Финальная домашняя работа

ФИО: Курепин Андрей Дмитриевич

Группа: МИНДА241

Факультет: Инженерия данных

Задача: спроектировать NoSQL БД для хранения оценок студентов.

GIT: hse_nosql_final

Требования к данным:

Студенты

Люди обучающие в университете на конкретном факультете и состоящие в учебной группе.

Факультет – направление в рамках которого студент обучается в институте.
Группа – объединение студентов по году поступления и факультету для быстрой коммуникации и упрощения процессов управления студентам со стороны руководителя Факультета.
Курсы – каждый студент имеет свой набор курсов, дисциплин, которые он обязан сдать для получения диплома. Курсы бывают с оценкой/без оценки (зачет), это важно учесть при проектировании.
Документы – список файлов, которые студент загрузил в свой профиль, имеют название, описание и ссылку на хранилище.
Добавочная информация – информация о студенте, со свободным заполнение: контакты, часы доступа, о себе и т.д.

Основные атрибуты: ФИО, Почта, Рабочий телефон, Логин, Пароль, Документы, Курсы, Добавочная информация, Название группы и Факультет.

Курсы

Дисциплины, которые преподаватель читает студентам в рамках семестра. Один курс могут вести разные преподаватели, в зависимости от направления и года обучения. Курсы назначаются студентов согласно регламентным процессам образовательного учреждения и могут меняться при изменении регламентов обучения.

Основные атрибуты: Название, Описание

Факультеты

Это направление, в рамках которого Студент обучается в институте. Сотрудники каждого направления выполняют работу по сопровождению, руководству и обучению студентов, поступивших на данный факультет.

Преподавательский состав – это сотрудники института, работающие на данном факультете. В рамках данного направления они осуществляют обучение студентов.
Руководитель – это главный преподаватель направления, который выполняет административную роль руководителя, данной орг. единицы.

Основные атрибуты: Название, Описание, Почта, Руководитель, Преподавательский состав, Дата основания и Регламентные документы.

Преподаватели

Это сотрудники института, которые осуществляют обучение студентов в рамках направления и дисциплины.

Курсы – список дисциплин, который преподаватель может читать студентам в рамках процесса обучения.
Добавочная информация – информация о сотруднике, со свободным заполнение: предпочтения в Кофе, контакты, часы доступа, о себе и т.д.

Основные атрибуты: ФИО, Почта, Рабочий телефон, Логин, Пароль, Документы, Курсы, Дата устройства на работу, Добавочная информация.

Потребности пользователей БД:

1. Студенты
    - Просмотр своего профиля
    - Просмотр выставленных оценки по текущим курсам
    - Поиск курсов из общего пула
    - Просмотр карточки преподавателя
2. Преподаватели
    - Поиск студентов с долгами
    - Получение информации о составе группы студентов для административной работы
    - Назначение нового курса студентам
    - Изменение информации о курсе
    - Выставление оценок студентам по курсам

- Изменение информации в профиле сотрудника

Особенности реализации

Коллекция «Студенты» умышленно денормализована, для быстрого получения информации по текущим оценкам Студента.
Атрибут «cources» хранит дублирующую информацию о ФИО Преподавателя и Названии курса для быстрого получения данных по оценкам без «лишних» операций объединения при загрузке профиля Студента.

Данная реализация имеет и недостатки, при обновлении ФИО преподавателя или Названия курса, необходимо обновить данные в курсах у каждого Студента, но такая операция крайне редкая.

Коллекция «Преподаватели» тоже деморализована, чтобы при загрузке профиля Преподавателя получать сразу всю необходимо информацию о курсах, которые ведет данный сотрудник.

К минусам данной реализации можно отнести необходимость обновлять данные по названию Курсов у каждого Преподавателя при изменении названия Курса, что является кране редкой операцией.

Схема сущностей:

Полный код JSON-схем для каждой коллекции представлены в git-репозитории в папке schemas. В документе приведены краткие(свернутые) схемы для упрощения визуализации и возможности вставить их в документ в читаемом виде.

Студенты

Кратная JSON-схема для коллекции «Студенты» представлена на Рисунок 1.

