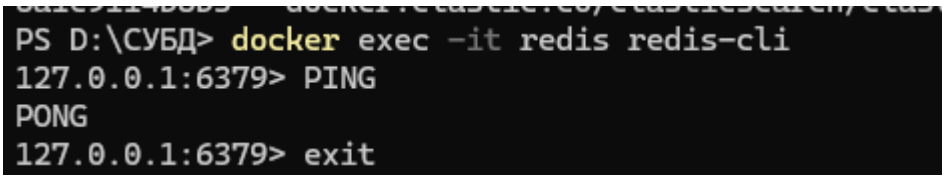
Задания на практическую работу:

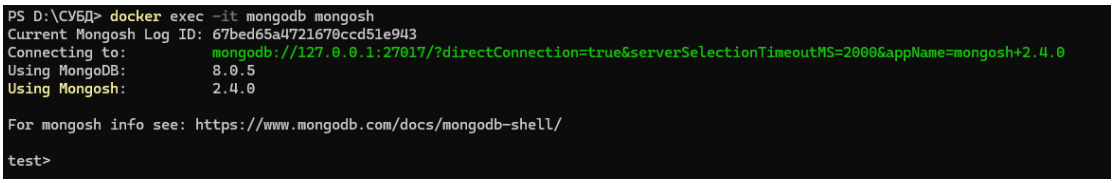
№1

Выполнить развертывание контейнеров или установить локально следующие службы, обеспечивающие различные виды представления хранимых данных:

