

Adam Biśta

1. Instalujemy mongoDB community Serwer, następnie mongo shell

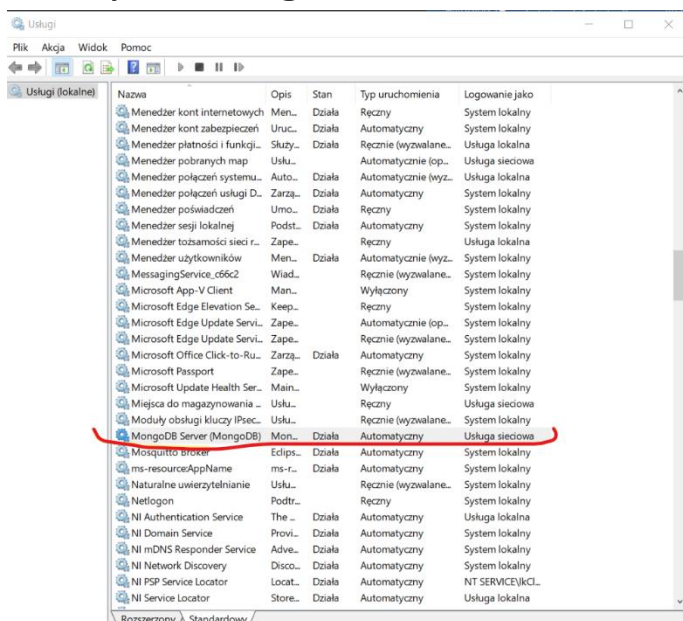
W terminalu wpisujemy:

```
Microsoft Windows [Version 10.0.19045.2846]
(c) Microsoft Corporation. Wszelkie prawa zastrzeżone.

C:\Users\Lenovo>mongod --version
db version v6.0.5
Build Info: {
  "version": "6.0.5",
  "gitVersion": "c9a99c120371d4d4c52cbb15dac34a36ce8d3b1d",
  "modules": [],
  "allocator": "tcmalloc",
  "environment": {
    "distmod": "windows",
    "distarch": "x86_64",
    "target_arch": "x86_64"
  }
}

C:\Users\Lenovo>
```

Następnie wpisujemy mongod aby włączyć serwer.
W wyniku tego, nasz serwer zostaje uruchomiony:



Następnie wpisujemy mongosh:

```
(c) Microsoft Corporation. Wszelkie prawa zastrzeżone.

C:\Users\Lenovo>mongosh
Current Mongosh Log ID: 645caa9b88d21cb4824155cf
Connecting to:      mongodb://127.0.0.1:27017/?directConnection=true&serverSelectionTimeoutMS=2000&appName=mongosh.8.2
Using MongoDB:      6.0.5
Using Mongosh:       1.8.2

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

-----
  The server generated these startup warnings when booting
  2023-05-11T09:54:18.444+02:00: Access control is not enabled for the database. Read and write access to data and configuration is unrestricted
-----

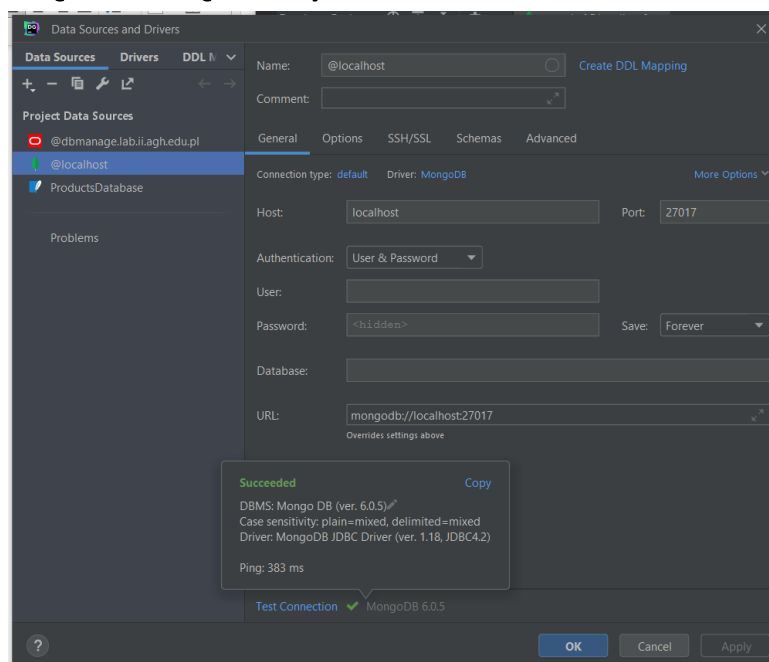
-----
  Enable MongoDB's free cloud-based monitoring service, which will then receive and display
  metrics about your deployment (disk utilization, CPU, operation statistics, etc).

  The monitoring data will be available on a MongoDB website with a unique URL accessible to you
  and anyone you share the URL with. MongoDB may use this information to make product
  improvements and to suggest MongoDB products and deployment options to you.

  To enable free monitoring, run the following command: db.enableFreeMonitoring()
  To permanently disable this reminder, run the following command: db.disableFreeMonitoring()
-----

test>
```

