hochschule hof
University of Applied Sciences

Non-Relational Databases

MongoDB --- 1

Prof. Dr. Jürgen Heym

Hof University of Applied Sciences

1 – MongoDB – Introduction

hochschule hof
University of Applied Sciences

- Important questions

  Which database/datastore or which combination of databases/datastores may solve a given problem?

- Q 1: *What is the type of the datastore?*
- Q 2: *What was the motor for the development of a given datastore?*
- Q 3: *How do we talk to the datastore?*
- Q 4: *What makes a datastore unique?*
- Q 5: *What kind of solutions provides a datastore?*
- Q 6: *How scales a datastore?*

1 – MongoDB – Introduction

hochschule hof
University of Applied Sciences

- Q 1: *What is the type of the datastore?*

• relational	(PostgreSQL)
• Key / value	(Riak, Redis, Hazelcast)
• column-oriented	(Hbase)
• document-oriented	(MongoDB, CouchDB)
• graph-based	(Neo4J)

1 – MongoDB – Introduction

hochschule hof
University of Applied Sciences

- Q 6: *How scales a datastore?*

  - Scaling features are in tight connection to performance of the datastore!
  - Scaling features need a context!

  - Horizontal scaling features (MongoDB, Hbase, Riak)
  - Vertical scaling features (PostgreSQL, Neo4J, Redis, HazelCast)
  - Mixed scaling features

1 – MongoDB – Introduction

hochschule hof
University of Applied Sciences

- DBMS Types
  - Document oriented Datastore
    - Document == Hash
    - Documents may contain nested structures
    - Flexibility permits variable spaces within documents
    - Differences in indexing strategies, ad-hoc queries, replication, consistency, …
    - Examples: MongoDB, CouchDB, …

1 – MongoDB – Introduction

hochschule hof
University of Applied Sciences

- MongoDB --- Characteristics
  - Derived from „humongous“ == enormous, giant, vast
  - Programming language C++
  - Leading NRD-Datastore
  - Open-Source
  - Document oriented

- MongoDB --- Features
  - Document-Oriented Storage
    - JSON-style documents with dynamic schemas offer simplicity and power.
  - Full Index Support
    - Index on any attribute.
  - Replication & High Availablity
    - Mirror across LANs and WANs for scale and peace of mind.

1 – MongoDB – Introduction

hochschule hof
University of Applied Sciences

- MongoDB --- Features
  - Auto-Sharding (Auto-Splitting)
    - Scale horizontally without compromising functionality.
  - Querying
    - Rich, document-based queries.
  - Fast In-Place Updates
    - Atomic modifiers for contention-free performance.
  - Map / Reduce
    - Flexible aggregation and data processing.
  - GridFS
    - Store files of any size without complicating your stack.
  - MongoDB Management Service (MMS)
    - The best way to run MongoDB in production. Secured. Supported. Certified.
  - Production Support

2 – MongoDB – Use-Cases

hochschule hof
University of Applied Sciences

- Use-Case: catalog based applications
  - Catalogs change permanently. For RDBMSs this is a nightmare!
  - Example: Online-Shops

Catalog	MongoDB
Stuck. You need to add new data and metadata to your catalogs, including unstructured and semi-structured data. Relational databases can't easily handle this data, which leaves you hamstrung.	**Do the Impossible.** MongoDB can incorporate any product, entity, attribute or any other type of data, no matter what it looks like. And It does so while preserving all the functionality needed to build a powerful application, including advanced features like recommendations.
Stagnant. RDBMS technology leaves teams impotent to do what the business asks. Instead, they wrestle with data modeling, transformation and schema problems for months, or even years.	**Faster.** Your teams move faster with MongoDB because its dynamic schemas let them iterate. They spend less time figuring out how to accommodate new entities or attributes, and instead ship new products in weeks.
$$$$. Large teams tied up for long periods of time make RDBMS catalog applications expensive to build and maintain. Proprietary software and hardware add to the cost. The business case becomes hard for you to justify.	**$$.** More productive teams, plus commodity hardware make your projects cost 10% what they would with a relational database.

3 – MongoDB – Installation

hochschule hof
University of Applied Sciences

- Ubuntu (22.04 LTS)
  - Official MongoDB packages are more recent than Ubuntu packages!
  - Needed packages
    - Mongodb-org — Meta-Package, installs all other packages
    - mongodb-org-database — MongoDB server binaries, e.g. amd64
    - mongodb-org-server — MongoDB database server
    - mongodb-org-shell — MongoDB shell client
    - mongodb-org-tools — MongoDB tools
    - Mongodb-org-mongos — MongoDB shared cluster query router
    - … and some more tools packages ….
  - Configuration files and scripts (Upstart-Systematic)
    - /etc/mongodb.conf — configuration options
      https://www.mongodb.com/docs/manual/reference/configuration-options/
    - sudo systemctl [start|stop|status|restart|enable|disable] mongod

3 – MongoDB – Installation

hochschule hof
University of Applied Sciences

- MongoDB installation

In order to install MongoDB on Ubuntu 22.04 we follow the official MongoDB how-to:

https://www.mongodb.com/community/forums/t/how-to-install-mongodb-6-0-on-ubuntu-22-04/

4 – MongoDB – Definitions

hochschule hof
University of Applied Sciences

- ## Database

  Database is a physical container for collections. Each database gets its own set of files on the file system. A single MongoDB server typically has multiple databases.

- ## Collection

  Collection is a group of MongoDB documents. It is the equivalent of an RDBMS table. A collection exists within a single database. Collections do not enforce a schema. Documents within a collection can have different fields. Typically, all documents in a collection are of similar or related purpose.

- ## Document

  A document is a set of key-value pairs. Documents have dynamic schema. Dynamic schema means that documents in the same collection do not need to have the same set of fields or structure, and common fields in a collection's documents may hold different types of data.

4 – MongoDB – Definitions

hochschule hof
University of Applied Sciences

RDBMS	MongoDB
Database	Database
Table	Collection
Tuple/Row	Document
column	Field
Table Join	Embedded Documents
Primary Key	Primary Key (Default key _id provided by mongodb itself)
Database Server and Client	
mysqld/Oracle	mongod
mysql/sqlplus	mongo

4 – MongoDB – Sample Document

hochschule hof
University of Applied Sciences

Sample Blog document and ist structure: komma seperated key value pairs

```
{
   _id: ObjectId(7df78ad8902c)
   title: 'MongoDB Overview',
   description: 'MongoDB is no sql database',
   by: 'tutorials point',
   url: 'http://www.tutorialspoint.com',
   tags: ['mongodb', 'database', 'NoSQL'],
   likes: 100,
   comments: [
      {
         user:'user1',
         message: 'My first comment',
         dateCreated: new Date(2011,1,20,2,15),
         like: 0
      },
      {
         user:'user2',
         message: 'My second comments',
         dateCreated: new Date(2011,1,25,7,45),
         like: 5
      }
   ]
}
```

