Bryce Jensen
CS-340-Q7703
3-1 Milestone: Database Indexing and Authentication
1/20/24

Part I: Importing and Indexing a Data Set

1. In Apporto, open the terminal window to access the Linux shell. Upload the Austin Animal Center (AAC) Outcomes data set into MongoDB by importing a CSV file using the appropriate MongoDB import tool. Use the database name “AAC” and collection name “animals.” Complete the import using the mongoimport tool, and take screenshots of both the import command and its execution.

   Tip: How to import a CSV file is covered in the mongoimport documentation in the Module Three Resources Additional Support section. The command you use should be like the ones used in Module One Assignment and Module Two Assignment. (The requirements and rubric documents contain examples for the mongoimport command, but these use the default JSON datatype; adjust the command to import a CSV file.) You’ll find the Austin Animal Center (AAC) Outcomes data set in the /usr/local/datasets/ directory. The file name is “aac_shelter_outcomes.csv”.

```
brystonjensen_snhu@78e20b13519a: /usr/local/datasets
brystonjensen_snhu@78e20b13519a:/usr/local/datasets$ mongoimport --username="${MONGO_USER}" --
password="${MONGO_PASS}" --port=${MONGO_PORT} --host=${MONGO_HOST} --db=AAC --collection anima
ls --authenticationDatabase=admin  --type=csv --headerline --drop ./aac_shelter_outcomes.csv
2024-01-19T22:34:14.750+0000    connected to: mongodb://localhost:27017/
2024-01-19T22:34:14.751+0000    dropping: AAC.animals
2024-01-19T22:34:14.902+0000    10000 document(s) imported successfully. 0 document(s) failed
to import.
brystonjensen_snhu@78e20b13519a:/usr/local/datasets$
```

2. After importing your data set, open the mongo shell.
   Create a simple index on the key “breed.”

```
mongosh mongodb://127.0.0.1:27017/?directConnection=true&serverSelectionTimeoutMS=2000
AAC> db.animals.createIndex({"breed":1})
breed_1
AAC>
```

Bryce Jensen
CS-340-Q7703
3-1 Milestone: Database Indexing and Authentication
1/20/24