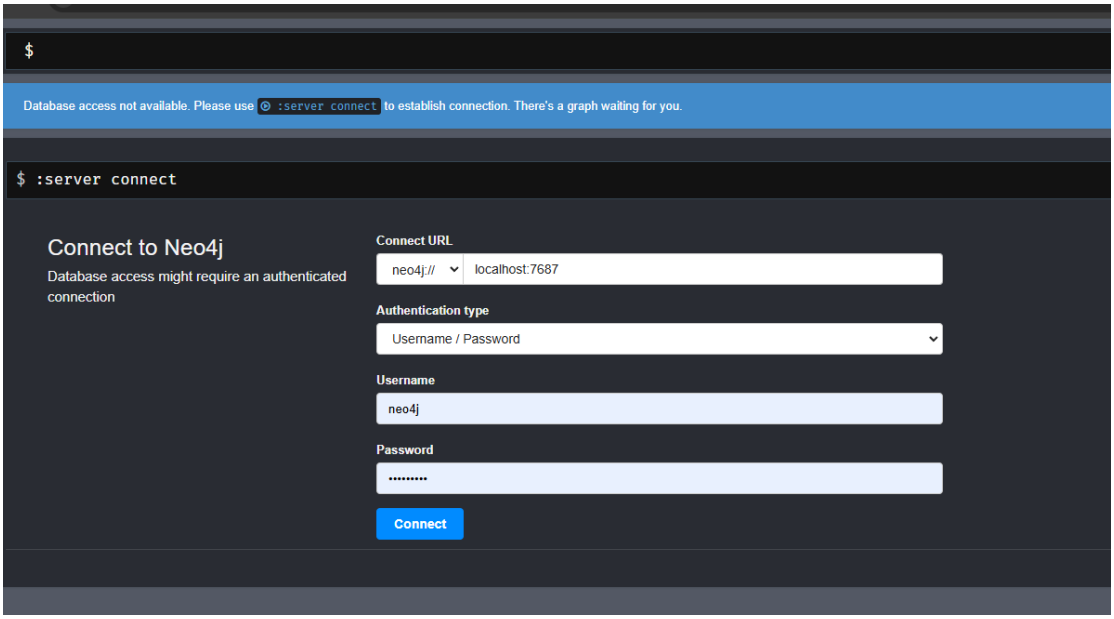
1. Redis



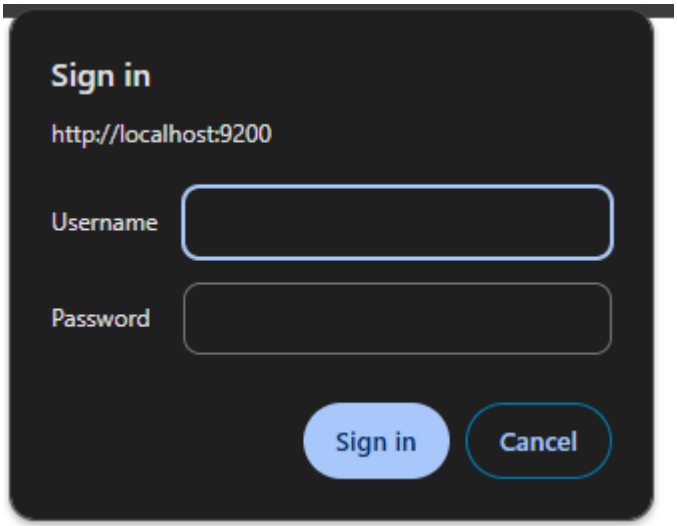
2. MongoDB



3. Neo4j



4. ElasticSearch



5. PostgreSQL

Выполнить для каждой службы проверку доступности и настройку места хранения данных и информации о состоянии (лог-записи).

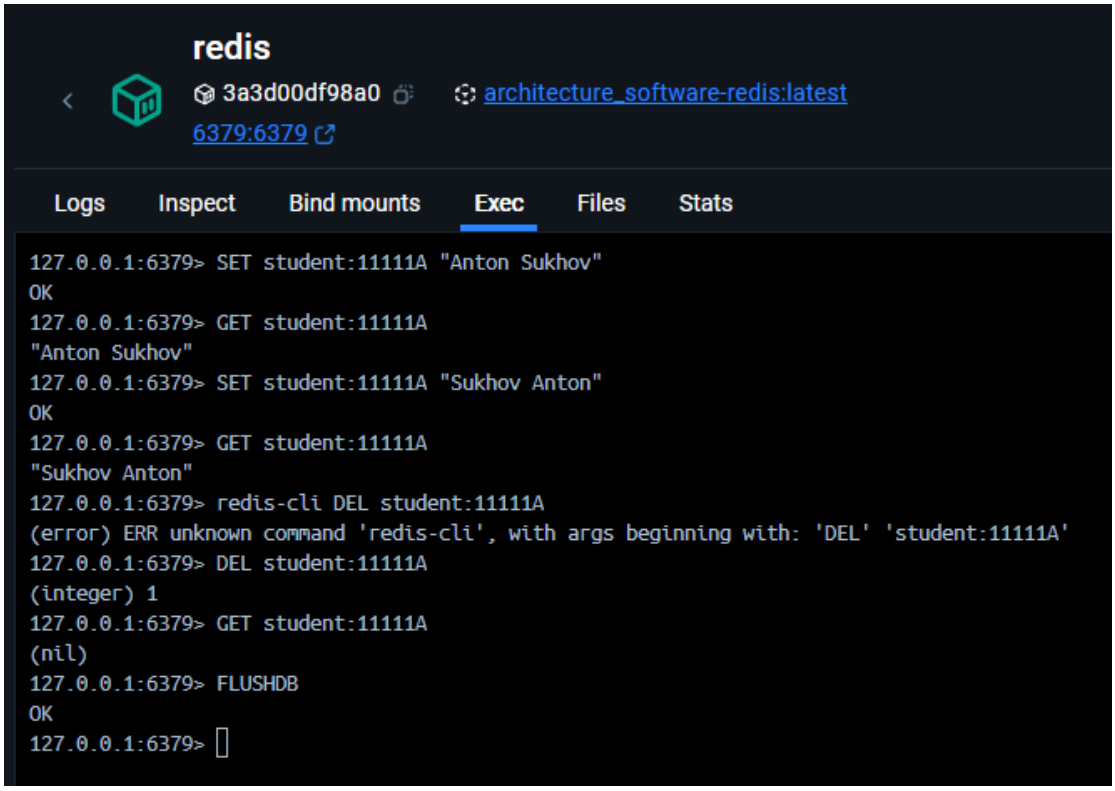
№2

Для каждого из типов хранения, развернутых в рамках практической работы №1, выполнить следующие операции:

- Создать хранилище данных
- Добавить данные в хранилище
- Прочитать данные из хранилища по ключу
- Изменить и сохранить данные в хранилище
- Удалить записи в хранилище
- Удалить хранилище

В качестве основных данных, вносимых в рамках работы, предполагается использовать список студентов группы и изучаемых курсов в рамках семестра.

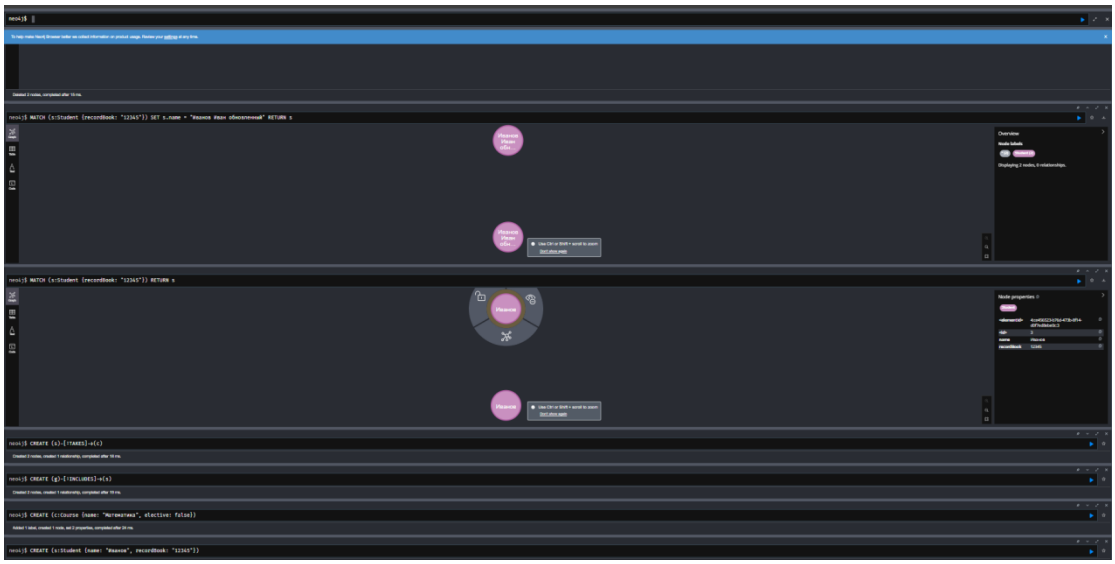
Для Redis – только список студентов с ключом в виде номера зачетной книжки

