

ASSESSMENT COVER SHEET

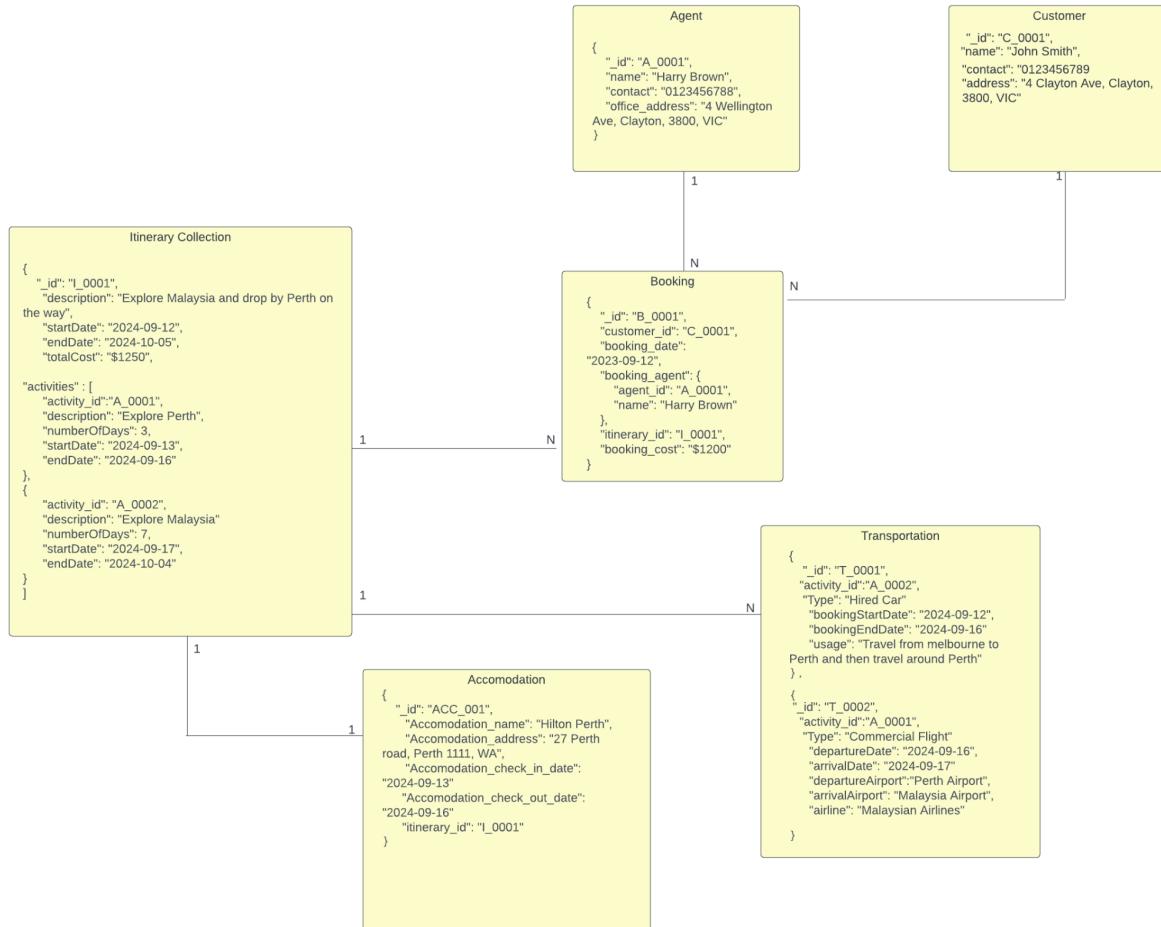
Student ID number	Unit Name and Code: FIT3176 Campus: Clayton Assignment Title: Assignment 1: NoSQL database design - Document-Oriented [MongoDB] (40%) Name of Lecturer: Farah Kabir Name of Tutor: Farah Kabir Tutorial Day and Time: Tuesday 5PM Phone Number: 0405245648 Email Address: Alee0050@student.monash.edu Has any part of this assignment been previously submitted as part of another unit/course? <input type="checkbox"/> Yes <input checked="" type="checkbox"/> No		
Given Name	Due Date:	4th of Oct	Date Submitted:
Family name	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) _____ 4 th of Oct (with application for extension until 7 th _____) Signature of lecturer/tutor _____ Farah Kabir _____ Please note that it is your responsibility to retain copies of your assessments.		
<p><i>Intentional plagiarism or collusion amounts to cheating under Part 7 of the Monash University (Council) Regulations</i></p> <p>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 The material used can be from any source (staff, students or the internet, published and unpublished works).</p> <p>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.</p> <p>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.</p> <p>Student Statement:</p> <ul style="list-style-type: none"> • 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 • 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: <ul style="list-style-type: none"> i. provide to another member of faculty and any external marker; and/or ii. submit it to a text matching software; and/or iii. 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. <p>SignatureAndrew Lee..... Date.....07/10/2023..... <small>* delete (iii) if not applicable</small></p>			

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

TaskA:

Data model for Monash Travel Agency (MTA)



Customer:

localhost:27017

alee0050_TaskA.Customer

0 DOCUMENTS 1 INDEXES

Documents Aggregations Schema Indexes Validation

Filter Type a query: { field: 'value' } Explain Reset Find Options

ADD DATA EXPORT DATA

```
_id: "C_0001"
name: "John Smith"
address: "4 Clayton Ave, Clayton, 3800, VIC"
contact: "0123456789"
```

1-1 of 1 ⏪ ⏩ ⏴ ⏵ ⏷ ⏸ ⏹

My Queries Databases Search alee0050_TaskA...

- CIBarChart
- C2
- C3
- C4
- parks
- wildlife
- alee0050_TaskA
 - Accommodation
 - Agent
 - Booking
 - Customer**
 - Itinerary
 - Transportation
 - booking
- config
- dbtest
- labweek7app
- libDB
- local
- sample_restaurants

Agent:

localhost:27017

alee0050_TaskA.Agent

0 DOCUMENTS 1 INDEXES

Documents Aggregations Schema Indexes Validation

Filter Type a query: { field: 'value' } Explain Reset Find Options

ADD DATA EXPORT DATA

```
_id: "A_0001"
name: "Harry Brown"
contact: "0123456789"
office_address: "4 Wellington Ave, Clayton, 3800, VIC"
```

1-1 of 1 ⏪ ⏩ ⏴ ⏵ ⏷ ⏸ ⏹

My Queries Databases Search alee0050_TaskA...

