

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ ОБРАЗОВАТЕЛЬНОЕ
УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ «САНКТ-ПЕТЕРБУРГСКИЙ
НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ ИНФОРМАЦИОННЫХ
ТЕХНОЛОГИЙ, МЕХАНИКИ И ОПТИКИ»

Факультет инфокоммуникационных технологий
09.03.03 «Мобильные и сетевые технологии»

Дисциплина
«ПРОЕКТИРОВАНИЕ И РЕАЛИЗАЦИЯ БАЗ ДАННЫХ»

ОТЧЕТ ПО ЛАБОРАТОРНОЙ РАБОТЕ №6
«Знакомство с MongoDB»

Выполнил: Пузырев Д.А.
Студент группы К3241

Проверил: Говоров А.И.
Преподаватель

Санкт-Петербург
2020 г.

Цель работы

Овладеть практическими навыками реализации баз данных в MongoDB.

Описание варианта

Система должна обеспечивать хранение сведений об имеющихся в гостинице номерах, о проживающих в гостинице клиентах и о служащих, убирающихся в номерах. Количество номеров в гостинице известно, и имеются номера трех типов: одноместный, двухместный и трехместный, отличающиеся стоимостью проживания в сутки. В каждом номере есть телефон. О каждом проживающем должна храниться следующая информация: номер паспорта, фамилия, имя, отчество, город, из которого он прибыл, дата поселения в гостинице, выделенный гостиничный номер. О служащих гостиницы должна храниться информация следующего содержания: фамилия, имя, отчество, где (этаж) и когда (день недели) он убирает. Служащий гостиницы убирает все номера на одном этаже в определенные дни недели, при этом в разные дни он может убирать разные этажи.

Работа с системой предполагает получение следующей информации:

- о клиентах, проживавших в заданном номере, в заданный период времени;
- о количестве клиентов, прибывших из заданного города;
- кто из служащих убирал номер указанного клиента в заданный день недели;
- сколько в гостинице свободных номеров;
- список клиентов с указанием места жительства, которые проживали в те же дни, что и заданный клиент, в определенный период времени.

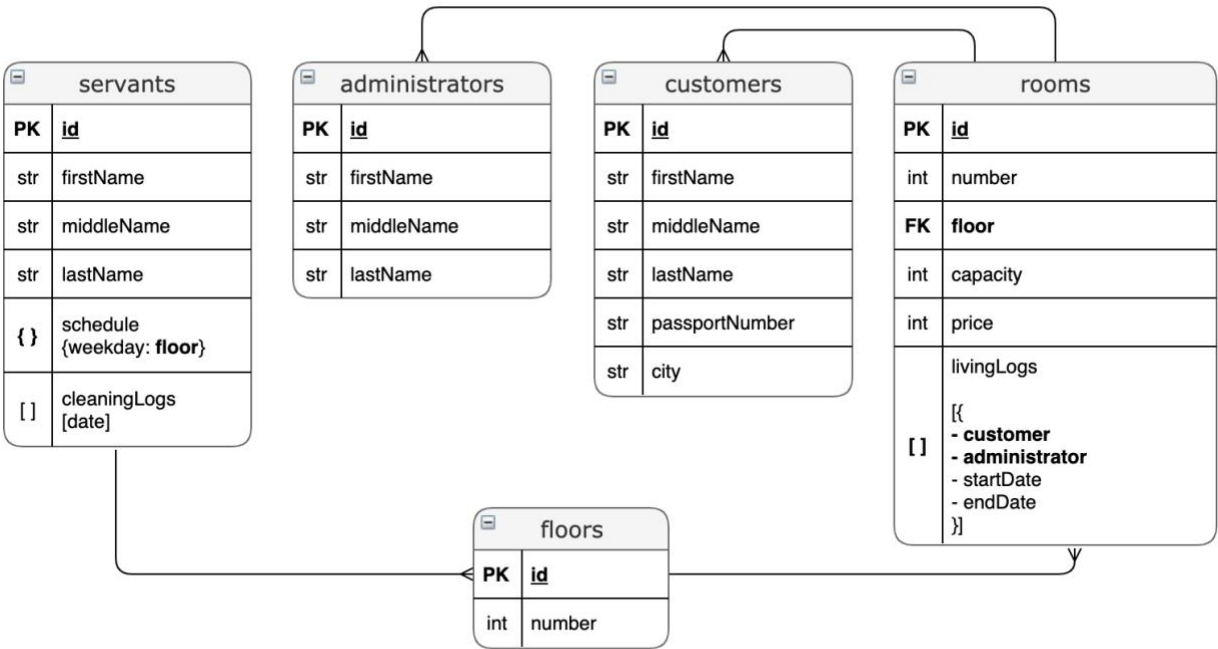
Администратор должен иметь возможность выполнить следующие операции:

- принять на работу или уволить служащего гостиницы;
- изменить расписание работы служащего;
- поселить или выселить клиента.

