МГТУ им. Н.Э. Баумана

Отчёт по рубежному контролю №1  
по курсу «Базовые компоненты и интернет-технологии»

Вариант 12.

Руководитель  
Гапанюк Ю.Е.  
28.10.2022

Студент группы ИУ5-33Б  
Поляков Д.Д.

28.10.2022

2022 г.

Полученное задание:

Необходимо создать два класса данных в соответствии с Вашим вариантом предметной области, которые связаны отношениями один-ко-многим и многие-ко-многим.

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

Необходимо разработать запросы в соответствии с Вашим вариантом.

Предметная область: класс\_1 – Язык программирования, класс\_2 – Среда разработки, вариант запросов: В.

Запросы:

1. «Язык программирования» и «Инструмент разработки» связаны соотношением один-ко-многим. Выведите список всех инструментов разработки, у которых название начинается с буквы «А», и названия их языков программирования.
2. «Язык программирования» и «Инструмент разработки» связаны соотношением один-ко-многим. Выведите список языков с минимальным количеством пользователей инструмента в каждом языке, отсортированные по минимальным пользователям.
3. «Язык программирования» и «Инструмент разработки» связаны соотношением многие-ко-многим. Выведите список всех связанных языков и инструментов, отсортированный по инструментам, сортировка по языкам произвольная.

Текст программы:

*from* operator *import* itemgetter  
  
  
*class* Programming\_language:  
 *"""Сотрудник"""  
  
 def* \_\_init\_\_(*self*, id, name):  
 *self*.id = id  
 *self*.name = name  
  
  
*class* Development\_tool:  
 *"""Отдел"""  
  
 def* \_\_init\_\_(*self*, id, name, users\_count, programming\_language\_id):  
 *self*.id = id  
 *self*.name = name  
 *self*.users\_count = users\_count  
 *self*.programming\_language\_id = programming\_language\_id  
  
  
*class* Programming\_languageDevelopment\_tool:  
 *"""  
 'Сотрудники отдела' для реализации  
 связи многие-ко-многим  
 """  
  
 def* \_\_init\_\_(*self*, Programming\_language\_id, Development\_tool\_id):  
 *self*.Development\_tool\_id = Development\_tool\_id  
 *self*.Programming\_language\_id = Programming\_language\_id  
  
  
*# Отделы*Development\_tools = [  
 Development\_tool(1, 'CLion', 75000, 1),  
 Development\_tool(2, 'gdb', 100000, 1),  
  
 Development\_tool(3, 'Visual Studio', 85000, 2),  
 Development\_tool(4, '.NET core', 50000, 2),  
 Development\_tool(5, 'ASP.NET', 45000, 2),  
 Development\_tool(6, 'Entity Framework', 35000, 2),  
  
 Development\_tool(7, 'Angular', 100000, 3),  
 Development\_tool(8, 'React', 120000, 3),  
 Development\_tool(9, 'NodeJS', 150000, 3),  
 Development\_tool(10, 'WebStorm', 75000, 3),  
  
 Development\_tool(11, 'PyCharm', 87500, 4),  
 Development\_tool(12, 'Django', 50000, 4),  
 Development\_tool(13, 'Tensor Flow', 35000, 4),  
  
 Development\_tool(14, 'Visual Studio Code', 187000, 1),  
  
 Development\_tool(15, 'A+ PROGRAMMING IDE', 10, 5),  
]  
  
*# Сотрудники*Programming\_languages = [  
 Programming\_language(1, 'C++', ),  
 Programming\_language(2, 'C#', ),  
 Programming\_language(3, 'JavaScript', ),  
 Programming\_language(4, 'Python', ),  
 Programming\_language(5, 'A+', ),  
  
]  
  
Programming\_languages\_Development\_tools = [  
 Programming\_languageDevelopment\_tool(1, 1),  
 Programming\_languageDevelopment\_tool(1, 2),  
 Programming\_languageDevelopment\_tool(1, 14),  
  
 Programming\_languageDevelopment\_tool(2, 3),  
 Programming\_languageDevelopment\_tool(2, 4),  
 Programming\_languageDevelopment\_tool(2, 5),  
 Programming\_languageDevelopment\_tool(2, 6),  
 Programming\_languageDevelopment\_tool(2, 14),  
  
 Programming\_languageDevelopment\_tool(3, 7),  
 Programming\_languageDevelopment\_tool(3, 8),  
 Programming\_languageDevelopment\_tool(3, 9),  
 Programming\_languageDevelopment\_tool(3, 10),  
 Programming\_languageDevelopment\_tool(3, 14),  
  
 Programming\_languageDevelopment\_tool(4, 11),  
 Programming\_languageDevelopment\_tool(4, 12),  
 Programming\_languageDevelopment\_tool(4, 13),  
 Programming\_languageDevelopment\_tool(4, 14),  
  
 Programming\_languageDevelopment\_tool(5, 14),  
 Programming\_languageDevelopment\_tool(5, 15),  
]  
  
  
*def* main():  
 *"""Основная функция"""  
  
 # Соединение данных один-ко-многим* one\_to\_many = [(d.name, e.name, e.users\_count)  
 *for* d *in* Programming\_languages  
 *for* e *in* Development\_tools  
 *if* e.programming\_language\_id == d.id]  
  
 *# Соединение данных многие-ко-многим* many\_to\_many\_temp = [(d.name, pl.Programming\_language\_id, pl.Development\_tool\_id)  
 *for* d *in* Programming\_languages  
 *for* pl *in* Programming\_languages\_Development\_tools  
 *if* d.id == pl.Programming\_language\_id]  
  
 many\_to\_many = [(programming\_language\_name, e.name)  
 *for* programming\_language\_name, Programming\_language\_id, Development\_tool\_id *in* many\_to\_many\_temp  
 *for* e *in* Development\_tools *if* e.id == Development\_tool\_id]  
  
 print('Задание В1')  
 res\_11 = list(filter(*lambda* i: (i[0][0] == 'A' *or* i[0][0] == 'a'), one\_to\_many))  
 print(res\_11)  
  
 print('\nЗадание В2')  
 res\_2 = []  
 *for* pl *in* Programming\_languages:  
 d\_tmp = list((pl.name, d.users\_count) *for* d *in* Development\_tools *if* d.programming\_language\_id == pl.id)  
 *if* len(d\_tmp) > 0:  
 res\_2.append(min(d\_tmp, key=*lambda* i: i[1]))  
 tmp = sorted(res\_2, key=itemgetter(1))  
 print(tmp)  
  
 print('\nЗадание В3')  
 *# Перебираем все отделы* many\_to\_many.sort(key=*lambda* i: i[1])  
 print(many\_to\_many)  
  
  
*if* \_\_name\_\_ == '\_\_main\_\_':  
 main()

Результаты выполнения:

