

MongoDB

dr Sara Migliorini

NoSQL Systems – MongoDB

- MongoDB è un sistema per la gestione di basi di dati NoSQL di tipo document-based.
 - Una base di dati è formata da **collezioni di documenti**.
 - Ciascuna **collezione** corrisponde al concetto di **tabella** (o entità).
 - Ciascuna collezione è formata da insiemi di **documenti (tuple)**.
- Altre tipologie di database NoSQL
 - Graph database (e.g. Neo4j)
 - Column-oriented database (e.g. Apache Cassandra)
 - Key-value database (e.g. Amazon DynamoDB)
 - Document-oriented database (e.g. MongoDB)

MongoDB: Caratteristiche

- Struttura a documenti che ricorda l'organizzazione ad oggetti dei linguaggi di programmazione moderni
- Flessibilità e assenza di schema (schema-less)
 - Dati la cui struttura evolve nel tempo
 - Dati non strutturati o semi-strutturati
- Scalabilità e performance
 - Fondamentalmente MongoDB è un **sistema distribuito**
 - La parte server è stata sviluppata pensando che la configurazione di default prevedesse un **cluster (replica)**
 - Ridondanza e alta disponibilità
- Linguaggio di interrogazione e indici per semplificare e velocizzare operazioni comuni, come `findOne()` e `updateOne()`

MongoDB: Tools

Administrative tools: per la gestione dei database

- MongoDB shell (mongosh): da riga di comando
- MongoDB Compass: interfaccia grafica
 - <https://www.mongodb.com/try/download/compass>

Backup e restore: per la creazione o ripristino di backup

- mongodbdump: per creare un backup binario
- mongodbrestore: per il ripristino di un backup

MongoDB: Tools

Performance and monitoring tools: per verificare le prestazioni del cluster e monitorare lo stato di salute del sistema

- mongostat: statistiche sulle prestazioni
- mongotop: monitorare il tempo richiesta dalle istanze per eseguire operazioni di lettura e scrittura

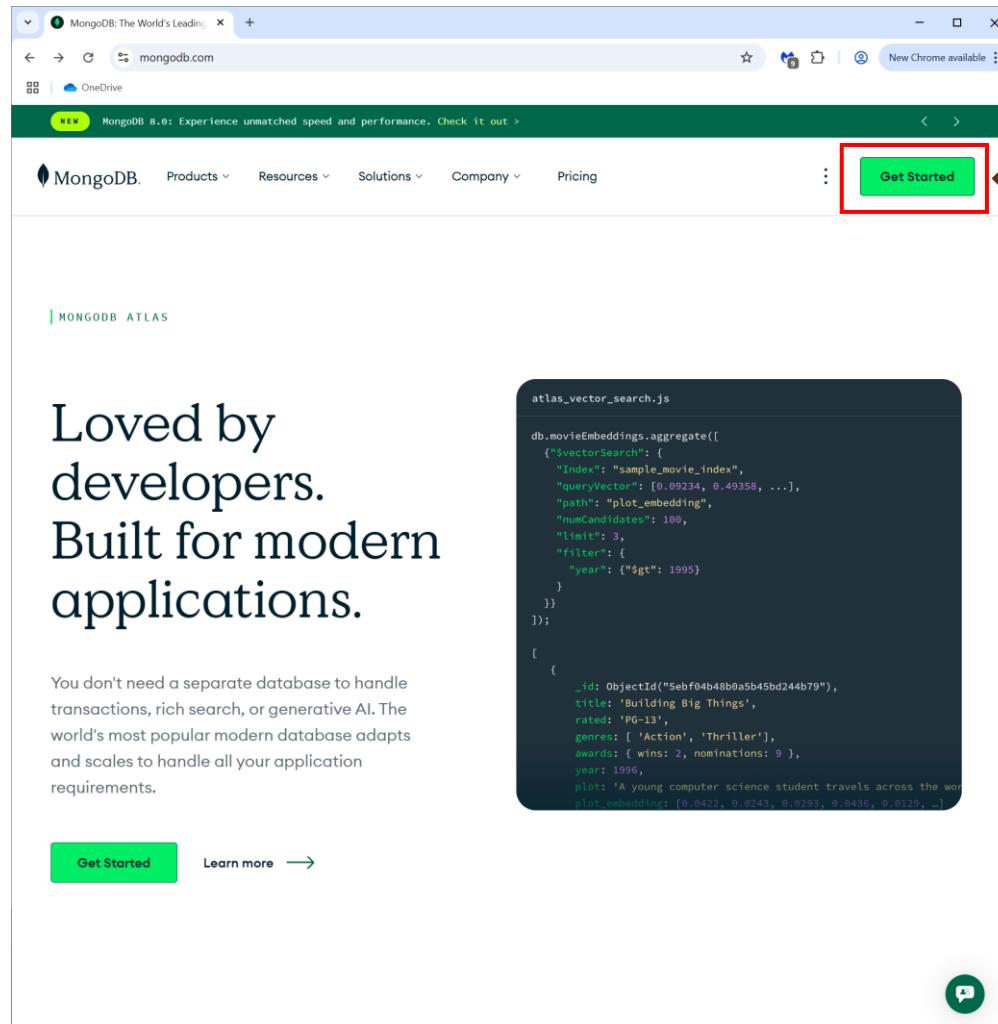
Developer and integration tools

- MongoDB per Visual Studio Code
- ...

MongoDB: Installazione

- MongoDB cluster:
 - Scaricare ed installare MongoDB Community Edition
 - <https://www.mongodb.com/try/download/community>
 - Creare un account su MongoDB Atlas ed utilizzare il cluster gratuito
 - <https://www.mongodb.com/try>

MongoDB: Cluster su Atlas

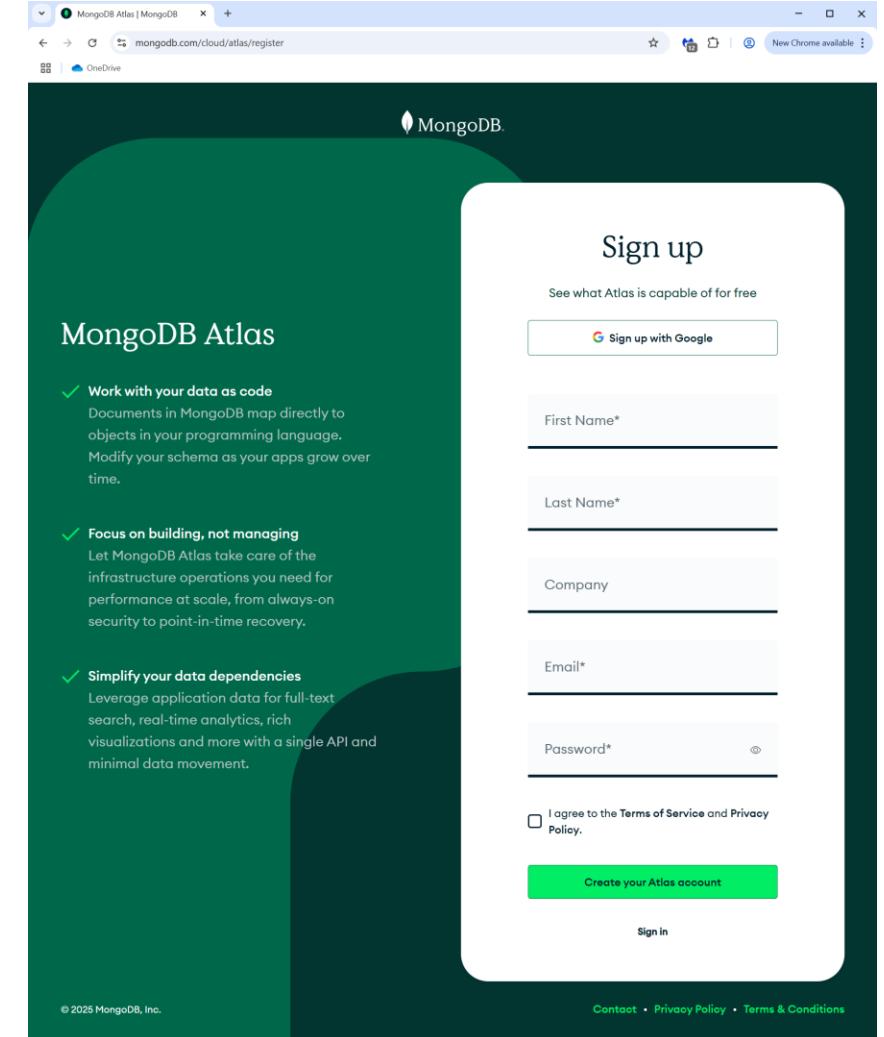


Loved by developers. Built for modern applications.

You don't need a separate database to handle transactions, rich search, or generative AI. The world's most popular modern database adapts and scales to handle all your application requirements.

[Get Started](#) [Learn more →](#)

```
atlas_vector_search.js
db.movieEmbeddings.aggregate([
  {
    "$vectorSearch": {
      "Index": "sample_movie_index",
      "queryVector": [0.09234, 0.49358, ...],
      "path": "plot_embedding",
      "numCandidates": 100,
      "limit": 3,
      "filter": {
        "year": {"$gt": 1995}
      }
    }
  },
  [
    {
      _id: ObjectId("5ebf04b48b0a5b45bd244b79"),
      title: "Building Big Things",
      rated: 'PG-13',
      genres: ['Action', 'Thriller'],
      awards: { wins: 2, nominations: 9 },
      year: 1996,
      plot: 'A young computer science student travels across the world embedding: [0.0422, 0.0243, 0.0293, 0.0436, 0.0129, ...]
    }
  ]
])
```



MongoDB Atlas

See what Atlas is capable of for free

[Sign up with Google](#)

Work with your data as code
Documents in MongoDB map directly to objects in your programming language. Modify your schema as your apps grow over time.

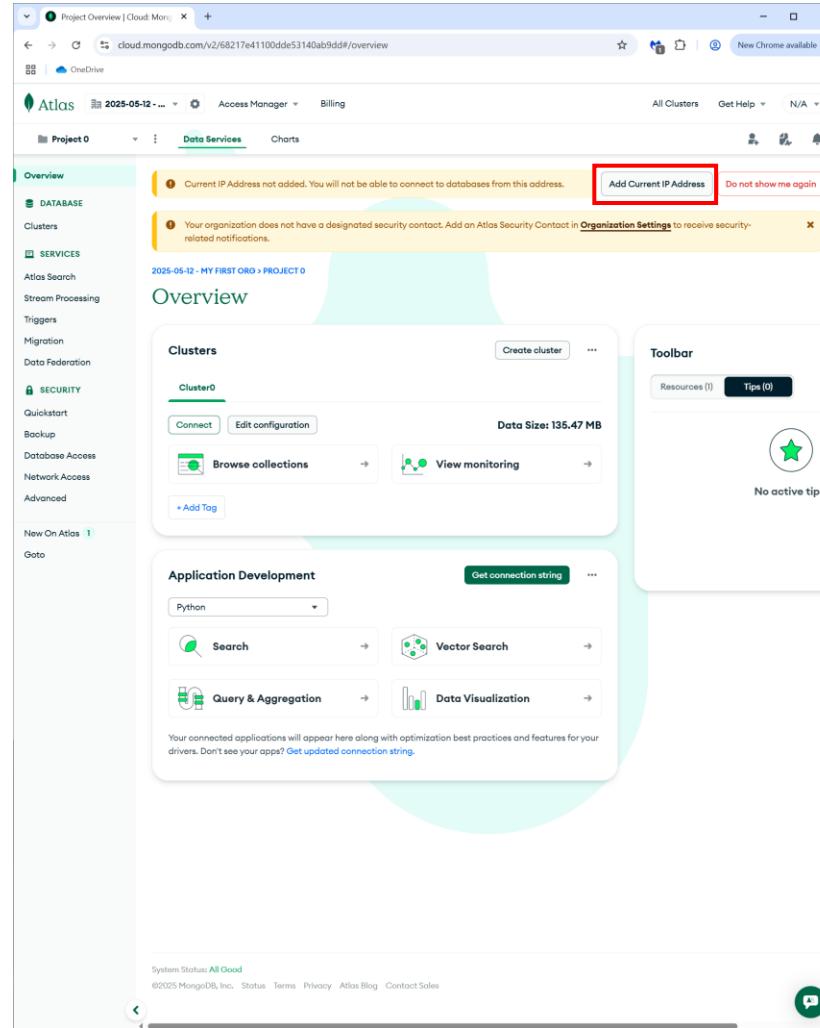
Focus on building, not managing
Let MongoDB Atlas take care of the infrastructure operations you need for performance at scale, from always-on security to point-in-time recovery.

Simplify your data dependencies
Leverage application data for full-text search, real-time analytics, rich visualizations and more with a single API and minimal data movement.

I agree to the [Terms of Service](#) and [Privacy Policy](#).

