Московский государственный технический университет им. Н.Э. Баумана.

Факультет «Информатика и управление»

Кафедра «Системы обработки информации и управления»

Курс «Разработка интернет-приложений»

Отчет по рубежному контролю №1

Вариант Г5

Выполнил:
студент группы ИУ5-54
Драгун Илья

Подпись и дата:

Проверил:
преподаватель каф. ИУ5
Гапанюк Ю.Е.

Подпись и дата:

Москва, 2021 г.

Описание задания

Рубежный контроль представляет собой разработку программы на языке Python, которая выполняет следующие действия:

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 №5 введите в класс, находящийся на стороне связи «много», произвольный количественный признак, например, «зарплата сотрудника».

Текст программы

```
from operator import itemgetter


class Musician:
    """Музыкант"""
    def __init__(self, id, fio, age, orchestra_id):
        self.id = id
```

```
        self.fio = fio
        self.age = age
        self.orchestra_id = orchestra_id


class Orchestra:
    """Оркестр"""
    def __init__(self, id, name):
        self.id = id
        self.name = name


class MusicianOrchestra:
    """
    'Музыканты в оркестре' для реализации
    связи многие-ко-многим
    """
    def __init__(self, Musician_id, orchestra_id):
        self.Musician_id = Musician_id
        self.orchestra_id = orchestra_id


# Оркестры
orchestras = [
    Orchestra(1, 'Alouette La Peregrinacion 68 Paul Mauriat'),
    Orchestra(2, 'Симфонический оперный'),
    Orchestra(3, 'Гленна-Миллера'),
    Orchestra(4, 'André Léon Marie Nicolas Rieu'),
]

# Музыканты
musicians = [
    Musician(1, 'Андреев Леопольд Георгиевич', 56, 1),
    Musician(2, 'Бернем Бо', 35, 2),
    Musician(3, 'Монро Эшли', 40, 3),
    Musician(4, 'Пугач Владимир Витальевич', 45, 3),
    Musician(5, 'Трэвис Скотт', 28, 4),
]

musicians_orchestras = [
    MusicianOrchestra(1, 1),
    MusicianOrchestra(2, 2),
    MusicianOrchestra(3, 3),
    MusicianOrchestra(4, 4),
    MusicianOrchestra(5, 4),
]


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

    # Соединение данных один-ко-многим
    one_to_many = [(s.fio, s.age, k.name)
                   for k in orchestras
                   for s in musicians
                   if s.orchestra_id == k.id]

    # Соединение данных многие-ко-многим
    many_to_many_temp = [(k.name, sk.orchestra_id, sk.Musician_id)
                         for k in orchestras
                         for sk in musicians_orchestras
                         if k.id == sk.orchestra_id]
```

```
    many_to_many = [(s.fio, s.age, orchestra_name)
                    for orchestra_name, orchestra_id, Musician_id in
many_to_many_temp
                    for s in musicians if s.id == Musician_id]

    print('Задание Г1')
    res_11 = [(s.fio, k.name)
                for k in orchestras
                for s in musicians
                if (s.orchestra_id == k.id)&(k.name[:1] == "А")]
    print(res_11)


    print('\nЗадание Г2')
    res_12_unsorted = []

    for k in orchestras:
        k_musicians = list(filter(lambda i: i[2] == k.name, one_to_many))
        if len(k_musicians) > 0:
            k_ages = [age for _, age, _ in k_musicians]
            k_ages_max = max(k_ages)
            res_12_unsorted.append((k.name, k_ages_max))
    res_12 = sorted(res_12_unsorted, key=itemgetter(1), reverse=True)
    print(res_12)

    print('\nЗадание Г3')
    res_13 = sorted(many_to_many, key=itemgetter(2))
    print(res_13)



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

Анализ результатов

```
Задание Г1
[('Андреев Леопольд Георгиевич', 'Alouette La Peregrinacion 68 Paul Mauriat'), ('Трэвис Скотт', 'André Léon Marie Nicolas Rieu')]

Задание Г2
[('Alouette La Peregrinacion 68 Paul Mauriat', 56), ('Гленна-Миллера', 45), ('Симфонический оперный', 35), ('André Léon Marie Nicolas Rieu', 28)]

Задание Г3
[('Андреев Леопольд Георгиевич', 56, 'Alouette La Peregrinacion 68 Paul Mauriat'), ('Пугач Владимир Витальевич', 45, 'André Léon Marie Nicolas Rieu'), ('Трэви
```