COMP306 – DATABASE MANAGEMENT SYSTEMS

My Name-Surname: Barış KAPLAN
My KU ID Number: 0069054
My KU Email Address: bkaplan18@ku.edu.tr
Term: Spring 2022
Homework Number: HW #5

MY ANSWERS:

1-)

```
(base) barissss-mbp:Desktop barissss$ mongoimport --db hw5 --collection zipcodes /Users/barissss/Desktop/zips.json --type json
2022-05-21T23:40:28.874+0300    connected to: mongodb://localhost/
2022-05-21T23:40:30.060+0300    29353 document(s) imported successfully. 0 document(s) failed to import.
(base) barissss-mbp:Desktop barissss$
```

The command I used for importing the data inside the zips.json file as the collection called zipcodes to the database called as hw5

```
> use hw5
switched to db hw5
> db
hw5
```

Using the database called hw5 (by using use hw5 command)

```
> show collections
zipcodes
>
```

Showing the collections under the hw5 database (by using 'show collections' command under the hw5 database)

```
> db.zipcodes.find({}).limit(15)
{ "_id" : "01007", "city" : "BELCHERTOWN", "loc" : [ -72.410953, 42.275103 ], "pop" : 10579, "state" : "MA" }
{ "_id" : "01011", "city" : "CHESTER", "loc" : [ -72.988761, 42.279421 ], "pop" : 1688, "state" : "MA" }
{ "_id" : "01002", "city" : "CUSHMAN", "loc" : [ -72.51565, 42.377017 ], "pop" : 36963, "state" : "MA" }
{ "_id" : "01010", "city" : "BRIMFIELD", "loc" : [ -72.188455, 42.116543 ], "pop" : 3706, "state" : "MA" }
{ "_id" : "01001", "city" : "AGAWAM", "loc" : [ -72.622739, 42.070206 ], "pop" : 15338, "state" : "MA" }
{ "_id" : "01005", "city" : "BARRE", "loc" : [ -72.108354, 42.409698 ], "pop" : 4546, "state" : "MA" }
{ "_id" : "01008", "city" : "BLANDFORD", "loc" : [ -72.936114, 42.182949 ], "pop" : 1240, "state" : "MA" }
{ "_id" : "01013", "city" : "CHICOPEE", "loc" : [ -72.607962, 42.162046 ], "pop" : 23396, "state" : "MA" }
{ "_id" : "01020", "city" : "CHICOPEE", "loc" : [ -72.576142, 42.176443 ], "pop" : 31495, "state" : "MA" }
{ "_id" : "01022", "city" : "WESTOVER AFB", "loc" : [ -72.558657, 42.196672 ], "pop" : 1764, "state" : "MA" }
{ "_id" : "01026", "city" : "CUMMINGTON", "loc" : [ -72.905767, 42.435296 ], "pop" : 1484, "state" : "MA" }
{ "_id" : "01028", "city" : "EAST LONGMEADOW", "loc" : [ -72.505565, 42.067203 ], "pop" : 13367, "state" : "MA" }
{ "_id" : "01031", "city" : "GILBERTVILLE", "loc" : [ -72.198585, 42.332194 ], "pop" : 2385, "state" : "MA" }
{ "_id" : "01012", "city" : "CHESTERFIELD", "loc" : [ -72.833309, 42.38167 ], "pop" : 177, "state" : "MA" }
{ "_id" : "01030", "city" : "FEEDING HILLS", "loc" : [ -72.675077, 42.07182 ], "pop" : 11985, "state" : "MA" }
>
```

Showing the first 15 documents in the zipcodes collection

For switching to the database called hw5: I used ‘use hw5’ command.

For importing json file called “zips.json” to the database called hw5: I used ‘mongoimport - -db hw5 - - collection zipcodes /Users/barissss/Desktop/zips.json - - json

For showing the first 15 documents inside the zipcodes collection: I used ‘db.zipcodes.find({}).limit(15)’ command.