- CIBarChart
- C2
- C3
- C4
- parks
- wildlife
- alee0050_TaskA
 - Accommodation
 - Agent**
 - Booking
 - Customer
 - Itinerary
 - Transportation
 - booking
- config
- dbtest
- labweek7app
- libDB
- local
- sample_restaurants

Booking:

localhost:27017

alee0050_TaskA.Booking

0 DOCUMENTS 1 INDEXES

Documents Aggregations Schema Indexes Validation

Filter Type a query: { field: 'value' }

ADD DATA **EXPORT DATA**

```
_id: "B_0001"
customer_id: "C_0001"
booking_date: "2023-09-12"
booking_agent: Object
itinerary_id: "I_0001"
booking_cost: "$1200"
```

1-1 of 1

My Queries Databases Search alee0050_TaskA... alee0050_TaskA... +

- ClBarChart
- C2
- C3
- C4
- parks
- wildlife
- alee0050_TaskA
 - Accommodation
 - Agent
 - Booking**
 - Customer
 - Itinerary
 - Transportation
 - booking
 - config
 - dbtest
 - labweek7app
 - libDB
 - local
 - sample_restaurants

Itinerary:

localhost:27017

alee0050_TaskA.Itinerary

0 DOCUMENTS 1 INDEXES

Documents Aggregations Schema Indexes Validation

Filter Type a query: { field: 'value' }

ADD DATA **EXPORT DATA**

```
_id: "I_0001"
description: "Explore Malaysia and drop by Perth on the way"
startDate: "2024-09-12"
endDate: "2024-10-05"
totalCost: "$1250"
activities: Array (2)
  ▾ 0: Object
    activity_id: "A_0001"
    description: "Explore Perth"
    numberOfDays: 3
    startDate: "2024-09-13"
    endDate: "2024-09-16"
  ▾ 1: Object
    activity_id: "A_0002"
    description: "Explore Malaysia"
    numberOfDays: 7
    startDate: "2024-09-17"
    endDate: "2024-10-04"
```

1-1 of 1

My Queries Databases Search alee0050_TaskA... alee0050_TaskA... +

- ClBarChart
- C2
- C3
- C4
- parks
- wildlife
- alee0050_TaskA
 - Accommodation
 - Agent
 - Booking
 - Customer
 - Itinerary**
 - Transportation
 - booking
 - config
 - dbtest
 - labweek7app
 - libDB
 - local
 - sample_restaurants

Accommodation:

localhost:27017 ...

My Queries Databases Search alee0050_TaskA...

alee0050_TaskA.Accommodation

Documents Aggregations Schema Indexes Validation

Filter Type a query: { field: 'value' } Explain Reset Find Options ↗

ADD DATA EXPORT DATA ↗

1-1 of 1 ⏪ ⏩ ⏴ ⏵ ⏷ ⏸ ⏹

```
_id: "ACC_001"
Accommodation_name: "Hilton Perth"
Accommodation_address: "27 Perth road, Perth 1111, WA"
Accommodation_check_in_date: "2024-09-13"
Accommodation_check_out_date: "2024-09-16"
itinerary_id: "I_0001"
```

0 DOCUMENTS 1 INDEXES

Accommodation ...

- CIBarChart
- C2
- C3
- C4
- parks
- wildlife
- alee0050_TaskA
 - Accommodation ...
 - Agent
 - Booking
 - Customer
 - Itinerary
 - Transportation
 - booking
 - config
 - dbtest
 - labweek7app
 - libDB
 - local
 - sample_restaurants

Transportation:

localhost:27017 ...

My Queries Databases Search alee0050_TaskA...

alee0050_TaskA.Transportation

Documents Aggregations Schema Indexes Validation

Filter Type a query: { field: 'value' } Explain Reset Find Options ↗

ADD DATA EXPORT DATA ↗

1-2 of 2 ⏪ ⏩ ⏴ ⏵ ⏷ ⏸ ⏹

```
_id: "T_0001"
activity_id: "A_0002"
Type: "Hired Car"
bookingStartDate: "2024-09-12"
bookingEndDate: "2024-09-16"
usage: "Travel from melbourne to Perth and then travel around Perth"
```

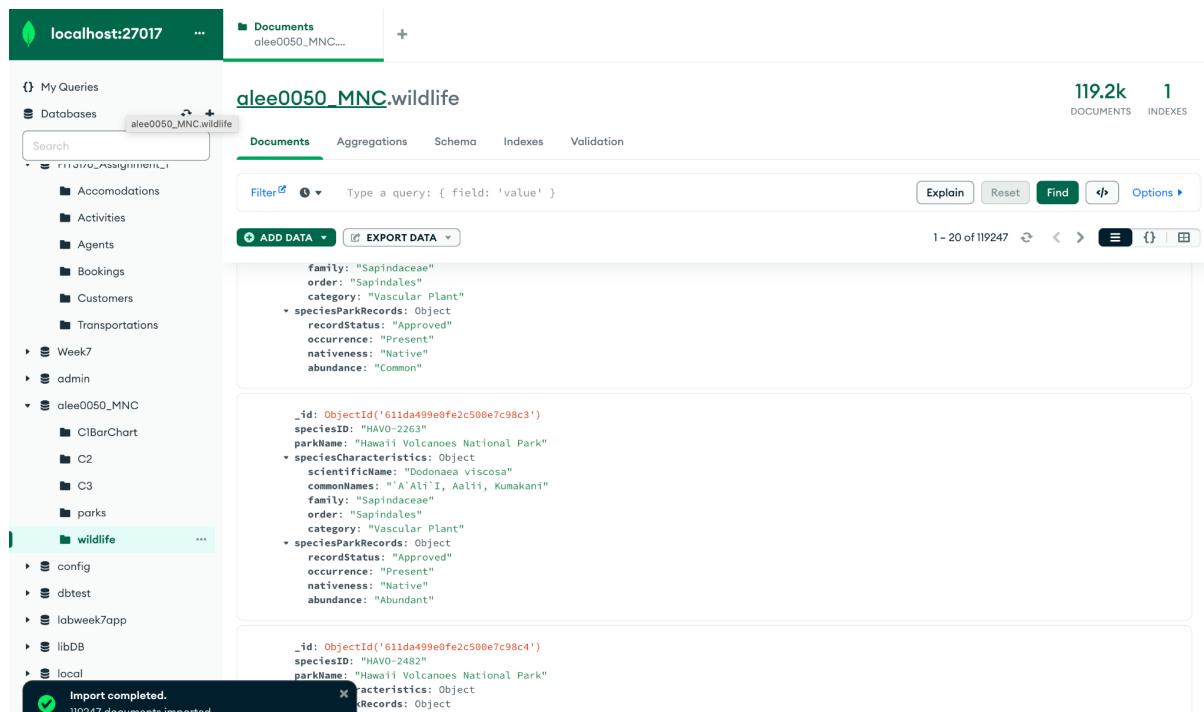
```
_id: "T_0002"
activity_id: "A_0001"
Type: "Commercial Flight"
departureDate: "2024-09-16"
arrivalDate: "2024-09-17"
departureAirport: "Perth Airport"
arrivalAirport: "Malaysia Airport"
airline: "Malaysian Airlines"
```

0 DOCUMENTS 1 INDEXES

Transportation ...