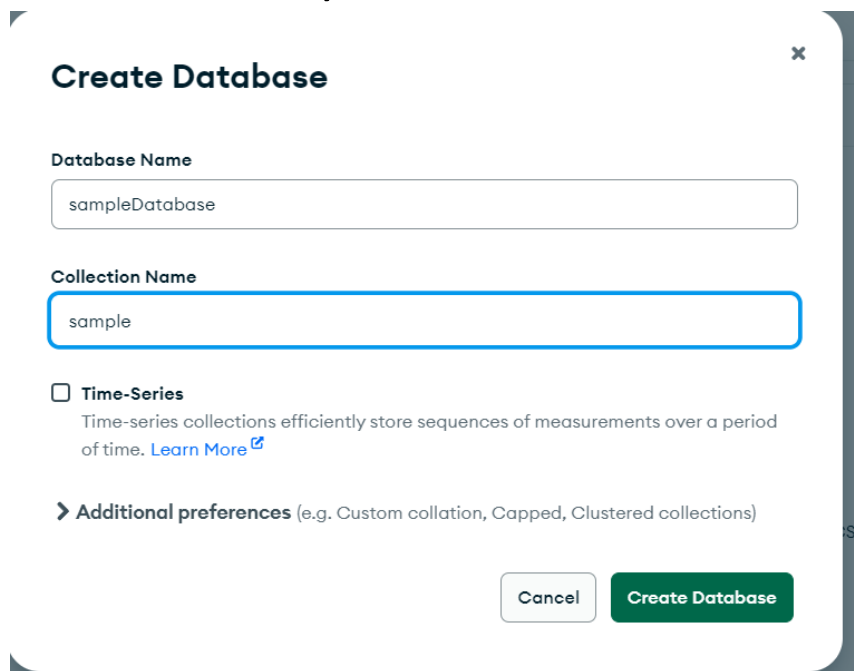
Użyłem dodatkowo narzędzia datagrip by połączyć się z bazą danych:



2. Tworzymy przykładową bazę danych

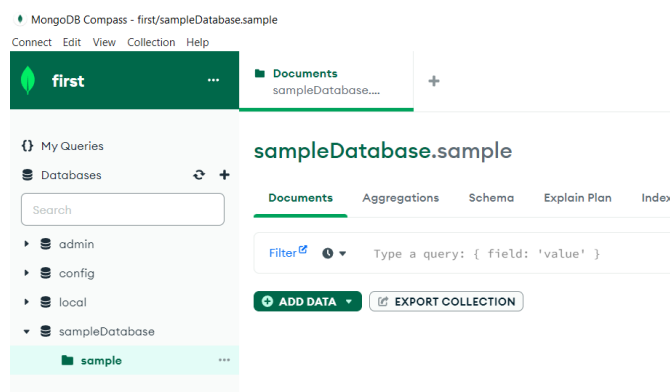
W zrobieniu tego zadania wykorzystałem MongoDB Compass. Automatycznie utworzył połączenie lokalne.

