Группа ИУ5-32Б
Студент Борисова Кристина

Рубежный контроль 1.

Полученное задание:

1) Необходимо создать два класса данных в соответствии с Вашим вариантом предметной области, которые связаны отношениями один-ко-многим и многие-ко-многим.
2) Необходимо создать списки объектов классов, содержащих тестовые данные (3-5 записей), таким образом, чтобы первичные и вторичные ключи соответствующих записей были связаны по идентификаторам.
3) Необходимо разработать запросы в соответствии с Вашим вариантом. Запросы сформулированы в терминах классов «Сотрудник» и «Отдел», которые используются в примере. Вам нужно перенести эти требования в Ваш вариант предметной области. При разработке запросов необходимо по возможности использовать функциональные возможности языка Python (list/dict comprehensions, функции высших порядков).

Вариант Б.

1. «Отдел» и «Сотрудник» связаны соотношением один-ко-многим. Выведите список всех связанных сотрудников и отделов, отсортированный по сотрудникам, сортировка по отделам произвольная.
2. «Отдел» и «Сотрудник» связаны соотношением один-ко-многим. Выведите список отделов с количеством сотрудников в каждом отделе, отсортированный по количеству сотрудников.
3. «Отдел» и «Сотрудник» связаны соотношением многие-ко-многим. Выведите список всех сотрудников, у которых фамилия заканчивается на «ов», и названия их отделов.

3	Водитель	Автопарк

Текст программы

```
from operator import itemgetter

"""Водитель"""
class Driver:
    def __init__(self, id, name, years_old, exper, id_transport):
        self.id = id
        self.name = name
        self.years_old = years_old
        self.exper = exper
        self.id_transport = id_transport
```

```python
"""Автопарк"""
class CarPark:
    def __init__(self, id, name, quanity_routes):
        self.id = id
        self.name = name
        self.quanity_routes = quanity_routes

class Driver_CarPark:
    def __init__(self, driver_id, carpark_id):
        self.driver_id = driver_id
        self.carpark_id = carpark_id


drivers = [
    Driver(1, "Emelyanov D.B.", 32, 3, 2),
    Driver(2, "Semenov E.Y", 47, 22, 6),
    Driver(3, "Dmitriev S.A", 26, 2, 17),
    Driver(4, "Pavlenko T.D", 54, 15, 13),
    Driver(5, "Ivanov P.A.", 40, 10, 3),
    Driver(6, "Petrov I.N.", 29, 5, 1),
    Driver(7, "Sidorov O.K.", 35, 7, 6),
    Driver(8, "Orlov V.M.", 30, 8, 2)
]

parks = [
    CarPark(1, "Guarantee", 4),
    CarPark(2, "Bus depot No. 9", 24),
    CarPark(3, "Enka", 12),
    CarPark(4, "Metro Depot", 15),
    CarPark(5, "City Transport", 18)
]

parks_drivers = [
    Driver_CarPark(1, 1),
    Driver_CarPark(2, 2),
    Driver_CarPark(3, 3),
    Driver_CarPark(3, 2),
    Driver_CarPark(4, 1),
    Driver_CarPark(5, 3),
    Driver_CarPark(6, 4),
    Driver_CarPark(7, 5),
    Driver_CarPark(8, 2)
]

def first_task(one_to_many):
    res_1 = sorted(one_to_many, key=itemgetter(0))
    return res_1

def second_task(one_to_many):
    temp_dict = {}
    for driver_name, driver_id_transport, park_name in one_to_many:
```

```
        if park_name in temp_dict:
            temp_dict[park_name] += 1
        else:
            temp_dict[park_name] = 1

    res_2 = [(park_name, count) for park_name, count in temp_dict.items()]
    res_2.sort(key=itemgetter(1), reverse=True)
    return res_2

def third_task(many_to_many, end_ch):
    res_3 = [(driver_name, park_name) for driver_name, driver_id_transport,
park_name in many_to_many if driver_name.split()[0].endswith(end_ch)]
    return res_3

def main():
    one_to_many = [(dr.name, dr.id_transport, cp.name)
                   for dr in drivers
                   for cp in parks
                   if dr.id_transport == cp.id]

    many_to_many_temp = [(dr.name, pd.driver_id, pd.carpark_id)
                         for dr in drivers
                         for pd in parks_drivers
                         if pd.driver_id == dr.id]

    many_to_many = [(driver_name, driver_id, cp.name)
                    for driver_name, driver_id, park_id in many_to_many_temp
                    for cp in parks if cp.id == park_id]

    print('Задание 1')
    print(first_task(one_to_many))

    print("\nЗадание 2")
    print(second_task(one_to_many))

    print("\nЗадание 3")
    print(third_task(many_to_many, 'ov'))

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

Вывод программы:

```
Задание 1
[('Emelyanov D.B.', 2, 'Bus depot No. 9'), ('Ivanov P.A.', 3, 'Enka'), ('Orlov V.M.', 2, 'Bus depot No. 9')
, ('Petrov I.N.', 1, 'Guarantee')]

Задание 2
[('Bus depot No. 9', 2), ('Enka', 1), ('Guarantee', 1)]

Задание 3
[('Emelyanov D.B.', 'Guarantee'), ('Semenov E.Y', 'Bus depot No. 9'), ('Ivanov P.A.', 'Enka'), ('Petrov I.N
.', 'Metro Depot'), ('Sidorov O.K.', 'City Transport'), ('Orlov V.M.', 'Bus depot No. 9')]
```