Пахомкин Кирсан Саналович ИУ5-51Б
Вариант А-17

1. «Оркестр» и «Дирижер» связаны соотношением один-ко-многим. Выведите список всех связанных оркестров и дирижеров, отсортированный по оркестрам, сортировка по дирижерам произвольная.

2. «Оркестр» и «Дирижер» связаны соотношением один-ко-многим. Выведите список отделов с суммарной зарплатой дирижеров в каждом оркестре, отсортированный по суммарной зарплате.

3. «Оркестр» и «Дирижер» связаны соотношением многие-ко-многим. Выведите список всех оркестров, у которых в названии присутствует слово «оркестр», и список работающих в них дирижеров.

```
# используется для сортировки
from operator import itemgetter


class Dir:
    """Дирижер"""

    def __init__(self, id, fio, sal, Ork_id):
        self.id = id
        self.fio = fio
        self.sal = sal
        self.Ork_id = Ork_id


class Ork:
    """Оркестр"""

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


class DirOrk:
    """
    'Дирижер оркестра' для реализации
    связи многие-ко-многим
    """

    def __init__(self, Ork_id, Dir_id):
        self.Ork_id = Ork_id
        self.Dir_id = Dir_id


# Оркестры
Orks = [
    Ork(1, 'оркестр духовой'),
    Ork(2, 'оркестр джазовый'),
    Ork(3, 'Симфонический'),

]

# Дирижеры
Dirs = [
```

```
    Dir(1, 'Пахомкин', 25000, 1),
    Dir(2, 'Петров', 35000, 2),
    Dir(3, 'Иваненко', 45000, 3),
    Dir(4, 'Толпаров', 35000, 3),
    Dir(5, 'Ибрагимов', 25000, 3),
]

Dirs_Orks = [
    DirOrk(1, 1),
    DirOrk(2, 2),
    DirOrk(3, 3),
    DirOrk(3, 4),
    DirOrk(3, 5),

]


def main():
    """Основная функция"""

    # Соединение данных один-ко-многим
    one_to_many = [(e.fio, e.sal, d.name)
                   for d in Orks
                   for e in Dirs
                   if e.Ork_id == d.id]

    # Соединение данных многие-ко-многим
    many_to_many_tDir = [(d.name, ed.Ork_id, ed.Dir_id)
                         for d in Orks
                         for ed in Dirs_Orks
                         if d.id == ed.Ork_id]

    many_to_many = [(e.fio, e.sal, Ork_name)
                    for Ork_name, Ork_id, Dir_id in many_to_many_tDir
                    for e in Dirs if e.id == Dir_id]

    print('Задание А1')
    res_11 = sorted(one_to_many, key=itemgetter(2))
    print(res_11)

    print('\nЗадание А2')
    res_12_unsorted = []
    for d in Orks:
        # Список
        d_Dirs = list(filter(lambda i: i[2] == d.name, one_to_many))
        # Если не пустой
        if len(d_Dirs) > 0:
            # Зарплаты дирижеров оркестра
            d_sals = [sal for _, sal, _ in d_Dirs]
            # Суммарная зарплата дирижеров оркестра
            d_sals_sum = sum(d_sals)
            res_12_unsorted.append((d.name, d_sals_sum))

    # Сортировка по суммарной зарплате
    res_12 = sorted(res_12_unsorted, key=itemgetter(1), reverse=True)
    print(res_12)

    print('\nЗадание А3')
    res_13 = {}
    # Перебираем все оркестры
    for d in Orks:
        if 'оркестр' in d.name:
            # Список дирижеров оркестра
            d_Dirs = list(filter(lambda i: i[2] == d.name, many_to_many))
```

```
            # Только ФИО дирижера
            d_Dirs_names = [x for x, _, _ in d_Dirs]
            # Добавляем результат в словарь
            # ключ - отдел, значение - список фамилий
            res_13[d.name] = d_Dirs_names

    print(res_13)


if __name__ == '__main__':
    main()
```

Задание A1
[('Иваненко', 45000, 'Симфонический'), ('Толпаров', 35000, 'Симфонический'), ('Ибрагимов', 25000, 'Симфонический'), ('Петров', 35000, 'оркестр джазовый'), ('Пахомкин', 25000, 'оркестр духовой')]

Задание A2
[('Симфонический', 105000), ('оркестр джазовый', 35000), ('оркестр духовой', 25000)]

Задание A3
{'оркестр духовой': ['Пахомкин'], 'оркестр джазовый': ['Петров']}