https://www.halvorsen.blog

Python and MongoDB

Hans-Petter Halvorsen

Free Textbook with lots of Practical Examples

Python for Software Development

Hans-Petter Halvorsen

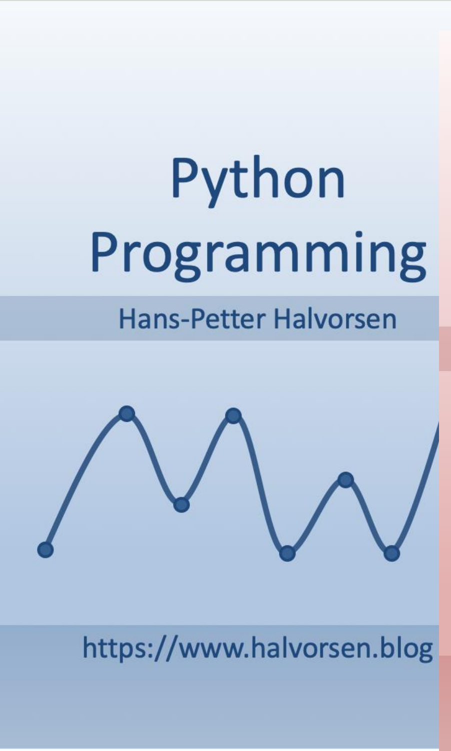
Python Software Development

Do you want to learn Software Development?

https://www.halvorsen.blog

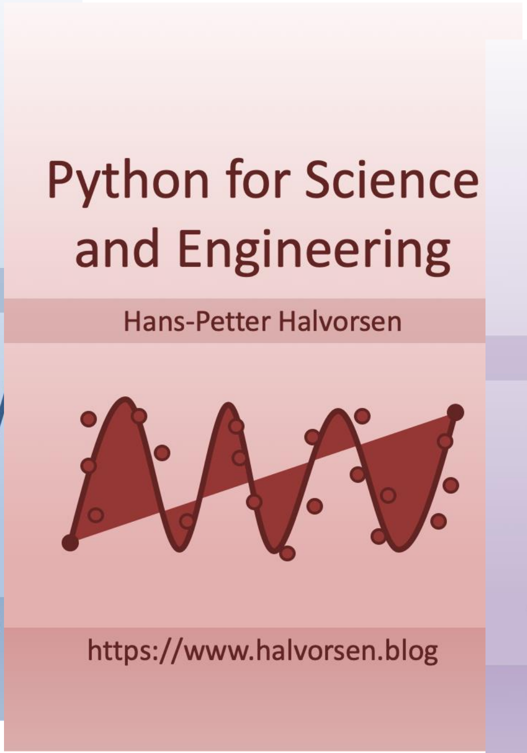
https://www.halvorsen.blog/documents/programming/python/

Additional Python Resources


Python Programming

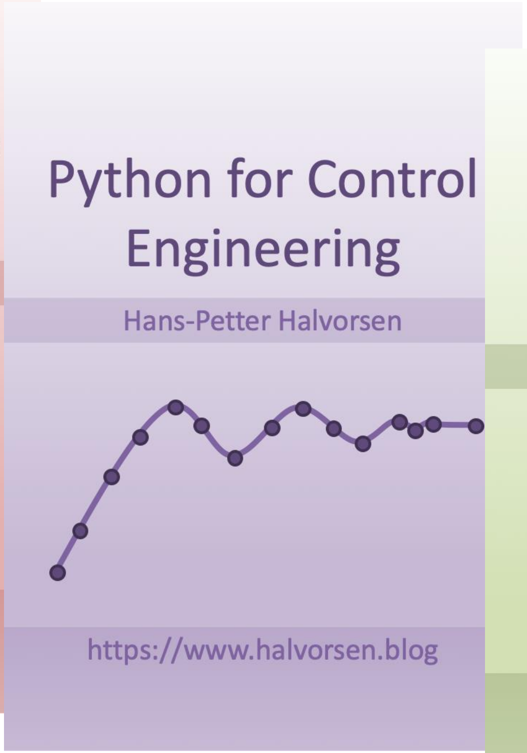
Hans-Petter Halvorsen

https://www.halvorsen.blog


Python for Science and Engineering

Hans-Petter Halvorsen

https://www.halvorsen.blog


Python for Control Engineering

Hans-Petter Halvorsen

https://www.halvorsen.blog

Python for Software Development

Hans-Petter Halvorsen

Python Software Development

Do you want to learn Software Development?

OK Cancel

https://www.halvorsen.blog

https://www.halvorsen.blog/documents/programming/python/

Contents

MongoDB

- MongoDB is a cross-platform document-oriented database program.
- MongoDB is a NoSQL database program
- MongoDB uses JSON-like documents
- Home Page: https://www.mongodb.com/

Software:

- MongoDB Community Server – Free version of the MongoDB Server which can be installed locally on your computer or a server
- MongoDB Atlas – Premade MongoDB ready to use in the Cloud
- MongoDB Compass – GUI for connecting to and manipulating your MongoDB database
- PyMongo – MongoDB Driver for Python

MongoDB Community Server

- Free version of the MongoDB Server
- MongoDB Server can be installed locally on your computer or on an external server

https://www.mongodb.com/try/download/community

MongoDB Community Server will be used in this Tutorial. So just download and install the MongoDB Community Server, then you are ready to follow this Tutorial

