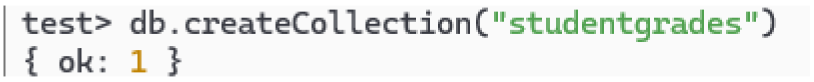
PRACTICAL 4 : Indexing using mongoDb

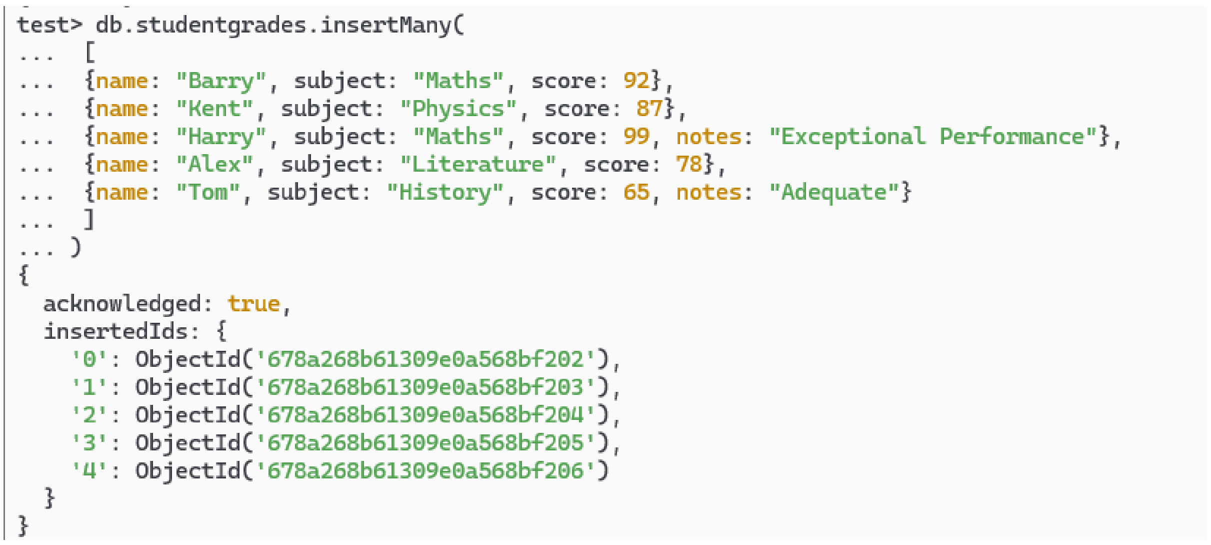
Use students

Creating collection



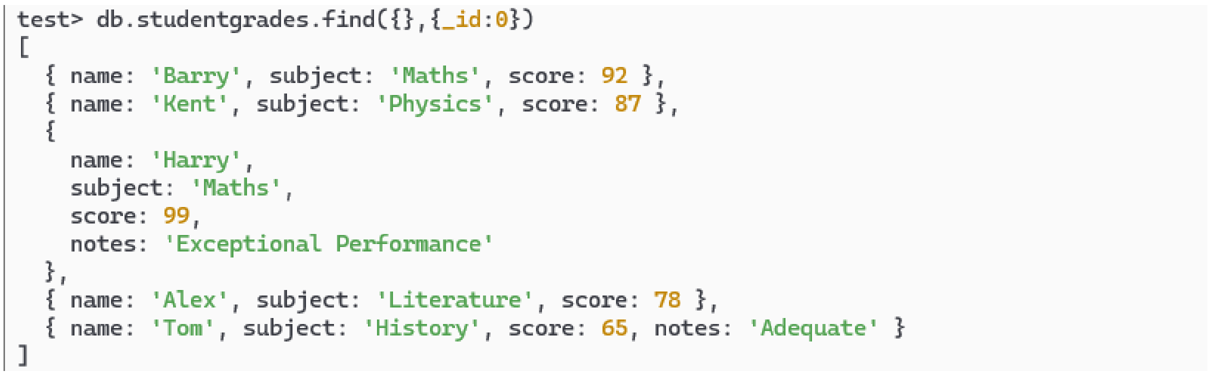
```
test> db.createCollection("studentgrades")
{ ok: 1 }
```

Inserting values



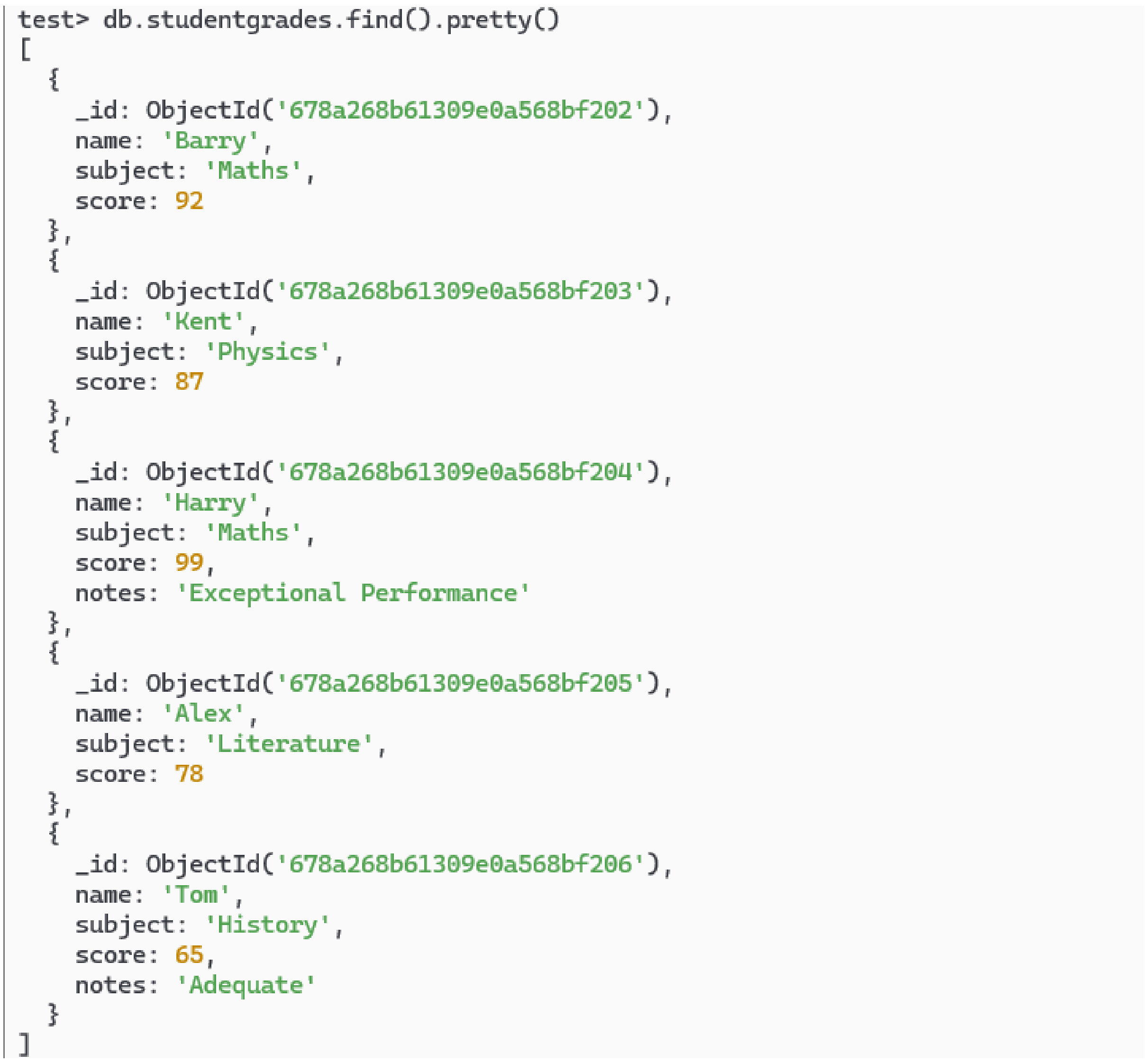
```
test> db.studentgrades.insertMany(
...   [
...   {name: "Barry", subject: "Maths", score: 92},
...   {name: "Kent", subject: "Physics", score: 87},
...   {name: "Harry", subject: "Maths", score: 99, notes: "Exceptional Performance"},
...   {name: "Alex", subject: "Literature", score: 78},
...   {name: "Tom", subject: "History", score: 65, notes: "Adequate"}
...   ]
... )
{
  acknowledged: true,
  insertedIds: {
    '0': ObjectId('678a268b61309e0a568bf202'),
    '1': ObjectId('678a268b61309e0a568bf203'),
    '2': ObjectId('678a268b61309e0a568bf204'),
    '3': ObjectId('678a268b61309e0a568bf205'),