Необходимо предусмотреть также возможность автоматической выдачи отчета о работе гостиницы за указанный квартал текущего года. Такой отчет должен содержать следующие сведения:

- число клиентов за указанный период в каждом номере с указанием ФИО клиента, города, откуда он прибыл, количества дней проживания;
- общая сумма дохода за каждый номер;
- суммарный доход по всей гостинице.

Новая модель базы данных



Создание и заполнение базы данных

```
f1 = ({number: 1})
f2 = ({number: 2})
f3 = ({number: 3})

db.floors.save(f1)
db.floors.save(f2)
db.floors.save(f3)

db.servants.insert([
  {
    firstName: "Анфиса",
    middleName: "Михайловна",
    lastName: "Богуш",
    schedule: {monday: f1, thursday: f3},
    cleaningLogs: [new Date('2020-06-08'), new Date('2020-06-11')]
  },
  {
    firstName: "Михаил",
    middleName: "Анатольевич",
    lastName: "Борисенко",
    schedule: {monday: f2, thursday: f2},
    cleaningLogs: [new Date('2020-06-08'), new Date('2020-06-11')]
  },
  {
    firstName: "Андрей",
    middleName: "Александрович",
    lastName: "Курпатов",
    schedule: {monday: f3, thursday: f1},
    cleaningLogs: [new Date('2020-06-08'), new Date('2020-06-11')]
  },
])

a1 = ({firstName: "Дмитрий", middleName: "Андреевич", lastName: "Пузырев"})
a2 = ({firstName: "Василий", middleName: "Михайлович", lastName: "Пупкин"})
a3 = ({firstName: "Андрей", middleName: "Викторович", lastName: "Назаров"})

db.administrators.save(a1)
db.administrators.save(a2)
db.administrators.save(a3)

c1 = ({firstName: "Александр", middleName: "Васильевич", lastName: "Прохоров",
passportNumber: "АС 259411", city: "Москва"})
c2 = ({firstName: "Андрей", middleName: "Михайлович", lastName: "Пингвинин",
passportNumber: "ВЕ 544108", city: "Санкт-Петербург"})
c3 = ({firstName: "Анна", middleName: "Михайловна", lastName: "Полякова",
passportNumber: "АС 531266", city: "Екатеринбург"})
c4 = ({firstName: "Евгения", middleName: "Михайловна", lastName: "Братонюк",
passportNumber: "АС 423422", city: "Санкт-Петербург"})
```

```
db.customers.save(c1)
db.customers.save(c2)
db.customers.save(c3)
db.customers.save(c4)

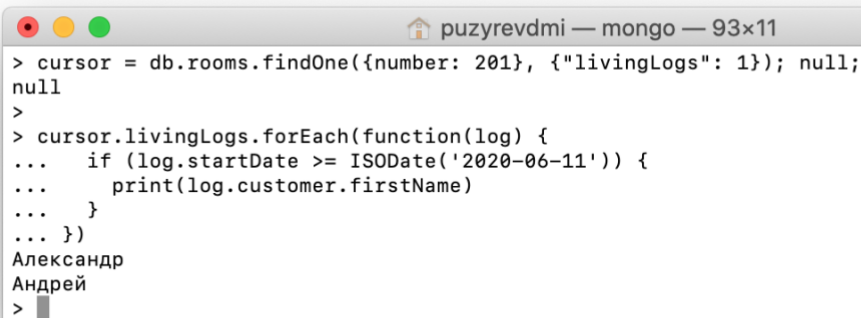
db.rooms.insert([
  {
    number: 101,
    capacity: 1,
    price: 1000,
    floor: f1,
    livingLogs: [
      {
        customer: c1,
        administrator: a1,
        startDate: new Date('2020-06-08'),
        endDate: new Date('2020-06-10'),
      }
    ]
  },
  {
    number: 201,
    capacity: 3,
    price: 2500,
    floor: f2,
    livingLogs: [
      {
        customer: c1,
        administrator: a1,
        startDate: new Date('2020-06-11'),
        endDate: new Date('2020-06-14'),
      },
      {
        customer: c2,
        administrator: a3,
        startDate: new Date('2020-06-14'),
      }
    ]
  },
  {
    number: 301,
    capacity: 2,
    price: 2000,
    floor: f3,
    livingLogs: [
      {
        customer: c3,
        administrator: a2,
        startDate: new Date('2020-06-08'),
        endDate: new Date('2020-06-12')
      }
    ]
  },
])
```