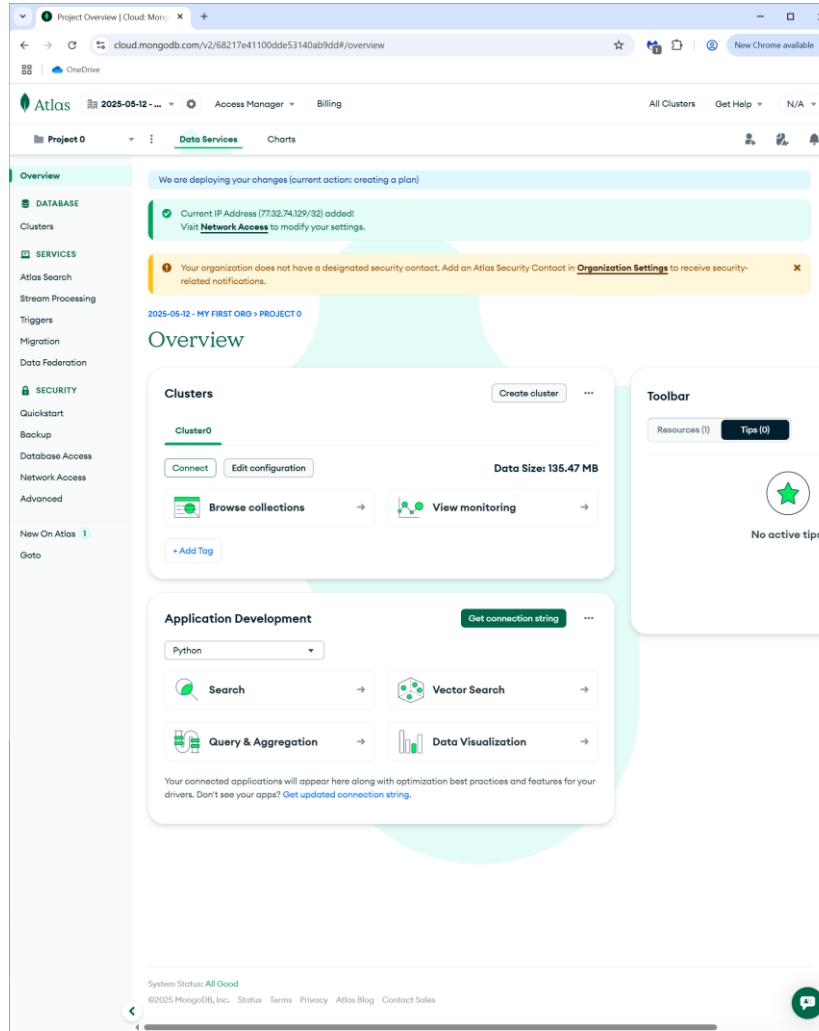
[Create your Atlas account](#) [Sign in](#)

MongoDB: Creazione Cluster su Atlas



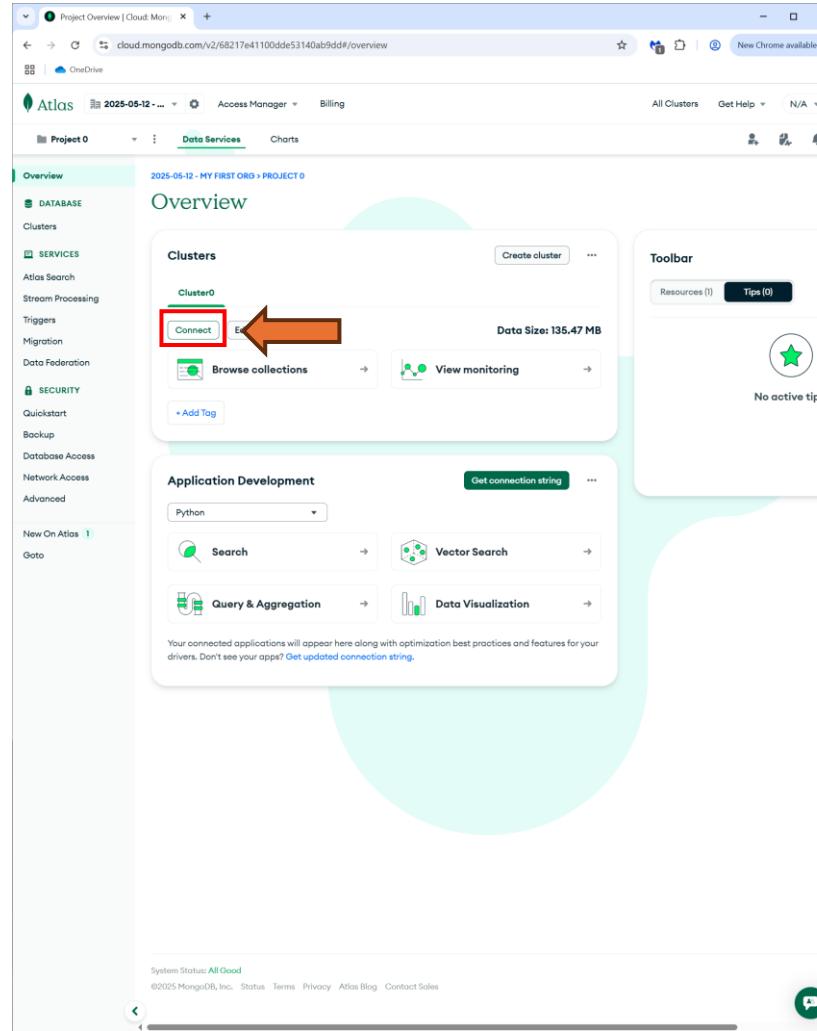
1. Aggiungere l'IP alla lista di quelli autorizzati alla connessione

MongoDB: Creazione Cluster su Atlas



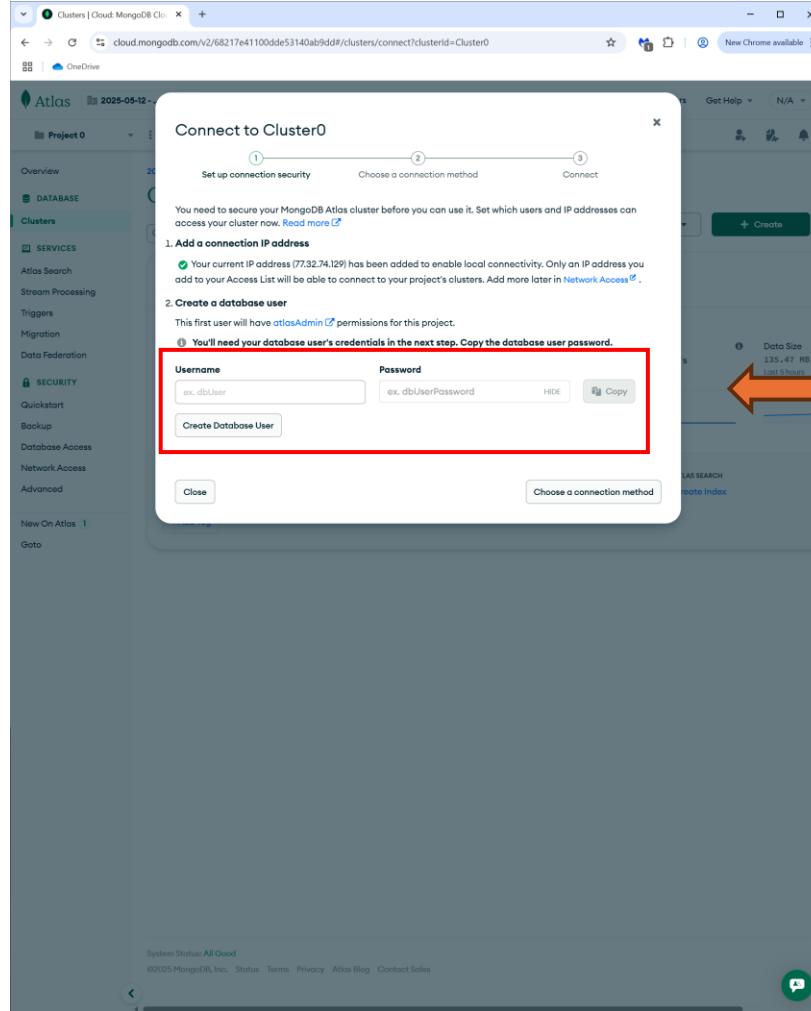
1. Aggiungere l'IP alla lista di quelli autorizzati alla connessione
2. Completare le configurazioni di sicurezza aggiungendo un'email di contatto

MongoDB: Creazione Cluster su Atlas

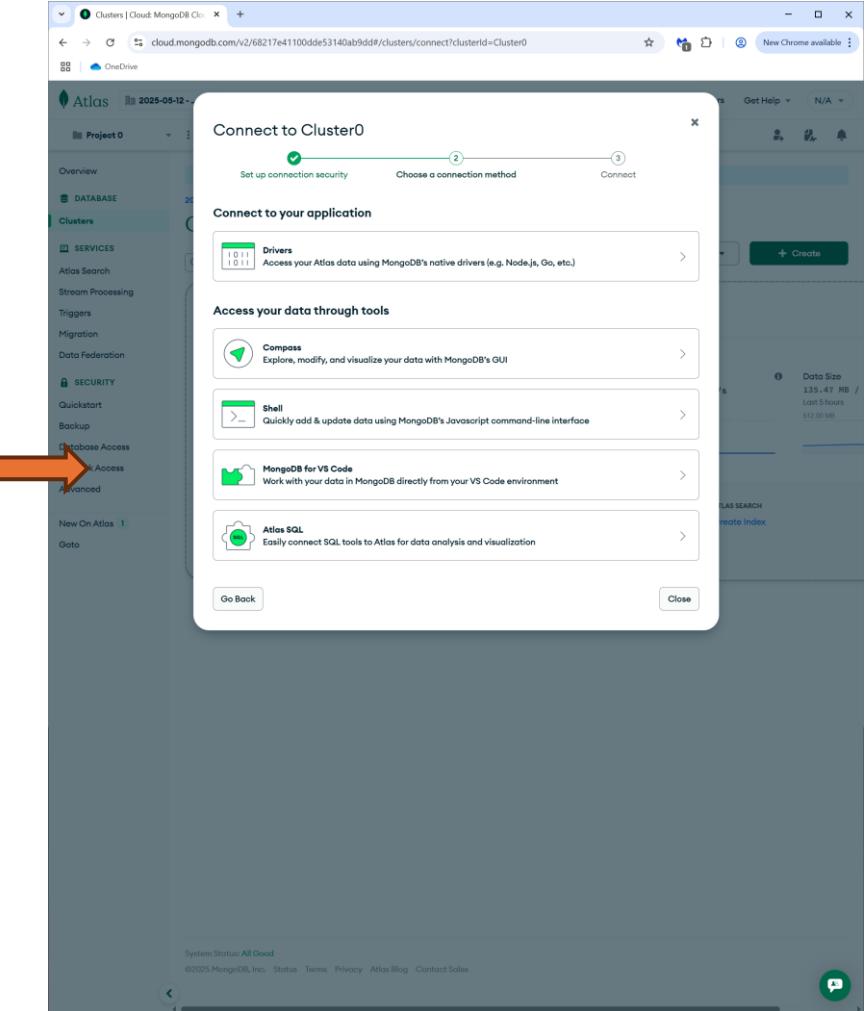


1. Aggiungere l'IP alla lista di quelli autorizzati alla connessione
2. Completare le configurazioni di sicurezza aggiungendo un'email di contatto
3. Connessione