- CIBarChart
- C2
- C3
- C4
- parks
- wildlife
- alee0050_TaskA
 - Accommodation
 - Agent
 - Booking
 - Customer
 - Itinerary
 - Transportation ...
 - booking
 - config
 - dbtest
 - labweek7app
 - libDB
 - local
 - sample_restaurants

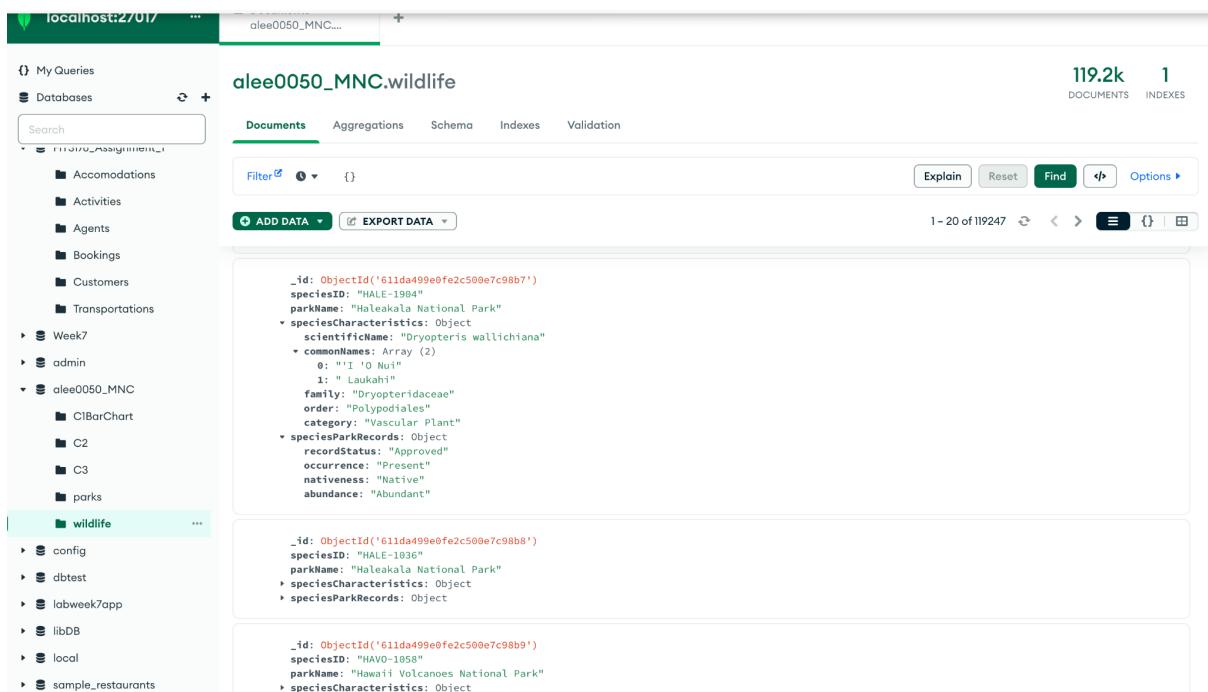
Task B1.1:



The screenshot shows the MongoDB Compass interface connected to the database 'alee0050_MNC.wildlife'. The 'wildlife' collection is selected. The interface includes a sidebar with various collections like Accommodations, Activities, Agents, Bookings, Customers, Transportations, Week7, admin, and alee0050_MNC. The main area shows a document with fields such as family, order, category, speciesParkRecords, recordStatus, occurrence, nativeness, and abundance. Below it, another document is shown with similar fields. At the bottom, a log message indicates 'Import completed. 119247 documents inserted.'

```
db.wildlife.updateMany( //updateMany just modifies all documents
  {"speciesCharacteristics.commonNames": {$regex: ","}}, //regular expression
  //is a form of pattern matching to check if the token is equals to ,
  [
    {
      $set: { // $set operator is used to set the data, in this context the
            // the subarray of splitted commas, will be displayed here
        "speciesCharacteristics.commonNames": {
          $split: ["$speciesCharacteristics.commonNames", ","]
        }
      }
    }
  ]
)
```

```
alee0050_MNC> db.wildlife.updateMany( //updateMany just modifies all documents
...     {"speciesCharacteristics.commonNames": {$regex: ","}}, //regular expression
...     //is a form of pattern matching to check if the token is equals to ,
...     [
...         {
...             $set: { // $set operator is ussed to set the data, in this context the
...                   //the subarray of splitted commas, will be displayed here
...                   "speciesCharacteristics.commonNames": {
...                       $split: ["$speciesCharacteristics.commonNames", ","]
...                   }
...             }
...         }
...     ]
... )
{
    acknowledged: true,
    insertedId: null,
    matchedCount: 21611,
    modifiedCount: 21611,
    upsertedCount: 0
}
```



The screenshot shows the MongoDB Compass interface. On the left, there's a sidebar with a tree view of databases and collections. The 'wildlife' collection is selected. The main area displays three documents from the 'wildlife' collection. The first document is for the species *Dryopteris wallichiana* at Haleakala National Park. The second document is for the family *Dryopteridaceae* at Haleakala National Park. The third document is for the order *Polypodiales* at Hawaii Volcanoes National Park.

Task B1.2:

```
db.wildlife.find().forEach(function(ts) { //iterate through each of the documents
    db.wildlife.updateMany( //update all for each of the documents taking in the id
        { _id: ts._id }, // _id reference to the each of the documents id in wildlife
        { $set: { ts: ts._id.getTimestamp() } } //get the timestamp from the id
    );
    print("Timestamp: " + ts._id.getTimestamp()); //print that out, debugging purpose
});
```

Using print statements, we are able to accurately display what is being printed, and being set under `ts`

The screenshot shows a MongoDB interface with the following details:

- Header:** localhost:27017
- Sidebar:** My Queries, Databases, Search bar, and a tree view of databases:
 - Accommodations
 - Activities
 - Agents
 - Bookings
 - Customers
 - Transportations
 - Week7
 - admin
 - alee0050_MNC
 - C1BarChart
 - C2
 - C3
 - parks
 - wildlife
 - config
 - dbtest
 - labweek7app
 - libDB
 - local
 - sample_restaurants
- Current Database:** alee0050_MNC.wildlife
- Document List:** Shows three documents from the 'wildlife' collection:
 - _id:** ObjectId('611da499e0fe2c500e7c98b2')
speciesID: "HAVO-2230"
parkName: "Hawaii Volcanoes National Park"
speciesCharacteristics: Object
 - scientificName:** "Purera glabra"
 - commonNames:** "Puhe"
 - family:** "Urticaceae"
 - order:** "Rosales"
 - category:** "Vascular Plant"
 - speciesParkRecords:** Object
 - recordStatus:** "Approved"
 - occurrence:** "Present"
 - nativity:** "Native"
 - abundance:** "Uncommon"**ts:** 2021-08-19T08:23:53.000+00:00
 - _id:** ObjectId('611da499e0fe2c500e7c98b3')
speciesID: "HALE-1055"
parkName: "Haleakala National Park"
speciesCharacteristics: Object
speciesParkRecords: Object
ts: 2021-08-19T08:23:53.000+00:00
 - _id:** ObjectId('611da499e0fe2c500e7c98b4')
speciesID: "HALE-1901"
parkName: "Haleakala National Park"
speciesCharacteristics: Object
speciesParkRecords: Object
ts: 2021-08-19T08:23:53.000+00:00
- Top Right:** Metrics (119.2k DOCUMENTS, 1 INDEXES), Explain, Reset, Find, Options buttons, and a navigation bar.