Запросы к базе данных

1. Имена клиентов, проживавших в номере с number = 201, начиная с 12 июня 2020 года:

```
cursor = db.rooms.findOne({number: 201}, {"livingLogs": 1});

cursor.livingLogs.forEach(function(log) {
  if (log.startDate >= ISODate('2020-06-11')) {
    print(log.customer.firstName)
  }
})
```



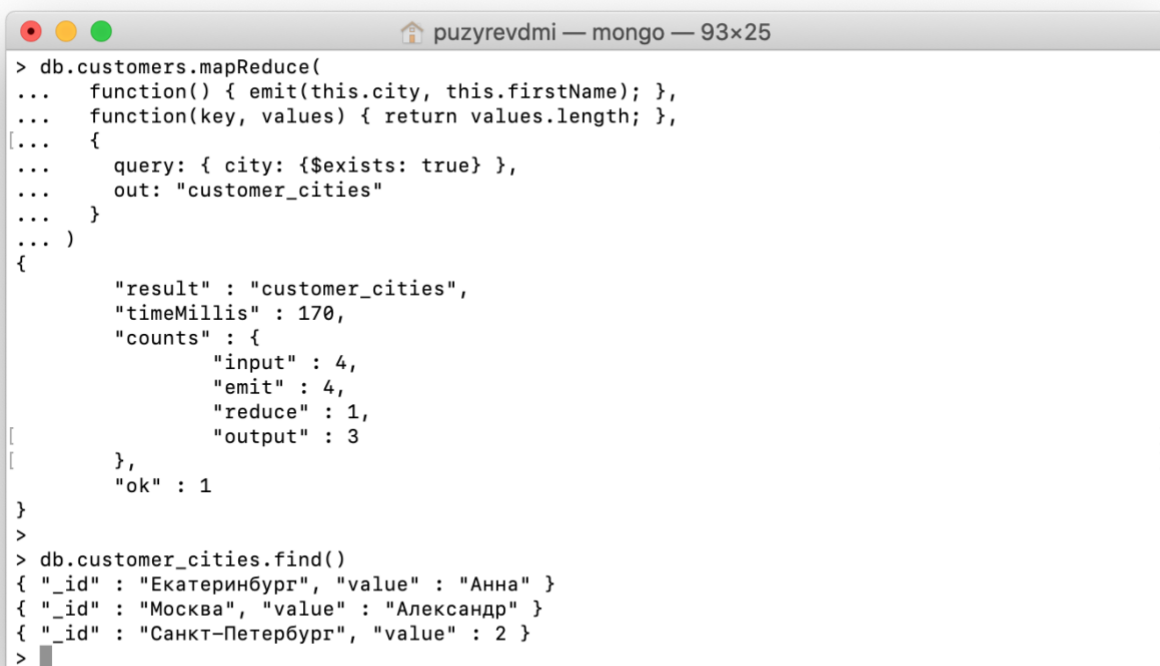
The screenshot shows a terminal window titled "puzyrevdmi — mongo — 93x11". The user enters the following commands:

```
> cursor = db.rooms.findOne({number: 201}, {"livingLogs": 1}); null;
null
>
> cursor.livingLogs.forEach(function(log) {
...   if (log.startDate >= ISODate('2020-06-11')) {
...     print(log.customer.firstName)
...   }
... })
Александр
Андрей
>
```

The output shows "null" for the first command, and the names "Александр" and "Андрей" for the second command, indicating that these two customers stayed in room 201 on or after June 12, 2020.