Show an example query using this index,

```
mongosh mongodb://127.0.0.1:27017/?directConnection=true&serverSelectionTimeoutMS=2000

AAC> db.animals.find({breed:"Border Collie"})
[
  {
    _id: ObjectId("65aaf8e650f02bacda40b9f1"),
    rec_num: 588,
    age_upon_outcome: '11 years',
    animal_id: 'A705192',
    animal_type: 'Dog',
    breed: 'Border Collie',
    color: 'Chocolate/White',
    date_of_birth: '2004-06-15',
    datetime: '2015-06-15 15:02:00',
    monthyear: '2015-06-15T15:02:00',
    name: 'Jaco',
    outcome_subtype: '',
    outcome_type: 'Return to Owner',
    sex_upon_outcome: 'Neutered Male',
    location_lat: 30.460619405566,
    location_long: -97.4980459509581,
    age_upon_outcome_in_weeks: 573.946626984127
  },
  {
    _id: ObjectId("65aaf8e650f02bacda40bc51"),
    rec_num: 1215,
    age_upon_outcome: '2 months',
    animal_id: 'A719782',
    animal_type: 'Dog',
    breed: 'Border Collie',
    color: 'Chocolate/White',
    date_of_birth: '2015-11-14',
    datetime: '2016-01-29 16:50:00',
    monthyear: '2016-01-29T16:50:00',
    name: 'Max',
    outcome_subtype: '',
    outcome_type: 'Adoption',
    sex_upon_outcome: 'Neutered Male',
    location_lat: 30.6362250953659,6,
    location_long: -97.7387952177054,5238095238
    age_upon_outcome_in_weeks: 10.9573412698413
  },
  { _id: ObjectId("65aaf8e650f02bacda40d216"),
    _id: ObjectId("65aaf8e650f02bacda40bed4"),
    rec_num: 1840,me: '1 year',
    age_upon_outcome: '1 year',
    animal_id: 'A720960',
    animal_type: 'Dog',ie',
    breed: 'Border Collie',
    color: 'Blue Merle',-11-08',
    date_of_birth: '2015-02-18',00',
    datetime: '2016-03-14 13:46:00',,
    monthyear: '2016-03-14T13:46:00',
    name: '*Aspen',: '',
    outcome_subtype: 'Partner',
    outcome_type: 'Transfer', Female',
    sex_upon_outcome: 'Intact Female',
    location_lat: 30.6287218760273,6,
    location_long: -97.3715286253088,1547619048
    age_upon_outcome_in_weeks: 55.7962301587302
  },
  { _id: ObjectId("65aaf8e650f02bacda40dc6e"),
    _id: ObjectId("65aaf8e650f02bacda40c72c"),
    rec_num: 3975,me: '8 months',
    age_upon_outcome: '3 years',
    animal_id: 'A694148',
    animal_type: 'Dog',ie',
    breed: 'Border Collie',
    color: 'Black/White',01-25',
    date_of_birth: '2011-12-21',00',
    datetime: '2014-12-22 18:05:00',,
    monthyear: '2014-12-22T18:05:00',
    name: 'Wes',ype: 'In Kennel',
    outcome_subtype: '',,
    outcome_type: 'Return to Owner',',
    sex_upon_outcome: 'Neutered Male',
    location_lat: 30.5761966044432,1,
    location_long: -97.5204312607125,3452380952
    age_upon_outcome_in_weeks: 156.821924603175
  },
  {>
    _id: ObjectId("65aaf8e650f02bacda40ca1d"),
    rec_num: 4729,
    age_upon_outcome: '1 year',
    animal_id: 'A681349',
    animal_type: 'Dog',
    breed: 'Border Collie',
    color: 'Black/White',
    date_of_birth: '2013-06-15',
    datetime: '2014-07-16 19:00:00',
    monthyear: '2014-07-16T19:00:00',
    name: 'Beau',
    outcome_subtype: 'Foster',
    outcome_type: 'Adoption',
    sex_upon_outcome: 'Neutered Male',
    location_lat: 30.6278502346631,
    location_long: -97.3864224007786,
    age_upon_outcome_in_weeks: 56.6845238095238
  },
  {
    _id: ObjectId("65aaf8e650f02bacda40d216"),
    rec_num: 6764,
    age_upon_outcome: '1 year',
    animal_id: 'A666851',
    animal_type: 'Dog',
    breed: 'Border Collie',
    color: 'Black/White',
    date_of_birth: '2012-11-08',
    datetime: '2013-12-06 12:39:00',
    monthyear: '2013-12-06T12:39:00',
    name: '*Chipmunk',
    outcome_subtype: '',
    outcome_type: 'Adoption',
    sex_upon_outcome: 'Spayed Female',
    location_lat: 30.4124963934857,
    location_long: -97.5934262387296,
    age_upon_outcome_in_weeks: 56.2181547619048
  },
  {
    _id: ObjectId("65aaf8e650f02bacda40dc6e"),
    rec_num: 9434,
    age_upon_outcome: '8 months',
    animal_id: 'A688661',
    animal_type: 'Dog',
    breed: 'Border Collie',
    color: 'Black/White',
    date_of_birth: '2014-01-25',
    datetime: '2014-09-29 18:15:00',
    monthyear: '2014-09-29T18:15:00',
    name: '',
    outcome_subtype: 'In Kennel',
    outcome_type: 'Died',
    sex_upon_outcome: 'Neutered Male',
    location_lat: 30.4498549420733,
    location_long: -97.6291535692241,
    age_upon_outcome_in_weeks: 35.3943452380952
  }
]
AAC>
```

