**Лабораторная работа №2**

**Средства поиска данных в базе MongoDB**

Целью запроса к базе MongoDB являются документы из определенной коллекции. Для поиска (отбора) документов предназначены методы коллекции find и findOne. Вызовы методов find и findOne имеют одинаковую структуру и содержат критерий отбора документов (селектор), имена выводимых атрибутов и правило сортировки выбранных документов. Различие методов в том, что метод find возвращает все документы, соответствующие критерию отбора, а метод findOne — только один (первый) документ из удовлетворяющих селектору.

Cледует выполнять команды, выделенные **жирным шрифтом**

Основной синтаксис вызова метода find ()имеет вид:

db.<Коллекция>.find ([{<Селектор>}

[, {<Список атрибутов>}]]);

В простейшем виде запрос db.<Коллекция>.find (); возвращает все документы заданной коллекции. Отсутствие селектора и списка атрибутов приводит к выводу всех документов с полными наборами их атрибутов. Параметр <список атрибутов> содержит имена выводимых или, напротив, исключаемых из вывода атрибутов найденных документов. Кроме параметров find и findOne имеют присоединенные методы, выполняющие сортировку, ограничение количества и способ вывода найденных документов. С присоединенными методами запрос имеет вид:

db.<Коллекция>.find ([{<Селектор>}

[, {<Список атрибутов>}]])

[.sort ({<Атрибут>: 1 |-1, …})]

[.limit (<Число выводимых документов>)]

[.skip (<Количество пропускаемых документов>)]

[.pretty ()]; — выводит каждый атрибут в отдельной строке.

Количество документов, удовлетворяющих запросу, возвращает метод коллекции count:

db.<Коллекция>.count ({<Селектор>}).

Вывод различных значений (удаление дубликатов) определенного атрибута в коллекции выполняет метод db.<Коллекция>.distinct (<Имя атрибута>). Например, запрос

**db.authors.distinct ("Au\_lname")**

вернет массив из фамилий авторов, выбранных из атрибута Au\_lname в документах коллекции authors

[ "Лесков", "Н. В. Гоголь" ].

Селектор содержит критерий для выбора документов и записывается в формат JSON-объекта: {<Условия на значения атрибутов>}. Пустой селектор, который обеспечивает выборку всех документов из коллекции, задается отсутствием условия {} или Null. Отдельные условия и логические операции в селекторе также записываются в формате JSON.

Виды условий в селекторе

1.<Атрибут>:<Значение> соответствует условию <Атрибут>=<Значение>.

Например, поиск документа по фамилии автора:

**db.authors.find ({"Au\_lname": "Лесков"});**

выдаст

{ "\_id" : ObjectId("58695c396b9421d84b9efb93"), "Au\_lname" : "Лесков", "Au\_fname" : "Николай", "birthday" : ISODate("1831-02-16T00:00:00Z") }

2. Модификаторы сравнений в условии $lt (less than, <), $lte (less than or equal, <=), $gt (greater than, >), $gte (greater than or equal, >=), $ne (not equal, #) задают условие выбора документов в форме неравенства. Для сохранения формата условия модификатор и значение для сравнения записывается в виде JSON-объекта:

{<Модификатор>: <Значение для сравнения>}.

Для демонстрации данного способа создадим коллекцию number

**> db.createCollection ("number")**

Для включения в коллекцию документов используем метод save()