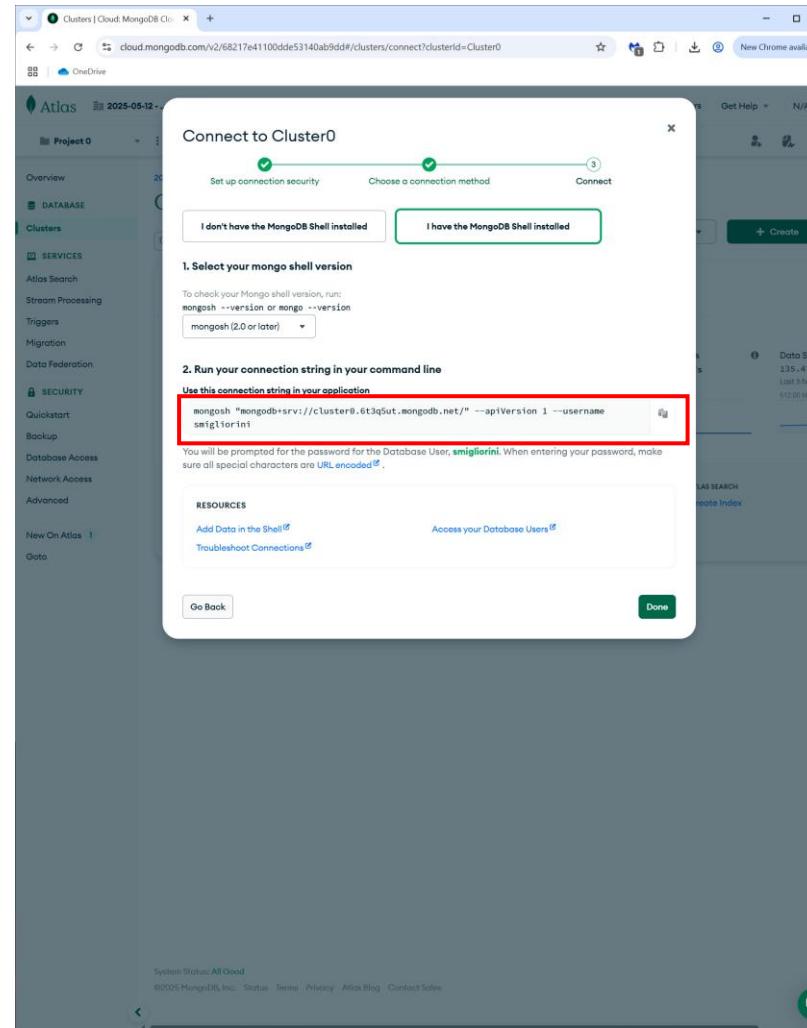
MongoDB: Creazione DB e Connessione



3a. Creazione utente
3b. Scelta del client



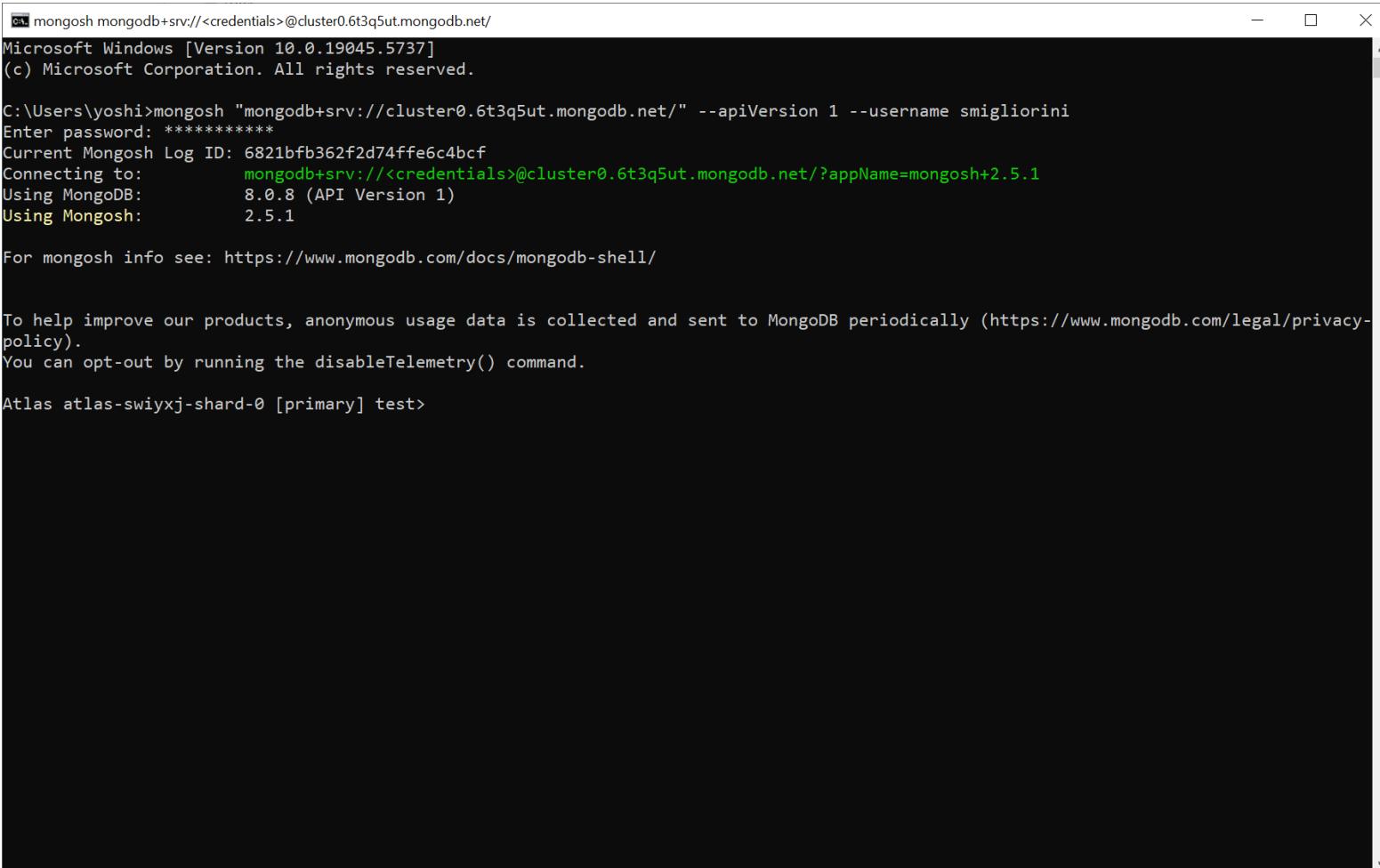
Connessione a MongoDB tramite mongosh



Connessione a MongoDB tramite mongosh

- La shell mongosh può essere installata in vario modo:
 - Download al link: <https://www.mongodb.com/try/download/shell>
 - Usare NPM (Node Package Manager) se Node.js è installato nel Sistema
 - Aprire il terminale
 - npm install -g mongosh
- Connessione ad un server MongoDB
 - Comando generico:
 - mongosh "mongodb://username:password@hostname:port/test"
 - Cluster Altas (copia ed incolla dall'interfaccia web):
 - mongosh "mongodb+srv://cluster0.6t3q5ut.mongodb.net/" --apiVersion 1 --username smigliorini

Connessione a MongoDB tramite mongosh



```
mongosh mongodb+srv://<credentials>@cluster0.6t3q5ut.mongodb.net/
Microsoft Windows [Version 10.0.19045.5737]
(c) Microsoft Corporation. All rights reserved.

C:\Users\yoshi>mongosh "mongodb+srv://cluster0.6t3q5ut.mongodb.net/" --apiVersion 1 --username smigliorini
Enter password: *****
Current Mongosh Log ID: 6821bfb362f2d74ffe6c4bcf
Connecting to:      mongodb+srv://<credentials>@cluster0.6t3q5ut.mongodb.net/?appName=mongosh+2.5.1
Using MongoDB:     8.0.8 (API Version 1)
Using Mongosh:     2.5.1

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

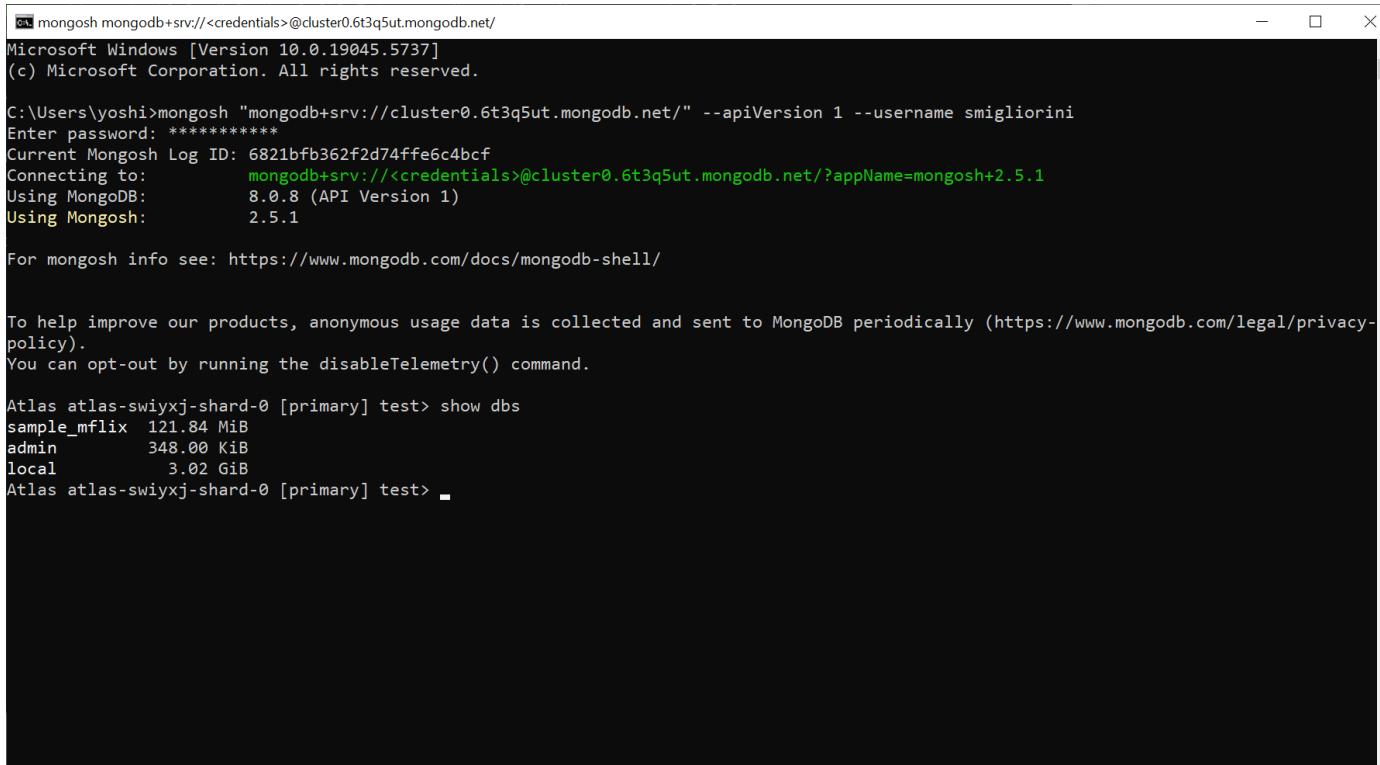
To help improve our products, anonymous usage data is collected and sent to MongoDB periodically (https://www.mongodb.com/legal/privacy-policy).
You can opt-out by running the disableTelemetry() command.

Atlas atlas-swiyxj-shard-0 [primary] test>
```

Comandi principali di mongosh

Mostrare tutti i database

- test> show dbs



```
mongosh mongodb+srv://<credentials>@cluster0.6t3q5ut.mongodb.net/
Microsoft Windows [Version 10.0.19045.5737]
(c) Microsoft Corporation. All rights reserved.

C:\Users\yoshi>mongosh "mongodb+srv://cluster0.6t3q5ut.mongodb.net/" --apiVersion 1 --username smigliorini
Enter password: *****
Current Mongosh Log ID: 6821bfb362f2d74ffe6c4bcf
Connecting to:      mongodb+srv://<credentials>@cluster0.6t3q5ut.mongodb.net/?appName=mongosh+2.5.1
Using MongoDB:     8.0.8 (API Version 1)
Using Mongosh:     2.5.1

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

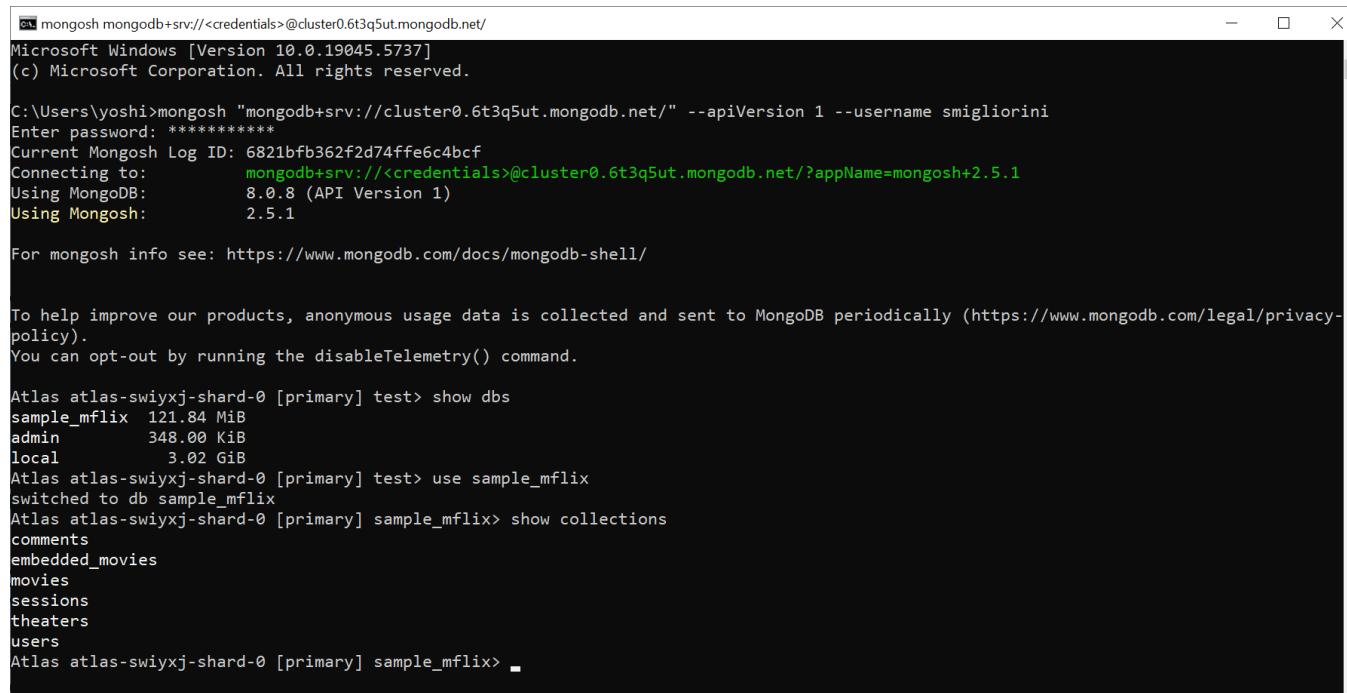
To help improve our products, anonymous usage data is collected and sent to MongoDB periodically (https://www.mongodb.com/legal/privacy-policy).
You can opt-out by running the disableTelemetry() command.

Atlas atlas-swiyxj-shard-0 [primary] test> show dbs
sample_mflix 121.84 MiB
admin       348.00 KiB
local       3.02 GiB
Atlas atlas-swiyxj-shard-0 [primary] test> -
```