    '4': ObjectId('678a268b61309e0a568bf206')
  }
}
```

Finding values



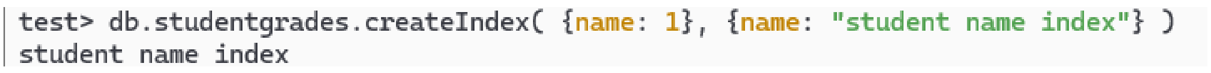
```
test> db.studentgrades.find({},{_id:0})
[
  { name: 'Barry', subject: 'Maths', score: 92 },
  { name: 'Kent', subject: 'Physics', score: 87 },
  {
    name: 'Harry',
    subject: 'Maths',
    score: 99,
    notes: 'Exceptional Performance'
  },
  { name: 'Alex', subject: 'Literature', score: 78 },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

PRACTICAL 4 : Indexing using mongoDb



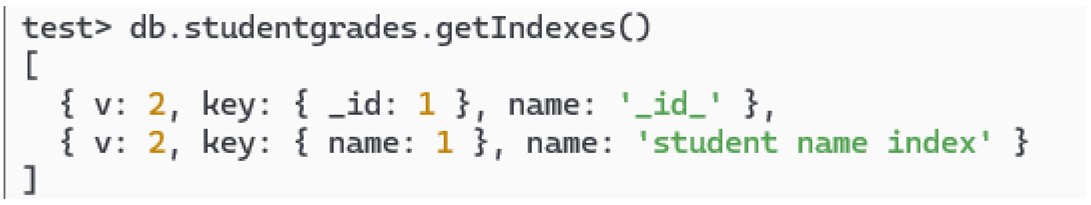
```
test> db.studentgrades.find().pretty()
[
  {
    _id: ObjectId('678a268b61309e0a568bf202'),
    name: 'Barry',
    subject: 'Maths',
    score: 92
  },
  {
    _id: ObjectId('678a268b61309e0a568bf203'),
    name: 'Kent',
    subject: 'Physics',
    score: 87
  },
  {
    _id: ObjectId('678a268b61309e0a568bf204'),
    name: 'Harry',
    subject: 'Maths',
    score: 99,