5 – MongoDB – Data-Modelling

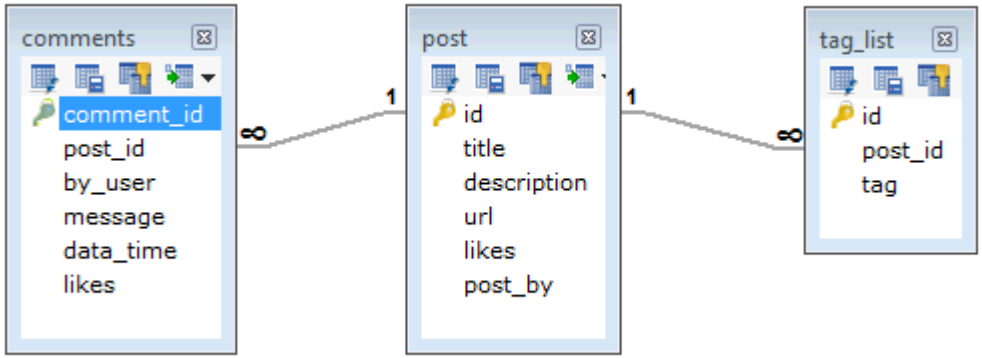
hochschule hof
University of Applied Sciences

- MongoDB Schemas
  - Data have within MongoDB a flexible schema!
  - Documents of a collection do not need the same set of fields or the same structure!
  - Same fields in documents of a collection may take different data types!
- MongoDB Design aspects
  - Develop the MongoDB design based on user and system requirements!
  - Combine object within a document, if you want to use them together. Otherwise separate them! Make sure that you don‘t need joins!
  - If necessary duplicate data in a limited way. RAM is cheap compared to CPU time!
  - Consider joins at write operations, not at read operations!
  - Optimize your schema concerning you key application!
  - Consider complex aggregations already within your schema!

5 – MongoDB – Data-Modelling

hochschule hof
University of Applied Sciences

- Example (RDBMS vs. MongoDB)
  - Your client needs a datastore for his blog website.
  - Requirements are:
    - Each blog entry gets a unique title, a description and an URL.
    - Each blog entry may be classified in one or more categories (tags).
    - Each blog entry has an author and a likes counter.
    - Each blog entry may have comments or not that are stored together with their author, timestamp and likes counter.
  - RDBMS schema



3 – MongoDB – Data-Modelling

hochschule hof
University of Applied Sciences

- Example (RDBMS vs. MongoDB)
  - MongoDB schema holds only one collection!

```
{
        _id: POST_ID
        title: TITLE_OF_POST,
        description: POST_DESCRIPTION,
        by: POST_BY,
        url: URL_OF_POST,
        tags: [TAG1, TAG2, TAG3],
        likes: TOTAL_LIKES,
        comments: [
           {
              user:'COMMENT_BY',
              message: TEXT,
              dateCreated: DATE_TIME,
              like: LIKES

           },
           {
              user:'COMMENT_BY',
              message: TEXT,
              dateCreated: DATE_TIME,
              like: LIKES

           }
        ]
}
```

5 – MongoDB – DDL / DML

hochschule hof
University of Applied Sciences

- DDL == Data Definition Language
  - Databases
    - Create database `use DATABASE_NAME`
    - Current database `db`
    - List databases `show dbs`
    - Databases are created as soon as you insert the first document.
    - Delete a database `db.dropDatabase()`

5 – MongoDB – DDL / DML

hochschule hof
University of Applied Sciences

- DDL == Data Definition Language

  - Collections
    - Create a collection `db.createCollection(NAME, OPTIONS)`
    - Example:
```
db.createCollection("mycol",
    { capped : true, size : 6142800, max : 10000 })
```

    - In mongodb you don't need to create collection.
      MongoDB creates collections automatically, when you insert the first document.

    - Drop a collection `db.COLLECTION_NAME.drop()`
    - The drop() method will return true, if the selected collection is dropped successfully otherwise it will return false.

Parameter	Type	Description
Name	String	Name of the collection to be created
Options	Document	(Optional) Specify options about memory size and indexing

5 – MongoDB – DDL / DML

hochschule hof
University of Applied Sciences

- DDL == Data Definition Language
  - Collection Options

Field	Type	Description
capped	Boolean	(Optional) If true, enables a capped collection. A capped collection is a size limited collection that automatically overwrites its oldest entries when it reaches its maximum size. **If you specify true, you need to specify the size parameter also.**
autoIndexID	Boolean	(Optional) If true, automatically create index on _id fields. Default is false.
size	number	(Optional) Specifies a maximum size in bytes for a capped collection.
max	number	(Optional) Specifies the maximum number of documents allowed in the capped collection.

5 – MongoDB – DDL / DML

hochschule hof
University of Applied Sciences

- DDL == Data Definition Language

  - Indexes
    - Indexes support the efficient resolution of queries. Without indexes, MongoDB must scan every document of a collection to select those documents that match the query statement. This scan is highly inefficient and require the mongod to process eventually a large volume of data.
    - Indexes are special data structures, that store a small portion of the data set in an easy to traverse form. The index stores the value of a specific field or set of fields, ordered by the value of the field as specified in index.
    - To create an index you need to use ensureIndex() method of mongodb.
    - Ascending index: 1, descending index: -1
    - Syntax

      ```
      db.col.ensureIndex({KEY:1})
      ```

    - Example

      ```
      db.col.ensureIndex({"title":1,"description":-1})
      ```

5 – MongoDB – DDL / DML

hochschule hof
University of Applied Sciences

- DDL == Data Definition Language

  - Index Options (1)

Parameter	Type	Description
background	boolean	Builds the index in the background so that building an index does not block other database activities. Specify true to build in the background. The default value is **false**.
unique	boolean	Creates a unique index so that the collection will not accept insertion of documents where the index key or keys match an existing value in the index. Specify true to create a unique index. The default value is **false**.
name	string	The name of the index. If unspecified, MongoDB generates an index name by concatenating the names of the indexed fields and the sort order.
dropDups	boolean	Creates a unique index on a field that may have duplicates. MongoDB indexes only the first occurrence of a key and removes all documents from the collection that contain subsequent occurrences of that key. Specify true to create unique index. The default value is **false**.
sparse	boolean	If true, the index only references documents with the specified field. These indexes use less space but behave differently in some situations (particularly sorts). The default value is **false**.

5 – MongoDB – DDL / DML

hochschule hof
University of Applied Sciences

- DDL == Data Definition Language

  - Index Options (2)

Parameter	Type	Description
expireAfterSeconds	integer	Specifies a value, in seconds, as a TTL to control how long MongoDB retains documents in this collection.
v	index version	The index version number. The default index version depends on the version of mongod running when creating the index.
weights	document	The weight is a number ranging from 1 to 99,999 and denotes the significance of the field relative to the other indexed fields in terms of the score.
default_language	string	For a text index, the language that determines the list of stop words and the rules for the stemmer and tokenizer. The default value is **english**.
language_override	string	For a text index, specify the name of the field in the document that contains, the language to override the default language. The default value is language.