Task B1.3:



localhost:27017

alee0050_MNC.parks

56 DOCUMENTS 1 INDEXES

Documents Aggregations Schema Indexes Validation

Filter Explain Reset Find Options ▾

ADD DATA EXPORT DATA

1 - 20 of 56

```
_id: ObjectId('611da00be0fe2c500e7c987a')
areaInAcres: 47390
latitude: 44.35
longitude: -68.21
parkName: "Acadia National Park"
stateCode: "ME"
dateEst: 1919-02-26T00:00:00.000+00:00
```

```
_id: ObjectId('611da00be0fe2c500e7c987b')
areaInAcres: 76519
latitude: 38.68
longitude: -109.57
parkName: "Arches National Park"
stateCode: "UT"
dateEst: 1971-11-12T00:00:00.000+00:00
```

```
_id: ObjectId('611da00be0fe2c500e7c987c')
areaInAcres: 242756
latitude: 43.75
longitude: -102.5
parkName: "Badlands National Park"
stateCode: "SD"
dateEst: 1978-11-10T00:00:00.000+00:00
```

```
_id: ObjectId('611da00be0fe2c500e7c987d')
s: 801163
t: 29.25
```

Import completed.
56 documents imported.

Task B1.4:

localhost:27017

alee0050_MNC.wildlife

119.2k DOCUMENTS 1 INDEXES

Documents Aggregations Schema Indexes Validation

Filter Explain Reset Find Options ▾

ADD DATA EXPORT DATA

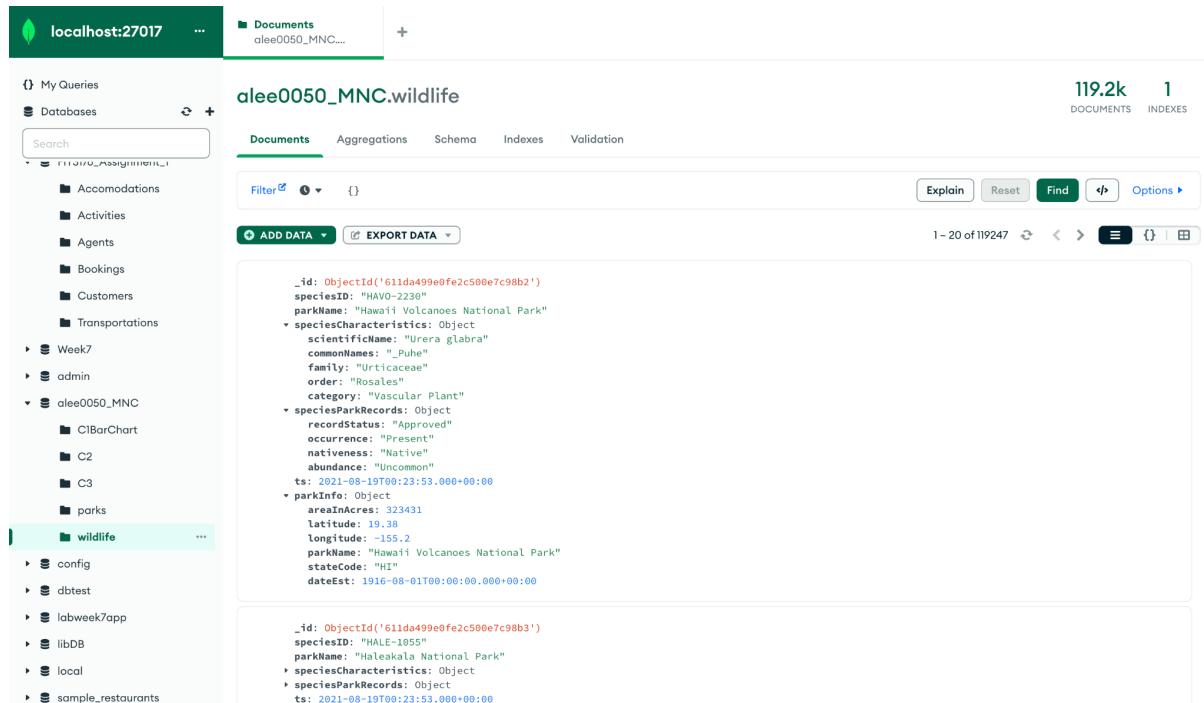
1 - 20 of 119247

```
_id: ObjectId('611da499e0fe2c500e7c98b2')
speciesID: "HAVO-2238"
parkName: "Hawaii Volcanoes National Park"
speciesCharacteristics: Object
scientificName: "Urera glabra"
commonNames: "Puhe"
family: "Urticaceae"
order: "Rosales"
category: "Vascular Plant"
speciesParkRecords: Object
recordStatus: "Approved"
occurrence: "Present"
nativeness: "Native"
abundance: "Uncommon"
ts: 2021-08-19T00:23:53.000+00:00
```

```
_id: ObjectId('611da499e0fe2c500e7c98b3')
speciesID: "HALE-1055"
parkName: "Haleakala National Park"
speciesCharacteristics: Object
speciesParkRecords: Object
ts: 2021-08-19T00:23:53.000+00:00
```

```
_id: ObjectId('611da499e0fe2c500e7c98b4')
speciesID: "HALE-1981"
parkName: "Haleakala National Park"
speciesCharacteristics: Object
speciesParkRecords: Object
ts: 2021-08-19T00:23:53.000+00:00
```

```
db.wildlife.aggregate([
  {
    $lookup: {
      from: "parks", //using the collection parks
      localField: "parkName", //this is like a foreign key concept
      foreignField: "parkName", //using left join needs a foreign key
      as: "parkInfo" //save that as parkinfo, this is saved manually as an array
    }
  },
  {
    $unwind: "$parkInfo" //now we use $unwind to deconstruct an array field to output document for each element
  },
  {
    $addFields: [
      "areaInAcres": "$parkInfo.areaInAcres",
      "latitude": "$parkInfo.latitude",
      "longitude": "$parkInfo.longitude",
      "parkName": "$parkInfo.parkName",
      "dateEst": "$parkInfo.dateEst",
      "stateCode": "$parkInfo.stateCode"
    ]
  },
  {
    $project: {
      "parkInfo._id": 0, //exclude the id in the array that contains park collection of ._id
      "areaInAcres": 0,
      "latitude": 0,
      "longitude": 0,
      "dateEst": 0,
      "stateCode": 0
    }
  },
  {
    $out: "wildlife" //using $out the results of aggregation of previous pipeline is saved into a new collection, or replaced.
  }
]);
```



The screenshot shows the MongoDB Compass interface. The top navigation bar includes 'localhost:27017', 'Documents', and a search bar. The main area displays the 'alee0050_MNC.wildlife' collection. The document count is 119.2k documents and 1 index. The left sidebar shows a tree view of databases and collections, with 'wildlife' currently selected. The right panel shows two documents in a table format.