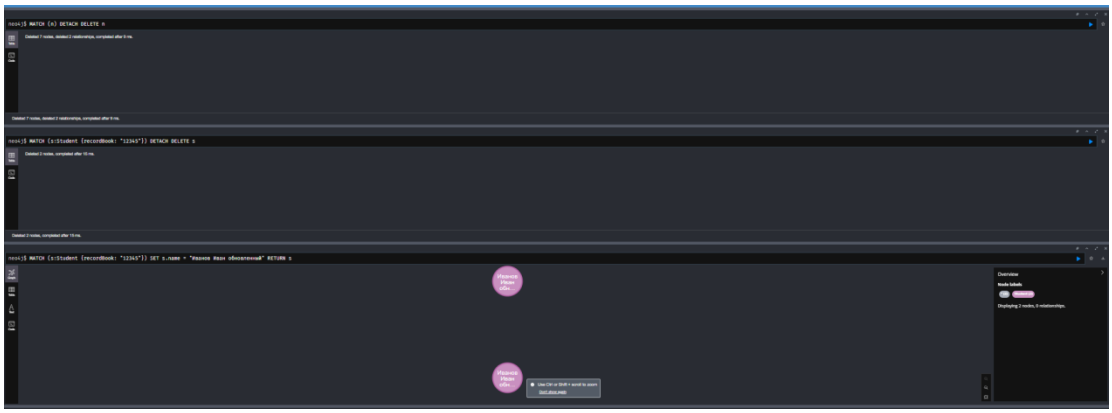


Для MongoDB – документ с данными и составом группы

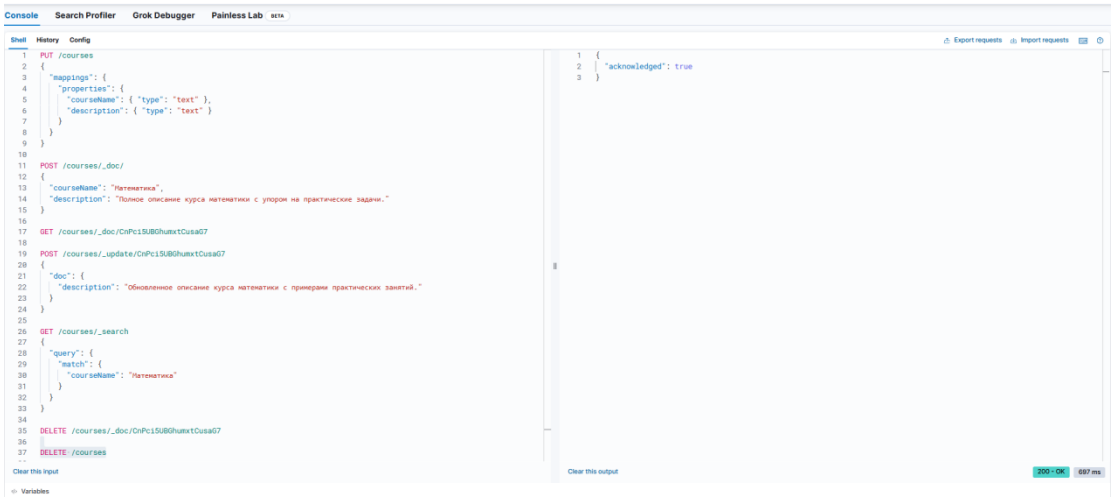
```
{ ok: 1 }
groupDB> db.group.insertOne({
...   groupName: "Группа А",
...   students: [
...     { name: "Иванов", recordBook: "12345" },
...     { name: "Петров", recordBook: "12346" }
...   ]
... })
...
{
  acknowledged: true,
  insertedId: ObjectId('67d1c4e86f1ac56b0a51e944')
}
groupDB> db.group.find({"students.recordBook":"12345"})
[
  {
    _id: ObjectId('67d1c4e86f1ac56b0a51e944'),
    groupName: 'Группа А',
    students: [
      { name: 'Иванов', recordBook: '12345' },
      { name: 'Петров', recordBook: '12346' }
    ]
  }
]
groupDB> db.group.updateOne(
...   { "students.recordBook": "12345" },
...   { $set: { "students.$.name": "Иванов Иван обновленный" } }
... )
...
{
  acknowledged: true,
  insertedId: null,
  matchedCount: 1,
  modifiedCount: 1,
  upsertedCount: 0
}
groupDB> db.group.updateOne(
...   { groupName: "Группа А" },
...   { $pull: { students: { recordBook: "12345" } } }
... )
...
{
  acknowledged: true,
  insertedId: null,
  matchedCount: 1,
  modifiedCount: 1,
  upsertedCount: 0
}
groupDB> db.group.drop()
true
groupDB>
```

Для Neo4J – Связи между группой-студентом-курсом, рассчитывая, на использование курсов по выбору.

