Good afternoon ladies and gentlemen. My name is Anna Likhuta. I am a novice programmer. My aim for today’s presentation is to deliver you information about one of the database types. I’d like to describe in detail MongoDB.

Firstly I would like to tell about general representation, some advantages of this database , then we will consider installation.

Then I would like to take a look at so-called CRUD operations. They are: create, read, update and delete document in database.

Following that we should talk about main ways, which provide the efficient execution of queries and can return computed results.

All useful link you can find in presentation.

MongoDB is a document database designed for ease of development and scaling. A record in MongoDB is a document, which is a data structure composed of field and value pairs. The values of fields may include other documents, arrays, and arrays of documents.

The advantages of using documents are:

* Documents (i.e. objects) correspond to native data types in many programming languages.
* Embedded documents and arrays reduce need for expensive joins.
* Dynamic schema supports fluent polymorphism.

 High performance data persistence:

* Support for embedded data models reduces I/O activity on database system.
* Indexes support faster queries

Horizontal scalability as part of its core functionality.

Now, let’s to install MongoDB. It is available in two editions: Community and Enterprise.We will use Community, because the source is available and free to use from the official website.

# You can choose necessary version for your computer operating system. Also, MongoDB Atlas Free Tier Cluster is useful tool. It is a cloud-hosted service for provisioning, running monitoring, and maintaining MongoDB deployments. It is a fast, easy, and free way to get started with MongoDB. Free version allows to operate the 512 Mb of cloudy space.

# To execute commands or operations, which are listed below, you need to be connected to one of the following services:

# **MongoDB Atlas Free Tier Cluster**.

# **Local MongoDB installation**.

# I use the mongo shell to insert data and perform query operations in this presentation.

# MongoDB stores [BSON documents](https://docs.mongodb.com/manual/core/document/#bson-document-format), i.e. data records, in [collections](https://docs.mongodb.com/manual/reference/glossary/#term-collection); the collections in databases.

# What is BSON?

# BSON is a binary representation of [JSON](https://docs.mongodb.com/manual/reference/glossary/#term-json) documents Each BSON type has both integer and string identifiers. Some of them are: double, string, object, array, bool, date, null, regex, javascript and others.

# A few words about Objectid. Each document stored in a collection requires a unique [\_id](https://docs.mongodb.com/manual/reference/glossary/#term-id) field that acts as a [primary key](https://docs.mongodb.com/manual/reference/glossary/#term-primary-key). If an inserted document omits the \_id field, the MongoDB driver automatically generates an [ObjectId](https://docs.mongodb.com/manual/reference/bson-types/" \l "objectid) for the \_id field.

# In MongoDB, databases hold collections of documents. Collections are analogous to tables in relational databases.

MongoDB documents are composed of field-and-value pairs. The value of a field can be any of the BSON [data types](https://docs.mongodb.com/manual/reference/bson-types/), including other documents, arrays, and arrays of documents.

Create or insert operations add new [documents](https://docs.mongodb.com/manual/core/document/#bson-document-format) to a [collection](https://docs.mongodb.com/manual/core/databases-and-collections/#collections). If the database or collection does not currently exist, create or insert operations will create the database or collection.

# To select a database to use turn to command use.

[Documents](https://docs.mongodb.com/manual/core/document/) have the following restrictions on field names:

* The field name \_id is reserved for use as a primary key; its value must be unique in the collection.
* Field names **cannot** contain the null character.
* Top-level field names **cannot** start with the dollar sign ($) character.

The maximum BSON document size is 16 megabyte. If it is larger than the maximum size, MongoDB provides the GridFS API.

Let ‘s now turn to CRUD operation, which include the following actions with document: create, read, update and delete.

[**db.collection.insertOne()**](https://docs.mongodb.com/manual/reference/method/db.collection.insertOne/#db.collection.insertOne) inserts a *single* [document](https://docs.mongodb.com/manual/core/document/#bson-document-format) into a collection.

[**db.collection.insertMany()**](https://docs.mongodb.com/manual/reference/method/db.collection.insertMany/#db.collection.insertMany) can insert *multiple* [documents](https://docs.mongodb.com/manual/core/document/#bson-document-format) into a collection. Pass an array of documents to the method.

All write operations in MongoDB are atomic on the level of a single document.

# For query document you can use [db.collection.find()](https://docs.mongodb.com/manual/reference/method/db.collection.find/#db.collection.find) method.

To select all documents in the collection, pass an empty document as the query filter parameter to the find method.

