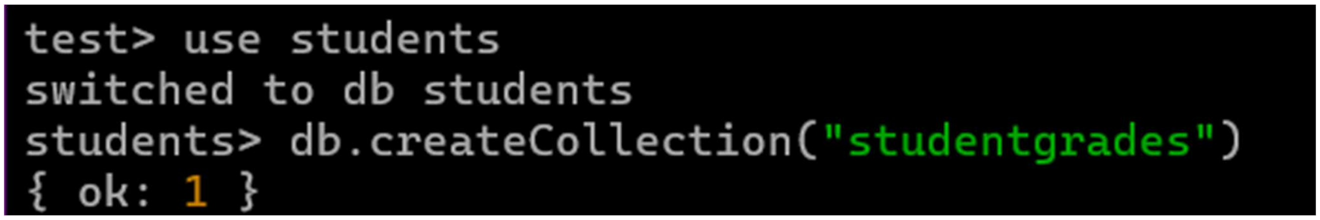
Practical 4 Indexing using MongoDB

Class: MSc. DSAI   Roll No.:L005

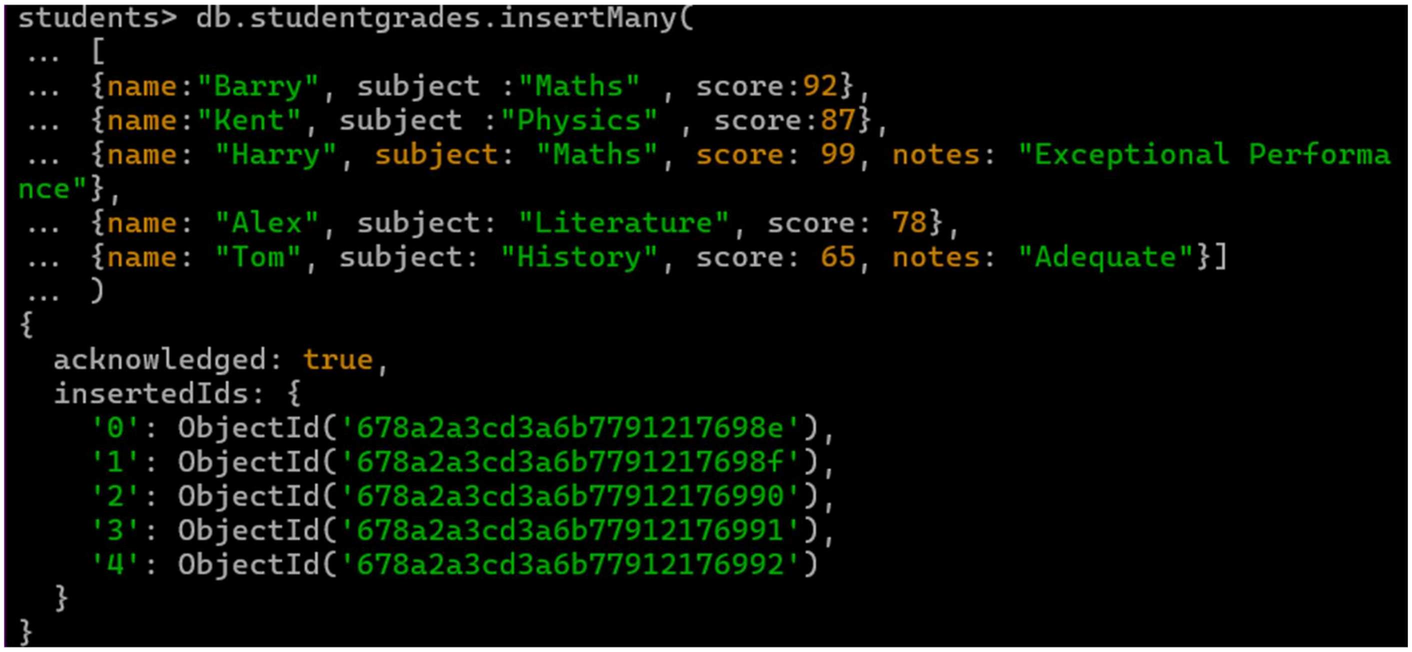
Q1 Mongo DB indexing

Create database and create collection of name Studentgrades

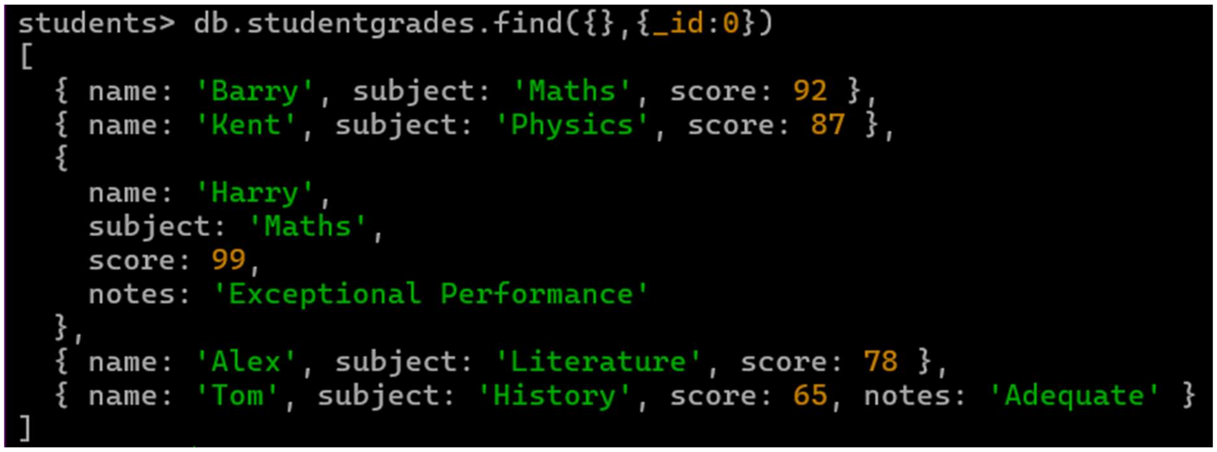


```
test> use students
switched to db students
students> db.createCollection("studentgrades")
{ ok: 1 }
```

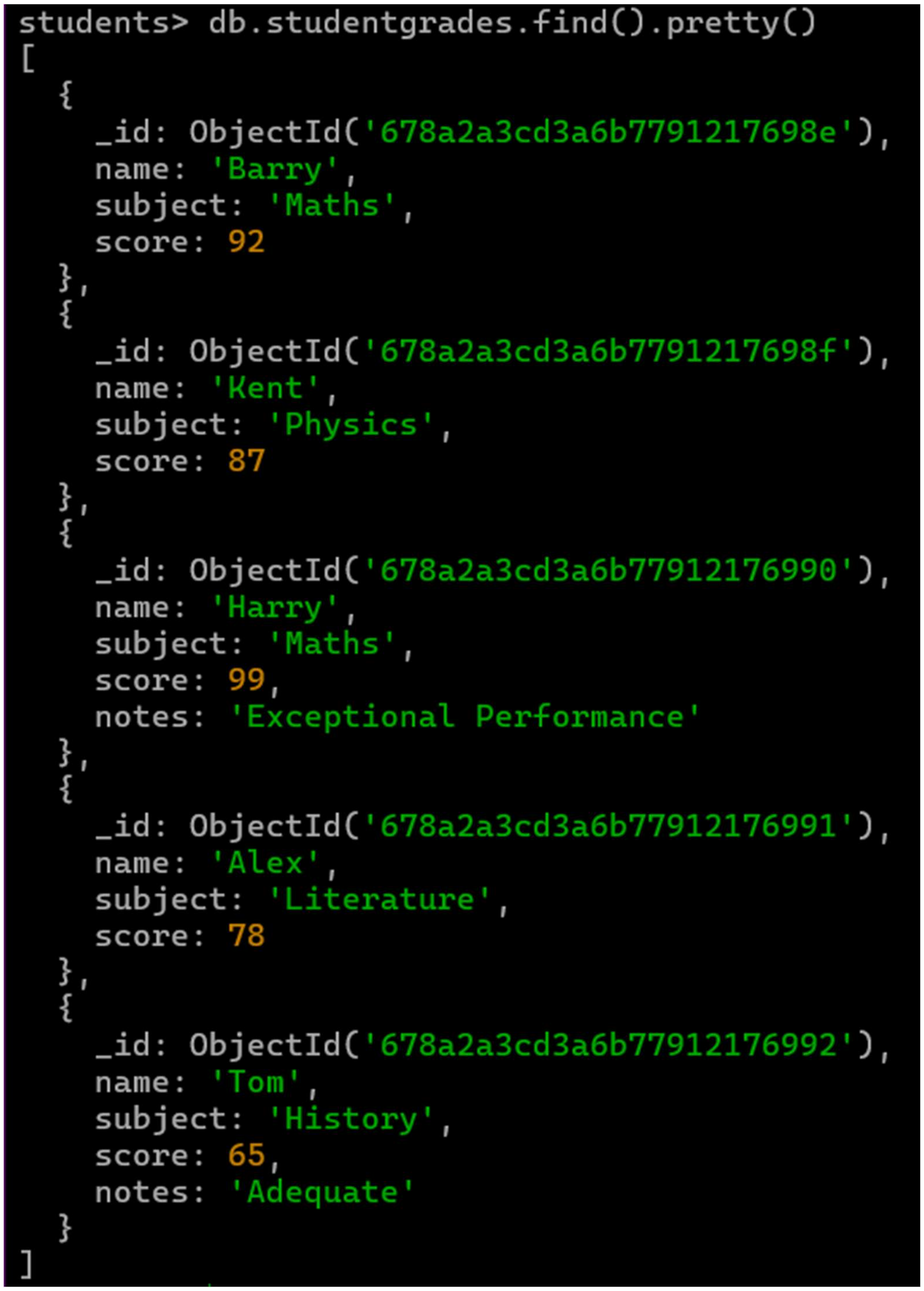
Insert data into the collection



```
students> db.studentgrades.insertMany(
...  [
...  {name:"Barry", subject :"Maths" , score:92},
...  {name:"Kent", subject :"Physics" , score:87},
...  {name: "Harry", subject: "Maths", score: 99, notes: "Exceptional Performa
nce"},
...  {name: "Alex", subject: "Literature", score: 78},
...  {name: "Tom", subject: "History", score: 65, notes: "Adequate"}]
...  )
{
  acknowledged: true,
  insertedIds: {
    '0': ObjectId('678a2a3cd3a6b7791217698e'),
    '1': ObjectId('678a2a3cd3a6b7791217698f'),
    '2': ObjectId('678a2a3cd3a6b77912176990'),
    '3': ObjectId('678a2a3cd3a6b77912176991'),
    '4': ObjectId('678a2a3cd3a6b77912176992')
  }
}
```

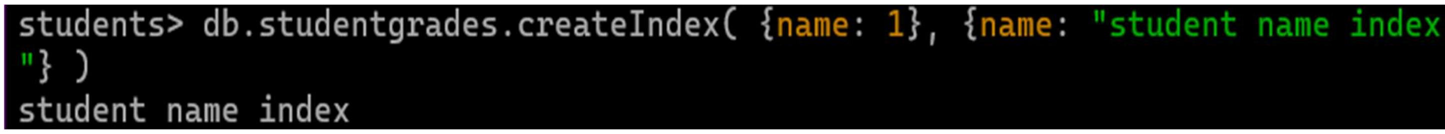


```
students> db.studentgrades.find({},{_id:0})
[
  { name: 'Barry', subject: 'Maths', score: 92 },
  { name: 'Kent', subject: 'Physics', score: 87 },
  {
    name: 'Harry',
    subject: 'Maths',
    score: 99,
    notes: 'Exceptional Performance'
  },
  { name: 'Alex', subject: 'Literature', score: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



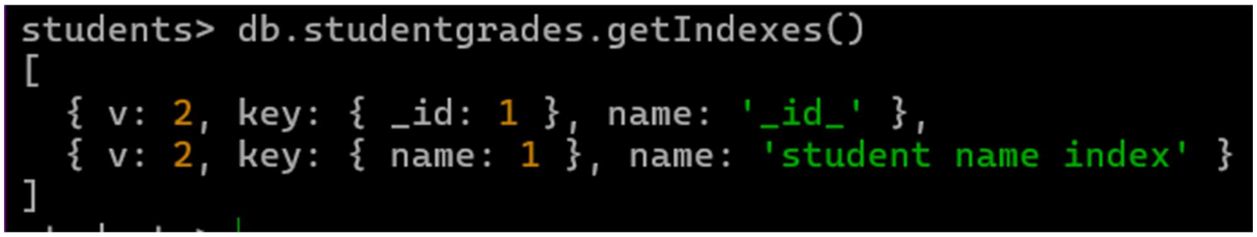
```
students> db.studentgrades.find().pretty()
[
  {
    _id: ObjectId('678a2a3cd3a6b7791217698e'),
    name: 'Barry',
    subject: 'Maths',
    score: 92
  },
  {
    _id: ObjectId('678a2a3cd3a6b7791217698f'),
    name: 'Kent',
    subject: 'Physics',
    score: 87
  },
  {
    _id: ObjectId('678a2a3cd3a6b77912176990'),
    name: 'Harry',
    subject: 'Maths',
    score: 99,
    notes: 'Exceptional Performance'
  },
  {
    _id: ObjectId('678a2a3cd3a6b77912176991'),
    name: 'Alex',
    subject: 'Literature',
    score: 78
  },
  {
    _id: ObjectId('678a2a3cd3a6b77912176992'),
    name: 'Tom',
    subject: 'History',
    score: 65,
    notes: 'Adequate'
  }
]
```

Creating index



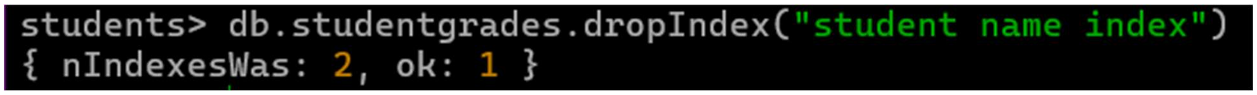
```
students> db.studentgrades.createIndex( {name: 1}, {name: "student name index
"} )
student name index
```

Finding the indexes in a collection



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

Drop indexes in a collection



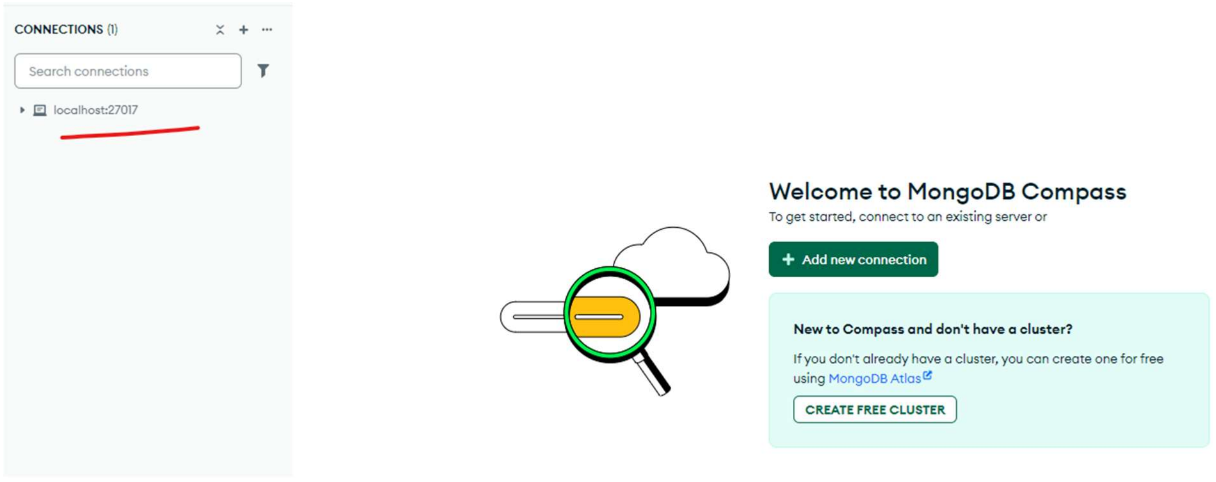
```
students> db.studentgrades.dropIndex("student name index")
{ nIndexesWas: 2, ok: 1 }
```

Drop all indexes in a collection

index. db.studentgrades.dropIndexes()

Q2 Create all the types of indexes (discussed in class) which will help in finding certain words in a document by using AIRPORT (dataset).

Open MongoDb Compass

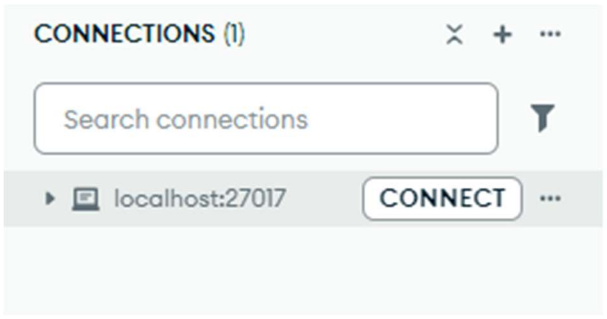


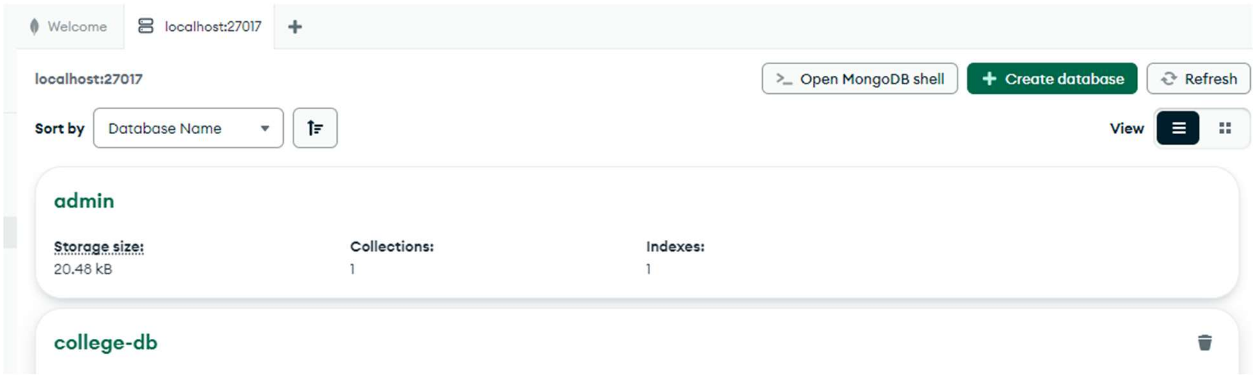
Welcome to MongoDB Compass

To get started, connect to an existing server or

New to Compass and don't have a cluster?

If you don't already have a cluster, you can create one for free using MongoDB Atlas

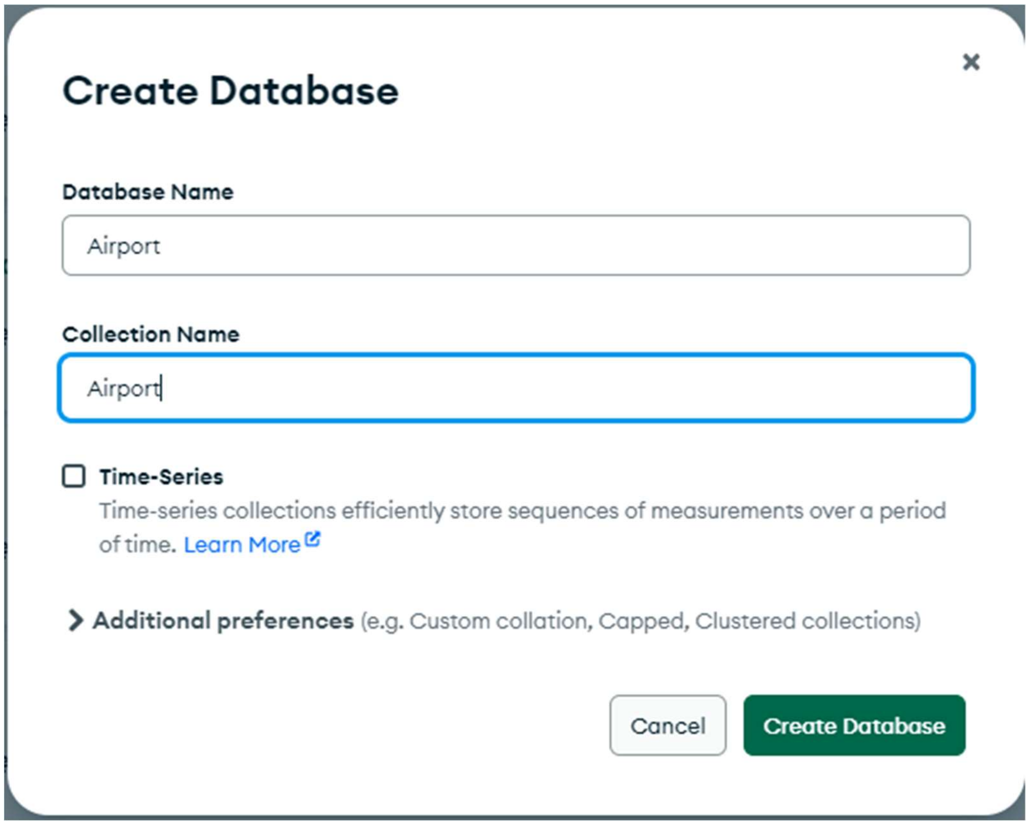




localhost:27017

admin		
Storage size:	Collections:	Indexes:
20.48 kB	1	1
college-db		

Creating Database


Create Database

Database Name

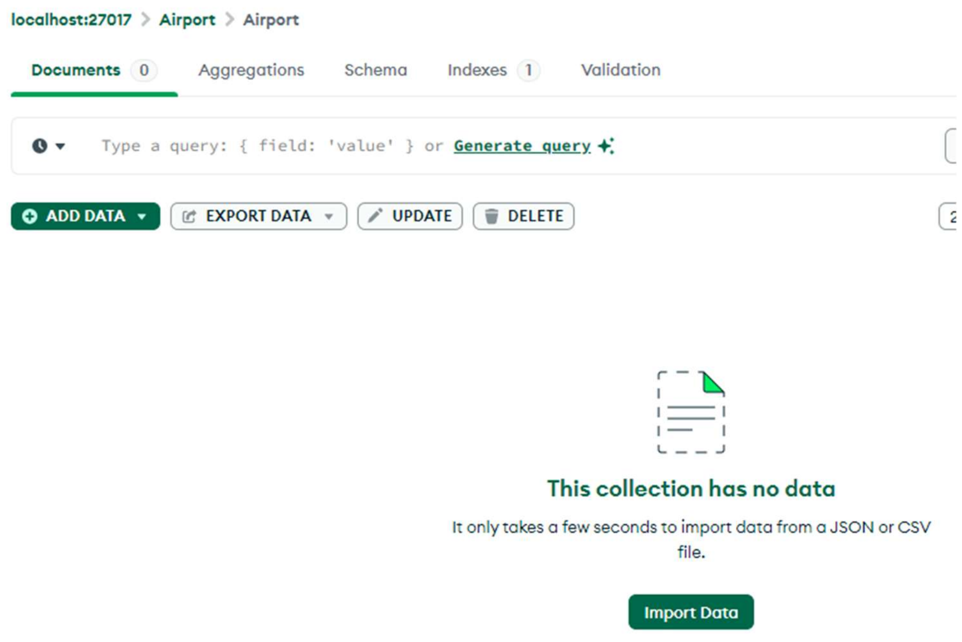
Airport

Collection Name

Airport

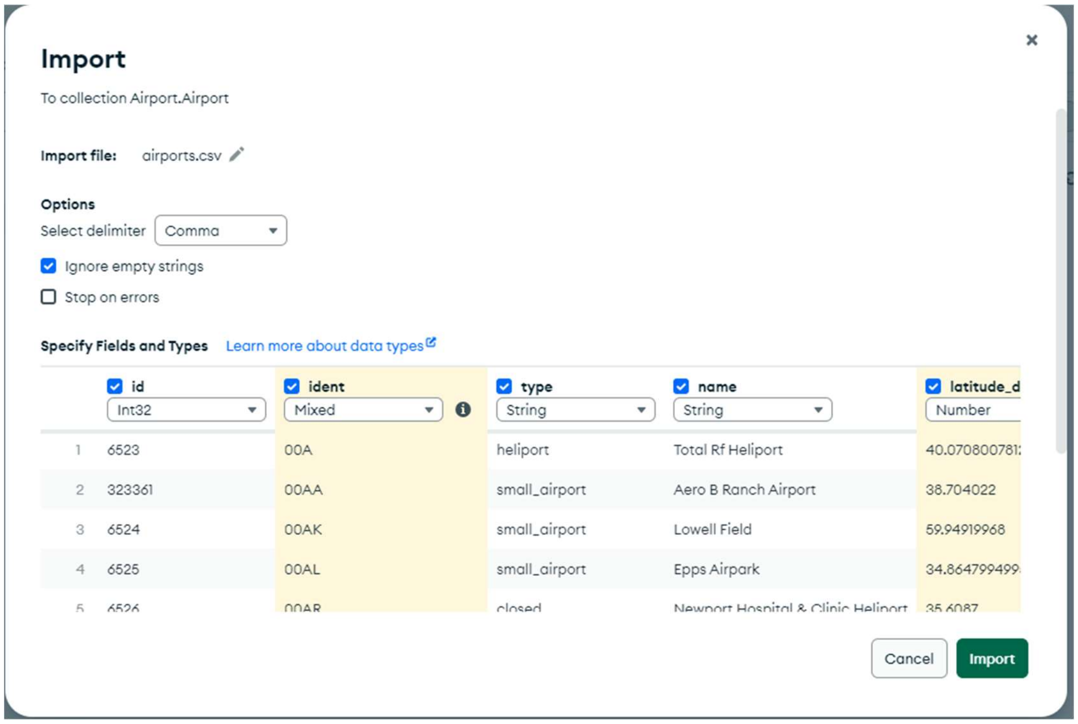
☐ **Time-Series**
Time-series collections efficiently store sequences of measurements over a period of time. Learn More

> **Additional preferences** (e.g. Custom collation, Capped, Clustered collections)


localhost:27017 > Airport > Airport

Documents 0
Aggregations
Schema
Indexes 1
Validation
Type a query: { field: 'value' } or Generate query
ADD DATA
EXPORT DATA
UPDATE
DELETE
This collection has no data
It only takes a few seconds to import data from a JSON or CSV file.
Import Data

Importing Data into Database


Import

To collection Airport.Airport

Import file: airports.csv

Options

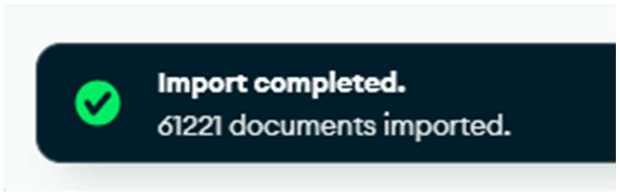
Select delimiter: Comma

✓ Ignore empty strings

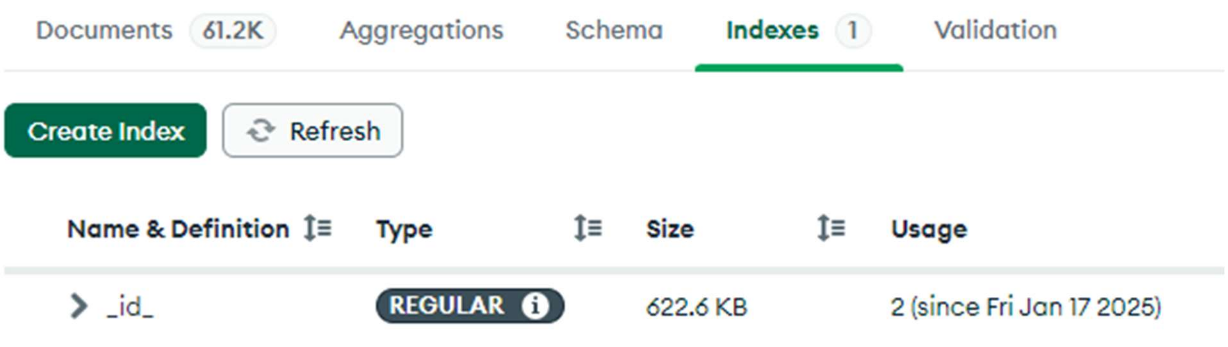
☐ Stop on errors

Specify Fields and Types   Learn more about data types

	✓ id	✓ ident	✓ type	✓ name	✓ latitude_d
	Int32	Mixed	String	String	Number
1	6523	00A	heliport	Total Rf Heliport	40.070800781
2	323361	00AA	small_airport	Aero B Ranch Airport	38.704022
3	6524	00AK	small_airport	Lowell Field	59.94919968
4	6525	00AL	small_airport	Epps Airpark	34.864799499
5	6526	00AR	closed	Newport Hospital & Clinic Heliport	35.6087

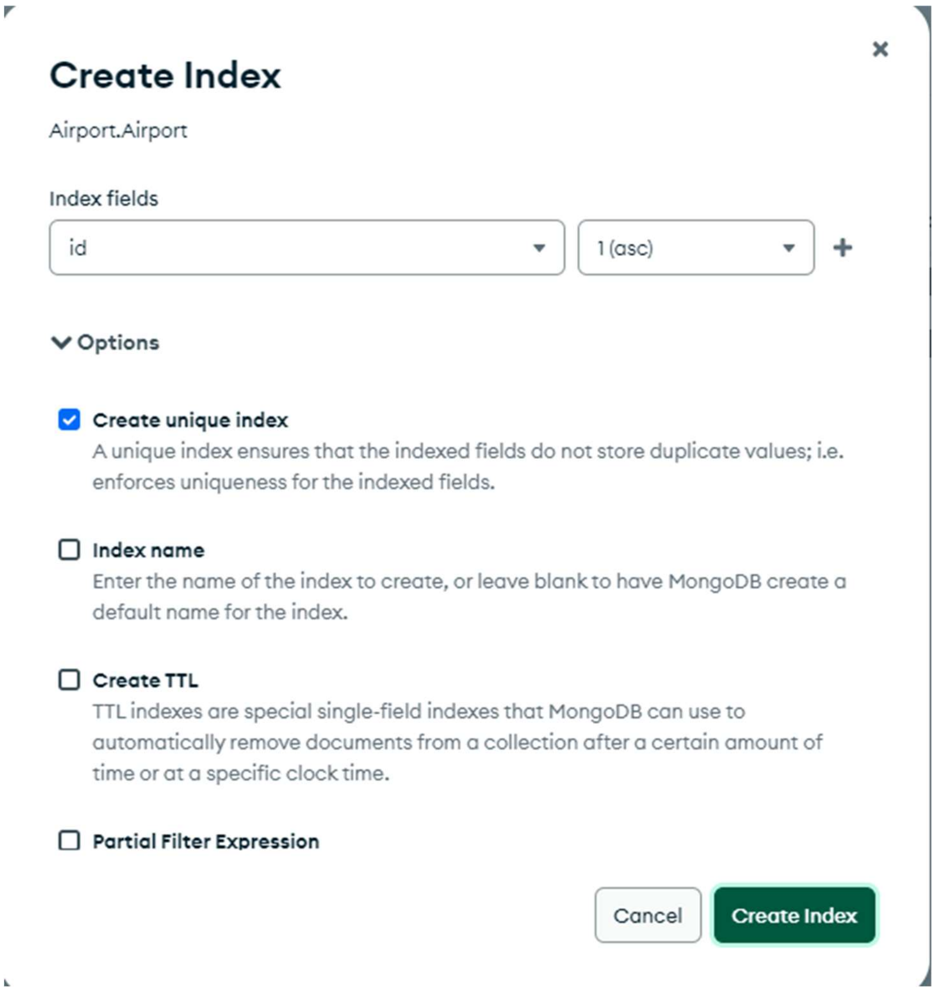


Import completed.
61221 documents imported.



Name & Definition	Type	Size	Usage
_id_	REGULAR	622.6 KB	2 (since Fri Jan 17 2025)

Unique Index in Ascending Order



Create Index

Airport.Airport

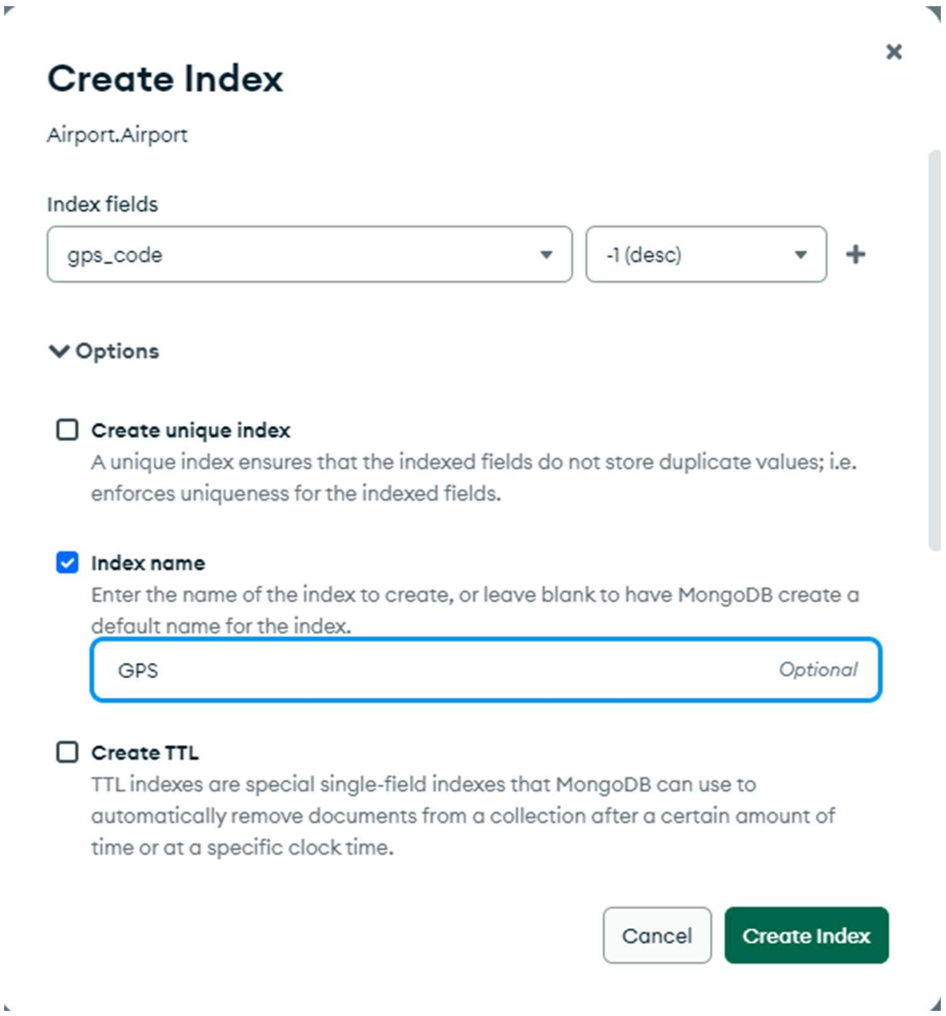
Index fields

id | 1 (asc)

Options

- [✓] **Create unique index**
  A unique index ensures that the indexed fields do not store duplicate values; i.e. enforces uniqueness for the indexed fields.
- [ ] **Index name**
  Enter the name of the index to create, or leave blank to have MongoDB create a default name for the index.
- [ ] **Create TTL**
  TTL indexes are special single-field indexes that MongoDB can use to automatically remove documents from a collection after a certain amount of time or at a specific clock time.
- [ ] **Partial Filter Expression**

Specific name Index in Descending Order



Create Index

Airport.Airport

Index fields

gps_code | -1 (desc)

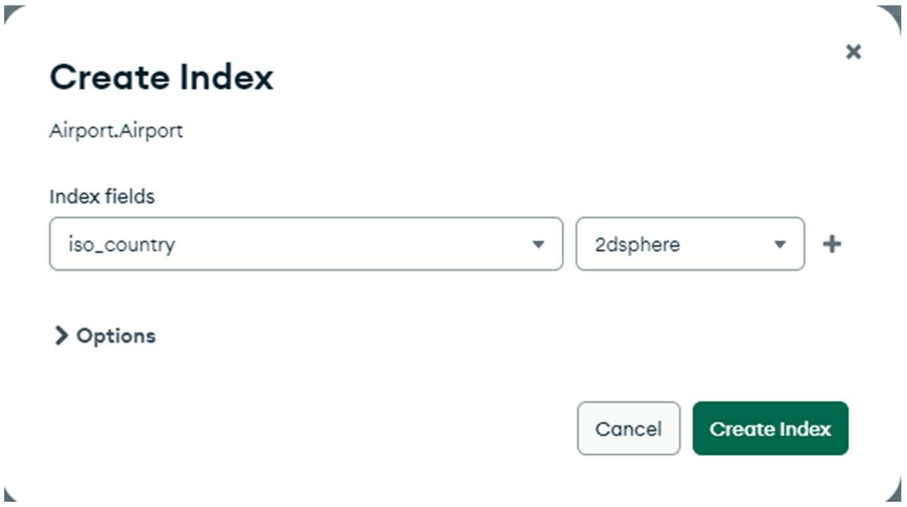
Options

☐ Create unique index
A unique index ensures that the indexed fields do not store duplicate values; i.e. enforces uniqueness for the indexed fields.

☑ Index name
Enter the name of the index to create, or leave blank to have MongoDB create a default name for the index.
GPS

☐ Create TTL
TTL indexes are special single-field indexes that MongoDB can use to automatically remove documents from a collection after a certain amount of time or at a specific clock time.

2D Sphere Index (Geospatial)



Create Index

Airport.Airport

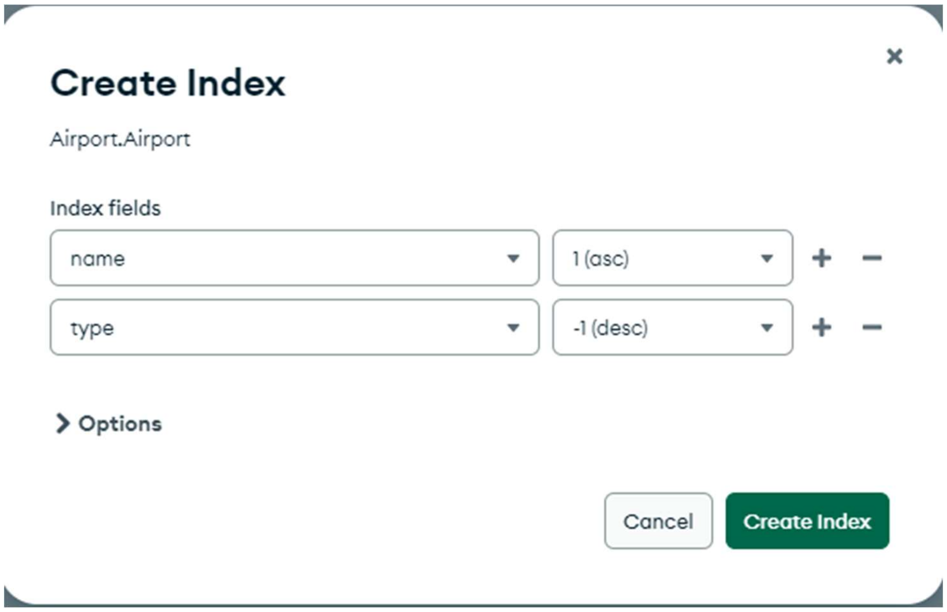
Index fields

iso_country | 2dsphere

Compound Index

Name in Ascending order

Type in Descending order



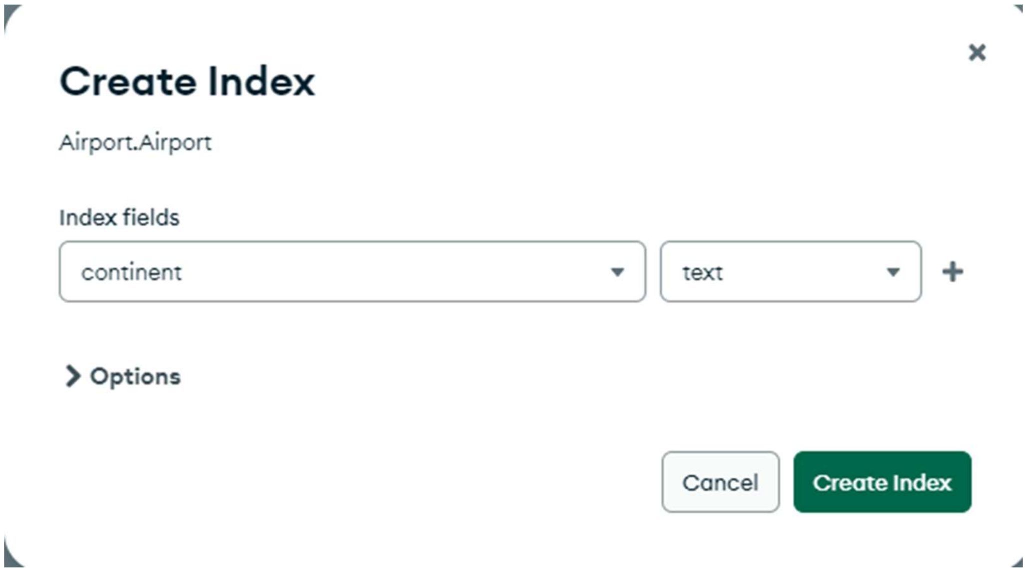
Create Index

Airport.Airport

Index fields

name	1 (asc)
type	-1 (desc)

Index with text type



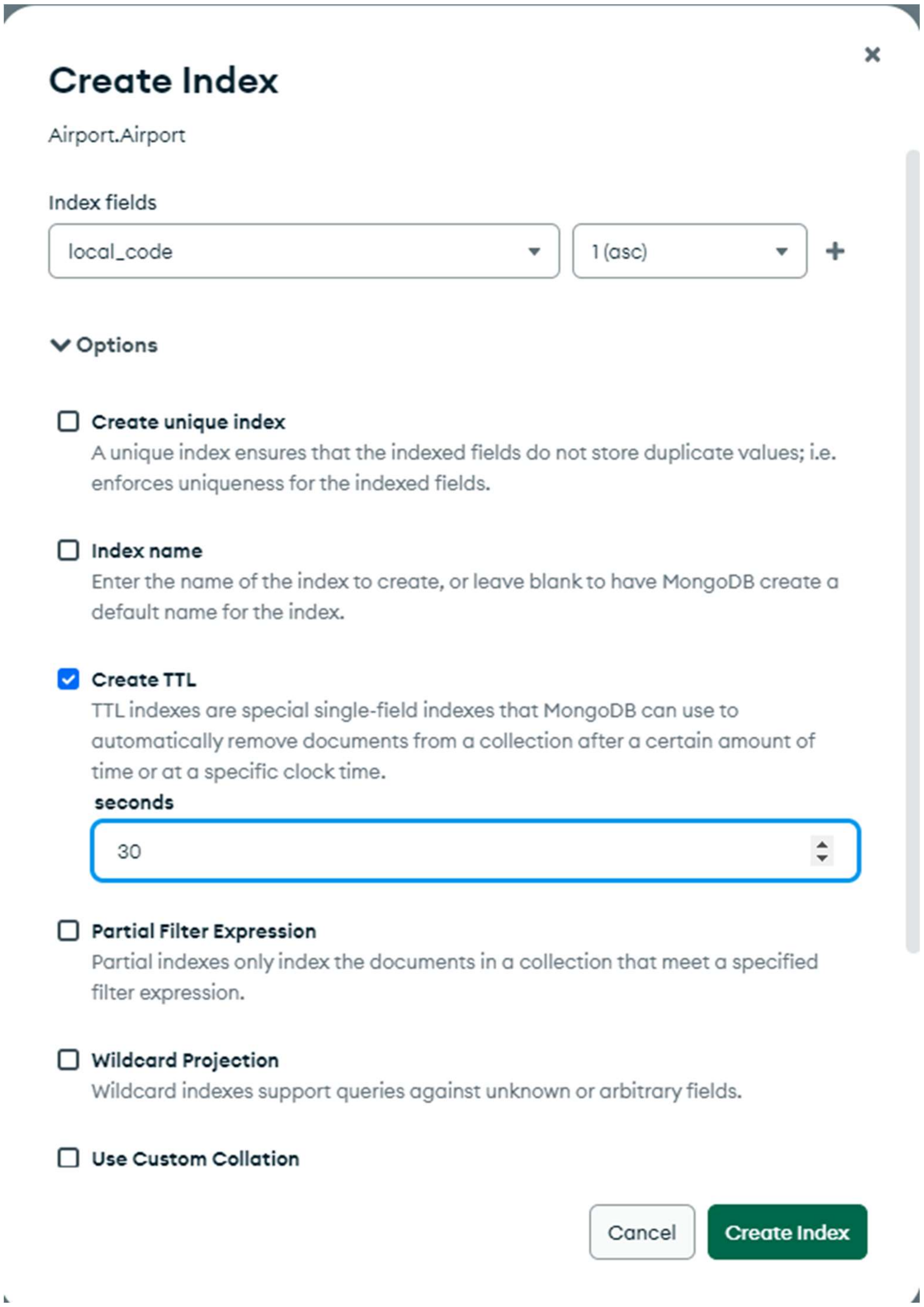
Create Index

Airport.Airport

Index fields

continent	text

TTL(Time to Live)(set 30 sec timer) Index



Create Index

Airport.Airport

Index fields

local_code | 1 (asc)

Options

[ ] Create unique index
A unique index ensures that the indexed fields do not store duplicate values; i.e. enforces uniqueness for the indexed fields.

[ ] Index name
Enter the name of the index to create, or leave blank to have MongoDB create a default name for the index.

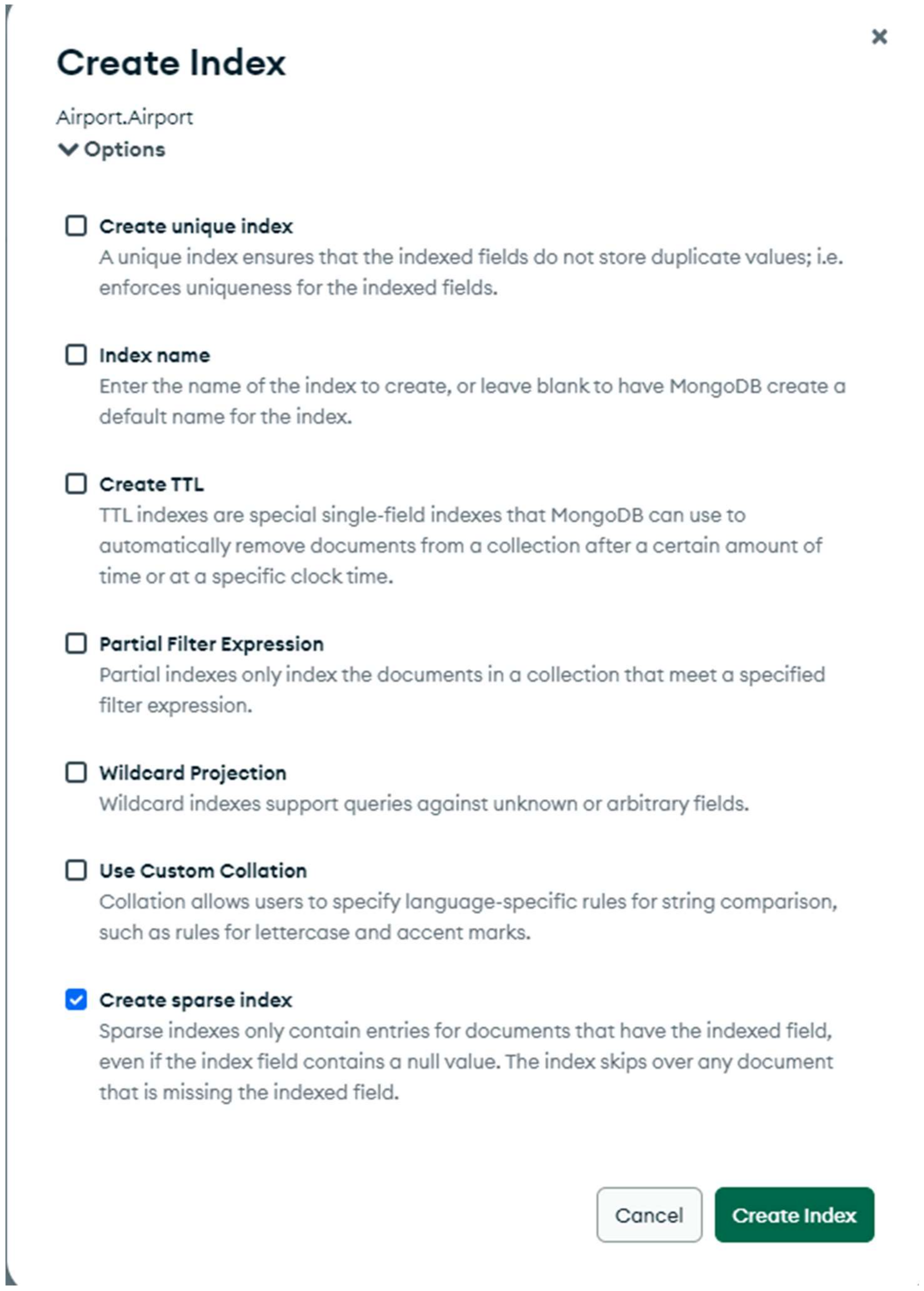
[✓] Create TTL
TTL indexes are special single-field indexes that MongoDB can use to automatically remove documents from a collection after a certain amount of time or at a specific clock time.
seconds
30

[ ] Partial Filter Expression
Partial indexes only index the documents in a collection that meet a specified filter expression.

[ ] Wildcard Projection
Wildcard indexes support queries against unknown or arbitrary fields.

[ ] Use Custom Collation

Sparse Index



Create Index

Airport.Airport

Options

☐ **Create unique index**
A unique index ensures that the indexed fields do not store duplicate values; i.e. enforces uniqueness for the indexed fields.

☐ **Index name**
Enter the name of the index to create, or leave blank to have MongoDB create a default name for the index.

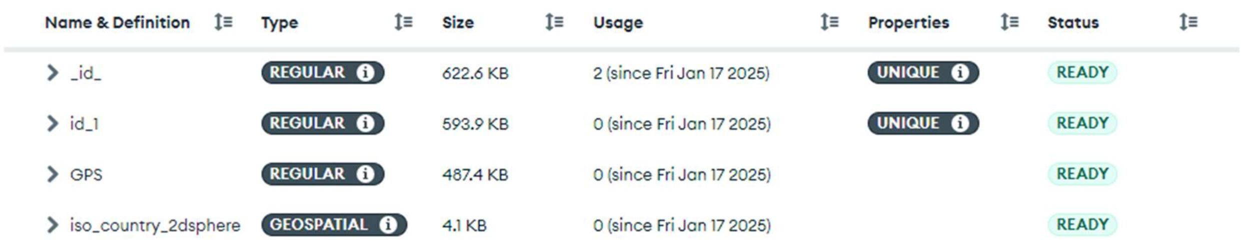
☐ **Create TTL**
TTL indexes are special single-field indexes that MongoDB can use to automatically remove documents from a collection after a certain amount of time or at a specific clock time.

☐ **Partial Filter Expression**
Partial indexes only index the documents in a collection that meet a specified filter expression.

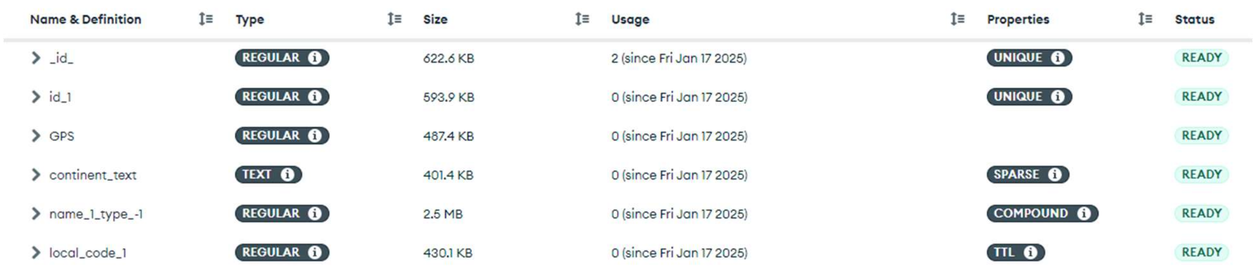
☐ **Wildcard Projection**
Wildcard indexes support queries against unknown or arbitrary fields.

☐ **Use Custom Collation**
Collation allows users to specify language-specific rules for string comparison, such as rules for lettercase and accent marks.

✓ **Create sparse index**
Sparse indexes only contain entries for documents that have the indexed field, even if the index field contains a null value. The index skips over any document that is missing the indexed field.



Name & Definition	Type	Size	Usage	Properties	Status
_id_	REGULAR	622.6 KB	2 (since Fri Jan 17 2025)	UNIQUE	READY
id_1	REGULAR	593.9 KB	0 (since Fri Jan 17 2025)	UNIQUE	READY
GPS	REGULAR	487.4 KB	0 (since Fri Jan 17 2025)		READY
iso_country_2dsphere	GEOSPATIAL	4.1 KB	0 (since Fri Jan 17 2025)		READY



Name & Definition	Type	Size	Usage	Properties	Status
_id_	REGULAR	622.6 KB	2 (since Fri Jan 17 2025)	UNIQUE	READY
id_1	REGULAR	593.9 KB	0 (since Fri Jan 17 2025)	UNIQUE	READY
GPS	REGULAR	487.4 KB	0 (since Fri Jan 17 2025)		READY
continent_text	TEXT	401.4 KB	0 (since Fri Jan 17 2025)	SPARSE	READY
name_1_type_-1	REGULAR	2.5 MB	0 (since Fri Jan 17 2025)	COMPOUND	READY
local_code_1	REGULAR	430.1 KB	0 (since Fri Jan 17 2025)	TTL	READY