Санкт-Петербургский Политехнический Университет Петра Великого
Физико-механический институт
Высшая школа прикладной математики и вычислительной физики

# Отчет по курсу «базы данных»

Студент: Мелко Т. А.

Группа: 5030102/20003

Преподаватель: Крашенинников Сергей Вениаминович

## *Формулировка задания*

### Исходное задание:

2. Институт

Перечень специальностей (1 специальность - 1 группа). Полное обучение — 1 год (2 семестра). Специальность: название, список дисциплин с указанием продолжительности (в семестрах) и с распределением по семестрам. Дисциплина: название, зачет/экз (некоторые дисциплины могут одновременно находиться в списках дисциплин для разных специальностей). Списки студентов по группам. Списки преподавателей по дисциплинам (одну дисциплину могут преподавать разные преподаватели). Потоковых занятий нет, каждое занятие проводится для одной группы. Нагрузка преподавателей ограничена двумя группами в семестр.
Требуется:
* Поддержка составления расписания (для каждой группы определить/переопределить преподавателей на семестр)
> Поддержка сдачи сессии.
* Поддержка выпуска отчисления студентов (отчисление — при одной двойке или незачете)
* Поддержка истории института по годам
* Поддержка восстановления студентов (всегда с осени)
> Поддержка запросов: список отчисленных студентов по итогам последней сессии; список
злобных преподавателей (ставящих двойки и незачеты)

### Дополнительные соглашения предметной области:

Будем использовать для реализации задания фреймворк на Python – Django. В Django работа с базами данными реализовано с помощью ORM. Будем использовать встроенные функции для работы с базой данных.

### Итоговое задание:

- Спроектировать и реализовать информационную систему, соответствующую исходному заданию с учетом дополнительных соглашений предметной области

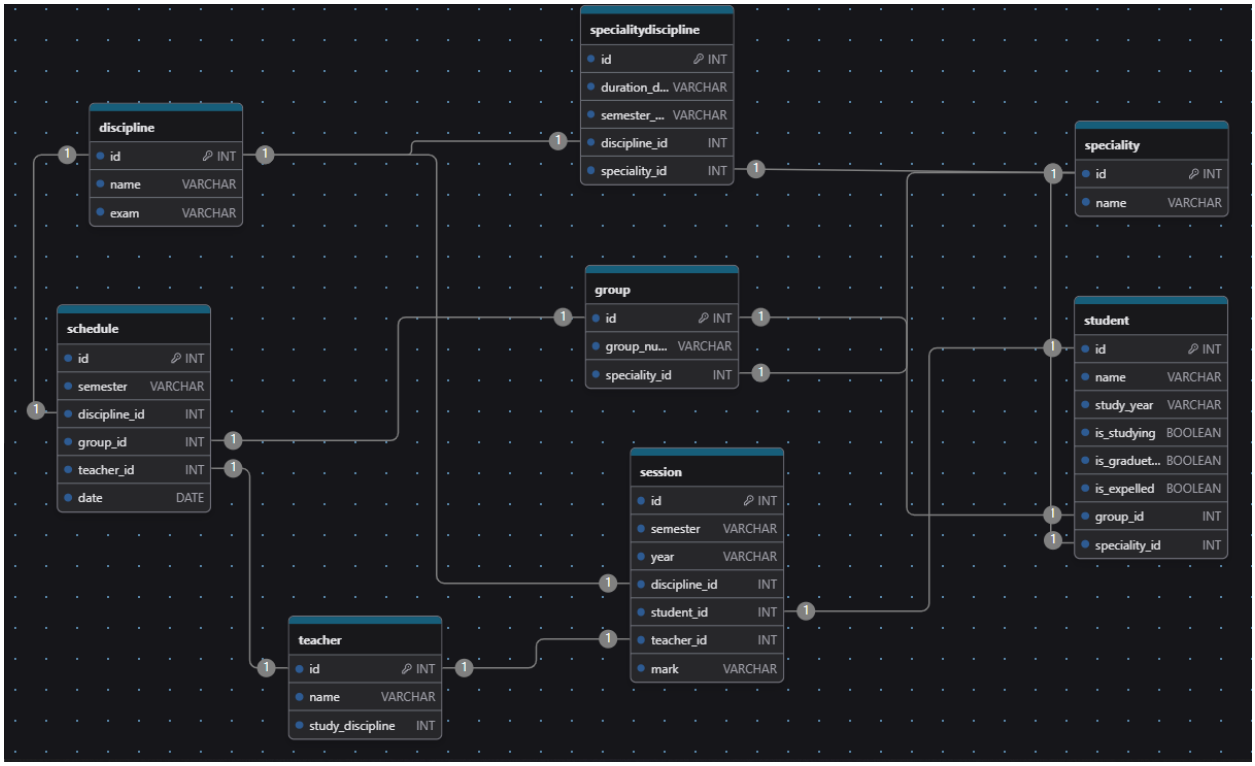
- Добавить поддержку стандартных операций:
    1. Поиск записи по ключевому полю
    2. Поиск записи по не ключевому полю
    3. Поиск записи по маске
    4. Добавление записи
    5. Добавление группы записей
    6. Изменение записи (определение изменяемой записи по ключевому полю)
    7. Изменение записи (определение изменяемой записи по не ключевому полю)
    8. Удаление записи (определение удаляемой записи по ключевому полю)
    9. Удаление записи (определение удаляемой записи по не ключевому полю)
    10. Удаление группы записей
    11. Сжатие базы данных (после удаления из БД 200 строк)
    12. Сжатие базы данных (после удаления, в результате которого в БД остается 200 строк)