and use the explain function to verify that the index will be used. Take screenshots of your example query.

```
mongosh mongodb://127.0.0.1:27017/?directConnection=true&serverSelectionTimeoutMS=2000
AAC> db.animals.createIndex({"breed":1})
breed_1
AAC> db.animals.find({breed:"Border Collie"}).explain("executionStats")
{
  explainVersion: '2',
  queryPlanner: {
    namespace: 'AAC.animals',
    indexFilterSet: false,
    parsedQuery: { breed: { '$eq': 'Border Collie' } },
    queryHash: 'A692FD15',
    planCacheKey: '3DF32E89',
    maxIndexedOrSolutionsReached: false,
    maxIndexedAndSolutionsReached: false,
    maxScansToExplodeReached: false,
    winningPlan: {
      queryPlan: {
        stage: 'FETCH',
        planNodeId: 2,
        inputStage: {
          stage: 'IXSCAN',
          planNodeId: 1,
          keyPattern: { breed: 1 },
          indexName: 'breed_1',
          isMultiKey: false,
          multiKeyPaths: { breed: [] },
          isUnique: false,
          isSparse: false,
          isPartial: false,
          indexVersion: 2,
          direction: 'forward',
          indexBounds: { breed: [ '["Border Collie", "Border Collie"]' ] }
        }
      },
      slotBasedPlan: {
        slots: '$$RESULT=s11 env: { s5 = KS(3C426F7264657220436F6C6C6965000104), s10 = {"breed" : 1}, s6 = KS(3C426F7264
657220436F6C6C696500FE04), s1 = TimeZoneDatabase(Australia/Canberra...America/Fortaleza) (timeZoneDB), s3 = 170570668041
9 (NOW), s2 = Nothing (SEARCH_META) }',
        stages: '[2] nlj inner [] [s4, s7, s8, s9, s10] \n' +
          '    left \n' +
          '        [1] cfilter {(exists(s5) && exists(s6))} \n' +
          '        [1] ixseek s5 s6 s9 s4 s7 s8 [] @"bc375937-6f5d-4dd7-9f15-07c5ab58a081" @"breed_1" true \n' +
          '    right \n' +
          '        [2] limit 1 \n' +
          '        [2] seek s4 s11 s12 s7 s8 s9 s10 [] @"bc375937-6f5d-4dd7-9f15-07c5ab58a081" true false \n'
      }
    },
    rejectedPlans: []
  },
  executionStats: {
    executionSuccess: true,
    nReturned: 7,
    executionTimeMillis: 1,
    totalKeysExamined: 7,
    totalDocsExamined: 7,
    executionStages: {
      stage: 'nlj',
      planNodeId: 2,
      nReturned: 7,
      executionTimeMillisEstimate: 0,
      opens: 1,
```

Bryce Jensen
CS-340-Q7703
3-1 Milestone: Database Indexing and Authentication
1/20/24

3. Create a compound index that will improve the performance of queries looking for breeds that have an “outcome_type” of “Transfer.” Show an example query using this compound index,

```
mongosh mongodb://127.0.0.1:27017/?directConnection=true&serverSelectionTimeoutMS=2000
AAC> db.animals.createIndex({"breed":1, "outcome_type":1})
breed_1_outcome_type_1
AAC> db.animals.find({breed:"Border Collie", outcome_type:"Transfer"})
[
  {
    _id: ObjectId("65ab02a7c239b5067f06decf"),
    rec_num: 1840,
    age_upon_outcome: '1 year',
    animal_id: 'A720966',
    animal_type: 'Dog',
    breed: 'Border Collie',
    color: 'Blue Merle',
    date_of_birth: '2015-02-18',
    datetime: '2016-03-14 13:46:00',
    monthyear: '2016-03-14T13:46:00',
    name: '*Aspen',
    outcome_subtype: 'Partner',
    outcome_type: 'Transfer',
    sex_upon_outcome: 'Intact Female',
    location_lat: 30.6287218760273,
    location_long: -97.3715286253088,
    age_upon_outcome_in_weeks: 55.7962301587302
  }
]
```

