

**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 Б
Вариант предметной области 34**

**Выполнил:
студент группы ИУ5-33Б
Маргорина Ирина**

**Проверил:
преподаватель каф. ИУ5
Гапанюк Ю. Е.**

Москва, 2025 г.

Листинг кода

```
from operator import itemgetter

class StoredProcedure:
    def __init__(self, id, name, execution_time, db_id):
        self.id = id
        self.name = name
        self.execution_time = execution_time
        self.db_id = db_id

class Database:
    def __init__(self, id, name):
        self.id = id
        self.name = name

class ProcedureDatabase:
    def __init__(self, db_id, proc_id):
        self.db_id = db_id
        self.proc_id = proc_id

# Тестовые данные
databases = [
    Database(1, 'Основная база'),
    Database(2, 'Архивная база'),
    Database(3, 'База отчетов'),
    Database(4, 'Тестовая база'),
]

procedures = [
    StoredProcedure(1, 'расчет_отчета', 150, 1),
    StoredProcedure(2, 'резервное_копирование', 300, 2),
    StoredProcedure(3, 'генерация_статистики', 200, 3),
    StoredProcedure(4, 'очистка_данных', 250, 3),
    StoredProcedure(5, 'проверка_ввода', 100, 3),
    StoredProcedure(6, 'формирование_отчетов', 180, 1),
]

# Связь многие-ко-многим
procedures_databases = [
    ProcedureDatabase(1, 1),
    ProcedureDatabase(2, 2),
    ProcedureDatabase(3, 3),
    ProcedureDatabase(3, 4),
    ProcedureDatabase(3, 5),
    ProcedureDatabase(4, 1),
    ProcedureDatabase(4, 3),
    ProcedureDatabase(1, 6),
    ProcedureDatabase(3, 6),
]

def main():
    # Соединение данных один-ко-многим
    one_to_many = [(p.name, p.execution_time, d.name)
                    for d in databases
                    for p in procedures
                    if p.db_id == d.id]

    # Соединение данных многие-ко-многим
```

```

many_to_many_temp = [(d.name, pd.db_id, pd.proc_id)
                      for d in databases
                      for pd in procedures_databases
                      if d.id == pd.db_id]

many_to_many = [(p.name, p.execution_time, db_name)
                 for db_name, db_id, proc_id in many_to_many_temp
                 for p in procedures if p.id == proc_id]

print('Задание 1')
res_1 = sorted(one_to_many, key=itemgetter(0))
for item in res_1:
    print(f"Процедура: {item[0]}, Время выполнения: {item[1]}ms, БД: {item[2]}")

print('\nЗадание 2')
res_2_unsorted = []
for d in databases:
    procedure_count = 0
    for proc in procedures:
        if proc.db_id == d.id:
            procedure_count += 1

    if procedure_count > 0:
        res_2_unsorted.append((d.name, procedure_count))

res_2 = sorted(res_2_unsorted, key=itemgetter(1))
for item in res_2:
    print(f"БД: {item[0]}, Количество процедур: {item[1]}")

print('\nЗадание 3')
res_3 = {}
for proc_name, exec_time, db_name in many_to_many:
    if proc_name.endswith('ов'):
        if proc_name not in res_3:
            res_3[proc_name] = []
        res_3[proc_name].append(db_name)

for proc_name, db_list in res_3.items():
    print(f"Процедура '{proc_name}' работает в БД: {' '.join(db_list)}")

if __name__ == '__main__':
    main()

```

Скриншот работы приложения:

Задание 1

Процедура: генерация_статистики, Время выполнения: 200ms, БД: База отчетов

Процедура: очистка_данных, Время выполнения: 250ms, БД: База отчетов

Процедура: проверка_ввода, Время выполнения: 100ms, БД: База отчетов

Процедура: расчет_отчета, Время выполнения: 150ms, БД: Основная база

Процедура: резервное_копирование, Время выполнения: 300ms, БД: Архивная база

Процедура: формирование_отчетов, Время выполнения: 180ms, БД: Основная база

Задание 2

БД: Архивная база, Количество процедур: 1

БД: Основная база, Количество процедур: 2

БД: База отчетов, Количество процедур: 3

Задание 3

Процедура 'формирование_отчетов' работает в БД: Основная база, База отчетов