2. Количество клиентов по городам:

```
db.customers.mapReduce(  
  function() { emit(this.city, this.firstName); },  
  function(key, values) { return values.length; },  
  {  
    query: { city: {$exists: true} },  
    out: "customer_cities"  
  }  
)  
  
db.customer_cities.find()
```

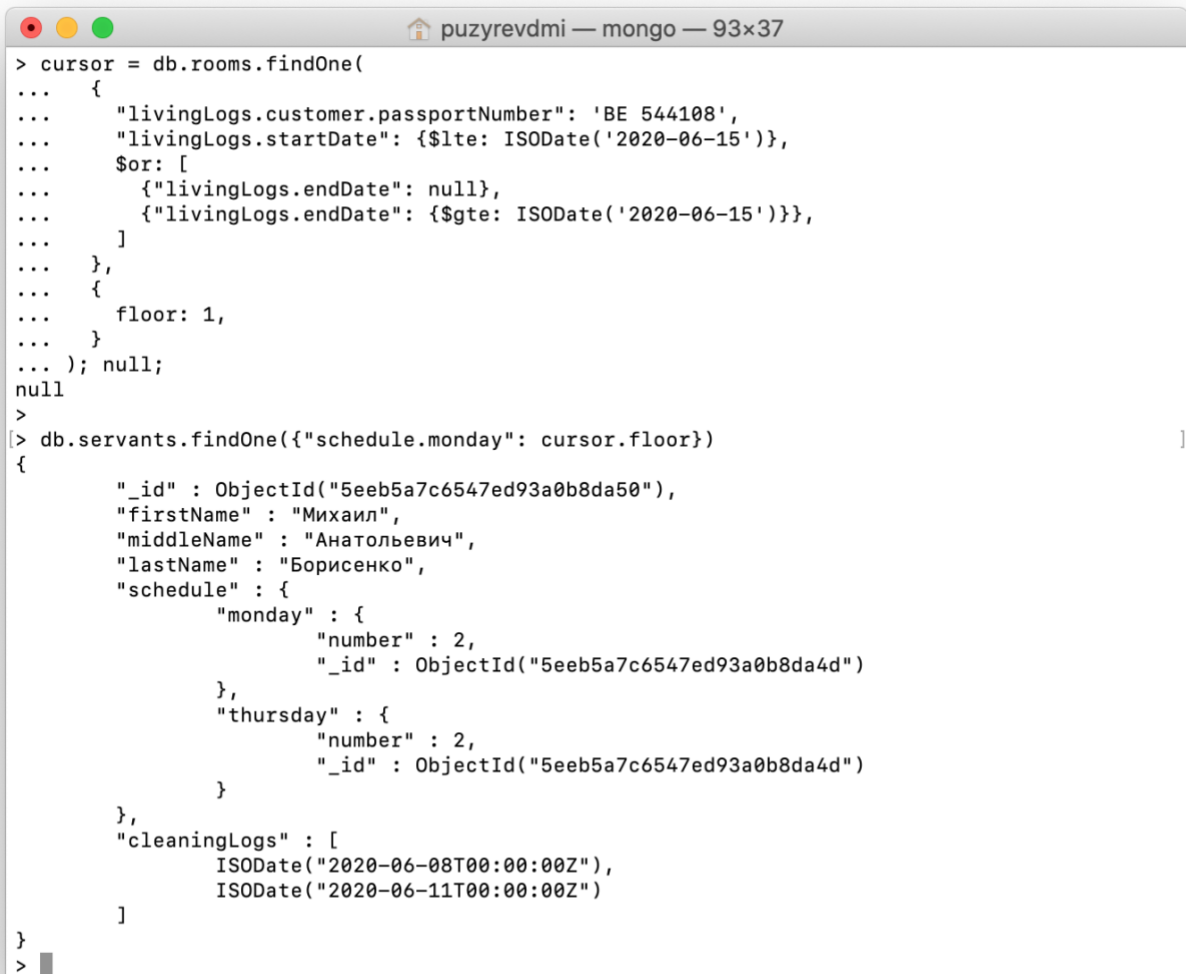


The screenshot shows a MongoDB shell window with the title "puzyrevdmi — mongo — 93x25". The user has entered a mapReduce command, and the shell has returned a JSON response. The response indicates that the mapReduce operation was successful, with 4 input records, 4 emit records, 1 reduce record, and 3 output records. The output is stored in the "customer_cities" collection. The user then runs a find command on the "customer_cities" collection, which returns three documents representing the cities and the number of customers for each.

```
> db.customers.mapReduce(  
...   function() { emit(this.city, this.firstName); },  
...   function(key, values) { return values.length; },  
[... {  
...     query: { city: {$exists: true} },  
...     out: "customer_cities"  
...   }  
... )  
{  
  "result" : "customer_cities",  
  "timeMillis" : 170,  
  "counts" : {  
    "input" : 4,  
    "emit" : 4,  
    "reduce" : 1,  
    "output" : 3  
  },  
  "ok" : 1  
}  
>  
> db.customer_cities.find()  
{ "_id" : "Екатеринбург", "value" : "Анна" }  
{ "_id" : "Москва", "value" : "Александр" }  
{ "_id" : "Санкт-Петербург", "value" : 2 }  
>
```

3. Кто из служащих убирал номер клиента с passportNumber = 'BE 544108' 15 июня 2020 года:

```
cursor = db.rooms.findOne(  
  {  
    "livingLogs.customer.passportNumber": 'BE 544108',  
    "livingLogs.startDate": {$lte: ISODate('2020-06-15')},  
    $or: [  
      {"livingLogs.endDate": null},  
      {"livingLogs.endDate": {$gte: ISODate('2020-06-15')}}],  
    ],  