MongoDB Atlas

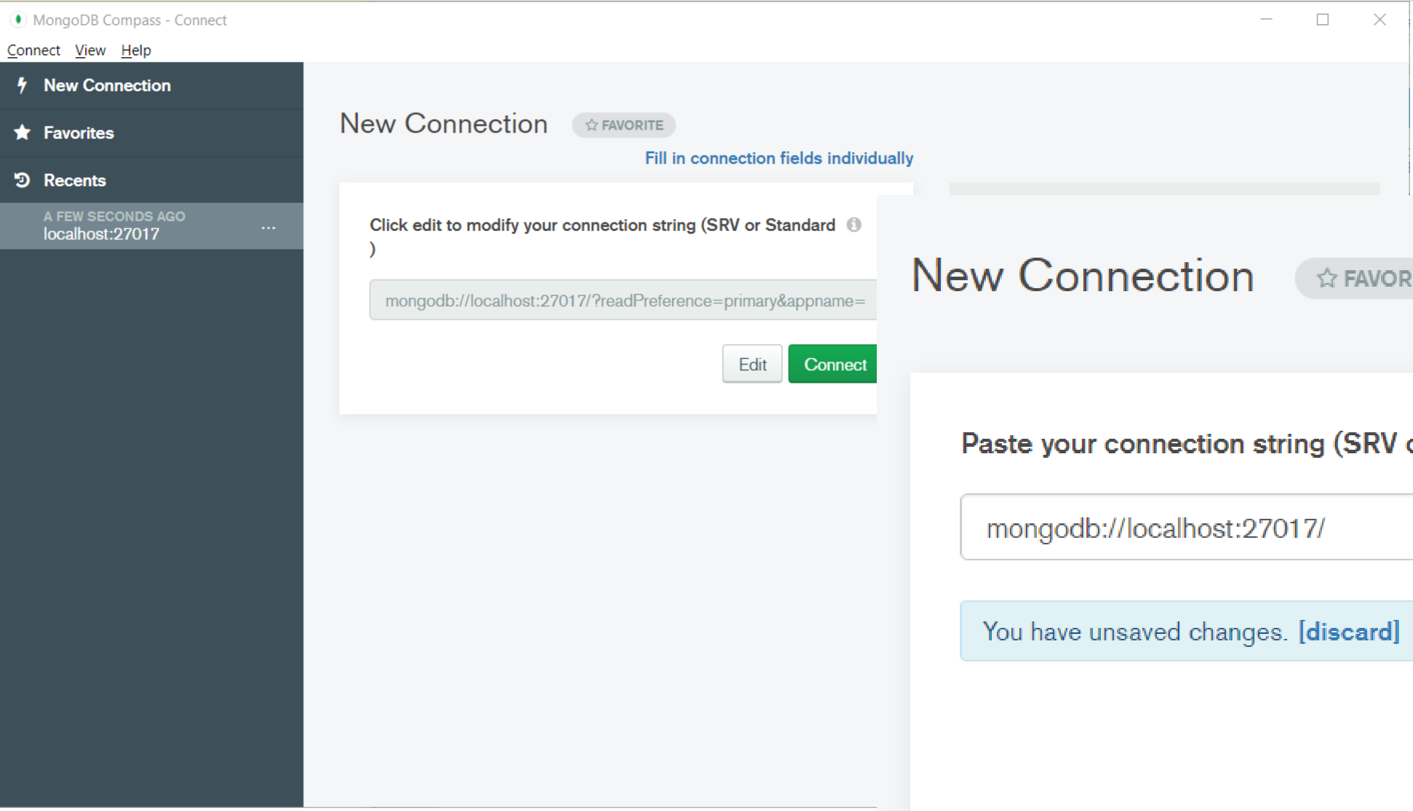
- Premade MongoDB ready to use in the Cloud
- You can use a Shared Clusters for free
- Purpose: Learning MongoDB or developing small applications

https://www.mongodb.com/cloud/atlas

MongoDB Compass

- MongoDB Compass is the official Graphical User Interface (GUI) for MongoDB
- With MongoDB Compass you can explore and manipulate your MongoDB data
- To use Compass, you must connect to an existing MongoDB database. You can connect to:
  - A MongoDB server that you have installed locally, or
  - A MongoDB Atlas cluster.

MongoDB Compass



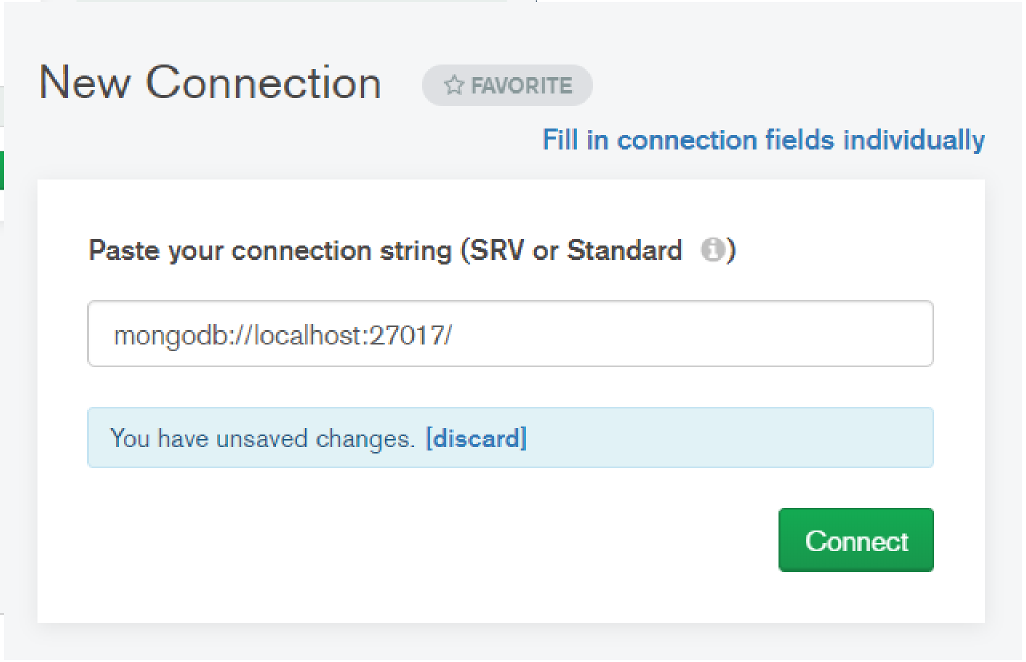
MongoDB Compass - Connect

A FEW SECONDS AGO
localhost:27017

New Connection

Click edit to modify your connection string (SRV or Standard)

```
mongodb://localhost:27017/?readPreference=primary&appname=
```



