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

**Факультет ИУ Кафедра ИУ5**

**Курс «Основы информатики» Отчет по Рубежному контролю №1 Вариант В4**

Выполнил студент группы ИУ5-33Б: Козлов А. А.

Подпись и дата:

Проверил преподаватель каф.: Гапанюк Ю. Е.

Подпись и дата:

Москва, 2024 г

**Постановка задачи**

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)

* 1. Класс «Отдел», содержащий поля:
     + ID записи об отделе;
     + Наименование отдела.
  2. (Для реализации связи многие-ко-многим) Класс «Сотрудники отдела», содержащий поля:
     + ID записи о сотруднике;
     + ID записи об отделе.

1. Необходимо создать списки объектов классов, содержащих тестовые данные (3-5 записей), таким образом, чтобы первичные и вторичные ключи соответствующих записей были связаны по идентификаторам.
2. 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 В.**

* 1. «Отдел» и «Сотрудник» связаны соотношением один-ко-многим. Выведите список всех сотрудников, у которых фамилия начинается с буквы «А», и названия их отделов.
  2. «Отдел» и «Сотрудник» связаны соотношением один-ко-многим. Выведите список отделов с минимальной зарплатой сотрудников в каждом отделе, отсортированный по минимальной зарплате.
  3. «Отдел» и «Сотрудник» связаны соотношением многие-ко-многим.

Выведите список всех связанных сотрудников и отделов, отсортированный по сотрудникам, сортировка по отделам произвольная.

* 1. **Таблица 1.** Варианты предметной области

|  |  |  |
| --- | --- | --- |
| **Вариант** | **Класс 1** | **Класс 2** |
| 4 | Компьютер | Дисплейный класс |

**Текст программы**

from operator import itemgetter

class Computer():

def init (self, id, name, cost, class\_id): self.id = id

self.name = name self.cost = cost self.class\_id = class\_id

class DisClass():

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

self.name = name

class CompDC():

def init (self, class\_id, comp\_id): self.class\_id = class\_id self.comp\_id = comp\_id

comps = [

DisClass(1, '312л'),

DisClass(2, '313л'),

DisClass(3, '314л'),

DisClass(4, '315л'),

DisClass(5, '316л'),]

comp\_dc = [

CompDC(1, 1),

CompDC(2, 2),

CompDC(3, 3),

CompDC(3, 4),

CompDC(3, 5),

def main():

one\_to\_many = [(c.name, c.cost, dc.name)

for dc in dis\_classes for c in comps

if c.class\_id == dc.id]

|  |  |  |  |
| --- | --- | --- | --- |
| Computer(1, | 'A123', | 100, | 1), |
| Computer(2, | 'B123', | 200, | 2), |
| Computer(3, | 'A233', | 150, | 3), |
| Computer(4, | 'D123', | 200, | 4), |
| Computer(5,  dis\_classes = [ | 'E123', | 300, | 5),] |

|  |  |
| --- | --- |
| CompDC(4, | 1), |
| CompDC(4, | 2), |
| CompDC(4, | 3), |
| CompDC(5, | 4), |
| CompDC(5, | 5),] |

many\_to\_many\_temp = [(dc.name, co.class\_id, co.comp\_id) for dc in dis\_classes

for co in comp\_dc

if dc.id == co.class\_id]

many\_to\_many = [(c.name, c.cost, class\_name)

for class\_name, class\_id, comp\_id in many\_to\_many\_temp for c in comps if c.id == comp\_id]

print('Задание B1') '''

«дисплейный класс» и «компьютер» связаны соотношением один-ко-многим. Выведите список всех компьютеров, у которых название начинается с буквы «А», и названия их дисплейных классов.

'''

resultB1 = sorted(x for x in one\_to\_many if x[0].startswith('A')) for i in resultB1:

print(f'Название: {i[0]}, класс: {i[2]}')

print('\nЗадание B2') '''

«дисплейный класс» и «компьютер» связаны соотношением один-ко-многим. Выведите список классов с минимальной ценой за компьютер в каждом дисплейном классе, отсортированный по минимальной цене.

'''

resultB2 = {}

for class\_name in set(x[2] for x in one\_to\_many):

min\_cost = min(x[1] for x in one\_to\_many if x[2] == class\_name) resultB2[class\_name] = min\_cost

resultB2 = sorted(resultB2.items(), key=itemgetter(1)) for i in resultB2:

print(f'Класс: {i[0]}, минимальная стоимость: {i[1]}')

print('\nЗадание B3') '''

«дисплейный класс» и «компьютер» связаны соотношением многие-ко-многим. Выведите список всех связанных компьютеров и дисплейных классов, отсортированный по компьютерам, сортировка по дисплейным классам произвольная.

'''

resultB3 = sorted(many\_to\_many, key=itemgetter(0)) for i in resultB3:

print(f'Название: {i[0]}, класс: {i[2]}')

main()

**Анализ результатов**