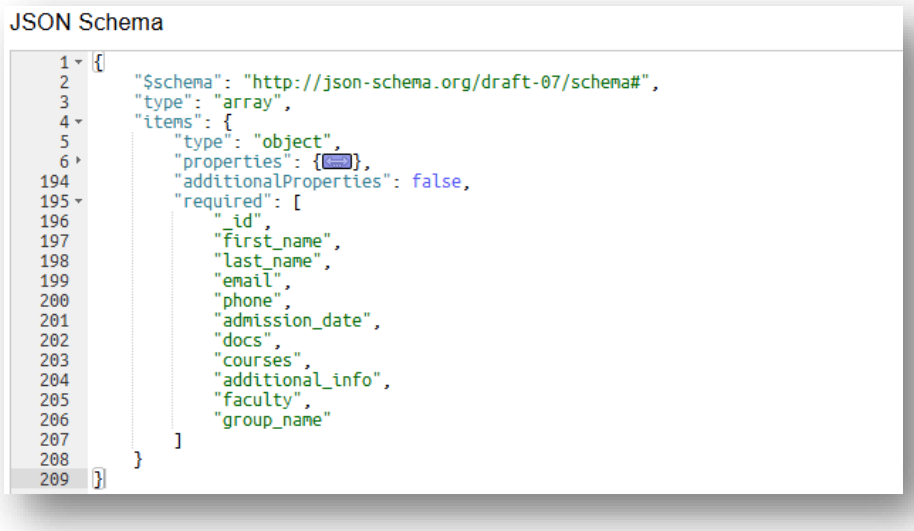


Рисунок 1, краткая схема коллекции «Студенты»

Курсы

Полная JSON-схема для коллекции «Курсы» представлена на Рисунок 2.

Рисунок 2, полная схема коллекции «Курсы»

Факультеты

Кратная JSON-схема для коллекции «Факультеты» представлена на Рисунок 3.

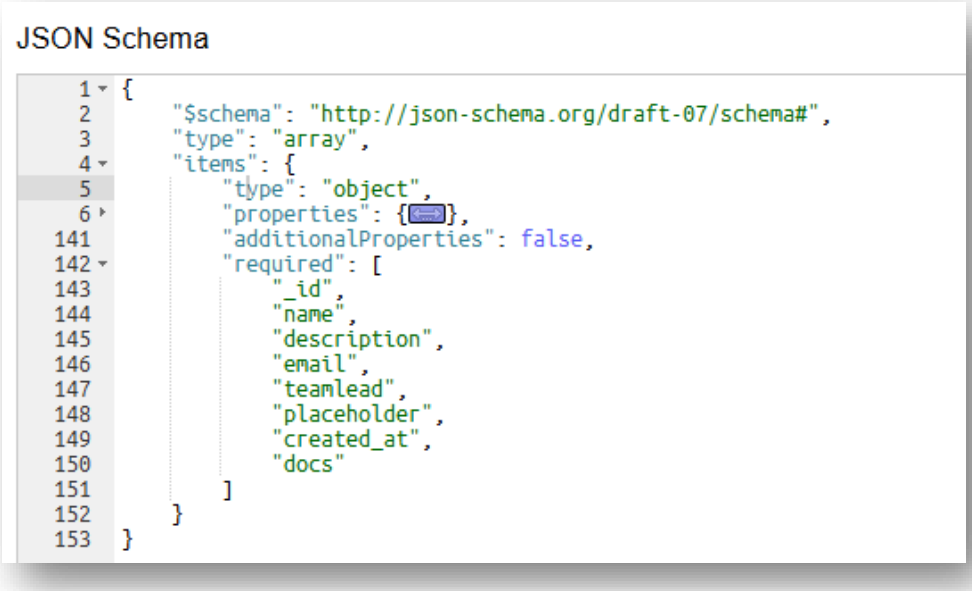


Рисунок 3, краткая схема коллекции «Факультеты»

Преподаватели

Кратная JSON-схема для коллекции «Преподаватели» представлена на Рисунок 4.

JSON Schema

```
{
    "$schema": "http://json-schema.org/draft-07/schema#",
    "type": "array",
    "items": {
        "type": "object",
        "properties": {},
        "additionalProperties": false,
        "required": [
            "_id",
            "first_name",
            "last_name",
            "email",
            "phone",
            "created_at",
            "docs",
            "additional_info",
            "courses"
        ]
    }
}
```

Рисунок 4, краткая схема коллекции «Преподаватели»

Индексы:

Для быстрого выполнения операций построим 5 индексов, Код для построения представлен на Рисунке 5.