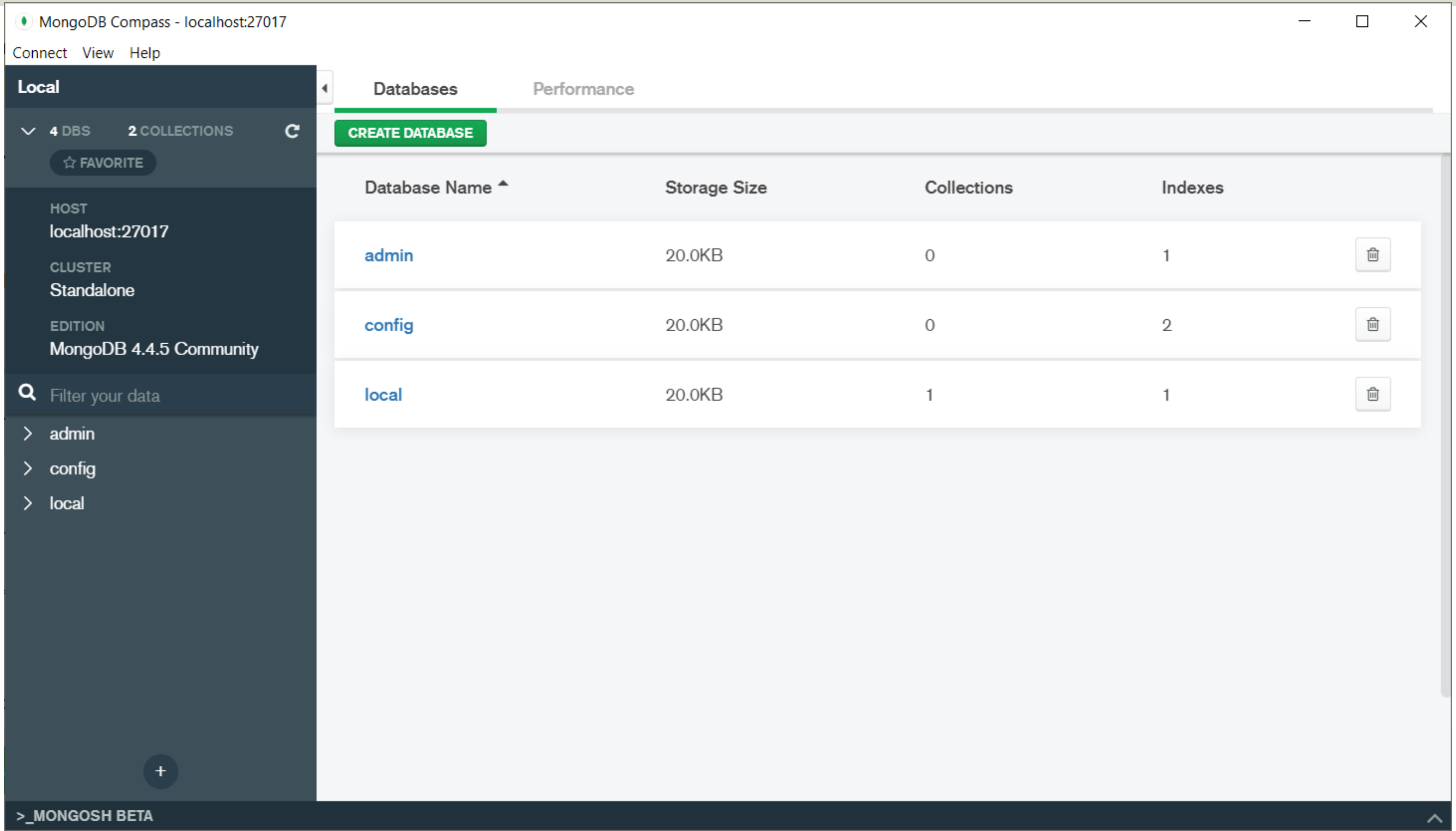
New Connection

Paste your connection string (SRV or Standard)

```
mongodb://localhost:27017/
```

You have unsaved changes. [discard]