**for(i=0; i<25; i++) {**

**db.number.save({num: i});**

**}**

В результате получим коллекцию

**> db.number.find()**

{ "\_id" : ObjectId("586d240aa1e9f68581a87062"), "num" : 0 }

{ "\_id" : ObjectId("586d240aa1e9f68581a87063"), "num" : 1 }

{ "\_id" : ObjectId("586d240aa1e9f68581a87064"), "num" : 2 }

{ "\_id" : ObjectId("586d240aa1e9f68581a87065"), "num" : 3 }

{ "\_id" : ObjectId("586d240aa1e9f68581a87066"), "num" : 4 }

{ "\_id" : ObjectId("586d240aa1e9f68581a87067"), "num" : 5 }

{ "\_id" : ObjectId("586d240aa1e9f68581a87068"), "num" : 6 }

{ "\_id" : ObjectId("586d240aa1e9f68581a87069"), "num" : 7 }

{ "\_id" : ObjectId("586d240aa1e9f68581a8706a"), "num" : 8 }

{ "\_id" : ObjectId("586d240aa1e9f68581a8706b"), "num" : 9 }

{ "\_id" : ObjectId("586d240aa1e9f68581a8706c"), "num" : 10 }

{ "\_id" : ObjectId("586d240aa1e9f68581a8706d"), "num" : 11 }

{ "\_id" : ObjectId("586d240aa1e9f68581a8706e"), "num" : 12 }

{ "\_id" : ObjectId("586d240aa1e9f68581a8706f"), "num" : 13 }

{ "\_id" : ObjectId("586d240aa1e9f68581a87070"), "num" : 14 }

{ "\_id" : ObjectId("586d240aa1e9f68581a87071"), "num" : 15 }

{ "\_id" : ObjectId("586d240aa1e9f68581a87072"), "num" : 16 }

{ "\_id" : ObjectId("586d240aa1e9f68581a87073"), "num" : 17 }

{ "\_id" : ObjectId("586d240aa1e9f68581a87074"), "num" : 18 }

{ "\_id" : ObjectId("586d240aa1e9f68581a87075"), "num" : 19 }

Выводится первые 20 результатов. Чтобы вывести следующую порцию, надо выполнить команду

**> it**

{ "\_id" : ObjectId("586d240aa1e9f68581a87076"), "num" : 20 }

{ "\_id" : ObjectId("586d240aa1e9f68581a87077"), "num" : 21 }

{ "\_id" : ObjectId("586d240aa1e9f68581a87078"), "num" : 22 }

{ "\_id" : ObjectId("586d240aa1e9f68581a87079"), "num" : 23 }

{ "\_id" : ObjectId("586d240aa1e9f68581a8707a"), "num" : 24 }

Чтобы запросить документы, для которых значение num не равно 5, следует ввести

**> db.number.find( {num: {"$ne":5}})**

{ "\_id" : ObjectId("586d240aa1e9f68581a87062"), "num" : 0 }

{ "\_id" : ObjectId("586d240aa1e9f68581a87063"), "num" : 1 }

{ "\_id" : ObjectId("586d240aa1e9f68581a87064"), "num" : 2 }

{ "\_id" : ObjectId("586d240aa1e9f68581a87065"), "num" : 3 }

{ "\_id" : ObjectId("586d240aa1e9f68581a87066"), "num" : 4 }

{ "\_id" : ObjectId("586d240aa1e9f68581a87068"), "num" : 6 }

{ "\_id" : ObjectId("586d240aa1e9f68581a87069"), "num" : 7 }

{ "\_id" : ObjectId("586d240aa1e9f68581a8706a"), "num" : 8 }

{ "\_id" : ObjectId("586d240aa1e9f68581a8706b"), "num" : 9 }

{ "\_id" : ObjectId("586d240aa1e9f68581a8706c"), "num" : 10 }

{ "\_id" : ObjectId("586d240aa1e9f68581a8706d"), "num" : 11 }

{ "\_id" : ObjectId("586d240aa1e9f68581a8706e"), "num" : 12 }

{ "\_id" : ObjectId("586d240aa1e9f68581a8706f"), "num" : 13 }

{ "\_id" : ObjectId("586d240aa1e9f68581a87070"), "num" : 14 }

{ "\_id" : ObjectId("586d240aa1e9f68581a87071"), "num" : 15 }

{ "\_id" : ObjectId("586d240aa1e9f68581a87072"), "num" : 16 }

{ "\_id" : ObjectId("586d240aa1e9f68581a87073"), "num" : 17 }

{ "\_id" : ObjectId("586d240aa1e9f68581a87074"), "num" : 18 }

{ "\_id" : ObjectId("586d240aa1e9f68581a87075"), "num" : 19 }

{ "\_id" : ObjectId("586d240aa1e9f68581a87076"), "num" : 20 }

Type "it" for more

Чтобы запросить документы, для которых значение num меньше равно 5, следует ввести

