, primary_key=True)
    name = Column(String)
    artist_id = Column(
        Integer, ForeignKey('artist.id'), nullable=False)

    artist = relationship(
        'Artist', foreign_keys='Album.artist_id', backref='albums')
```

```
Base.metadata.create_all(engine)

Session = sessionmaker(bind=engine)
session = Session()

artist = Artist(name='MyBand')
album = Album(name='MyAlbum', artist=artist)
session.add_all([artist, album])
session.commit()
```

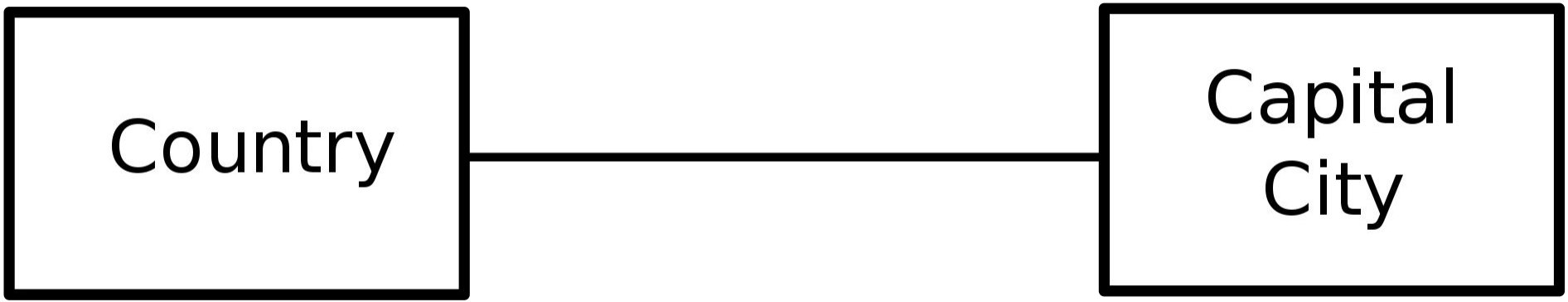
Задание 16.02

Создать таблицу Студент(Student) с помощью sqlalchemy. Студент характеризуется именем(firstname) и фамилией(lastname) и группой к которой он приурочен.

Создать две группы. Добавить в каждую по три студента.

Связь Один-к-одному (OneToOne)

Основная задача - дробление большой таблицы(модели) на мелкие составляющие.



Связь Один-к-одному (OneToOne) SQL

```
CREATE TABLE booklet (
  id SERIAL NOT NULL,
  description VARCHAR,
  album_id INTEGER NOT NULL,
  PRIMARY KEY (id),
  FOREIGN KEY(album_id) REFERENCES album
(id)
)
```

Связь Один-к-одному (OneToOne) sqlalchemy

```
class Booklet(Base):
    __tablename__ = 'booklet'
    id = Column(Integer, primary_key=True)
    description = Column(String)
    album_id = Column(
        Integer, ForeignKey('album.id'), nullable=False)

    album = relationship(
        'Album',
        foreign_keys='Booklet.album_id',
        backref=backref('booklet', uselist=False))
```

```
booklet = Booklet(description='blabla',
album=album)
print(album.booklet.description)
print(booklet.album.name)
```

Задание 16.03

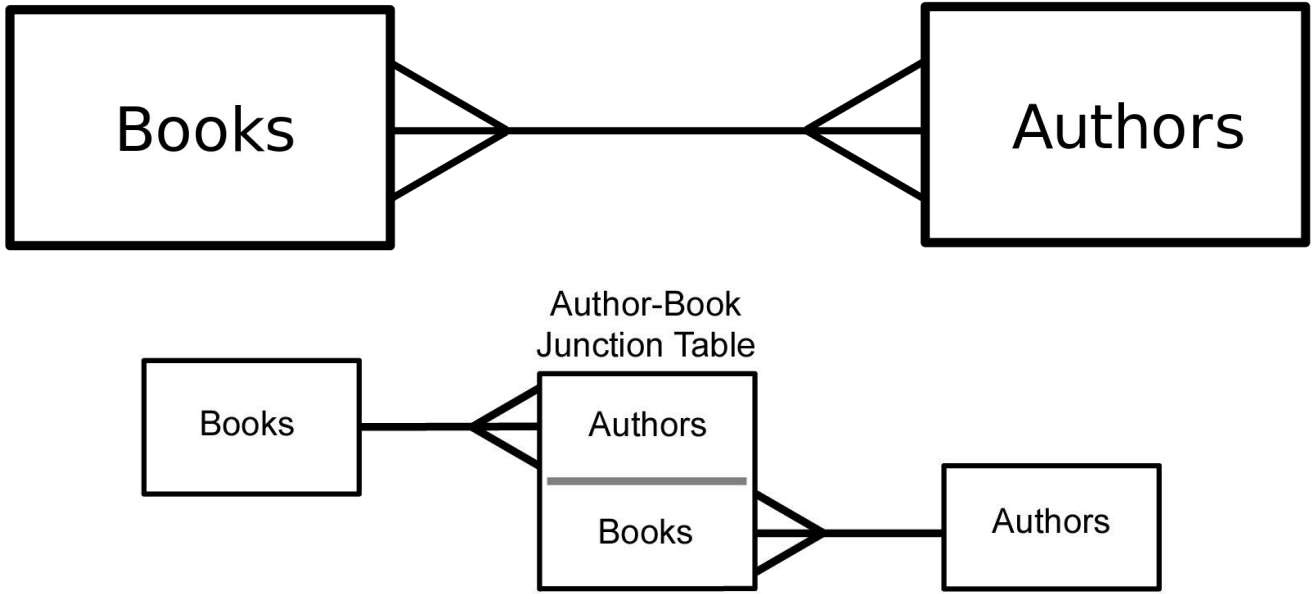
Создать таблицу Школьный дневник(Diary) с помощью sqlalchemy. Дневник характеризуется Средним баллом и студентом к которому он приурочен.

Получить всех студентов и создать для каждого дневник

Связь многие-ко-многим(ManyToMany)

Основная задачи: связать множество одних сущностей с множеством других

Пример: блюда и ингредиенты, студенты и книги.



Связь многие-ко-многим(ManyToMany) SQL

```
CREATE TABLE track (
   id SERIAL NOT NULL,
   name VARCHAR,
   duration FLOAT,
   PRIMARY KEY (id)
);

CREATE TABLE association (
   album_id INTEGER,
   track_id INTEGER,
   FOREIGN KEY(album_id) REFERENCES album (id),
   FOREIGN KEY(track_id) REFERENCES track (id)
)
```

Связь многие-ко-многим(ManyToMany) sqlalchemy

```
association_table = Table('association', Base.metadata,
   Column('album_id', Integer, ForeignKey('album.id')),
   Column('track_id', Integer, ForeignKey('track.id'))
)

class Track(Base):
   __tablename__ = 'track'
   id = Column(Integer, primary_key=True)
   name = Column(String)
   duration = Column(Float)
   albums = relationship('Album',
secondary=association_table, backref='tracks')
```

Создание таблицы ассоциации

Задание 16.04

Создать таблицу Книга(Book) с помощью sqlalchemy. Книга характеризуется названием, количеством страниц и студентами у которых эта книга.

Создать 5 книг. Получить всех студентов и добавить каждому студенту эти пять книг.