MongoDB Compass



HOST
localhost:27017

CLUSTER
Standalone

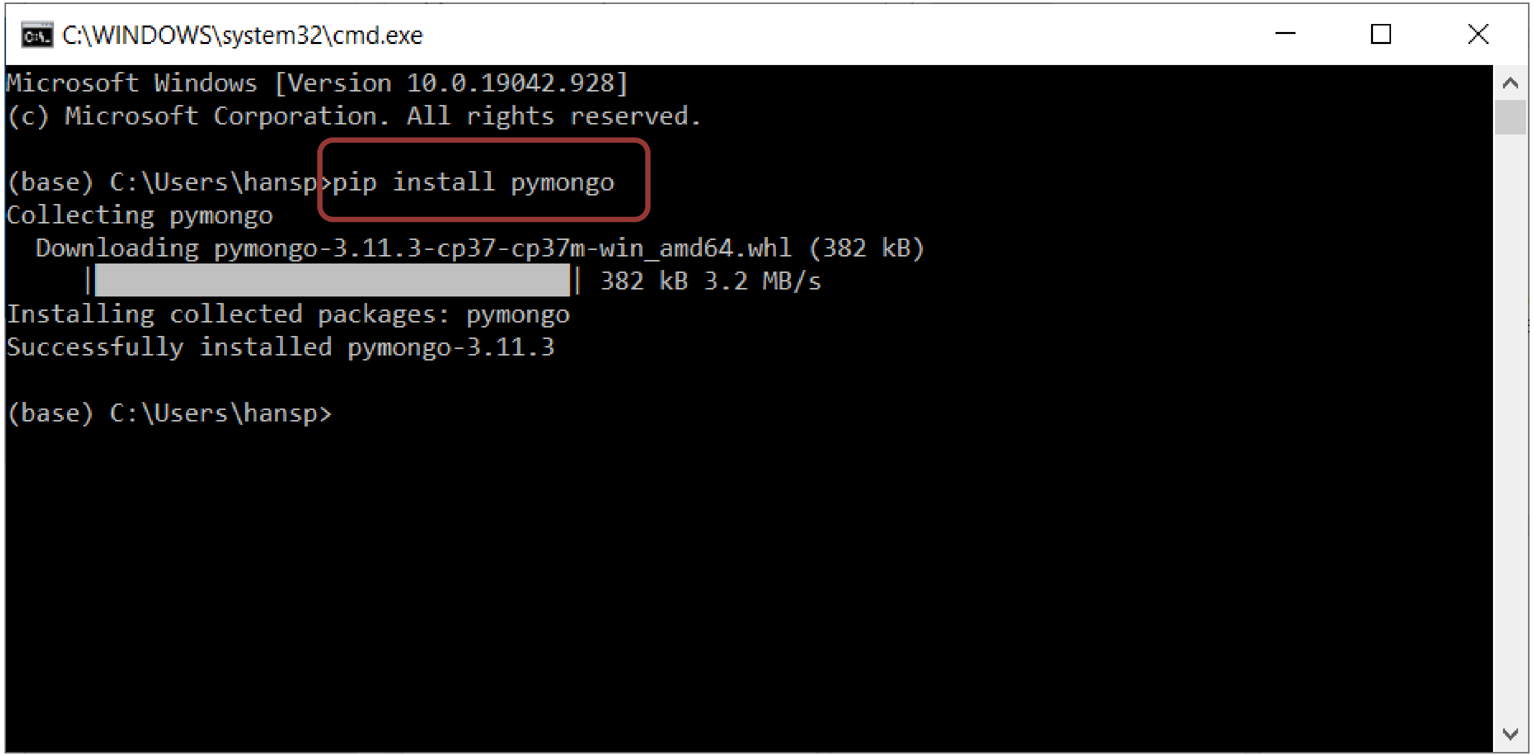
EDITION
MongoDB 4.4.5 Community

Database Name	Storage Size	Collections	Indexes
admin	20.0KB	0	1
config	20.0KB	0	2
local	20.0KB	1	1

PyMongo

- The PyMongo package contains tools for interacting with MongoDB database from Python
- The PyMongo package is a native Python driver for MongoDB
- Install using PIP: `pip install pymongo`
- https://pypi.org/project/pymongo/

PyMongo Installation


C:\WINDOWS\system32\cmd.exe

```
Microsoft Windows [Version 10.0.19042.928]
(c) Microsoft Corporation. All rights reserved.

(base) C:\Users\hansp>pip install pymongo
Collecting pymongo
  Downloading pymongo-3.11.3-cp37-cp37m-win_amd64.whl (382 kB)
     |████████████████████████████████| 382 kB 3.2 MB/s
Installing collected packages: pymongo
Successfully installed pymongo-3.11.3

(base) C:\Users\hansp>
```

SQL vs MongoDB

Note the following:

- A **collection** in MongoDB is the same as a table in SQL databases.
- A **document** in MongoDB is the same as a record in SQL databases.

Python Examples

Database CRUD

All Database Systems supports CRUD

C – Create or Insert Data

R – Retrieve Data

U – Update Data

D – Delete Data

Let's go through some Python examples

https://www.halvorsen.blog

Insert Data

Hans-Petter Halvorsen

Python

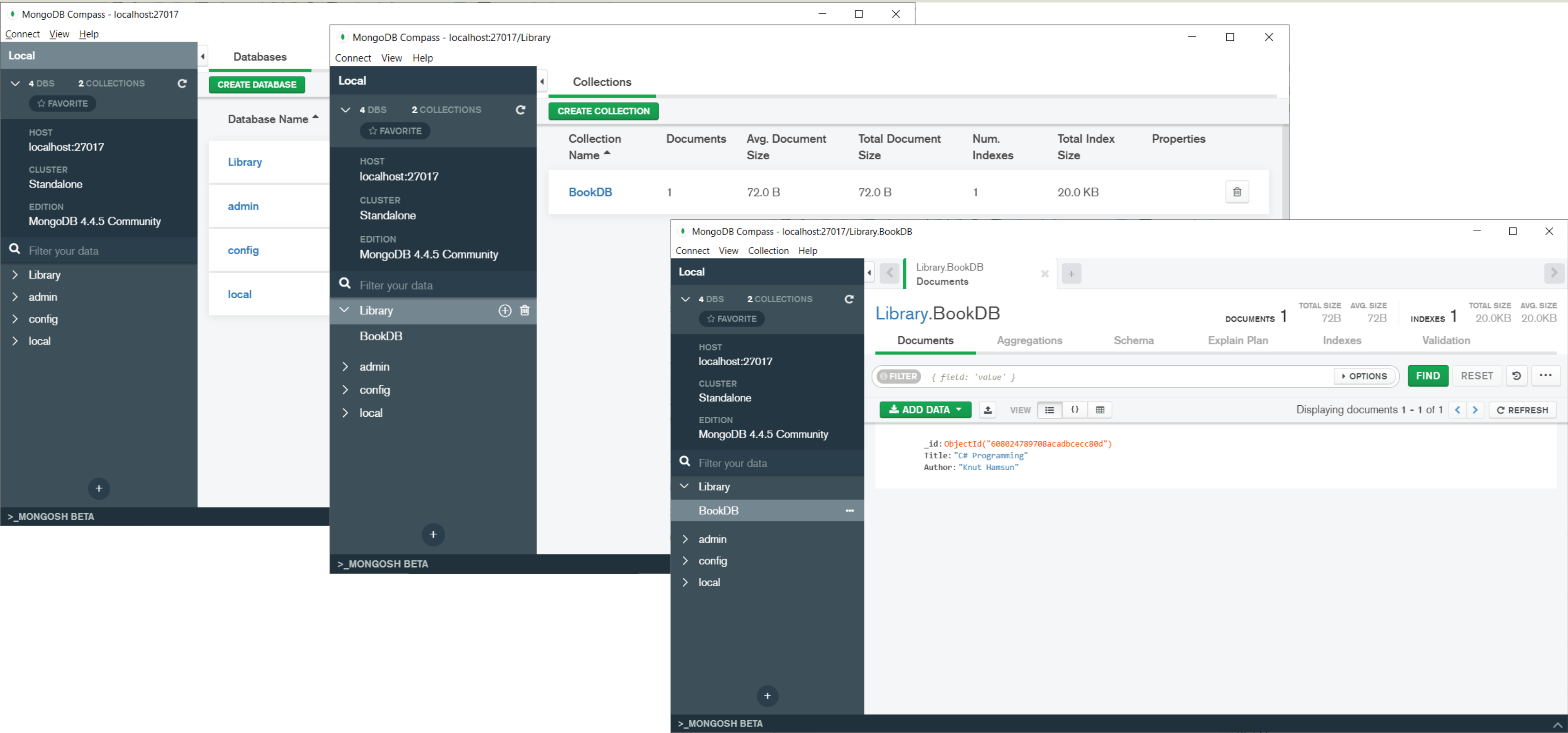
Python script that creates a Database ("Library"), a Collection ("BookDB") and a Document.

