**МИНОБРНАУКИ РОССИИ**
**САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ**
**ЭЛЕКТРОТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ**
**«ЛЭТИ» ИМ. В.И. УЛЬЯНОВА (ЛЕНИНА)**
**Кафедра МО ЭВМ**

# ИНДИВИДУАЛЬНОЕ ДОМАШНЕЕ ЗАДАНИЕ

**по дисциплине «Введение в нереляционные системы управления базами данных»**

**Тема: Анализ случаев COVID**

Студент гр. 8303 _____ Гришин К. И.

Студент гр. 8303 _____ Крыжановский К. Е.

Студентка гр. 8303 _____ Самойлова А. С.

Преподаватель _____ Заславский М.М.

Санкт-Петербург
2021

**ЗАДАНИЕ**

Студенты: Гришин К. И., Крыжановский К. Е., Самойлова А. С.

Группа 8303

Тема работы: Анализ случаев COVID

Исходные данные:

Веб-приложение для анализа случаев COVID. Предоставляет просмотр статистики заболеваний и вакцинирования населения как планеты в целом, так и каждой страны в частности.

Содержание пояснительной записки:

«Введение»

«Качественные требования к решению»

«Сценарии использования»

«Модель данных»

«Разработанное приложение»

«Вывод»

Предполагаемый объем пояснительной записки:

Не менее 20 страниц

| | | |
|---|---|---|
| Студент гр. 8303 | _____ | Гришин К. И. |
| Студент гр. 8303 | _____ | Крыжановский К. Е. |
| Студентка гр. 8303 | _____ | Самойлова А. С. |
| Преподаватель | _____ | Заславский М. М. |

**СОДЕРЖАНИЕ**

# ВВЕДЕНИЕ

Цель данной работы – реализация приложения, которое позволит наблюдать статистику по Covid-19 как в табличном варианте, с использованием различных фильтров, так и в виде графиков, демонстрирующих динамику распространения болезни.

# 1. КАЧЕСТВЕННЫЕ ТРЕБОВАНИЯ К РЕШЕНИЮ

Требуется разработать приложение с использованием MongoDB, позволяющее удобно просматривать статистику заболеваний и вакцинации за определенный промежуток времени в указанной стране.

# 2. СЦЕНАРИИ ИСПОЛЬЗОВАНИЯ

## 1.1 Просмотр базы данных

Предусловие:

Пользователь находится на главной странице

Основной сценарий:

1. Пользователь нажимает кнопку «База Данных»
2. Пользователь выбирает «Весь период пандемии» или задаёт определенный диапазон
3. Пользователь выбирает «Все страны» или задает конкретную
4. Пользователь выбирает столбец, по которому необходимо проводить сортировку

Результат:

Пользователь получает таблицу с данными

## 1.2 Загрузка данных

Предусловие:

Пользователь находится на странице с отображением базы данных

Основной сценарий:

1. Пользователь нажимает кнопку «Загрузить базу данных»
2. В открывшемся модальном окне пользователь выбирает файл для импорта базы данных
3. Пользователь нажимает кнопку «Открыть»

Альтернативный сценарий:

Пользователь нажал кнопку «Отмена» или загрузил не валидный файл.

Результат:

В базу данных экспортированы данные из *.json* файла.

### 1.3 Выгрузка данных

Предусловие:

Пользователь находится на странице с отображением базы данных

Основной сценарий:

1. Пользователь нажимает кнопку «Скачать базу данных»
2. После обработки запроса сервером начнется скачивание файла в директорию загрузок браузера.

Результат:

Пользователь имеет *.json* файл с экспортированной базой данных

### 1.4 Просмотр диаграмм со статистикой новых случаев заболеваний

Предусловие:

Пользователь находится на главной странице

Основной сценарий:

1. Пользователь выбирает «Заболеваемость» (выбрана по умолчанию)
2. Пользователь выбирает страну
3. Пользователь выбирает диапазон дат

Альтернативный сценарий:

Параметры по умолчанию:

страна: отображаются данные о всем мире

диапазон дат: отображаются данные за весь период пандемии

Результат:

При изменении параметров, пользователю перестраивается диаграмма общего количества новых заболевших в сутки.