Tworzenie bazy:

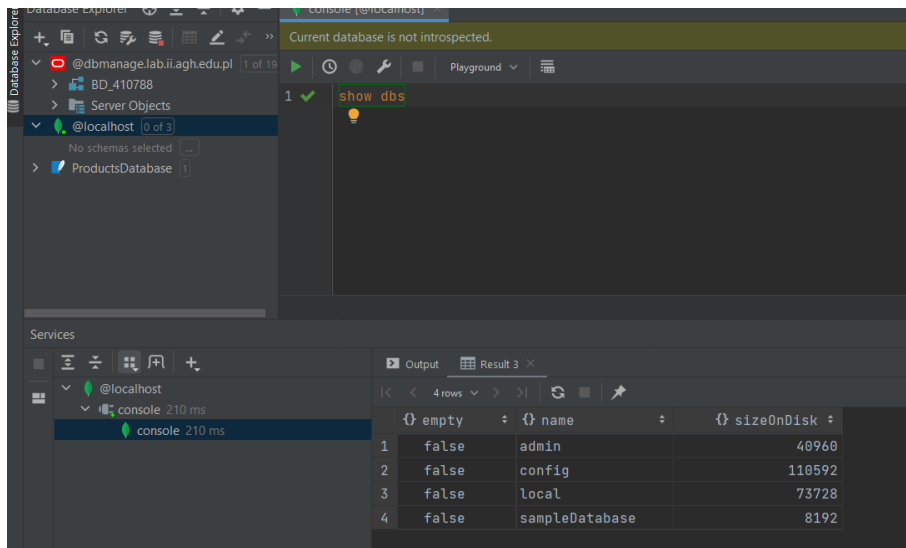


The screenshot shows the 'Create Database' dialog box in MongoDB Compass. It has a title bar with a close button (x). The dialog contains two input fields: 'Database Name' with the value 'sampleDatabase' and 'Collection Name' with the value 'sample'. Below these fields is a checkbox labeled 'Time-Series' which is unchecked. A description for 'Time-Series' states: 'Time-series collections efficiently store sequences of measurements over a period of time. [Learn More](#)'. Below this is a section for 'Additional preferences' with a right-pointing arrow and text '(e.g. Custom collation, Capped, Clustered collections)'. At the bottom right are two buttons: 'Cancel' and 'Create Database'.

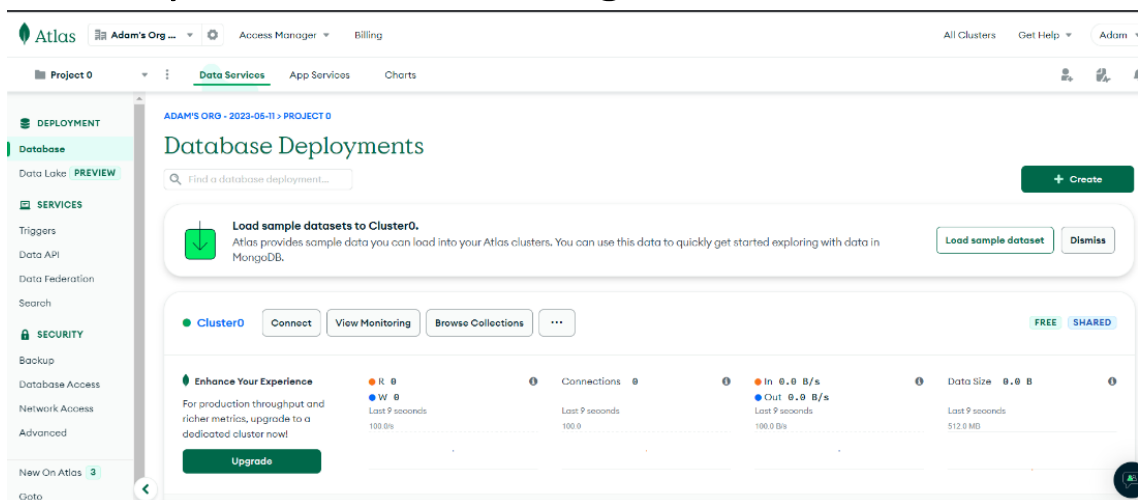
Wynik:



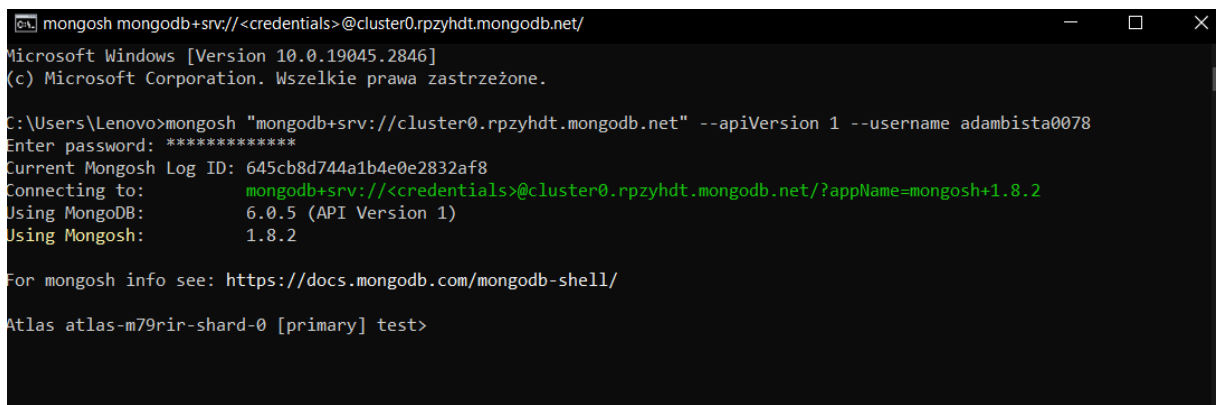
Testowanie w datagripie:



3. Import przykładowych zbiorów danych Utworzyłem konto dla MongoDB Atlas:



Łączenie się z mongosh:



Dodajemy przykładowe bazy danych:

The screenshot shows the 'Load Sample Datasets to Cluster0' page in MongoDB Atlas. At the top, there's a green header with the Atlas logo and the title 'Load Sample Datasets to Cluster0'. Below the title, a subtitle reads 'Choose sample data to explore features, data modeling patterns, and tutorials.' A yellow warning box states: 'If you previously loaded sample data into this deployment, loading all datasets will result in a namespace conflict.' Below this, a blue toggle switch is set to 'Select All' with a total size of '-350 MB'. A green success message indicates 'Sample dataset successfully loaded. Access it in Collections or by connecting with the MongoDB Shell.' Below the message are buttons for 'Cluster0', 'Connect', 'View Monitoring', 'Browse Collections', and a menu icon. At the bottom, there's a status bar with 'Enhance Your Experience' text, read/write statistics (R 0, W 148.9), connection count (5.0), and in/out data rates (In 633.8, Out 468).

Sprawdzenie w konsoli czy udało się załadować bazy:

```
Atlas atlas-m79rir-shard-0 [primary] test> show dbs
admin 336.00 KiB
local 25.56 GiB
Atlas atlas-m79rir-shard-0 [primary] test> show dbs
sample_airbnb      52.31 MiB
sample_analytics   9.14 MiB
sample_geospatial 1.34 MiB
sample_guides      40.00 KiB
sample_mflix       46.54 MiB
sample_restaurants 6.55 MiB
sample_supplies    1.11 MiB
sample_training    51.59 MiB
sample_weatherdata 2.74 MiB
admin              336.00 KiB
local              25.56 GiB
Atlas atlas-m79rir-shard-0 [primary] test>
```