Bryce Jensen
CS-340-Q7703
3-1 Milestone: Database Indexing and Authentication
1/20/24

and use the explain function to confirm the index will be used. Take screenshots of your example query.

```
Select mongosh mongodb://127.0.0.1:27017/?directConnection=true&serverSelectionTimeoutMS=2000
AAC> db.animals.explain().find({breed:"Border Collie", outcome_type:"Transfer"})
{
  explainVersion: '2',
  queryPlanner: {
    namespace: 'AAC.animals',
    indexFilterSet: false,
    parsedQuery: {
      '$and': [
        { breed: { '$eq': 'Border Collie' } },
        { outcome_type: { '$eq': 'Transfer' } }
      ]
    },
    queryHash: '01F29F72',
    planCacheKey: '2AA57DD2',
    maxIndexedOrSolutionsReached: false,
    maxIndexedAndSolutionsReached: false,
    maxScansToExplodeReached: false,{ s5 = KS(3C426F7264657220436F6C6C6965000104), s10 = {"breed" : 1}, s6 = KS(3C426F72    winningPlan: {6500FE04), s1 = TimeZoneDatabase(Australia/C
anberra...America/Fortaleza) (timeZoneDB), s14 = "Transfer      queryPlan: {864 (NOW), s2 = Nothing (SEARCH_META) }',
        stage: 'FETCH',filter {traversef(s13, lambda(11.0) { ((l1.0 == s14) ?: false) }, false)} \n' +
        planNodeId: 2,nner [] [s4, s7, s8, s9, s10] \n' +
        inputStage: { \n' +
          stage: 'IXSCAN',filter {(exists(s5) && exists(s6))} \n' +
          planNodeId: 1, ixseek s5 s6 s9 s4 s7 s8 [] @"bc375937-6f5d-4dd7-9f15-07c5ab58a081" @"breed_1" true \n' +
          keyPattern: { breed: 1, outcome_type: 1 },
          indexName: 'breed_1_outcome_type_1',
          isMultiKey: false,k s4 s11 s12 s7 s8 s9 s10 [s13 = outcome_type] @"bc375937-6f5d-4dd7-9f15-07c5ab58a081" true
          multiKeyPaths: { breed: [], outcome_type: [] },
          isUnique: false,
          isSparse: false,
          isPartial: false,
          indexVersion: 2,
          direction: 'forward',
          indexBounds: {
            breed: [ '["Border Collie", "Border Collie"]' ],r' },
            outcome_type: [ '["Transfer", "Transfer"]' ]
          }
        }nfo: {
      },: '78e20b13519a',
      slotBasedPlan: {
        slots: '$$RESULT=s11 env: { s1 = TimeZoneDatabase(Australia/Canberra...America/Fortaleza) (timeZoneDB), s3 = 1705707863366 (NOW), s2 = Nothing (SEARCH_META), s5 = KS(3C426F72
6465722043 6F6C6C6965003C5472616E736665720001 04), s10 = {"breed" : 1}, s6 = KS(3C426F7264657220436F6C6C6965003C5472616E736665720 0FE04) }',
        stages: '[2] nlj inner [] [s4, s7, s8, s9, s10] \n' +
          '    left \n' +ferSizeBytes: 104857600,
          '        [1] cfilter {(exists(s5) && exists(s6))} \n' +
          '        [1] ixseek s5 s6 s9 s4 s7 s8 [] @"bc375937-6f5d-4dd7-9f15-07c5ab58a081" @"breed_1_outcome_type_1" true \n' +ernalDocumentSourceGroupMaxMemoryBytes: 104857600,