- Замерить время выполнения стандартных операций для таблиц, содержащих 1000, 10000, 100000 записей.
- Сделать выводы.

## *Используемый программный инструментарий*

Django 5.0.4
Faker 24.11.0
SQLite

## *Схема базы данных*




## *Реализация стандартных операций*

**Поиск по ключевому полю, по не-ключевому полю, по маске**

```
search_query = request.GET.get('search', '')

    if search_query:
        objects = objects.filter(
            Q(id__icontains=search_query) |
            Q(semester__semester__icontains=search_query) |
            Q(group_group__icontains=search_query) |
            Q(teacher_teacher__icontains=search_query) |
            Q(discipline_discipline__icontains=search_query) |
            Q(date__icontains=search_query)
        )
```

**Добавление записи**

```
Student.objects.create(
        name = "Timofey",
        group = 1,
        speciality = "Программная инженерия",
        study_year = 2024
    )
```

**Добавление группы записей**

```
Student.objects.bulk_create([
        Student(
            name = "Timofey",
            group = 1,
            speciality = "Программная инженерия",
            study_year = 2024
        ),
        Student(
            name = "Egor",
            group = 1,
            speciality = "Программная инженерия",
            study_year = 2024
        )
    ])
```

**Изменение записи**

```
if request.method == 'POST':
        form = ScheduleForm(request.POST, instance=schedule)
        if form.is_valid():
            form.save()
            messages.success(request, 'Расписание успешно изменено')
```

**Удаление записи**

```
Student.objects.filter(id = choice).delete()
```

**Удаление группы записей**

```
Student.objects.filter(speciality = "Программная инженерия").delete()
```

**Сжатие базы данных**

```
tmp = Student.objects.all().order_by('-id')[:200]
    tmp.delete()
```

## *Сравнение временных затрат при реализации стандартных операций*

…

| Число записей | 1000 | 10000 | 100000 |
|---|---|---|---|
| Поиск по ключевому полю | 0.0008913850784301757 | 0.0005851697921752930 | 0.0005590534210205078 |
| Поиск по не ключевому полю | 0.0008440780639648438 | 0.0006331348419189453 | 0.0006280303001403809 |
| Поиск по маске | 0.0007984852790832 52 | 0.0007778954505920410 | 0.0006324148178100586 |
| Добавление записи | 0.0052496528625488828 | 0.0045626664161682130 | 0.0045673251152038570 |
| Добавление группы записей | 0.0047575807571411114 | 0.0047450685501098630 | 0.0047514653205871580 |
| Изменение записи (определение изменяемой записи по ключевому полю) | 0.0027878451347351074 | 0.0028661036491394040 | 0.0028516626358032226 |
| Изменение записи (определение изменяемой записи по не ключевому полю) | 0.0030649638175964354 | 0.0030742979049682620 | 0.0030376744270324708 |
| Удаление записи (определение удаляемой записи по ключевому полю) | 0.0043516469001770 02 | 0.0046465349197387690 | 0.0042664027214050 29 |
| Удаление группы записей | 0.0007600903511047364 | 0.0010634016990661620 | 0.0037117981910705566 |
| Сжатие базы данных (после удаления из БД 200 строк) | 0.0173685550689697 27 | 0.0196712017059326170 | 0.0159568786621093 75 |
| Сжатие базы данных (после удаления, в результате которого в БД остается 200 строк) | 0.0433843135833740 2 | 0.4324469566345215 | 4.5712852478027 34 |

## *Выводы*

1)Поиск по ключевому полю выполняется за логарифмическое время O(log n), так как дынные упорядоченные массив, и поэтому возможен бинарный поиск.

2)Поиск по не ключевому полю зависит от количества строк, и поэтому зависимость линейная O(n).

3)Поиск по маске, как и поиск по не ключевому полю, осуществляется за O(n).

4)Добавление записи, как и добавление группы записей, не зависит от размера таблицы и происходит за O(1).

5)Изменение записи по ключевому полю происходит за логарифмическое время O(log n).

6)Изменение записи по не ключевому полю происходит за O(n).

7)Удаление записи по ключевому полю происходит за O(log n).

8)Удаление группы записей (по ключевому полю) выполняется за O(n).

9)Сжатие базы данных зависит от количества строк в таблице, однако время выполнения ниже линейное, поэтому можно сказать, что операция сжатия выполняется за O(n).