Comandi principali di mongosh

Selezionare un particolare database su cui lavorare

- test> use sample_mflix
- switched to db sample_mflix



```
cmd mongosh mongodb+srv://<credentials>@cluster0.6t3q5ut.mongodb.net/
Microsoft Windows [Version 10.0.19045.5737]
(c) Microsoft Corporation. All rights reserved.

C:\Users\yoshi>mongosh "mongodb+srv://cluster0.6t3q5ut.mongodb.net/" --apiVersion 1 --username smigliorini
Enter password: *****
Current Mongosh Log ID: 6821bfb362f2d74ffe6c4bcf
Connecting to: mongodb+srv://<credentials>@cluster0.6t3q5ut.mongodb.net/?appName=mongosh+2.5.1
Using MongoDB: 8.0.8 (API Version 1)
Using Mongosh: 2.5.1

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

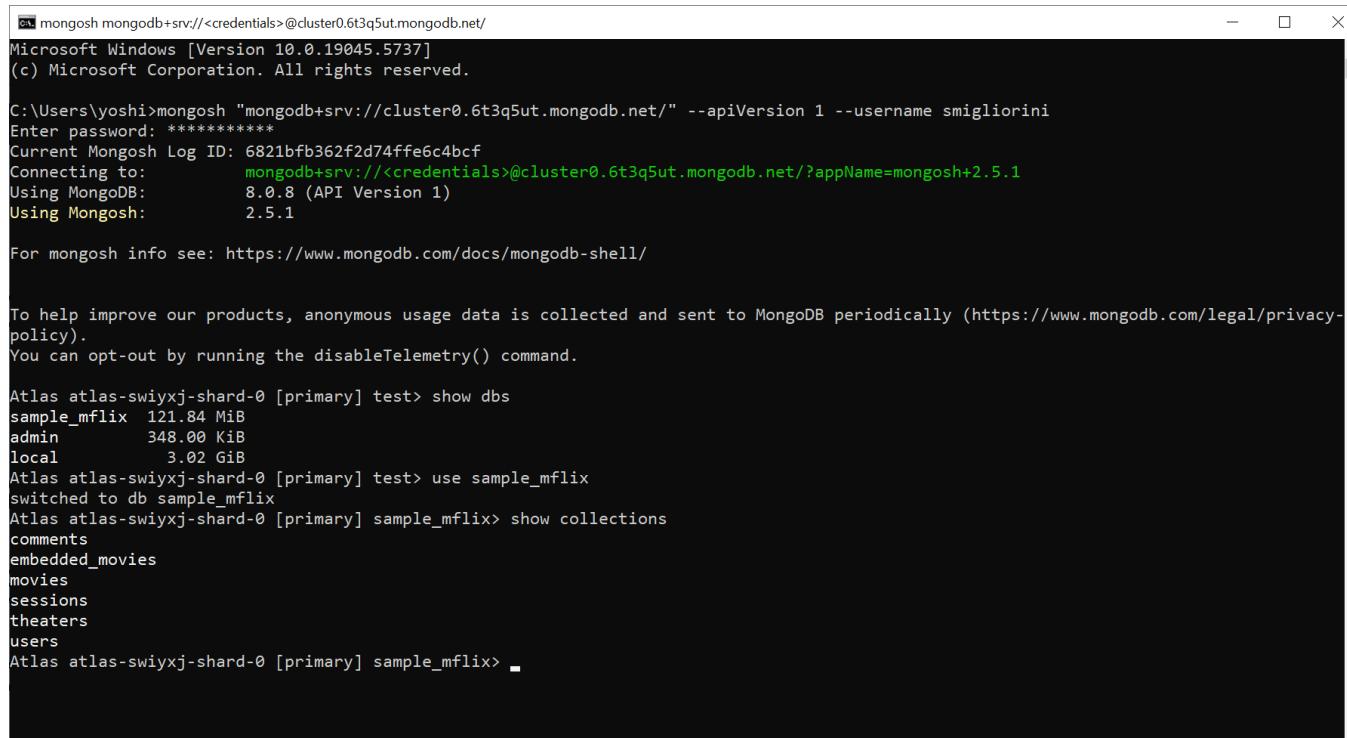
To help improve our products, anonymous usage data is collected and sent to MongoDB periodically (https://www.mongodb.com/legal/privacy-policy).
You can opt-out by running the disableTelemetry() command.

Atlas atlas-swiyxj-shard-0 [primary] test> show dbs
sample_mflix 121.84 MiB
admin      348.00 KiB
local      3.02 GiB
Atlas atlas-swiyxj-shard-0 [primary] test> use sample_mflix
switched to db sample_mflix
Atlas atlas-swiyxj-shard-0 [primary] sample_mflix> show collections
comments
embedded_movies
movies
sessions
theaters
users
Atlas atlas-swiyxj-shard-0 [primary] sample_mflix> ■
```

Comandi principali di mongosh

Mostrare le collezioni di documenti presenti in un database

- sample_mflix> show collections



```
mongosh mongodb+srv://<credentials>@cluster0.6t3q5ut.mongodb.net/
Microsoft Windows [Version 10.0.19045.5737]
(c) Microsoft Corporation. All rights reserved.

C:\Users\yoshi>mongosh "mongodb+srv://cluster0.6t3q5ut.mongodb.net/" --apiVersion 1 --username smigliorini
Enter password: ****
Current Mongosh Log ID: 6821bfb362f2d74ffe6c4bcf
Connecting to: mongodb+srv://<credentials>@cluster0.6t3q5ut.mongodb.net/?appName=mongosh+2.5.1
Using MongoDB: 8.0.8 (API Version 1)
Using Mongosh: 2.5.1

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

To help improve our products, anonymous usage data is collected and sent to MongoDB periodically (https://www.mongodb.com/legal/privacy-policy).
You can opt-out by running the disableTelemetry() command.

Atlas atlas-swiyxj-shard-0 [primary] test> show dbs
sample_mflix 121.84 MiB
admin 348.00 KiB
local 3.02 GiB
Atlas atlas-swiyxj-shard-0 [primary] test> use sample_mflix
switched to db sample_mflix
Atlas atlas-swiyxj-shard-0 [primary] sample_mflix> show collections
comments
embedded_movies
movies
sessions
theaters
users
Atlas atlas-swiyxj-shard-0 [primary] sample_mflix>
```

Comandi principali di mongosh

Visualizzare tutti i comandi disponibili

- test> help

Uscire dalla shell

- test> exit

```
mongosh mongodb+srv://<credentials>@cluster0.6tq5ut.mongodb.net/
embedded_movies
movies
sessions
theaters
users
Atlas atlas-swiyxj-shard-0 [primary] sample_mflix> help

Shell Help:

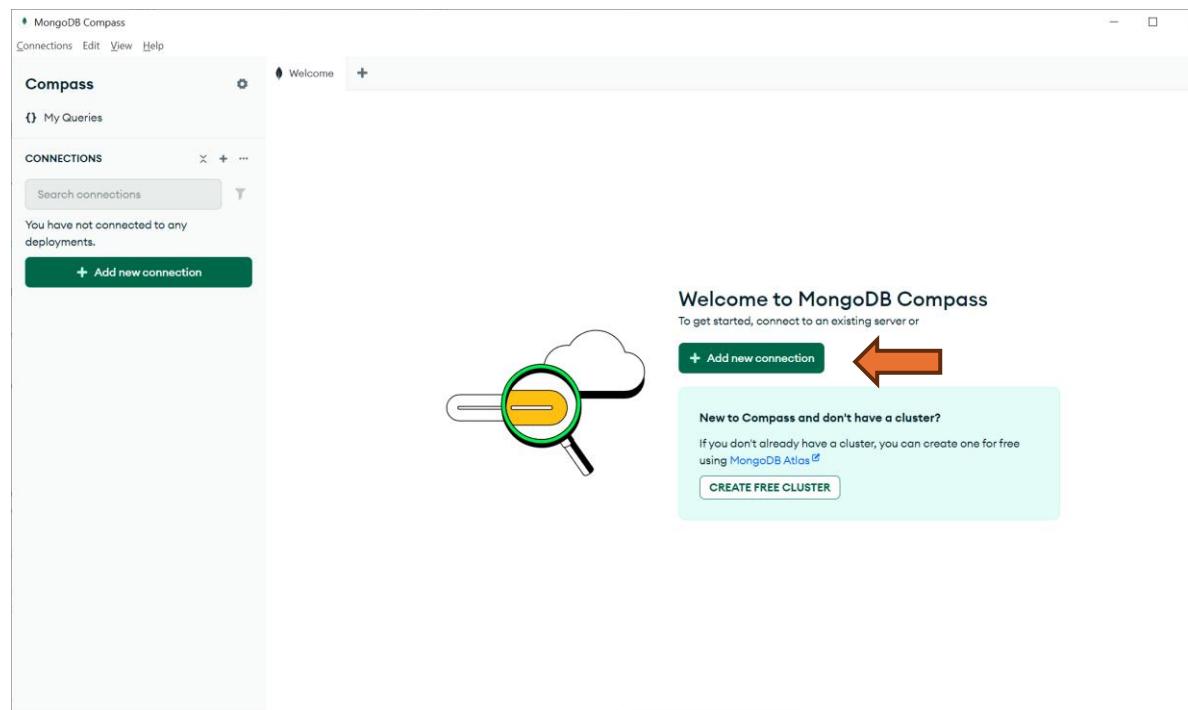
log                                     'log.info(<msg>)': Write a custom info/warn/error/debug message to the log file
'log.getPath()': Gets a path to the current log file

use                                     Set current database
show                                     'show databases'/'show dbs': Print a list of all available databases
                                         'show collections'/'show tables': Print a list of all collections for current database
                                         'show profile': Prints system.profile information
                                         'show users': Print a list of all users for current database
                                         'show roles': Print a list of all roles for current database
                                         'show log <type>': log for current connection, if type is not set uses 'global'
                                         'show logs': Print all logs
exit                                    Quit the MongoDB shell with exit/exit()/.exit
quit                                     Quit the MongoDB shell with quit/quit()
Mongo                                    Create a new connection and return the Mongo object. Usage: new Mongo(URI, options [optional])
connect                                 Create a new connection and return the Database object. Usage: connect(URI, username [optional], password [optional])
it                                       result of the last line evaluated; use to further iterate
version                                 Shell version
load                                     Loads and runs a JavaScript file into the current shell environment
enableTelemetry                         Enables collection of anonymous usage data to improve the mongosh CLI
disableTelemetry                        Disables collection of anonymous usage data to improve the mongosh CLI
passwordPrompt                          Prompts the user for a password
sleep                                    Sleep for the specified number of milliseconds
print                                    Prints the contents of an object to the output
printjson                               Alias for print()
convertShardKeyToHashed                 Returns the hashed value for the input using the same hashing function as a hashed index.    cls
                                         Clears the screen like console.clear()
isInteractive                           Returns whether the shell will enter or has entered interactive mode

For more information on usage: https://docs.mongodb.com/manual/reference/method
Atlas atlas-swiyxj-shard-0 [primary] sample_mflix>
```

MongoDB Compass: Connessione

- Interfaccia grafica alternativa all'uso della shell mongosh
- Rappresentazione visuale dei database e delle collezioni
- Visualizza le performance di query ed indici



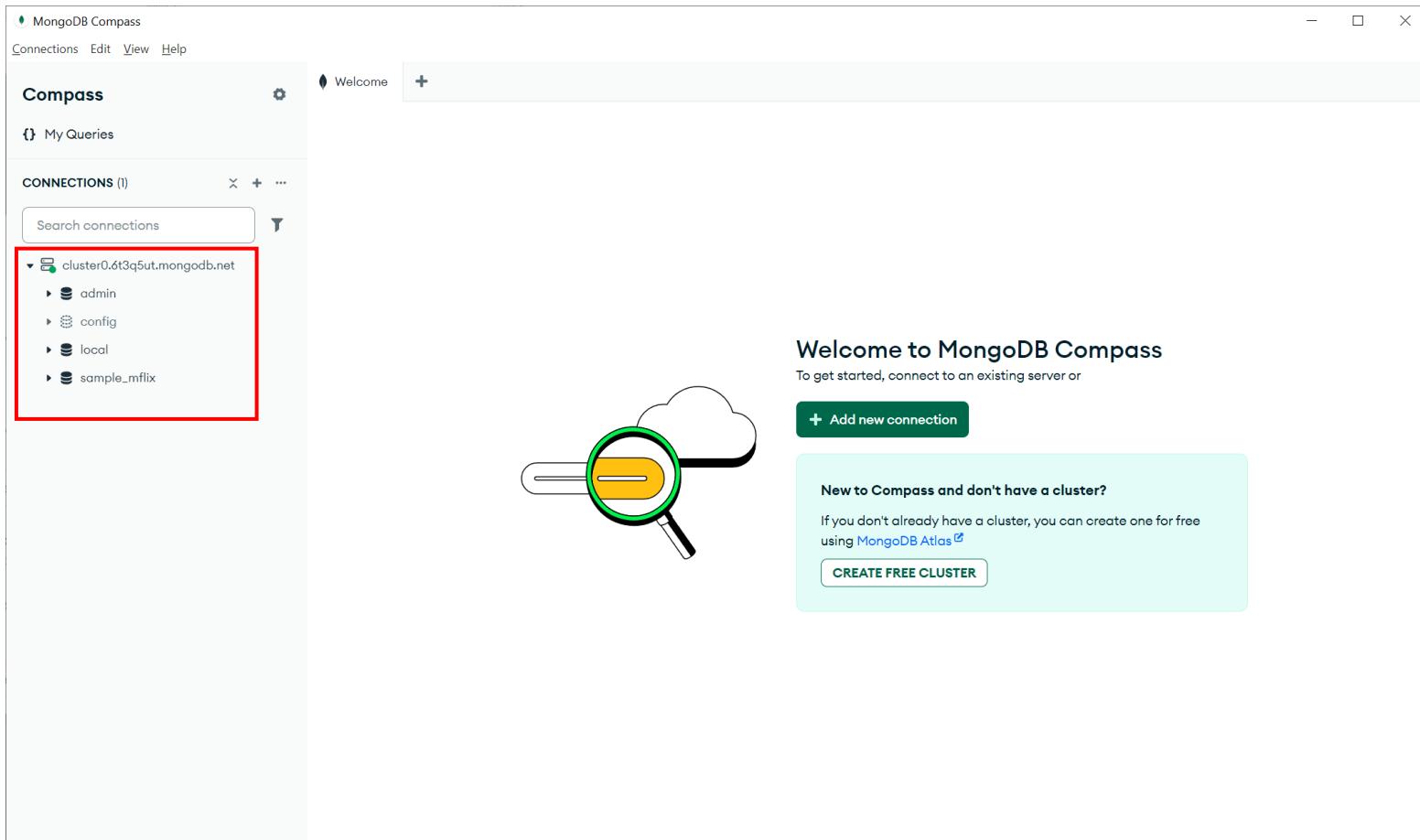
MongoDB Compass: Connessione

The image displays two side-by-side screenshots illustrating the connection setup between MongoDB Compass and an Atlas cluster.