_id	speciesID	parkName	speciesCharacteristics	speciesParkRecords	parkInfo
<code>_id: ObjectId('611da499e0fe2c500e7c98b2')</code>	"HAVO-2230"	"Hawaii Volcanoes National Park"	Object	Object	Object
<code>_id: ObjectId('611da499e0fe2c500e7c98b3')</code>	"HALE-1055"	"Haleakala National Park"	Object	Object	Object

Task B1.5:

localhost:27017 ... +

Validation alee0050_MNC....

My Queries Databases Search

alee0050_MNC.wildlife

Documents Aggregations Schema Indexes Validation

119.2k 1

DOCUMENTS INDEXES

Accommodations Activities Agents Bookings Customers Transportations

Week7 admin

alee0050_MNC

- CIBarChart
- C2
- C3
- parks

wildlife

- config
- dbtest
- labweek7app
- libDB
- local
- sample_restaurants

Add validation rules

Create rules to enforce data structure of documents on updates and inserts.

Add Rule Learn more about validations

```

db.runCommand({ //it uses the current DB to run whatever logic that is used below
  collMod: "wildlife", //collMod is used to validate
  validator: {
    $jsonSchema: {
      bsonType: "object",
      required: ["speciesParkRecords"],
      properties: {
        speciesParkRecords: {
          bsonType: "object",
          required: ["recordStatus", "occurrence", "nativeness", "abundance"],
          properties: {
            recordStatus: {
              bsonType: "string",
              enum: [/* */],
              'American Crow',
              'Bluebell',
              'Bushtit',
              'Cabezon',
              'Catbird',
              'Cenizo',
              'Chico',
              'Claret Cup',
              'Clover Bush',
              'Cocodrilo De Tumbes',
              'Common Mullein',
              'Common Poorwill',
              'Cranebill',
              'Dames Rocket',
              'Devil's Shoelaces',
              'Downy Chess',
              'Filaree',
              'Fringed Sage',
              'Golden Pea',
              'Goosefoot',
              'Grass-Leaf Loco',
              'Ground Daisy',
              'Kinnikinnick',
              'Leather Flower',
              'Liver Leaf*',
              'Manati',
              'Northern Goshawk',
              'Northern Pintail',
              'Osha',
              'Pigeon Hawk',
              'Purple Cockle',
              'Ranchers' Fireweed',
              'Robin',
              'Rushpink',
              'Shadbush',
              'Short-Tailed Weasel',
              'Skunkbush',
              'Skyrocket Gilia',
              'Speckled Trout',
              'Speedwell',
              'Storksbill',
              'Verdolagas',
              'Wapiti',
              'White-Footed Mouse',
              'Whortleberry',
              'Wild Iris',
              'Wild Rose',
              'Willowherb',
              'Wiregrass',
              'Approved',
              'In Review',
              'None',
              'P.Nut Sedge'],
              description: "recordStatus not valid"
            },
            occurrence: {
              bsonType: "string",
              enum: ['Approved',
              'In Review',
              'Not Confirmed',
              'Not Present',
              'Not Present (False Report)',
              'Not Present (Historical Report)',
              'Present'],
              description: "occurrence not valid"
            },
            nativeness: {
              bsonType: "string",
              enum: [/* */]
            }
          }
        }
      }
    }
  }
}
  
```

```

    "nateness": {
      "bsonType": "string",
      "enum:": ['Native',
      'Not Confirmed',
      'Not Native',
      'Present',
      'Unknown'],
      "description": "nateness not valid"
    },
    "abundance": {
      "bsonType": "string",
      "enum": ['Abundant',
      'Common',
      'Native',
      'Not Native',
      'Occasional',
      'Rare',
      'Uncommon',
      'Unknown'],
      "description": "abundance not valid"
    }
  },
  "validationLevel": "moderate",
  "validationAction": "error"
}

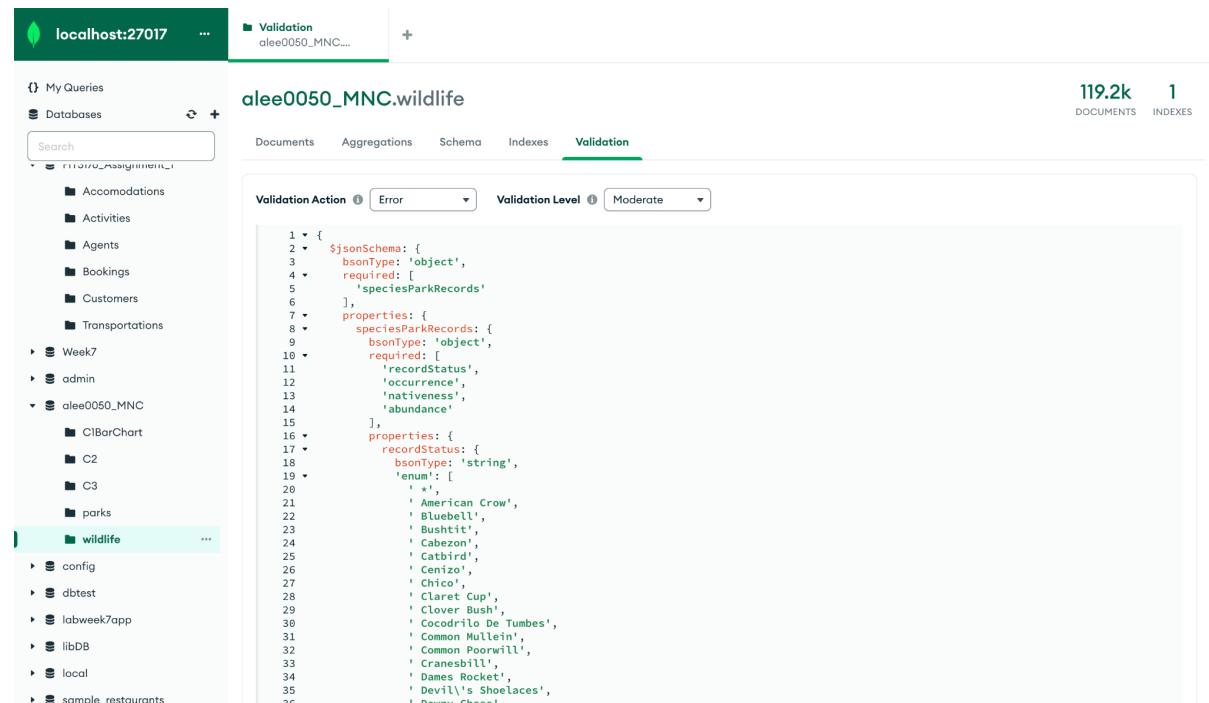
```

Directly copied pasted the enum from command prompt, excuse the formatting.

```

...
}
},
validationLevel: "moderate",
validationAction: "error"
})
{ ok: 1 }