### 1.5 Просмотр максимального/среднего/минимального/общего количества новых случаев заболеваний

Предусловие:

Пользователь находится на главной странице

Основной сценарий:

1. Пользователь выбирает «Заболеваемость» (выбрана по умолчанию)
2. Пользователь выбирает страну
3. Пользователь выбирает диапазон дат
4. Пользователь выбирает функцию агрегации (Общее, Среднее, Минимальное, Максимальное)

Альтернативный сценарий:

Параметры по умолчанию:

страна: отображаются данные о всем мире

диапазон дат: отображаются данные за весь период пандемии

Результат:

Модальное окно с агрегированным значением. При выборе минимального или максимального также показывается дата, в которое наблюдалось такое значение.

### 1.6 Просмотр диаграмм со статистикой новых вакцинировавшихся

Предусловие:

Пользователь находится на главной странице

Основной сценарий:

1. Пользователь выбирает «Вакцинации»
2. Пользователь выбирает страну
3. Пользователь выбирает диапазон дат

Альтернативный сценарий:

Параметры по умолчанию:

страна: отображаются данные о всем мире

диапазон дат: отображаются данные за весь период пандемии

Результат:

При изменении параметров, пользователю перестраивается диаграмма общего количества новых заболевших в сутки.

### 1.7 Просмотр максимального/среднего/минимального/общего количества вакцинировавшихся

Предусловие:

Пользователь находится на главной странице

Основной сценарий:

1. Пользователь выбирает «Вакцинации»
2. Пользователь выбирает страну
3. Пользователь выбирает диапазон дат
4. Пользователь выбирает функцию агрегации (Общее, Среднее, Минимальное, Максимальное)

Альтернативный сценарий:

Параметры по умолчанию:

страна: отображаются данные о всем мире

диапазон дат: отображаются данные за весь период пандемии

Результат:

Модальное окно с агрегированным значением. При выборе минимального или максимального также показывается дата, в которое наблюдалось такое значение.

### 1.8 Попарное сравнение заболеваемости в странах

Предусловие:

Пользователь находится на главной странице

Основной сценарий:

1. Пользователь нажимает кнопку «Сравнение стран»
2. Пользователь выбирает первую страну
3. Пользователь выбирает вторую страну
4. Пользователь выбирает диапазон дат

Альтернативный сценарий:

Параметры по умолчанию:

диапазон дат: отображаются данные за весь период пандемии

Если не выбраны обе страны, диаграмма не строится.

Результат:

При изменении параметров, пользователю перестраивается диаграмма попарного сравнения заболеваемости в странах.

### 1.9 График зависимости заболевших в стране от плотности населения

Предусловие:

Пользователь находится на главной странице

Основной сценарий:

1. Пользователь нажимает кнопку «Заболевания и плотность населения»
2. Пользователь выбирает диапазон дат.

Альтернативный сценарий:

Параметры по умолчанию:

диапазон дат: отображаются данные за весь период пандемии

Результат:

При изменении параметров пользователю перестраивается диаграмма зависимости заболеваемости от плотности населения.

## 1.10 Макет UI по которому разрабатывалось приложение

Приложение разрабатывалось согласно макету на рисунке 1. Итоговое приложение имеет иную цветовую схему, переходы на различные экраны реализованы в отдельном меню. Выбор функции агрегации свернут в одну кнопку на графике. Фильтры для таблиц устанавливаются в соответствующем меню.