  },  
  {  
    floor: 1,  
  }  
)  
  
db.servants.findOne({"schedule.monday": cursor.floor})
```

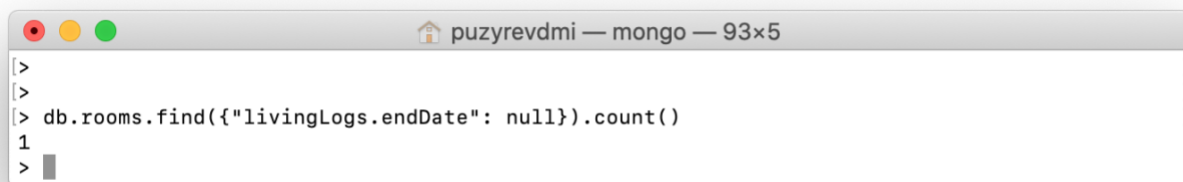


The screenshot shows a MongoDB terminal window with the title 'puzyrevdmi — mongo — 93x37'. The user enters a query to find a room and then a servant based on that room's floor. The terminal output shows the room document is null, and the servant document is a JSON object containing personal information and a cleaning schedule.

```
> cursor = db.rooms.findOne(  
...  {  
...    "livingLogs.customer.passportNumber": 'BE 544108',  
...    "livingLogs.startDate": {$lte: ISODate('2020-06-15')},  
...    $or: [  
...      {"livingLogs.endDate": null},  
...      {"livingLogs.endDate": {$gte: ISODate('2020-06-15')}}],  
...    ],  
...  },  
...  {  
...    floor: 1,  
...  }  
... ); null;  
null  
>  
> db.servants.findOne({"schedule.monday": cursor.floor})  
{  
  "_id" : ObjectId("5eeb5a7c6547ed93a0b8da50"),  
  "firstName" : "Михаил",  
  "middleName" : "Анатольевич",  
  "lastName" : "Борисенко",  
  "schedule" : {  
    "monday" : {  
      "number" : 2,  
      "_id" : ObjectId("5eeb5a7c6547ed93a0b8da4d")  
    },  
    "thursday" : {  
      "number" : 2,  
      "_id" : ObjectId("5eeb5a7c6547ed93a0b8da4d")  
    }  
  },  
  "cleaningLogs" : [  
    ISODate("2020-06-08T00:00:00Z"),  
    ISODate("2020-06-11T00:00:00Z")  
  ]  
}
```


4. Количество свободных номеров в гостинице:

```
db.rooms.find({"livingLogs.endDate": null}).count()
```



The screenshot shows a terminal window titled "puzyrevdmi — mongo — 93x5". The terminal contains the following text:

```
>  
>  
> db.rooms.find({"livingLogs.endDate": null}).count()  
1  
> █
```

Вывод

В ходе выполнения работы были освоены навыки создания базы данных с помощью MongoDB. Получены навыки написания запросов к базе данных.