Московский государственный технический университет им. Н.Э. Баумана

Факультет «Информатика и системы управления»
Кафедра ИУ5 «Системы обработки информации и управления»

Курс «Парадигмы и конструкции языков программирования»

Отчет по РК №1
Вариант запросов Г
Вариант предметной области 27

Выполнил:
студент группы ИУ5-33Б
Драчев Иван

Проверил:
преподаватель каф. ИУ5
Гапанюк Ю. Е.

Москва, 2025 г.

Запросы

1) “Преподаватель” и “Учебный курс” связаны отношением один-ко-многим. Выберите список преподавателей с ученой степенью и курсы, на которых они ведут
2) “Преподаватель” и “Учебный курс” связаны отношением один-ко-многим. Выберите учебных курсов, на которых есть преподаватели со степенью кандидата
3) “Преподаватель” и “Учебный курс” связаны отношением многие-ко-многим. Выберите список всех преподавателей и учебных курсов в отсортированном виде

Листинг

```python
from enum import Enum


class Degree(Enum):
    doctor = 'доктор'
    candidate = 'кандидат'
    scientist = 'ученый'


class Department:
    __slots__ = ('id', 'name', 'faculty')

    def __init__(self, id: int, name: str, faculty: str) -> None:
        self.id: int = id
        self.name: str = name
        self.faculty: str = faculty


class Teacher:
    __slots__ = ('id', 'name', 'academic_degree', 'department_id')

    def __init__(self, id: int, name: str, academic_degree: Degree,
department_id: int) -> None:
        self.id: int = id
        self.name: str = name
        self.academic_degree: Degree = academic_degree
        self.department_id: int = department_id


class CourseAssignment:
    __slots__ = ('department_id', 'teacher_id')

    def __init__(self, department_id: int, teacher_id: int) -> None:
        self.department_id: int = department_id
        self.teacher_id: int = teacher_id


departments: list[Department] = [
    Department(1, 'Математика', 'ФН'),
    Department(2, 'Физика', 'ФН'),
    Department(3, 'Программирование', 'ИУ'),
    Department(4, 'История', 'Гуманитарный'),
    Department(5, 'Литература', 'Гуманитарный'),
    Department(6, 'Химия', 'Естественных наук')
]
```

```python
teachers: list[Teacher] = [
    # основные кафедры для препрдавателей

    # математика
    Teacher(1, 'Иванов', Degree.scientist, 1),
    # математика
    Teacher(2, 'Петрова', Degree.candidate, 1),
    # физика
    Teacher(3, 'Овечкин', Degree.scientist, 2),
    # история
    Teacher(4, 'Козлова', Degree.candidate, 4),
    # программирование
    Teacher(5, 'Николаев', Degree.doctor, 3)
]

# один преподаватель может вести на в нескольких кафедрах
course_assignments: list[CourseAssignment] = [
    CourseAssignment(1, 1),

    CourseAssignment(1, 2),
    CourseAssignment(3, 2),
    CourseAssignment(5, 2),

    CourseAssignment(2, 3),
    CourseAssignment(6, 3),

    CourseAssignment(4, 4),
    CourseAssignment(3, 4),

    CourseAssignment(3, 5),
    CourseAssignment(4, 5),
]


def main() -> None:
    # связь один ко многим
    one_to_many: list[tuple[str, Degree, str]] = [
        (teacher.name, teacher.academic_degree, department.name)
        for teacher in teachers
        for assignment in course_assignments
        for department in departments
        if teacher.id == assignment.teacher_id and department.id ==
assignment.department_id
    ]

    temp: list[tuple[str, int, int]] = [
        (department.name, ca.department_id, ca.teacher_id)
        for department in departments
        for ca in course_assignments
        if department.id == ca.department_id
    ]

    # связь многие ко многим
    many_to_many: list[tuple[str, Degree, str]] = [
        (teacher.name, teacher.academic_degree, department_name)
        for department_name, _, teacher_id in temp
        for teacher in teachers if teacher.id == teacher_id
    ]

    print('Задание 1')
    first_res: list[tuple[str, str]] = find_doctors_of_science(one_to_many)
    print('Преподаватели с ученой степенью:')
    for teacher_name, department_name in first_res:
```

```
        print(f'{teacher_name} - {department_name}')
    print()

    print('Задание 2')
    second_res: list[tuple[str, str]
                    ] = find_departments_with_candidates(one_to_many)
    print('Кафедры с преподавателем кандидатами:')
    for department_name, teacher_name in second_res:
        print(f'{department_name}: {teacher_name}')
    print()

    print('Задание 3')
    third_res = sorted(many_to_many, key=lambda x: (x[2], x[0]))
    print('Все связанные преподаватели и кафедры:')
    for teacher_name, degree, department_name in third_res:
        print(f'{department_name}: {teacher_name} - {degree.value}')


def find_doctors_of_science(one_to_many: list[tuple[str, Degree, str]]) ->
list[tuple[str, str]]:
    res: list[tuple[str, str]] = [
        (teacher_name, department_name)
        for teacher_name, degree, department_name in one_to_many
        if degree.value == Degree.scientist.value
    ]

    return res


def find_departments_with_candidates(one_to_many: list[tuple[str, Degree,
str]]) -> list[tuple[str, str]]:
    department_candidates_dict: dict[str, list[str]] = {}

    for teacher_name, degree, department_name in one_to_many:
        if degree == Degree.candidate:
            if department_name not in department_candidates_dict:
                department_candidates_dict[department_name] = []
            department_candidates_dict[department_name].append(teacher_name)

    res: list[tuple[str, str]] = []
    for department_name, candidates in
sorted(department_candidates_dict.items()):
        candidates_str = ", ".join(candidates)
        res.append((department_name, candidates_str))

    return res


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

Скриншот работы приложения

```
ivan@ivancomp ~/p/p/rk1 (main)> p main.py
Задание 1
Преподаватели с ученой степенью:
Иванов - Математика
Овечкин - Физика
Овечкин - Химия

Задание 2
Кафедры с преподавателем кандидатами:
История: Козлова
Литература: Петрова
Математика: Петрова
Программирование: Петрова, Козлова

Задание 3
Все связанные преподаватели и кафедры:
История: Козлова - кандидат
История: Николаев - доктор
Литература: Петрова - кандидат
Математика: Иванов - ученый
Математика: Петрова - кандидат
Программирование: Козлова - кандидат
Программирование: Николаев - доктор
Программирование: Петрова - кандидат
Физика: Овечкин - ученый
Химия: Овечкин - ученый
ivan@ivancomp ~/p/p/rk1 (main)>
```