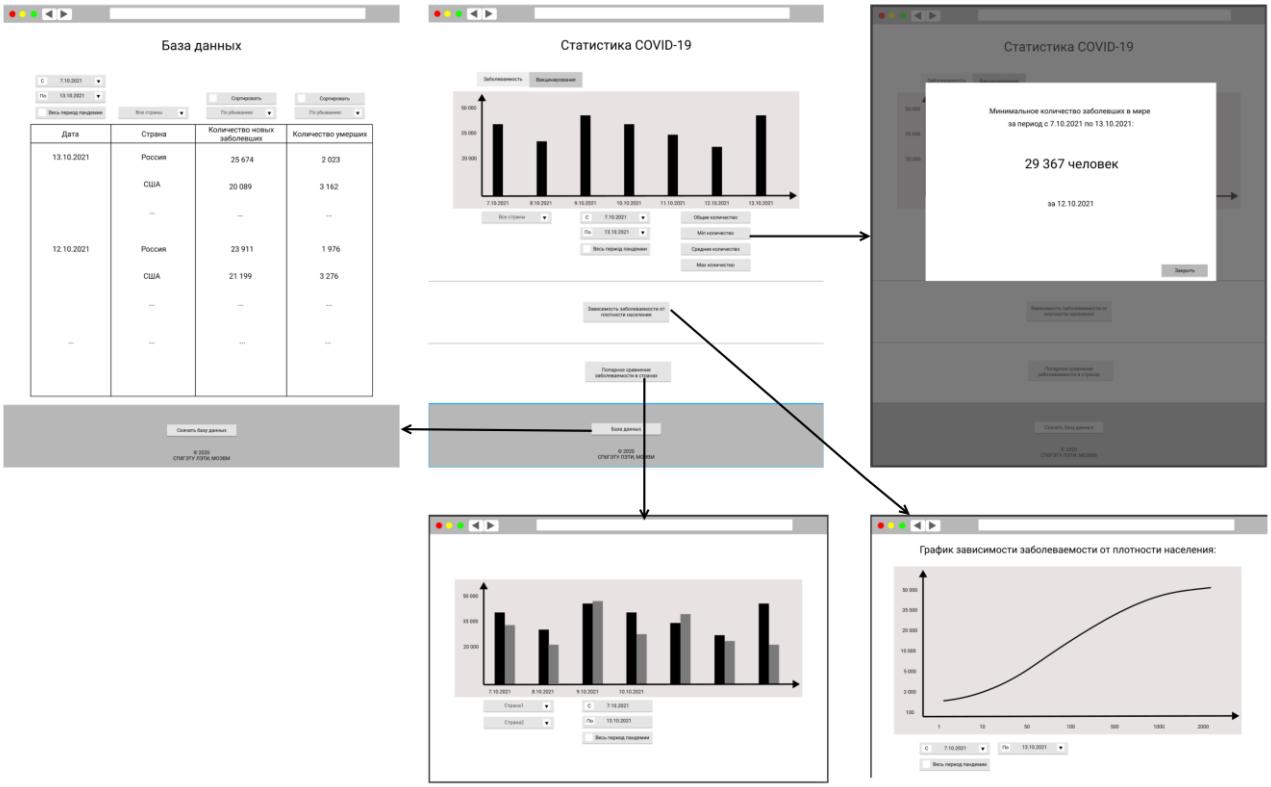


Рисунок 1. Изначальный макет приложения.

# 3. МОДЕЛЬ ДАННЫХ

## 3.1 Нереляционная модель данных (MongoDB)

База данных содержит три коллекции:

- Countries:

```
{
  _id: ObjectId,
  iso_code: String,
  continent: String,
  location: String,
  population: Integer,
  population_density: Double,
  median_age: Double,
  aged_65_older: Double,
  aged_70_older: Double
}
```

- Cases

```
{
  _id: ObjectId,
  date: DateTime,
  iso_code: String,
  total_cases: Integer,
  new_cases: Integer,
  new_cases_smoothed: Double,
  total_cases_per_million: Double,
  new_cases_per_million: Double,
  new_cases_smoothed_per_million: Double
}
```

- Vaccinations

```
{
  _id: ObjectId,
  date: DateTime,
  iso_code: String,
  people_vaccinated: Integer,
  people_fully_vaccinated: Integer,
  new_vaccinations: Integer,
  new_vaccinations_smoothed: Double,
  total_vaccinations_per_hundred: Double,
  people_vaccinated_per_hundred: Double,
  people_fully_vaccinated_per_hundred: Double,
  new_vaccinations_smoothed_per_million: Double
}
```

### 3.1.1 Оценка объема одного документа

- Countries
  - _id: ObjectId, 8 байт
  - iso_code: String, пусть максимальная длина iso_code равна 8, то 8 байт
  - continent: String, пусть максимальная длина iso_code равна 30, то 30 байт
  - location: String, пусть максимальная длина iso_code равна 30, то 30 байт
  - population: Integer, 8 байт
  - population_density: Double, 8 байт
  - median_age: Double, 8 байт
  - aged_65_older: Double, 8 байт
  - aged_70_older: Double, 8 байт

Итого: для одного документа коллекции *Countries* нужно:

$$V_{Co} = 8 + 8 + 30 + 30 + 8 + 8 + 8 + 8 + 8 = 116 \text{ байт}$$

- Cases
  - _id: ObjectId, 8 байт
  - date: DateTime, 8 байт
  - iso_code: String, пусть максимальная длина iso_code равна 8, то 8 байт
  - new_cases: Integer, 8 байт
  - total_cases: Integer, 8 байт
  - new_cases_smoothed: Double, 8 байт
  - total_cases_per_million: Double, 8 байт
  - new_cases_per_million: Double, 8 байт
  - new_cases_smoothed_per_million: Double, 8 байт

