MONASH University
Information Technology

GROUP ASSIGNMENT COVER SHEET

Student ID Number	Surname	Given Names
29082498	*Wang*	**Yunxuan**
27135519	*Tang*	Hongliang

* Please include the names of all other group members.

Unit name and code	FIT5137 Advanced database technology	
Title of assignment	Assignment 1: MongoDB & Cassandra	
Lecturer/tutor	Hongli Song Shuyi Sun	
Tutorial day and time	Firday 8am-10am (Lab01) Friday 10am-12pm(lab02)	**Campus** Clayton
Is this an authorised group assignment? ☑ **Yes** ☐ **No**		
Has any part of this assignment been previously submitted as part of another unit/course? ☐ **Yes** ☑ **No**		
Due Date 15/09/2021		**Date submitted.** 15/09/2021

All work must be submitted by the due date. If an extension of work is granted this must be specified with the signature of the lecturer/tutor.

Extension granted until (date) ..... **Signature of lecturer/tutor .....**

Please note that it is your responsibility to retain copies of your assessments.

Intentional plagiarism or collusion amounts to cheating under Part 7 of the Monash University (Council) Regulations

Plagiarism: Plagiarism means taking and using another person's ideas or manner of expressing them and passing them off as one's own. For example, by failing to give appropriate acknowledgement. The material used can be from any source (staff, students or the internet, published and unpublished works).

Collusion: Collusion means unauthorised collaboration with another person on assessable written, oral or practical work and includes paying another person to complete all or part of the work.

Where there are reasonable grounds for believing that intentional plagiarism or collusion has occurred, this will be reported to the Associate Dean (Education) or delegate, who may disallow the work concerned by prohibiting assessment or refer the matter to the Faculty Discipline Panel for a hearing.

Student Statement:

- I have read the university's Student Academic Integrity Policy and Procedures.
- I understand the consequences of engaging in plagiarism and collusion as described in Part 7 of the Monash University (Council) Regulations http://adm.monash.edu/legal/legislation/statutes
- I have taken proper care to safeguard this work and made all reasonable efforts to ensure it could not be copied.
- No part of this assignment has been previously submitted as part of another unit/course.
- I acknowledge and agree that the assessor of this assignment may for the purposes of assessment, reproduce the assignment and:

i.provide to another member of faculty and any external marker; and/or
i.submit it to a text matching software; and/or
i.submit it to a text matching software which may then retain a copy of the assignment on its database for the purpose of future plagiarism checking.

- I certify that I have not plagiarised the work of others or participated in unauthorised collaboration when preparing this assignment.

Signature ..... *Date*.....

* delete (iii) if not applicable

Signature _____Yunxuan Wang_____Date:___15/09/2021_____ Signature _____Date:_____

Signature _____hongliang Tang_____Date:____15/09/2021_____ Signature _____Date:_____

Signature _____Date:_____ Signature _____Date:_____

MONASH University
Information Technology

Privacy Statement
The information on this form is collected for the primary purpose of assessing your assignment and ensuring the academic integrity requirements of the University are met. Other purposes of collection include recording your plagiarism and collusion declaration, attending to course and administrative matters and statistical analyses. If you choose not to complete all the questions on this form it may not be possible for Monash University to assess your assignment. You have a right to access personal information that Monash University holds about you, subject to any exceptions in relevant legislation. If you wish to seek access to your personal information or inquire about the handling of your personal information, please contact the University Privacy Officer: privacyofficer@adm.monash.edu.au

Contribution Declaration Form
(to be completed by all team members)

Please fill in the form with the contribution from each student towards the assignment.

1 NAME AND CONTRIBUTION DETAILS

Student ID	Student Name	Contribution Percentage
29082498	Yunxuan Wang	50%
27135519	Hongliang Tang	50%

List of parts that each student did:

- Yunxuan Wang:
- Task C.2 (tasks C.*2.1, C.2.2 ,C.2.3,C.2.4*)
- Task C.3 (tasks C.*3.1, C.3.2*)
- Task C.4 (Cassandra Part)

- Hongliang Tang:
- Task C.1 (tasks C.*1.1, C.1.2 ,C.1.3,C.1.4,C.1.5,C.1.6,C.1.7,C.1.8,C1.9*)

2 DECLARATION

We declare that:

- The information we have supplied in or with this form is complete and correct.
- We understand that the information we have provided in this form will be used for individual assessment of the assignment.

3 SIGNATURE

Signatures

Yunxuan Wang

Hongliang Tang

Date

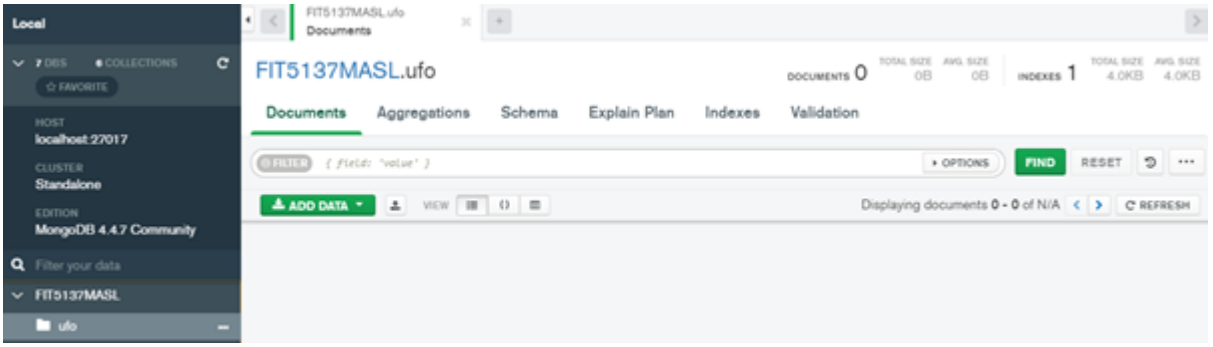
Day Month Year

15/ 09 / 2021

C.1. Analysis using MongoDB

C.1.1.

Output Screenshot:



C.1.2.

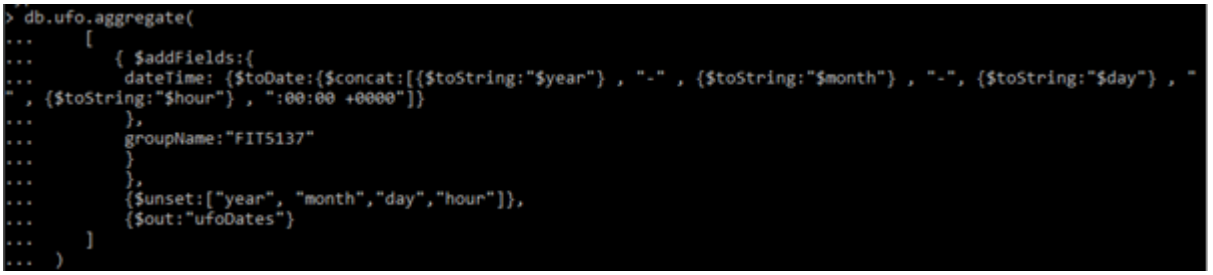
Output Screenshot:

Displaying documents **1 - 20** of 44970


```
_id: ObjectId("60f6b847a5e4e6647265ae6c")
state: "WA"
day: 14
month: 11
year: 1997
hour: 21
shape: "formation"
city: "OLYMPIA"
weatherObs: Object
  windCond: Object
      windchill: -0.08011228
      wdire: "South"
      wspd: -0.357641004
    pressure: -0.163149246
    temp: -1.618530875
    hail: 0
    rain: 0
    vis: -2.461665734
    dewpt: -0.867471751
    thunder: 0
    fog: 0
    tornado: 0
    hum: 1.598872339
    snow: 0
    conds: "Clear"
  countyName: "THURSTON"
  sightingObs: Array
    0: Object
```

C.1.3.