**> db.number.find( {num: {"$lte":5}})**

{ "\_id" : ObjectId("586d240aa1e9f68581a87062"), "num" : 0 }

{ "\_id" : ObjectId("586d240aa1e9f68581a87063"), "num" : 1 }

{ "\_id" : ObjectId("586d240aa1e9f68581a87064"), "num" : 2 }

{ "\_id" : ObjectId("586d240aa1e9f68581a87065"), "num" : 3 }

{ "\_id" : ObjectId("586d240aa1e9f68581a87066"), "num" : 4 }

{ "\_id" : ObjectId("586d240aa1e9f68581a87067"), "num" : 5 }

3. Инвертирование (отрицание) условия $not. <Атрибут>:{$not: {<Сравнение>}}}. Модификатор $not применим только с модификаторами сравнения $lt, $lte и т. д. Условию с модификатором $not также удовлетворяют документы, не имеющие проверяемого атрибута.

Например,

**<db.number.find( {num: {$not:{"$gte":5}}})**

{ "\_id" : ObjectId("586d240aa1e9f68581a87062"), "num" : 0 }

{ "\_id" : ObjectId("586d240aa1e9f68581a87063"), "num" : 1 }

{ "\_id" : ObjectId("586d240aa1e9f68581a87064"), "num" : 2 }

{ "\_id" : ObjectId("586d240aa1e9f68581a87065"), "num" : 3 }

{ "\_id" : ObjectId("586d240aa1e9f68581a87066"), "num" : 4 }

4. $regex — регулярные выражения в селекторе (поиск по строковым атрибутам).

Создадим еще одну коллекцию-список

**> db.createCollection ("список")**

Добавим в коллекцию документы

> db.список.insert({"Фамилия и инициалы":"Котов А В","Цех":"Механический","Табельный номер":"01"})

WriteResult({ "nInserted" : 1 })

> db.список.insert({"Фамилия и инициалы":"Котаев П В","Цех":"Сборочный","Табельный номер":"02"})

WriteResult({ "nInserted" : 1 })

> db.список.insert({"Фамилия и инициалы":"Светин П С","Цех":"Сборочный","Табельный номер":"03"})

WriteResult({ "nInserted" : 1 })

> db.список.insert({"Фамилия и инициалы":"Светов В В","Цех":"Литейный","Табельный номер":"03"})

WriteResult({ "nInserted" : 1 })

> db.список.insert({"Фамилия и инициалы":"Ветров П В","Цех":"Сборочный","Табельный номер":"05"})

WriteResult({ "nInserted" : 1 })

Краткая запись условия с регулярным выражением: <Строковый атрибут>:/<Шаблон>/<Опции i и m>,

где шаблон — искомый контекст.

Запрос

**> db.список.find ({"Фамилия и инициалы":/Кот/});**

выдаст результат

{ "\_id" : ObjectId("586d583062ed48682494547e"), "Фамилия и инициалы" : "Котов А В", "Цех" : "Механический", "Табельный номер" : "01" }

{ "\_id" : ObjectId("586d58bc62ed48682494547f"), "Фамилия и инициалы" : "Котаев П В", "Цех" : "Сборочный", "Табельный номер" : "02" }

Опции: i — сравнение выполняется без учета регистра символов,

m — сравнение применяется к многострочным текстам.

Например,

запрос

**> db.список.find ({"Фамилия и инициалы":/ве/i});**

находит

{ "\_id" : ObjectId("586d591162ed486824945480"), "Фамилия и инициалы" : "Светин П С", "Цех" : "Сборочный", "Табельный номер" : "03" }

{ "\_id" : ObjectId("586d597762ed486824945481"), "Фамилия и инициалы" : "Светов В В", "Цех" : "Литейный", "Табельный номер" : "03" }

{ "\_id" : ObjectId("586d59bf62ed486824945482"), "Фамилия и инициалы" : "Ветров П В", "Цех" : "Сборочный", "Табельный номер" : "05" }

5. $in — проверка совпадения значения атрибута с одним из элементов в массиве: {<Атрибут>: {$in: [<Список значений>]}.