2-)

```
> db.zipcodes.aggregate( [
...     { $match: { pop: { $gte: 30000 }, state:"CA" }}
... ] )
{ "_id" : "90001", "city" : "LOS ANGELES", "loc" : [ -118.247896, 33.973093 ], "pop" : 51841, "state" : "CA" }
{ "_id" : "90002", "city" : "LOS ANGELES", "loc" : [ -118.246213, 33.94969 ], "pop" : 40629, "state" : "CA" }
{ "_id" : "90003", "city" : "LOS ANGELES", "loc" : [ -118.272739, 33.965335 ], "pop" : 53938, "state" : "CA" }
{ "_id" : "90004", "city" : "LOS ANGELES", "loc" : [ -118.302863, 34.076163 ], "pop" : 64062, "state" : "CA" }
{ "_id" : "90005", "city" : "LOS ANGELES", "loc" : [ -118.301197, 34.058508 ], "pop" : 35864, "state" : "CA" }
{ "_id" : "90006", "city" : "LOS ANGELES", "loc" : [ -118.291687, 34.049323 ], "pop" : 63389, "state" : "CA" }
{ "_id" : "90007", "city" : "LOS ANGELES", "loc" : [ -118.287095, 34.029442 ], "pop" : 46985, "state" : "CA" }
{ "_id" : "90008", "city" : "LOS ANGELES", "loc" : [ -118.341123, 34.011643 ], "pop" : 33073, "state" : "CA" }
{ "_id" : "90018", "city" : "LOS ANGELES", "loc" : [ -118.315173, 34.028972 ], "pop" : 48267, "state" : "CA" }
{ "_id" : "90011", "city" : "LOS ANGELES", "loc" : [ -118.258189, 34.007856 ], "pop" : 96074, "state" : "CA" }
{ "_id" : "90016", "city" : "LOS ANGELES", "loc" : [ -118.352787, 34.029826 ], "pop" : 43669, "state" : "CA" }
{ "_id" : "90019", "city" : "LOS ANGELES", "loc" : [ -118.33426, 34.048158 ], "pop" : 64996, "state" : "CA" }
{ "_id" : "90028", "city" : "LOS ANGELES", "loc" : [ -118.302211, 34.066535 ], "pop" : 34926, "state" : "CA" }
{ "_id" : "90022", "city" : "EAST LOS ANGELES", "loc" : [ -118.155319, 34.023638 ], "pop" : 65865, "state" : "CA" }
{ "_id" : "90023", "city" : "LOS ANGELES", "loc" : [ -118.197498, 34.024478 ], "pop" : 47136, "state" : "CA" }
{ "_id" : "90024", "city" : "LOS ANGELES", "loc" : [ -118.440796, 34.063691 ], "pop" : 38370, "state" : "CA" }
{ "_id" : "90025", "city" : "LOS ANGELES", "loc" : [ -118.448717, 34.044662 ], "pop" : 37746, "state" : "CA" }
{ "_id" : "90026", "city" : "LOS ANGELES", "loc" : [ -118.264641, 34.076629 ], "pop" : 74751, "state" : "CA" }
{ "_id" : "90027", "city" : "LOS ANGELES", "loc" : [ -118.292516, 34.104031 ], "pop" : 50484, "state" : "CA" }
{ "_id" : "90028", "city" : "LOS ANGELES", "loc" : [ -118.325363, 34.100549 ], "pop" : 30152, "state" : "CA" }
Type "it" for more
>
```

Command I used for finding the specified documents in Q2 and the output of this command

```
{ "_id" : "90031", "city" : "LOS ANGELES", "loc" : [ -118.211279, 34.078349 ], "pop" : 39706, "state" : "CA" }
{ "_id" : "90033", "city" : "LOS ANGELES", "loc" : [ -118.208442, 34.048676 ], "pop" : 57469, "state" : "CA" }
{ "_id" : "90034", "city" : "LOS ANGELES", "loc" : [ -118.400482, 34.028977 ], "pop" : 53930, "state" : "CA" }
{ "_id" : "90037", "city" : "LOS ANGELES", "loc" : [ -118.286284, 34.002982 ], "pop" : 56922, "state" : "CA" }
{ "_id" : "90032", "city" : "LOS ANGELES", "loc" : [ -118.175323, 34.081785 ], "pop" : 46602, "state" : "CA" }
{ "_id" : "90038", "city" : "LOS ANGELES", "loc" : [ -118.321489, 34.089769 ], "pop" : 33001, "state" : "CA" }
{ "_id" : "90042", "city" : "LOS ANGELES", "loc" : [ -118.192902, 34.114527 ], "pop" : 60003, "state" : "CA" }
{ "_id" : "90043", "city" : "LOS ANGELES", "loc" : [ -118.33211, 33.987099 ], "pop" : 45397, "state" : "CA" }
{ "_id" : "90044", "city" : "LOS ANGELES", "loc" : [ -118.290119, 33.955089 ], "pop" : 83958, "state" : "CA" }
{ "_id" : "90045", "city" : "LOS ANGELES", "loc" : [ -118.394128, 33.963075 ], "pop" : 36480, "state" : "CA" }
{ "_id" : "90046", "city" : "COLE", "loc" : [ -118.357979, 34.09743 ], "pop" : 48423, "state" : "CA" }
{ "_id" : "90047", "city" : "LOS ANGELES", "loc" : [ -118.307304, 33.956896 ], "pop" : 47975, "state" : "CA" }
{ "_id" : "90049", "city" : "LOS ANGELES", "loc" : [ -118.473967, 34.066 ], "pop" : 35507, "state" : "CA" }
{ "_id" : "90029", "city" : "LOS ANGELES", "loc" : [ -118.294393, 34.089982 ], "pop" : 41120, "state" : "CA" }
{ "_id" : "90057", "city" : "LOS ANGELES", "loc" : [ -118.276262, 34.062172 ], "pop" : 39017, "state" : "CA" }
{ "_id" : "90059", "city" : "LOS ANGELES", "loc" : [ -118.24628, 33.929331 ], "pop" : 34536, "state" : "CA" }
{ "_id" : "90063", "city" : "HAZARD", "loc" : [ -118.185432, 34.044017 ], "pop" : 61123, "state" : "CA" }
{ "_id" : "90065", "city" : "LOS ANGELES", "loc" : [ -118.226637, 34.107307 ], "pop" : 45578, "state" : "CA" }
{ "_id" : "90066", "city" : "LOS ANGELES", "loc" : [ -118.429769, 34.002956 ], "pop" : 53095, "state" : "CA" }
{ "_id" : "90201", "city" : "BELL GARDENS", "loc" : [ -118.17205, 33.969177 ], "pop" : 99568, "state" : "CA" }
Type "it" for more
>
```