5 – MongoDB – DDL / DML

hochschule hof
University of Applied Sciences

- MongoDB Data Types

Datatype	Description
String	This is most commonly used data type to store the data. String in mongodb must be UTF-8 valid.
Integer	This type is used to store a numerical value. Integer can be 32 bit or 64 bit depending upon your server.
Boolean	This type is used to store a boolean (true/ false) value.
Double	This type is used to store floating point values.
Min / Max keys	This type is used to compare a value against the lowest and highest BSON elements.
Arrays	This type is used to store arrays or list or multiple values into one key.
Timestamp	Ctimestamp. This can be handy for recording when a document has been modified or added.
Object	This data type is used for embedded documents.
Null	This type is used to store a Null value.

5 – MongoDB – DDL / DML

hochschule hof
University of Applied Sciences

- MongoDB Data Types

Datatype	Description
Date	This data type is used to store the current date or time in UNIX time format. You can specify your own date time by creating an object of Date and passing day, month, year into it.
Object ID	This data type is used to store the document's ID.
Binary data	This data type is used to store binary data.
Code	This data type is used to store JavaScript code into a document.
Regular expression	This data type is used to store regular expressions.

5 – MongoDB – DDL / DML

hochschule hof
University of Applied Sciences

- DML == Data Manipulation Language

  - Insert Data
    - To insert data into a MongoDB collection, you need MongoDB's **insert()** or **save()** method:

```
db.COLLECTION_NAME.insertOne(document)
db.COLLECTION_NAME.insertMany([doc1,doc2, ...])
db.COLLECTION_NAME.replaceOne(filter,document)
```

    - If the collection doesn't exist in the database, MongoDB will create this collection and then insert the document the collection.
    - If we don't specify the _id parameter, then MongoDB assigns an unique ObjectId for this document.
    - The ObjectId (_id) is a 12 bytes hexadecimal number, unique for every document in a collection: 4 bytes timestamp, 3 bytes machine id, 2 bytes process id, 3 bytes incrementer.
    - If you specify the _id-Parameter and you use the method **replaceOne()**, then the document will be replaced.

5 – MongoDB – DDL / DML

hochschule hof
University of Applied Sciences

- DML == Data Manipulation Language
  - Replace a document
    - Example

```
db.books.replaceOne(
{
        _id: ObjectId(7df78ad8902c)
},
{
        _id: ObjectId(7df78ad8902c),
        title: 'MongoDB Overview',
        description: 'MongoDB is no sql database',
        tags: ['mongodb', 'database', 'NoSQL'],
        likes: 100
})
```

5 – MongoDB – DDL / DML

hochschule hof
University of Applied Sciences

- DML == Data Manipulation Language
  - Insert multiple documents
    - Example

```
db.books.insertMany([
{
        title: 'MongoDB Overview',
        description: 'MongoDB is no sql database',
        tags: ['mongodb', 'database', 'NoSQL'],
        likes: 100
},{
        title: 'NoSQL Database',
        description: 'NoSQL databases don't have tables',
        tags: ['mongodb', 'database', 'NoSQL'],
        likes: 20
}
])
```

5 – MongoDB – DDL / DML

hochschule hof
University of Applied Sciences

- Querying the database

  - Basic syntax of method **find()**

    ```
    db.COLLECTION_NAME.find()
    ```

  - Basic syntax for formatting the output with method pretty(). This changes the output only in mongo not in mongosh!

    ```
    db.COLLECTION_NAME.find().pretty()
    ```

5 – MongoDB – DDL / DML

hochschule hof
University of Applied Sciences

- Querying the database
  - Comparison Operators

Operation	Syntax	Example	RDBMS Equivalent
Equality	{<key>:<value>}	`db.col.find({"by":"tutorials"})`	where by = 'tutorials'
Less Than	{<key>:{$lt:<value>}}	`db.col.find({"likes":{$lt:50}})`	where likes < 50
Less Than Equals	{<key>:{$lte:<value>}}	`db.col.find({"likes":{$lte:50}})`	where likes <= 50
Greater Than	{<key>:{$gt:<value>}}	`db.col.find({"likes":{$gt:50}})`	where likes > 50
Greater Than Equals	{<key>:{$gte:<value>}}	`db.col.find({"likes":{$gte:50}})`	where likes >= 50
Not Equals	{<key>:{$ne:<value>}}	`db.col.find({"likes":{$ne:50}})`	where likes != 50

5 – MongoDB – DDL / DML

hochschule hof
University of Applied Sciences

- Querying the database

  - Logical AND
    - If you pass in the **find()** method multiple keys by separating them by ',' then MongoDB treats it as an **AND** condition.
    - Example

```
db.col.find({key1:value1, key2:value2})
```

  - Logical OR
    - Syntax:

```
db.col.find( { $or: [ {key1: value1}, {key2:value2} ] } )
```

5 – MongoDB – DDL / DML

hochschule hof
University of Applied Sciences

- Exercise

1. Create a MongoDB database <<your Username>>_library
2. Create within your database a collection books.
3. Insert three documents in your collection.
4. Use the find() method to search your documents.
5. Develop different use cases that use all the comparison operators and logical operators.

5 – MongoDB – DDL / DML

hochschule hof
University of Applied Sciences

- Updating Documents

  - MongoDB's **findOneAndUpdate()** method isused to update documents in a collection.

    Syntax:

    ```
    db.col.findOneAndUpdate(filter,update,options)
    ```

5 – MongoDB – DDL / DML

hochschule hof
University of Applied Sciences

- Updating Documents

  - Example

```
db.col.find()
{ "_id" : ObjectId(5983548781331adf45ec5), "title":"MongoDB Overview"}
{ "_id" : ObjectId(5983548781331adf45ec6), "title":"NoSQL Overview"}
{ "_id" : ObjectId(5983548781331adf45ec7), "title":"Tutorials Point Overview"}

db.books.findOneAndUpdate({_id:
ObjectId("5983548781331adf45ec5")},{$set:{likes:5}});

db.col.find()
{ "_id" : ObjectId(5983548781331adf45ec5),
  "likes":5, "title":"MongoDB Overview"}
{ "_id" : ObjectId(5983548781331adf45ec6), "title":"NoSQL Overview"}
{ "_id" : ObjectId(5983548781331adf45ec7), "title":"Tutorials Point Overview"}
```

5 – MongoDB – DDL / DML

hochschule hof
University of Applied Sciences

- Replacing Documents

  - The **findOneAndReplace()** method replaces the existing document with the new document passed in the method.

    Basic syntax of mongodb **findOneAndReplace()** method is shown below:

    - Syntax:

      ```
      db.col.findOneAndReplace(filter,replacement,options)
      ```

5 – MongoDB – DDL / DML

hochschule hof
University of Applied Sciences

- Deleting Documents

  - MongoDB's **deleteOne()** and **deleteMany()** methods are used to delete documents from a collection. Both methods accept the same parameters. One is the deletion criteria and the second is the optional write-concern.

    - Syntax:

      `db.col.deleteMany(filter,options)` (delete multiple docs)

      or

      `db.col.deleteOne(filter,options)` (delete one doc)

5 – MongoDB – DDL / DML

hochschule hof
University of Applied Sciences

- Querying the database (continued)

  - Projections

    Basic Syntax: show key1, key2, … but don't show the ObjectId.

    ```
    db.col.find({SELECTION_CRITERIA},{KEY1:1,...,_id:0})
    ```

    Example

    ```
    {"_id" : ObjectId(5983548781331adf45ec5), "title":"MongoDB Overview"}
    {"_id" : ObjectId(5983548781331adf45ec6), "title":"NoSQL Overview"}
    {"_id" : ObjectId(5983548781331adf45ec7), "title":"Tutorials Point Overview"}

    db.col.find({},{"title":1,_id:0})

    {"title":"New MongoDB Tutorial"}
    {"title":"NoSQL Overview"}
    {"title":"Tutorials Point Point Overview "}
    ```

5 – MongoDB – DDL / DML

hochschule hof
University of Applied Sciences

- Querying the database (continued)
  - Limitations

    Use the limit() method to limit the number of shown documents.

    ```
    db.col.find().limit(NUMBER)
    ```

  - Skipping documents

    Use the skip() method to skip a certain number of documents in the result set.

    ```
    db.col.find().limit(NUMBER).skip(NUMBER2)
    ```

5 – MongoDB – DDL / DML

hochschule hof
University of Applied Sciences

- Querying the database (continued)

  - Sorting the result set

    To sort documents in MongoDB, you need to use **sort()** method.

    The **sort()** method accepts a document containing a list of fields along with their sorting order. To specify sorting order 1 and -1 are used. 1 is used for ascending order while -1 is used for descending order.

```
db.col.find({},{"title":1,_id:0}).sort({"title":-1})
```

5 – MongoDB – DDL / DML

hochschule hof
University of Applied Sciences

- Exercise (part 2)

1. Change individual attributes in different documents.
2. Change one attribute in multiple documents at once.
3. Replace a document with a new one.
4. Verify the outcome of all different methods for deleting a document.
5. Practice the projection of different attributes.
6. Limit the output using method limit().
7. Verfy the influence of method skip().
8. Change the sort sequence of search queries.

Literature

hochschule hof
University of Applied Sciences

- Seven NoSQL Datbases in a Week
  Sudarshan Kadambi; Xun Wu; Aaron Ploetz; Devram Kandhare, Oreilly®
  ISBN 978-1-78728-886-7
- MongoDB Inc.
  https://www.mongodb.com/
- MongoDB ORG
  https://www.mongodb.org/
- MongoDB Tutorial
  http://www.tutorialspoint.com/mongodb/index.htm
- MongoDB Configuration options
  http://docs.mongodb.org/manual/reference/configuration-options/

hochschule hof
University of Applied Sciences

Non-Relational Databases

MongoDB --- 2

Prof. Dr. Jürgen Heym

Hof University of Applied Sciences

6 – MongoDB – Telephone Number Search

hochschule hof
University of Applied Sciences

- Let‘s generate telephone numbers!

```
function populatePhonebook(areaCode,quantity) {
  for(var i=0; i < quantity; i++) {

  // set phonenumber length (3 to 9 digits)
  var phoneNumberLength = 3+((Math.random() * 6) << 0);

  // generate Phonenumber and length adjustment
  var phoneNumber = 1+((Math.random() * Math.pow(10,phoneNumberLength)) << 0);
  phoneNumberLength = 1+Math.floor(Math.log(phoneNumber)/Math.log(10));
  if (phoneNumberLength<3) { continue; }


  // Make phonebook entry and save
  var num = areaCode * Math.pow(10,phoneNumberLength) + phoneNumber;

  db.phones.insert({
      _id: num,
      components: {
         areaCode: areaCode,
         phoneNumber : phoneNumber
      },
      display: areaCode + "/" + phoneNumber
    });
  } // End for-Loop
}
```

6 – MongoDB – Telephone Number Search

hochschule hof
University of Applied Sciences

- Basic Telephone Number Search
  - Search conditions
    - Search for 9281/455
    - Sort order: ascending area code and telephone number

```
db.phones.find(
    {display:"09281/455"}
)
```

6 – MongoDB – Telephone Number Search

hochschule hof
University of Applied Sciences

- General Telephone Number Search
  - Search conditions
    - Telephone number: 455
    - Area code: any
    - Sort order: ascending area code

```
db.phones.find(
   {"components.phoneNumber":455},{display:1,_id:0}
).sort({"components.areaCode":1})
```

6 – MongoDB – Telephone Number Search

hochschule hof
University of Applied Sciences

- Search telephone numbers of a limited digit range
  - Search conditions
    - All telephone numbers having exactly 3 digits
    - Sort order: ascending telephone number

```
var number_range = {}
number_range['$gte']=100
number_range['$lt']=1000

db.phones.find(
   {"components.phoneNumber":number_range},{display:1,_id:0}
).sort({"components.phoneNumber":1})
```

6 – MongoDB – Telephone Number Search

hochschule hof
University of Applied Sciences

- Search telephone numbers containing a given number sequence
  - Search conditions
    - Known sequence: 070
    - Sort order: ascending telephone number

```
db.phones.find(
   {display:/070/},{display:1,_id:0}
).sort({"components.phoneNumber":1})
```

6 – MongoDB – Telephone Number Search

hochschule hof
University of Applied Sciences

- Search telephone numbers ending with a given number sequence
  - Search conditions
    - Phone number last digits: 070
    - Sort order: ascending telephone number

```
db.phones.find(
    {display:/070$/},{display:1,_id:0}
).sort({"components.phoneNumber":1})
```

6 – MongoDB – Telephone Number Search

hochschule hof
University of Applied Sciences

- Search telephone numbers containing several given number sequences
  - Search conditions
    - Known digits: 070?5
    - Sort order: descending telephone number

```
db.phones.find(
   {display:/070.5/},{display:1,_id:0}
).sort({"components.phoneNumber":-1})
```

6 – MongoDB – Telephone Number Search

hochschule hof
University of Applied Sciences

- Search telephone numbers starting with a given number sequence
  - Search conditions
    - Phone number first digits: 201
    - Sort order: ascending telephone number

```
db.phones.find(
   {display:/\/201/},{display:1,_id:0}
).sort({"components.phoneNumber":1})
```

6 – MongoDB – Telephone Number Search

hochschule hof
University of Applied Sciences

- Search telephone numbers starting with a given number sequence and limited digits at first position
  - Search conditions
    - Startsequences: 200, 400 or 600
    - Sort order: ascending telephone number

```
db.phones.find(
   {display:/\/[246]00/},{display:1,_id:0}
).sort({"components.phoneNumber":1})
```

6 – MongoDB – Telephone Number Search

hochschule hof
University of Applied Sciences

- Search telephone numbers with know starting and ending digits
  - Search conditions
    - Start sequence: 20
    - End sequence: 13
    - Any number of digits between start sequence and end sequence
    - Sort order: ascending telephone number

```
db.phones.find(
   {display:/\/20(.*)13$/},{display:1,_id:0}
).sort({"components.phoneNumber":1})
```

6 – MongoDB – Telephone Number Search

hochschule hof
University of Applied Sciences

- Search Performance

  - Which indexes has a collection?
    - At creation time MongoDB creates automatically an index on objectIds (_id).
    - Get all indexes of a collection

      ```
      db.collection.getIndexes()
      ```

  - How can we accelerate?
    - A B-Tree-Index on field display would perhaps accelerate our search.
    - Use method explain() to get information about search behaviour.