Например:

<**db.список.find({"Цех":{$in:["Механический","Литейный"]}})**

{ "\_id" : ObjectId("586d583062ed48682494547e"), "Фамилия и инициалы" : "Котов А В", "Цех" : "Механический", "Табельный номер" : "01" }

{ "\_id" : ObjectId("586d597762ed486824945481"), "Фамилия и инициалы" : "Светов В В", "Цех" : "Литейный", "Табельный номер" : "03" }

6. $nin — проверка отсутствия определенных значений атрибута или самого атрибута (альтернатива для $in).

<Атрибут>: {$nin: [<Список значений>]} находит документы, в которых заданный атрибут не совпадает ни с одним из элементов в списке значений или документ не имеет проверяемого атрибута.

Например:

**> db.список.find({"Цех":{$in:["Механический","Столярный"]}})**

{ "\_id" : ObjectId("586d583062ed48682494547e"), "Фамилия и инициалы" : "Котов А В", "Цех" : "Механический", "Табельный номер" : "01" }

7. Выбор документов, одновременно удовлетворяющих нескольким условиям. В краткой форме отдельные условия задаются в общем списке:

<Атрибут1>:<Значение1>, <Атрибут2> : <Значение2>,... Заданные списком условия связываются по «И».

Например, запрос

**> db.список.find ({"Цех":"Сборочный","Табельный номер":"03"})**

{ "\_id" : ObjectId("586d591162ed486824945480"), "Фамилия и инициалы" : "Светин П С", "Цех" : "Сборочный", "Табельный номер" : "03" }.

Полная (в JSON) форма связывания по «И» использует модификатор $and с массивом условий: $and: [{<Условие1>}, …]

Например:

**> db.список.find ({$and: [{"Цех": "Сборочный"}, {"Табельный номер": "03"}]})**

{ "\_id" : ObjectId("586d591162ed486824945480"), "Фамилия и инициалы" : "Светин П С", "Цех" : "Сборочный", "Табельный номер" : "03" }

8. Связывание условий по «ИЛИ» $or: [<Список условий>].

Например,

**> db.список.find ({$or: [{"Цех":"Механический"}, {"Табельный номер":"05"}]})**

{ "\_id" : ObjectId("586d583062ed48682494547e"), "Фамилия и инициалы" : "Котов А В", "Цех" : "Механический", "Табельный номер" : "01" }

{ "\_id" : ObjectId("586d59bf62ed486824945482"), "Фамилия и инициалы" : "Ветров П В", "Цех" : "Сборочный", "Табельный номер" : "05" }

9. Связывание условий по «инвертированному ИЛИ»: $nor:

[<Список условий >] — выбирает документы, в которых не выполняются все условия или отсутствует атрибут, участвующий в условии.

Например, запрос:

**> db.список.find ({$nor: [{"Цех":"Механический"}, {"Табельный номер":"05"}]})**

дает результат

{ "\_id" : ObjectId("586d58bc62ed48682494547f"), "Фамилия и инициалы" : "Котаев П В", "Цех" : "Сборочный", "Табельный номер" : "02" }

{ "\_id" : ObjectId("586d591162ed486824945480"), "Фамилия и инициалы" : "Светин П С", "Цех" : "Сборочный", "Табельный номер" : "03" }

{ "\_id" : ObjectId("586d597762ed486824945481"), "Фамилия и инициалы" : "Светов В В", "Цех" : "Литейный", "Табельный номер" : "03" }

10. Поиск документов по наличию/отсутствию атрибута:

<Атрибут>: {$exists: false|true}.