          '    right \n' +ngSortMemoryUsageBytes: 104857600,
          '        [2] limit 1 \n' +geOnMongoS: 0,
          '        [2] seek s4 s11 s12 s7 s8 s9 s10 [] @"bc375937-6f5d-4dd7-9f15-07c5ab58a081" true false \n'
      }ernalDocumentSourceSetWindowFieldsMaxMemoryBytes: 104857600,
    },ternalQueryFrameworkControl: 'trySbeEngine'
    rejectedPlans: [
      {
        queryPlan: {
          stage: 'FETCH',
          planNodeId: 2,
          filter: { outcome_type: { '$eq': 'Transfer' } },
          inputStage: {
            stage: 'IXSCAN',
            planNodeId: 1,
            keyPattern: { breed: 1 },
            indexName: 'breed_1',
            isMultiKey: false,
            multiKeyPaths: { breed: [] },
            isUnique: false,
            isSparse: false,
            isPartial: false,
            indexVersion: 2,
            direction: 'forward',
            indexBounds: { breed: [ '["Border Collie", "Border Collie"]' ] }
          }
        },
        slotBasedPlan: {
          slots: '$$RESULT=s11 env: { s3 = 1705707863366 (NOW), s6 = KS(3C426F7264657220436F6C6C696500FE04), s5 = KS(3C426F7264657220436F6C6C6965000104), s2 = Nothing (SEARCH_META),
s10 = {"breed" : 1}, s14 = "Transfer", s1 = TimeZoneDatabase(Australia/Canberra...America/Fortaleza) (timeZoneDB) }',
          stages: '[2] filter {traversef(s13, lambda(11.0) { ((l1.0 == s14) ?: false) }, false)} \n' +
            '[2] nlj inner [] [s4, s7, s8, s9, s10] \n' +
            '    left \n' +
            '        [1] cfilter {(exists(s5) && exists(s6))} \n' +
            '        [1] ixseek s5 s6 s9 s4 s7 s8 [] @"bc375937-6f5d-4dd7-9f15-07c5ab58a081" @"breed_1" true \n' +
            '    right \n' +
            '        [2] limit 1 \n' +
            '        [2] seek s4 s11 s12 s7 s8 s9 s10 [s13 = outcome_type] @"bc375937-6f5d-4dd7-9f15-07c5ab58a081" true false \n'
        }
      }
    ]
  },
  command: {
    find: 'animals',
    filter: { breed: 'Border Collie', outcome_type: 'Transfer' },
    '$db': 'AAC'
  },
  serverInfo: {
    host: '78e20b13519a',
    port: 27017,
    version: '7.0.2',
    gitVersion: '02b3c655e1302209ef046da6ba3ef6749dd0b62a'
  },
  serverParameters: {
    internalQueryFacetBufferSizeBytes: 104857600,
    internalQueryFacetMaxOutputDocSizeBytes: 104857600,
    internalLookupStageIntermediateDocumentMaxSizeBytes: 104857600,
    internalDocumentSourceGroupMaxMemoryBytes: 104857600,
    internalQueryMaxBlockingSortMemoryUsageBytes: 104857600,
    internalQueryProhibitBlockingMergeOnMongoS: 0,
    internalQueryMaxAddToSetBytes: 104857600,
    internalDocumentSourceSetWindowFieldsMaxMemoryBytes: 104857600,
    internalQueryFrameworkControl: 'trySbeEngine'
  },
  ok: 1
AAC>
AAC>
```