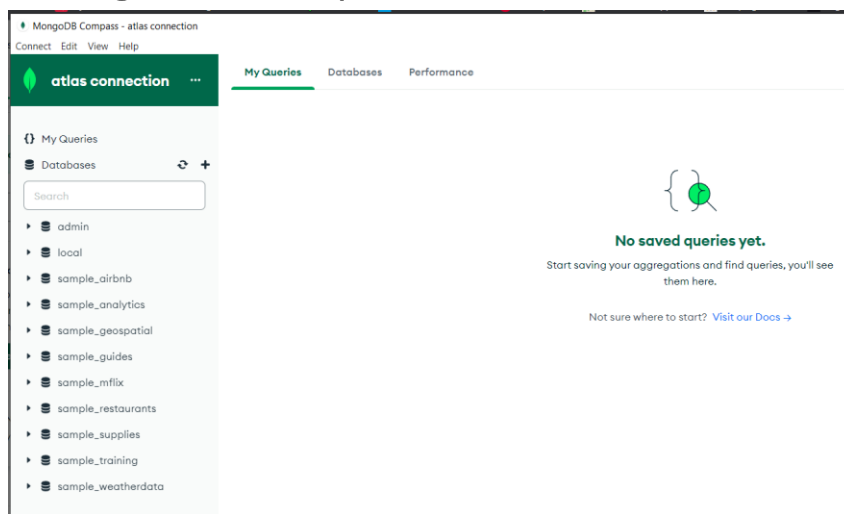
Aby połączyć się z MongoDB Compass należy umieścić podaną komendę w MongoDB Compass w sekcji connect:

2. Copy the connection string, then open MongoDB Compass

```
mongodb+srv://adambista0078:<password>@cluster0.rpzyhdt.mongodb.net/
```

You will be prompted for the password for the **adambista0078** user's (Database User) username. When entering your password, make sure that any special characters are [URL encoded](#).

MongoDB Compass:



4. Przegląd dwóch ciekawych baz danych:

W mongosh wpisujemy:

```
Atlas atlas-m79rir-shard-0 [primary] test> use sample_airbnb
switched to db sample_airbnb
Atlas atlas-m79rir-shard-0 [primary] sample_airbnb> show collections
listingsAndReviews
```

Następnie:

```
Atlas atlas-m79rir-shard-0 [primary] sample_airbnb> db.listingsAndReviews.find().pretty()
```

To nam wyświetli ciąg danych w bazie.

W MongoDB Atlas:



Przykładowe dane z tej kolekcji:

```
{
  "_id": "238803427",
  "date": ISODate("2018-02-27T05:00:00.000Z"),
  "listing_id": "10091713",
  "reviewer_id": "13988353",
  "reviewer_name": "Selina",
  "comments": "We travelled with a 4 month old baby. Ben's place was a great location, close to everything - shops, food, transport. Ben was a great communicator and offered us a late check out which was perfect."
},
{
  "_id": "240499794",
  "date": ISODate("2018-03-05T05:00:00.000Z"),
  "listing_id": "10091713",
  "reviewer_id": "155977269",
  "reviewer_name": "Kristie",
  "comments": "Wonderful location, spectacular views."
},
}
```

Każdy obiekt w tej kolekcji (nazywany także dokumentem) reprezentuje jedną recenzję:

_id: Jest to unikalne ID dla każdej recenzji. MongoDB generuje to automatycznie dla każdego nowego dokumentu, chociaż można to też ustawić ręcznie.

date: Data, kiedy recenzja została napisana. Jest to zapisane jako ISODate, co jest formatem daty zgodnym z ISO 8601.

listing_id: Jest to ID nieruchomości, do której odnosi się recenzja. Wszystkie recenzje z tym samym listing_id dotyczą tej samej nieruchomości.

reviewer_id: Jest to ID osoby, która napisała recenzję. Każdy recenzent ma unikalne ID.

reviewer_name: To jest imię osoby, która napisała recenzję.

comments: To są rzeczywiste treści recenzji.

Inna baza danych:

W Mongosh wpisujemy:

```
Atlas atlas-m79rir-shard-0 [primary] sample_airbnb> use sample_geospatial
switched to db sample_geospatial
Atlas atlas-m79rir-shard-0 [primary] sample_geospatial> show collections
shipwrecks
Atlas atlas-m79rir-shard-0 [primary] sample_geospatial> _
```

Następnie:

```
Atlas atlas-m79rir-shard-0 [primary] sample_geospatial> db.shipwrecks.find().pretty()
```