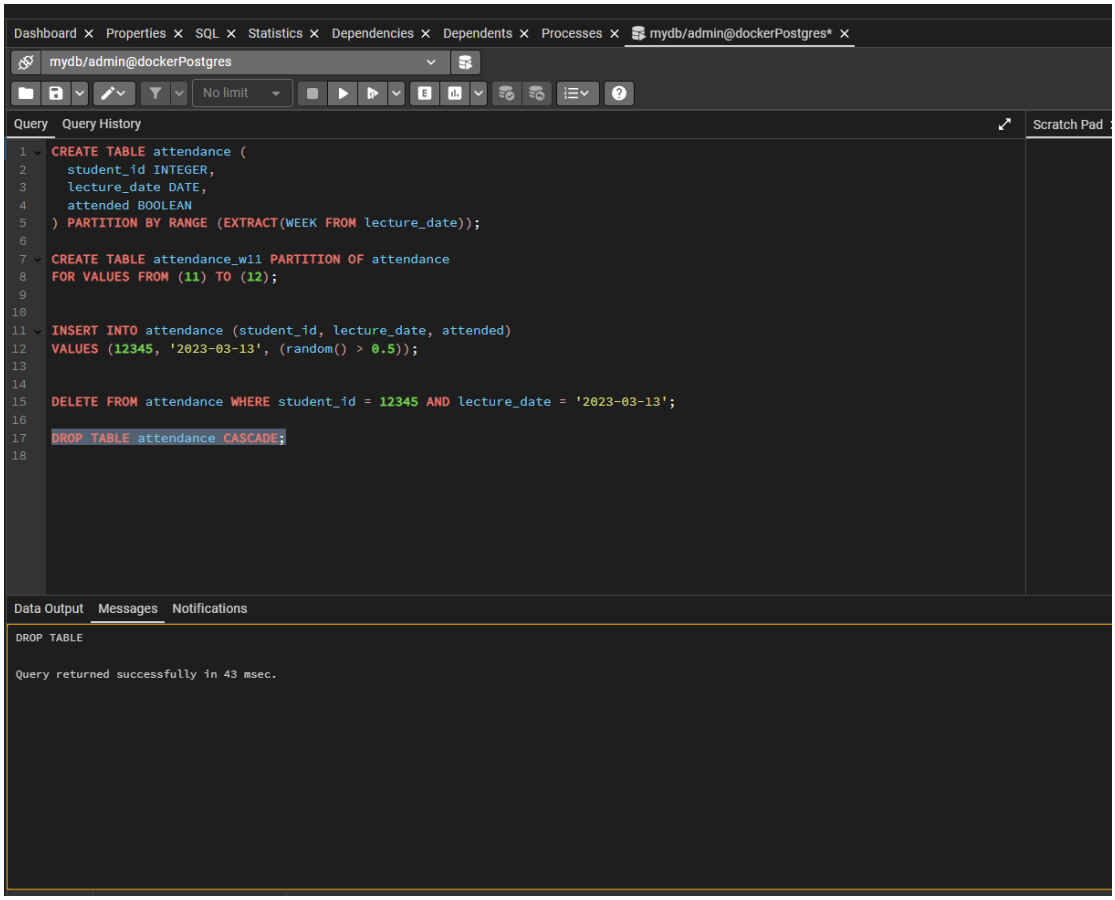




Для ElasticSearch – данные с полнотекстовым описанием курса.