A compound query can set conditions for more than one field in the collection’s documents. Implicitly, a logical AND connection joins the clauses of a compound query. The query selects the documents in the collection that matches all the conditions.

Using the [$or](https://docs.mongodb.com/manual/reference/operator/query/or/#op._S_or) operator, you can set a compound query that joins each clause with a logical OR connection so that the query selects the documents in the collection that matches at least one condition.

MongoDB supports regular expressions $regex queries to perform string pattern matches.

MongoDB provides the following methods for updating documents in a collection:

### [db.collection.updateOne()](https://docs.mongodb.com/manual/reference/method/db.collection.updateOne/" \l "db.collection.updateOne" \o "db.collection.updateOne()) method - update a Single Document.

### [db.collection.updateMany()](https://docs.mongodb.com/manual/reference/method/db.collection.updateMany/#db.collection.updateMany) method - update Multiple Documents

# To delete all documents from a collection, pass an empty [filter](https://docs.mongodb.com/manual/core/document/#document-query-filter) document {} to the [db.collection.deleteMany()](https://docs.mongodb.com/manual/reference/method/db.collection.deleteMany/#db.collection.deleteMany) method.

The method returns a document with the status of the operation.

## If you need to delete all documents that match a condition, you can set criteria, or filters, that identify the documents to delete.

To delete at most a single document that matches a specified filter (even though multiple documents may match the specified filter) use the **[db.collection.deleteOne()](https://docs.mongodb.com/manual/reference/method/db.collection.deleteOne/" \l "db.collection.deleteOne" \o "db.collection.deleteOne())**method**.**

The **[db.collection.bulkWrite()](https://docs.mongodb.com/manual/reference/method/db.collection.bulkWrite/" \l "db.collection.bulkWrite" \o "db.collection.bulkWrite())** method provides the ability to perform bulk insert, update, and remove operations. Bulk write operations affect a singlecollection.

[**bulkWrite()**](https://docs.mongodb.com/manual/reference/method/db.collection.bulkWrite/#db.collection.bulkWrite) supports the following write operations:

* [insertOne](https://docs.mongodb.com/manual/reference/method/db.collection.bulkWrite/#bulkwrite-write-operations-insertone) [updateOne](https://docs.mongodb.com/manual/reference/method/db.collection.bulkWrite/#bulkwrite-write-operations-updateonemany) [updateMany](https://docs.mongodb.com/manual/reference/method/db.collection.bulkWrite/#bulkwrite-write-operations-updateonemany) [replaceOne](https://docs.mongodb.com/manual/reference/method/db.collection.bulkWrite/#bulkwrite-write-operations-replaceone) [deleteOne](https://docs.mongodb.com/manual/reference/method/db.collection.bulkWrite/#bulkwrite-write-operations-deleteonemany) [deleteMany](https://docs.mongodb.com/manual/reference/method/db.collection.bulkWrite/#bulkwrite-write-operations-deleteonemany)

Let’s now proceed to consider basic operators.

The first part will be about query operators, which include logical operators, operators for comparison.

The second part is about update operators for fields.

Moving on to our next point aggregation.

Aggregation operations process data records and return computed results. Aggregation operations group values from complex documents together, and can perform a variety of operations on the grouped data to return a single result. MongoDB provides some ways to perform aggregation: the [aggregation pipeline](https://docs.mongodb.com/manual/aggregation/#aggregation-framework), the [map-reduce function](https://docs.mongodb.com/manual/aggregation/#aggregation-map-reduce).

In MongoDB, map-reduce operations use custom JavaScript functions to map, or associate, values to a key.

For most aggregation operations, the Aggregation Pipeline provides better performance and clearer interface. However, map-reduce operations provide some flexibility that is not presently available in the aggregation pipeline.

One more question, which remains to discuss is index.

Indexes support the efficient execution of queries in MongoDB. Without indexes, MongoDB must perform a collection scan, i.e. scan every document in a collection, to select those documents that match the query statement. If an appropriate index exists for a query, MongoDB can use the index to limit the number of documents it must inspect.

Indexes are special data structures that store a small portion of the collection’s data set in an easy to traverse form.

The [**db.collection.createIndex**](https://docs.mongodb.com/manual/reference/method/db.collection.createIndex/#db.collection.createIndex)**()** method only creates an index if an index of the same specification does not already exist.

As a summary I would like to say, that we considered main aspects of MongoDB, which is NOSQL database. You will be able to use MongoDB in your own project or application, if you need it.

Thank you for attention.