**HKU Business School**

**MSBA7024 Database Design and Management**

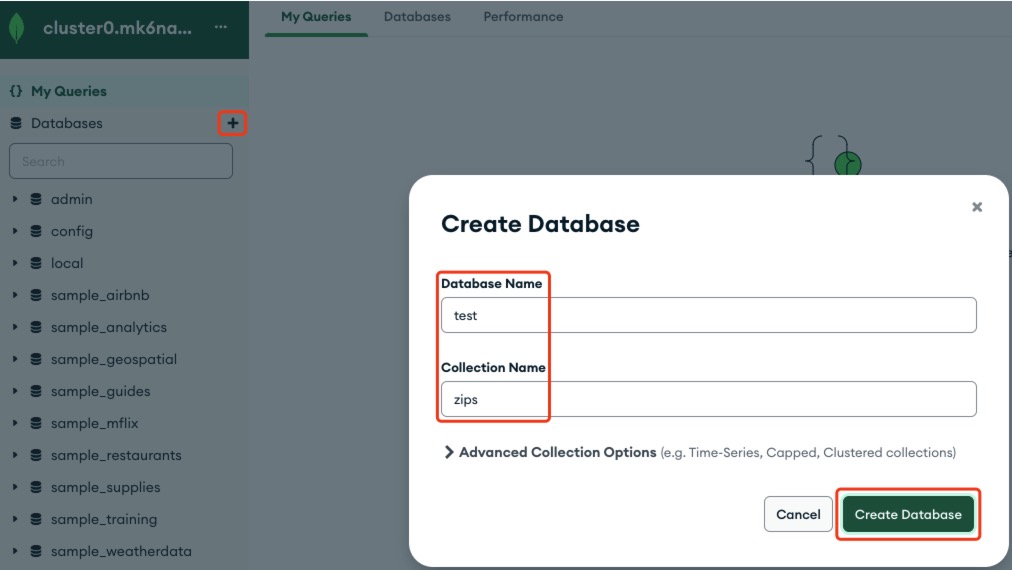
**Assignment 3**

In this assignment, you are asked to write some queries to manipulate data in MongoDB through MongoDB Shell. Please follow these steps:

1. Access a MongoDB Atlas cluster, following the instructions in the tutorial.
2. Download the data and load them into MongoDB.
3. Download the data file to your computer from:

<https://media.mongodb.org/zips.json>

1. Import data using MongoDB Compass:
2. Open MongoDB Compass and connect to a cluster (e.g., cluster0 created in class).
3. Create the test database and the zips collection.



1. Click the zips collection in the test database in the left-hand pane.
2. Click the *Add Data* dropdown and select *Import File*.
3. Select the zips.json file downloaded in the first step. Choose JSON as the file type. Click *IMPORT.*
4. Connect to the cluster with the MongoDB Shell (MONGOSH inside MongoDB Compass). Inside the Mongo Shell, type the following:

show databases

use test

It should show the list of the current databases and change the current database to test.

1. You can now execute Mongo Shell commands against the test database, e.g.,

show collections

db.zips.find().count() ← this will give you 29353 if the data import was successful.

1. Write MongoDB Shell commands in Javascript to do the following with reference to the zips collection in the test database. The data model can be found at: [https://docs.mongodb.com/manual/tutorial/aggregation-zip-code-data-set/#data-model](https://docs.mongodb.com/manual/tutorial/aggregation-zip-code-data-set/" \l "data-model)

Include the MongoDB Shell command and the output in your answer in this file. Question 0 has been done for you as an example.

1. Retrieve the data of all records where the city name is YUMA.

Answer:

db.zips.find({"city":"YUMA"})

[

{

\_id: '38390',

city: 'YUMA',

loc: [ -88.381878, 35.867964 ],

pop: 1063,

state: 'TN'

},

{

\_id: '80759',

city: 'YUMA',

loc: [ -102.707174, 40.130063 ],

pop: 3811,

state: 'CO'

},

{

\_id: '85364',

city: 'YUMA',

loc: [ -114.642362, 32.701507 ],

pop: 57131,

state: 'AZ'

}

]

1. Retrieve the zip code, city name, population, and state name of the record with zip code ‘19125’.

Answer:

db.zips.find( { \_id: "19125" }, { loc: 0 } )

|  |
| --- |
| { \_id: **'19125'**, city: **'PHILADELPHIA'**, pop: 24521, state: **'PA'** } |

1. Retrieve the zip code, city name and population of all records with a population of 1000.

Answer:

db.zips.find( { pop: 1000 }, { loc: 0, state: 0 } )

|  |
| --- |
| { \_id: **'22724'**, city: **'JEFFERSONTON'**, pop: 1000 } |

|  |
| --- |
| { \_id: **'34773'**, city: **'SAINT CLOUD'**, pop: 1000 } |

|  |
| --- |
| { \_id: **'56145'**, city: **'JEFFERS'**, pop: 1000 } |

|  |
| --- |
| { \_id: **'63622'**, city: **'BELGRADE'**, pop: 1000 } |

|  |
| --- |
| { \_id: **'69032'**, city: **'HAYES CENTER'**, pop: 1000 } |

87

1. Retrieve the zip code, city name, and population of all records where the state name is NY and the population no more than 1.