Dwa przykładowe dane które się wyświetlą:

```
{
  _id: ObjectId("578f6fa2df35c7fbd8aed8d6"),
  recrd: "",
  vesslterms: "",
  feature_type: 'Wrecks - Visible',
  chart: 'US,US,reprt,L-1218/15',
  latdec: 9.5690002,
  londec: -79.0378342,
  gp_quality: "",
  depth: 0,
  sounding_type: "",
  history: "",
  quasou: "",
  watlev: 'always dry',
  coordinates: [ -79.0378342, 9.5690002 ]
},
{
  _id: ObjectId("578f6fa2df35c7fbd8aed8d7"),
  recrd: "",
  vesslterms: "",
  feature_type: 'Wrecks - Visible',
  chart: 'US,US,reprt,L-1218/15',
  latdec: 9.5574865,
  londec: -78.8790131,
  gp_quality: "",
  depth: 0,
  sounding_type: "",
  history: "",
  quasou: "",
  watlev: 'always dry',
  coordinates: [ -78.8790131, 9.5574865 ]
}
```

W MongoDB Atlas:

- ▶ sample_analytics
- ▼ sample_geospatial
 - | shipwrecks
- ▶ sample_guides
- ▶ sample_mflix
- ▶ sample_restaurants
- ▶ sample_supplies
- ▶ sample_training
- ▶ sample_weatherdata

[Filter](#) [Type a query: { field: 'value' }](#) [Reset](#)

QUERY RESULTS: 1-20 OF MANY

```
_id: ObjectId('578f6fa2df35c7fbd8aed8c4')
recrd: ""
vesslterms: ""
feature_type: "Wrecks - Visible"
chart: "US,U1,graph,DNC H1409860"
latdec: 9.3547792
londec: -79.9081268
```

[<](#) PREVIOUS 1-20 of many results

Każdy obiekt w tej kolekcji (nazywany także dokumentem) reprezentuje jeden wrak. Oto krótkie wyjaśnienie pól w każdym dokumencie:

`_id`: Jest to unikalne ID dla każdego wraku. MongoDB generuje to automatycznie dla każdego nowego dokumentu, chociaż można to także ustawić ręcznie.

`recrd`: To pole mogłoby zawierać dodatkowe informacje o rekordzie, ale w tych przykładach jest puste.

`vesslterms`: To pole mogłoby zawierać terminy związane z nieruchomością, ale w tych przykładach jest puste.

`feature_type`: Typ obiektu, tutaj "Wrecks - Visible" wskazuje, że wraki są widoczne.

`chart`: to pole zawiera informacje o mapie lub wykresie, na którym wrak jest oznaczony.

`latdec` i `londec`: Są to dziesiętne wartości szerokości i długości geograficznej, które wskazują na lokalizację wraku.

`gp_quality`: To pole mogłoby zawierać informacje o jakości danych geoprzestrzennych, ale w tych przykładach jest puste.

`depth`: Głębokość, na której znajduje się wrak.

`sounding_type`: To pole mogłoby zawierać informacje o typie sondowania używanym do zidentyfikowania wraku, ale w tych przykładach jest puste.

`history`: To pole mogłoby zawierać informacje historyczne o wraku, ale w tych przykładach jest puste.

`quasou`: Nieznane pole, mogłoby zawierać dodatkowe informacje.

`watlev`: Poziom wody w odniesieniu do wraku. W tych przykładach jest "always dry", co sugeruje, że wraki są zawsze nad wodą.

`coordinates`: Tablica zawierająca długość i szerokość geograficzną wraku. Wartości te odpowiadają wartościom w polach `latdec` i `londec`.

5. Tworzenie nowej bazy danych:

Posłużę się do tego MongoDB Compass, użyję serwera lokalnego

Create Database

Database Name

AB

Collection Name

student

☐ 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)

