Московский государственный технический университет имени Н. Э. Баумана

Факультет Радиотехнический

Кафедра ИУ5 Системы обработки информации и управления

Отчет по рубежному контролю №1 по курсу Базовые компоненты интернет-технологий

7
(количество листов)

Вариант № 17 (E)

Исполнитель
студент группы РТ5-31б _____ Павлов С.Д.
“ 12 ” октября 2021 г.

Проверил
Доцент кафедры ИУ5 _____ Гапанюк Ю.Е.
“___” _____ 2021 г.

Москва, 2021 г.

Задание

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 No2 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.

Запросы Е

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

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 программы

Файл Rk1_Classes.py:

```
#Класс дирижера:
class Conductor:
    conductors_number = 0 #Счетчик для id
    def __init__(self, Orchestra_id=None, Name="", Sername="", Middle_name="",
Salary=None, Orchestra_name=""):
        self.conductor_id = Conductor.conductors_number
        Conductor.conductors_number+=1
        self.orchestra_id = Orchestra_id
        self.name         = Name
        self.sername      = Sername
        self.middle_name  = Middle_name
        self.salary       = Salary

    #Краткий вывод экземпляра:
    def show(self, Logic=True):
        line = self.name + " " + self.sername
        if Logic == True: print(line + " (id = " + str(self.conductor_id) + ",
оркестр_id = " + str(self.orchestra_id) + ")", end="")
        return line

#Класс оркестра:
class Orchestra:
    orchestras_number = 0 #Счетчик для id
    def __init__(self, Name=""):
        self.orchestra_id = Orchestra.orchestras_number
        Orchestra.orchestras_number+=1
        self.name = Name

    #Краткий вывод экземпляра:
    def show(self, Logic=True):
        line = "\"" +self.name + "\""
        if Logic == True: print(line + " (id = " + str(self.orchestra_id) +
")", end="")
        return line

#Класс дирижеров оркестров (для реализации связей и хранения данных):
class Storage:
    orchestra_dict  = {} #Словарь относительно оркестров
    orcestra_list   = [] #Списки оркестров и дирижеров
    conductors_list = [] #

    #Добавление новых дирижеров:
```

```
    def add_conductor(self, Conductors, Orchestra_id):
        conductors_of_orchestra = Storage.orchestra_dict.get(Orchestra_id,
None) #Поиск соответсвующего оркестра
        if conductors_of_orchestra != None:
            if isinstance(Conductors, list) == True: #Если такой имеется и
соблюден синтаксис добавляем необходимых дирижеров
                for i in range(len(Conductors)):
conductors_of_orchestra.append(Conductors[i].conductor_id) #в лист связей
                Storage.conductors_list.extend(Conductors)
#в хранилище дирижеров
            else: print("Ошибка синтаксиса!")
        else: print("Отсутсвует данный оркестр!")

    #Добавление новых оркестров:
    def add_orchestra(self, Orchestras):
        if isinstance(Orchestras, list) == True:
            for i in range(len(Orchestras)):
self.orchestra_dict[Orchestras[i].orchestra_id] = [] #в лист связей
            Storage.orcestra_list.extend(Orchestras)
#в хранилище оркестров
        else: print("Ошибка синтаксиса!")

    #Возвращение дирижера по id:
    def return_conductor_by_id(self, Conductor_id):
        try: return Storage.conductors_list[Conductor_id]
        except: return None

    #Возвращение id оркестра по его имени:
    def find_orcestra_by_name(Storage, Name):
        for i in range(len(Storage.orcestra_list)):
            if Storage.orcestra_list[i].name == Name: return i
        return None

    #Создание отсортированного по возрастанию списка средних зарплат в
оркестрах:
    def count_average_salary(self):
        sallary_range_list = []
        for i in range(len(Storage.orchestra_dict)):
            summ = 0;
            for i2 in range(len(Storage.orchestra_dict[i])):
                summ +=
Storage.conductors_list[Storage.orchestra_dict[i][i2]].salary #Сумма месячных
зарплат в этом оркестре
            Average = round(summ/len(Storage.orchestra_dict[i]), 2)
#Находим среднее
            sallary_range_list.append([Average, i])
#Добавляем его соответсвенно с id оркестра
        sallary_range_list.sort()
```

```
        return sallary_range_list
```