    notes: 'Exceptional Performance'
  },
  {
    _id: ObjectId('678a268b61309e0a568bf205'),
    name: 'Alex',
    subject: 'Literature',
    score: 78
  },
  {
    _id: ObjectId('678a268b61309e0a568bf206'),
    name: 'Tom',
    subject: 'History',
    score: 65,
    notes: 'Adequate'
  }
]
```

Creating indexes in mongoDB



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

Finding indexes in mongoDB



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

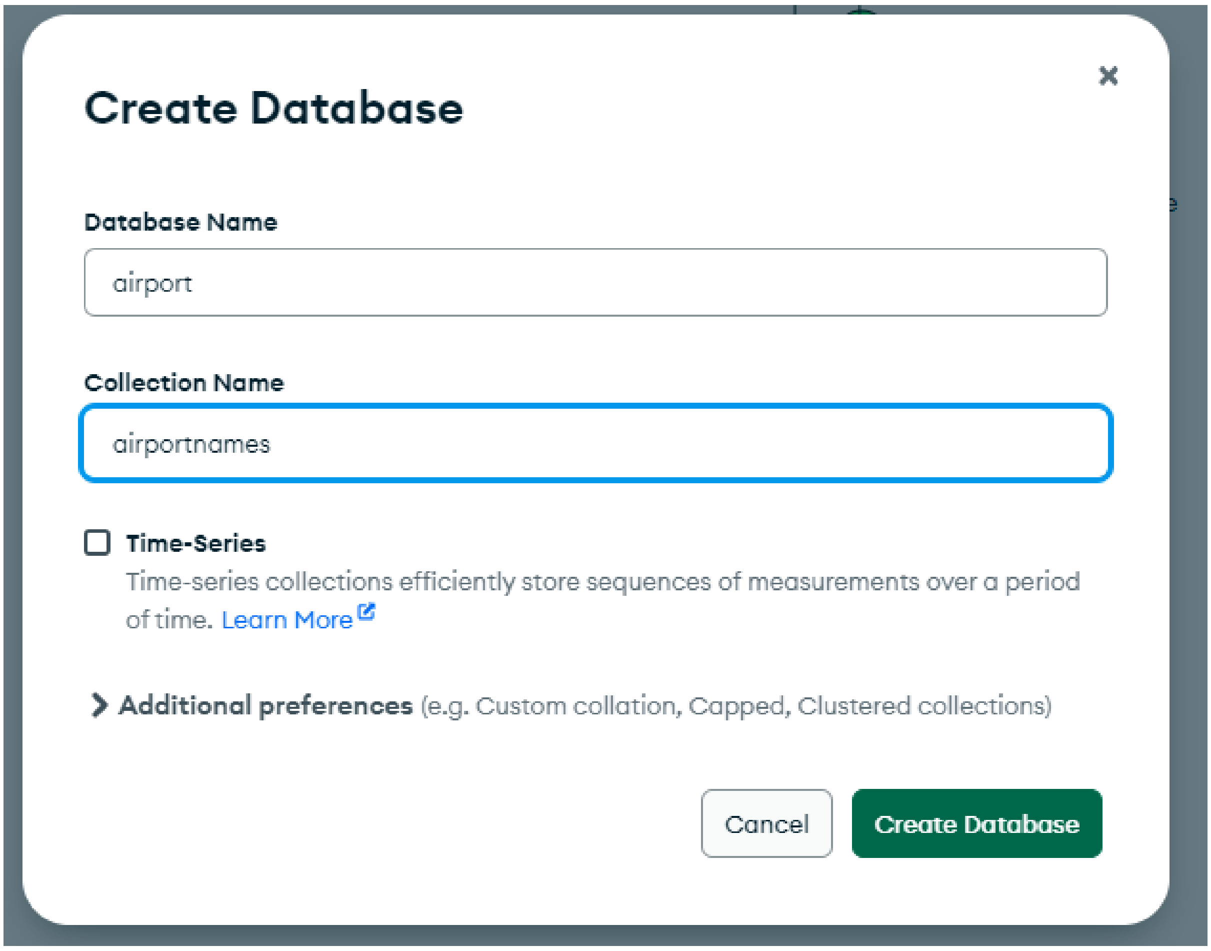
PRACTICAL 4 : Indexing using mongoDb

Drop indexes from collection

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

```
test>  db.studentgrades.createIndex( {name: 1}, {name: "student name index"} )
student name index
test> db.studentgrades.dropIndex({name:1})
{ nIndexesWas: 2, ok: 1 }
test> db.studentgrades.dropIndexes()
{
  nIndexesWas: 1,
  msg: 'non-_id indexes dropped for collection',
  ok: 1
}
```

Q2.



