Очень краткий мануал по MongoDB
http://snakeproject.ru/rubric/article.php?art=mongodb_manual

Очень просто об устройстве бд MongoDB:
MongoDB - документо-ориентированная система
В базе данных хранятся коллекции, в коллекциях - документы
В документе есть пары: ключ - значение
Пример:

```
{
   "name": "Mike",
   "lastname": "Kozlov",
   "address": {
      "country" : "Russia",
      "city" : "Moscow",
      "index" : 111511
      }
}
```

Ключ "address" тут является ссылкой на другое значение

Типы значений могут быть разными, например:
String, Symbol, Boolean, Date, Timestamp, Integer, Double, Null, JavaScript и другие

У каждого документа автоматически создается уникальный идентификатор - **_id**

Простые команды:

Выбор существующей или создание новой базы данных:
use anyBase

Узнать текущую базу данных:
db

Удалить базу данных:
db.dropDatabase()

Создать коллекцию(вообще создается автоматически при добавлении документов, но можно создать явно):
db.createCollection("ps")

Удаление коллекции:
db.ps.drop()

Переименовать коллекцию:
db.ps.renameCollection("peoples")

Добавить один документ в коллекцию:
db.peoples.insertOne({"name": "Mike", "lastname": "Kozlov"})

Добавить несколько документов в коллекцию:
db.peoples.insertMany({"name": "Mike", "lastname": "Kozlov"},{"name": "Ivan", "lastname": "Kruglov"})

Добавить любое количество документов в коллекцию:
db.peoples.insert({"name": "Mike", "lastname": "Kozlov"})

Пример добавления с вложенными данными:
db.peoples.insert({"name": "Mike", "lastname": "Kozlov", address: {"country": "Russia", "city": "Moscow", "index": 111511}})

Посчитать количество документов в коллекции и при выборке:
db.peoples.count()
db.peoples.find({name: "Mike"}).count()

Выборка из коллекции:
db.peoples.find()

Выборка из коллекции уникальных значений поля "name":
db.peoples.distinct("name")

Выборка по одному и нескольким условиям:
db.peoples.find({name: "Mike"})
db.peoples.find({name: "Mike", lastname: "Kozlov"})

Выборка документов без какого-то свойства(полю присваиваем значение 0 либо true\false):
db.peoples.find({name: "Mike"}, {lastname: 0})
db.peoples.find({name: "Mike"}, {lastname: false})

Выборка всех документов без какого-то свойства:
db.peoples.find({}, {lastname: 0})

Выборка по вложенному элементу:
db.peoples.find({"address.country": "Russia"})

Выборка с ограничением по количеству выводимых документов:
db.peoples.find().limit(2)

Выборка с сортировкой по полю(по возрастанию - 1, по убыванию - -1):
db.peoples.find().sort({name: 1})

Выборка с несколькимифункциями:
db.peoples.find().sort({name: 1}).limit(2)

Выборка с помощью JavaScript:
func = function() { return this.name=="Mike"; }
db.peoples.find(func)

Выборка с группировкой:
key: {name : true} - поле, по которуму группируем (в данном случае - "name")
initial: {count : 0} - начальное значение результата
reduce : function (items, prev){prev.count += 1} - функция подсчета, прибавит результату группы 1 по нахождении группируемого элемента
db.peoples.group({key: {name : true}, initial: {count : 0}, reduce : function (items, prev){prev.count += 1}})

Выборка с условными операторами:
$eq - =
$gt - >
$lt - <
$gte - >=
$lte - <=
$ne - <>
Пример с двумя условиями больше и меньше указанных значений:
db.peoples.find({"address.index": {$gt : 111500, $lt : 111700}})

Изменение данных

Изменение документа:
Функция update принимает три параметра для обновления документа:
1 документ, который будет изменен

2 документ, на который будет изменен
3 Параметры обновления документов (upsert и multi)
3.1.1 upsert имеет значение true, то будет обновлен документ, если он найден, и создан новый, если документа нет
3.1.2 upsert имеет значение false, то не будет создан новый документ, если найденного документа нет
3.2.1 multi определяет, должен ли обновиться первый элемент в выборке (используется по умолчанию, если не указан)
3.2.2 multi определяет, должны обновляться все документы выборки