In a SQL database we use the INSERT statement to insert data in a table.

In MongoDB we use the **insert_one()** and **insert_many()** methods to insert data into a collection.

```
import pymongo

client = pymongo.MongoClient("mongodb://localhost:27017/")
database = client["Library"]
collection = database["Book"]

document = { "Title": "C# Programming", "Author": "Knut Hamsun" }

x = collection.insert_one(document)
```

MongoDB Compass



Collection Name	Documents	Avg. Document Size	Total Document Size	Num. Indexes	Total Index Size	Properties
BookDB	1	72.0 B	72.0 B	1	20.0 KB	

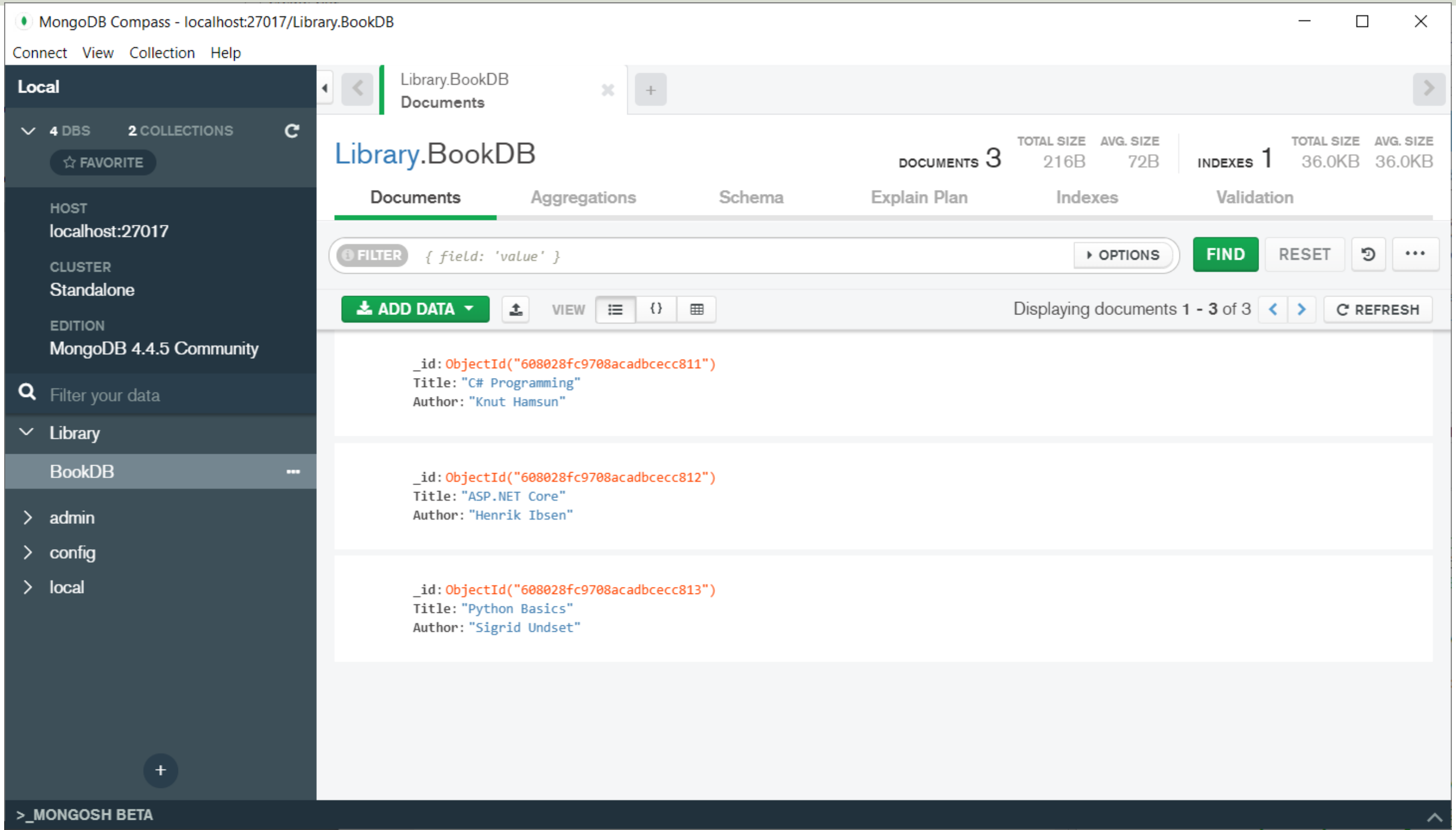
```
_id: ObjectId("608024789708acadbcecc80d")
Title: "C# Programming"
Author: "Knut Hamsun"
```