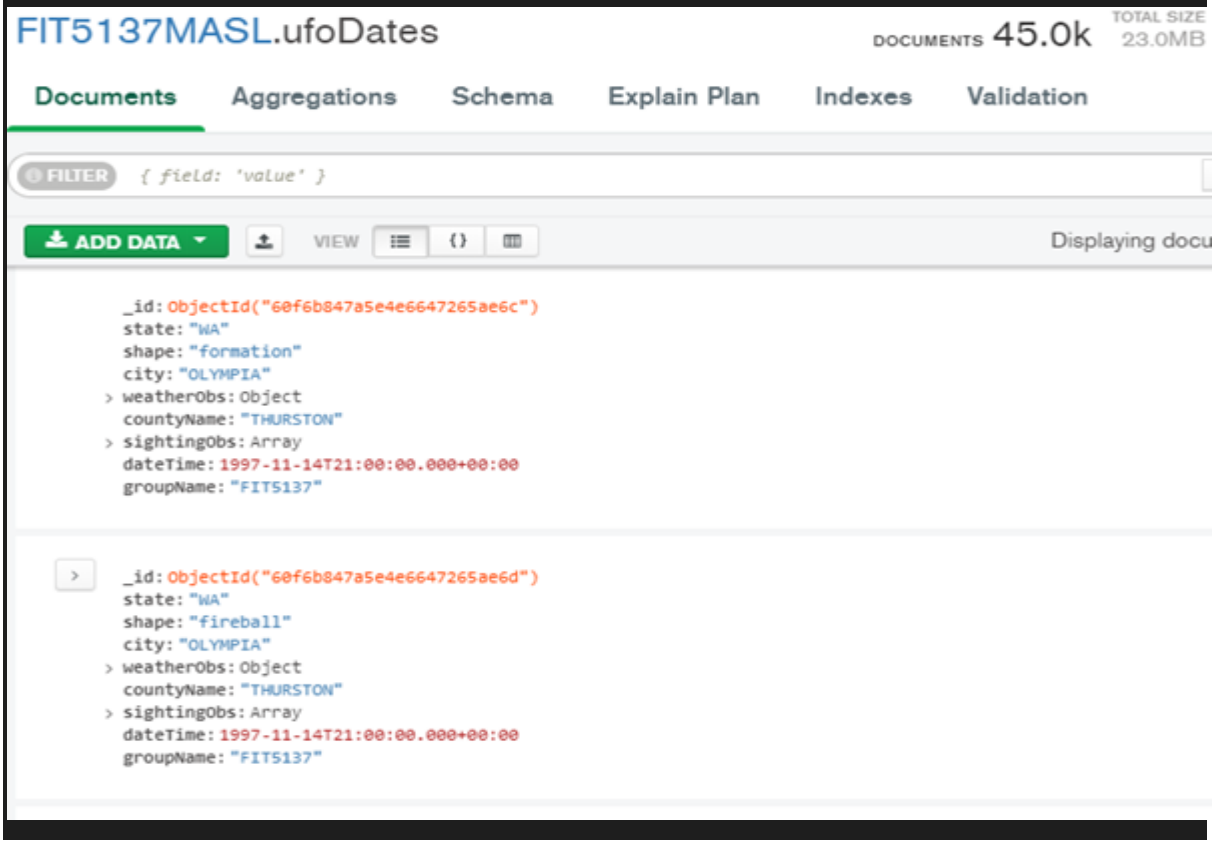
MongoDB command:



```
> db.ufo.aggregate(
...     [
...         { $addFields:{
...          dateTime: {$toDate:{$concat:[{$toString:"$year"} , "-" , {$toString:"$month"} , "-", {$toString:"$day"} , "
" , {$toString:"$hour"} , ":00:00 +0000"]}
...          },
...          groupName:"FIT5137"
...          }
...          },
...          {$unset:["year", "month","day","hour"]},
...          {$out:"ufoDates"}
...     ]
...  )
```

```
db.ufo.aggregate(

  [{

      $addFields: {

        dateTime: {

          $toDate: { $concat: [{ $toString: "$year" }, "-", { $toString: "$month" }, "-", {
$toString: "$day" }, " ", { $toString: "$hour" }, ":00:00 +0000"] }

        },

        groupName: "FIT5137"

      } },

    { $unset: ["year", "month", "day", "hour"] },

    { $out: "ufoDates" }

  ])
```

Output Screenshot:



C.1.4

MongoDB command:

```
> db.ufoDates.insertOne(
...     {
...         state:"MA",
...         shape:"sphere",
...         city:"BOSTON",
...         countyName:"SUFFOLK",
...         dateTime: new Date("1998-07-14T23:00:00.000+00:00"),
...         groupName:"FIT5137",
...         sightingObs:[{duration:"40 min",
...                       text:"I was going to my work on my night shift at the St Albin's hospital and saw an unearthly
ray of shooting lights which could be none other than a UFO!",
...                       summary:"Unearthly ray of shooting lights"}]
...     }
... )
{
        "acknowledged" : true,
        "insertedId" : ObjectId("613e47e64efc36b9b7ba1df0")
}
```

```
db.ufoDates.insertOne(
  {
    state: "MA",
    shape: "sphere",
```

```
    city: "BOSTON",
    countyName: "SUFFOLK",
    dateTime: new Date("1998-07-14T23:00:00.000+00:00"),
    groupName: "FIT5137",
    sightingObs: [{
      duration: "40 min",
      text: "I was going to my work on my night shift at the St Albin's hospital and saw an unearthly ray of shooting lights which could be none other than a UFO!",
      summary: "Unearthly ray of shooting lights"
    }]
  }
)
```

```
_id: ObjectId("613e47e64efc36b9b7ba1df0")
state: "MA"
shape: "sphere"
city: "BOSTON"
countyName: "SUFFOLK"
dateTime: 1998-07-14T23:00:00.000+00:00
groupName: "FIT5137"
sightingObs: Array
  0: Object
      duration: "40 min"
      text: "I was going to my work on my night shift at the St Albin's hospital an..."
      summary: "Unearthly ray of shooting lights"
```

C.1.5.
MongoDB command:

