

DATABASE SYSTEMS LAB



LAB TASK # 13

Submitted By

Umair Azad (19P-0030)

**Fast National University of Computer and Emerging
Sciences, Peshawar**

Department of Computer Science

Exercises:

Use the cities collections that you have already imported in the last lab.

- 1) Write a Query to return all the documents whose cities population is less than 30 not equal to zero, then uses the limit clause to limit the number of documents being returned to just 2.

```
> db.cities.find({population:{$lt:30,$ne:0}}).limit(2).pretty()
{
  "_id" : ObjectId("62866c7dc69fdd0df9700350"),
  "name" : "Tanggul",
  "country" : "ID",
  "timezone" : "Asia/Jakarta",
  "population" : 3,
  "location" : {
    "latitude" : -8.1645,
    "longitude" : 113.4525
  }
}
{
  "_id" : ObjectId("62866c7fc69fdd0df97040e9"),
  "name" : "Ereencav",
  "country" : "MN",
  "timezone" : "Asia/Choibalsan",
  "population" : 23,
  "location" : {
    "longitude" : 49.8807,
    "latitude" : 115.72526
  }
}
```

- 2) Write a Query to count the number of the documents whose timezone is “Asia/Jakarta”.

```
> db.cities.count({timezone : "Asia/Jakarta"})
1430
>
```

- 3) Write a Query to return all the documents whose country is “PK” and country timezone is “Asia/Karachi” and return the documents based on the descending order of population.

```
> db.cities.find({country : "PK",timezone:"Asia/Karachi"}).sort({
... population : -1}).forEach(printjson)
```

```
{
  "_id" : ObjectId("62866c80c69fdd0df970645e"),
  "name" : "Karachi",
  "country" : "PK",
  "timezone" : "Asia/Karachi",
  "population" : 11624219,
  "location" : {
    "longitude" : 24.9056,
    "latitude" : 67.0822
  }
}

{
  "_id" : ObjectId("62866c80c69fdd0df9706432"),
  "name" : "Lahore",
  "country" : "PK",
  "timezone" : "Asia/Karachi",
  "population" : 6310888,
  "location" : {
    "longitude" : 31.54972,
    "latitude" : 74.34361
  }
}

{
  "_id" : ObjectId("62866c80c69fdd0df97064af"),
  "name" : "Faisalābād",
  "country" : "PK",
  "timezone" : "Asia/Karachi",
  "population" : 2506595,
  "location" : {
    "longitude" : 31.41667,
    "latitude" : 73.08333
  }
}

{
  "_id" : ObjectId("62866c80c69fdd0df97063d1"),
  "name" : "Rāwalpindi",
  "country" : "PK",
  "timezone" : "Asia/Karachi",
  "population" : 1743101,
  "location" : {
```

- 4) Write a query to get all the Indexes of cities collection and then add the index on population field and then drop the index on population field.

```
> db.cities.getIndexes()
[ { "v" : 2, "key" : { "_id" : 1 }, "name" : "_id_" } ]
> db.cities.createIndex({population:1})
{
  "numIndexesBefore" : 1,
  "numIndexesAfter" : 2,
  "createdCollectionAutomatically" : false,
  "ok" : 1
}
> db.cities.dropIndex({population:1})
{
  "nIndexesWas" : 2, "ok" : 1 }
>
```

- 5) Use MongoDB compass filter tab to write queries for finding:
- All those cities whose time zone is Europe/Andorra.

The screenshot shows the MongoDB Compass interface for the 'P19-0030.cities' collection. The 'FILTER' tab is active, displaying the query `{ "timezone": "Europe/Andorra" }`. The 'FIND' button is highlighted. Below the filter bar, there are buttons for 'ADD DATA', 'VIEW', and a toggle for 'JSON' view. The results section shows three documents, each with fields: `_id`, `name`, `country`, `timezone`, `population`, and `location`. The first document is for 'Sant Julià de Lòria' with a population of 8022. The second is for 'Pas de la Casa' with a population of 2363. The third is for 'Ordino' with a population of 2363. All three documents have a `timezone` of 'Europe/Andorra'.

P19-0030.cities 99.8% DOCUMENTS

Documents Aggregations Schema Explain Plan Indexes Validation

FILTER { "timezone": "Europe/Andorra" } OPTIONS FIND

ADD DATA VIEW JSON

Displaying documents 1 - 9 of 9

```
{
  "_id": ObjectId('62866c77c69fdd0df96f5bd1'),
  "name": "Sant Julià de Lòria",
  "country": "AD",
  "timezone": "Europe/Andorra",
  "population": 8022,
  "location": Object
}
```

```
{
  "_id": ObjectId('62866c77c69fdd0df96f5bd2'),
  "name": "Pas de la Casa",
  "country": "AD",
  "timezone": "Europe/Andorra",
  "population": 2363,
  "location": Object
}
```

```
{
  "_id": ObjectId('62866c77c69fdd0df96f5bd3'),
  "name": "Ordino",
  "country": "AD",
  "timezone": "Europe/Andorra",
  "population": 2363,
  "location": Object
}
```

- ii. All those cities whose population is greater than 12955000 and country is AR.

P19-0030.cities

99.1
DOCUM

Documents Aggregations Schema Explain Plan Indexes Validation

FILTER {population : {\$gt:12955000} ,country : "AR"} **OPTIONS** **FIND**

ADD DATA **VIEW** **JSON** **YAML** **CSV**

Displaying documents 1 - 1 of 1

```
_id: ObjectId('62866c77c69fdd0df96f5eb4')
name: "Buenos Aires"
country: "AR"
timezone: "America/Argentina/Buenos_Aires"
population: 13076300
> location: Object
```

- iii. A city whose longitude equals to 1.6. Your query should return location and population fields only. (hint: use project)

P19-0030.cities

99.1
DOCUM

Documents Aggregations Schema Explain Plan Indexes Validation

FILTER {"location.longitude":{\$eq:1.6}} **OPTIONS** **FIND**

PROJECT {location:1,population:1,_id:0}

SORT { field: -1 } or [['field', -1]] **MAX TIME MS** 60000

COLLATION { locale: 'simple' } **SKIP** 0 **LIMIT** 0

VIEW **JSON** **YAML** **CSV**

Displaying documents 1 - 2 of 2

```
population: 73176
> location: Object

population: 75350
> location: Object
```