Московский государственный технический университет им. Н.Э. Баумана

Факультет «Радиотехнический»
Кафедра «Системы обработки информации и управления»

Курс «Парадигмы и конструкции языков программирования»

Отчет по рубежному контролю № 1
Вариант № 2

Выполнил:
студент группы РТ5-31Б:
Баранов Даниил Романович
Подпись и дата:

Проверил:
преподаватель каф. ИУ5
Гапанюк Юрий Евгеньевич
Подпись и дата:

Москва, 2025 г.

Описание задания

Рубежный контроль представляет собой разработку программы на языке Python, которая выполняет следующие действия:
l) 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).
Для реализации запроса №2 введите в класс, находящийся на стороне связи «много», произвольный количественный признак, например, «зарплата сотрудника».

Текст программы

```
class schoolboy():

    def __init__(self, fio, studentID,age,classID):
        self.fio = fio
        self.age = age
        self.studentID = studentID
        self.classID = classID

class classrooms():

    def __init__(self, classID,nums):
        self.classID = classID
        self.nums = nums

class school_class():
    #для создания связи многие ко многим
    def __init__(self, studentID, classID):
        self.studentID = studentID
        self.classID = classID


schb = [
        schoolboy("Петоров Антон Олегович", 1, 14, 1),
        schoolboy("Козлов Андрей Данилович", 2, 15, 2),
        schoolboy("Андуряев Владислав Константинович", 3, 16, 3),
        schoolboy("Твердович Даниил Романович", 4, 17, 3),
        schoolboy("Критькова Екатерина Александровна", 5, 16, 2)
    ]  # школьники

clsr = [
        classrooms(1, 9),
        classrooms(2, 10),
        classrooms(3, 11)
```

```
    ]  # классы

schcl = [
        school_class(1, 1),
        school_class(2, 2),
        school_class(3, 3),
        school_class(4, 3),
        school_class(5, 2)
    ]  # доп класс для связи м:м


def counters(any_list, num_for_find):
    storage = 0
    for el in any_list:
        if el == num_for_find:
            storage += 1
    return storage


def finder_for_fio(any_dict):
    fior = dict()
    for item in any_dict:
        moment = item[0]
        classr = item[1]
        fcheck = moment.split()
        fio = fcheck[0]

        if (fio.endswith('ов')) or (fio.endswith('ова')):
            fior[moment] = classr
    return fior


def filters(anything):
    for i in anything:#вычленяем каждый элемент списка -> фио и класс
        return len(i.split()[0])#далее получаем фио, оно первое, и делаем сплит, тк строка, ->
получили список i в
    #котором есть ф и о -> берем 0 элменет, то есть 1, а именно фамиию, то что нужно,
сортировка по фамииям
    # и возвращаем его длину, и на этом программа кончается, тк return и до следущего i мы не
доходим, то есть
    # не обрабатываем номер класса

def main():

    one_to_many_first = [
        (st.fio, st.age,cls.nums)
        for st in schb
        for cls in clsr
        if st.classID == cls.classID]#связываем через id школльника и класс

    one_to_many_twice = [
        (cls.nums)
        for st in schb
        for cls in clsr
        if st.classID == cls.classID
    ]

    many_to_many_1 = [
        (st.fio, stcl.classID)
        for st in schb
```

```
        for stcl in schcl
        # должны совпдать тк при делении м/м в промеж звене возникает внешний ключ, это он и
есть + studentID проверку добавил для того чтобы отсечь дубликаты
        if (st.classID == stcl.classID) and (st.studentID == stcl.studentID)
    ]

    many_to_many_2 = [
        (fio, classroom.nums)  # обращаемся к атрибуту nums объекта classroom
        for fio, class_id in many_to_many_1
        for classroom in clsr
        if class_id == classroom.classID
    ]

    print("\n")
    print("Задание Б1(сортировка по фамилиям сотрудников):")

    sc_people = []

    for el in one_to_many_first:
        sc_people.append(el)
    rez = sorted(sc_people)

    for i in rez:
        print(i)

    print("\n")
    print("Задание Б2(сортировка по классам с меньшего по старший)(формат вывода-> класс,кол-
во людей):")

    rez_class = dict()

    counter = 0

    for i in one_to_many_twice:
        if counter == 0:
            i_old = i
            counter+=1
            rez_class[i] = counters(one_to_many_twice, i)
        else:
            if i_old==i:
                rez_class[i] = counters(one_to_many_twice,i)
            else:
                i_old = i
            counter += 1
    rez_class[i] = counters(one_to_many_twice, i)

    sc_class = sorted(list(rez_class.items()))
    for itemer in sc_class:
        print(itemer)
    """     for classroom,num_of_st in rez_class.items():
        print(f"Класс: {classroom} -> {num_of_st} чел.")#так же корректный вывод только через
словарь, более красивый
    """

    print("\n")
    print("Задание Б3(сортировка с фамилиям школьников, получение фамилий оканчивающихся на ов
+ их класс)(формат: фио школьника и его класс):")

    #Вариант сортировки №1 встроенный по букве фамилии
```

```
    #fio_rez = sorted(list(finder_for_fio(many_to_many_2).items()))#сортировка по первой букве фамилии
    # сортировка по первой букве фамилии

    # Вариант сортировки №2 самописный, по длине фамилии
    fio_rez = sorted(list(finder_for_fio(many_to_many_2).items()),key = filters)
    #это будет эквивалентно вызову функции filters(кажый из элементов м/м/2) за нас это делает функция sorted
    #она сама, автоматически подставит все необходимые компоненты, в нашем соучае элементы словаря
    #здесьь же , в скобках она переходит в список , и из него уже подставляются значения в фильтр
    for student in fio_rez:
        print(student)

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

Экранные формы с примерами выполнения программы

```
Задание Б1(сортировка по фамилиям сотрудников):
('Андуряев Владислав Константинович', 16, 11)
('Козлов Андрей Данилович', 15, 10)
('Критькова Екатерина Александровна', 16, 10)
('Петоров Антон Олегович', 14, 9)
('Твердович Даниил Романович', 17, 11)


Задание Б2(сортировка по классам с меньшего по старший)(формат вывода-> класс,кол-во людей):
(9, 1)
(10, 2)
(11, 2)


Задание Б3(сортировка с фамилиям школьников, получение фамилий оканчивающихся на ов + их класс)(формат: фио школьника и его класс):
('Козлов Андрей Данилович', 10)
('Петоров Антон Олегович', 9)
('Критькова Екатерина Александровна', 10)
PS D:\учеба\Python>
```