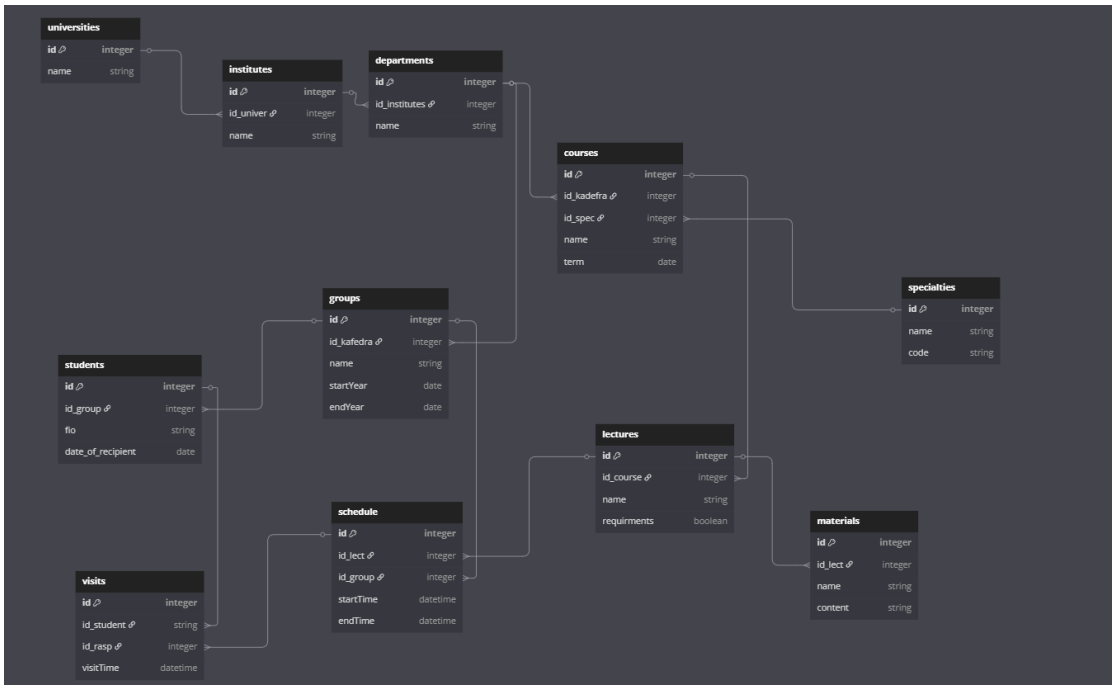
Для PosgreSQL – данные о посещении лекций студентами с партиционированием по неделе посещения. Данные о посещении при заполнении рекомендуется сформировать с помощью случайного выбора.