Create Database

Database Name: airport

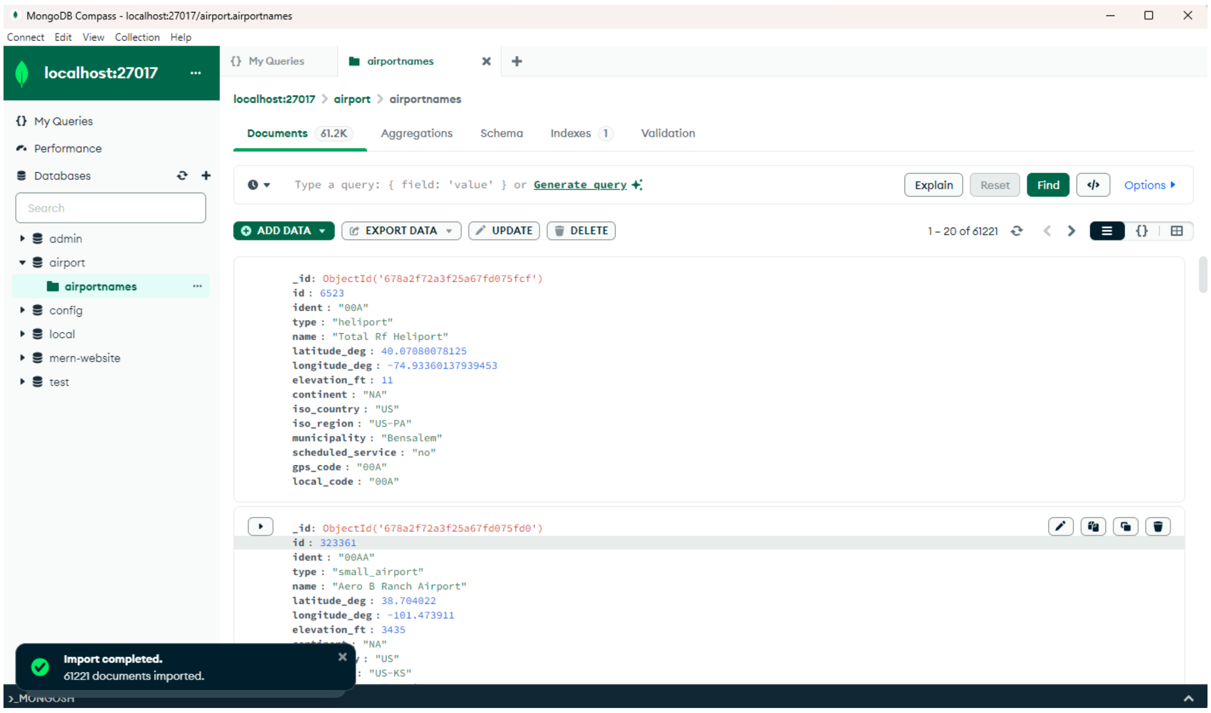
Collection Name: airportnames

☐ 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)

PRACTICAL 4 : Indexing using mongoDb

Importing Data

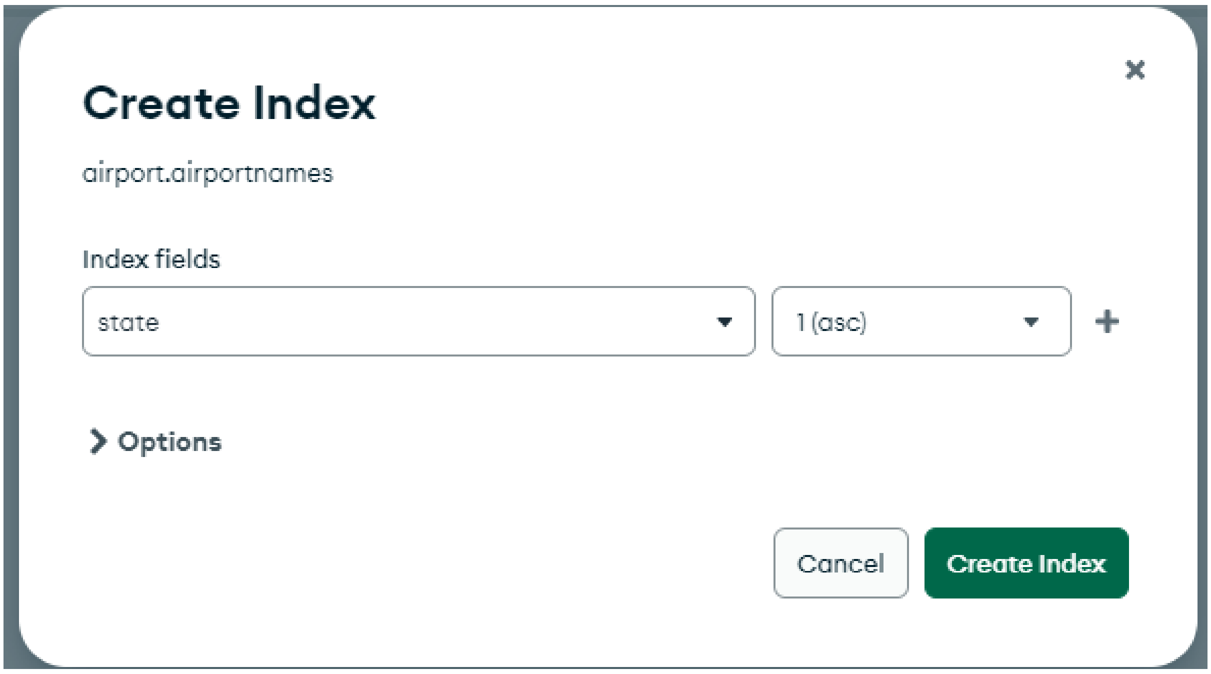


```
_id: ObjectId('678a2f72a3f25a67fd075fcf')
id : 6523
ident : "00A"
type : "heliport"
name : "Total Rf Heliport"
latitude_deg : 40.07080078125
longitude_deg : -74.93360137939453
elevation_ft : 11
continent : "NA"
iso_country : "US"
iso_region : "US-PA"
municipality : "Bensalem"
scheduled_service : "no"
gps_code : "00A"
local_code : "00A"
```

```
_id: ObjectId('678a2f72a3f25a67fd075fd0')
id : 323361
ident : "00AA"
type : "small_airport"
name : "Aero B Ranch Airport"
latitude_deg : 38.704022
longitude_deg : -101.473911
elevation_ft : 3435
continent : "NA"
            : "US"
            : "US-KS"
```

Import completed.