Answer:

db.zips.find({state: "NY", pop: {$lt : 1 }}, {loc: 0, state: 0})

|  |
| --- |
| { \_id: **'12922'**, city: **'CHILDWOLD'**, pop: 0 } |

|  |
| --- |
| { \_id: **'13333'**, city: **'EAST SPRINGFIELD'**, pop: 0 } |

|  |
| --- |
| { \_id: **'13436'**, city: **'RAQUETTE LAKE'**, pop: 0 } |

1. Retrieve the zip code, city name, and population of the top 3 records with the smallest population and a zip code beginning with ‘11’.

Answer:

db.zips.find({ \_id: {$regex: /^11/} }, {loc: 0, state: 0}).sort({pop: 1}).limit(3)

|  |
| --- |
| { \_id: **'11042'**, city: **'NEW HYDE PARK'**, pop: 1 } |

|  |
| --- |
| { \_id: **'11251'**, city: **'BROOKLYN NAVY YA'**, pop: 18 } |

|  |
| --- |
| { \_id: **'11371'**, city: **'FLUSHING'**, pop: 49 } |

1. Retrieve the state name and the number of zip codes (called count) of the 3 states with the most zip codes. Sort the results in descending order of the count.

Answer:

db.zips.aggregate([

{$group: {\_id: {state: "$state"}, count: {$sum: 1}}},

{$sort:{"count":-1}},

{$limit : 3}

])

|  |
| --- |
| { \_id: { state: **'TX'** }, count: 1671 } |

|  |
| --- |
| { \_id: { state: **'NY'** }, count: 1595 } |

|  |
| --- |
| { \_id: { state: **'CA'** }, count: 1516 } |

1. Create a new collection called “students” as follow:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **id** | **name** | **gender** | **Chinese** | **Math** | **English** |
| 1 | Tom | M | 90 | 87 | 80 |
| 2 | Alex | M | 88 | 95 | 75 |
| 3 | Lucy | F | 92 | 79 | 90 |
| 4 | Vivian | F | 94 | 88 | 85 |

Answer:

db.createCollection("students")

|  |
| --- |
| { ok: 1 } |

db.students.insertMany(

[

{ \_id: 1, name: "Tom", gender: "M", Chinese: 90, Math: 87, English: 80},

{ \_id: 2, name: "Alex", gender: "M", Chinese: 88, Math: 95, English: 75},

{ \_id: 3, name: "Lucy", gender: "F", Chinese: 92, Math: 79, English: 90},

{ \_id: 4, name: "Vivian", gender: "F", Chinese: 94, Math: 88, English: 75},

]

)

|  |
| --- |
| { acknowledged: true, |
| insertedIds: { **'0'**: 1, **'1'**: 2, **'2'**: 3, **'3'**: 4 } } |

Answer Q7-10 using the collection “students” created in Q6.

1. Retrieve the id and name of all students with Chinese or English grade over 90.

Answer

db.students.find(

{$or:

[{Chinese: {$gt: 90}}, {English: {$gt: 90}}]

},

{\_id: 1, name: 1}

)

|  |
| --- |
| { \_id: 3, name: **'Lucy'** } |

|  |
| --- |
| { \_id: 4, name: **'Vivian'** } |

1. Retrieve the total number of female students.

Answer:

db.students.find({gender: {$eq: "F"}}).count()

|  |
| --- |
| 2 |

1. Retrieve the average math grade of all female students and the average of all male students.

Answer:

db.students.aggregate([

{

$group: {

\_id: "$gender",

total\_count: { $sum: 1 },

Average\_Math: { $avg: { $cond: [ {gender: {$eq :["$gender", "M"]}}, "$Math", 0 ] } },

}

}

])

|  |
| --- |
| { \_id: **'M'**, total\_count: 2, Average\_Math: 91 } |

|  |
| --- |
| { \_id: **'F'**, total\_count: 2, Average\_Math: 83.5 } |

1. Drop the collection ‘students’.

Answer:

db.students.drop()

|  |
| --- |
| true |

Please submit your answers (MS Word or PDF file) through Moodle (moodle.hku.hk) before the deadline. Use your student ID to name your file (e.g., 3035xxxxxx.pdf) and put your name and university number inside your submission. This is an individual assignment, so please make sure you work on it yourself and do not share your work with others. If your submission is late for 24 hours or less, 20% will be deducted. If your submission is late for more than 24 hours, no credit will be given.