По условию <Атрибут>: {$exists: false} находятся те документы, в которых проверяемый атрибут отсутствует или его значение не задано (null). Например, запрос

**> db.список.find({"Год\_рожд":{$exists: false}})**

выдает все документы, так как у них отсутствует атрибут **Год\_рожд**

{ "\_id" : ObjectId("586d583062ed48682494547e"), "Фамилия и инициалы" : "Котов А В", "Цех" : "Механический", "Табельный номер" : "01" }

{ "\_id" : ObjectId("586d58bc62ed48682494547f"), "Фамилия и инициалы" : "Котаев П В", "Цех" : "Сборочный", "Табельный номер" : "02" }

{ "\_id" : ObjectId("586d591162ed486824945480"), "Фамилия и инициалы" : "Светин П С", "Цех" : "Сборочный", "Табельный номер" : "03" }

{ "\_id" : ObjectId("586d597762ed486824945481"), "Фамилия и инициалы" : "Светов В В", "Цех" : "Литейный", "Табельный номер" : "03" }

{ "\_id" : ObjectId("586d59bf62ed486824945482"), "Фамилия и инициалы" : "Ветров П В", "Цех" : "Сборочный", "Табельный номер" : "05" }

db.authors.find ({"Год\_рожд":{$exists: false}}) находит авторов, у которых не задан атрибут "Год\_рожд".

11. Проверка типа для значения атрибута: {<атрибут>: {$type: <код BSON-типа>}}. Выбирает документы, у которых указанный атрибут имеет заданный тип.

Основные коды BSON-типов:

Double — 1 (числовой, задает целые числа и числа с фиксированной точкой), String — 2,

Boolean — 8,

Date — 9,

null — 10.

Например,

**> db.список.find ({"Цех": {$type: 2}})**

{ "\_id" : ObjectId("586d583062ed48682494547e"), "Фамилия и инициалы" : "Котов А В", "Цех" : "Механический", "Табельный номер" : "01" }

{ "\_id" : ObjectId("586d58bc62ed48682494547f"), "Фамилия и инициалы" : "Котаев П В", "Цех" : "Сборочный", "Табельный номер" : "02" }

{ "\_id" : ObjectId("586d591162ed486824945480"), "Фамилия и инициалы" : "Светин П С", "Цех" : "Сборочный", "Табельный номер" : "03" }

{ "\_id" : ObjectId("586d597762ed486824945481"), "Фамилия и инициалы" : "Светов В В", "Цех" : "Литейный", "Табельный номер" : "03" }

{ "\_id" : ObjectId("586d59bf62ed486824945482"), "Фамилия и инициалы" : "Ветров П В", "Цех" : "Сборочный", "Табельный номер" : "05" }

выберет все документы, так как у них атрибут "Год\_рожд" задан строкой текста.

**Управление выводом атрибутов в запросе**

Cписок выводимых атрибутов задается вторым параметром в методах find и findOne. В параметре атрибут, который необходимо вывести, указывается со значением 1, а не требуемый для вывода — 0.

db.<коллекция>.find ([{<селектор>} [, {<имя атрибута>: 1 |0, …}]]);

Например, запрос

**> db.список.find ({},{"Фамилия и инициалы": 1, "Цех":1})**

{ "\_id" : ObjectId("586d583062ed48682494547e"), "Фамилия и инициалы" : "Котов А В", "Цех" : "Механический" }

{ "\_id" : ObjectId("586d58bc62ed48682494547f"), "Фамилия и инициалы" : "Котаев П В", "Цех" : "Сборочный" }

{ "\_id" : ObjectId("586d591162ed486824945480"), "Фамилия и инициалы" : "Светин П С", "Цех" : "Сборочный" }

{ "\_id" : ObjectId("586d597762ed486824945481"), "Фамилия и инициалы" : "Светов В В", "Цех" : "Литейный" }

{ "\_id" : ObjectId("586d59bf62ed486824945482"), "Фамилия и инициалы" : "Ветров П В", "Цех" : "Сборочный" }

выведет из всех документов атрибуты **Фамилия и инициалы** , **Цех** и ключ \_id, который выводится по умолчанию, если его вывод не отключен явно.

Если в списке перечислены только атрибуты, не требующие вывода, то незаданные атрибуты выводятся по умолчанию.