Output SS2 (continuation), for question2

```
> it
{ "_id" : "90220", "city" : "RANCHO DOMINGUEZ", "loc" : [ -118.239844, 33.890654 ], "pop" : 47631, "state" : "CA" }
{ "_id" : "90221", "city" : "EAST RANCHO DOMI", "loc" : [ -118.203724, 33.897324 ], "pop" : 47844, "state" : "CA" }
{ "_id" : "90241", "city" : "DOWNEY", "loc" : [ -118.13062, 33.941612 ], "pop" : 34348, "state" : "CA" }
{ "_id" : "90230", "city" : "CULVER CITY", "loc" : [ -118.399115, 33.994927 ], "pop" : 32207, "state" : "CA" }
{ "_id" : "90247", "city" : "GARDENA", "loc" : [ -118.296122, 33.892463 ], "pop" : 42072, "state" : "CA" }
{ "_id" : "90242", "city" : "DOWNEY", "loc" : [ -118.139465, 33.921789 ], "pop" : 36988, "state" : "CA" }
{ "_id" : "90258", "city" : "HOLLY PARK", "loc" : [ -118.346978, 33.913214 ], "pop" : 78511, "state" : "CA" }
{ "_id" : "90255", "city" : "HUNTINGTON PARK", "loc" : [ -118.216053, 33.976879 ], "pop" : 72139, "state" : "CA" }
{ "_id" : "90262", "city" : "LYNWOOD", "loc" : [ -118.201289, 33.924076 ], "pop" : 61606, "state" : "CA" }
{ "_id" : "90266", "city" : "MANHATTAN BEACH", "loc" : [ -118.399562, 33.889594 ], "pop" : 31984, "state" : "CA" }
{ "_id" : "90274", "city" : "PALOS VERDES EST", "loc" : [ -118.374763, 33.770894 ], "pop" : 60381, "state" : "CA" }
{ "_id" : "90277", "city" : "REDONDO BEACH", "loc" : [ -118.383221, 33.830656 ], "pop" : 33102, "state" : "CA" }
{ "_id" : "90278", "city" : "REDONDO BEACH", "loc" : [ -118.371459, 33.870654 ], "pop" : 34873, "state" : "CA" }
{ "_id" : "90291", "city" : "VENICE", "loc" : [ -118.463463, 33.993772 ], "pop" : 32987, "state" : "CA" }
{ "_id" : "90301", "city" : "INGLEWOOD", "loc" : [ -118.355575, 33.955048 ], "pop" : 36481, "state" : "CA" }
{ "_id" : "90280", "city" : "SOUTH GATE", "loc" : [ -118.201349, 33.94617 ], "pop" : 87026, "state" : "CA" }
{ "_id" : "90503", "city" : "TORRANCE", "loc" : [ -118.354236, 33.839709 ], "pop" : 40351, "state" : "CA" }
{ "_id" : "90501", "city" : "TORRANCE", "loc" : [ -118.31183, 33.826817 ], "pop" : 37691, "state" : "CA" }
{ "_id" : "90504", "city" : "TORRANCE", "loc" : [ -118.329517, 33.870815 ], "pop" : 30245, "state" : "CA" }
{ "_id" : "90601", "city" : "WHITTIER", "loc" : [ -118.037139, 34.001119 ], "pop" : 30501, "state" : "CA" }
Type "it" for more
```