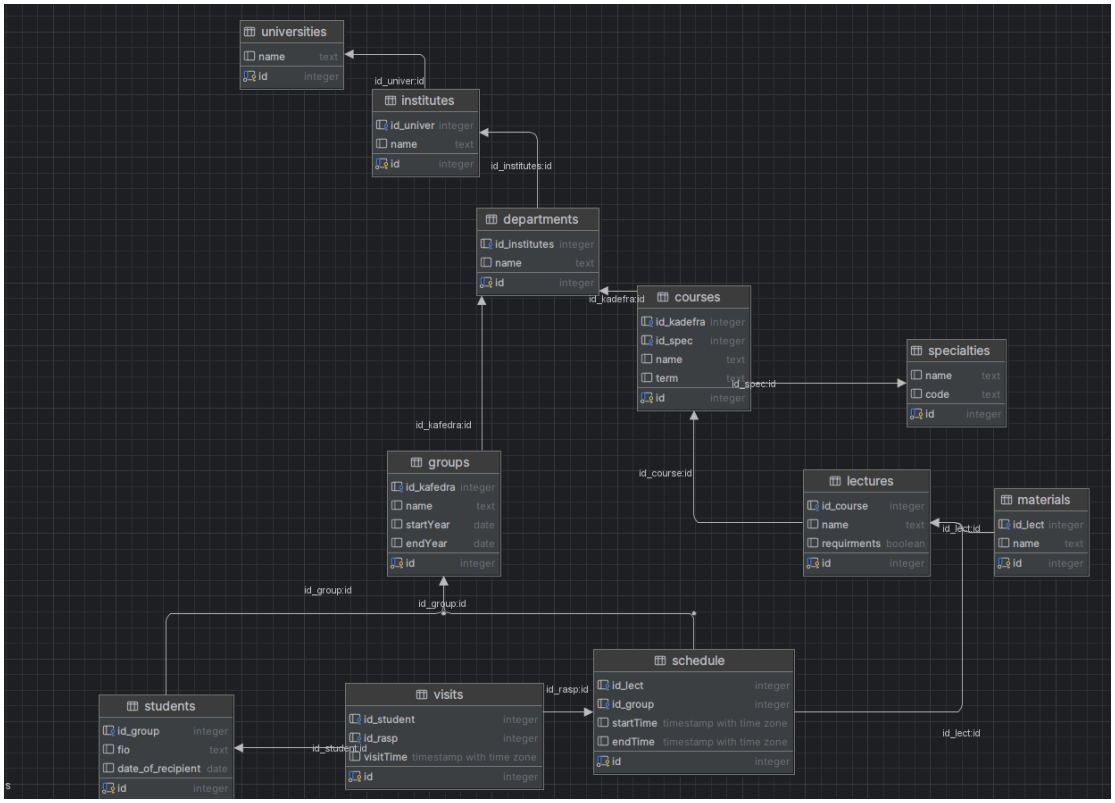
№3

На основе информации о группах учащихся, курсах обучения, лекционной программе и составу лекционных курсов и практических занятий, а также структуре связей между курсами, специальностями, студентами кафедры и данными о посещении студентами занятий, сформировать структуру хранения и связей в реляционной базе данных. Разместить информацию по различным видам представления хранения данных (структуры ключ-значение, объекты документ-композиция, наборы типизированных связей, полнотекстовая информация с метаданными, транзакционные данные с партицированием) для обеспечения оптимальной структуры для выборки информации в целях аналитических запросов. Обосновать свой выбор с точки зрения характеристик типов хранилищ.

Основная схема:



Postgres:



Redis:

	field	value
1	id_group	1
2	fio	Сухов Антон Алексеевич
3	date_of_recipient	2022-09-01

Mongo DB:


```
[
  {
    "_id": 1,
    "institutes": [
      {
        "_id": 1,
        "name": "Институт ИКБ",
        "departments": [
          {
            "_id": 1,
            "name": "Факультет разработки ПО"
          }
        ]
      }
    ],
    "name": "РТУ МИРЭА"
  }
]
```

ElasticSearch:

```
PUT /materials
{
  "mappings": {
    "properties": {
      "id": {
        "type": "integer"
      },
      "id_lect": {
        "type": "integer"
      },
      "name": {
        "type": "text",
        "analyzer": "russian"
      },
      "lecture_text": {
        "type": "text",
        "analyzer": "russian"
      }
    }
  }
}
```

```
POST /materials/_doc/1
{
  "id": 1,
  "id_lect": 1,
  "name": "Лекция 1: Основные концепции",
  "lecture_text": "Лекция 1: Введение в архитектуру программного обеспечения 1. Введение Архитектура программного
    обеспечения (ПО) играет ключевую роль в процессе разработки сложных программных систем. Она определяет
    структуру системы, взаимодействие между компонентами и основные принципы, обеспечивающие достижение требуемых
    функциональных и нефункциональных характеристик. 2. Определение архитектуры ПО Существует множество
    определений архитектуры ПО. Одно из наиболее распространенных: архитектура программного обеспечения — это
    совокупность решений об организации системы, включающая выбор структурных элементов, их интерфейсов и
    взаимодействий, а также принципы и руководства, управляющие их разработкой и эволюцией. Википедия —
    свободная энциклопедия +1 StudFiles +1 3. Значение архитектуры ПО Правильно спроектированная архитектура
    обеспечивает: Масштабируемость: возможность системы расти и адаптироваться к увеличению нагрузки.
    Читай-город +1 Википедия — свободная энциклопедия +1 Надежность: устойчивость к сбоям и способность
```

Neo4j: