Advanced Management of Data
Exercise Topic 4:
NoSQL

SQL vs NoSQL

Structured Query Language

- originally based upon relational algebra and tuple relational calculus
- consists of different “sublanguages”:
  - data query language (DQL)
  - data definition language (DDL)
  - data control language (DCL)
  - data manipulation language (DML)
- complex standard that is not fully implemented by most DBMS

Non SQL or **Not Only SQL**

- simplicity by design
- finer control over availability
- faster and more flexible
- eventual consistency
- different data models:
  - Key-value
  - Column-oriented
  - Document-oriented
  - Graph
  - Relational

MongoDB

- one of the most widespread NoSQL DBMS
- free, open-source and cross-platform
- document-oriented (uses JSON-like documents with schemas)
- main features:
  - Ad hoc queries
  - Indexing
  - Replication
  - Load balancing
  - File storage
  - Aggregation
  - Server-side JavaScript execution
  - Capped collections

Get a MongoDB database and user interface

- you need a MongoDB database to do the practical part of this exercise
- the easiest way is to install one locally at your own computer
- follow the instructions to install and start the server according to your operating system
  ➔https://docs.mongodb.com/manual/administration/install-community/
- we are going to use the classic command line utility “MongoDB Shell”
  - there is the nice graphical user interface “MongoDB Compass Community” (you are likely to install it with the server)
  - but you might learn more from typing commands than by clicking some buttons, that do all the work for you
- you could also use an online service like MongoDB Atlas
  ➔https://cloud.mongodb.com/links/registerForAtlas
  + you don’t need to install and run the database server on your computer and you can use it from anywhere
  - you have to register there and more configuration is needed than by using it locally
  - you have to download the MongoDB server package anyway, as the MongoDB Shell is included with the server

Connect to your MongoDB database

- start your MongoDB server “mongod”
- start the “MongoDB Shell” user interface “mongo” to connect to your database server (it does automatically if you installed locally on your computer with default values)
- you should see a window with a command prompt “>”
- Task: create a new database `AMD` and a collection `exercise`

Create a new database

- databases and collections are created at first write access, so you can just use the new database

```
use AMD
```

- don't try to escape the name, as `"AMD"` won't work and `'AMD'` will create another database with the apostrophe in the name, as this is just a shell helper for

```
db = db.getSiblingDB("AMD")
```

- as no data is written, no database is created until now, which can be tested by

```
show dbs
show databases
db.adminCommand("listDatabases")
```

Create a new collection

```
db.createCollection("exercise")
```

- by creating the collection, also the database is created
- collections can be shown by

```
show collections
db.getCollectionNames()
```

- Task: in our exercises, `topic 1` was the `Unified Modeling Language` and it took us `3 weeks`, so add this information to the collection

Insert first exercise

```
db.exercise.insert(
  {
    _id: 1,
    topic: "Unified Modeling Language",
    weeks: 3
  }
)
```

- Task: add the information from the other exercises to the collection: `topic 2` was `Extensions of SQL` for `7 weeks`, `topic 3` was `Object-Relational Database Systems` for `3 weeks` and `topic 4` is `NoSQL` for `1 week`

Insert other exercises

```
db.exercise.insert(
  [
    {
      _id: 2, topic: "Extensions of SQL", weeks: 7
    },
    {
      _id: 3, topic: "Object-Relational Database Systems", weeks: 3
    },
    {
      _id: 4, topic: "NoSQL", weeks: 1
    }
  ]
)
```

- Task: get all exercises, which took us exactly 3 weeks

Find exercises

```
db.exercise.find(
  {
    weeks: 3
  }
)
```

- Task: get all exercises, which took us at least 3 weeks

Find more exercises

```
db.exercise.find(
  { weeks: { $gte: 3 } }
)
```

- for more comparators see https://docs.mongodb.com/manual/reference/operator/query-comparison/
- our last query could have been written as

```
db.exercise.find(
  { weeks: { $eq: 3 } }
)
```

- Task: get all exercises, which took us at less than 3 weeks, but return only the `topic`

Get a projection on exercises

```
db.exercise.find(
  { weeks: { $lt: 3 } },
  { _id: false, topic: 1 }
)
```

- the field `_id` is always returned, unless you specify otherwise
- a projection cannot contain include (`true` or `1`) and exclude (`false` or `0`) specifications, with the one exception of the exclusion of `_id`
- for more information see https://docs.mongodb.com/manual/reference/method/db.collection.find/
- Task: add the used programming languages: topics 2 and 3 used `PL/pgSQL` and topic 4 uses `JavaScript`

Add more information to exercises

```
db.exercise.update(
  { _id: { $in: [2, 3] } },
  { $set: { language: "PL/pgSQL" } },
  { multi: true }
)

db.exercise.update(
  { _id: 4 },
  { $set: { language: "JavaScript" } }
)
```

- just declaring new `field:value` would replace the document (only `_id` stays), but by using `$set` attributes can be added
- unless `multi` is set to `true`, only the first match is updated
- for more information see https://docs.mongodb.com/manual/reference/method/db.collection.update/
- Task: get all exercises and sort them ascending by `topic`

Sort exercises

```
db.exercise.find().sort(
  { topic: 1 }
)
```

- there can be used multiple sorting criteria with mixed ascending and descending sort order
- for more information see https://docs.mongodb.com/manual/reference/method/cursor.sort/
- Task: delete all exercises, where we didn't use `JavaScript`

Delete exercises

```
db.exercise.remove(
  { language: { $ne: "JavaScript" } }
)
```

- as you might have noticed, also the first exercise got deleted, although it didn't have the field language, but the logic matches anything that doesn't contain the field with the required value
- Task: as almost nothing is left in our collection, remove `exercise` itself

Delete exercises

```
db.exercise.drop()
```

- as the collection is removed and it was the last collection in our database, the database is removed, too

What's next?

- you just finished a (very basic) MongoDB programming tutorial concerning the CRUD operations
- now it's time to write something more complex to practice and improve these skills
- a good entry point how to access your MongoDB from your preferred programming language would be the MongoDB online manual

➔https://docs.mongodb.com/manual/