Итого: для одного документа коллекции *Cases* нужно:

$$V_{ca} = 8 + 8 + 8 + 8 + 8 + 8 + 8 + 8 + 8 = 72 \text{ байта}$$

- Vaccinations
  - _id: ObjectId, 8 байт
  - date: DateTime, 8 байт
  - iso_code: String, пусть максимальная длина iso_code равна 8, то 8 байт
  - people_vaccinated: Integer, 8 байт
  - people_fully_vaccinated: Integer, 8 байт

- new_vaccinations: Integer, 8 байт
- new_vaccinations_smoothed: Double, 8 байт
- total_vaccinations_per_hundred: Double, 8 байт
- people_vaccinated_per_hundred: Double, 8 байт
- people_fully_vaccinated_per_hundred: Double, 8 байт
- new_vaccinations_smoothed_per_million: Double, 8 байт

Итого: для одного документа коллекции *Vaccinations* нужно:

$$V_v = 8 + 8 + 8 + 8 + 8 + 8 + 8 + 8 + 8 + 8 + 8 = 88$$

### 3.1.2 Расчет объема данных MongoDB

Объем данных для хранения $N_{co}$ стран, $N_{ca}$ заболеваний и $N_v$ вакцинаций:

$$\mathrm{V}(N_{co}, N_{ca}, N_v) = N_{co} \cdot V_{co} + N_{ca} \cdot V_{ca} + N_v \cdot V_v$$

Каждый день добавляется по одному документу в коллекциях *Cases* и *Vaccinations*. В предположении, на одну страну нужно ~1000 записей (на 3 года), то:

$$N_{ca} = N_v = 1000 \cdot N_{co}$$

$$V(N_{co}) = N_{co} \cdot V_{co} + 1000 \cdot N_{co} \cdot V_{ca} + 1000 \cdot N_{co} \cdot V_v =$$

$$= N_{co} \cdot (116 + 1000 \cdot 72 + 1000 \cdot 88) = 160116 \cdot N_{co}$$

Пусть $\mathrm{N}_{co} = 20$, тогда:

$$V(20) = 20 \cdot 160116 = 3202320 \text{ Б} = 3 \text{ МБ}$$

### 3.1.3 Расчет чистого объема данных

Объем данных, необходимый для документа в *Countries* не будет изменен.

$$V_{co}[pure] = 116 \text{ байт}$$

Объем данных, необходимый для документа в *Cases* не должен учитывать объем поля *iso_code*. $\mathrm{V}_{ca}[pure] = 64$ байт

Объем данных, необходимый для документа в Vaccinations не должен учитывать объем поля iso_code. $V_v[pure] = 80$ байт

$$\frac{V_{pure}}{V} = \frac{V_{co}[pure] + V_{ca}[pure] + V_v[pure]}{V_{co} + V_{ca} + V_v} = \frac{116 + 64 + 80}{116 + 72 + 88} = 0.94$$

### 3.1.4 Запросы

- максимальное/минимальное/среднее/суммарное количество новых заболевших в сутки за весь/заданный период пандемии

  Используется одна коллекция

```
db.cases.aggregate([{
    $match: {
        iso_code: {
            $in: ['RU']
        } or None,
        date: {
            $gte: ISODate('2021-10-08'),
            $lte: ISODate('2021-10-09')
        }
    }
}, {
    $group: {
        _id: "$iso_code",
        max_disease_new_cases: {$max: "$new_cases"},
        min_disease_new_cases: {$min: "$new_cases"},
        avg_disease_new_cases: {$avg: "$new_cases"},
        sum_disease_new_cases: {$sum: "$new_cases"},
    }
}, {
    $project: {
        iso_code: '$_id',
        max_disease_new_cases: '$max_disease_new_cases',
        min_disease_new_cases: '$min_disease_new_cases',
        avg_disease_new_cases: '$avg_disease_new_cases',
        sum_disease_new_cases: '$sum_disease_new_cases',
        _id: 0
    }
}])
```

- максимальное/минимальное/среднее/суммарное количество новых вакцинированных в сутки за весь/заданный период пандемии