Output SS3, for question-2

```
> it
{ "_id" : "90505", "city" : "TORRANCE", "loc" : [ -118.350733, 33.810635 ], "pop" : 33933, "state" : "CA" }
{ "_id" : "90606", "city" : "LOS NIETOS", "loc" : [ -118.065639, 33.977019 ], "pop" : 30185, "state" : "CA" }
{ "_id" : "90620", "city" : "BUENA PARK", "loc" : [ -118.011359, 33.840607 ], "pop" : 42966, "state" : "CA" }
{ "_id" : "90630", "city" : "CYPRESS", "loc" : [ -118.038696, 33.818613 ], "pop" : 43055, "state" : "CA" }
{ "_id" : "90631", "city" : "LA HABRA HEIGHTS", "loc" : [ -117.949703, 33.932173 ], "pop" : 59113, "state" : "CA" }
{ "_id" : "90640", "city" : "MONTEBELLO", "loc" : [ -118.112986, 34.013342 ], "pop" : 59068, "state" : "CA" }
{ "_id" : "90650", "city" : "NORWALK", "loc" : [ -118.081767, 33.90564 ], "pop" : 94188, "state" : "CA" }
{ "_id" : "90605", "city" : "WHITTIER", "loc" : [ -118.035568, 33.941338 ], "pop" : 34035, "state" : "CA" }
{ "_id" : "90638", "city" : "LA MIRADA", "loc" : [ -118.010081, 33.906681 ], "pop" : 40452, "state" : "CA" }
{ "_id" : "90660", "city" : "PICO RIVERA", "loc" : [ -118.088269, 33.98857 ], "pop" : 58891, "state" : "CA" }
{ "_id" : "90706", "city" : "BELLFLOWER", "loc" : [ -118.126527, 33.886676 ], "pop" : 61650, "state" : "CA" }
{ "_id" : "90701", "city" : "CERRITOS", "loc" : [ -118.068046, 33.866568 ], "pop" : 69130, "state" : "CA" }
{ "_id" : "90604", "city" : "WHITTIER", "loc" : [ -118.012075, 33.929931 ], "pop" : 36371, "state" : "CA" }
{ "_id" : "90731", "city" : "SAN PEDRO", "loc" : [ -118.291425, 33.733894 ], "pop" : 58821, "state" : "CA" }
{ "_id" : "90744", "city" : "WILMINGTON", "loc" : [ -118.264451, 33.785475 ], "pop" : 49178, "state" : "CA" }
{ "_id" : "90803", "city" : "LONG BEACH", "loc" : [ -118.134073, 33.761932 ], "pop" : 32492, "state" : "CA" }
{ "_id" : "90802", "city" : "LONG BEACH", "loc" : [ -118.182025, 33.770553 ], "pop" : 33906, "state" : "CA" }
{ "_id" : "90745", "city" : "CARSON", "loc" : [ -118.268352, 33.822968 ], "pop" : 50251, "state" : "CA" }
{ "_id" : "90804", "city" : "SIGNAL HILL", "loc" : [ -118.155187, 33.782993 ], "pop" : 36092, "state" : "CA" }
{ "_id" : "90805", "city" : "LONG BEACH", "loc" : [ -118.180102, 33.863457 ], "pop" : 74011, "state" : "CA" }
Type "it" for more
>
```

Output SS4, for question2

```
> it
{ "_id" : "90723", "city" : "PARAMOUNT", "loc" : [ -118.163152, 33.896867 ], "pop" : 46679, "state" : "CA" }
{ "_id" : "90808", "city" : "LONG BEACH", "loc" : [ -118.110299, 33.824145 ], "pop" : 37809, "state" : "CA" }
{ "_id" : "90810", "city" : "CARSON", "loc" : [ -118.215006, 33.810985 ], "pop" : 36694, "state" : "CA" }
{ "_id" : "90806", "city" : "SIGNAL HILL", "loc" : [ -118.187443, 33.799319 ], "pop" : 44982, "state" : "CA" }
{ "_id" : "90815", "city" : "LONG BEACH", "loc" : [ -118.119249, 33.793908 ], "pop" : 38603, "state" : "CA" }
{ "_id" : "90813", "city" : "LONG BEACH", "loc" : [ -118.183488, 33.78202 ], "pop" : 58189, "state" : "CA" }
{ "_id" : "91016", "city" : "MONROVIA", "loc" : [ -118.001376, 34.143959 ], "pop" : 39015, "state" : "CA" }
{ "_id" : "91104", "city" : "PASADENA", "loc" : [ -118.12689, 34.167776 ], "pop" : 37973, "state" : "CA" }
{ "_id" : "91001", "city" : "ALTADENA", "loc" : [ -118.13919, 34.191187 ], "pop" : 36013, "state" : "CA" }
{ "_id" : "91006", "city" : "ARCADIA", "loc" : [ -118.026374, 34.132389 ], "pop" : 38550, "state" : "CA" }
{ "_id" : "91107", "city" : "PASADENA", "loc" : [ -118.088905, 34.150997 ], "pop" : 31390, "state" : "CA" }
{ "_id" : "91205", "city" : "GLENDALE", "loc" : [ -118.24245, 34.137798 ], "pop" : 38428, "state" : "CA" }
{ "_id" : "91301", "city" : "OAK PARK", "loc" : [ -118.75718, 34.157163 ], "pop" : 35520, "state" : "CA" }
{ "_id" : "91206", "city" : "GLENDALE", "loc" : [ -118.232217, 34.155605 ], "pop" : 30415, "state" : "CA" }
{ "_id" : "91306", "city" : "WINNETKA", "loc" : [ -118.57492, 34.2092 ], "pop" : 39261, "state" : "CA" }
{ "_id" : "91304", "city" : "CANOGA PARK", "loc" : [ -118.611059, 34.219671 ], "pop" : 43675, "state" : "CA" }
{ "_id" : "91320", "city" : "NEWBURY PARK", "loc" : [ -118.935798, 34.177393 ], "pop" : 31941, "state" : "CA" }
{ "_id" : "91311", "city" : "CHATSWORTH", "loc" : [ -118.591357, 34.258253 ], "pop" : 37225, "state" : "CA" }
{ "_id" : "91335", "city" : "RESEDA", "loc" : [ -118.539071, 34.200663 ], "pop" : 62117, "state" : "CA" }
{ "_id" : "91331", "city" : "ARLETA", "loc" : [ -118.420692, 34.258081 ], "pop" : 88114, "state" : "CA" }
```