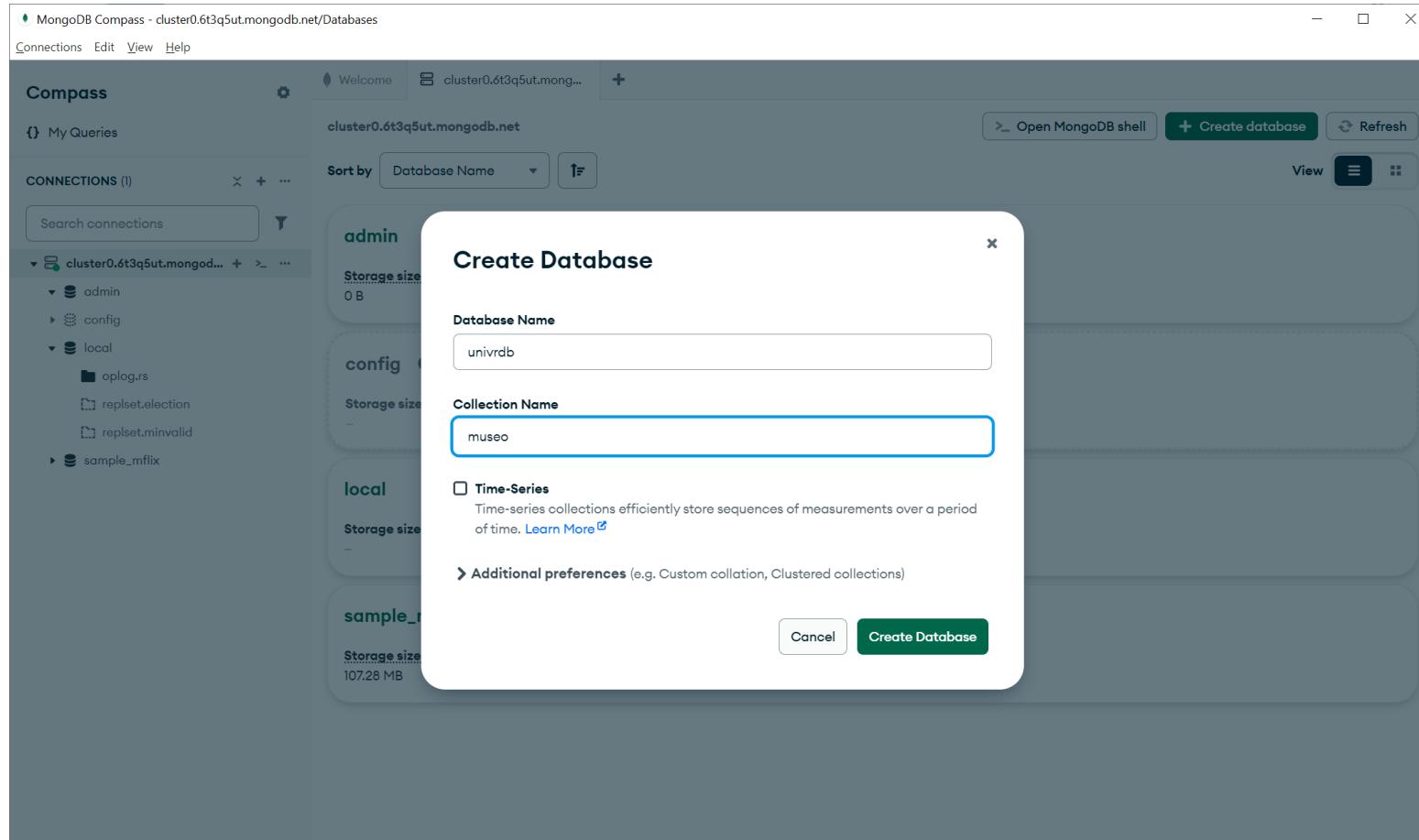
Left Screenshot (MongoDB Atlas): Shows the "Clusters" view for a project named "Project 0". A modal window titled "Connect to Cluster0" is open, showing the "Choose a connection method" step. It includes a section for "Connecting with MongoDB Compass" with the sub-instruction "I have MongoDB Compass installed". Below this, it says "1. Choose your version of Compass" (set to "1.38 or later") and "2. Copy the connection string, then open MongoDB Compass". A red box highlights the connection string: `mongodb+srv://smiglierini:<db_password>@cluster0.6t3q5ut.mongodb.net/`. An orange arrow points from this string to the corresponding field in the MongoDB Compass screenshot below.

Right Screenshot (MongoDB Compass): Shows the "New Connection" dialog. The "URI" field contains the same connection string: `mongodb+srv://smiglierini:<db_password>@cluster0.6t3q5ut.mongodb.net/`. Another orange arrow points from this field to the same string in the Atlas screenshot above. The dialog also includes fields for "Name" (set to "cluster0.6t3q5ut.mongodb.net") and "Color" (set to "No Color"). There are buttons for "Save", "Connect", and "Save & Connect". To the right of the dialog, there are two help cards: "How do I find my connection string in Atlas?" and "How do I format my connection string?", each with a "See example" link.

MongoDB Compass: Consultazione DB

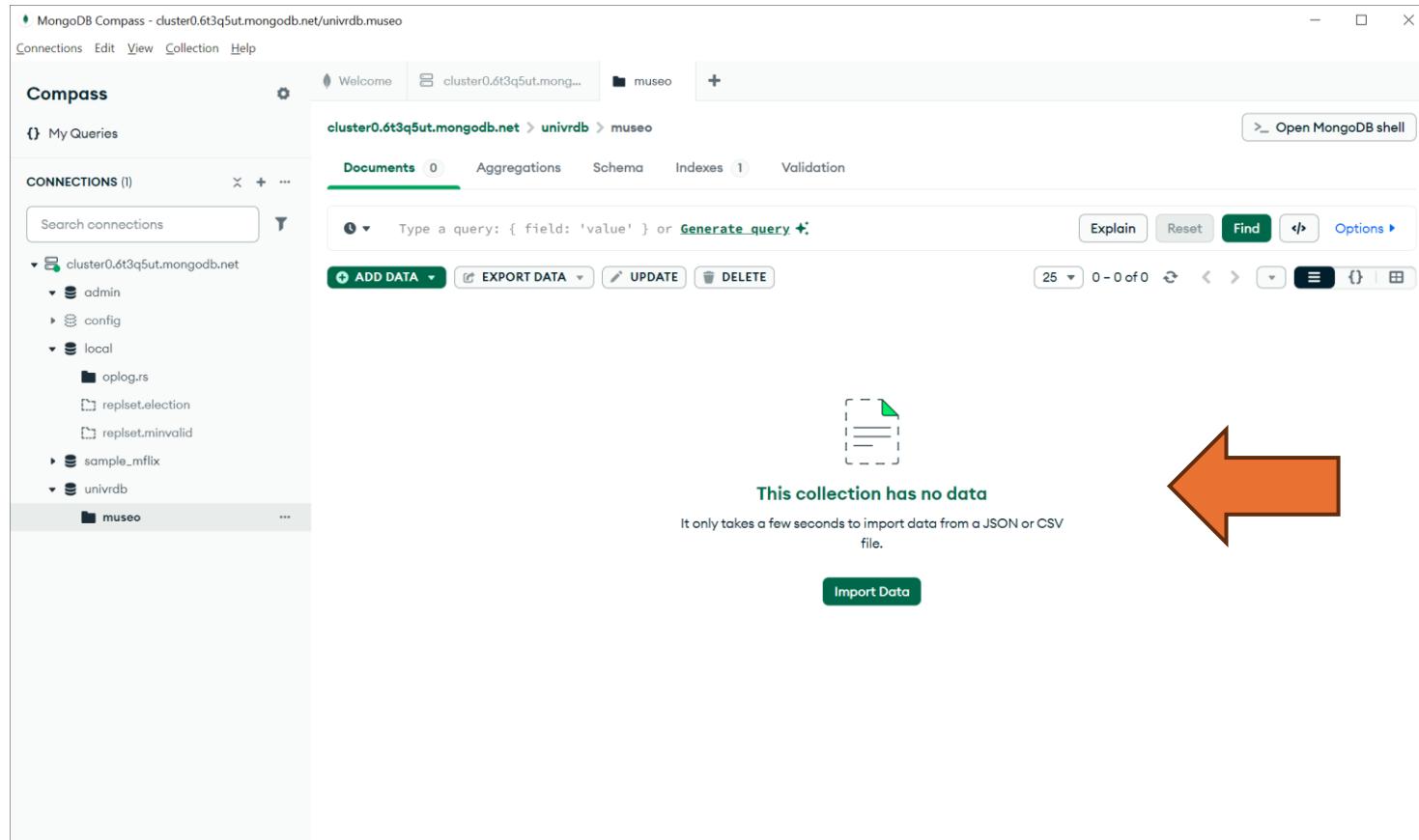


MongoDB Compass: Creazione Database



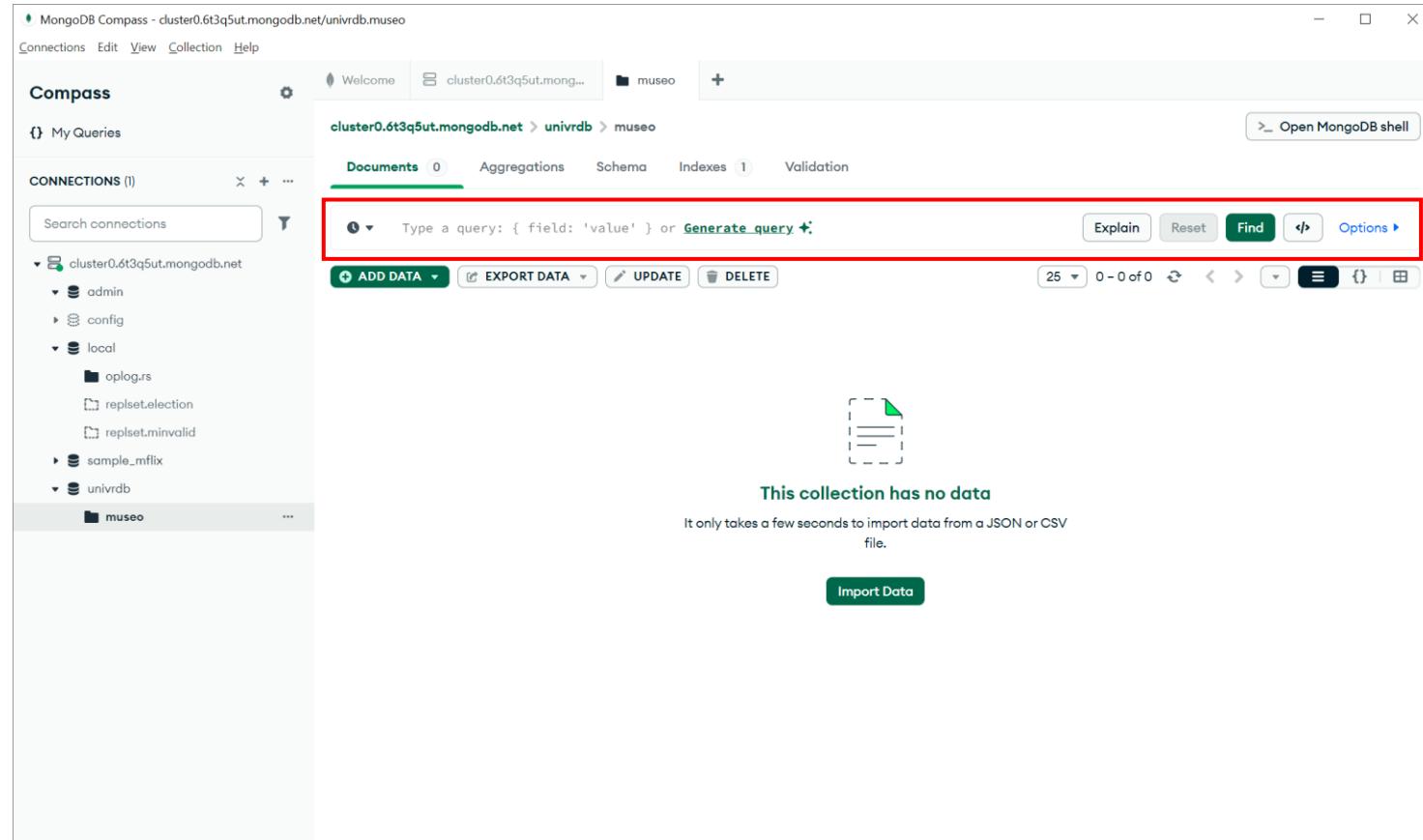
Un database contiene almeno una collezione di documenti.

MongoDB Compass: Creazione Database



Una volta creato un database ed una collezione è possibile importare i dati (documenti) che fanno parte della conessione tramite file CSV oppure JSON.

MongoDB Compass: Creazione Database



Selezionata una collezione è possibile vederne un campione oppure interrogarla tramite Query textbox.

MongoDB Compass: Creazione Database

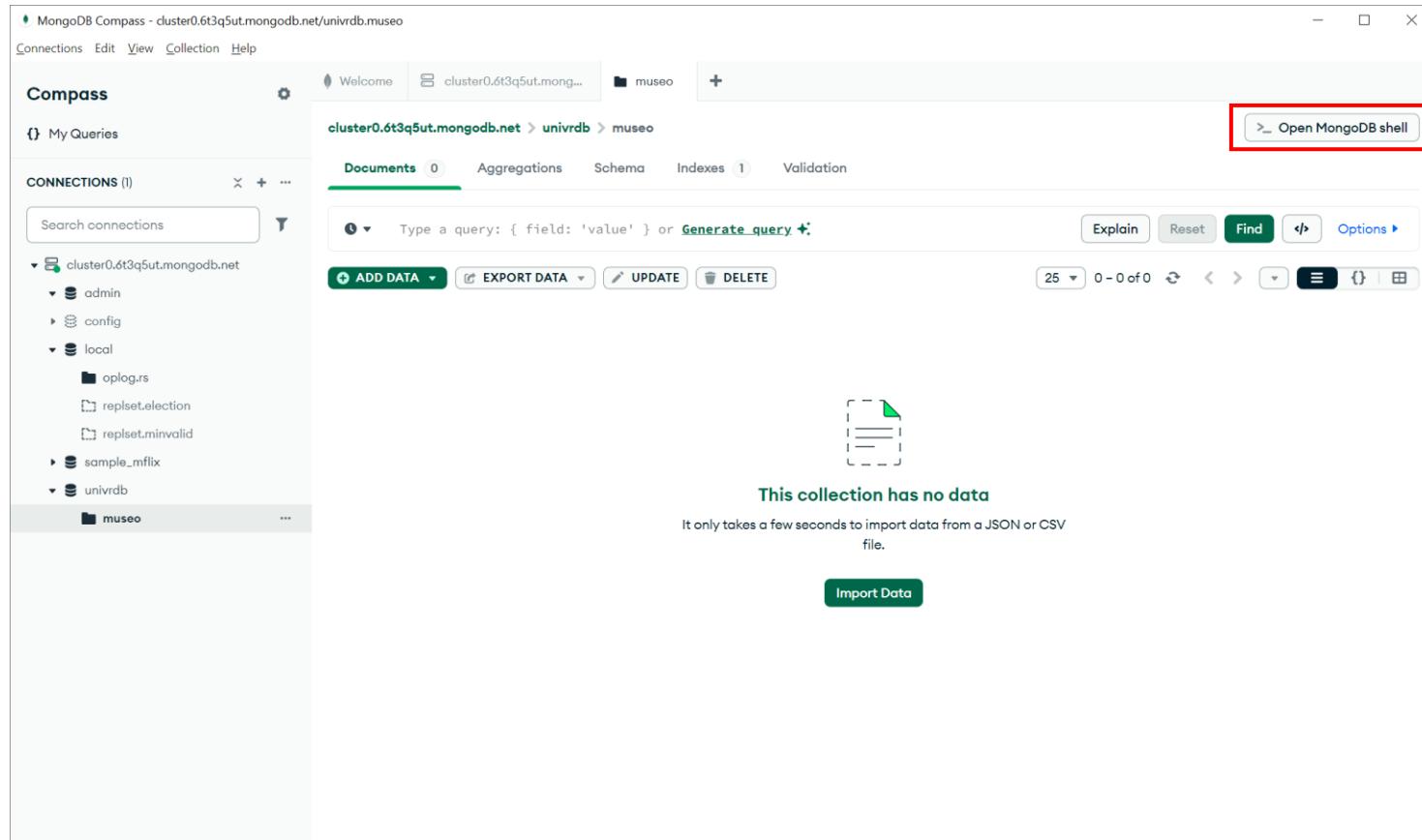
The screenshot shows the MongoDB Compass interface connected to a cluster at `cluster0.6t3q5ut.mongodb.net`. The current database is `sample_mflix`, and the collection is `movies`. There are 21.3K documents in the collection.