  Используется одна коллекция

```
db.vaccinations.aggregate([{
    $match: {
        iso_code: {
            $in: ['RU']
        } or None,
        date: {
            $gte: ISODate('2021-10-08'),
            $lte: ISODate('2021-10-09')
        }
    }
}, {
    $group: {
        _id: "$iso_code",
        max_new_vaccinations: {$max: "$new_vaccinations"},
        min_new_vaccinations: {$min: "$new_vaccinations"},
        avg_new_vaccinations: {$avg: "$new_vaccinations"},
        sum_new_vaccinations: {$sum: "$new_vaccinations"},
    }
}, {
    $project: {
```

```
        iso_code: '$_id',
        max_new_vaccinations: '$max_new_vaccinations',
        min_new_vaccinations: '$min_new_vaccinations',
        avg_new_vaccinations: '$avg_new_vaccinations',
        sum_new_vaccinations: '$sum_new_vaccinations',
        _id: 0
    }
}])
```

- диаграммы общего количества новых заболевших в сутки за определённый период пандемии

  Используется одна коллекция

```
db.cases.aggregate([{
    $match: {
        iso_code: {
            $in: ['RU']
        } or None,
        date: {
            $gte: ISODate('2021-10-08'),
            $lte: ISODate('2021-10-09')
        }
    }
}, {
    $group: {
        _id: "$date",
        sum_disease_new_cases: {
            $sum: "$new_cases"
        }
    }
}, {
    $project: {
        date: '$_id',
        sum_disease_new_cases: '$sum_disease_new_cases',
        _id: 0
    }
}])
```

- Зависимость количества заболевших от плотности населения

```
db.cases.aggregate([{
    $match: {
        date: {
            $gte: ISODate('2021-10-08'),
            $lte: ISODate('2021-10-09')
        }
    }
}, {
    $group: {
        _id: "$iso_code",
        total_cases: {
            $sum: "$new_cases"
        }
    }
}, {
    $project: {
        iso_code: '$_id',
        total_cases: '$total_cases',
        _id: 0
```

```
    }
}, {
    $lookup: {
        from: 'countries',
        localField: 'iso_code',
        foreignField: 'iso_code',
        as: 'countries'
    }
}, {
    $project: {
        iso_code: '$iso_code',
        total_cases: '$total_cases',
        countries: {$arrayElemAt: ['$countries', 0]}
    }
}, {
    $project: {
        iso_code: '$iso_code',
        total_cases: '$total_cases',
        population_density: '$countries.population_density'\
    }
}])
```

## 3.2 Реляционная база данных

База данных содержит три таблицы:

- Countries

| Поле | Тип данных | Описание |
|---|---|---|
| *iso_code* | VarChar(8) | ISO 3166-1 код страны |
| *continent* | VarChar(30) | Континент географического положения |
| *location* | VarChar(30) | Географическое положение |
| *population* | Integer | Население в 2020 |
| *population_density* | Double | Плотность населения |
| *median_age* | Double | Средний возраст населения, прогноз ООН на 2020 год |
| *aged_65_older* | Double | Доля населения в возрасте 65 лет и старше, по данным за последний год |
| *aged_70_older* | Double | Доля населения в возрасте 70 лет и старше в 2015 г. |

- Cases

| Поле | Тип данных | Описание |
|---|---|---|
| *iso_code* | VarChar(8) | ISO 3166-1 код страны |
| *date* | DateTime | Дата наблюдения |
| *total_cases* | Integer | Всего подтвержденных случаев COVID-19 |
| *new_cases* | Integer | Новые подтвержденные случаи COVID-19 |
| *new_cases_smoothed* | Double | Новые подтвержденные случаи COVID-19 (сглаживание за 7 дней) |
| *total_cases_per_million* | Double | Всего подтвержденных случаев COVID-19 на 1.000.000 человек |

| | | |
|---|---|---|
| *new_cases_per_million* | Double | Новые подтвержденные случаи COVID-19 на 1.000.000 человек |
| *new_cases_smoothed_per_million* | Double | Новые подтвержденные случаи COVID-19 (сглаживание за 7 дней) на 1.000.000 человек |