61221 documents imported.

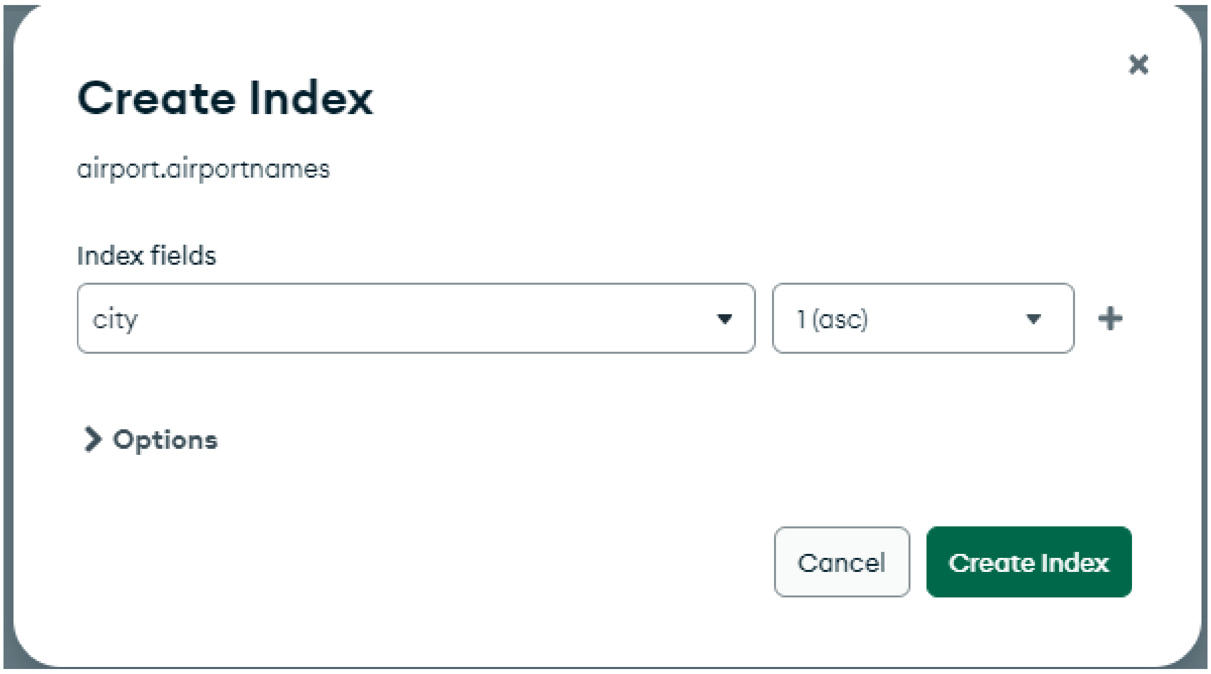
1. **Single-Field Index on state**



Index fields: state — 1 (asc)

PRACTICAL 4 : Indexing using mongoDb

2. **Single-Field Index on** city



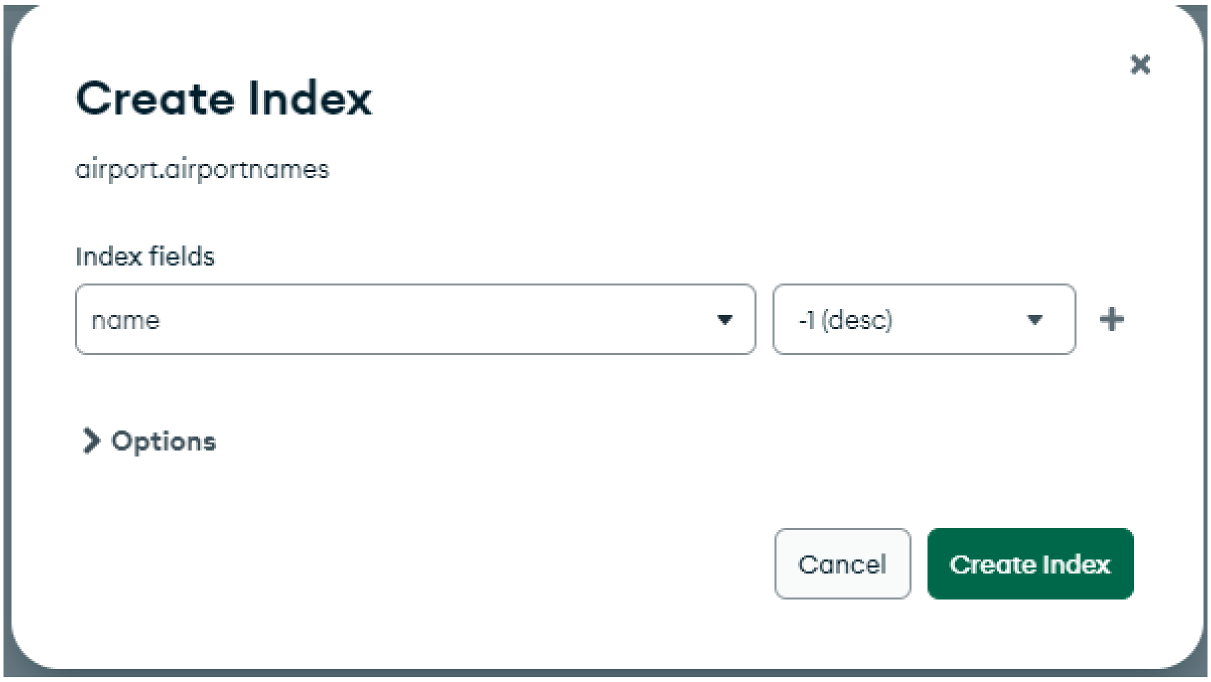
Create Index

airport.airportnames

Index fields

city | 1 (asc)

3. Single-Field Index on name



Create Index

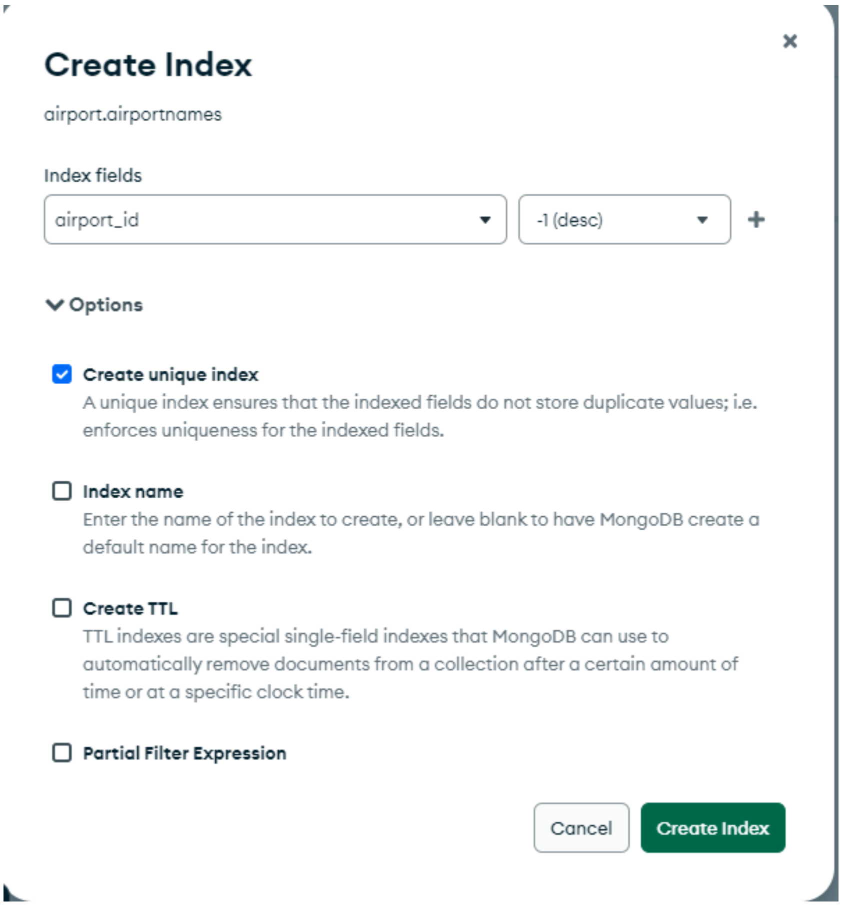