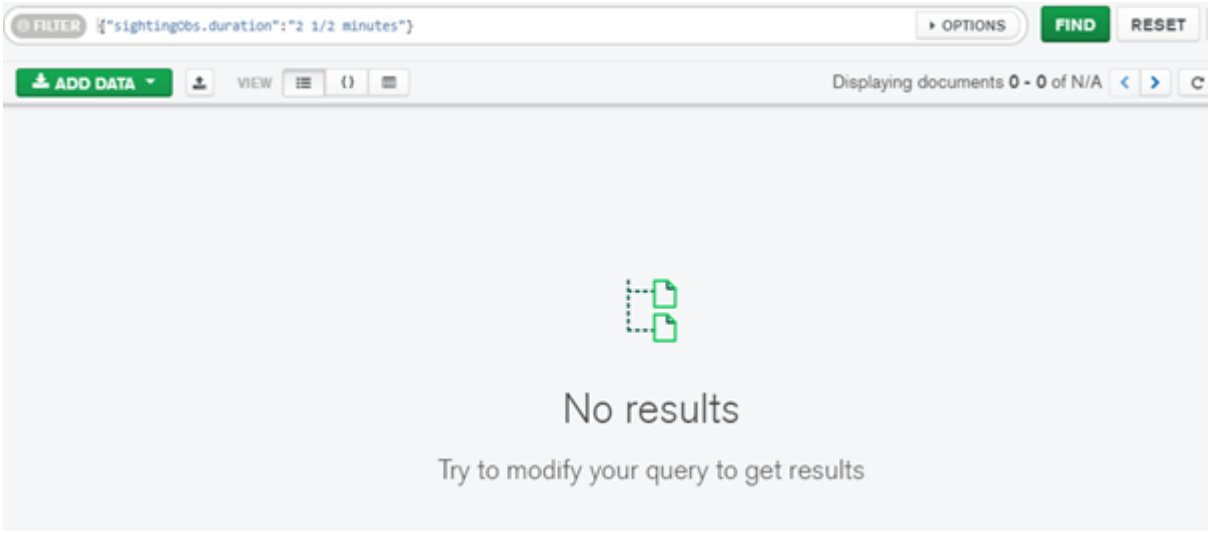
```
> db.ufoDates.update({},
...     {$pull:{sightingObs:{duration:"2 1/2 minutes"}}},
...     {multi:true}
...     )
WriteResult({ "nMatched" : 44971, "nUpserted" : 0, "nModified" : 1 })
```

```
db.ufoDates.update({},
  { $pull: { sightingObs: { duration: "2 1/2 minutes" } } },
  { multi: true }
```

```
)
```

Output Screenshot:

Now, there is no observation with a duration of "2 1/2 minutes"



C.1.6

MongoDB command:

```
> db.ufo.aggregate([
...     {$match:{city:"SAN FRANCISCO",state:"CA",year:{$gte:1990,$lte:2000}}},
...     {$unwind:"$sightingObs"},
...     {$count:"total number of sightings observed"}
... ])
{ "total number of sightings observed" : 16 }
>
```

```
db.ufoDates.aggregate([

  { $match: { city: "SAN FRANCISCO", state: "CA", dateTime: { $gte: new
Date("1990-01-01"), $lt: new Date("2001-01-01") } } },

  { $unwind: "$sightingObs" },

  { $count: "total number of sightings observed" }

])
```

(ii) **MongoDB command:**

```
> db.ufoDates.aggregate([
...     {$match:{shape:"fireball"}},
...     {$group:
...        {_id:null,
...        averageTemp:{$avg:"$weatherObs.temp"},
...        averageHumid:{$avg:"$weatherObs.hum"},
...        averagePressure:{$avg:"$weatherObs.pressure"},
...        averageRainfall:{$avg:"$weatherObs.rain"}
...        }
...     },
...     {$project:{averageTemp:{$round:["$averageTemp",3]},
...     averageHumid:{$round:["$averageHumid",3]},
...     averagePressure:{$round:["$averagePressure",3]},
...     averageRainfall:{$round:["$averageRainfall",3]},
...     _id:0
... }}
... ])
{ "averageTemp" : 0.051, "averageHumid" : 0.077, "averagePressure" : 0.025, "averageRainfall" : 0.019 }
```

```
db.ufoDates.aggregate([

  { $match: { shape: "fireball" } },

  {

    $group:

    {

      _id: null,

      averageTemp: { $avg: "$weatherObs.temp" },

      averageHumid: { $avg: "$weatherObs.hum" },

      averagePressure: { $avg: "$weatherObs.pressure" },

      averageRainfall: { $avg: "$weatherObs.rain" }

    }

  },

  {

    $project: {

      averageTemp: { $round: ["$averageTemp", 3] },

      averageHumid: { $round: ["$averageHumid", 3] },

      averagePressure: { $round: ["$averagePressure", 3] },
```

```
      averageRainfall: { $round: ["$averageRainfall", 3] },

      _id: 0

    }

  }

])
```

(iii) Output:

```
{ "monthName" : "July", "highest number of UFO sightings" : 5526 }
```

MongoDB command:

```
> db.ufoDates.aggregate([
...     { $unwind: "$sightingObs" },
...     {
...        $group: {
...           _id: {
...              month: { $month: "$dateTime" }
...           }, "amount": { $sum: 1 }
...        }
...     },
...     { $sort: { "amount": -1 } },
...     { $limit: 1 },
...     {
...        $project:
...        {
...           "monthName":
...           {
...              $switch:
...              {
...                 branches: [
...                    { case: { $eq: ["$_id.month", 1] }, then: "January" },
...                    { case: { $eq: ["$_id.month", 2] }, then: "February" },
...                    { case: { $eq: ["$_id.month", 3] }, then: "March" },
...                    { case: { $eq: ["$_id.month", 4] }, then: "April" },
...                    { case: { $eq: ["$_id.month", 5] }, then: "May" },
...                    { case: { $eq: ["$_id.month", 6] }, then: "June" },
...                    { case: { $eq: ["$_id.month", 7] }, then: "July" },
...                    { case: { $eq: ["$_id.month", 8] }, then: "August" },
...                    { case: { $eq: ["$_id.month", 9] }, then: "September" },
...                    { case: { $eq: ["$_id.month", 10] }, then: "October" },
...                    { case: { $eq: ["$_id.month", 11] }, then: "November" },
...                    { case: { $eq: ["$_id.month", 12] }, then: "December" },
...                 ],
...                 default: "unknown digit"
...              }
...           },
...           "_id": 0,
...           "highest number of UFO sightings": "$amount"
...        }
...     }
... ])
{ "monthName" : "July", "highest number of UFO sightings" : 5526 }
```

```
db.ufoDates.aggregate([

  { $unwind: "$sightingObs" },
```

```
{
  $group: {
    _id: {
      month: { $month: "$dateTime" }
    }, "amount": { $sum: 1 }
  }
},
{ $sort: { "amount": -1 } },
{ $limit: 1 },
{
  $project:
  {
    "monthName":
    {
      $switch:
      {
        branches: [
          { case: { $eq: ["$_id.month", 1] }, then: "January" },
          { case: { $eq: ["$_id.month", 2] }, then: "February" },
          { case: { $eq: ["$_id.month", 3] }, then: "March" },
          { case: { $eq: ["$_id.month", 4] }, then: "April" },
          { case: { $eq: ["$_id.month", 5] }, then: "May" },
          { case: { $eq: ["$_id.month", 6] }, then: "June" },
```

```
            { case: { $eq: ["$_id.month", 7] }, then: "July" },
            { case: { $eq: ["$_id.month", 8] }, then: "August" },
            { case: { $eq: ["$_id.month", 9] }, then: "September" },
            { case: { $eq: ["$_id.month", 10] }, then: "October" },
            { case: { $eq: ["$_id.month", 11] }, then: "November" },
            { case: { $eq: ["$_id.month", 12] }, then: "December" },
          ],
          default: "unknown digit"
        }
      },
      "_id": 0,
      "highest number of UFO sightings": "$amount"
    }
  }
])
```