- Vaccinations

| **Поле** | **Тип данных** | **Описание** |
|---|---|---|
| *iso_code* | VarChar(8) | ISO 3166-1 код страны |
| *date* | DateTime | Дата наблюдения |
| *people_vaccinated* | Integer | Общее количество людей, получивших хотя бы одну дозу вакцины |
| *people_fully_vaccinated* | Integer | Общее количество людей, получивших все дозы, предписанные протоколом вакцинации |
| *new_vaccinations* | Integer | Введены новые дозы вакцинации против COVID-19 (рассчитываются только для последовательных дней) |
| *new_vaccinations_smoothed* | Double | Введены новые дозы вакцинации против COVID-19 (7-дневное сглаживание). Для стран, которые не сообщают данные о вакцинации на ежедневной основе, мы предполагаем, что вакцинация менялась одинаково ежедневно в течение всех периодов, за которые данные не были представлены. Это дает полную серию ежедневных показателей, которые затем усредняются по скользящему 7-дневному окну. |
| *total_vaccinations_per_hundred* | Double | Общее количество доз вакцины против COVID-19 на 100 человек в общей численности населения |
| *people_vaccinated_per_hundred* | Double | Общее количество людей, получивших хотя бы одну дозу вакцины, на 100 человек в общей численности населения |
| *people_fully_vaccinated_per_hundred* | Double | Общее количество людей, получивших все дозы, предписанные протоколом вакцинации, на 100 человек в общей численности населения |

### 3.2.1 Графическое представление

Графическое представление SQL модели данных представлено на рис. 2.

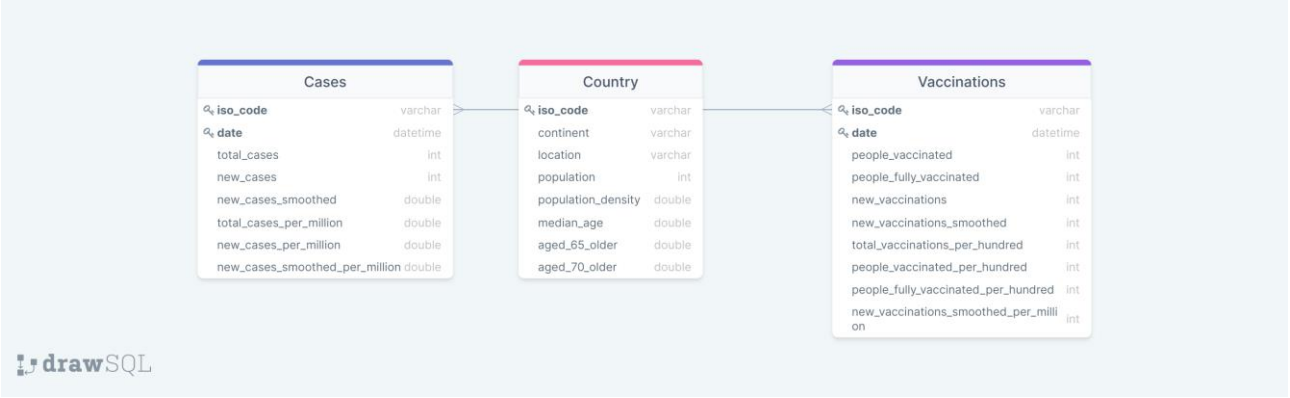


Рисунок 2. Графическое представление SQL модели данных.

### 3.2.2 Оценка объема одного ряда таблицы

- Countries
  - iso_code: VarChar(8), 8 байт
  - continent: VarChar(30), 30 байт
  - location: VarChar(30), 30 байт
  - population: Integer, 8 байт
  - population_density: Double, 8 байт
  - median_age: Double, 8 байт
  - aged_65_older: Double, 8 байт
  - aged_70_older: Double, 8 байт

Итого: для одного ряда таблицы *Countries* нужно:

$$V_{co} = 8 + 30 + 30 + 8 + 8 + 8 + 8 + 8 = 108 \text{ байт}$$

- Cases
  - iso_code: VarChar(8), 8 байт
  - date: DateTime, 8 байт
  - total_cases: Integer, 8 байт
  - new_cases: Integer, 8 байт
  - new_cases_smoothed: Double, 8 байт
  - total_cases_per_million: Double, 8 байт
  - new_cases_per_million: Double, 8 байт
  - new_cases_smoothed_per_million: Double, 8 байт

Итого: для одного ряда таблицы *Cases* нужно:

$$V_{ca} = 8 + 8 + 8 + 8 + 8 + 8 + 8 + 8 = 64 \text{ байта}$$

- Vaccinations
  - iso_code: VarChar(8), 8 байт
  - date: DateTime, 8 байт
  - people_vaccinated: Integer, 8 байт
  - people_fully_vaccinated: Integer, 8 байт
  - new_vaccinations: Integer, 8 байт
  - new_vaccinations_smoothed: Double, 8 байт
  - total_vaccinations_per_hundred: Double, 8 байт
  - people_vaccinated_per_hundred: Double, 8 байт
  - people_fully_vaccinated_per_hundred: Double, 8 байт
  - new_vaccinations_smoothed_per_million: Double, 8 байт