indexes > JS all_index.js > ...

```
// Общий файл с индексами

// Индекс для поиск всех студентов группы
db.Students.createIndex({ group_name: 1 });

// Индекс для поиск всех студентов определенного факультета и группы
db.Students.createIndex({ "faculty._id": 1, "group_name": 1 });

// Индекс для поиск/сортировки студентов по дате поступления
db.Students.createIndex({ admission_date: 1 });

// Индекс для поиск всех Преподавателей, кто может вести данный курс
db.Teachers.createIndex({ "courses._id": 1 });

// Индекс для поиск курса по его названию
db.Courses.createIndex({ name: 1 });
```

Рисунок 5, код для построения индексов

Результат выполнения команд по построению индексов представлен на Рисунке 6.

```
university> db.Students.createIndex({ group_name: 1 });
group_name_1
university> db.Students.createIndex({ 'faculty._id': 1, 'group_name': 1 });
faculty._id_1_group_name_1
university> db.Students.createIndex({ admission_date: 1 });
admission_date_1
university> db.Teachers.createIndex({ 'courses._id': 1 });
courses._id_1
university> db.Courses.createIndex({ name: 1 });
name_1
university> db.Courses.createIndex({ name: 'text' });
name_text
university>
```

Рисунок 6, результат построения индексов

indexes > JS all_index.js > ...

```
1   // Общий файл с индексами
2
3   // Индекс для поиск всех студентов группы
4   db.Students.createIndex({ group_name: 1 });
5
6   // Индекс для поиск всех студентов определенного факультета и группы
7   db.Students.createIndex({ 'faculty._id': 1, 'group_name': 1 });
8
9   // Индекс для поиск/сортировки студентов по дате поступления
10  db.Students.createIndex({ admission_date: 1 });
11
12  // Индекс для поиск всех Преподавателей, кто может вести данный курс
13  db.Teachers.createIndex({ 'courses._id': 1 });
14
15  // Индекс для поиск и сортировки курсов по его названию
16  db.Courses.createIndex({ name: 1 });
17  db.Courses.createIndex({ name: 'text' });
```

Рисунок 7, код для построения индексов с описанием

Запросы:

Общий файл с запросами приведен в репозитории в папке ./requeries, all-queries.js

1. Поиск информации о Студенте по его идентификатору, Рисунок 8

```
university> db.Students.find({_id: ObjectId('67d5aaf7ba7587ca4ecc07fc')}, { docs: 0, courses: 0});
[
  {
    _id: ObjectId('67d5aaf7ba7587ca4ecc07fc'),
    first_name: 'Mark',
    last_name: 'Barnes',
    email: 'willisgregory@example.net',
    phone: '1929.911.4333x844',
    admission_date: ISODate('2022-06-05T22:46:36.119Z'),
    additional_info: {
      about_me: 'Note win course.',
      other_contacts: {
        icq: 'Ball up get camera sound.',
        telegram: 'Among language lay deep bit.',
        other: 'moydodyr.ru'
      }
    },
    faculty: {
      _id: ObjectId('67d5aa4fba7587ca4ecbfd3f'),
      name: 'Quality nice though fine fill.'
    },
    group_name: 'sic-62'
  }
]
```

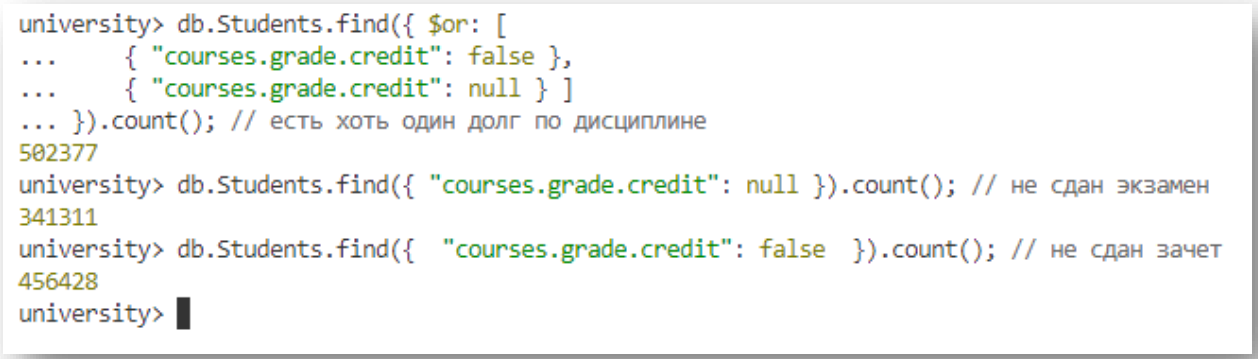
Рисунок 8, поиск Студента по идентификатору