(iv) **Output:**

```
{ "_id" : "BLUE", "maxTemp" : 0.925, "minTemp" : -0.801 }
{ "_id" : "GREEN", "maxTemp" : 1.651, "minTemp" : -1.915 }
{ "_id" : "ORANGE", "maxTemp" : 1.099, "minTemp" : -2.313 }
{ "_id" : "RED", "maxTemp" : 2.775, "minTemp" : -2.027 }
{ "_id" : "YELLOW", "maxTemp" : 1.61, "minTemp" : -1.414 }
>
```

MongoDB command:

```
> db.ufoDates.aggregate([
...     {$match:{colour:{$ne:null}}},
...     {$project:{colour:{$split:["$colour"," "]},weatherObs:1}},
...     {$unwind:"$colour"},
...     {$group:{
...        "_id":{$toUpper:"$colour"},
...        maxTemp:{$max:"$weatherObs.temp"},
...        minTemp:{$min:"$weatherObs.temp"}
...     }},
...     {$project:{
...        "_id":1,
...        maxTemp:{$round:["$maxTemp",3]},
...        minTemp:{$round:["$minTemp",3]}
...     }},
...     {$sort:{"_id":1}}
...     ]
... )
{ "_id" : "BLUE", "maxTemp" : 0.925, "minTemp" : -0.801 }
{ "_id" : "GREEN", "maxTemp" : 1.651, "minTemp" : -1.915 }
{ "_id" : "ORANGE", "maxTemp" : 1.099, "minTemp" : -2.313 }
{ "_id" : "RED", "maxTemp" : 2.775, "minTemp" : -2.027 }
{ "_id" : "YELLOW", "maxTemp" : 1.61, "minTemp" : -1.414 }
>
```

```
db.ufoDates.aggregate([

  { $match: { colour: { $ne: null } } },

  { $project: { colour: { $split: ["$colour", " "] }, weatherObs: 1 } },

  { $unwind: "$colour" },

  {

    $group: {

      "_id": { $toUpper: "$colour" },

      maxTemp: { $max: "$weatherObs.temp" },

      minTemp: { $min: "$weatherObs.temp" }

    }

  },

  {
```

```
    $project: {
      "_id": 1,
      maxTemp: { $round: ["$maxTemp", 3] },
      minTemp: { $round: ["$minTemp", 3] }
    }
  },
  { $sort: { "_id": 1 } }
]
)
```

(v) **MongoDB command and output:**

```
> db.ufoDates.aggregate([
...     {$match:{shape:"oval"}},
...     {$group:{"_id":"$weatherObs.windCond.wdire",
...     count:{$sum:1}
...     }},
...     {$sort:{count:-1}},
...     {$limit:1},
...     {$project:{
...        "wind direction":"$_id",
...        count:1,
...        _id:0
...     }}
... ])
{ "count" : 764, "wind direction" : "North" }
```

```
db.ufoDates.aggregate([
  { $match: { shape: "oval" } },
  {
    $group: {
      "_id": "$weatherObs.windCond.wdire",
```

```
      count: { $sum: 1 }
    }
  },
  { $sort: { count: -1 } },
  { $limit: 1 },
  {
    $project: {
      "wind direction": "$_id",
      count: 1,
      _id: 0
    }
  }
])
```

(vi) **MongoDB command and output**

```
> db.ufoDates.createIndex({
...     "sightingObs.text": "text",
...     "sightingObs.summary": "text"
... })
{
        "createdCollectionAutomatically" : false,
        "numIndexesBefore" : 1,
        "numIndexesAfter" : 2,
        "ok" : 1
}
>
```

```
> db.ufoDates.aggregate([
...     {
...         $match: {
...            $text:{$search:"light",$caseSensitive:false}
...         }
...     },
...     {
...         $count:"Number of UFO sighting observations contain the word ‘light’"
...     }
... ])
{ "Number of UFO sighting observations contain the word ‘light’" : 4556 }
```

```
db.ufoDates.createIndex({

  "sightingObs.text": "text",

  "sightingObs.summary": "text"

})



db.ufoDates.aggregate([

  {

    $match: {

      $text: { $search: "light", $caseSensitive: false }

    }

  },

  {

    $count: "Number of UFO sighting observations contain the word ‘light’"

  }

])
```

C.1.7.

MongoDB command and output:

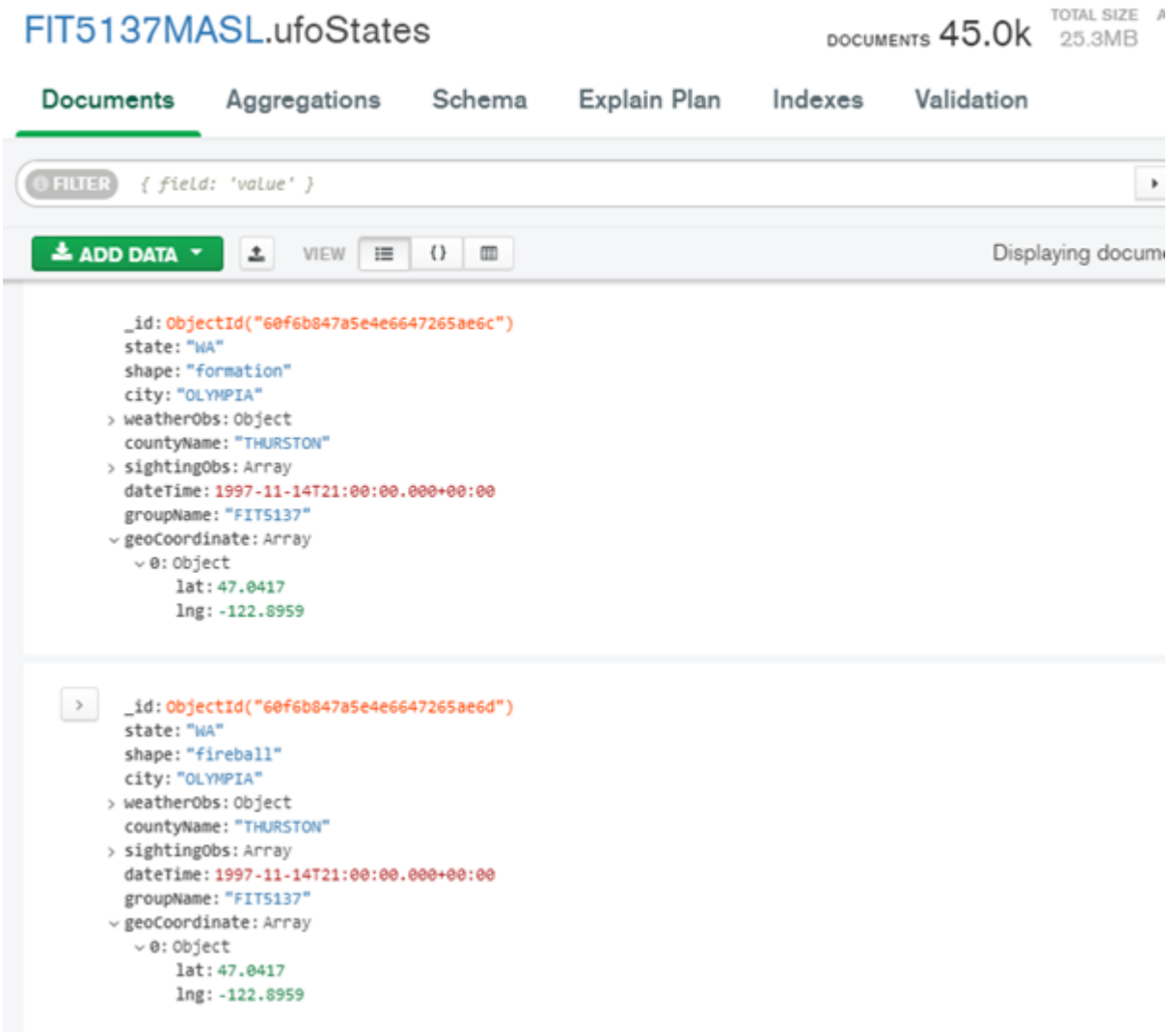
```
> db.ufoDates.aggregate([
...    {
...       $lookup: {
...          from: "states",
...          let: { ufoState: "$state", ufoStateCity: "$city" ,ufoStateCountyName: "$countyName"},
...          pipeline: [
...             {
...                $match:
...                {
...                   $expr:
...                   {
...                      $and:
...                         [
...                            { $eq: ["$countyName", "$$ufoStateCountyName"] },
...                            { $eq: ["$city", "$$ufoStateCity"] },
...                            { $eq: ["$state", "$$ufoState"] }
...                         ]
...                   }
...                }
...             },
...             {
...                $project: {
...                   city: 0,
...                   countyName: 0,
...                   _id: 0,
...                   state: 0
...                }
...             }
...          ],
...          as: "geoCoordinate"
...
...       }
...    },
...    {
...       $out: { db: "FIT5137MASL", coll: "ufoStates" }
...    }
... ])
```

```
> show collections
states
ufo
ufoDates
ufoStates
```

```
db.ufoDates.aggregate([

   {

      $lookup: {

         from: "states",

         let: { ufoState: "$state", ufoStateCity: "$city"
,ufoStateCountyName: "$countyName"},

         pipeline: [

            {

               $match:

               {

                  $expr:
```

```
                {
                    $and:
                        [
                            { $eq: ["$countyName",
"$$ufoStateCountyName"] },
                            { $eq: ["$city", "$$ufoStateCity"] },
                            { $eq: ["$state", "$$ufoState"] }
                        ]
                }
            }
        },
        {
            $project: {
                city: 0,
                countyName: 0,
                _id: 0,
                state: 0
            }
        }
    ],
    as: "geoCoordinate"

  }
},
{
```

```
        $out: { db: "FIT5137MASL", coll: "ufoStates" }

    }

])
```