      ```
      db.phones.find({display:"9281/455"})
                        .explain({verbose:"executionStats"})
      ```

6 – MongoDB – Telephone Number Search

hochschule hof
University of Applied Sciences

- Search Performance

  - Example output of method explain()

```
db.phones.find({display:/\/455/})
        .explain("executionStats")
{
...
"executionStats" : {
                "executionSuccess" : true,
                "nReturned" : 2,
                "executionTimeMillis" : 42,
                "totalKeysExamined" : 0,
                "totalDocsExamined" : 18480,
...
}
```

6 – MongoDB – Telephone Number Search

hochschule hof
University of Applied Sciences

- Search Performance

  - Create an index on field „display“

```
db.phones.createIndex(
    {display:1},
    {unique:true}
)
```

  - Verify the search performance.
  - You should see that the newly created index is used!

6 – MongoDB – Telephone Number Search

hochschule hof
University of Applied Sciences

- Profiling Level

  - Profiling Level 1 stores only slow queries.
  - Profiling Level 2 stores all queries.
  - Profiling datastore: db.system.profile

  - Practical work:
    - Set profilingLevel to the value of 2.
    - Execute the query again.
    - Analyze the profiling data.

```
db.setProfilingLevel(2)
db.phones.find({display:/\/455/})
db.system.profile.find()
```

6 – MongoDB – Telephone Number Search

hochschule hof
University of Applied Sciences

- Search Performance

  - Create an index on „areaCode“ in the „background“.

```
db.phones.createIndex(
    {"components.areaCode":1},
    {background:1}
)
```

  - Get all indexes.

```
db.phones.getIndexes()
```

Literature

hochschule hof
University of Applied Sciences

- Sieben Wochen, sieben Datenbanken
  Moderne Datenbanken und die NoSQL-Bwegung
  E. Redmon & J. R. Wilson, Oreilly®
  ISBN 978-3-86899-791-0
- MongoDB Inc.
  https://www.mongodb.com/
- MongoDB ORG
  https://www.mongodb.org/
- MongoDB Tutorial
  http://www.tutorialspoint.com/mongodb/index.htm
- MongoDB Konfigurationsoptionen
  http://docs.mongodb.org/manual/reference/configuration-options/

hochschule hof
University of Applied Sciences

Non-Relational Databases

MongoDB --- 3

Prof. Dr. Jürgen Heym

Hof University of Applied Sciences

7 – MongoDB – DML – Update

hochschule hof
University of Applied Sciences

- Exercise (part 1)

1. Create a new database „TelecomContracts_ddmmyy“, where „ddmmyy“ is your birthday.
2. Create a collection „customers“ that will store address information about our customers.
3. Create a JavaScript function that allows for creating new customers comfortably: insertCustomer(cid,lastname,firstname,street,postcode,city).
4. Save function insertCustomer in file insertCustomer.js and load the file.
5. Insert three customer documents in your collection:

```
{
   "cid"       : "221-0159",
   "lastname"  : "Iakimenkowa",
   "firstname" : "Irina",
   "street"    : "352, Birdland Road",
   "postcode"  : "95028",
   "city"      : "Paradise City"
}
```

7 – MongoDB – DML – Update

hochschule hof
University of Applied Sciences

- Updating Documents (Mongo version 3)

  - MongoDB's version 3 **update()** method is used to update documents in a collection.

    The update() method updates values in existing documents.

    - Syntax:

      ```
      db.col.update(SELECTION_CRITERIA, UPDATED_DATA)
      ```

    - By default mongodb will update only a single document, to update multiple documents you need to set a parameter 'multi' to true.

      Syntax:

      ```
      db.col.update(SELECTION_CRITERIA, UPDATED_DATA,{multi:true})
      ```

7 – MongoDB – DML – Update

hochschule hof
University of Applied Sciences

- Example (Mongo version 3)

```
db.col.find()
{ "_id" : ObjectId(5983548781331adf45ec5), "title":"Overview"}
{ "_id" : ObjectId(5983548781331adf45ec6), "title":"NoSQL Overview"}

db.col.update({'title':'Overview'},{$set:{'title':'New MongoDB Tutorial'}})

db.col.find()
{ "_id" : ObjectId(5983548781331adf45ec5), "title":"New MongoDB Tutorial"}
{ "_id" : ObjectId(5983548781331adf45ec6), "title":"NoSQL Overview"}
```

7 – MongoDB – DML – Update

hochschule hof
University of Applied Sciences

- Updating Documents (Mongo version 6)

  - MongoDB's version 6 **updateOne()** method is used to update exactly one document in a collection. To update several documents use **updateMany()**.

    The update() method updates values in existing documents.

    - Syntax:

      ```
      db.col.updateOne(SELECTION_CRITERIA, UPDATED_DATA)
      ```

    - By default mongodb will update only a single document, to update multiple documents you need to set a parameter 'multi' to true.

      Syntax:

      ```
      db.col.updateMany(SELECTION_CRITERIA, UPDATED_DATA)
      ```

7 – MongoDB – DML – Update

hochschule hof
University of Applied Sciences

- Example (Mongo version 6)

```
db.inventory.updateOne(
   { cid:"221-0159" },
   {
     $set: { address2: "Upper Floor" },
     $currentDate: { lastModified: true }
   }
)
```

This update operation:

- uses the $set operator to update the value of the size.uom field to "cm" and the value of the status field to "P", and
- uses the $currentDate operator to update the value of the lastModified field to the current date. If lastModified field does not exist, $currentDate will create the field. See $currentDate for details.

7 – MongoDB – DML – Update

hochschule hof
University of Applied Sciences

- Example (Mongo version 6)

```
db.inventory.updateMany(
   { "qty": { $lt: 50 } },
   {
     $set: { "size.uom": "in", status: "P" },
     $currentDate: { lastModified: true }
   }
)
```

This update operation:

- Finds all documents with key “qty” lower than 50,
- uses the $set operator to update the value of the size.uom field to "in" and the value of the status field to "P", and
- uses the $currentDate operator to update the value of the lastModified field to the current date. If lastModified field does not exist, $currentDate will create the field. See $currentDate for details.