abc

Cancel

Create Database

My Queries

Databases

Search

AB

student

admin

config

local

AB.student

Documents

Aggregations

Schema

Explain Plan

Indexes

Validation

Filter

Type a query: { field: 'value' }

Reset

ADD DATA

EXPORT COLLECTION

0 - 0

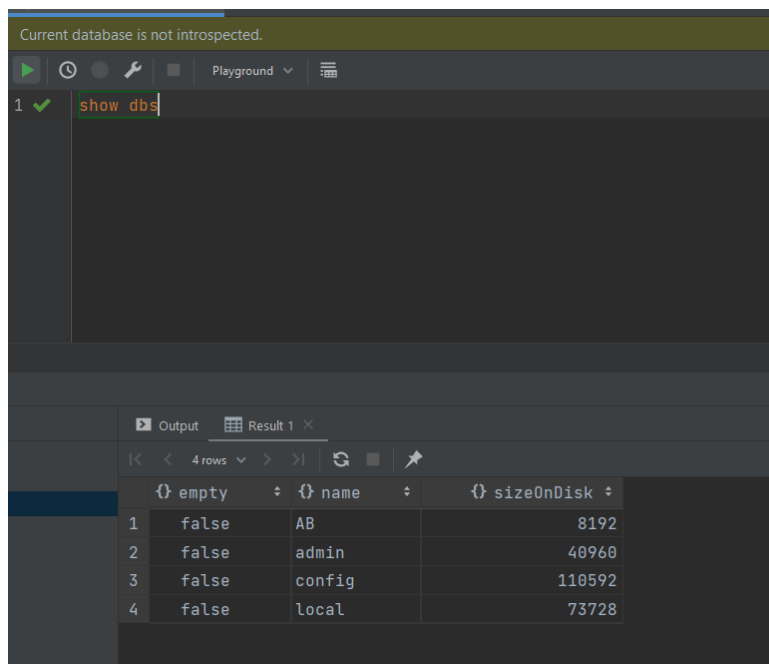


This collection has no data

It only takes a few seconds to import data from a JSON or CSV file.

Import Data

DataGrip:



Niech struktura dokumenty dla kolekcji „student” będzie wyglądać następująco:

```
{
  _id: ObjectId(),
  name: "Adam Biśta",
  age: 20,
  university: "AGH",
  major: "Computer Science",
  courses: [
    {
      courseName: "Database Systems",
      grade: "A"
    },
    {
      courseName: "Operating Systems",
      grade: "B"
    },
    // ... więcej przedmiotów
  ]
}
```

Wykonujemy poniższe polecenia w dataGripie:

Wstawianie:

```
use AB;
db.student.insert({
  name: "Adam Biśta",
  age: 20,
  university: "AGH",
  major: "Computer Science",
  courses: [
    {
      courseName: "Database Systems",
      grade: "A"
    },
    {
      courseName: "Operating Systems",
      grade: "B"
    }
  ]
})
```

Output

Result 3

1 row

result

1 switched to db AB

Wynik w konsoli:

```
test> show dbs
AB      8.00 KiB
admin   40.00 KiB.00 KiB
config  108.00 KiB.00 KiB
local   72.00 KiB.00 KiB
test> use AB
switched to db AB
AB> show collections
student
AB> db.student.find().pretty()

AB> db.student.find().pretty()
[
  {
    _id: ObjectId("645cc3fa097d4d3af1b82d7f"),
    name: 'Adam Biśta',
    age: 20,
    university: 'AGH',
    major: 'Computer Science',
    courses: [
      { courseName: 'Database Systems', grade: 'A' },
      { courseName: 'Operating Systems', grade: 'B' }
    ]
  }
]
AB>
```

Modyfikowanie:

```
use AB;
db.student.update(
  { name: "Adam Biśta" }, // kryteria wyszukiwania
  { $set: { "courses.$[elem].grade" : "A+" } }, // nowa wartość
  { arrayFilters: [ { "elem.courseName": "Operating Systems" } ] } // określ, który element tablicy ma zostać zaktualizowany
)
```

Wynik:

```
AB> db.student.find().pretty()
[
  {
    _id: ObjectId("645cc3fa097d4d3af1b82d7f"),
    name: 'Adam Biśta',
    age: 20,
    university: 'AGH',
    major: 'Computer Science',
    courses: [
      { courseName: 'Database Systems', grade: 'A' },
      { courseName: 'Operating Systems', grade: 'A+' }
    ]
  }
]
AB>
```

Usuwanie:

```
use AB;
db.student.remove(
  { name: "Adam Biśta" } // kryteria wyszukiwania
)
```

Wynik:

```
AB> db.student.find().pretty()
AB>
```

Spróbujmy wyszukać ten obiekt, który usunelismy:

```
use AB;
db.student.find(
  { name: "Adam Biśta" } // kryteria wyszukiwania
)
```

Output Result 6 Result 7

0 rows

map