Bryce Jensen
CS-340-Q7703
3-1 Milestone: Database Indexing and Authentication
1/20/24

Part II: User Authentication

1. Create a new user account called "aacuser" for the database AAC in the mongo shell. Refer to steps 6–7 of the MongoDB Manual Enable Access Control tutorial for help with this task. You will need to modify the commands so the account name is "aacuser." Additional information with respect to user management may be found in the User Management in MongoDB document.

   Note: You will need to create your user in the admin database, even though it will only have a role in the aac database. When you use the mongo shell (mongosh) to connect to the database with your user account, you will need to reset two environment variables to reflect the username and password you just created. (The MongoDB hostname and port will be pulled automatically from your environment variables.)

```
mongosh mongodb://127.0.0.1:27017/?directConnection=true&serverSelectionTimeoutMS=2000
admin> db.createUser(
... {
... user: "aacuser",
... pwd: "simplepass",
... roles: [ { role: "readWrite", db: "aac" } ]
... }
... )
{ ok: 1 }
admin> db.getUsers()
{
  users: [
    {
      _id: 'admin.aacuser',
      userId: new UUID("9ad0b099-5319-4f92-9a33-53f15402c8d5"),
      user: 'aacuser',
      db: 'admin',
      roles: [ { role: 'readWrite', db: 'aac' } ],
      mechanisms: [ 'SCRAM-SHA-1', 'SCRAM-SHA-256' ]
    },
    {
      _id: 'admin.root',
      userId: new UUID("e530898f-0b7a-4bf0-ad2d-044f09c07e81"),
      user: 'root',
      db: 'admin',
      roles: [ { role: 'root', db: 'admin' } ],
      mechanisms: [ 'SCRAM-SHA-1', 'SCRAM-SHA-256' ]
    }
  ],
  ok: 1
}
admin>
```

Tip: To make it easier to log in as your user, open up a second terminal session in your Linux environment. In this session, set the following environment variables:

MONGO_USER=aacuser

MONGO_PASS=The password you set when you created the aacuser account.

Once this is complete, you can run mongosh in that window as your new user.

Bryce Jensen
CS-340-Q7703
3-1 Milestone: Database Indexing and Authentication
1/20/24

2. Take a screenshot of your login process to MongoDB using the mongo shell. Be sure you can access MongoDB and list the databases using both the admin and aacuser accounts. This task will verify that your accounts are working. You should be able to include the login commands for both accounts in one screenshot, but if you cannot, include two screenshots to show both login commands.
   When you have successfully attached to your MongoDB instance with mongosh, use the following command to verify that you have connected as a specific user:
   `db.runCommand({connectionStatus:1})`
   Include the results in your screenshot for each user.

```
mongosh mongodb://127.0.0.1:27017/?directConnection=true&serverSelectionTimeoutMS=2000
admin> db.runCommand({connectionStatus: 1})
{
  authInfo: {
    authenticatedUsers: [ { user: 'root', db: 'admin' } ],
    authenticatedUserRoles: [ { role: 'root', db: 'admin' } ]
  },
  ok: 1
}
admin> show dbs
AAC       1.63 MiB
admin   156.00 KiB
city     16.60 MiB
config  108.00 KiB
enron     7.64 MiB
local    72.00 KiB
admin>
```

```
mongosh mongodb://127.0.0.1:27017/?directConnection=true&serverSelectionTimeoutMS=2000
Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

Install the latest PowerShell for new features and improvements! https://aka.ms/PSWindows

PS C:\windows\system32> docker exec -it cs340Mongo-brystonjensen_snhu /bin/bash
root@78e20b13519a:/# su - brystonjensen_snhu
brystonjensen_snhu@78e20b13519a:~$ MONGO_USER=aacuser
brystonjensen_snhu@78e20b13519a:~$ MONGO_PASS=simplepass
brystonjensen_snhu@78e20b13519a:~$ mongosh
Current Mongosh Log ID: 65ab19cdc48c9ca19f463a9c
Connecting to:          mongodb://127.0.0.1:27017/?directConnection=true&serverSelectionTimeout
MS=2000&appName=mongosh+2.0.2
Using MongoDB:          7.0.2
Using Mongosh:          2.0.2

For mongosh info see: https://docs.mongodb.com/mongodb-shell/

admin> db.runCommand({connectionStatus : 1})
{
  authInfo: {
    authenticatedUsers: [ { user: 'aacuser', db: 'admin' } ],
    authenticatedUserRoles: [ { role: 'readWrite', db: 'AAC' } ]
  },
  ok: 1
}
admin> show dbs
AAC  1.63 MiB
admin>
```