Министерство науки и высшего образования Российской Федерации
Федеральное государственное бюджетное образовательное учреждение высшего образования «Московский государственный технический университет имени Н.Э. Баумана (национальный исследовательский университет)»
(МГТУ им. Н.Э. Баумана)

Факультет «Радиотехнический»

Кафедра ИУ5 «Системы обработки информации и управления»

Рубежный контроль №1

по дисциплине «Разработка интернет-приложений»

Выполнил:
студент группы РТ5-51Б
Н.Н. Васянин

Проверил:
к.т.н., доцент кафедры ИУ5
Ю.Е. Гапанюк

2021 г.

Задание:

1) Необходимо создать два класса данных в соответствии с Вашим вариантом предметной области, которые связаны отношениями один-ко-многим и многие-ко-многим.

Пример классов данных для предметной области Сотрудник-Отдел:

1. Класс «Сотрудник», содержащий поля:
   - ID записи о сотруднике;
   - Фамилия сотрудника;
   - Зарплата (количественный признак);
   - ID записи об отделе. (для реализации связи один-ко-многим)
2. Класс «Отдел», содержащий поля:
   - ID записи об отделе;
   - Наименование отдела.
3. (Для реализации связи многие-ко-многим) Класс «Сотрудники отдела», содержащий поля:
   - ID записи о сотруднике;
   - ID записи об отделе.

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).

Для реализации запроса №2 введите в класс, находящийся на стороне связи «много», произвольный количественный признак, например, «зарплата сотрудника».

Результатом рубежного контроля является документ в формате PDF, который содержит текст программы и результаты ее выполнения.

Вариант Е5

1. «Отдел» и «Сотрудник» связаны соотношением один-ко-многим. Выведите список всех отделов, у которых в названии присутствует слово «отдел», и список работающих в них сотрудников.
2. «Отдел» и «Сотрудник» связаны соотношением один-ко-многим. Выведите список отделов со средней зарплатой сотрудников в каждом отделе, отсортированный по средней зарплате. Средняя зарплата должна быть округлена до 2 знака после запятой (*отдельной функции*

вычисления среднего значения в Python нет, нужно использовать комбинацию функций вычисления суммы и количества значений; для округления необходимо использовать функцию https://docs.python.org/3/library/functions.html#round).

3. «Отдел» и «Сотрудник» связаны соотношением многие-ко-многим. Выведите список всех сотрудников, у которых фамилия начинается с буквы «А», и названия их отделов.

Текст программ

```
from operator import itemgetter


class Emp:
    """Музыкант"""

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


class Dep:
    """Оркестр"""

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


class EmpDep:
    """
    'Музыкант оркестр' для реализации
    связи многие-ко-многим
    """

    def __init__(self, dep_id, emp_id):
        self.dep_id = dep_id
        self.emp_id = emp_id


# Оркестры
deps = [
    Dep(1, 'симфонический оркестр'),
    Dep(2, 'эстрадный оркестр'),
    Dep(3, 'военный'),

    Dep(11, 'симфанический (другой) оркестр'),
    Dep(22, 'эстрадный (другой) оркестр'),
    Dep(33, '(другая) военный'),
]

# Музыканты
emps = [
    Emp(1, 'Артамонов', 25000, 1),
    Emp(2, 'Петров', 35000, 2),
    Emp(3, 'Иваненко', 45000, 3),
    Emp(4, 'Иванов', 35000, 3),
    Emp(5, 'Иванин', 25000, 3),
```

```
]

emps_deps = [
    EmpDep(1, 1),
    EmpDep(2, 2),
    EmpDep(3, 3),
    EmpDep(3, 4),
    EmpDep(3, 5),

    EmpDep(11, 1),
    EmpDep(22, 2),
    EmpDep(33, 3),
    EmpDep(33, 4),
    EmpDep(33, 5),
]


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

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

    # Соединение данных многие-ко-многим
    many_to_many_temp = [(d.name, ed.dep_id, ed.emp_id)
                         for d in deps
                         for ed in emps_deps
                         if d.id == ed.dep_id]

    many_to_many = [(e.fio, e.sal, dep_name)
                    for dep_name, dep_id, emp_id in many_to_many_temp
                    for e in emps
                    if e.id == emp_id]

    print('\nЗадание Е1')

    def taskone():
        b = []
        c = []
        for a in many_to_many:
            if "оркестр" in a[-1]:
                b.append(a[-1])
                c.append(a[0])
        print(b, '\n', c)

    taskone()

    print('\nЗадание Е2')
    res_12_unsorted = []
    for d in deps:
        count = 0
        d_emps = list(filter(lambda i: i[2] == d.name, one_to_many))
        if len(d_emps) > 0:
            d_sals = [sal for _, sal, _ in d_emps]
            count += 1
            d_sals_sum = sum(d_sals)
            aver_sum = round(d_sals_sum / count, 2)

            res_12_unsorted.append((d.name, aver_sum))
    res_12 = sorted(res_12_unsorted, key=itemgetter(1))
    print(res_12)
```

```
    print('\nЗадание E3')

    for d in many_to_many:
        if d[0].find('A') == 0:
            a = []
            c = []
            a.append(d[0])
            c.append(d[-1])
            print(a, c)


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

Экранные формы с примерами выполнения программы:

```
Задание E1
['симфонический оркестр', 'эстрадный оркестр', 'симфанический (другой) оркестр', 'эстрадный (другой) оркестр']
 ['Артамонов', 'Петров', 'Артамонов', 'Петров']

Задание E2
[('симфонический оркестр', 25000.0), ('эстрадный оркестр', 35000.0), ('военный', 105000.0)]

Задание E3
['Артамонов'] ['симфонический оркестр']
['Артамонов'] ['симфанический (другой) оркестр']
```