2. Поиск Студентов, состоящих в одной группе, Рисунок 9

```
university> db.Students.find({ group_name: 'kms-13'}, { docs: 0, courses: 0, faculty: 0});
[
  {
    _id: ObjectId('67d5aaf8ba7587ca4ecd3182'),
    first_name: 'Kenneth',
    last_name: 'Bradley',
    email: 'raymond18@example.org',
    phone: '16889634932',
    admission_date: ISODate('2022-06-08T05:36:26.633Z'),
    additional_info: {
      other_contacts: {
        icq: 'Such material big box.',
        telegram: 'Call lead maybe dog.',
        other: 'moydodyr.ru'
      },
      free_in_time: '9-23'
    },
    group_name: 'kms-13'
  },
  {
    _id: ObjectId('67d5b465683e1d9f41e9c29f'),
    first_name: 'Stephanie',
    last_name: 'Valenzuela',
    email: 'floresrachel@example.com',
    phone: '1(741)314-8847',
    admission_date: ISODate('2022-09-30T13:09:39.875Z'),
    additional_info: { about_me: 'Go rather need there.', free_in_time: '6-21' },
    group_name: 'kms-13'
  }
]
```

Рисунок 9, поиск Студента в одной группе по ее названию

3. Поиск студентов с долгами (есть дисциплины без зачетов/оценок), Рисунок 10



```
university> db.Students.find({ $or: [
...     { "courses.grade.credit": false },
...     { "courses.grade.credit": null } ]
... }).count(); // есть хоть один долг по дисциплине
502377
university> db.Students.find({ "courses.grade.credit": null }).count(); // не сдан экзамен
341311
university> db.Students.find({  "courses.grade.credit": false  }).count(); // не сдан зачет
456428
university>
```

Рисунок 10, поиск Студента с долгами

4. Поиск преподавателя по идентификатору, Рисунок 11

```
university> db.Teachers.find({_id: ObjectId('67d5a5495fe1b373ef5aec8d')}, { docs: 0, courses: 0});
[
  {
    _id: ObjectId('67d5a5495fe1b373ef5aec8d'),
    first_name: 'Jeffrey',
    last_name: 'Saunders',
    email: 'kristenmontgomery@example.net',
    phone: '1001-936-289-',
    created_at: ISODate('1930-11-25T23:43:26.806Z'),
    additional_info: { about_me: 'Way despite rest form.', coffee: 'Американо' }
  }
]
```

Рисунок 11, поиск Преподавателя по идентификатору

5. Поиск всех Преподавателей, кто может преподавать определенный курс, Рисунок 12

```
university> db.Teachers.find({'courses._id': ObjectId('67d5a5495fe1b373ef5ae8a5')}, {first_name: 1, last_name: 1, phone: 1});
[
  {
    _id: ObjectId('67d5a5495fe1b373ef5aece5'),
    first_name: 'Travis',
    last_name: 'Jones',
    phone: '1314-396-9859x5'
  },
  {
    _id: ObjectId('67d5a5495fe1b373ef5aed35'),
    first_name: 'Lauren',
    last_name: 'Davis',
    phone: '1+1-999-916-9'
  },
  {
    _id: ObjectId('67d5a5495fe1b373ef5aed3d'),
    first_name: 'Raymond',
    last_name: 'Benitez',
    phone: '1473.288.9777x6'
  }
]
```

Рисунок 12, найденные Сотрудники могут вести данную дисциплину

6. Добавление нового Преподавателя Рисунок 13

```
university> db.Teachers.insertOne({
...     first_name: 'Дмитрий',
...     last_name: 'Калугин-Балашов',
...     email: 'test-email@example.com',
...     phone: '1234567890',
...     created_at: new Date(),
...     additional_info: {
...       about_me: 'Преподаватель дисциплины Нереляционные базы данных',
...       other_contacts: {
...         telegram: '@test'
...       }
...     }
...   });
{
  acknowledged: true,
  insertedId: ObjectId('67d5c8b9c6545d28046b140c')
}
university> db.Teachers.find({'_id': ObjectId('67d5c8b9c6545d28046b140c')});
[
  {
    _id: ObjectId('67d5c8b9c6545d28046b140c'),
    first_name: 'Дмитрий',
    last_name: 'Калугин-Балашов',
    email: 'test-email@example.com',
    phone: '1234567890',
    created_at: ISODate('2025-03-15T18:36:41.423Z'),
    additional_info: {
      about_me: 'Преподаватель дисциплины Нереляционные базы данных',
      other_contacts: { telegram: '@test' }
    }
  }
]
```

