Микулин Михаил
ИУ5-36Б

Рубежный контроль №1

Вариант запросов Д
Предметная область 17

17	Дирижер	Оркестр

Оркестр и Дирижер связаны соотношением один-ко-многим. Выведите список всех дирижеров, у которых фамилия заканчивается на «ни», и названия их оркестров.

Оркестр и Дирижер связаны соотношением один-ко-многим. Выведите список оркестров со средним опытом дирижеров, отсортированный по среднему опыту.

Оркестр и Дирижер связаны соотношением многие-ко-многим. Выведите список всех оркестр, у которых название начинается с буквы «R», и список работающих в них дирижеров.

Текст программы

conductor.py

```
class Conductor:

   def __init__(self, id, fio, orchestra_id, years_of_experience = 0):

       self.id = id

       self.fio = fio

       self.orchestra_id = orchestra_id

       self.years_of_experience = years_of_experience
```

orchestra.py

```
class Orchestra:
    def __init__(self, id, name, home_theater):
        self.id = id
        self.name = name
        self.home_theater = home_theater
```

connection.py

```
class ConductorOrchestra:
    def __init__(self, conductor_id, orchestra_id):
        self.conductor_id = conductor_id
        self.orchestra_id = orchestra_id
```

main.py

```
from src.conductor import *
from src.orchestra import *
from src.connection import *
orchestras = [
    Orchestra(1, "Royal Concertgebouw Orchestra", "Amsterdam"),
    Orchestra(2, "Berlin Philharmonic Orchestra", "Berlin"),
    Orchestra(3, "Los Angeles Philharmonic;", "Los-Angeles")
]

conductors = [
    Conductor(1, "Томас Бичем", 1, 10),
    Conductor(2, "Карло Мария Джулини", 2, 15),
    Conductor(3, "Артуро Тосканини", 3, 16),
    Conductor(4, "Вильгельм Фуртвенглер", 2, 19),
    Conductor(5, "Николаус Арнонкур", 1, 12)
]

many_to_many = [
    ConductorOrchestra(3, 1),
    ConductorOrchestra(2, 1),
    ConductorOrchestra(1, 2),
    ConductorOrchestra(1, 3),
    ConductorOrchestra(4, 2)
```

```
]

def first_task():
    r = [(
    conductor.fio, orchestra.name)
    for conductor in conductors
        for orchestra in orchestras
            if (conductor.orchestra_id == orchestra.id)
                and (conductor.fio[-2:] == "ни")]
    return r

def second_task():
    r = [[_.name, 0, 0] for _ in orchestras]
    for conductor in conductors:
        r[conductor.orchestra_id - 1][1] += conductor.years_of_experience
        r[conductor.orchestra_id - 1][2] += 1

    r = [(_[0], _[1]/_[2]) for _ in r]
    r.sort(key=lambda r: r[1])

    return r

def third_task():
    r = [(_.name, []) for _ in orchestras]
    connections = [(orchestras[connection.orchestra_id-1].name,
conductors[connection.conductor_id-1].fio) for connection in many_to_many]
    for _ in connections:
        for elm in r:
            if elm[0] == _[0]:
                elm[1].append(_[1])
    to_be_del = []
    for i in range (len(r)):
        if r[i][0][0] != "R":
            to_be_del.append(i)

    for i in range (len(to_be_del)):
        del r[i]
    return r

def main():
    print(first_task())
    print(second_task())
    print(third_task())
if __name__ == '__main__':
    main()
```

Результат программы

//первый запрос
[('Карло Мария Джулини', 'Berlin Philharmonic Orchestra'), ('Артуро Тосканини', 'Los Angeles Philharmonic;')]

//второй запрос
[('Royal Concertgebouw Orchestra', 11.0), ('Los Angeles Philharmonic;', 16.0), ('Berlin Philharmonic Orchestra', 17.0)]

//третий запрос
[('Berlin Philharmonic Orchestra', ['Томас Бичем', 'Вильгельм Фуртвенглер'])]