Итого: для одного ряда таблицы *Vaccinations* нужно:

$$V_v = 8 + 8 + 8 + 8 + 8 + 8 + 8 + 8 + 8 + 8 = 80$$

### 3.2.3 Расчет объема данных SQL-like

Объем данных, требуемый для хранения $N_{co}$ стран и $N_{ca}$ заболеваний и $N_v$ вакцинаций:

$$V(N_{co}, N_{ca}, N_v) = N_{co} \cdot V_{co} + N_{ca} \cdot V_{ca} + N_v \cdot V_v$$

Каждый день добавляется по 1 ряду в таблицы *Cases* и *Vaccinations*. В предположении, на одну страну нужно ~1000 записей (на 3 года), то:

$$N_v = N_{ca} = 1000 \cdot N_{co}$$

$$V(N_{co}) = N_{co} \cdot V_{co} + 1000 \cdot N_{co} \cdot V_{ca} + 1000 \cdot N_{co} \cdot V_v =$$

$$= N_{co} \cdot (108 + 1000 \cdot 64 + 1000 \cdot 80) = 144108 \cdot N_{co}$$

Пусть $N_{co}$ равен 20, тогда

$$V(20) = 20 \cdot 144108 = 2882160 \text{ Б} = 2.8 \text{ МБ}$$

### 3.2.4 Расчет чистого объема данных

Объем данных, необходимых для таблицы *countries* не будет изменён.

$$V_{co}[pure] = 108 \text{ байт}$$

Объем данных, необходимых для таблицы *cases* не должен учитывать объём поля *iso_code*. $V_{ca}[pure] = 56$ байт

Объем данных, необходимых для таблицы *vaccinations* не должен учитывать объём поля *iso_code*. $V_v[pure] = 72$ байт

$$\frac{V_{pure}}{V} = \frac{V_{co}[pure] + V_{ca}[pure] + V_v[pure]}{V_{co} + V_{ca} + V_v} = \frac{108 + 56 + 72}{108 + 64 + 80} = 0.94$$

### 3.2.5 Запросы

- максимальное/минимальное/среднее/суммарное количество новых заболевших в сутки за весь/заданный период пандемии

```
1.  SELECT MAX(new_cases) AS max_cases,
2.         MIN(new_cases) AS min_cases,
3.         AVG(new_cases) AS avg_cases,
4.         SUM(new_cases) AS sum_cases,
5.  FROM (
6.      SELECT Cases.date, Cases.iso_code, SUM(Cases.new_cases) AS new_cases FROM Cases
7.      WHERE (Cases.date >= DATE '2021-10-08') AND (Cases.date <= DATE '2021-10-09') AND
    (Cases.iso_code="RU")
8.      GROUP BY Cases.iso_code
9.  )
```

- максимальное/минимальное/среднее/суммарное количество новых вакцинированных в сутки за весь/заданный период пандемии

```
1.  SELECT MAX(new_vaccinations) AS max_new_vaccinations,
2.         MIN(new_vaccinations) AS min_new_vaccinations,
3.         AVG(new_vaccinations) AS avg_new_vaccinations,
4.         SUM(new_vaccinations) AS sum_new_vaccinations,
5.  FROM (
6.      SELECT Vaccinations.date, Vaccinations.iso_code, SUM(Vaccinations.new_vaccinations) AS
    new_vaccinations
7.      FROM Vaccinations
8.      WHERE (Vaccinations.date >= DATE '2021-10-08') AND (Vaccinations.date <= DATE '2021-10-09')
    AND (Cases.iso_code="RU")
9.      GROUP BY Vaccinations.iso_code
10. )
```

- диаграммы общего количества новых заболевших в сутки за определённый период пандемии

```
1.  SELECT Cases.date, Cases.iso_code, SUM(Cases.new_cases) AS total_cases FROM Cases
2.  WHERE (Cases.date >= DATE '2021-10-08') AND (Cases.date <= DATE '2021-10-09') AND
    (Cases.iso_code == "RU")
```

- Зависимость количества заболевших от плотности населения

```
1.  SELECT Country.iso_code, Country.population_density, cases FROM Country
2.  INNER JOIN (
3.      SELECT iso_code, SUM(Cases.new_cases) AS cases FROM Cases
4.      WHERE (Cases.date >= DATE '2021-10-08') AND (Cases.date <= DATE '2021-10-09')
5.      GROUP BY Cases.iso_code
6.  ) AS T
7.  ON Country.iso_code=T.iso_code
```

