# GCD Assignments week 7

## Assignment 1: ZIPS.JSON

1. Download zips.json
2. Open a command window and run

mongoimport --db scratch --collection zips --file zips.json

Sanders-MacBook-Pro-3:MongoDB sandervanlaar$ mongoimport --db scratch --collection zips --file zips.json

connected to: 127.0.0.1

2015-01-26T15:17:53.925+0100 check 9 29353

2015-01-26T15:17:54.005+0100 imported 29353 objects

1. Inspect the data
   1. View the database
   2. View the collection
   3. View the data
2. Indexes
   1. Run find query on the data (without index):

db.zips.find().count()

db.zips.find({"city":"FLAGSTAFF"}).count()

db.zips.find({"city":"FLAGSTAFF"}).explain()

* 1. Run find query on the data (with index):

db.zips.ensureIndex({"city" : 1})

db.zips.find().count()

db.zips.find({"city":"FLAGSTAFF"}).count()

db.zips.find({"city":"FLAGSTAFF"}).explain()

* 1. Explain the difference.

1. Queries
   1. How many records does the zips collection consist of?
   2. How many zipcodes are there in the state of Massachusetts?
   3. Give all state names from the zips collection.
   4. Sort the state names alphabetically.
   5. How many states are there in the zips collection?
   6. How many cities have population of less than 50?
   7. Which cities have population of less than 50?

Tip: use <http://docs.mongodb.org/manual/core/read-operations-introduction/>

And <http://docs.mongodb.org/manual/tutorial/query-documents/>

1. GeoNear index
   1. Create a 2d index on location:

db.zips.ensureIndex({"loc": "2d"})

* 1. Find locations within 5000 meters from Flagstaff

Tip: <http://docs.mongodb.org/manual/reference/operator/query/near/>

Put the statements and (parts of) the output in your portfolio.

## Assigment 2: Mini Project

1. Design a MongoDB schema, choose one of the cases below
2. Develop and create the database
3. Insert at least 2 sample documents and subdocuments
4. Give an overview of the data (tip: use the .pretty( ) method)
5. Give a query and the result of that query on the sample data (see MongoDB documentations on arrays)

Tip: to implement 1:N relations, use arrays, see documentation.

Put the statements to create the database and the queries in your portfolio.

**Cases**

Case 1: [Customer orders]

Customer --- 1:N --- Order --- 1:N --- Product

Attributes: Customer (custnr, name), Order (date), Product (nr, name, quantity)

Case 2: [Enrollment]

Student --- N:1 --- Class --- N:N --- Course

Attributes: Student (nr, name), Class (code, semester), Course (name, credits)

Case 3: [Blog site]

User --- 1:N --- Post --- 1:N --- Comment --- 1:1 --- Commenter

Attributes: User (name), Post (date, text), Comment (date, text), Commenter (name)

Case 4: [Facebook]

User --- 1:N --- Status --- 1:N --- Comment --- 1:1 --- Commenter

Attributes: User (name), Status (date, text), Comment (date, text), Commenter (name)

Case 5: [TV series]

TVseries --- 1:N --- Season --- 1:N --- Episode --- 1:N --- Review

Attributes: TVseries (title), Season (seasonNr), Episode (episodeNr, title), Review (date, text, nrOfStars)