Insert Multiple Documents

- To insert a record, or document as it is called in MongoDB, into a collection, we use the **insert_one()** method.
- To insert multiple documents into a collection in MongoDB, we use the **insert_many()** method.

```
import pymongo

client = pymongo.MongoClient("mongodb://localhost:27017/")
database = client["Library"]
collection = database["Book"]

documents = [
    { "Title": "C# Programming", "Author": "Knut Hamsun" },
    { "Title": "ASP.NET Core", "Author": "Henrik Ibsen" },
    { "Title": "Python Basics", "Author": "Sigrid Undset" }
    ]

x = collection.insert_many(documents)
```

MongoDB Compass



```
_id: ObjectId("608028fc9708acadbcecc811")
Title: "C# Programming"
Author: "Knut Hamsun"

_id: ObjectId("608028fc9708acadbcecc812")
Title: "ASP.NET Core"
Author: "Henrik Ibsen"

_id: ObjectId("608028fc9708acadbcecc813")
Title: "Python Basics"
Author: "Sigrid Undset"
```

Retrieve Data

Retrieve Data

In a SQL database we use the SELECT to retrieve data in a table.

In MongoDB we use the **find()** and **find_one()** methods to find data in a collection.

```
import pymongo

client = pymongo.MongoClient("mongodb://localhost:27017/")
database = client["Library"]
collection = database["Book"]

x = collection.find_one()

print(x)
```

```
{'_id': ObjectId('608028fc9708acadbcecc811'), 'Title': 'C#
Programming', 'Author': 'Knut Hamsun'}
```

Retrieve All Data

```
import pymongo

client = pymongo.MongoClient("mongodb://localhost:27017/")
database = client["Library"]
collection = database["Book"]

for x in collection.find():
    print(x)
```

```
{'_id': ObjectId('608028fc9708acadbcecc811'), 'Title': 'C#
Programming', 'Author': 'Knut Hamsun'}
{'_id': ObjectId('608028fc9708acadbcecc812'), 'Title':
'ASP.NET Core', 'Author': 'Henrik Ibsen'}
{'_id': ObjectId('608028fc9708acadbcecc813'), 'Title':
'Python Basics', 'Author': 'Sigrid Undset'}
```

Retrieve specific Data

```
import pymongo

client = pymongo.MongoClient("mongodb://localhost:27017/")
database = client["Library"]
collection = database["Book"]

query = { "Author": "Knut Hamsun" }

documents = collection.find(query)

for x in documents:
    print(x)
```

```
{'_id': ObjectId('608028fc9708acadbcecc811'), 'Title': 'C#
Programming', 'Author': 'Knut Hamsun'}
```

Sort the Results

Use the sort() method to sort the result in ascending or descending order.
The **sort()** method takes one parameter for "fieldname" and one parameter for "direction" (ascending is the default direction).

```
import pymongo

client = pymongo.MongoClient("mongodb://localhost:27017/")
database = client["Library"]
collection = database["Book"]

documents = collection.find().sort("Title")

for x in documents:
    print(x)
```

```
{'_id': ObjectId('608028fc9708acadbcecc812'), 'Title': 'ASP.NET Core', 'Author': 'Henrik Ibsen'}
{'_id': ObjectId('608028fc9708acadbcecc811'), 'Title': 'C# Programming', 'Author': 'Knut Hamsun'}
{'_id': ObjectId('608028fc9708acadbcecc813'), 'Title': 'Python Basics', 'Author': 'Sigrid Undset'}
```