airport.airportnames

Index fields

name | -1 (desc)

PRACTICAL 4 : Indexing using mongoDb

4. Unique Index on airport_id



Create Index

airport.airportnames

Index fields

airport_id | -1 (desc)

Options

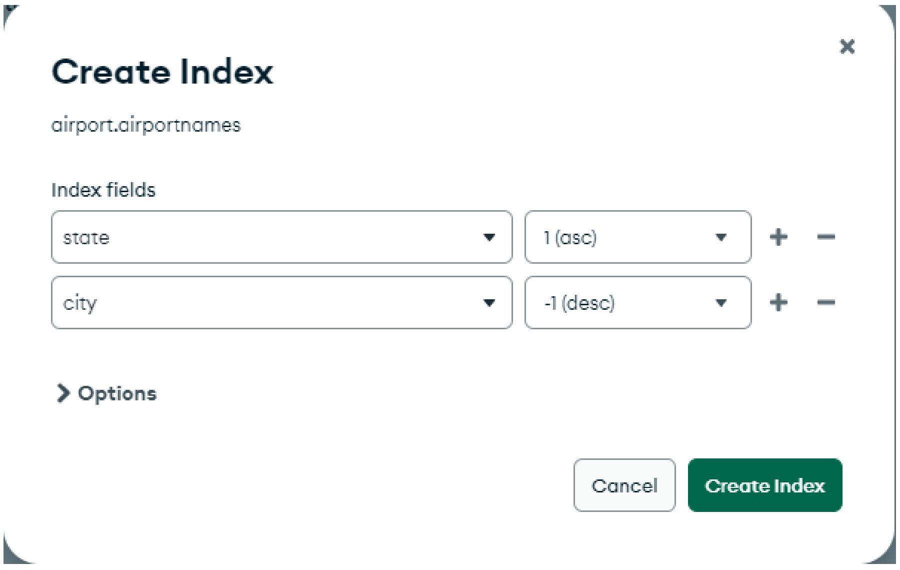
[✓] Create unique index
A unique index ensures that the indexed fields do not store duplicate values; i.e. enforces uniqueness for the indexed fields.