Рисунок 13, добавляем и выводим нового Преподавателя

7. Добавляем Преподавателю курс, который он преподаёт, Рисунок 14

```
university> db.Teachers.updateOne(
...     { _id: ObjectId('67d5c8b9c6545d28046b140c') },
...     { $push: { courses: { _id: ObjectId('67d5a5495fe1b373ef5ae8a7'), name: 'NoSQL' } } }
...   );
{
  acknowledged: true,
  insertedId: null,
  matchedCount: 1,
  modifiedCount: 1,
  upsertedCount: 0
}
university> db.Teachers.find({'_id': ObjectId('67d5c8b9c6545d28046b140c')});
[
  {
    _id: ObjectId('67d5c8b9c6545d28046b140c'),
    first_name: 'Дмитрий',
    last_name: 'Калугин-Балашов',
    email: 'test-email@example.com',
    phone: '1234567890',
    created_at: ISODate('2025-03-15T18:36:41.423Z'),
    additional_info: {
      about_me: 'Преподаватель дисциплины Нереляционные базы данных',
      other_contacts: { telegram: '@test' }
    },
    courses: [ { _id: ObjectId('67d5a5495fe1b373ef5ae8a7'), name: 'NoSQL' } ]
  }
]
university>
```

Рисунок 14, добавляем дисциплину Преподавателю

8. Поиск Курса по ключевому слову, Рисунок 15

```
university> db.Courses.find({ $text: { $search: 'pattern' } }, {_id: 0, description: 0});
[
  { name: 'Similar pattern or there.' },
  { name: 'As pattern east form.' },
  { name: 'Take pattern describe.' },
  { name: 'Pattern name third.' },
  { name: 'Meeting situation myself over pattern.' },
  { name: 'Pattern none hard to.' },
  { name: 'Pattern send house.' },
  { name: 'Wrong manage particularly pattern.' },
  { name: 'Ten pattern property population do.' },
  { name: 'Pattern citizen me pick suggest.' },
  { name: 'Pattern part themselves message newspaper.' },
  { name: 'Everyone pattern knowledge bag.' },
  { name: 'Test pattern writer design.' },
  { name: 'Pattern action mouth feel same.' },
  { name: 'No red pattern audience tax.' },
  { name: 'Blue pattern letter size.' },
  { name: 'Doctor pattern girl them age.' },
  { name: 'Thank college pattern law.' },
  { name: 'Throughout go pattern involve.' },
  { name: 'Site water pattern instead.' }
]
```

Рисунок 15, найдем все курсы со словом «pattern»

9. Изменение информации о курсах Студента, Рисунок 16-17

```
university> db.Students.find({_id: ObjectId('67d5a54b5fe1b373ef5aee37')}, { first_name: 1, last_name: 1, courses: 1});
[
  {
    _id: ObjectId('67d5a54b5fe1b373ef5aee37'),
    first_name: 'Shelly',
    last_name: 'Gonzales',
    courses: [
      {
        course_id: ObjectId('67d5a5495fe1b373ef5aebb9'),
        course_name: 'Sort list trip address bad be.',
        year: 2020,
        semester: 2,
        grade: { credit: true },
        maintainer: ObjectId('67d5a5495fe1b373ef5aeca5'),
        maintainer_fio: 'Cole  Johnson'
      },
      {
        course_id: ObjectId('67d5a5495fe1b373ef5aea56'),
        course_name: 'None guess finish need.',
        year: 2021,
        semester: 2,
        grade: { credit: false },
        maintainer: ObjectId('67d5a5495fe1b373ef5aeca8'),
        maintainer_fio: 'Ryan  Warner'
      }
    ]
  }
]
```

