ПРАКТИЧЕСКАЯ РАБОТА 5. ЗАПРОСЫ И ЗАПРОСЫ С УСЛОВИЕМ В MONGODB

5.1 Цель и содержание

Цель работы: знакомство с запросами в MongoDB.

Задачи работы: научиться выполнять различные виды запросов к базе данных mongo.

5.2 Простые запросы

Аналогом SELECT из SQL в MongoDB является «Find». Метод «Find» используется для выборки документов из MongoDB. Возвращает массив документов в виде коллекции, если документов нет – пустую коллекцию.

Синтаксис метода «Find»: db.<коллекция>.find(<запрос>, <поля>),где:

<запрос> – критерий отбора с помощью формального запроса операторов;

<поля> – задает поля, которые будут возвращены в результате обработки запроса, и поле «_id», если не указано.

Оба параметра функции не являются обязательными. При вызовефункции без параметров возвратятся все документы коллекции.

Примеры использования Find:

– получение всех документов коллекции:

```
db.phones.find()
```

– получение документов коллекции, в которых поле «Brand» равно «Apple»:

```
db.phones.find({ Brand: “Apple”})
```

– получение документов коллекции, в которых поле «Brand» равно «Nokia» и «OSFamily» равно «Windows»:

```
db.phones.find({ Brand: “Nokia”, OSFamily: “Windows”})
```

Для получения только одного поля документа для всех документов необходимо ввести:

```
db.phones.find(null, {Brand: 1})
или
db.phones.find({}, {Brand: 1})
```

Поле «_id» возвращается всегда, но можно явно исключить его:

```
db.phones.find(null, {Brand: 1, _id: 0})
```

При составлении запроса нельзя смешивать включения и исключения полей. Исключение составляет только поле «_id».

Можно или включить или исключить определенные поля из запроса.

5.3 Сортировка, постраничный вывод результата запроса

Для сортировки используется метод «sort». Синтаксис метода: sort(<параметры сортировки>), где <параметры сортировки> – документ, содержащий поля, по которым будет производиться сортировка на результирующем наборе данных.

Используя поля, по которым необходимо сортировать, необходимо указывать 1 для сортировки по возрастанию и -1 для сортировки по убыванию, например:

```
db.phones.find().sort({Brand: 1})
db.phones.find().sort({Brand: 1, cost: -1})
```

Также как и РСУБД, MongoDB может использовать индексы для сортировки. Следует помнить, что без использования индексов MongoDB ограничивает размер сортируемых данных.

Метод «limit» используется для ограничения количества документов, получаемых в результате выполнения запроса.

Синтаксис метода: limit(<количество документов>)

Для получения десяти документов в выводе необходимо вызвать «limit» с параметром «10»:

```
db.phones.find().limit(10)
```

Метод «count» подсчитывает количество документов в результирующей выборке. Синтаксис метода: count(<учитывать skip/limit>), где <учитывать skip/limit> – указывает, следует ли учитывать влияние методов «skip» и «limit». По умолчанию метод игнорирует применение методов «skip» и «limit».

Для получения количества документов в результирующей выборке необходимо ввести:

```
db.phones.find().count()
```

Метод «count» можно также использовать применительно к коллекции, передавая в него запрос из find, например:

```
db.phones.count({Brand: "Nokia"})
```

Метод «skip» используется для пропуска некоторого количества документов результирующей выборки. Синтаксис метода: skip(<количество документов>)

Пропуск первых десяти результатов выглядит следующим образом:

```
db.phones.find().limit(10)
```

Методы «limit» и «skip» используются для постраничного вывода данных. Например, для постраничного получения записей можно использовать следующий прием:

```
Db.phones.find().limit(10)
Db.phones.find().limit(10).skip(10)
```

```
Db.phones.find().limit(10).skip(20)
…
```

5.4 Запросы с условием

Для создания запросов с условием применяются следующиеоператоры:

$lt – меньше чем;

$lte – меньше или равно;

$gt – больше;

$gte – больше или равно;

$ne – не равно;

$in – вхождение в массив значений;

$nin – противоположность оператора $in;

$not – логический оператор НЕ.

Синтаксис модификаторов:

{ <поле>: {<модификатор>: <значение>}}, где:

<поле> – имя поля, для которого применяется модификатор;

<модификатор> – модификатор;

<значение> – значение модификатора.

В качестве примера рассмотрим запрос для выбора всех телефонов стоимостью от 10000 до 20000 тыс. руб. Запрос выглядит следующим образом:

```
db.phones.find( { cost: { $gte: 10000, $lte: 20000 } } );
```

Запрос для получения всех телефонов, выпущенных не фирмой «Apple»:

```
db.phones.find( {Brand: {$ne: “Apple”}})
```

Получение всех телефонов, чья стоимость равна 10000, 15000, или20000 тыс. руб.:

```
db.phones.find({cost: {$in [10000, 15000, 20000]}})
```

Операторы $or и $and применяются, когда нужно выбрать документы по совпадению одного из значений или по совпадению всех значений соответственно. Являются реализациями логических операций ИЛИ и И.

Синтаксис:

{ $or($and): [{ <Выражение1> }, { <выражение2> }, ... }] }, где:

<Выражение i> – логическое выражение, применяемое для сравнения. Например, для того чтобы выбрать телефоны, выпущенные фирмой «Samsung» или имеющие дисплей более 4,6 дюйма:

```
db.phones.find({$or: [{Brand: “Samsung”}, {DisplayDiag: {$gte: 4.6}}]})
```

5.5 Методика и порядок выполнения работы

Индивидуальное задание:

1. Создайте пару простых запросов для выборки данных из БД.
2. Создайте сложные запросы с каждым из перечисленных модификаторов.
3. Создайте запросы с использованием методов сортировки, ограничения и пропуска данных.
4. Всего у вас должно получиться не менее десяти уникальных запросов.