Update Data

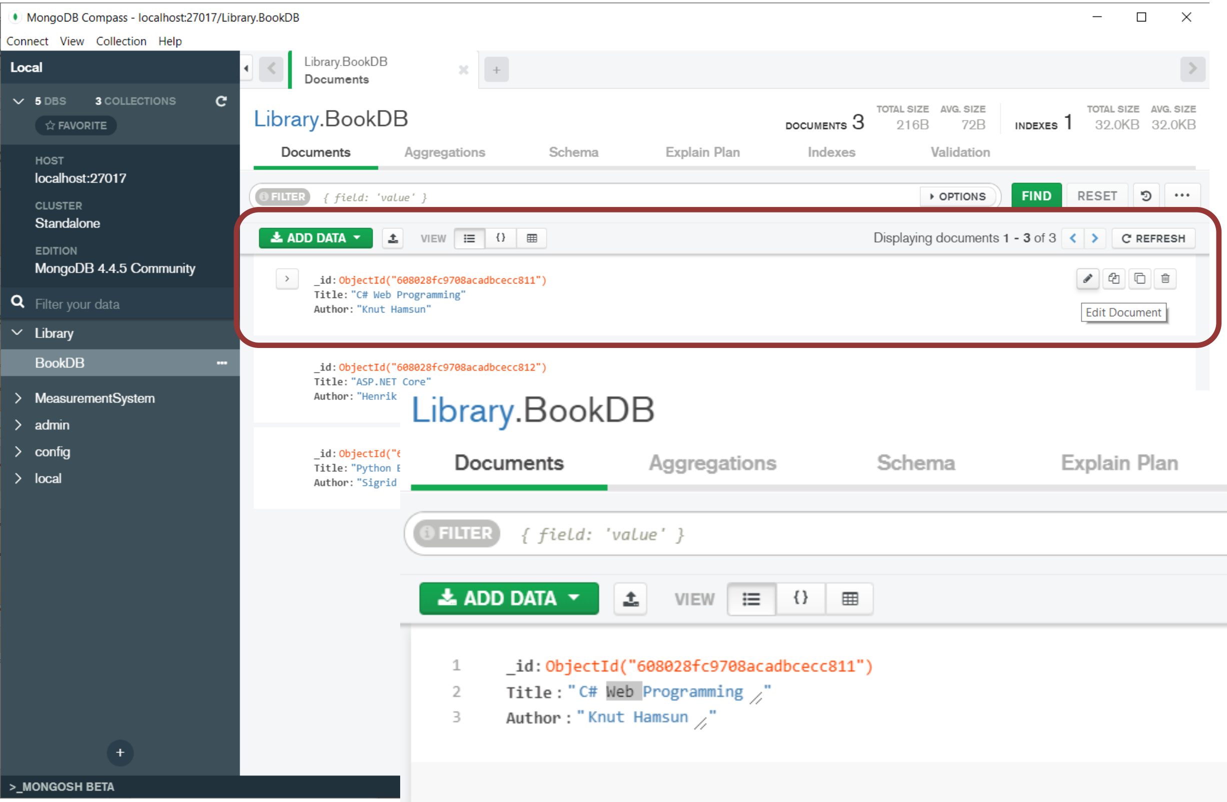
Update Data

You can update a record, or document as it is called in MongoDB, by using the **update_one()** method.

```
import pymongo

client = pymongo.MongoClient("mongodb://localhost:27017/")
database = client["Library"]
collection = database["Book"]

query = { "Title": "C# Programming" }
newvalue = { "$set": { "Title": "C# Web Programming" } }

collection.update_one(query, newvalue)

documents = collection.find()

for x in documents:
    print(x)
```

Update Data

We can also Update Data in the Database using the MongoDB Compass



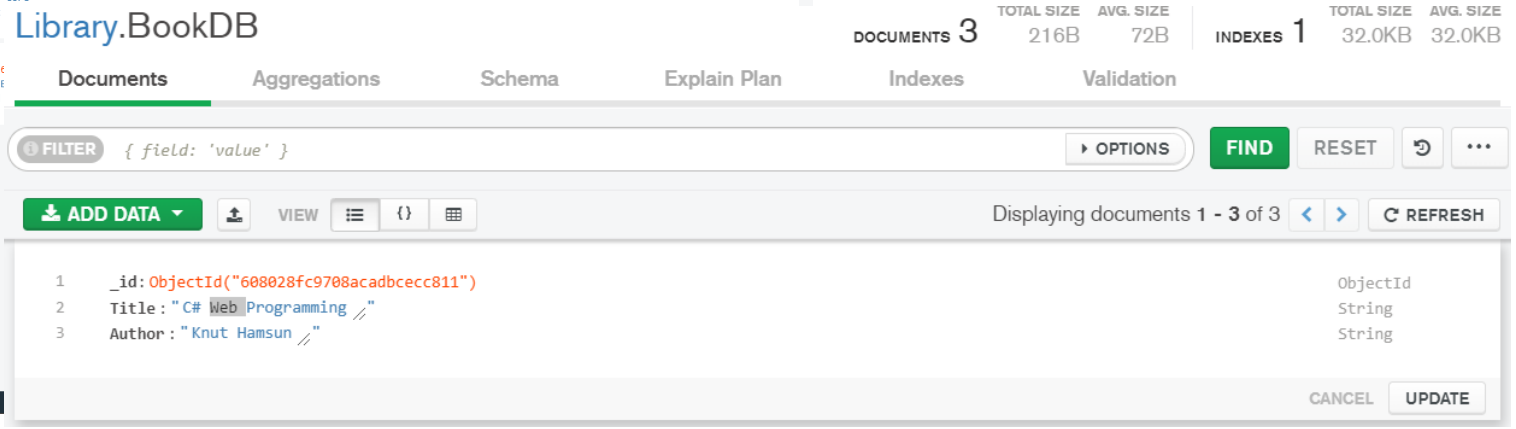
MongoDB Compass - localhost:27017/Library.BookDB

Connect View Collection Help

```
{ field: 'value' }
```

Displaying documents 1 - 3 of 3

```
_id: ObjectId("608028fc9708acadbcecc811")
Title: "C# Web Programming"
Author: "Knut Hamsun"

_id: ObjectId("608028fc9708acadbcecc812")
Title: "ASP.NET Core"
Author: "Henrik

_id: ObjectId("6
Title: "Python E
Author: "Sigrid
```



```
{ field: 'value' }
```

Displaying documents 1 - 3 of 3

```
_id: ObjectId("608028fc9708acadbcecc811")   ObjectId
Title : "C# Web Programming"                 String
Author : "Knut Hamsun"                       String
```