The interface includes a sidebar for connections, a search bar, and a main area for querying and managing documents. The document view shows two movie documents:

```
_id: ObjectId('573a1390f29313caabcd42e8')
plot: "A group of bandits stage a brazen train hold-up, only to find a determ..."
> genres: Array (2)
> runtime: 11
> cast: Array (4)
> poster: "https://m.media-amazon.com/images/M/MV5BMTU3NjE5NzYtYTYYNS00MDVmLWIwYj...
> title: "The Great Train Robbery"
> fullplot: "Among the earliest existing films in American cinema - notable as the ..."
> languages: Array (1)
> released: 1903-12-01T00:00:00.000+00:00
> directors: Array (1)
> rated: "TV-G"
> awards: Object
> lastupdated: "2015-08-13 00:27:59.177000000"
> year: 1903
> imdb: Object
> countries: Array (1)
> type: "movie"
> tomatoes: Object
num_mflix_comments: 0

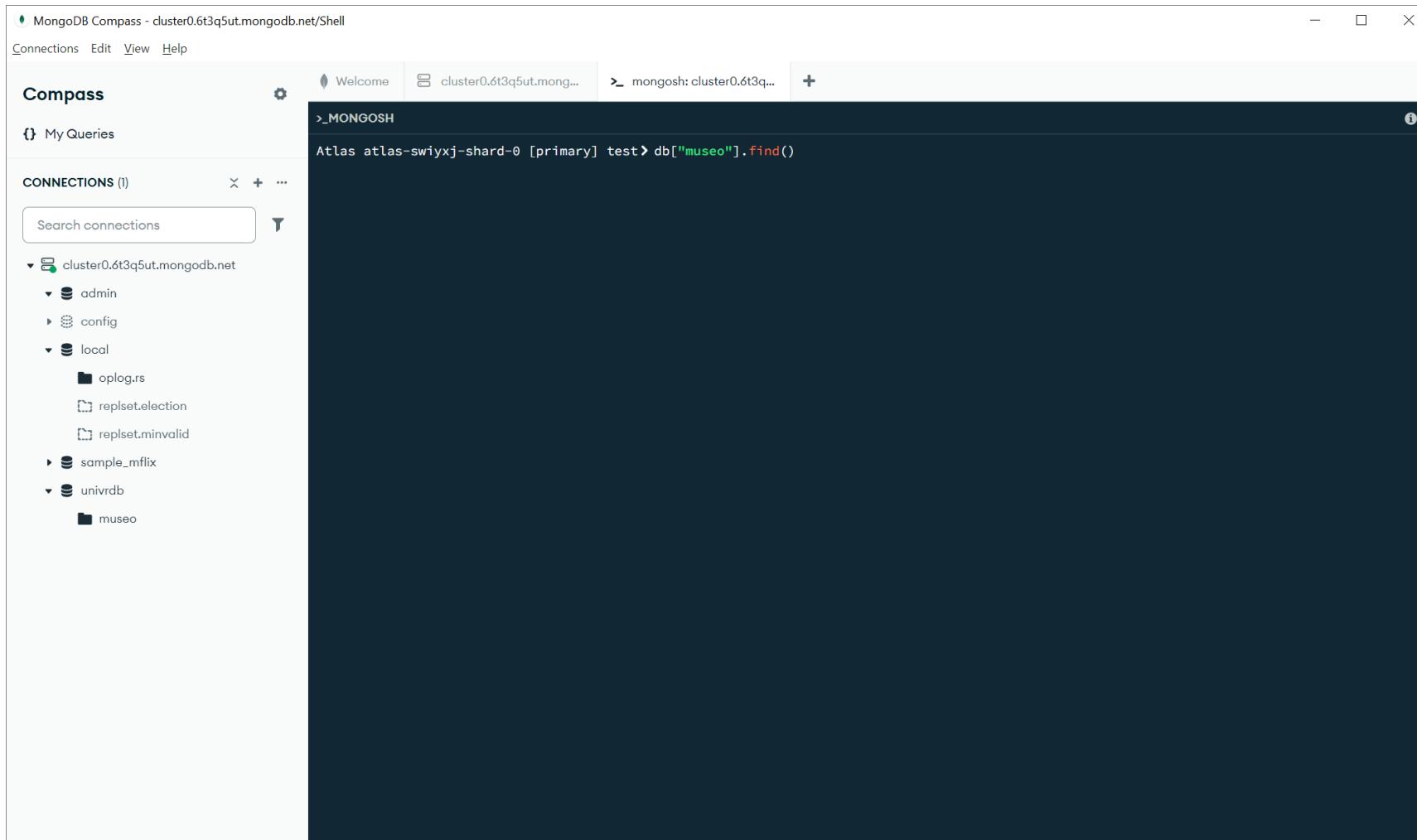
_id: ObjectId('573a1390f29313caabcd446f')
plot: "A greedy tycoon decides, on a whim, to corner the world market in whea...
> genres: Array (2)
> runtime: 14
> cast: Array (4)
> num_mflix_comments: 1
> title: "A Corner in Wheat"
> fullplot: "A greedy tycoon decides, on a whim, to corner the world market in whea..."
```

MongoDB Compass: Creazione Database

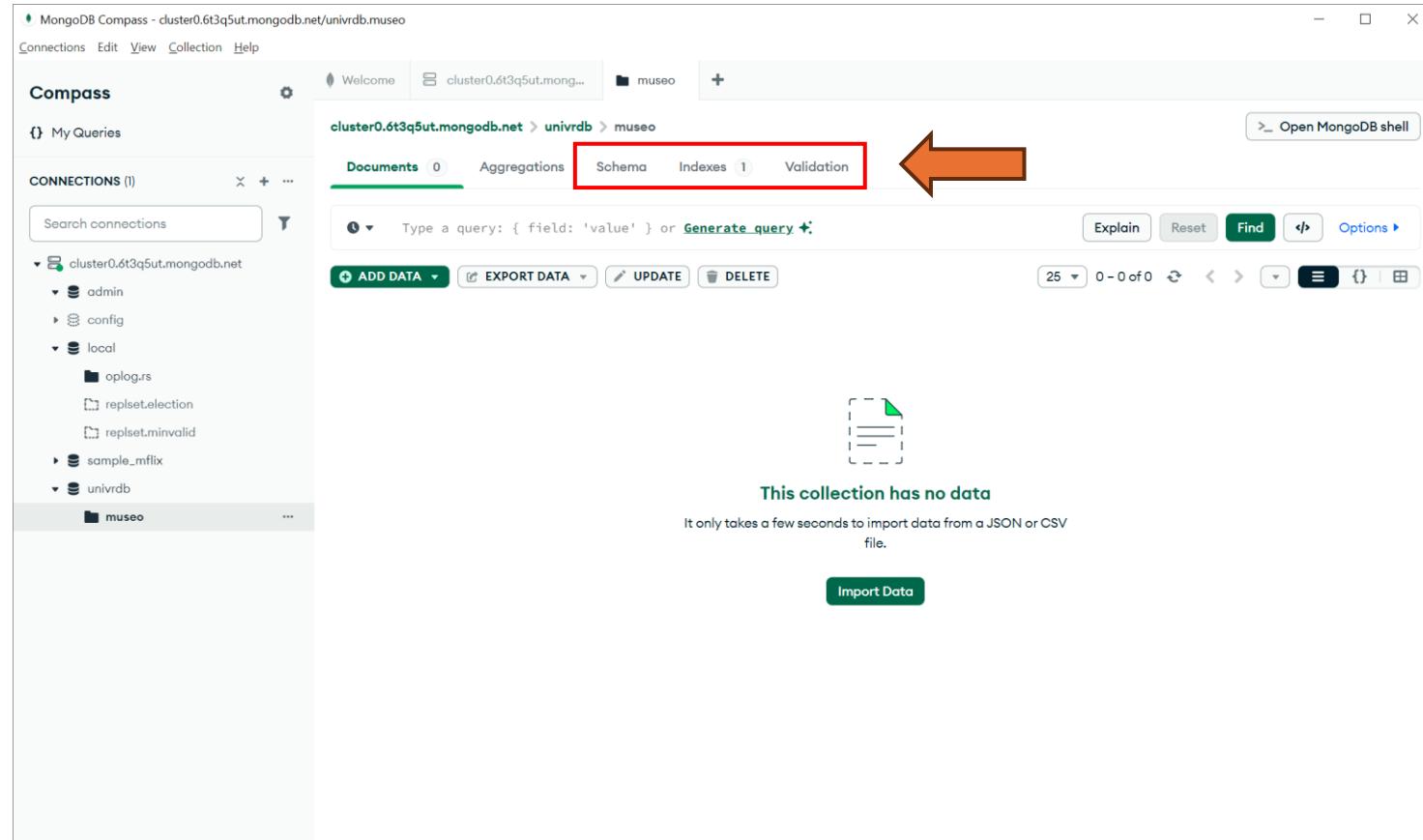


Usare la shell MongoDB integrata in Compass.

MongoDB Compass: Creazione Database



MongoDB Compass: Creazione Database



- Vedere lo schema dei documenti della collezione.
- Consultare gli indici della collezione.
- Validation rules: forzare delle regole di struttura dei documenti che dovranno essere soddisfatte durante gli aggiornamenti e gli inserimenti.

MongoDB Compass: Creazione Database

The image shows two side-by-side screenshots of the MongoDB Compass application interface.

Left Screenshot (Indexes Tab): This screenshot shows the "Indexes" tab for the "movies" collection in the "sample_mflix" database. It lists two indexes:

Name & Definition	Type	Size	Usage	Properties	Status
id	REGULAR	1.1 MB	1 (since Mon May 12 2025)	UNIQUE	READY
cast_text_fullplot_text_genres_text_title_text	TEXT	17.1 MB	0 (since Mon May 12 2025)	COMPOUND	READY

Right Screenshot (Validation Tab): This screenshot shows the "Validation" tab for the same collection. It includes a "Create validation rules" section with a "Generate rules" button and an "Add Rule" button. A note at the bottom states: "Generate rules via schema analysis from existing sample data or add them manually to enforce document structure during updates and inserts".

Operazioni CRUD

Operazioni tipiche su una base di dati MongoDB

- **Create**: inserire un nuovo documento in un database MongoDB
- **Read**: interrogare uno o più documenti
- **Update**: modificare un documento esistente
- **Delete**: rimuovere un documento esistente

Punto di partenza

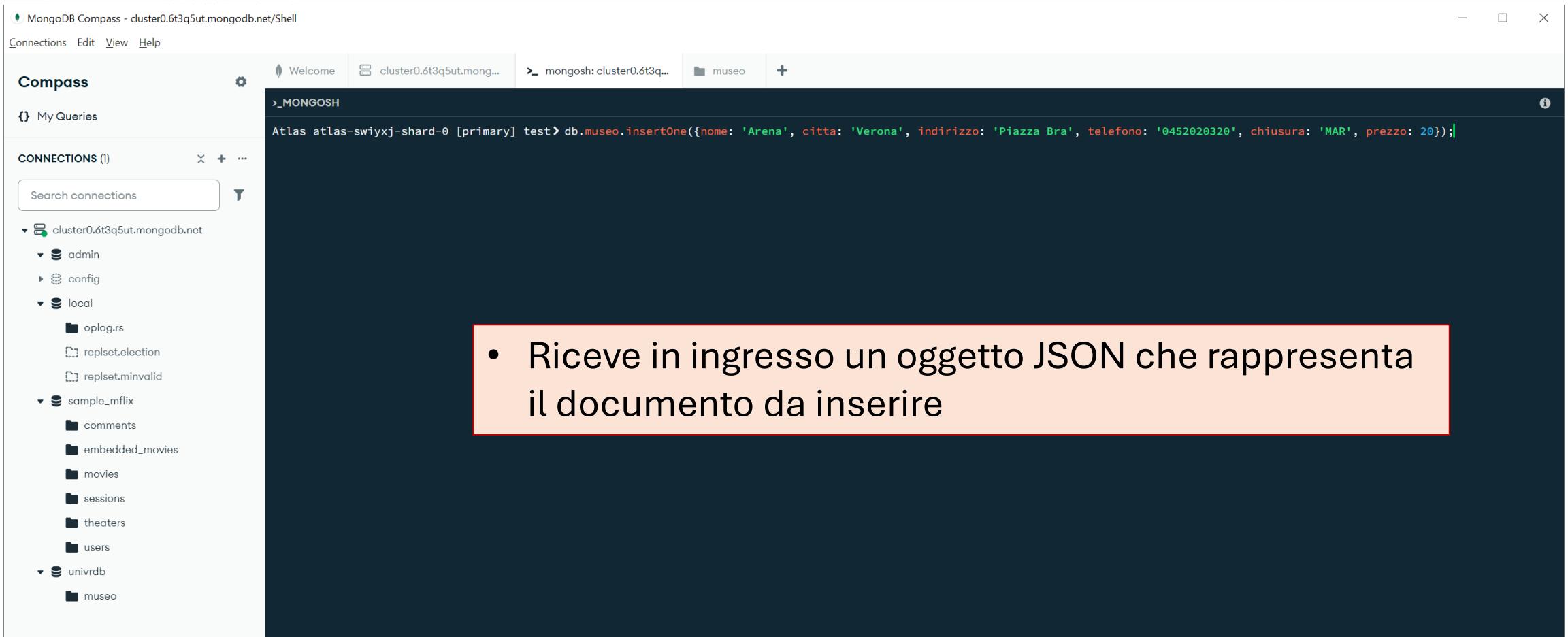
- show dbs
- use <dbname>

Create: Inserimento

Inserimento di un documento in una collezione (insertOne)