7 – MongoDB – DML – Update

hochschule hof
University of Applied Sciences

- Exercise (part 2)

1. Read the page about update of MongoDB v6 reference manual:
   `https://www.mongodb.com/docs/manual/tutorial/update-documents/`
2. Change the lastname attribute in one of your address documents.
   Use the „_id“ attribute as the selection criterium.
3. Control the result using the find() method. The output should show only the following attributes: cid, lastname, firstname.
4. One of your addresses should have a second address attribute (address2) with an apartment information „Apartment 5-22“. Verify the updated document.
5. Delete attribute address of one document and add an attribute „postbox“ with value 1602 to this document. Verify the updated document.

7 – MongoDB – DML – Update

hochschule hof
University of Applied Sciences

- Solution (part2 for Mongo version 3)

2. `db.contracts.update({"_id":ObjectId(59...c5)},{$set:{lastname:"Fox"}})`

3. `db.contracts.find({"_id":ObjectId(59...c5)},`
   `{_id:0,cid:1,lastname:1,firstname:1})`
4. `db.contracts.update({"_id":ObjectId(59...c5)},`
   `{$set:{address2:"Appartment 5-22"}})`
   `db.contracts.find()`

5. `db.contracts.update({"_id":ObjectId(59...)},{$unset:{address:1}})`
   `db.contracts.find()`
   `db.contracts.update({"_id":ObjectId(59...)},{$set:{postbox:"1602"}})`
   `db.contracts.find()`

7 – MongoDB – DML – Update

hochschule hof
University of Applied Sciences

- Exercise (part 3)

Each customer may have zero, one or several contracts with different tariff and options.

For each tariff we create a new contract that may be canceled separately.

Contracts will be numbered sequentially and get a leading „C“, e.g. „C-001“.

Valid tariffs with their options are

Phone [Flat-DE, Flat-EU]

TV [HD-PLUS]

Internet [100, 64, 16, WLAN]

1. Use the update method to store a new contract for a customer with tariff phone, option FLAT-EU, and contract date 12/15/2016.
2. The same customer now signs a second contract with tariff Internet including several options Internet 100 & WLAN and today’s date.
   Use directive $push !

7 – MongoDB – DML – Update

hochschule hof
University of Applied Sciences

- Solution for Mongo version 3

1.
```
db.contracts.update(
{"_id":ObjectId(59...c5)},
  {$set:{ contracts:[
            {cnr:"C-001",tariffs:["Phone"],options:["Flat-DE"]}
          ]
        }
  }
)

db.contracts.find().pretty()
```

2.
```
db.contracts.update(
{"_id":ObjectId(59...c5)},
  {$push:{ contracts:
             {cnr:"C-002",tariffs:["Internet"],options:["100","WLAN"]}
         }
  }
)

db.contracts.find()
```

7 – MongoDB – DML – Update

hochschule hof
University of Applied Sciences

- Challenge (part 4)

3. The third customer now signs two contracts including several tariffs Phone, TV and Internet with options Internet 100 & WLAN and today's date. Use directive $push in connection with $each!

   Read the MongoDB reference pages for method update and the update operators:
   https://www.mongodb.com/docs/manual/tutorial/update-documents/
   https://www.mongodb.com/docs/manual/reference/operator/update/

4. Search and show all customers that have a contract with WLAN-Option.

5. For marketing purposes search and show all customers that have a contract with tariff Internet but without option WLAN.

Literature

hochschule hof
University of Applied Sciences

- Sieben Wochen, sieben Datenbanken
  Moderne Datenbanken und die NoSQL-Bwegung
  E. Redmon & J. R. Wilson, Oreilly®
  ISBN 978-3-86899-791-0
- MongoDB Update-Methode
  http://docs.mongodb.org/manual/reference/method/db.collection.update/
- MongoDB Update-Operatoren und –Modifikatoren
  http://docs.mongodb.org/manual/reference/operator/update/

hochschule hof
University of Applied Sciences

Non-Relational Databases

MongoDB --- 4

Prof. Dr. Jürgen Heym

Hof University of Applied Sciences

8 – MongoDB Users & Roles

hochschule hof
University of Applied Sciences

1. Let‘s create a site user administrator „siteUserAdmin“ and activate user authentication.

   a) Start MongoDB without access control and create user „siteUserAdmin“ with role „userAdminAnyDatabase“ in database admin.

```
use admin

db.createUser(
  {
    user: "siteUserAdmin",
    pwd:  "nosql",
    roles: [ { role: "userAdminAnyDatabase", db: "admin" } ]
  }
)
```

   b) Edit the MongoDB configuration file /etc/mongod.conf and activate authentication setting option **`security. authorization: enabled`**.
   Restart mongod: **`service mongod restart`**

   See: https://docs.mongodb.com/manual/tutorial/enable-authentication/

8 – MongoDB Users & Roles

hochschule hof
University of Applied Sciences

2. Create a user administrator for database „myDB“.

   a) Enter MongoDB and authenticate user „siteUserAdmin“ on db „admin“.

```
mongo –u siteUserAdmin –p nosql admin
```

   b) Create user „myUserAdmin“ with role „userAdmin“ in database „myDB“

```
use myDB
db.createUser(
  {
    user: "myUserAdmin", pwd:  "nosql", roles: [ "userAdmin" ]
  }
)
```

   c) Verification

```
use admin
db.system.users.find({user:"myUserAdmin"})
```

8 – MongoDB Users & Roles

hochschule hof
University of Applied Sciences

3. Create a new user for database „myDB“

   a) Enter MongoDB and authenticate user „myUserAdmin“ on db „myDB“.

```
mongo –u myUserAdmin –p nosql myDB
```

   b) Create user „myUser“ with role „readWrite“ in database „myDB“.

```
use myDB
db.createUser( {
   user: "myUser", pwd:  "nosql", roles: [ "readWrite" ]
 } )
```

   c) Verify as user „siteUserAdmin“ and / or „myUserAdmin“.

```
use admin
db.system.users.find({user:"myUser"})
```

8 – MongoDB Users & Roles

hochschule hof
University of Applied Sciences

4. Test user „myUser“

   a) Enter MongoDB and authenticate user „myUser“ on db „myDB“.

```
mongo –u myUser –p nosql myDB
```

   b) Save a document in collection „myCol“ in database „myDB“ and verify the results.

```
use myDB
db.myCol.insert({firstname:"myFirstName",lastname:"myLastName"})
```

   c) Verification

```
db.myCol.find()
```

8 – MongoDB Users & Roles

hochschule hof
University of Applied Sciences

5. Create a user with role on two databases.

   a) Enter MongoDB and authenticate user „siteUserAdmin“ on db „admin“.

```
mongo –u siteUserAdmin –p nosql admin
```

   b) Create user „myUser2“ with role „read“ in database „myDB“ and „myDB2“.

```
use myDB
db.createUser( {
   user: "myUser2", pwd: "nosql",
   roles: [ { role: "read", db: "myDB" },
            { role: "read", db: "myDB2" } ] } )
```

   c) Verify as user „siteUserAdmin“!

   d) Verify read-write in database „myDB“ as user „myUser2“ .