```



The screenshot shows the MongoDB Compass interface. At the top, it says "localhost:27017" and "Validation alee0050_MNC....". Below that, there's a navigation bar with "My Queries", "Databases", "Search", and tabs for "Documents", "Aggregations", "Schema", "Indexes", and "Validation". The "Validation" tab is selected. On the left, there's a sidebar with a tree view of databases and collections, including "wildlife" which is currently selected. The main area displays the validation schema:

```

1 <= {
2   "$jsonSchema": [
3     "bsonType": "object",
4     "required": [
5       "speciesParkRecords"
6     ],
7     "properties": {
8       "speciesParkRecords": {
9         "bsonType": "object",
10      "required": [
11        "recordStatus",
12        "occurrence",
13        "nateness",
14        "abundance"
15      ],
16      "properties": {
17        "recordStatus": {
18          "bsonType": "string",
19          "enum": [
20            "x",
21            "American Crow",
22            "Bluebell",
23            "Bushtit",
24            "Cabezon",
25            "Catbird",
26            "Cenizo",
27            "Chico",
28            "Claret Cup",
29            "Clover Bush",
30            "Cocodrilo De Tumbes",
31            "Common Mullein",
32            "Common Poorwill",
33            "Cranesbill",
34            "Dames Rocket",
35            "Devil's Shoelaces",
36            "Downy Chess"
37          ]
38        }
39      }
40    }
41  }
42}

```

Run db.wildlife.distinct('speciesParkRecords.recordStatus') or any other 3 to find out all the enumerated data types, in order to implement validation.

B2.1:

```
db.wildlife.aggregate([
  {
    $group: {
      _id: {$year: "$parkInfo.dateEst"}, //since we already moved dateEst into parkinfo
      totalParks: {$sum: 1} //1 for true to gather sum of all documents in group
    }
  },
  {
    $sort: {_id: -1} //sort by descending order, this indicates size() - 1, which is the last position
  },
  {
    $project: [
      year: "$_id", //display what year in the grouping
      totalParks: 1, //display totalparks
      _id: 0 //no need for id for the year, since year is already there
    ]
  }
])

```

B2.2:

```
//B2.2
db.wildlife.aggregate([
  {
    $match: {
      "speciesCharacteristics.category": "Bird"
      //match the speciescharacteristics category
      //it can be found using db.wildlife.distinct("speciesCharacteristics.category")
      //this will show all the categories that is in db wildlife
    }
  },
  {
    $group: {
      _id: "$parkInfo.parkName", //group them up by the parkName
      birdCount: {$sum: 1} //count all the birds by adding one incremental, this would indicate for each document it iterates through
      //add 1 to a counter
    }
  },
  {
    $project: {
      parkName: "$_id", //display id as its parkName
      birdCount: 1, //display birdCount
      _id: 0 //don't display ID as there is no need
    }
  }
])

```

B2.3:

```
//B2.3
db.wildlife.updateMany({}, [ //given by the question that we are able to modify our collection

  {
    $set: { //I have opted in to create a field in parkInfo array of GeoJsonLocation
      "parkInfo.GeoJsonLocation": {
        type: "Point", //it is a point, of which longitude and latitude of each record is stored
        coordinates: ["$parkInfo.longitude", "$parkInfo.latitude"]
      }
    }
  }
]

db.wildlife.createIndex({"parkInfo.GeoJsonLocation": "2dsphere"}); //then we create an index for this "2dsphere"
//this will allow us to calculate the distance gathered at one point, and do calculations with radian to find out the distance in one point

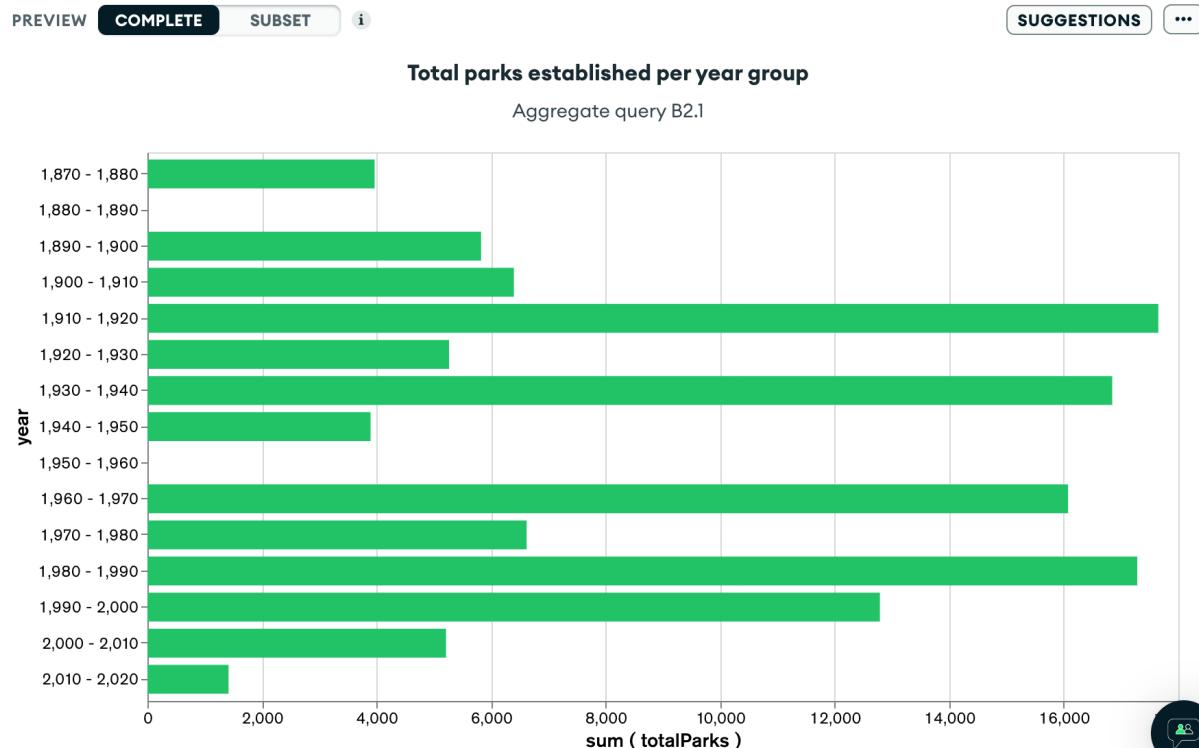
//https://stackoverflow.com/questions/38112376/find-closest-entries-in-mongodb-to-coordinates
db.wildlife.find({
  "parkInfo.GeoJsonLocation": {
    $nearSphere: {
      $geometry: {
        type: "Point",
        coordinates: [-95.8412, 42.1109] //couldn't get this to work
        //using the co-ordinates from example lab, and stack overflow for the query
      },
      $maxDistance: 400000 // 400km is the max distance from point of coords
    }
  }
});
});
```

B2.4:

B2.5:

```
db.wildlife.aggregate([
  {
    $match: {
      "speciesCharacteristics.commonNames": /^a/i //matches everything that starts with "a" case insensitive
    }
  },
  {
    $group: { //group them with parkName, category, order and family
      _id: [
        park: "$sparkName",
        category: "$speciesCharacteristics.category",
        order: "$speciesCharacteristics.order",
        family: "$speciesCharacteristics.family"
      ]
    }
  },
  {
    //https://stackoverflow.com/questions/16368638/mongodb-distinct-aggregation
    //"$addToSet is used to count the number of distinct objects"
    $group: {
      _id: "$_id.park",
      distinctCat: { $addToSet: "$_id.category" }, //addToSet functions like a hashset, where it takes in distinct values, if its duplicate
      //it will discard
      distinctOrder: { $addToSet: "$_id.order" },
      distinctFamily: { $addToSet: "$_id.family" }
    }
  },
  {
    $project: {
      "distinctCatCounter": { "$round": [{ "$size": "$distinctCat" }, 2] }, //round to 2 decimal places
      "distinctOrderCounter": { "$round": [{ "$size": "$distinctOrder" }, 2] }, //size indicates the size of the
      "distinctFamilyCounter": { "$round": [{ "$size": "$distinctFamily" }, 2] }
    }
  }
]).pretty();
```

C.1.1:



Justification:

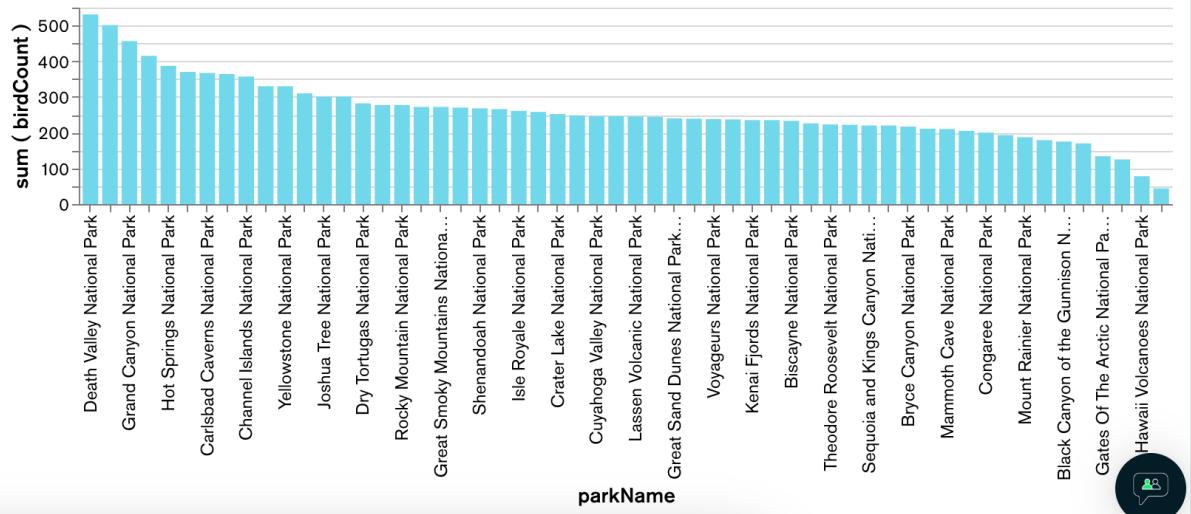
I have used a bar chart for establishing the relationship between total parks and per year groups, this visualisation of a bar chart gives a clear image and indication of how many parks that have been established between a certain specific year group, incremental by 10 years. In my opinion, given its nature, it is not something that requires significant understanding, it is easy to show something that everyone can understand, thus making this data appropriate and easy to access for the public.

C.1.2:

[PREVIEW](#) **COMPLETE** [SUBSET](#) [i](#)
[SUGGESTIONS](#) [...](#)