Output SS5, for question2

For Q2, the output is very large. I have included first 5 part from the output (see above). For Q2, I have used the following command:

db.zipcodes.aggregate([{ $match: { pop: { $gte: 30000 }, state:"CA" }}])

3rd question:

```
> db.zipcodes.find({$or: [ { "loc.0": { $lt: -120 } , "loc.1": {$gt: 40} } , {"state": {$not:{$eq: "CA"}}}] })
{ "_id" : "01007", "city" : "BELCHERTOWN", "loc" : [ -72.410953, 42.275103 ], "pop" : 10579, "state" : "MA" }
{ "_id" : "01011", "city" : "CHESTER", "loc" : [ -72.988761, 42.279421 ], "pop" : 1688, "state" : "MA" }
{ "_id" : "01002", "city" : "CUSHMAN", "loc" : [ -72.51565, 42.377017 ], "pop" : 36963, "state" : "MA" }
{ "_id" : "01010", "city" : "BRIMFIELD", "loc" : [ -72.188455, 42.116543 ], "pop" : 3706, "state" : "MA" }
{ "_id" : "01001", "city" : "AGAWAM", "loc" : [ -72.622739, 42.070206 ], "pop" : 15338, "state" : "MA" }
{ "_id" : "01005", "city" : "BARRE", "loc" : [ -72.108354, 42.409698 ], "pop" : 4546, "state" : "MA" }
{ "_id" : "01008", "city" : "BLANDFORD", "loc" : [ -72.936114, 42.182949 ], "pop" : 1240, "state" : "MA" }
{ "_id" : "01013", "city" : "CHICOPEE", "loc" : [ -72.607962, 42.162046 ], "pop" : 23396, "state" : "MA" }
{ "_id" : "01020", "city" : "CHICOPEE", "loc" : [ -72.576142, 42.176443 ], "pop" : 31495, "state" : "MA" }
{ "_id" : "01022", "city" : "WESTOVER AFB", "loc" : [ -72.558657, 42.196672 ], "pop" : 1764, "state" : "MA" }
{ "_id" : "01026", "city" : "CUMMINGTON", "loc" : [ -72.905767, 42.435296 ], "pop" : 1484, "state" : "MA" }
{ "_id" : "01028", "city" : "EAST LONGMEADOW", "loc" : [ -72.505565, 42.067203 ], "pop" : 13367, "state" : "MA" }
{ "_id" : "01031", "city" : "GILBERTVILLE", "loc" : [ -72.198585, 42.332194 ], "pop" : 2385, "state" : "MA" }
{ "_id" : "01012", "city" : "CHESTERFIELD", "loc" : [ -72.833309, 42.38167 ], "pop" : 177, "state" : "MA" }
{ "_id" : "01030", "city" : "FEEDING HILLS", "loc" : [ -72.675077, 42.07182 ], "pop" : 11985, "state" : "MA" }
{ "_id" : "01032", "city" : "GOSHEN", "loc" : [ -72.844092, 42.466234 ], "pop" : 122, "state" : "MA" }
{ "_id" : "01027", "city" : "MOUNT TOM", "loc" : [ -72.679921, 42.264319 ], "pop" : 16864, "state" : "MA" }
{ "_id" : "01033", "city" : "GRANBY", "loc" : [ -72.520001, 42.255704 ], "pop" : 5526, "state" : "MA" }
{ "_id" : "01034", "city" : "TOLLAND", "loc" : [ -72.908793, 42.070234 ], "pop" : 1652, "state" : "MA" }
{ "_id" : "01050", "city" : "HUNTINGTON", "loc" : [ -72.873341, 42.265301 ], "pop" : 2084, "state" : "MA" }
Type "it" for more
>
```