Файл main.py:

```
from Rk1_Classes import *

def main():
    S = Storage()

    #Создаем оркестры:
    S.add_orchestra([Orchestra("Малый симфонический"), Orchestra("Практикантов
(малый)"), Orchestra("Большой симфонический"), Orchestra("Очень малый
малый")])
    new_orchestra = Orchestra("Не знаю, как назвать :<")
    S.add_orchestra([new_orchestra])

    #Создаем дирижеров:
    S.add_conductor([Conductor(0, "Иван",    "Крылов", "Матвеевич",  250000),
Conductor(0, "Григорий", "Авдеев", "Артемович",    300000)],  0)
    S.add_conductor([Conductor(1, "Troll",   "Face",   "Memow",      15000),
Conductor(1, "V", "Анонимный",      "Anonimous",    1830000)], 1)
    S.add_conductor([Conductor(2, "Василий", "Шуткин", "Тотсамович", 257000),
Conductor(2, "Товарищ",  "Майор",  "Вездесущевич", 100)],     2)
    S.add_conductor([Conductor(S.find_orcestra_by_name("Очень малый малый"),
"Антон",     "Антонов",        "Михеевич",      248000)],
                    S.find_orcestra_by_name("Очень малый малый"))
    S.add_conductor([Conductor(S.find_orcestra_by_name("Очень малый малый"),
"Рептилоид", "Мировопорядков", "Иллюминатович", 248000)],
                    S.find_orcestra_by_name("Не знаю, как назвать :<"))

    #================================[E1]================================#
    workers_list = [] #Лист дирижеров оркестров, удовлетворяющих требованию

    #Выведем все оркестры, в названия которых входит слово "малый"(с учетом
регистра):
    print("Оркестры, содержащие в названии слово \"малый\" (с учетом
регистра):")
    for i in range(len(S.orcestra_list)):
        if "малый" in S.orcestra_list[i].name:
            S.orcestra_list[i].show(); print("; ", end ="")
            workers_list.extend(S.orchestra_dict[i])
    print("\n\nДирижеры этих оркестров:")

    #Выведем всех дирижеров этих оркестров:
    for i in range(len(workers_list)):
        (S.return_conductor_by_id(workers_list[i])).show(); print("; ",
end="")
```

```
    #================================[E2]================================#
    #Рассчитаем среднюю зарплату и занесем во вложенный список:
    print("\n\n\nОркестры по возрастанию средней зарплаты в них:")
    average_salaries_list = S.count_average_salary()

    #Выведем его:R
    for i in range(len(average_salaries_list)):
        orcestra_id = average_salaries_list[i][1]
        print("%-26s%-9s%-3s%6d%7s" %
(S.orcestra_list[orcestra_id].show(False), "(id = " + str(orcestra_id) + ")",
"->", average_salaries_list[i][0], "рублей"))

    #================================[E3]================================#
    #Выведем список всех дирижеров, чья фамилия начинается на А и названия их
отделов:
    print("\n\n\nСписок всех дирижеров, чья фамилия начинается на А и их
названия оркестров:")
    for i in range(len(S.conductors_list)):
        if "A" in S.conductors_list[i].sername:
            orcestra_id = S.conductors_list[i].orchestra_id
            print("%-16s%-9s%-9s%6s" % (S.conductors_list[i].show(False), "(id
= " + str(S.conductors_list[i].conductor_id) + ")",
                                        "оркестр:",
S.orcestra_list[orcestra_id].name))

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

Результат выполнения программы:

```
C:\Program Files (x86)\Microsoft Visual Studio\Shared\Python37_64\python.exe
Оркестры, содержащие в названии слово "малый" (с учетом регистра):
"Практикантов (малый)" (id = 1); "Очень малый малый" (id = 3);

Дирижеры этих оркестров:
Troll Face (id = 2, оркестр_id = 1); V Анонимный (id = 3, оркестр_id = 1); Антон Антонов (id = 6, оркестр_id = 3);


Оркестры по возрастанию средней зарплаты в них:
"Большой симфонический"   (id = 2) -> 128550 рублей
"Очень малый малый"       (id = 3) -> 248000 рублей
"Не знаю, как назвать :<" (id = 4) -> 248000 рублей
"Малый симфонический"     (id = 0) -> 275000 рублей
"Практикантов (малый)"    (id = 1) -> 922500 рублей



Список всех дирижеров, чья фамилия начинается на А и их названия оркестров:
Григорий Авдеев (id = 1) оркестр: Малый симфонический
V Анонимный     (id = 3) оркестр: Практикантов (малый)
Антон Антонов   (id = 6) оркестр: Очень малый малый
Press any key to continue . . .
```