

PRACTICAL 4

INDEXING USING MONGODB

1. Mongo DB indexing

```
test> use students
switched to db students
```

```
students> db.createCollection("studentgrades")
{ ok: 1 }
```

```
students> db.studentgrades.insertMany(
... [
... {name:"Barry",subject:"Maths",score:92},
... {name:"Kent",subject:"Physics",score:87},
... {name:"Harry",subject:"Maths",score:99,notes:"ExceptionalPerformance"},
... {name:"Alex",subject:"Literature",score:78},
... {name:"Tom",subject:"History",score:65,notes:"Adequate"}
... ])
{
  acknowledged: true,
  insertedIds: {
    '0': ObjectId('678a27b777716b61d6228fb5'),
    '1': ObjectId('678a27b777716b61d6228fb6'),
    '2': ObjectId('678a27b777716b61d6228fb7'),
    '3': ObjectId('678a27b777716b61d6228fb8'),
    '4': ObjectId('678a27b777716b61d6228fb9')
  }
}
```

```
students> db.studentgrades.find({},{_id:0})
[
  { name: 'Barry', subject: 'Maths', score: 92 },
  { name: 'Kent', subject: 'Physics', score: 87 },
  {
    name: 'Harry',
    subject: 'Maths',
    score: 99,
    notes: 'ExceptionalPerformance'
  },
  { name: 'Alex', subject: 'Literature', score: 78 },
  { name: 'Tom', subject: 'History', score: 65, notes: 'Adequate' }
]
```

```
students> db.studentgrades.find().pretty()
[
  {
    _id: ObjectId('678a27b777716b61d6228fb5'),
    name: 'Barry',
    subject: 'Maths',
    score: 92
  },
  {
    _id: ObjectId('678a27b777716b61d6228fb6'),
    name: 'Kent',
    subject: 'Physics',
    score: 87
  },
  {
    _id: ObjectId('678a27b777716b61d6228fb7'),
    name: 'Harry',
    subject: 'Maths',
    score: 99,
    notes: 'ExceptionalPerformance'
  },
  {
    _id: ObjectId('678a27b777716b61d6228fb8'),
    name: 'Alex',
    subject: 'Literature',
    score: 78
  },
  {
    _id: ObjectId('678a27b777716b61d6228fb9'),
    name: 'Tom',
    subject: 'History',
    score: 65,
    notes: 'Adequate'
  }
]
```

Creating Index in MongoDB:

```
students> db.studentgrades.createIndex({name:1},{name:"student name index"})
student name index
```

Finding indexes :

You can find all the available indexes in a MongoDB collection by using the getIndexes method.

```
students> db.studentgrades.getIndexes()
[
  { v: 2, key: { _id: 1 }, name: '_id_' },
  { v: 2, key: { name: 1 }, name: 'student name index' }
]
```

Dropping index : To delete an index from a collection, use the dropIndex Smethod while specifying the index name to be dropped.

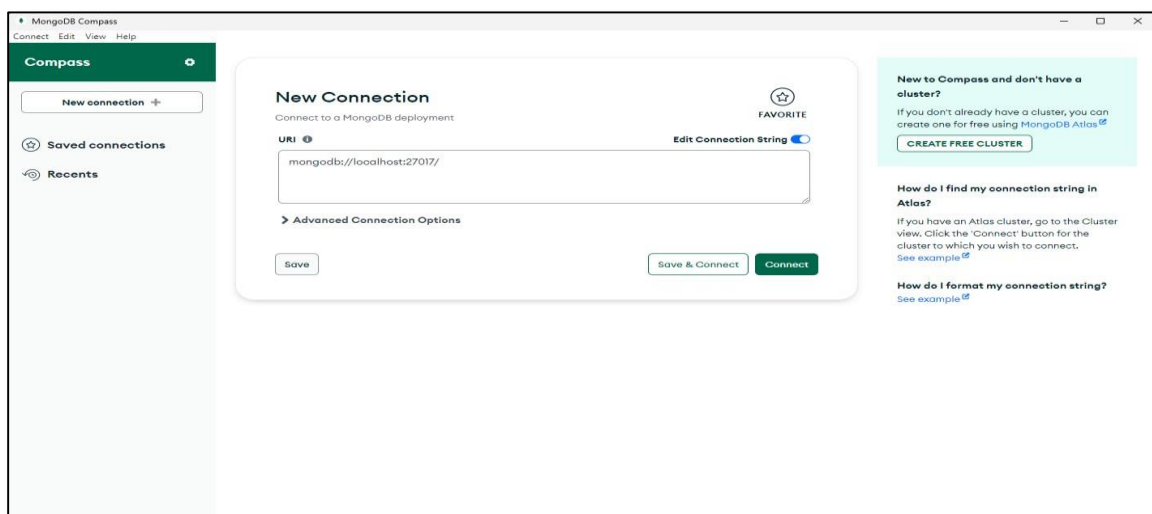
```
students> db.studentgrades.dropIndex("student name index")
{ nIndexesWas: 2, ok: 1 }
```

Dropping all the indexes:

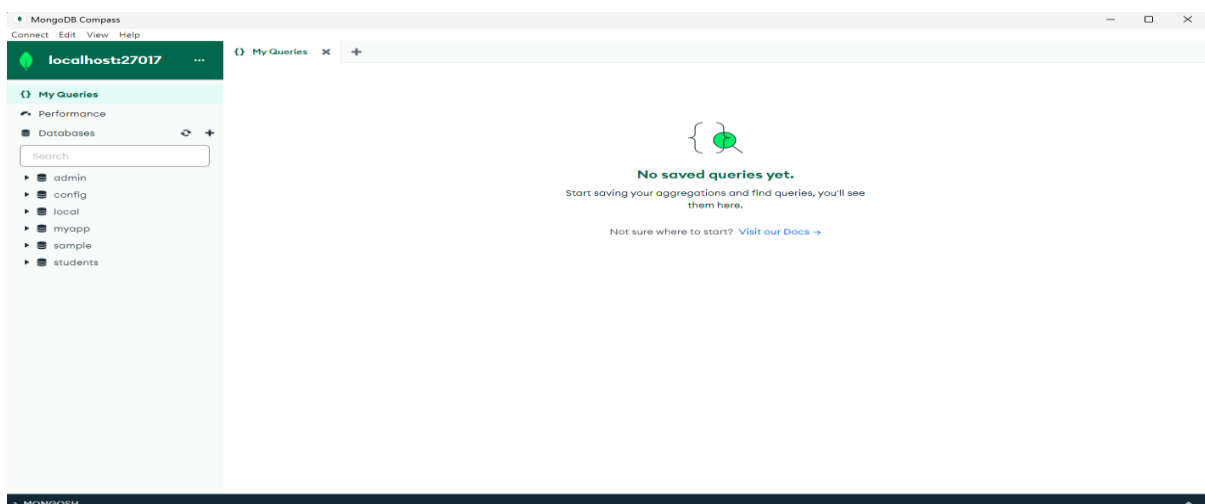
The dropIndexes command can also drop all the indexes excluding the default _id index.

```
students> db.studentgrades.getIndexes()
[
  { v: 2, key: { _id: 1 }, name: '_id_' },
  { v: 2, key: { name: 1 }, name: 'student name index' }
]
```

2. Create all the types of indexes (discussed in class) which will help in finding certain words in a document by using AIRPORT (dataset).

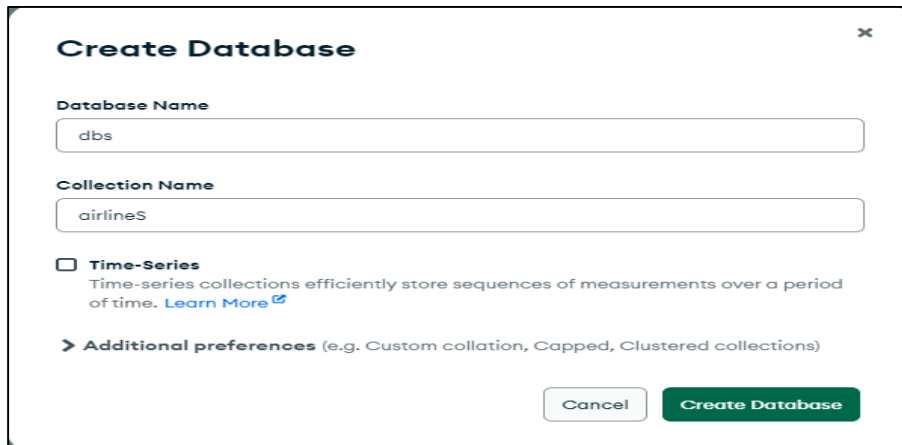


Click on connect button



Click on + button

Enter the database name and collection name

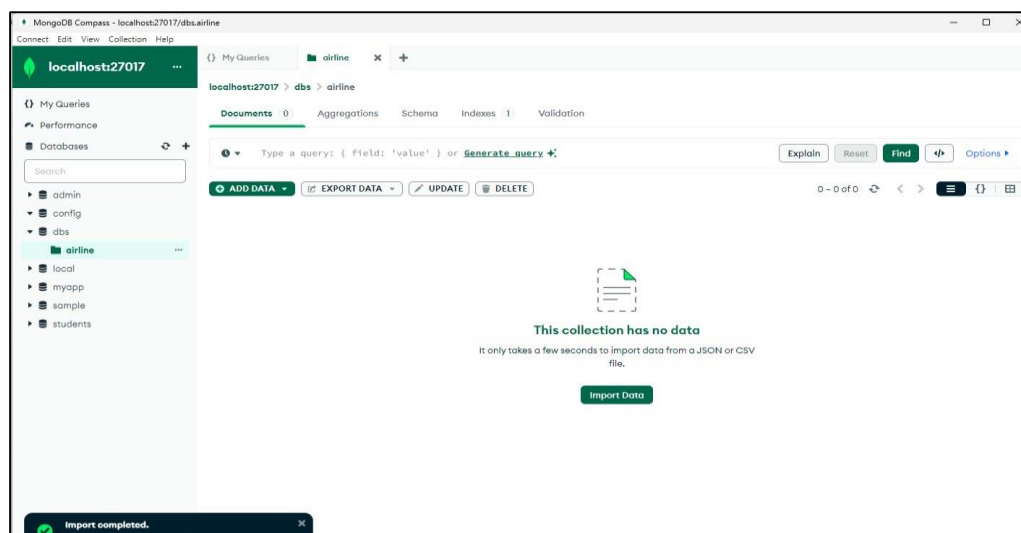


The 'Create Database' dialog box in MongoDB Compass. It features a title bar with a close button. Below the title, there are two input fields: 'Database Name' with the value 'dbs' and 'Collection Name' with the value 'airlineS'. A checkbox labeled 'Time-Series' is present, with a description: 'Time-series collections efficiently store sequences of measurements over a period of time. [Learn More](#)'. Below this, there is a link '> Additional preferences (e.g. Custom collation, Capped, Clustered collections)'. At the bottom right, there are two buttons: 'Cancel' and 'Create Database'.

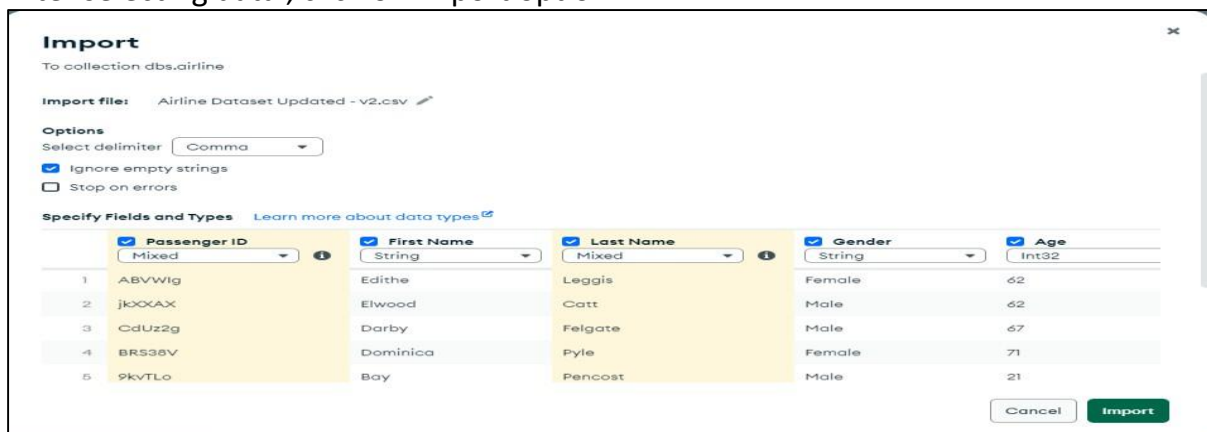
Click on create database option

Click on import data option

Select the downloaded file



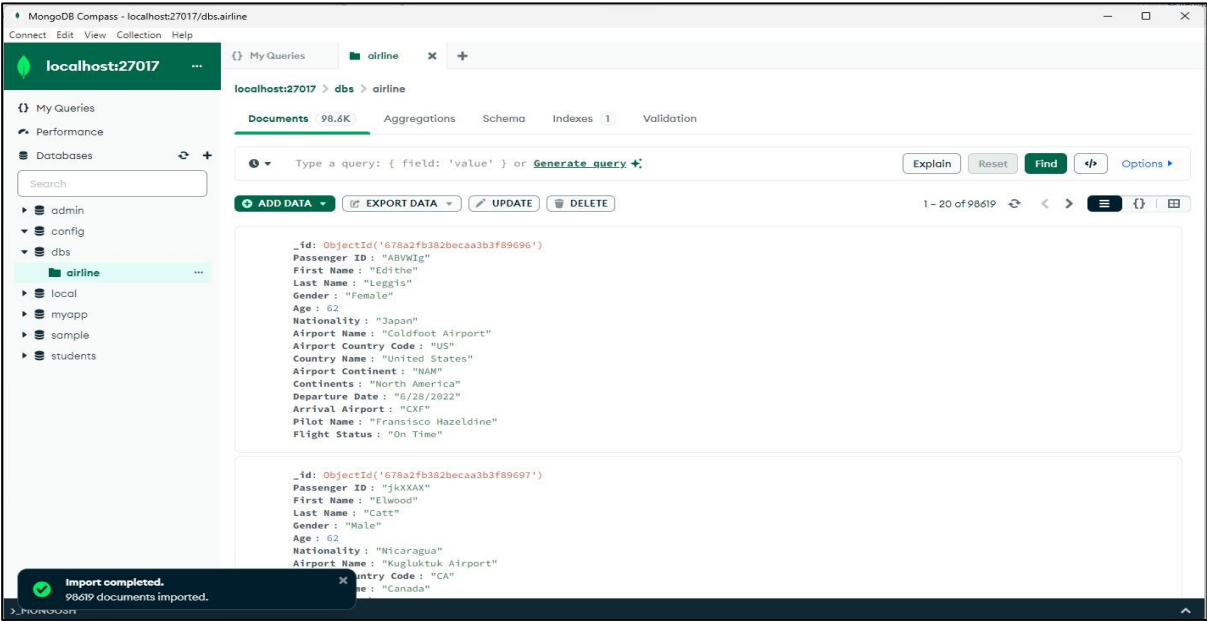
After selecting data , click on import option



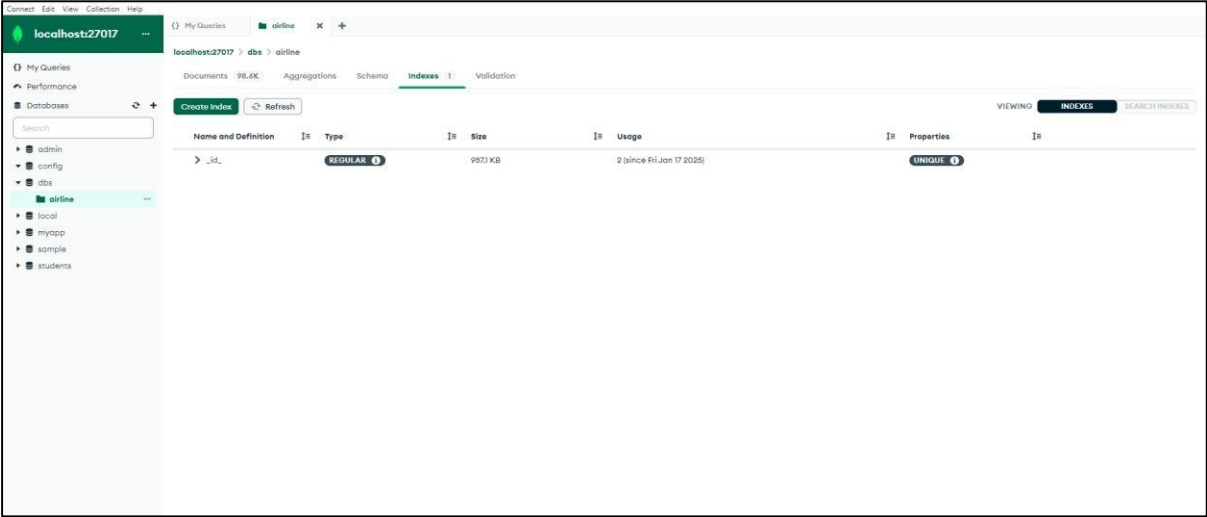
The 'Import' dialog box in MongoDB Compass. It shows the target collection as 'dbs.airline'. The 'Import file' is 'Airline Dataset Updated - v2.csv'. Under 'Options', the 'Select delimiter' is set to 'Comma', and 'Ignore empty strings' is checked. The 'Specify Fields and Types' section shows a table with columns: Passenger ID, First Name, Last Name, Gender, and Age. Each column has a dropdown menu for its type. The data rows are as follows:

| | Passenger ID | First Name | Last Name | Gender | Age |
|---|--------------|------------|-----------|--------|-----|
| 1 | ABVWig | Edithe | Leggis | Female | 62 |
| 2 | jkXXAX | Elwood | Catt | Male | 62 |
| 3 | CdUz2g | Darby | Felgate | Male | 67 |
| 4 | BR538V | Dominica | Pyle | Female | 71 |
| 5 | 9kvTL0 | Bay | Pencost | Male | 21 |

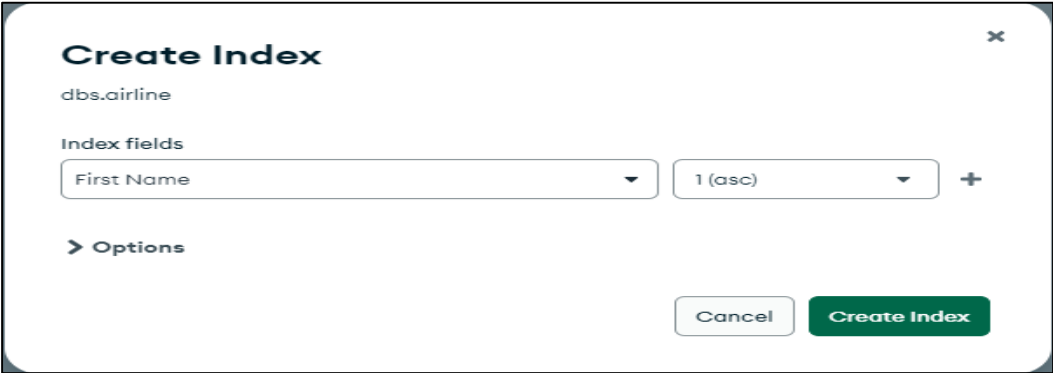
At the bottom right, there are 'Cancel' and 'Import' buttons.



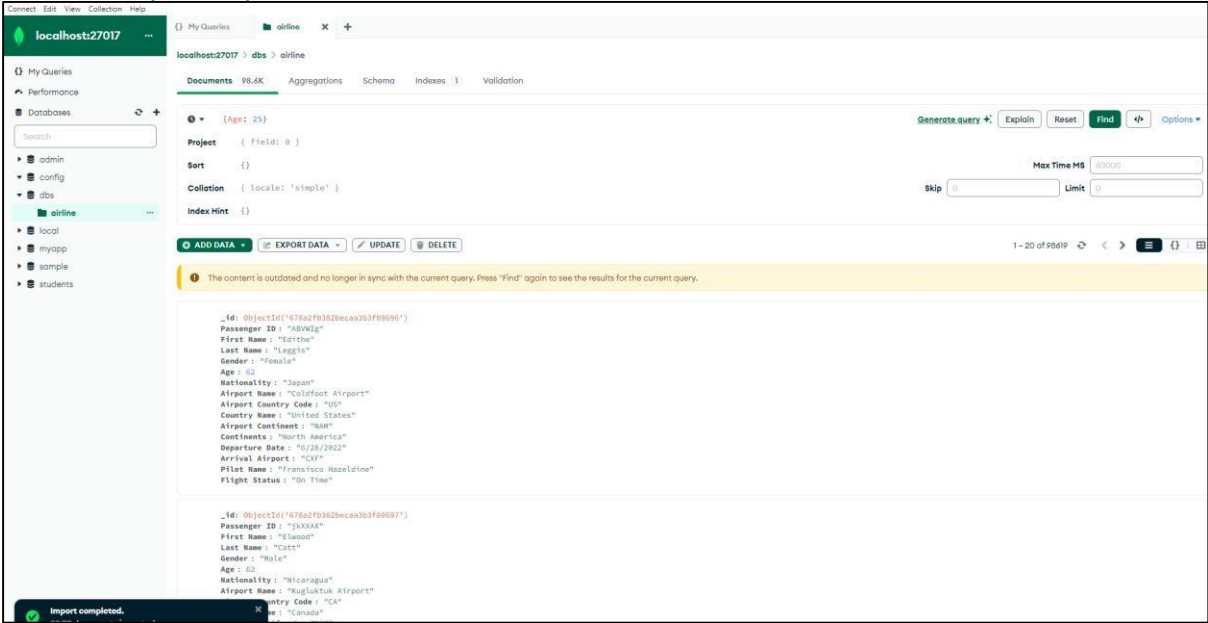
After importing data, go to index option and select create index option



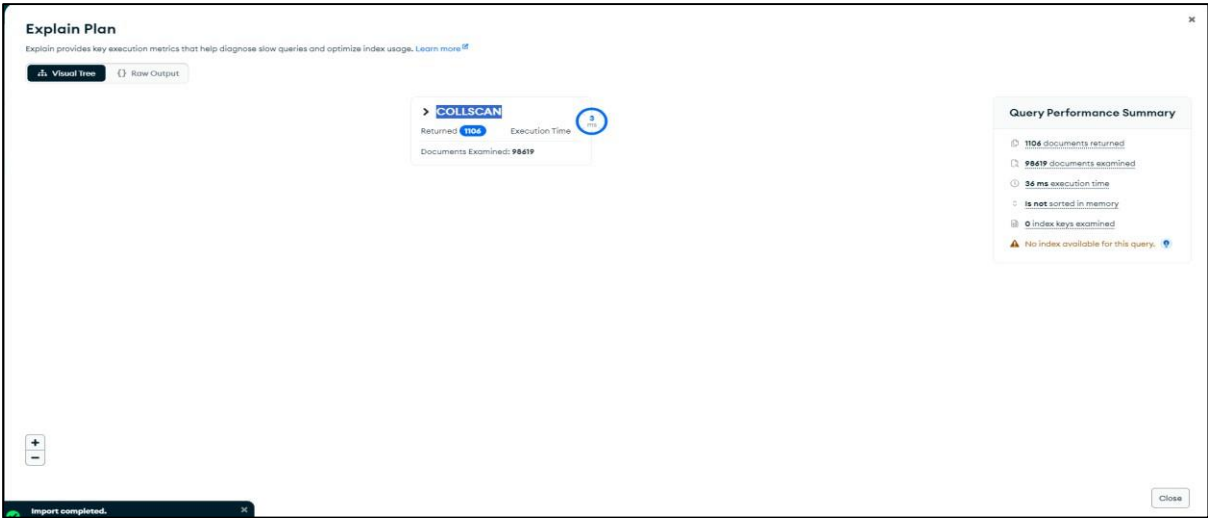
Create index



Find the particular data in dataset
Click on explain option



To see the execution time without indexing



Go to the index and Create index



Again execute the query and to see the execution time

localhost:27017

My Queries

Performance

Databases

admin

config

db

airline

local

myapp

sample

students

localhost:27017 > db > airline

Documents 98.4K

Aggregations

Schema

Indexes 1

Validation

{ Age: 25 }

Project { field: 0 }

Sort { }

Collation { locale: 'simple' }

Index Hint { }

Generate query Explain Reset Find Options

Max Time MS 30000

Skip 0 Limit 0

1 - 20 of 98419

ADD DATA EXPORT DATA UPDATE DELETE

The content is outdated and no longer in sync with the current query. Press "Find" again to see the results for the current query.

_id: ObjectId('678a2f9382becaa303f69696')

Passenger ID: '48VWZg'

First Name: 'Editha'

Last Name: 'Leggls'

Gender: 'Female'

Age: 62

Nationality: 'Japan'

Airport Name: 'Colofout Airport'

Airport Country Code: 'US'

Country Name: 'United States'

Airport Continent: 'NAM'

Continents: 'North America'

Departure Date: '6/28/2022'

Arrival Airport: 'COP'

Pilot Name: 'Francisco Nazeldine'

Flight Status: 'On Time'

_id: ObjectId('678a2f9382becaa303f69697')

Passenger ID: 'jKXXXX'

First Name: 'Elwood'

Last Name: 'Catt'

Gender: 'Male'

Age: 62

Nationality: 'Nicaragua'

Airport Name: 'Ruglukuk Airport'

Country Code: 'CA'

Country Name: 'Canada'

Explain Plan

Explain provides key execution metrics that help diagnose slow queries and optimize index usage. [Learn more](#)

Visual Tree

Raw Output

> FETCH

Returned 1106

Execution Time 3 ms

> IXSCAN

Returned 1106

Execution Time 0 ms

Index Name: Age_1

Multi Key Index: no

Query Performance Summary

1106 documents returned

1106 documents examined

3 ms execution time

is not sorted in memory

1106 index keys examined

Query used the following index

Age ↑

Import completed.

98419 documents imported

Close