The first output of Q3 where only the 1st case in the Q3 is satisfied

For Q3, I have used the following command:

db.zipcodes.find({$or: [{ "loc.0": { $lt: -120 } , "loc.1": {$gt: 40} } , {"state": {$not:{$eq: "CA"}}}] })

```
> it
{ "_id" : "97019", "city" : "CORBETT", "loc" : [ -122.241746, 45.522116 ], "pop" : 2355, "state" : "OR" }
{ "_id" : "97018", "city" : "COLUMBIA CITY", "loc" : [ -122.812174, 45.892474 ], "pop" : 1003, "state" : "OR" }
```

The second output of Q3 where both of the conditions are satisfied

```
> it
{ "_id" : "96128", "city" : "STANDISH", "loc" : [ -120.406847, 40.350863 ], "pop" : 340, "state" : "CA" }
{ "_id" : "96130", "city" : "SUSANVILLE", "loc" : [ -120.646442, 40.398282 ], "pop" : 19347, "state" : "CA" }
{ "_id" : "96134", "city" : "TULELAKE", "loc" : [ -121.434688, 41.931621 ], "pop" : 2613, "state" : "CA" }
{ "_id" : "96132", "city" : "TERMO", "loc" : [ -120.517378, 40.946667 ], "pop" : 199, "state" : "CA" }
{ "_id" : "96137", "city" : "PENINSULA VILLAG", "loc" : [ -121.109224, 40.270359 ], "pop" : 1843, "state" : "CA" }
{ "_id" : "96136", "city" : "WENDEL", "loc" : [ -120.352156, 40.346233 ], "pop" : 148, "state" : "CA" }
{ "_id" : "96121", "city" : "MILFORD", "loc" : [ -120.389508, 40.182763 ], "pop" : 376, "state" : "CA" }
{ "_id" : "96123", "city" : "RAVENDALE", "loc" : [ -120.16001, 40.831705 ], "pop" : 89, "state" : "CA" }
```

The third output of Q3 where only the 2nd case in the Q3 is satisfied

4th question:

```
> db.zipcodes.aggregate([ {$group:{_id:"$city",totalPop: {$sum: "$pop"}}}, {$sort: {totalPop: -1}}, {$limit: 5}])
{ "_id" : "CHICAGO", "totalPop" : 2452177 }
{ "_id" : "BROOKLYN", "totalPop" : 2341387 }
{ "_id" : "HOUSTON", "totalPop" : 2123053 }
{ "_id" : "LOS ANGELES", "totalPop" : 2102295 }
{ "_id" : "PHILADELPHIA", "totalPop" : 1639862 }
>
```

The command I used for the 4th question, and the output of this command

For Q4, I have used the following command:

db.zipcodes.aggregate([{$group:{_id:"$city",totalPop: {$sum: "$pop"}}}, {$sort: {totalPop: -1}}, {$limit: 5}])

5th question:

```
> db.zipcodes.aggregate([{"$group" : {"_id": "$state", "count": {"$sum" : 1}}},{"$match": {"count":{ $gte: 800} }},{"$sort": {"count": -1}}])
{ "_id" : "TX", "count" : 1671 }
{ "_id" : "NY", "count" : 1595 }
{ "_id" : "CA", "count" : 1516 }
{ "_id" : "PA", "count" : 1458 }
{ "_id" : "IL", "count" : 1237 }
{ "_id" : "OH", "count" : 1007 }
{ "_id" : "MO", "count" : 994 }
{ "_id" : "IA", "count" : 922 }
{ "_id" : "MN", "count" : 882 }
{ "_id" : "MI", "count" : 876 }
{ "_id" : "VA", "count" : 816 }
{ "_id" : "KY", "count" : 809 }
{ "_id" : "FL", "count" : 804 }
>
```

The command I used in the Q5, and the output of this command

For Q5, I have used the following command:

db.zipcodes.aggregate([{"$group" : {"_id": "$state", "count": {"$sum" : 1}}},{"$match": {"count":{ $gte: 800} }},{"$sort": {"count": -1}}])