## 3.3 Сравнение реляционной и нереляционной БД

- Избыточность модели
    i. Для нереляционной БД – 0.94
    ii. Для реляционной БД – 0.94
- Объем данных
    i. Для нереляционной БД – 3202320 Б = 3 МБ
    ii. Для реляционной БД – 2882160 Б = 2.8 МБ
- Сложность запросов
    - Поиск максимального количество заболевших в день за определенный период
        i. Для нереляционной БД – $O(N)$, $N$ - количество наблюдений
        ii. Для реляционной БД – $O(N)$, $N$ - количество наблюдений
    - Поиск зависимости количества заболевших от плотности населения за определенный период.
        i. Для нереляционной БД – $(N^2 \cdot M)$, $N$ – количество наблюдений, $M$ – количество стран
        ii. Для реляционной БД - $(N^2 \cdot M)$, $N$ – количество наблюдений, $M$ – количество стран

Исходя из полученных во время сравнения значений можно сделать вывод, что для данной задачи нет принципиальной разницы для выбора СУБД. Стоит исходить из удобства использования и надежности СУБД.

Плюсы использования РСУБД:

- Незначительно меньшее занимаемое пространство
- Более короткие запросы к БД.

Плюсы использования НСУБД MongoDB:

- Удобный инструментарий для просмотра содержимого и отладки запросов БД
- Гибкость документов, которая позволяет не занимать память для неиспользуемых полей.
- Интегрированность БД, которая защищает данные от разного рода инъекций.

# 4. РАЗРАБОТАННОЕ ПРИЛОЖЕНИЕ

Разработанное приложение хранит данные о случаях Covid-19. Позволяет просмотреть диаграмму динамики новых случаев в сутки, а также количество вакцинированных.

Данные для отображения определяются диапазоном дат и страной, которую вводит пользователь, как для новых случаев заболевания, так и для вакцинаций возможно применить одну из четырех агрегирующих функций: сумма, среднее, минимальное, максимальное число.

Также имеется возможность отобразить диаграмму новых случаев заболевания двух стран одновременно на одном диапазоне дат для дальнейшего сравнения.

Доступна функция по просмотру зависимости количества заболевших от плотности населения.

Все данные, которыми оперирует приложение можно также просмотреть в виде таблиц коллекций с возможностью применения различной фильтрации.

## 4.1 Использованные технологии

СУБД: MongoDB

Сервер: Python, Flask, PyMongo

Клиент: JS, React, Material-UI, ChartJS

## 4.2 Ссылка на приложение

*GitHub release*: https://github.com/moevm/nosql2h21-covid-mongo/releases/latest

## 4.3 Схема экранов приложения

# ВЫВОД

В ходе выполнения работы было разработано приложение для анализа случаев COVID. В качестве СУБД используется MongoDB, для данной задачи было проведено сравнение нереляционной и реляционной модели.

**Недостатки полученного решения**

Основным недостатком приложения является отсутствие функции автообновления базы данных во время работы сервера. На данный момент для обновления данных нужно либо вручную импортировать файл с данными через интерфейс, либо пересобирать образ, чтобы данные синхронизировались с репозиторием. Простым решением данной проблемы является установка таймера, который через определенные промежутки времени сверяется с репозиторием. С другой стороны, можно настроить сервис, который будет получать оповещения с *GitHub* каждый раз, когда вносятся изменения в репозиторий.

Менее заметна проблема с использованной в приложении библиотекой *ChartJS,* которая рендерит диаграммы, она достаточно нестабильна при работе с динамически изменяемыми данными, в некоторые моменты времени можно наблюдать различные визуальные артефакты. Для решения этих проблем необходимо переработать компоненты диаграмм или выбрать другое решение.

**Будущее развитие решения.**

В первую очередь необходимо обеспечить адаптивную верстку страниц. Использованная в приложении библиотека Material-UI имеет достаточный инструментарий для реализации адаптивных компонент.

Запросы к серверной части приложения имеют достаточно разрозненную структуру, из-за чего сильно усложняется их обработка, стандартизация запросов позволит создать единые интерфейсы для их обработки.

Так же в доработке нуждается UX/UI, выборы диапазонов лучше реализовать на самих диаграммах, а страны удобнее выбирать на схематичной карте мира.