- db.museo.insertOne({ nome: 'Arena',
citta: 'Verona',
indirizzo: 'Piazza Bra',
telefono: '04500303',
chiusura: 'MAR',
prezzo: 20,
});
- **Nota:**
 - db.museo = db["museo"] = db.collection('museo')

Create: Inserimento



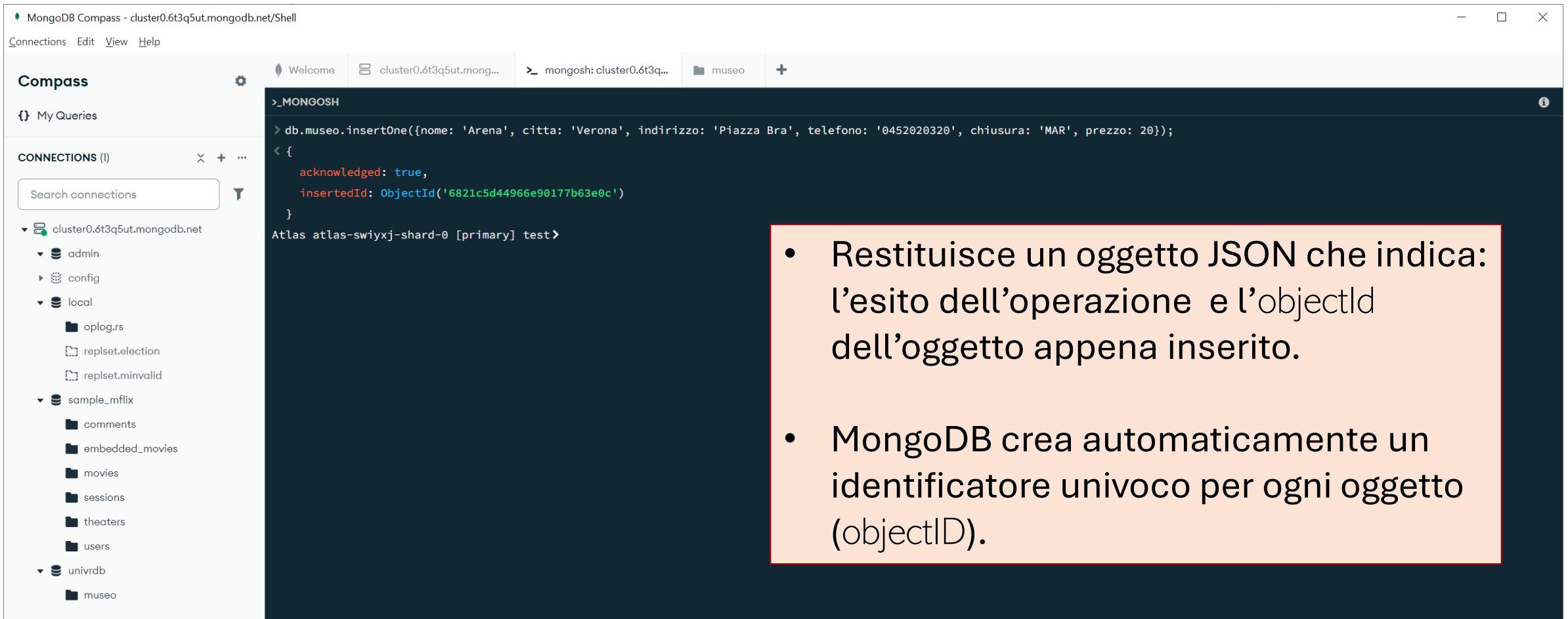
The screenshot shows the MongoDB Compass interface. On the left, the 'Connections' sidebar lists several databases: cluster0.6t3q5ut.mongodb.net, admin, config, local, sample_mflix, univrdb, and museo. The 'museo' database is currently selected. The main pane displays the MONGOSH shell with the following command:

```
>_MONGOSH
Atlas atlas-swiyxj-shard-0 [primary] test> db.museo.insertOne({nome: 'Arena', citta: 'Verona', indirizzo: 'Piazza Bra', telefono: '0452020320', chiusura: 'MAR', prezzo: 20});
```

A red box highlights the following bullet point:

- Riceve in ingresso un oggetto JSON che rappresenta il documento da inserire

Create: Inserimento



The screenshot shows the MongoDB Compass interface. On the left, the sidebar displays connections and databases. The main area shows the mongo shell interface with the following command and response:

```
_MONGOSH
> db.museo.insertOne({nome: 'Arena', citta: 'Verona', indirizzo: 'Piazza Bra', telefono: '0452020320', chiusura: 'MAR', prezzo: 20});
< {
  acknowledged: true,
  insertedId: ObjectId('6821c5d44966e90177b63e0c')
}
Atlas atlas-swiyxj-shard-0 [primary] test>
```

A red box highlights the response from the shell, which includes the inserted document and its unique ObjectId.

- Restituisce un oggetto JSON che indica: l'esito dell'operazione e l'objectId dell'oggetto appena inserito.
- MongoDB crea automaticamente un identificatore univoco per ogni oggetto (objectId).

Create: Inserimento

Inserimento di più documenti in una collezione (insertMany)

```
• db.museo.insertMany([
  { nome: 'Arena',
    citta: 'Verona',
    ...
  },
  { nome: 'Castelvecchio',
    citta: 'Verona',
    ...
  },
]);
```

Create: Inserimento

Riassunto dei Metodi disponibili:

- db.collection.insertOne()
- db.collection.insertMany()

Read: Ricerca

Ricerca di un documento in una collezione con un certo filtro

- db.museo.find({citta: 'Verona'});

- Riceve in ingresso un oggetto JSON contenente il filtro (opzionale).
- Restituisce gli oggetti JSON che soddisfano il filtro.

The screenshot shows the MongoDB Compass interface. On the left, the 'Connections' sidebar lists connections to 'cluster0.6t3q5ut.mongodb.net' (selected), 'admin', 'config', and 'local'. Below it are 'oplog.rs', 'repset.election', and 'repset.minvalid'. The main area shows the mongo shell interface with the command: > db.museo.find({citta: 'Verona'}). The response is a single document:

```
_id: ObjectId('6821c5d44966e90177b63e0c'),  
nome: 'Arena',  
citta: 'Verona',  
indirizzo: 'Piazza Bra',  
telefono: '0452020320',  
chiusura: 'MAR',  
prezzo: 20
```

At the bottom, it says 'Atlas atlas-swiyxj-shard-0 [primary] test>'.

Read: Ricerca

- db.**museo**.**find**({})
 - Recupera tutti i documenti della collezione museo
- db.**museo**.**find**({ <field1>: <value1>, ... })
 - Recupera tutti i documenti della collezione museo che soddisfano il filtro
 - Le condizioni sono in AND.
 - Operatori generici di confronto:
 - { <field1>: { <operator1>: <value1> }, ... }
 - \$in, \$nin
 - \$eq, \$ne
 - \$gt, \$gte, \$lt, \$lte
 - ecc...

Read: Ricerca

- db.museo.find({ \$or: [{ status: 'A' }, { qty: { \$lt: 30 } }] });
 - Condizioni in OR
- db.museo.find({ status: 'A',
 \$or: [{ qty: { \$lt: 30 } }, { item: { \$regex: '^p' } }] });
 - Condizioni in AND + OR

Ricerca con valori nulli

Ricerca con valori nulli

- db.museo.find({ item: null });
- db.museo.find({ item: { \$ne : null } });

Read: Ricerca con proiezione

Ricerca con proiezione

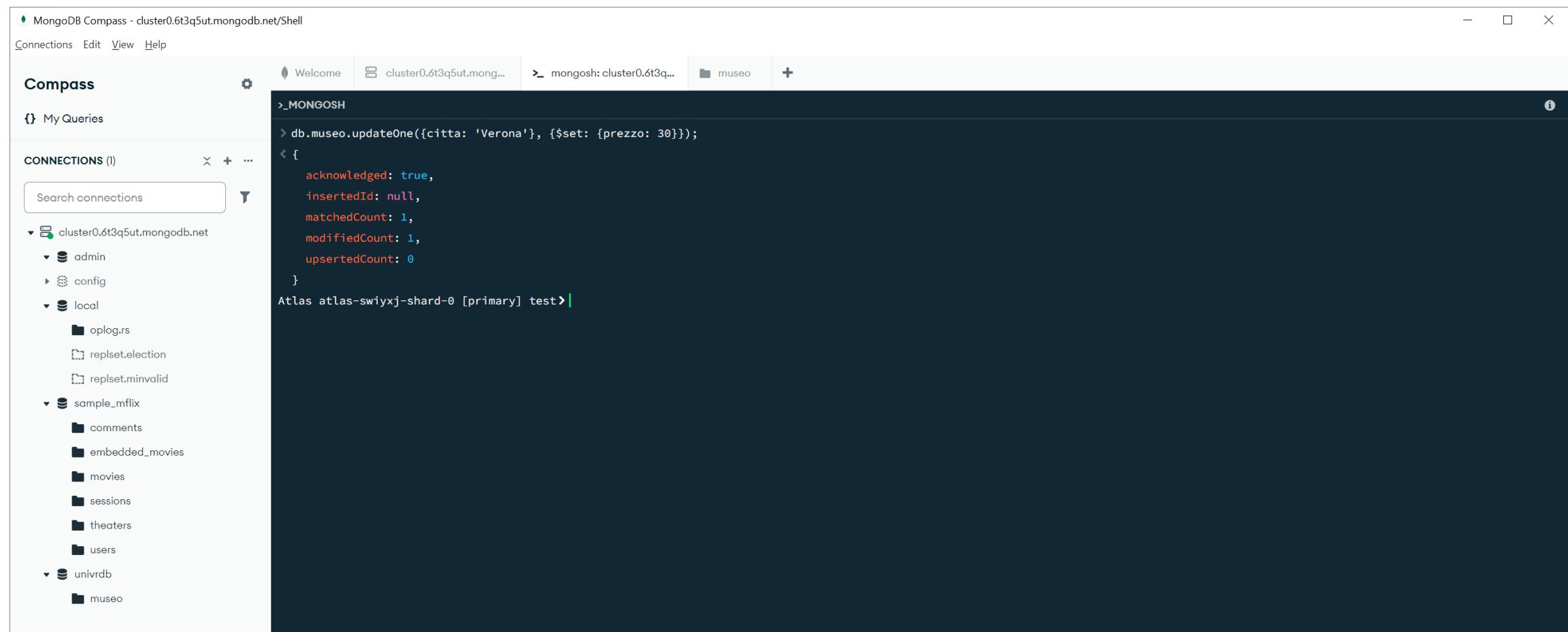
- db.museo.find({ status: 'A' })
.project({ nome: 1, citta: 1, indirizzo: 0, chiusura: 0, prezzo: 0 });
- db.museo.find({ status: 'A' })
.project({ indirizzo: 0, chiusura: 0, prezzo: 0 });
 - Esclude specifici attribute (valore = 0), default = 1

Aggiornamento: Update

Aggiornare un documento: updateOne()

- db.museo.updateOne({citta: 'Verona'}, {\$set: {price: 30}});
 - L'oggetto JSON passato come primo parametro serve da filtro
 - \$set: permette di specificare un insieme di campi con il valore aggiornato
 - Ritorna un oggetto JSON che contiene dei metadati:
 - matchedCount: notifica il numero di documenti che soddisfano il filtro
 - modifiedCount: notifica il numero di documenti che sono stati modificati
 - Con updateOne() viene modificato sempre un solo documento → il primo che soddisfa il criterio