8 – MongoDB Users & Roles

hochschule hof
University of Applied Sciences

- Roles
  - A role grants privileges to perform the specified actions on resource. Each privilege is either specified explicitly in the role or inherited from another role or both.
- Privileges
  - A privilege consists of a specified resource and the actions permitted on the resource.
  - A resource is a database, collection, set of collections, or the cluster.
  - An action specifies the operation allowed on the resource.
- Inherited Privileges
  - A role can include one or more existing roles in its definition, in which case the role inherits all the privileges of the included roles.

8 – MongoDB Users & Roles

hochschule hof
University of Applied Sciences

- Users and Roles
  - You can assign roles to users during the user creation and update.
  - A user assigned a role receives all the privileges of that role.
  - A user can have one or multiple roles in different databases.

- Built-in Roles

  - ***read***
    Provides the ability to read data on all non-system collections and on the following system collections: system.indexes, system.js, and system.namespaces collections.
  - ***readWrite***
    Provides all the privileges of the read role and the ability to modify data on all non-system collections and the system.js collection.

8 – MongoDB Users & Roles

hochschule hof
University of Applied Sciences

- Built-in Roles (continued)
  - ***dbAdmin***
    Provides the ability to perform administrative tasks such as schema-related tasks, indexing, gathering statistics. This role does not grant privileges for user and role management.
  - ***dbOwner***
    Provides the ability to perform any administrative action on the database. This role combines the privileges granted by the *readWrite*, *dbAdmin* and *userAdmin* roles.
  - ***userAdmin***
    Provides the ability to create and modify roles and users on the current database. Since the userAdmin role allows users to grant any privilege to any user, including themselves, the role also indirectly provides *superuser* access to either the database or, if scoped to the admin database, the cluster.

8 – MongoDB Users & Roles

hochschule hof
University of Applied Sciences

- Built-in Roles (continued)
  - ***backup***
    Provides privileges needed to back up data. This role provides sufficient privileges to use the MongoDB Cloud Manager backup agent, Ops Manager backup agent, or to use mongodump.
  - ***restore***
    Provides privileges needed to restore data with mongorestore without the --oplogReplay option or without system.profile collection data.

8 – MongoDB Users & Roles

hochschule hof
University of Applied Sciences

- All-Database Roles
  - **These roles in the admin database apply to all but the local and config databases in a mongod instance!**
  - ***readAnyDatabase***
    Provides the same *read-only* permissions as read, except it applies for all databases.
  - ***readWriteAnyDatabase***
    Provides the same read and write permissions as *readWrite*, except …
  - ***userAdminAnyDatabase***
    Provides the same access to user administration operations as *userAdmin*, except ...
  - ***dbAdminAnyDatabase***
    Provides the same access to database administration operations as *dbAdmin*, except …

- Superuser
  - **root**
    - Provides access to the operations and all the resources of the readWriteAnyDatabase, dbAdminAnyDatabase, userAdminAnyDatabase, clusterAdmin, restore, and backup combined.

8 – MongoDB Users & Roles

hochschule hof
University of Applied Sciences

- Show users roles / access rights

1. Enter MongoDB and authenticate user „siteUserAdmin“ on db „admin“.

```
mongo –u siteUserAdmin –p nosql admin
```

2. Show user info for user „myUser“ in database „myDB“:

```
use myDB
db.getUser("myUser")
```

8 – MongoDB Users & Roles

hochschule hof
University of Applied Sciences

- Show a roles privileges

1. Enter MongoDB and authenticate user „siteUserAdmin“ on db „admin“.

```
mongo –u siteUserAdmin –p nosql admin
```

2. Show role info for role „read“ in database „myDB“:

```
use myDB
db.getRole("read",{showPrivileges:true})
```

All granted and inherited privileges are shown!

8 – MongoDB Users & Roles

hochschule hof
University of Applied Sciences

- Grant roles

1. Enter MongoDB and authenticate user „siteUserAdmin“ on db „admin“.

```
mongo –u siteUserAdmin –p nosql admin
```

2. Grant role „readWrite“ to user „myUser“ on database „myDB“:

```
use myDB
db.grantRolesToUser(
    myUser,
    [
          {role:"readWrite", db:"myDB"}
    ]
)
```

8 – MongoDB Users & Roles

hochschule hof
University of Applied Sciences

- Revoke roles

1. Enter MongoDB and authenticate user „siteUserAdmin“ on db „admin“.

```
mongo –u siteUserAdmin –p nosql admin
```

2. Revoke role „readWrite“ from user „myUser“ on database „myDB“:

```
use myDB
db.revokeRolesFromUser(
    myUser,
    [
          {role:"readWrite", db:"myDB"}
    ]
)
```

8 – MongoDB Users & Roles

hochschule hof
University of Applied Sciences

- Change user password

1. Enter MongoDB and authenticate user „siteUserAdmin“ on db „admin“.

```
mongo –u siteUserAdmin –p nosql admin
```

2. Change password for user „myUser“ in database „myDB“:

```
use myDB
db.changeUserPassword("myUser","newPasswordString")
```

To change your own password you have to proceed with a more complex procedure!

See: *http://docs.mongodb.org/manual/tutorial/change-own-password-and-custom-data/*

Literature

- MongoDB Access Control 1
  https://docs.mongodb.org/manual/administration/configuration/
- MongoDB Access Control 2
  http://docs.mongodb.org/manual/tutorial/enable-authentication-without-bypass/
- MongoDB Tutorial about Users and Roles
  http://docs.mongodb.org/manual/tutorial/add-user-administrator/
- MongoDB Change Your Own Password
  http://docs.mongodb.org/manual/tutorial/change-own-password-and-custom-data/

hochschule hof
University of Applied Sciences

Non-Relational Databases

MongoDB --- 5

Prof. Dr. Jürgen Heym

Hof University of Applied Sciences

9 – MongoDB Backup & Restore

hochschule hof
University of Applied Sciences

- General aspects of mongodump
  - mongodump creates backups of MongoDBs (binary export)
  - mongodump is an essential part of every MongoDB backup strategy
  - mongorestore restores a MongoDB from MongoDB backups
  - mongodump uses mongod or mongos
  - mongodump can backup MongoDBs directly from the data files
    - Even without having any active mongod running!
  - caution: mongodump overwrites existing dumps!