Например, запрос

> db.список.find ({}, {\_id: 0, "Цех": 0})

{ "Фамилия и инициалы" : "Котов А В", "Табельный номер" : "01" }

{ "Фамилия и инициалы" : "Котаев П В", "Табельный номер" : "02" }

{ "Фамилия и инициалы" : "Светин П С", "Табельный номер" : "03" }

{ "Фамилия и инициалы" : "Светов В В", "Табельный номер" : "03" }

{ "Фамилия и инициалы" : "Ветров П В", "Табельный номер" : "05" }

выведет все атрибуты, кроме \_id и **Цех**, из всех документов.

В одном запросе не допускается смешивать выводимые и невыводимые атрибуты документа. Так, запрос db.authors. find ({}, {"Au\_lname": 1, "Au\_fname": 0}) является ошибочным. Исключение составляет атрибут \_id — идентификатор документов.

**Сортировка документов в запросе**

Для сортировки результата запроса find () используется присоединенный метод db.<Коллекция>.find ().sort ({<Атрибут>: 1 |-1,..}). Указание атрибута со значением 1 приводит к сортировке по возрастанию, —1 — по убыванию значений атрибута.

Например, запрос

**> db.список.find ({}, {\_id:0,"Фамилия и инициалы": 1, "Цех": 1}).sort ({"Фамилия и инициалы": 1})**

{ "Фамилия и инициалы" : "Ветров П В", "Цех" : "Сборочный" }

{ "Фамилия и инициалы" : "Котаев П В", "Цех" : "Сборочный" }

{ "Фамилия и инициалы" : "Котов А В", "Цех" : "Механический" }

{ "Фамилия и инициалы" : "Светин П С", "Цех" : "Сборочный" }

{ "Фамилия и инициалы" : "Светов В В", "Цех" : "Литейный" }

выведет фамилии , отсортированные по возрастанию фамилии.

**Ограничение множества выводимых документов**

Для ограничения количества выводимых документов используются присоединенные методы:

· limit (<Количество выводимых документов>),

· skip (<Количество пропускаемых в начале документов>).

Например, запрос

**> db.список.find (null, {\_id:0, "Фамилия и инициалы":1}).limit (3).skip (2)**

{ "Фамилия и инициалы" : "Светин П С" }

{ "Фамилия и инициалы" : "Светов В В" }

{ "Фамилия и инициалы" : "Ветров П В" }

выведет из коллекции пять первых фамилий , пропустив две первых фамилии.

**Задание для самостоятельной работы**

Создать базу данных с коллекцией, содержащей результаты реализации канцелярских товаров различными магазинами. Каждый документ должен содержать следующие атрибуты: название магазина, наименование товара, количество поступивших на реализацию единиц товара, количество проданных единиц товара, цена товара. Составить следующие запросы:

1 запрос содержащий результаты реализации указанного (по выбору студента) товара

2. запрос содержащий результаты реализации указанного(по выбору студента) товара по цене ниже средней (среднюю цену товара вычислить самостоятельно)

3. запрос содержащий результаты реализации указанного (по выбору студента) товара в указанных (по выбору студента) магазинах

4. запрос содержащий только название магазина, наименование товара и цену товара

5. запрос содержащий все атрибуты, в котором название магазинов расположены в алфавитном порядке

Коллекция должна содержать не менее пятнадцати документов. Название базы данных должно содержать фамилию студента.

**Вопросы для защиты работы**

1.Основной синтаксис вызова метода find ()

2. Виды условий в селекторе

3.Управление выводом атрибутов в запросе

4. Сортировка документов в запросе

5. Методы, используемые для ограничения количества выводимых документов

**Содержание отчета**

1.Номер и название лабораторной работы

2. Экранные формы, показывающие порядок выполнения лабораторной

работы, и результаты, полученные в ходе её выполнения.

3.Постановка задачи для самостоятельной работы

4. Экранные формы, показывающие порядок выполнения задания для

Самостоятельной работы с соответствующими пояснениями, и результаты, полученные в ходе её выполнения.

5.Ответы на вопросы для защиты