C.1.8.

MongoDB command and output:

```
> db.ufoStates.aggregate([
...     {
...        $match:{geoCoordinate:{$size:1}}
...     },
...     {
...        $addFields:
...        {
...           "location":
...           {
...              $let: {
...                 vars: {
...                    location: { $first: "$geoCoordinate" }
...                 },
...                 in: {
...                    type: "Point", coordinates: ["$$location.lng", "$$location.lat"]
...                 }
...              }
...           }
...        }
...     },
...     {
...        $project: {
...           "geoCoordinate": 0
...        }
...     },
...     { $out: "ufoStatesGeojson" }
... ])
>
>
```

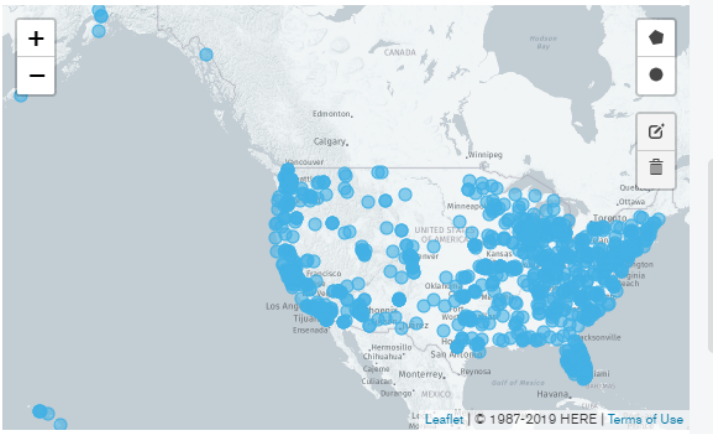
```
db.ufoStates.aggregate([

   {

      $match:{geoCoordinate:{$size:1}}

   },

   {

      $addFields:

      {

         "location":

         {

            $let: {

               vars: {

                  location: { $first: "$geoCoordinate" }

               },

               in: {
```

```
                    type: "Point", coordinates: ["$$location.lng",
"$$location.lat"]

                }

            }

        }

    }

},

{

    $project: {

        "geoCoordinate": 0

    }

},

{ $out: "ufoStatesGeojson" }

])
```

```
_id: ObjectId("60f6b847a5e4e6647265ae6c")
state: "WA"
shape: "formation"
city: "OLYMPIA"
weatherObs: Object
  windCond: Object
  pressure: -0.163149246
  temp: -1.618530875
  hail: 0
  rain: 0
  vis: -2.461665734
  dewpt: -0.867471751
  thunder: 0
  fog: 0
  tornado: 0
  hum: 1.598872339
  snow: 0
  conds: "Clear"
countyName: "THURSTON"
sightingObs: Array
dateTime: 1997-11-14T21:00:00.000+00:00
groupName: "FIT5137"
location: Object
  type: "Point"
  coordinates: Array
    0: -122.8959
    1: 47.0417
```


location

coordinates



C.1.9.

MongoDB command and output:

```
> db.ufoStatesGeojson.createIndex({ "location.coordinates": "2dsphere" })
{
        "createdCollectionAutomatically" : false,
        "numIndexesBefore" : 1,
        "numIndexesAfter" : 2,
        "ok" : 1
}
```

```
db.ufoStatesGeojson.createIndex({ "location.coordinates": "2dsphere" })
```

```
> db.ufoStatesGeojson.aggregate([
..     {
..        $unwind:"$sightingObs"
..     },
..     {
..        $group: {
..           "_id": {
..              "state": "$state",
..              "city": "$city",
..              "countyName": "$countyName"
..           },
..           "sightings":{$sum:1},
..           "location":{$first:"$location"}
..        }
..     },
..     {$sort:{"sightings":-1}},
..     {$limit:1}
.. ])
{ "_id" : { "state" : "AZ", "city" : "PHOENIX", "countyName" : "MARICOPA" }, "sightings" : 347, "location" : { "type" :
"Point", "coordinates" : [ -112.0891, 33.5722 ] } }
```

```
db.ufoStatesGeojson.aggregate([

    {

       $unwind:"$sightingObs"

    },

    {

       $group: {

          "_id": {

             "state": "$state",

             "city": "$city",

             "countyName": "$countyName"

          },

          "sightings":{$sum:1},

          "location":{$first:"$location"}

       }

    },

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

    {$limit:1}

])
```