Average bird Species Per Park

Aggregate query C1.2

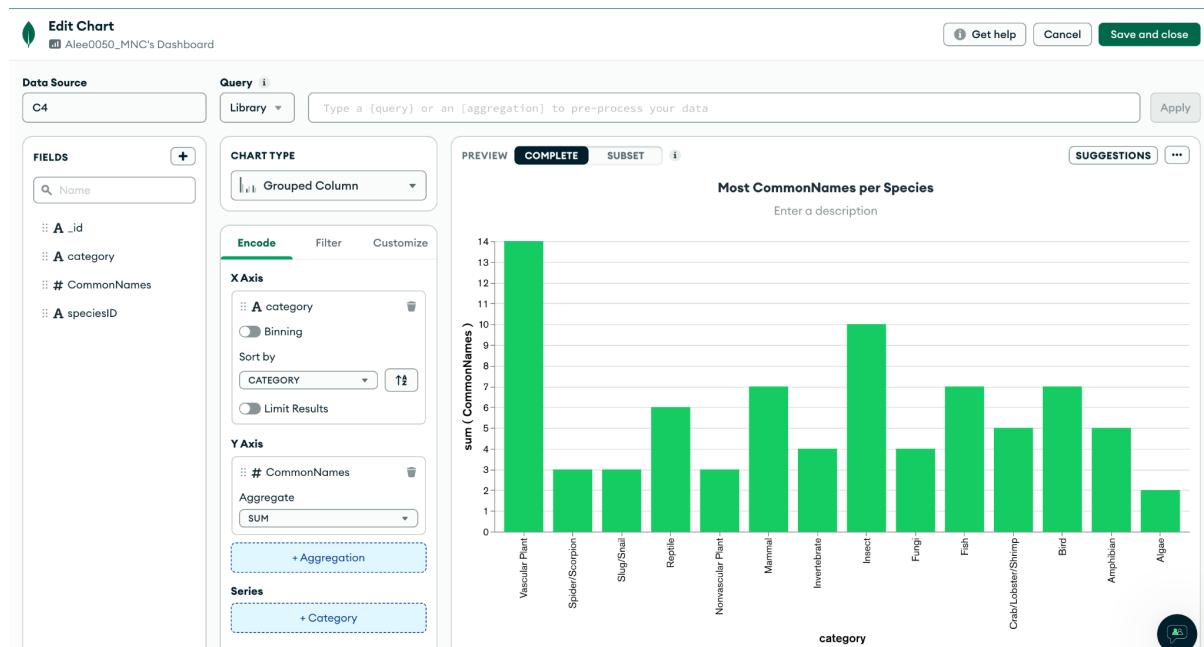


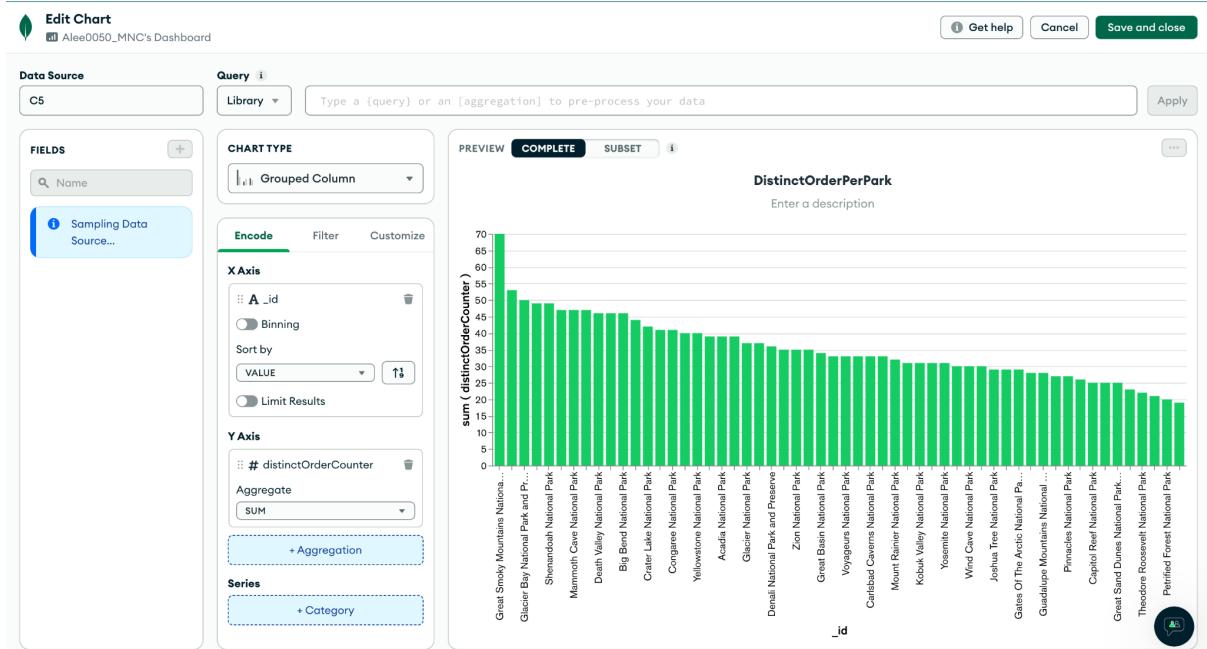
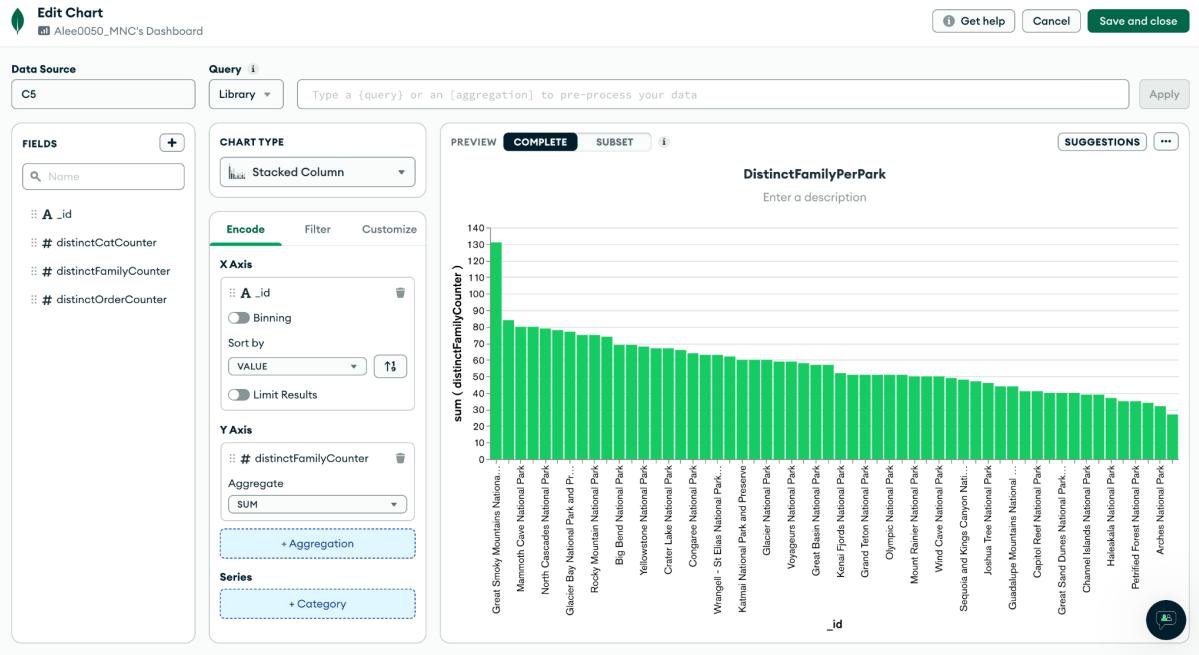
Justification:

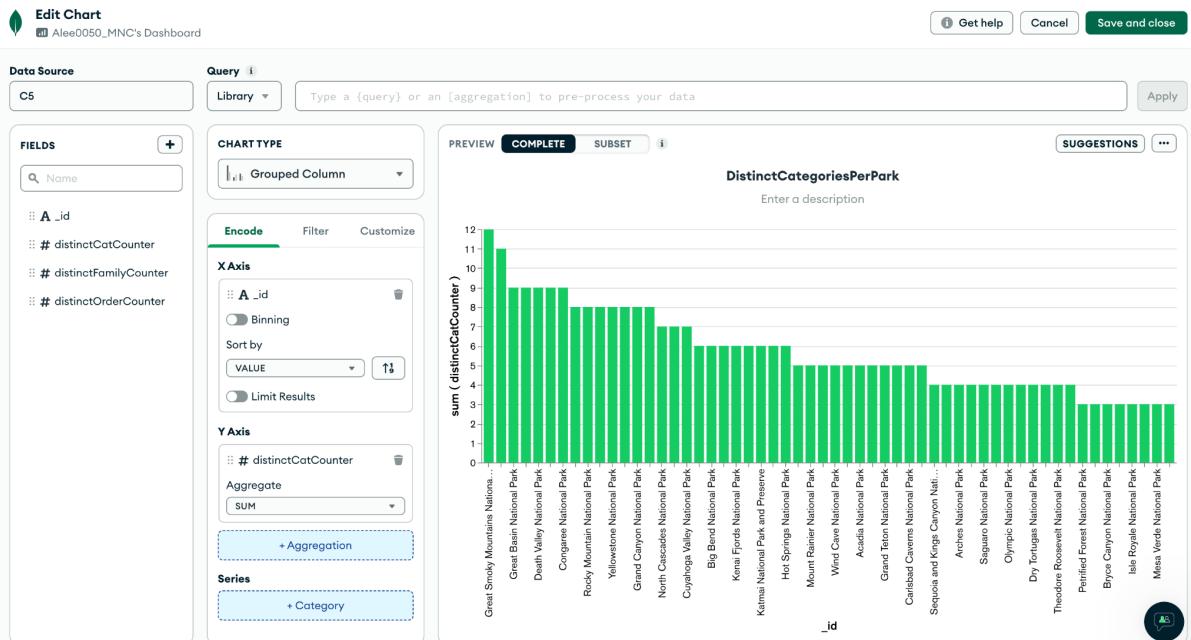
This displays the average bird species in one park, a barchart showcasing the different parks with the amount of birds per park using a barchart. It is easy to make a comparison between the two variables, as their relationship usually revolves around rise over run, furthermore the interpretation is only one, which makes it easy for others to interpret.

C1.3:

C1.4:



C1.5:




Justification:

Since we have three different variables to test the relationship between the park, we want to make 3 different distinct y variables in order to showcase the relationship.