9 – MongoDB Backup & Restore

hochschule hof
University of Applied Sciences

- Backup of system collections
  - Following the „least privilege principle“ you need role backup to be able to dump all databases.
  - To dump a special database you need at least read access to this database. Role backup has this access right.
  - If one wants to dump collection system.profile of any of your database, one needs the roles clusterAdmin and dbAdmin.
  - If one wants to dump user specific roles, one needs access to database admin, there all users and roles are stored.
  - If one wants to dump all users of a special database, one has to search these users in collection admin.system.users. Roles backup and userAdminAnyDatabase are allowed to do so.
  - If one wants to dump all role definitions of a special database, one has to search these roles in collection admin.system.roles. Roles backup and userAdminAnyDatabase are allowed to do so.

9 – MongoDB Backup & Restore

hochschule hof
University of Applied Sciences

- mongodump
  - syntax

```
mongodump [options]
```

  - important optionen

```
--help
--verbose                 or      -v
--quiet
--host <hostname><:port>  or      -h <hostname><:port>
--port <port>             default value: 27017
--username <username>     or      -u <username>
--password <password>     or      -p <password>
--authenticationDatabase <db>   authenticate against <db>
--db <database>           or      -d <database>
--collection <collection>         or      -c <collection>
--out <path>              or      -o <path>   default: dump
--query <json>            or      -q <json>
--repair
--dumpDbUsersAndRoles
```

9 – MongoDB Backup & Restore

hochschule hof
University of Applied Sciences

- mongodump
  - examples
    1. Create a backup user in database “admin” with role “backup” with password “nosql”.

```
use admin
db.createUser({user:"backupUser",pwd:"nosql",roles:[{role:"backup",db:"admin"}]})
```

    2. Backup of collection mycol in MongoDB mydb:

```
mongodump –u backupUser –p nosql
          --authenticationDatabase admin
          --collection mycol --db mydb --out yourDumpSubDir
```

    3. Example. Hint: there are no shortcuts for –port and –authenticationDatabase.

```
mongodump -h mongodb1.example.net --port 37017
          -u backupUser -p nosql
          --authenticationDatabase admin
          -c mycol -d mydb
          -o /opt/backup/mongodump-2014-12-16
```

9 – MongoDB Backup & Restore

hochschule hof
University of Applied Sciences

- General aspects of mongorestore
  - mongorestore can restore MongoDBs dumps made with mongodump
  - mongorestore is an essential part of every MongoDB backup strategy
  - mongorestore uses mongod or mongos
  - mongorestore can restore MongoDBs directly from the data files
    - Even without having any active mongod running!
  - mongorestore may replace or add dumps to existing MongoDBs
  - caution:
    - mongorestore overwrites existing indices and generates new indices!
    - all mongorestore operations are inserts, no updates.

9 – MongoDB Backup & Restore

hochschule hof
University of Applied Sciences

- mongorestore
  - syntax

```
mongorestore [options] [<<pathToDump>>]
```

  - important options

```
--help
--verbose                  or      -v
--quiet
--host <hostname><:port>or      -h <hostname><:port>
--port <port>              default value: 27017
--username <username>      or      -u <username>
--password <password>      or      -p <password>
--authenticationDatabase <db>   authenticate against <db>
--db <database>            or      -d <database>
--collection <collection>          or      -c <collection>
--filter <json>
--drop
--noIndexRestore
```

9 – MongoDB Backup & Restore

hochschule hof
University of Applied Sciences

- mongorestore
  - examples

1. Restore collection mycol to targetDB from dump in dump/oldDBdump:

```
mongorestore –u backupUser –p nosql
             --authenticationDatabase admin
             --collection mycol --db mydb dump/oldDBdump
```

2. Restore all MongoDBs from dump on host mongodb1.example.net under port 37017 with authentification

```
mongorestore --host mongodb1.example.net --port 37017
             --username backupUser --password nosql
              --authenticationDatabase admin
             /opt/backup/mongodump-2014-12-16
```

Literature

- MongoDB Mongodump
  http://docs.mongodb.org/manual/reference/program/mongodump/
- MongoDB Mongorestore
  http://docs.mongodb.org/manual/reference/program/mongorestore/

hochschule hof
University of Applied Sciences

Non-Relational Databases

MongoDB --- 6

Prof. Dr. Jürgen Heym

Hof University of Applied Sciences

10 – MongoDB Data Import

hochschule hof
University of Applied Sciences

- GeoLite2 Free Databases

  - **Databases**

    GeoLite2 databases are free IP geolocation databases comparable to, but less accurate than, MaxMind's GeoIP2 databases. The GeoLite2 Country and City databases are updated on the first Tuesday of each month. The GeoLite2 ASN database is updated every Tuesday.

  - **IP Geolocation Usage**

    IP geolocation is inherently imprecise. Locations are often near the center of the population. Any location provided by a GeoIP database should not be used to identify a particular address or household.

    Use the Accuracy Radius as an indication of geolocation accuracy for the latitude and longitude coordinates we return for an IP address. The actual location of the IP address is likely within the area defined by this radius and the latitude and longitude coordinates.

  - **License**

    The GeoLite2 databases are distributed under the Creative Commons Attribution-ShareAlike 4.0 International License. The attribution requirement may be met by including the following in all advertising and documentation mentioning features of or use of this database.

10 – MongoDB Data Import

hochschule hof
University of Applied Sciences

- GeoLite2 Files
  - **GeoLite2-Country-Locations-en.csv**
    - Columns

```
geoname_id,
locale_code,
continent_code,
continent_name,
country_iso_code,
country_name
```

    - Example lines

```
1861060,en,AS,Asia,JP,Japan
1873107,en,AS,Asia,KP,"North Korea"
1880251,en,AS,Asia,SG,Singapore
1899402,en,OC,Oceania,CK,"Cook Islands"
```

10 – MongoDB Data Import

hochschule hof
University of Applied Sciences

- GeoLite2 Files
  - **GeoLite2-Country-Blocks-IPv4.csv**
    - Columns

```
network,
geoname_id,
registered_country_geoname_id,
represented_country_geoname_id,
is_anonymous_proxy,
is_satellite_provider
```

    - Example lines

```
1.0.16.0/20,1861060,1861060,,0,0          1861060 == Japan, see prev page
1.0.32.0/19,1814991,1814991,,0,0
1.0.64.0/18,1861060,1861060,,0,0
1.0.128.0/17,1605651,1605651,,0,0
```

10 – MongoDB Data Import

hochschule hof
University of Applied Sciences

- MongoDB Challenge
  - Data Import CSV2MongoDB (20 points)

1. Develop a JSON document structure for storing Country-IPv4-Information in a MongoDB fulfilling the following conditions (8 points).
   a) For each country only one document is accepted!
   b) For each country all information from GeoLite2-Country-Locations-en.csv must appear in the country document.
   c) For each country all IPv4 networks must appear in the country document.

2. Write a PHP-Skript (loadCountryIPv4.php) that loads the complete Country-IPv4-Structure into MongoDB CountryIPv4_ddmmyy (12 points).

Literature

- GeoLite 2 Free Downloadable Databases
  https://dev.maxmind.com/geoip/geoip2/geolite2/
- MongoDB Mongorestore
  http://docs.mongodb.org/manual/reference/program/mongorestore/