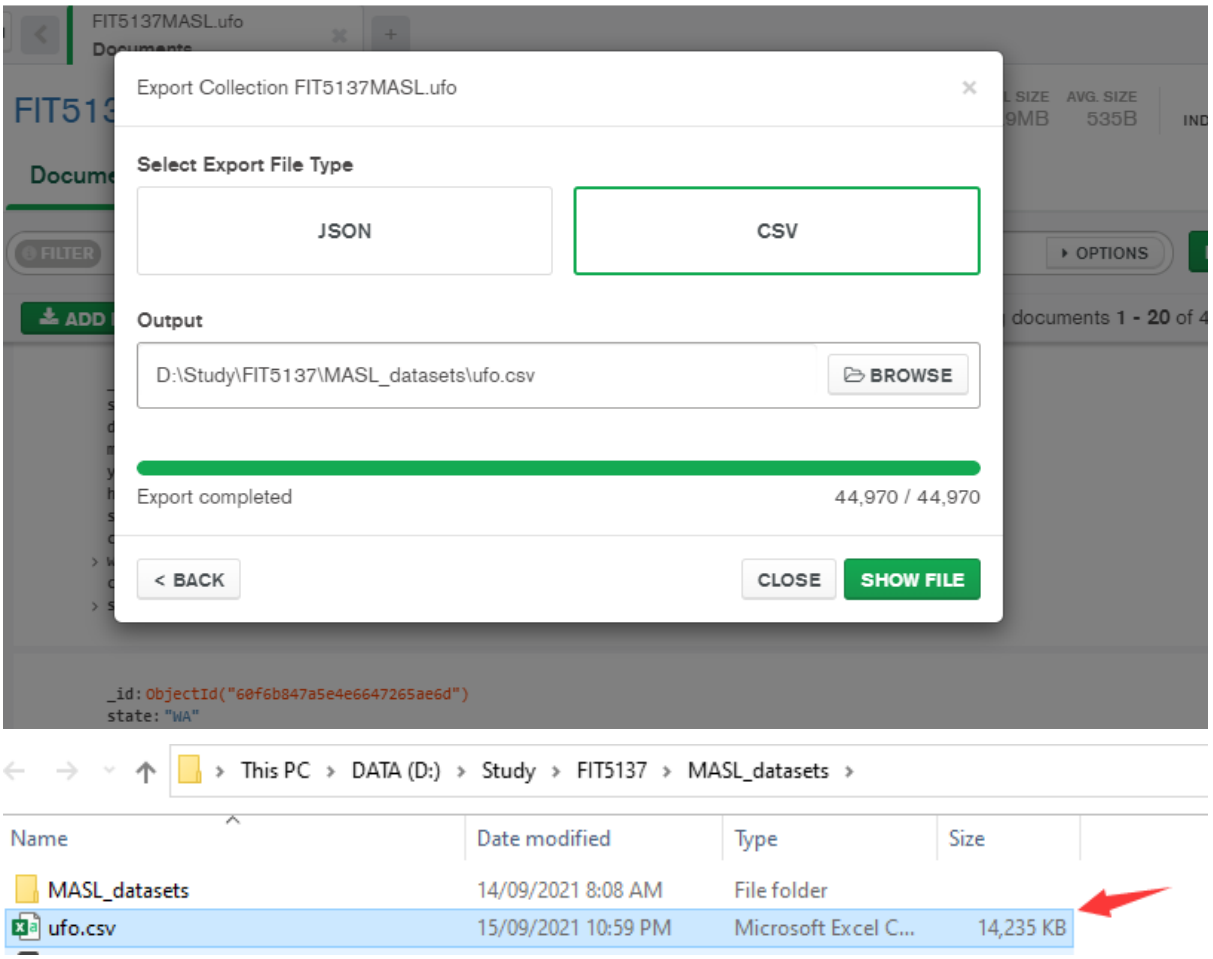
The highest sightings location is [-112.0891,33.5722]

```
> db.ufoStatesGeojson.aggregate([
...     {
...        "$geoNear":
...              {
...                 near:{type:"Point", coordinates:[-112.0891,33.5722]},
...                 distanceField:"dist.calculated",
...                 maxDistance:100000,
...                 minDistance:10000,
...                 includeLocs:"dist.location",
...                 key:"location",
...                 spherical: true
...              }
...     },
...     {
...        $group:{
...           "_id":{
...              "city":"$city"
...           }
...        }
...     },
...     {$sort:{"_id.city":1}}
...
... ])
{ "_id" : { "city" : "ANTHEM" } }
{ "_id" : { "city" : "APACHE JUNCTION" } }
{ "_id" : { "city" : "ARIZONA CITY" } }
{ "_id" : { "city" : "AVONDALE" } }
{ "_id" : { "city" : "BLACK CANYON CITY" } }
{ "_id" : { "city" : "BUCKEYE" } }
{ "_id" : { "city" : "CAREFREE" } }
{ "_id" : { "city" : "CASA GRANDE" } }
{ "_id" : { "city" : "CAVE CREEK" } }
{ "_id" : { "city" : "CHANDLER" } }
{ "_id" : { "city" : "CONGRESS" } }
{ "_id" : { "city" : "COOLIDGE" } }
{ "_id" : { "city" : "EL MIRAGE" } }
{ "_id" : { "city" : "FLORENCE" } }
{ "_id" : { "city" : "FOUNTAIN HILLS" } }
{ "_id" : { "city" : "GILA BEND" } }
{ "_id" : { "city" : "GILBERT" } }
{ "_id" : { "city" : "GLENDALE" } }
{ "_id" : { "city" : "GOLD CANYON" } }
{ "_id" : { "city" : "GOODYEAR" } }
Type "it" for more
```

```
db.ufoStatesGeojson.aggregate([
  {
    "$geoNear":
        {
          near:{type:"Point", coordinates:[-112.0891,33.5722]},
          distanceField:"dist.calculated",
          maxDistance:100000,
          minDistance:10000,
          includeLocs:"dist.location",
          key:"location",
          spherical: true
        }
  },
```

```
  {
    $group:{
      "_id":{
        "city":"$city"
      }
    }
  },
  {$sort:{"_id.city":1}}

])
```

C.1.10.



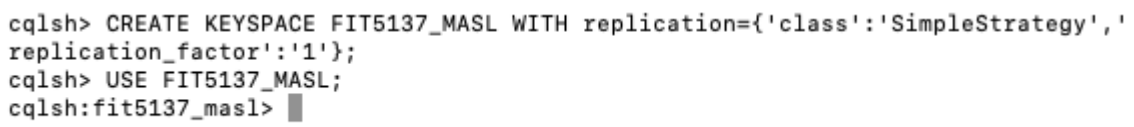
C.2. Analysis using Cassandra

C.2.1

CQL command:

CREATE KEYSPACE FIT5137_MASL WITH
replication={'class':'SimpleStrategy','replication_factor':'1'};
USE FIT5137_MASL;

Output Screenshot:



C.2.2

Explanation: Create ufos table,Set sightingObs as udt,Set month and state as partition key,(day,hour,year,city,countyName,weatherObs_conds,weatherObs_dewpt,weatherObs_fog, weatherObs_hail,weatherObs_heatindex,weatherObs_hum,weatherObs_precip,weatherObs_pressure,weatherObs_rain,weatherObs_snow,weatherObs_temp,weatherObs_thunder,weatherObs_tornado,weatherObs_vis,weatherObs_windCond_wdire,weatherObs_windCond_wgust, weatherObs_windCond_windchill,weatherObs_windCond_wspd) as cluster key to prevent as much data loss as possible.

