МГТУ им. Н.Э. Баумана

Кафедра «Системы обработки информации и управления»

Рубежный контроль №1
«Базовые компоненты интернет-технологий»

Студент группы ИУ5-31Б:
Большаков Георгий

Преподаватель кафедры ИУ5:
Гапанюк Юрий Евгеньевич

Москва, 2022

Вариант А. Предметная область 3.

1. «Водитель» и «Автопарк» связаны соотношением один-ко-многим. Выведите список всех водителей, у которых фамилия кончается на «ov», и названия их автопарков.
2. «Водитель» и «Автопарк» связаны соотношением один-ко-многим. Выведите список автопарков со средним рейтингом в каждом автопарке, отсортированный по среднему рейтингу.
3. «Водитель» и «Автопарк» связаны соотношением многие-ко-многим. Выведите список всех автопарком, у которых название начинается с буквы «Y», и список работающих в них водителей.

Листинг

```
from operator import itemgetter


class Driver:
    def __init__(self, Id, Name, Rating, IdDr):
        self.id = Id           # id водителя
        self.name = Name       # имя водителя
        self.rating = Rating   # рейтинг водителя
        self.ida = IdDr


class Autopark:
    def __init__(self, Id, Name):
        self.id = Id
        self.name = Name


class DrivOfAu: # связь автопарк - водитель
    def __init__(self, IdA, IdD):
        self.ida = IdA  #id автопарка
        self.idd = IdD  #id водителя


drivers = [
    Driver(1, 'Dubov', 4.3, 1),
    Driver(2, 'Zotov', 4.6, 2),
    Driver(3, 'Medvedev', 5.0, 3),
    Driver(4, 'Yudin', 4.87, 3),
    Driver(5, 'Volkov', 4.77, 3),
    Driver(6, 'Petrov', 4.23, 1),
    Driver(7, 'Markov', 4.54, 2),
]

autos = [
    Autopark(1, 'Uber'),
    Autopark(2, 'Yandex.Taxi'),
    Autopark(3, 'Yo-Taxi'),
    Autopark(4, 'Citymobil'),
    Autopark(5, 'Gett Taxi'),
    Autopark(6, 'Drive'),
]

# Связь автопарк - водитель
driverofauto = [
    DrivOfAu(1, 1),
    DrivOfAu(1, 6),
    DrivOfAu(2, 2),
    DrivOfAu(2, 7),
    DrivOfAu(3, 3),
    DrivOfAu(3, 4),
    DrivOfAu(3, 5),
    DrivOfAu(4, 1),
    DrivOfAu(4, 7),
    DrivOfAu(5, 2),
    DrivOfAu(5, 6),
    DrivOfAu(6, 3),
    DrivOfAu(6, 4),
    DrivOfAu(6, 5),
]


def main():
    # Соединение данных один-ко-многим
    one_to_many = [(d.name, d.rating, a.name)
                   for a in autos for d in drivers if d.ida == a.id]
    #print(one_to_many)
```

```python
# Соединение данных многие-ко-многим
many_to_many_temp = [(a.name, da.ida, da.idd)
                     for a in autos
                     for da in driverofauto
                     if a.id == da.ida]

many_to_many = [(d.name, d.rating, au_name)
                for au_name, auto_id, driver_id in many_to_many_temp
                for d in drivers
                if d.id == driver_id]


print('\n\nЗадание Д1')
res_1 = {}
# Перебираем все фамилии
for d in drivers:
    if 'ov' == d.name[-2:]:
        # Список фамилий на -ов + рейтинг + автопарк
        d_ov = list(filter(lambda i: i[0] == d.name, many_to_many))

        # Только автопарк водителя
        d_autos_names = [x for _, _, x in d_ov]     # [id, driver_name, autopark_name]

        # Добавляем результат в словарь
        # Ключ - фамилия, значение - список автопарков
        res_1[d.name] = d_autos_names
print(res_1)


print('\n\nЗадание Д2')
res_2_unsorted = []
# Перебираем все автопакри
for d in autos:
    # Список водителей в автопарке
    d_autos = list(filter(lambda i: i[2] == d.name, many_to_many))

    # Если автопарк не пустой
    if len(d_autos) > 0:
        # Рейтинг водителя в автопарке
        d_rating = [rating for _, rating, _ in d_autos]

        # Суммарный рейтинг водителя в автопарке
        d_rating_sum = sum(d_rating) \
        # Средний рейтинг
        d_rating_avg = '{:.2f}'.format(d_rating_sum / len(d_rating))

        res_2_unsorted.append((d.name, d_rating_avg))

# Сортировка по среднему рейтингу
res_2 = sorted(res_2_unsorted, key=itemgetter(1), reverse=True)
print(res_2)


print('\n\nЗадание Д3')
res_3 = {}
# Перебираем все автопарки
for d in autos:
    # Если автопарк начинается на Y
    if 'Y' == d.name[0]:
        # Список водителей автопарка
        d_autos = list(filter(lambda i: i[2] == d.name, many_to_many))

        # Только фамилия водителей
        d_autos_names = [x for x, _, _ in d_autos]

        # Добавляем результат в словарь
```

```
            # ключ - автопарк, значение - список фамилий
            res_3[d.name] = d_autos_names
    print(res_3, sep='\n', end='\n\n')

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

Результат выполнения:

```
Задание Д1
{'Dubov': ['Uber', 'Citymobil'], 'Zotov': ['Yandex.Taxi', 'Gett Taxi'], 'Volkov': ['Yo-Taxi', 'Drive'], 'Petrov': ['Uber', 'Gett Taxi'], 'Markov': ['Yandex.Taxi', 'Citymobil']}


Задание Д2
[('Yo-Taxi', '4.88'), ('Drive', '4.88'), ('Yandex.Taxi', '4.57'), ('Citymobil', '4.42'), ('Gett Taxi', '4.42'), ('Uber', '4.27')]


Задание Д3
{'Yandex.Taxi': ['Zotov', 'Markov'], 'Yo-Taxi': ['Medvedev', 'Yudin', 'Volkov']}
```

Задание Д1
{'Dubov': ['Uber', 'Citymobil'], 'Zotov': ['Yandex.Taxi', 'Gett Taxi'], 'Volkov': ['Yo-Taxi', 'Drive'], 'Petrov': ['Uber', 'Gett Taxi'], 'Markov': ['Yandex.Taxi', 'Citymobil']}

Задание Д2
[('Yo-Taxi', '4.88'), ('Drive', '4.88'), ('Yandex.Taxi', '4.57'), ('Citymobil', '4.42'), ('Gett Taxi', '4.42'), ('Uber', '4.27')]

Задание Д3
{'Yandex.Taxi': ['Zotov', 'Markov'], 'Yo-Taxi': ['Medvedev', 'Yudin', 'Volkov']}