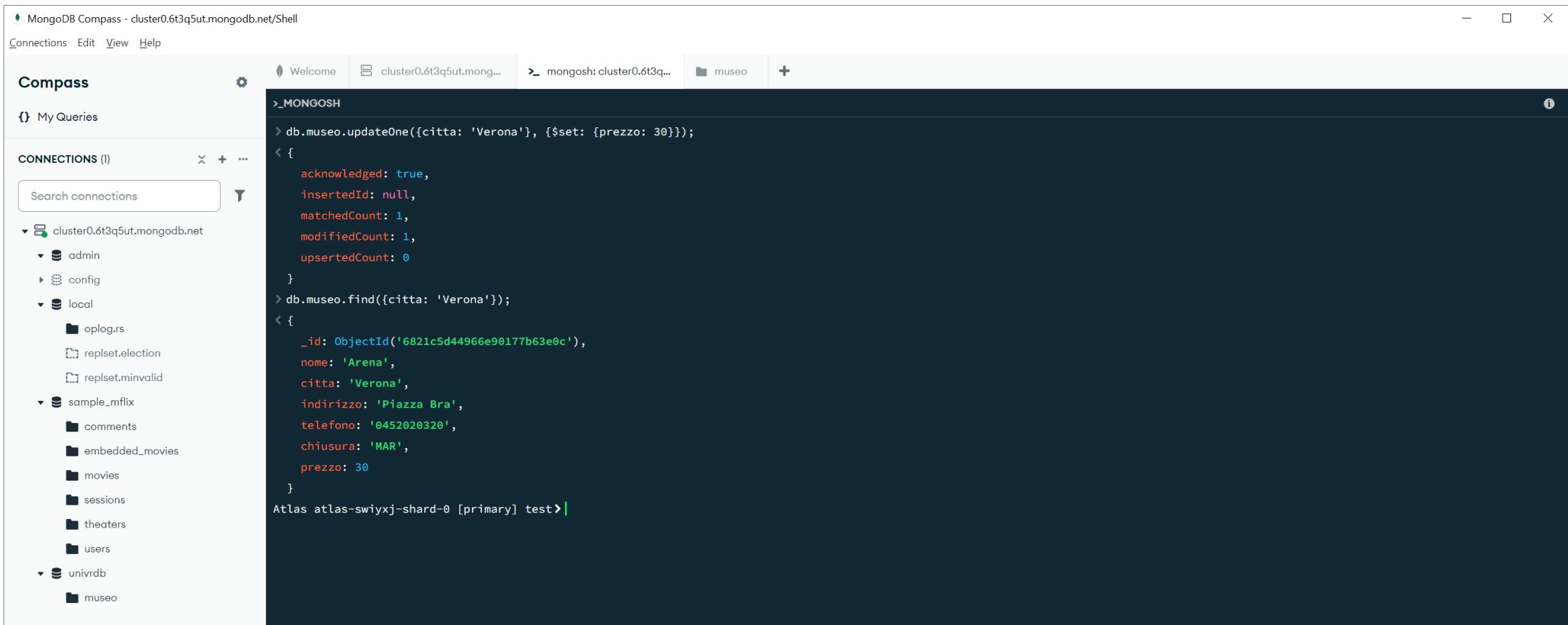
Aggiornamento: Update



The screenshot shows the MongoDB Compass interface. On the left, the 'Connections' sidebar lists several databases: cluster0.6t3q5ut.mongodb.net (selected), admin, config, local, oplog.rs, replset.election, replset.minivalid, sample_mflix (with comments, embedded_movies, movies, sessions, theaters, users), and univrdb (with museo). The main panel displays the MongoDB shell (mongosh) with the following command and its execution results:

```
>_MONGOSH
> db.museo.updateOne({città: 'Verona'}, {$set: {prezzo: 30}});
< {
  acknowledged: true,
  insertedId: null,
  matchedCount: 1,
  modifiedCount: 1,
  upsertedCount: 0
}
Atlas atlas-swiyxj-shard-0 [primary] test>
```

Aggiornamento: Update



The screenshot shows the MongoDB Compass interface. On the left, the connections sidebar lists several databases: cluster0.6t3q5ut.mongodb.net (selected), admin, config, local (with oplog.rs, replset.election, and replset.minvalid), sample_mflix (with comments, embedded_movies, movies, sessions, theaters, and users), and univrdb (with museo). The main panel displays the MONGOSH shell. A command is being run to update a document in the museo collection:

```
> db.museo.updateOne({citta: 'Verona'}, {$set: {prezzo: 30}});< {  acknowledged: true,  insertedId: null,  matchedCount: 1,  modifiedCount: 1,  upsertedCount: 0}> db.museo.find({citta: 'Verona'});< {  _id: ObjectId('6821c5d44966e90177b63e0c'),  nome: 'Arena',  citta: 'Verona',  indirizzo: 'Piazza Bra',  telefono: '0452020320',  chiusura: 'MAR',  prezzo: 30}>
```

The command executed was `db.museo.updateOne({citta: 'Verona'}, {$set: {prezzo: 30}});`. The response indicates that one document was updated, with its `prezzo` field set to 30. A subsequent `db.museo.find({citta: 'Verona'})` command returns the updated document, showing the new value of 30.

Aggiornamento: Update

Riassunto dei metodi disponibili

- db.collection.updateOne()
- db.collection.updateMany()
- db.museo.replaceOne(<filter>, <replacement>, options)
 - **Sostituisce un singolo documento mantenendo inalterato l'ObjectId**
 - db.museo.replaceOne(

```
{ item: 'paper' },  
{ item: 'paper',  
instock: [  
    { warehouse: 'A', qty: 60 },  
    { warehouse: 'B', qty: 40 }  
]
```

});

Cancellazione: Delete

Cancellazione di un documento da una collezione:

- db.`museo.deleteOne({citta: 'Verona'})`;
- Parametro: oggetto JSON che permette di selezionare particolari documenti.
- `deleteOne()` cancella sempre il primo documento che rispetta la condizione di filtro.
- Ritorna un oggetto JSON con dei metadati, tra cui:
 - `deletedCount`: numero di documenti che sono stati cancellati
- `deleteMany()`

Schema in Documenti MongoDB

- Il modello relazionale predilige la proprietà di normalizzazione del dato: una stessa informazione è presente una sola volta all'interno della base di dati e le relazioni permettono di creare collegamenti da informazioni (evitare ridondanza!).
- In un database NoSQL è prevista la possibilità di avere ridondanza e strutture de-normalizzate per incrementare le prestazioni e garantire flessibilità (dato non strutturato o semi-strutturato).
- I documenti MongoDB consentono il concetto di **incapsulamento**: la proprietà di un documento può essere un altro documento.
 - L'incapsulamento riduce la necessità di JOIN e facilita il recupero dell'informazione.
 - La migliore struttura ed il miglior criterio di encapsulamento dipende dai requisiti dell'applicazione (tipico modo di interrogare l'informazione).

Incapsulamento

```
{ "_id": ObjectId("f34303859302"),  
  "title": "Journey to the Center of the Earth",  
  "published_year": 1864,  
  "genres": ["Adventures", "Science Fiction"],  
  "author": {  
    "first_name": "Jules",  
    "last_name": "Verne",  
    "birth_year": 1828,  
    "nationality": "French",  
    "biography": "A pioneering author in the science fiction..."  
  },  
  "rating": 4.5,  
  "copies_sold": 1500000  
}
```



Incapsulamento → ridondanza
Cosa succede se ho un altro libro dello stesso autore?

Riferimento

```
// Course Collection
{
  "_id": ObjectId("d04940004ncdd"),
  "course_name": "Introduction to Artificial Intelligence",
  "professor_id": ObjectId("4590f394ks"),
  "duration": "6 months",
  "credits": 6
}
// Professor Collection
{
  "_id": ObjectId("4590f394ks"),
  "first_name": "Alan",
  "last_name": "Turing",
  "department": "Computer science",
  "publications": 45
}
```

- Attraverso i riferimenti è possibile creare dei collegamenti tra i dati tramite «link» o «puntatori».
- Le applicazioni risolveranno questi riferimenti andando a recuperare i dati relativi.

Denormalizzazione

```
// Course Collection
{
  "_id": ObjectId("d04940004ncdd"),
  "course_name": "Introduction to Artificial Intelligence",
  "professor_id": ObjectId("4590f394ks"),
  "professor_name": "Alan Turin",
  "duration": "6 months",
  "credits": 6
}
// Professor Collection
{
  "_id": ObjectId("4590f394ks"),
  "first_name": "Alan",
  "last_name": "Turing",
  "department": "Computer science",
  "publications": 45
}
```

- Approccio a metà tra encapsulamento e riferimento che consente di velocizzare l'accesso a certe informazioni.

MongoDB Formato e tipi di dato

- MongoDB memorizza le informazioni nel formato BSON (Binary JSON) che è del tutto simile a JSON ma offre una più ampia varietà di tipi base e delle ottimizzazioni per la memorizzazione e la ricerca.
- Tipi di dato supportati da BSON:
 - “_id”: ObjectId(“407f1f77bcf86cd799439012”),
 - “doubleField”: 55.4,
 - “stringField”: “Hello, world!”,
 - “objectField”: {"name": “John”, “age”: 34},
 - “arrayField”: [1,2,3,4,5],
 - “binaryField”: BinData(0, “binary-content”),
 - “booleanField”: true,
 - “dataField”: ISODate(“2024-05-12T10:00:00Z”),

MongoDB Formato e tipi di dato

- “nullField”: null,
- “regexField”: /[^]abc/,
- “javascriptField”: Code(“function() { print(‘success!’); }”),
- “javascriptWithScopeField”: Code(“return this.x + this.y; {x: 3, y: 5}”),
- “intField”: 42,
- “timestampField”: Timestamp(1601410400, 1),
- “int64Field”: NumberLong(9876543210),
- “decimal128Field”: NumberDecimal(“0.123456789”),
- “minKeyField”: MinKey(),
- “maxKeyField”: MaxKey(),

MongoDB: Aggregazioni

- Le aggregazioni sono uno strumento che consente di combinare più trasformazioni e interrogazioni, definendo una pipeline di processamento.

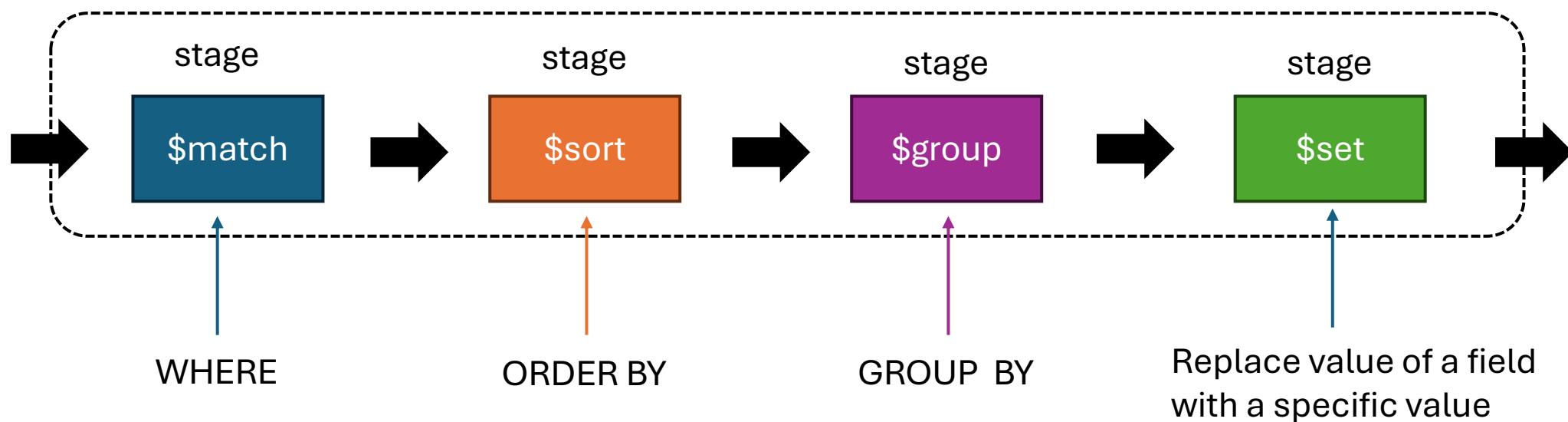
```
{  
  "_id": ObjectId("059440..."),  
  "student_id": ObjectId("4d034930..."),  
  "course_id": ObjectId("2f402002..."),  
  "grade": 76,  
  "semester": "Fall 2023",  
}
```

```
db.student_grades.aggregate([  
  {$match: { semester: "Fall 2023" }},  
  {$group: { _id: "student_id" }},  
  {$project:{ $divide: [ $totalGrade", $totalCourses] }  
})
```

Calcola la media per studente

MongoDB: Aggregazioni

- In una pipeline di trasformazione, ad ogni passo viene passato all'operazione corrente un insieme di documenti, mentre l'insieme ottenuto dalla trasformazione è passato all'operazione successiva.



MongoDB: Aggregazioni (operatori)

SQL clause, term, function	MongoDB aggregation operators
WHERE	\$match
GROUP BY	\$group
HAVING	\$match
SELECT	\$project
LIMIT	\$limit
OFFSET	\$skip
ORDER BY	\$sort
SUM()	\$sum
COUNT()	\$sum AND \$sortByCount
JOIN	\$lookup
SELECT INTO NEW_TABLE	\$out
MERGE INTO TABLE	\$merge
UNION ALL	\$unionWith

MongoDB: Aggregazioni (operatori)

SQL clause, term, function	MongoDB aggregation operators
WHERE	\$match
GROUP BY	\$group
HAVING	\$match
SELECT	\$project
LIMIT	\$limit
OFFSET	\$skip
ORDER BY	Esistono altri operatori: consultare la documentazione!
SUM()	\$sum
COUNT()	\$sum AND \$sortByCount
JOIN	\$lookup
SELECT INTO NEW_TABLE	\$out
MERGE INTO TABLE	\$merge
UNION ALL	\$unionWith

MongoDB: Join

- L'operatore di aggregazione `$lookup` permette di eseguire un **left outer join** rispetto ad una collezione della stessa base di dati.

```
{  
  $lookup:  
    {  
      from: <collection to join>, —————> Collezione esterna presente nello stesso DB  
      localField: <field from the input documents>, —————> Attributo del documento corrente da usare per il (equi) JOIN  
      foreignField: <field from the documents of the "from" collection>, —————> Attributo del documento esterno da usare per il JOIN  
      let: { <var_1>: <expression>, ..., <var_n>: <expression> },  
      pipeline: [ <pipeline to run> ],  
      as: <output array field> —————> Nome dell'attributo che conterrà l'elenco dei documenti collegati  
    }  
}
```

MongoDB: Join

```
db.customers.aggregate([
  {
    $lookup: {
      // The collection to join.
      from: "orders",
      // The field from the customers collection.
      localField: "_id",
      // The field from the orders collection.
      foreignField: "customerId",
      // The output array containing joined documents.
      as: "orders"
    }
  }
]);
```

```
[
  {
    "_id": "cust100",
    "email": "jane@example.com",
    "name": "Jane Doe",
    "orders": [
      {
        "_id": "ord200",
        "customerId": "cust100",
        "itemIds": [
          "item300",
          "item301"
        ]
      },
      {
        "_id": "ord202",
        "customerId": "cust100",
        "itemIds": [
          "item301",
          "item303"
        ]
      }
    ],
    ...
  ]
]
```