CQL command:

/*Create table and UDT*/
Create Type *FIT5137_MASL*.sight(
  duration text,
  sighttext text,
  summary text
);
/*sightingObs as UDT*/
CREATE TABLE *FIT5137_MASL*.ufos (id text, city text,colour text,countyName text,day int,hour int,month int,shape text,sightingObs list<frozen<sight>>,
  state text,weatherObs_conds text,

```
    weatherObs_dewpt float,
    weatherObs_fog  int,
    weatherObs_hail int,
    weatherObs_heatindex float,
    weatherObs_hum float,
    weatherObs_precip float,
    weatherObs_pressure float,
    weatherObs_rain int,
    weatherObs_snow int,
    weatherObs_temp  float,
    weatherObs_thunder int,
    weatherObs_tornado  int,
    weatherObs_vis float ,weatherObs_windCond_wdire text,
    weatherObs_windCond_wgust float,
    weatherObs_windCond_windchill float,
    weatherObs_windCond_wspd float,year int
    ,primary
key((month,state),day,hour,year,city,countyName,weatherObs_conds,weatherObs_dewpt,wea
therObs_fog,weatherObs_hail,weatherObs_heatindex,weatherObs_hum,weatherObs_precip,
weatherObs_pressure,weatherObs_rain,weatherObs_snow,weatherObs_temp,weatherObs_thu
nder,weatherObs_tornado,weatherObs_vis,weatherObs_windCond_wdire,weatherObs_wind
Cond_wgust,weatherObs_windCond_windchill,weatherObs_windCond_wspd));

/*import ufo.csv data into ufos table */
copy
FIT5137_MASL.ufos(id,city,colour,countyname,day,hour,month,shape,sightingobs,state,weat
herObs_conds,weatherObs_dewpt,weatherObs_fog,weatherObs_hail,weatherObs_heatindex,
weatherObs_hum,weatherObs_precip,weatherObs_pressure,weatherObs_rain,weatherObs_sn
ow,weatherObs_temp,weatherObs_thunder,weatherObs_tornado,weatherObs_vis,weatherObs
_windCond_wdire,weatherObs_windCond_wgust,weatherObs_windCond_windchill,weather
Obs_windCond_wspd,year) FROM  '/Users/yunxuanwang/Documents/FIT5137/new/ufo.csv'
with delimiter=','  and HEADER = TRUE;
```

Output Screenshot:

```
cqlsh:fit5137_masl> copy FIT5137_MASL.ufos(id,city,colour,countyname,day,hour,mo
nth,shape,sightingobs,state,weatherObs_conds,weatherObs_dewpt,weatherObs_fog,wea
therObs_hail,weatherObs_heatindex,weatherObs_hum,weatherObs_precip,weatherObs_pr
essure,weatherObs_rain,weatherObs_snow,weatherObs_temp,weatherObs_thunder,weathe
rObs_tornado,weatherObs_vis,weatherObs_windCond_wdire,weatherObs_windCond_wgust,
weatherObs_windCond_windchill,weatherObs_windCond_wspd,year) FROM  '/Users/yunxu
anwang/Documents/FIT5137/new/ufo.csv' with delimiter=','  and HEADER = TRUE;
Using 3 child processes

Starting copy of fit5137_masl.ufos with columns [id, city, colour, countyname, d
ay, hour, month, shape, sightingobs, state, weatherobs_conds, weatherobs_dewpt,
weatherobs_fog, weatherobs_hail, weatherobs_heatindex, weatherobs_hum, weatherob
s_precip, weatherobs_pressure, weatherobs_rain, weatherobs_snow, weatherobs_temp
, weatherobs_thunder, weatherobs_tornado, weatherobs_vis, weatherobs_windcond_wd
ire, weatherobs_windcond_wgust, weatherobs_windcond_windchill, weatherobs_windco
nd_wspd, year].
Processed: 44970 rows; Rate:    6424 rows/s; Avg. rate:    6034 rows/s
44970 rows imported from 1 files in 0 day, 0 hour, 0 minute, and 7.453 seconds (
0 skipped).
```

C.2.3

(i)

Explanation: Group by the partition key

CQL command :

select count(sightingObs),month,state from ufos group by month,state allow filtering;

Output Screenshot:

system.count(sightingobs)	month	state
0	4	RI
12	8	CO
9	7	IA
5	4	KS
8	11	MI
8	4	NY
1	11	NM
9	9	TN
17	10	IL
4	10	ID
0	11	MA
4	11	NJ
2	1	ID
49	3	CA
4	10	AK
1	6	AK
12	2	AZ
2	5	MT
4	8	NM
3	7	HI
27	5	PA

(ii)

Explanation: Create an index on weatherObs_conds, so that we can use where condition on it.

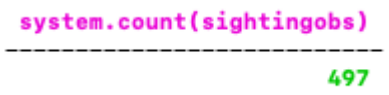
CQL command:

----ii

create index on ufos(weatherObs_conds);

select count(sightingObs) from ufos where weatherObs_conds='Overcast';

Output Screenshot:



(iii)

Explanation: Create an index on year, so that we can use where condition on it.And use sum/count to get the average

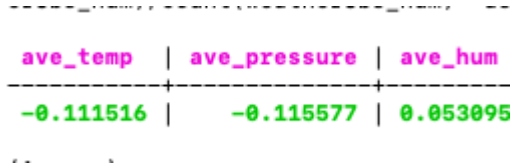
CQL command:

-----iii

create index on ufos(year);

select sum(weatherObs_temp)/count(weatherObs_temp) as Ave_Temp,sum(weatherObs_pressure)/count(weatherObs_pressure) as Ave_Pressure,sum(weatherObs_hum)/count(weatherObs_hum) as Ave_Hum from ufos where year=2000;

Output Screenshot:



C.2.4

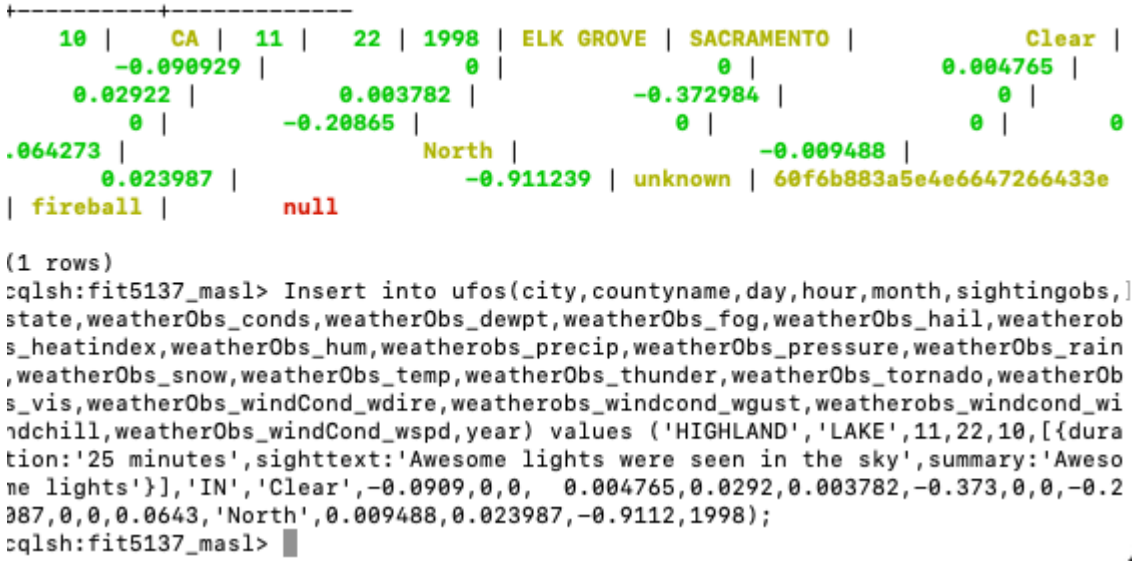
Explanation: Get all the information of the given time at first, and use the weather information, and insert a new record to the table.