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

[ ] 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

Compound index on state,city



Create Index

airport.airportnames

Index fields

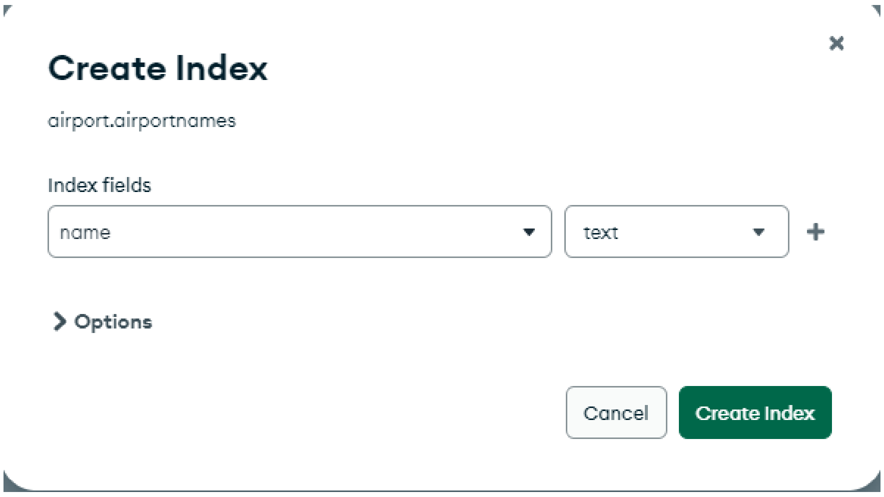
state | 1 (asc)

city | -1 (desc)

Options

PRACTICAL 4 : Indexing using mongoDb

6. **Text Index on** name



Create Index

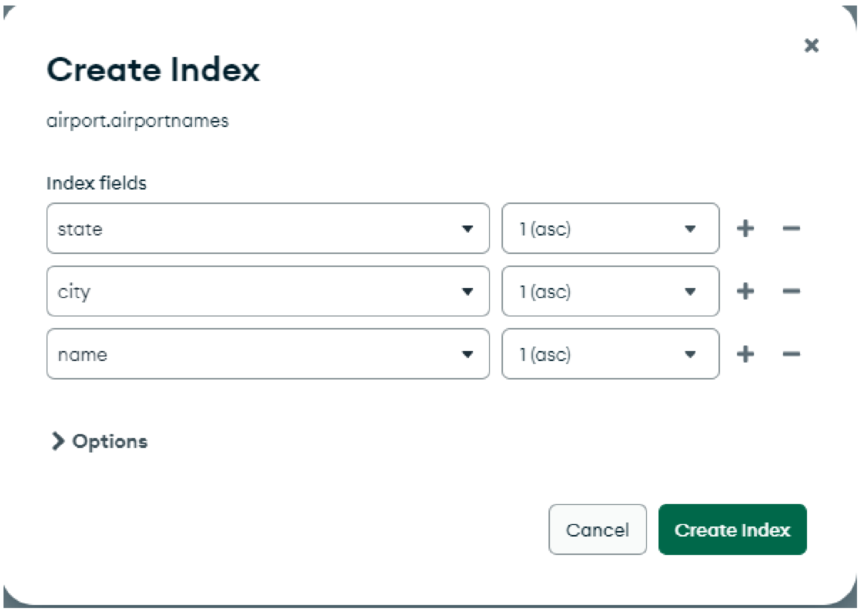
airport.airportnames

Index fields

name | text

7. Multi-Field Compound Indexes

Index on state, city, and name



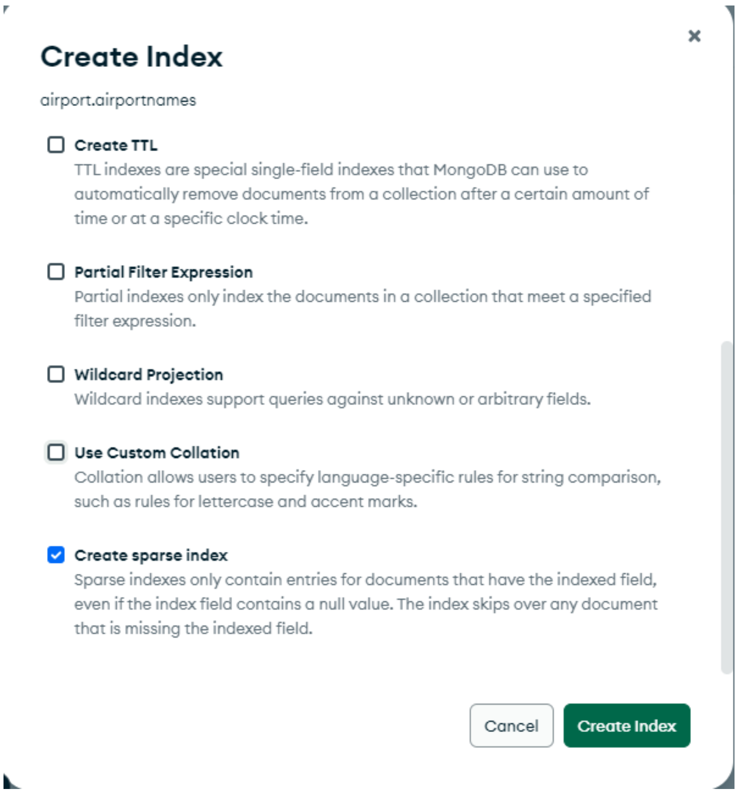
Create Index

airport.airportnames

Index fields

state	1 (asc)
city	1 (asc)
name	1 (asc)

8. Spare index



Create Index

airport.airportnames

☐ 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.