Рисунок 16, выведем информацию о Курсах студента

Рисунок 17, удалим у Студента курс и выведем повторно

10. Найдем статистку поступления Студентов по годам

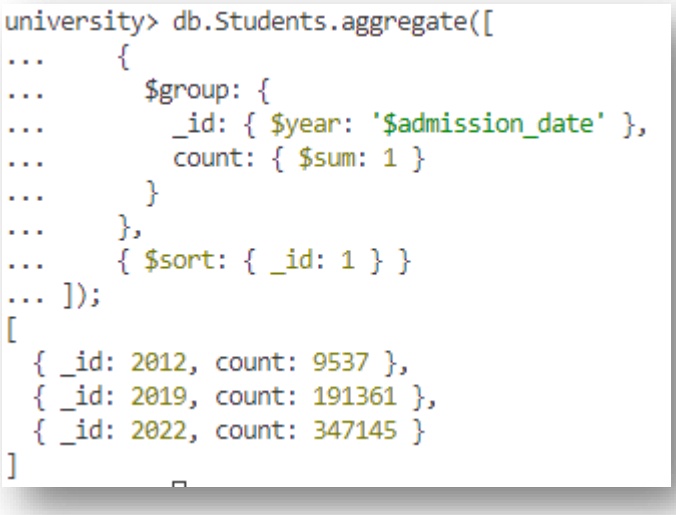


Рисунок 18, найдем статистику поступления по годам

Общий файл с запросами и комментариями приведен на Рисунках 19-21.

queries > JS all-queries.js > $group > count

```
// 1. Поиск информации о Студенте по его идентификатору
db.Students.find({_id: ObjectId('67d5aaf7ba7587ca4ecc07fc')}, { docs: 0, courses: 0});

//2.  Поиск студентов, состоящих в одной группе
db.Students.find({ group_name: 'kms-13'}, { docs: 0, courses: 0, faculty: 0});

// 3. Поиск студентов с долгами (есть дисциплины без зачетов/оценок)
db.Students.find({
    $or: [
      { 'courses.grade.credit': false }, // нет экзамена
      { 'courses.grade.credit': null } // нет зачета
    ]
});

// Запросы количества
db.Students.find({ $or: [
    { 'courses.grade.credit': false },
    { 'courses.grade.credit': null } ]
}).count(); // есть хоть один долг по дисциплине
db.Students.find({ 'courses.grade.credit': null }).count(); // не сдан экзамен
db.Students.find({ 'courses.grade.credit': false  }).count(); // не сдан зачет

// 4. Поиск преподавателя по идентификатору
db.Teachers.find({_id: ObjectId('67d5a5495fe1b373ef5aec90')}, { docs: 0});

// 5. Поиск всех Преподавателей, кто может преподавать определенный курс
db.Teachers.find(
  {'courses._id': ObjectId('67d5a5495fe1b373ef5ae8a5')},
  {first_name: 1, last_name: 1, phone: 1}
);
```

Рисунок 19, запросы с 1 по 5

```
// 5. Поиск всех Преподавателей, кто может преподавать определенный курс
db.Teachers.find(
  {'courses._id': ObjectId('67d5a5495fe1b373ef5ae8a5')},
  {first_name: 1, last_name: 1, phone: 1}
);

// 6. Добавление нового Преподавателя

db.Teachers.insertOne({
    first_name: 'Дмитрий',
    last_name: 'Калугин-Балашов',
    email: 'test-email@example.com',
    phone: '1234567890',
    created_at: new Date(),
    additional_info: {
      about_me: 'Преподаватель дисциплины Нереляционные базы данных',
      other_contacts: {
        telegram: '@test'
      }
    }
  });

db.Teachers.find({'_id': ObjectId('67d5c8b9c6545d28046b140c')});

// 7. Добавление преподавателю нового курс
db.Teachers.updateOne(
    { _id: ObjectId('67d5c8b9c6545d28046b140c') },
    { $push: { courses: { _id: ObjectId('67d5a5495fe1b373ef5ae8a7'), name: 'NoSQL' }}}
  );

// 8. Поиск курса по ключевому слову
db.Courses.find({ $text: { $search: 'pattern' } }, {_id: 0, description: 0});
```

Рисунок 20, запросы с 5 по 8

```
// 9.   Удалим информацию о курсе у студента, предположим он перевелся

db.Students.find(
  {_id: ObjectId('67d5a54b5fe1b373ef5aee37')},
  { first_name: 1, last_name: 1, courses: 1}
);

db.Students.updateOne(
    { _id: ObjectId('67d5a54b5fe1b373ef5aee37') },
    { $pull: { courses: { course_id: ObjectId('67d5a5495fe1b373ef5aea56') }}}
);

// 10.   Найдем статистку по принятию студентов по годам

db.Students.aggregate([
    {
      $group: {
        _id: { $year: '$admission_date' },
        count: { $sum: 1 }
      }
    },
    { $sort: { _id: 1 } }
]);
```

Рисунок 21, запросы с 9 по 10