6th question:

```
> db.createCollection("customers", {
... validator: {
... $jsonSchema : {
... bsonType : "object",
... required : [ "name" , "zipcode" , "avg_rating" ],
... properties : {
... name : {
... bsonType : "string",
... description : "must be a string and is required"
... },
... zipcode : {
... bsonType : "string",
... description : "must be a string and is required"
... },
... avg_rating : {
... bsonType : ["double"],
... minimum : 0.0
... ,
... maximum : 10.0,
... description : "must be a double in [0.0,10.0] and is required"
... },
... last_order : {
... bsonType : "object",
... required : ["year"],
... properties : {
... year : {
... bsonType : "int",
... description : "must be an integer and is required"
... },
... tags : {
... bsonType : "array",
... description : "must be an array if it exists",
... items : {
... bsonType : "string"
... }
... } } } } } } } )
{ "ok" : 1 }
```

Creating a collection called "customers" by using db.createCollection() command and the output of this command

```
> show collections
customers
zipcodes
>
```

Showing the collections after creating the collection called "customers"

7th question:

```
> db.customers.insertMany([
... {name: "Barış Kaplan", zipcode: "99503", avg_rating: 8.3}
... ,
... {name: "Jake G.", zipcode: "90025", avg_rating: 3.5, last_order: {year: NumberInt(2009)}},
... {name: "Adam S.", zipcode: "33126", avg_rating: 4.9, last_order: {year: NumberInt(2019),tags: ["book","fiction"]}},
... {name: "Harold H.", zipcode: "90010", avg_rating: 1.0, last_order: {year: NumberInt(3005)}}
... ])
{
        "acknowledged" : true,
        "insertedIds" : [
                ObjectId("628ac253d447fafdb873c999"),
                ObjectId("628ac253d447fafdb873c99a"),
                ObjectId("628ac253d447fafdb873c99b"),
                ObjectId("628ac253d447fafdb873c99c")
        ]
}
>
```

Inserting all of the records specified in Q7 in 1 query to the collection called "customers"

```
> db.customers.find()
{ "_id" : ObjectId("628a8d1762156794 7a460586"), "name" : "Barış Kaplan", "zipcode" : "99503", "avg_rating" : 8.3 }
{ "_id" : ObjectId("628a8da8621567947a460587"), "name" : "Jake G.", "zipcode" : "90025", "avg_rating" : 3.5, "last_order" : { "year" : 2009 } }
{ "_id" : ObjectId("628a8e9c621567947a460588"), "name" : "Adam S.", "zipcode" : "33126", "avg_rating" : 4.9, "last_order" : { "year" : 2019, "tags" : [ "book", "fiction" ] } }
{ "_id" : ObjectId("628a8ef9621567947a460589"), "name" : "Harold H.", "zipcode" : "90010", "avg_rating" : 1, "last_order" : { "year" : 3005 } }
>
```

By using the command called "db.customers.find()",showing all the documents in the collection called "customers"

8th question:

```
> db.customers.deleteMany({"last_order.year": {$gt: 2022} })
{ "acknowledged" : true, "deletedCount" : 1 }
> db.customers.find()
{ "_id" : ObjectId("628b5b465e99d0e002f0f584"), "name" : "Barış Kaplan", "zipcode" : "99503", "avg_rating" : 8.3 }
{ "_id" : ObjectId("628b5b465e99d0e002f0f585"), "name" : "Jake G.", "zipcode" : "90025", "avg_rating" : 3.5, "last_order" : { "year" : 2009 } }
{ "_id" : ObjectId("628b5b465e99d0e002f0f586"), "name" : "Adam S.", "zipcode" : "33126", "avg_rating" : 4.9, "last_order" : { "year" : 2019, "tags" : [ "book", "fiction" ] } }
>
```

Executing the command db.customers.deleteMany() for deleting all documents which include an order after the year 2022 , and after that, viewing all of the documents inside the collection called "customers" by using the command called "db.customers.find()". You can see from the above screenshot that the order at the year 3005, which is after the year 2022, is deleted successfully from the customers collection.

The command I used for deleting all the documents which include an order after the year 2022 is below:

db.customers.deleteMany({"last_order.year": {$gt: 2022} }) (Command-1)

The command I used for showing all the documents under the customers collection is below:

db.customers.find() (Command-2)

9th question:

```
> db.customers.updateOne({name: "Barış Kaplan"},{$set: {avg_rating: 15.0} } )
WriteError({
        "index" : 0,
        "code" : 121,
        "errmsg" : "Document failed validation",
        "op" : {
                "q" : {
                        "name" : "Barış Kaplan"
                },
                "u" : {
                        "$set" : {
                                "avg_rating" : 15
                        }
                },
                "multi" : false,
                "upsert" : false
        },
        "errInfo" : {
                "failingDocumentId" : ObjectId("628b5b465e99d0e002f0f584"),
                "details" : {
                        "operatorName" : "$jsonSchema",
                        "schemaRulesNotSatisfied" : [
                                {
                                        "operatorName" : "properties",
                                        "propertiesNotSatisfied" : [
                                                {
                                                        "propertyName" : "avg_rating",
                                                        "details" : [
                                                                {
                                                                        "operatorName" : "maximum",
                                                                        "specifiedAs" : {
                                                                                "maximum" : 10
                                                                        },
                                                                        "reason" : "comparison failed",
                                                                        "consideredValue" : 15
                                                                }
                                                        ]
                                                }
                                        ]
                                }
                        ]
                }
        }
})
```

