Группа ИУ5-32Б

Студент Астахов Иван

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

Таблица 1. Варианты предметной обла

№ варианта	Класс 1	Класс 2
1	Студент	Группа

Программа:

```
from operator import itemgetter
"""Студент"""
class Student:
    def __init__(self, id, name, age, id_group):
        self.id = id
        self.name = name
        self.age = age
        self.id_group = id_group
"""Группа"""
class Group:
    def __init__(self, id, name, course):
        self.id = id
        self.name = name
        self.course = course
class Student_Group:
    def __init__(self, student_id, group_id):
        self.student_id = student_id
        self.group_id = group_id
students = [
    Student(1, "Emelyanov D.B.", 20, 2),
    Student(2, "Semenov E.Y", 22, 6),
    Student(3, "Dmitriev S.A", 21, 17),
    Student(4, "Pavlenko T.D", 23, 13),
    Student(5, "Ivanov P.A.", 20, 3),
    Student(6, "Petrov I.N.", 21, 1),
    Student(7, "Sidorov O.K.", 22, 6),
    Student(8, "Orlov V.M.", 20, 2)
]
groups = [
    Group(1, "Group A", 1),
    Group(2, "Group B", 2),
    Group(3, "Group C", 3),
    Group(4, "Group D", 4),
    Group(5, "Group E", 5)
]
groups_students = [
    Student_Group(1, 1),
    Student_Group(2, 2),
    Student_Group(3, 3),
    Student_Group(3, 2),
    Student_Group(4, 1),
    Student_Group(5, 3),
    Student_Group(6, 4),
    Student_Group(7, 5),
    Student_Group(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 student_name, student_id_group, group_name in one_to_many:
        if group_name in temp_dict:
            temp_dict[group_name] += 1
        else:
            temp_dict[group_name] = 1

    res_2 = [(group_name, count) for group_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 = [(student_name, group_name) for student_name, student_id,
group_name in many_to_many if
             student_name.split()[0].endswith(end_ch)]
    return res_3
def main():
    one_to_many = [(st.name, st.id_group, gr.name)
                   for st in students
                   for gr in groups
                   if st.id_group == gr.id]

    many_to_many_temp = [(st.name, sg.student_id, sg.group_id)
                         for st in students
                         for sg in groups_students
                         if sg.student_id == st.id]

    many_to_many = [(student_name, student_id, gr.name)
                    for student_name, student_id, group_id in
many_to_many_temp
                    for gr in groups if gr.id == group_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, 'Group B'), ('Ivanov P.A.', 3, 'Group C'), ('Orlov V.M.', 2, 'Group B'), ('Petrov I.N.', 1, 'Group A')]

Задание 2
[('Group B', 2), ('Group C', 1), ('Group A', 1)]

Задание 3
[('Emelyanov D.B.', 'Group A'), ('Semenov E.Y', 'Group B'), ('Ivanov P.A.', 'Group C'), ('Petrov I.N.', 'Group D'), ('Sidorov O.K.', 'Group E'), ('Orlov V.M.', 'Group B')]
```