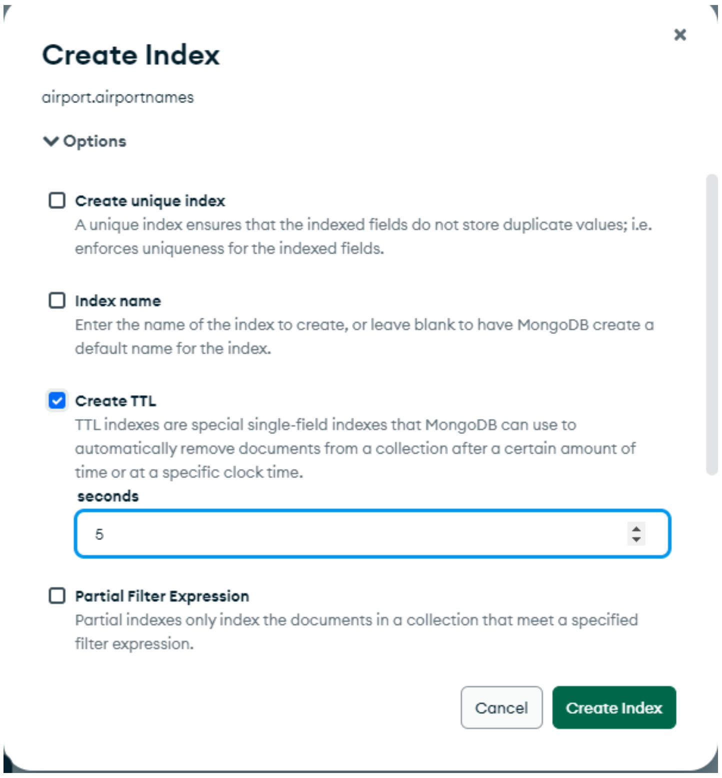
☐ Partial Filter Expression
Partial indexes only index the documents in a collection that meet a specified filter expression.

☐ 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.

9. TTL index



Create Index

airport.airportnames

Options

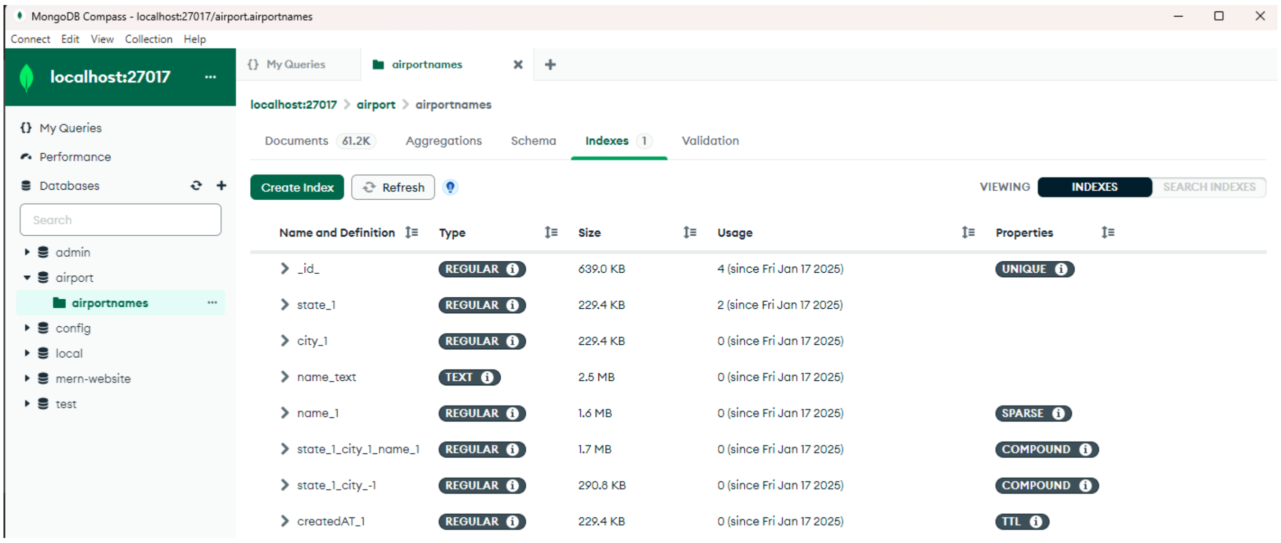
☐ Create unique index
A unique index ensures that the indexed fields do not store duplicate values; i.e. enforces uniqueness for the indexed fields.

☐ 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
5

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

PRACTICAL 4 : Indexing using mongoDb



Name and Definition	Type	Size	Usage	Properties
_id_	REGULAR	639.0 KB	4 (since Fri Jan 17 2025)	UNIQUE
state_1	REGULAR	229.4 KB	2 (since Fri Jan 17 2025)	
city_1	REGULAR	229.4 KB	0 (since Fri Jan 17 2025)	
name_text	TEXT	2.5 MB	0 (since Fri Jan 17 2025)	
name_1	REGULAR	1.6 MB	0 (since Fri Jan 17 2025)	SPARSE
state_1_city_1_name_1	REGULAR	1.7 MB	0 (since Fri Jan 17 2025)	COMPOUND
state_1_city_-1	REGULAR	290.8 KB	0 (since Fri Jan 17 2025)	COMPOUND
createdAT_1	REGULAR	229.4 KB	0 (since Fri Jan 17 2025)	TTL