CQL command:

select * from ufos where day=11 and hour=22 and month=10 and year=1998 allow filtering;

Insert into ufos(city,countyname,day,hour,month,sightingobs,state,weatherObs_conds,weatherObs_dewpt,weatherObs_fog,weatherObs_hail,weatherobs_heatindex,weatherObs_hum,weatherobs_precip,weatherObs_pressure,weatherObs_rain,weatherObs_snow,weatherObs_temp,weatherObs_thunder,weatherObs_tornado,weatherObs_vis,weatherObs_windCond_wdire,weatherobs_windcond_wgust,weatherobs_windcond_windchill,weatherObs_windCond_wspd,year) values ('HIGHLAND','LAKE',11,22,10,[{duration:'25 minutes',sighttext:'Awesome lights were seen in the sky',summary:'Awesome lights'}],'IN','Clear',-0.0909,0,0, 0.004765,0.0292,0.003782,-0.373,0,0,-0.2087,0,0,0.0643,'North',0.009488,0.023987,-0.9112, 1998);

Output Screenshot:



C.3. Reflections

C.3.1

(i)For mongoDB:

Ebay uses MongoDB to provide search suggestions and quickly process queries. This is done by creating a list of search suggestions in MongoDB and then indexing the data by a series of shipping bureaus such as product categories(Why Business Are Moving Ahead With MongoDB?,2017).Ebay uses MongoDB for data storage search and so on. MongoDB's search capabilities are powerful, and eBay uses it for search recommendations to provide users with effective suggestions.

Shutterfly uses MongoDB to store metadata associated with uploaded photos(Harrison,2011).Shutterfly, a photo-sharing and personal publishing company, manages the persistent storage of a large number of images, because part of its application does not emphasize consistency, and because mongodb does not require consistency of data, it does not require predefined data types, so use mongodb for data management in this section, Unstructured data can be better managed and stored.

Aadhaar uses MongoDB to capture, process, search, and analyze large unstructured datasets and store large amounts of images(Lead,2018).Aadhaar is a personal identification number that can be used as proof of identity and address anywhere in India. Aadhaar chose MongoDB because it is a NoSQL method. Aadhaar uses MongoDB to process unstructured data and store biometric data and images, which are advantages that other data management systems do not have.

For Cassandra:

Uber running Cassandra on Mesos,Uber uses Cassandra to handle heavy loading and processing across data centers((*How Uber Manages a Million Writes Per Second Using Mesos and Cassandra Across Multiple Datacenters - High Scalability* -, 2016b).Uber needed a well-performing and agile database, so it used Cassandra clusters to carry heavy loads and work across data centers, as well as resource isolation and performance isolation between clusters

Netflix uses the Cassandra cluster to service the application, replicating data asynchronously across multiple geographic locations((Finley, 2011).Cassandra has no practical architectural limitations in terms of data size and number of rows/columns. Netflix uses the Cassandra cluster to provide services to applications synchronously for data management, mainly asynchronous replication of data across multiple geographic locations.

Spotify uses Cassandra to store user profile attributes and metadata for playlists, artists, and other entities((*Personalization at Spotify Using Cassandra*, 2015).Cassandra supports cross-site replication, and it can increase storage capacity by increasing the number of nodes in the cluster. Therefore, Spotify chooses Cassandra. Spotify uses Cassandra cluster to store raw data, and uses some algorithms to group these data.

(ii)

Database	**Strength**	**Weakness**
MongoDB	● MongoDB's schema is not predefined,so it handles unstructured data. ● MongoDB supports deep query functions, such as aggregate sorting ● MongoDB supports data redundancy, fault tolerance, and disaster recovery. ● Some query syntax that is easier to learn than SQL ● It is a document-oriented NoSQL database ,so when processing data for storage, it is not bound to the data.	● MongoDB is not suitable for relational data ● MongoDB does not support connections as relational databases, requiring multiple queries to retrieve data from multiple collections. ● Because relationships are not well defined, data replication can make it difficult to work with data sets. ● Duplicate data occupies unnecessary memory space.
Cassandra	● For small queries, the query speed	● Each column must be predefined.

	is fast. ● column-oriented database. ● Each node can service any request, and if one node fails, data can be retrieved from the other nodes. ● Any inserts or updates are written immediately, without the need to read existing data ● Cassandra can be configured for final consistency.	● Data sorting needs to be implemented in a predefined way. ● Joins or subqueries are not supported.So duplicate data needs to be manually maintained with each update or insert. ● There is no special query support. ● A range query on Cassandra can only be performed on a column that is part of a cluster column or on which a secondary index is created.

C.3.2

When we want to add a new record to the database, Cassandra requires that the record we add contains the primary key. If the data we're adding doesn't contain all the primary keys, we can't add them, but MongoDB doesn't need to, because if the record we're adding doesn't contain an _Id, it automatically adds one. Cassandra is more similar to SQL, but it has its limitations. For example, it can only perform group by based on partitioning keys, sometimes some columns need to be indexed to perform queries, and it doesn't have Order Derby, etc. MongoDB aggregation function is more powerful, you can directly use the built-in aggregation function. Therefore, using MongoDB in this aspect is much more convenient and saves a lot of time, because there is no need to spend time selecting partition keys and cluster keys. Because sometimes the partition key and cluster key are not selected correctly, some data will be lost. Cassandra doesn't have its aggregation framework and may need some external tools, and MongoDB is better for unstructured data. Given a JSON file at the beginning, there is no explicit schema definition, so there is no need to design the type of each column, set the primary key, and clean up the data in MongoDB. In this respect, Mongodb is more time-efficient, and MongoDB is more friendly to unstructured data. So MongoDB is better for MASL.

C.4. Connecting to Drivers

List of steps to connect cassandra divers:

1. Install cassandra driver,In the terminal run:

   alias python='python3.9’

   python -m pip install cassandra-driver
2. In the terminal run:
   python
3. Open new terminal run:
   cassandra

Reference

Why Business Are Moving Ahead With MongoDB?. (2017, September 26). Brainvire. https://www.brainvire.com/blog/why-business-are-moving-ahead-with-mongodb/#How_eBay_Implemented_MongoDB

Harrison,G.(2011,January 28).Real World NoSQL: MongoDB at Shutterfly.Gigaom.https://gigaom.com/2011/01/28/real-world-nosql-mongodb-at-shutterfly/

Lead, U. P. R. (2018, October 22). MongoDB: Aadhaar's Original Database Management & Analytics Partner. Unitus Ventures. https://unitus.vc/updates/mongodb-aadhaar-database/

How Uber Manages a Million Writes Per Second Using Mesos and Cassandra Across Multiple Datacenters - High Scalability -. (2016b). High Scalability. http://highscalability.com/blog/2016/9/28/how-uber-manages-a-million-writes-per-second-using-mesos-and.html

Finley, K. (2011, January 28). Why and How Netflix Adopted NoSQL Databases. ReadWrite. https://readwrite.com/2011/01/28/how-netflix-adopted-nosql/

Personalization at Spotify using Cassandra. (2015, January 9). Spotify Engineering. https://engineering.atspotify.com/2015/01/09/personalization-at-spotify-using-cassandra/