Пример (найденный документ - {name : "Mike"}, будет заменен документом {name: "Mike", lastname: "Degtyarev"}):
db.peoples.update({ name : "Mike" }, { name: "Mike", lastname: "Degtyarev" }, { upsert: true })

Изменение одного поля первого в выборке документа, укажем оператор $set, если докт не имеет изменяемого поля, создастся новое поле в докте:
db.peoples.update({name : "Mike", lastname: "Degtyarev"}, {$set: {lastname: "Kozlov"}})

В данном случае изменится первый в выборке документ, указав оператор multi:true, изменятся все документы из выборки:
db.peoples.update({name : "Mike", lastname: "Degtyarev"}, {$set: {lastname: "Kozlov"}}, {multi:true})

Удаление данных

Удаление поля, укажем оператор $unset:
db.peoples.update({name : "Mike"}, {$unset: {lastname: "Kozlov"}})
db.peoples.update({name : "Mike"}, {$unset: {lastname: "Kozlov", name: "Mike"}})

Удаление документа:
db.peoples.deleteOne({name : "Mike"})

Удаление множества документов:
db.peoples.remove({name : "Mike"})

Удаление всех элементов коллекции:
db.peoples.remove({})

Связи коллекций (связи полей коллекций):

Пример 1, назначение руками ключа _id:

Тестовые две таблицы (связь по полю _id таблицы company и поля company таблицы man):
db.company.insert({"_id" : "Bank1", year: 2000})
db.company.insert({"_id" : "Bank2", year: 2003})
db.man.insert({name: "Mike1", company: "Bank1"})
db.man.insert({name: "Mike2", company: "Bank2"})
db.man.insert({name: "Mike3", company: "Bank2"})

Теперь получим документы из коллекции company со связью с таблицей man:
m = db.man.find()
db.company.find({_id: m.company})

Пример 2, автоматический:

Создаем тестовые данные, метод save генерирует уникальный _id:
bank1=({"name": "bank1", "year": 2000})
bank2=({"name": "bank2", "year": 2003})
db.company.save(bank1)
db.company.save(bank2)

```
man1=({"name": "Mike1", "company": "Bank1", company: new DBRef('company', bank1._id)})
man2=({"name": "Mike2", "company": "Bank2", company: new DBRef('company', bank2._id)})
man3=({"name": "Mike3", "company": "Bank2", company: new DBRef('company', bank2._id)})
db.man.save(man1)
db.man.save(man2)
db.man.save(man3)
```

Выборка по ключу:

```
m = db.man.findOne({name: "Mike1"})
db.company.find({_id: m.company.$id})
```

Выборка курсором по ключу:

```
var pcur = db.man.find();
while(pcur.hasNext()){ obj = pcur.next(); print(obj["name"], " work in ", db.company.findOne({_id: obj.company.
$id}).name); }
```

Работа с курсорами

По аналогии с другими БД типа MS SQL Server\Oracle\Postgres в MongoDB так-же присутствуют курсоры
Метод "hasNext" проверяет, есть ли в выборке следующий документ
Метод "next" выбирает текущий документ и переводит курсор к следующему документу

Выборка курсором:

```
var pcur = db.peoples.find();
while(pcur.hasNext()){ obj = pcur.next(); print(obj["name"]); }
```

Выборка курсором с применением функций и условием:

```
var pcur = db.peoples.find({name: "Mike"});
pcur.limit(2).sort({"name":-1})
while(pcur.hasNext()){ obj = pcur.next(); print(obj["name"]); }
```

Индексы

Создание индекса по полю:

```
db.man.createIndex({"name" : 1})
```

Создание уникального индекса по полю:

```
db.man.createIndex({"name" : 1}, {"unique" : true})
```

Создание уникальных индексов по нескольким полям:

```
db.man.createIndex({"name" : 1, "company" : 1}, {"unique" : true})
```

Выборка всех индексов:

```
db.system.indexes.find()
```

Выборка индексов коллекции (поле name - имя индекса):

```
db.man.getIndexes()
```

Удаление индекса коллекции по его имени:

```
db.man.dropIndex("name_1_company_1")
```