The query I am executing for updating my rating to 15, and the output I received from the MongoDB after I execute the query (in the 9th question)

When we have designed the schema for the collection called “customers”, we have specified the maximum value of the field called “avg_rating” as 10.0. In this question, we are asked to update the avg_rating field of us to 15.0. Since 15.0 is greater than the maximum value of the avg_rating field (which is 10.0), the response I obtained from the MongoDB states that the schema rules are not satisfied , and it also states that an error is thrown with the reason “comparison failed”.

```
WriteError({
        "index" : 0,
        "code" : 121,
        "errmsg" : "Document failed validation",
        "op" : {
                "q" : {
                        "name" : "Barış Kaplan"
                },
                "u" : {
                        "$set" : {
                                "avg_rating" : 15
                        }
                },
                "multi" : false,
                "upsert" : false
        },
        "errInfo" : {
                "failingDocumentId" : ObjectId("628b5b465e99d0e002f0f584"),
                "details" : {
                        "operatorName" : "$jsonSchema",
                        "schemaRulesNotSatisfied" : [
                                {
                                        "operatorName" : "properties",
                                        "propertiesNotSatisfied" : [
                                                {
                                                        "propertyName" : "avg_rating",
```

MongoDB Response SS2

```
                                        "propertiesNotSatisfied" : [
                                                {
                                                        "propertyName" : "avg_rating",
                                                        "details" : [
                                                                {
                                                                        "operatorName" : "maximum",
                                                                        "specifiedAs" : {
                                                                                "maximum" : 10
                                                                        },
                                                                        "reason" : "comparison failed",
                                                                        "consideredValue" : 15
                                                                }
                                                        ]
                                                }
                                        ]
                                }
                        ]
                }
        }
})
WriteError@src/mongo/shell/bulk_api.js:465:48
mergeBatchResults@src/mongo/shell/bulk_api.js:871:49
executeBatch@src/mongo/shell/bulk_api.js:940:13
Bulk/this.execute@src/mongo/shell/bulk_api.js:1182:21
DBCollection.prototype.updateOne@src/mongo/shell/crud_api.js:600:17
@(shell):1:1
```

MongoDB Response SS3

10th question:

```
> db.customers.aggregate([{
... $lookup: {
...  from: "zipcodes",
... localField: "zipcode",
... foreignField: "_id",
... as: "city_and_state"
... }
... }, {$project: {city_and_state:1,name:1}},
... {$project: {_id:0,"city_and_state.loc":0, "city_and_state.pop":0,"city_and_state._id":0}}
... ])
{ "name" : "Barış Kaplan", "city_and_state" : [ { "city" : "ANCHORAGE", "state" : "AK" } ] }
{ "name" : "Adam S.", "city_and_state" : [ { "city" : "MIAMI", "state" : "FL" } ] }
{ "name" : "Jake G.", "city_and_state" : [ { "city" : "LOS ANGELES", "state" : "CA" } ] }
>
```

The command I used in the question 10, and the output of this command

The command I have used in the question 10 is as below:

```
db.customers.aggregate([{
$lookup: {
from: "zipcodes",
localField: "zipcode",
foreignField: "_id",
as: "city_and_state"
}
}, {$project: {city_and_state:1,name:1}},
{$project: {_id:0,"city_and_state.loc":0, "city_and_state.pop":0,"city_and_state._id":0}}
])
```

11th question:

```
(base) barissss-mbp:Desktop barissss$ mongoexport --db hw5 --collection customers --out /Users/barissss/Desktop/customers.json
2022-05-24T01:32:57.313+0300    connected to: mongodb://localhost/
2022-05-24T01:32:57.319+0300    exported 3 records
(base) barissss-mbp:Desktop barissss$
```

The usage of the mongoexport command and the output of this command.

The command I used for exporting “customers” collections into the “customers.json” file:

mongoexport –db hw5 –collection customers -out /Users/barissss/Desktop/customers.json

MY Name-Surname: Barış Kaplan
My KU ID Number: 0069054 (69054)
My KU Email Address: bkaplan18@ku.edu.tr
Lecture Name: COMP306, Database Management Systems
Homework Topic: NoSQL (MongoDB)
Homework Number : Homework #5