Delete Data

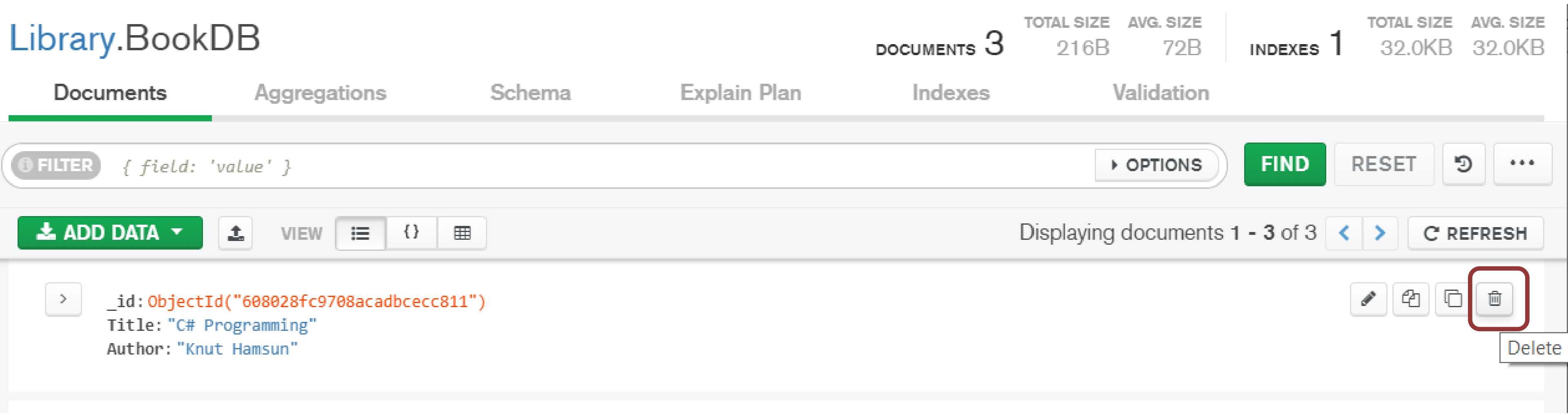
Delete Data

You can delete a record, or document as it is called in MongoDB, by using the **delete_one()** method.

```
import pymongo

client = pymongo.MongoClient("mongodb://localhost:27017/")
database = client["Library"]
collection = database["Book"]


query = { "Title": "C# Programming" }

collection.delete_one(query)


documents = collection.find()

for x in documents:
    print(x)
```

Delete Data

We can also Update Data in the Database using the MongoDB Compass



```
_id: ObjectId("608028fc9708acadbcecc811")
Title: "C# Programming"
Author: "Knut Hamsun"
```

Datalogging Example

Hans-Petter Halvorsen

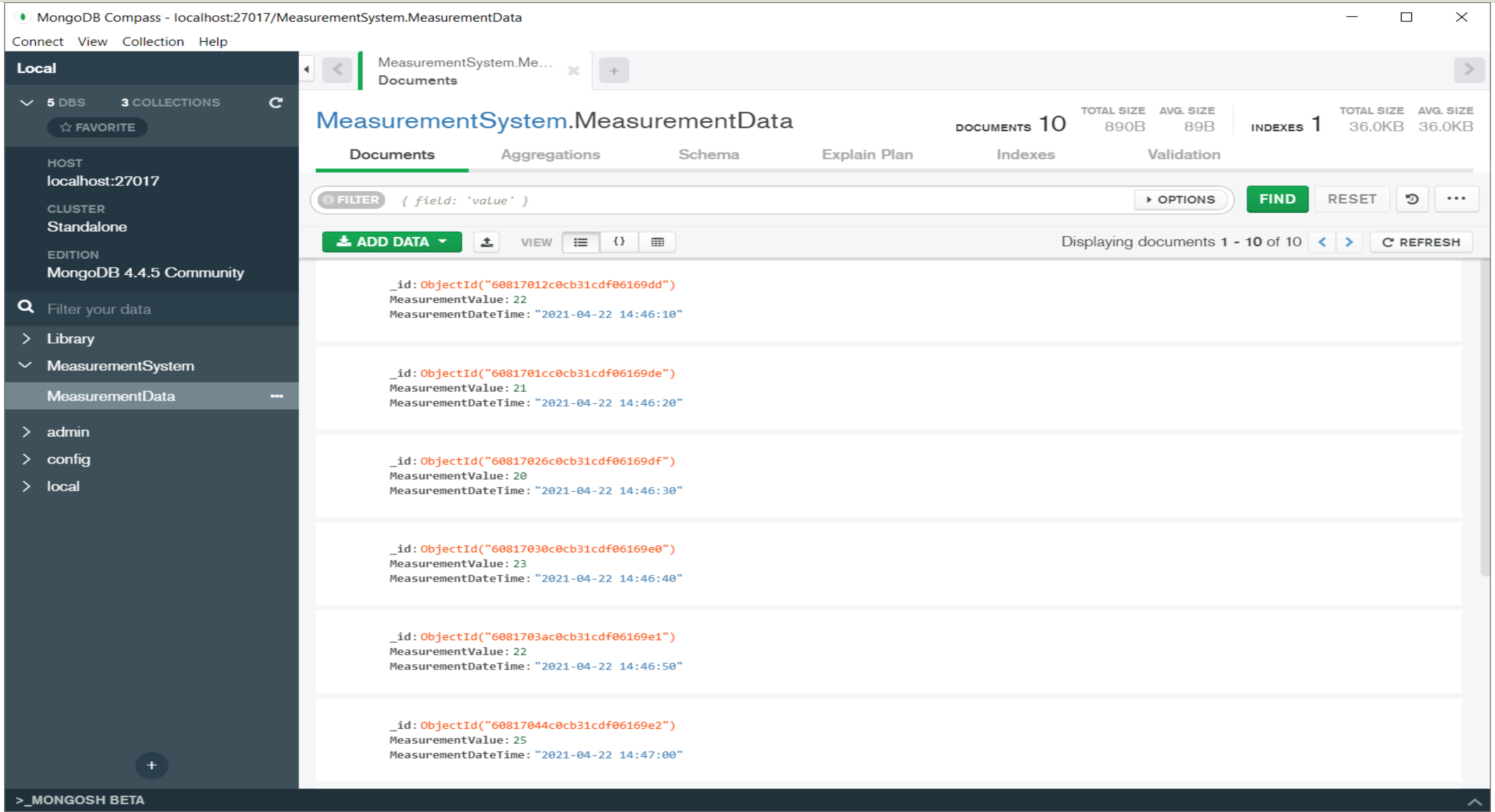
Datalogging Example

- We can log data from a DAQ device or similar
- We start by creating a simple Random Generator that simulates a Temperature Sensor and log these data to the MongoDB database
- Then we will in another script read the data from the database and plot them.

Logging Data

```python
import pymongo
import random
import time
from datetime import datetime

# Create Database
client = pymongo.MongoClient("mongodb://localhost:27017/")
database = client["MeasurementSystem"]
collection = database["MeasurementData"]

Ts = 10 # Sampling Time
N = 10
for k in range(N):
    # Generate Random Data
    LowLimit = 20
    UpperLimit = 25
    MeasurementValue = random.randint(LowLimit, UpperLimit)

    #Find Date and Time
    now = datetime.now()
    datetimeformat = "%Y-%m-%d %H:%M:%S"
    MeasurementDateTime = now.strftime(datetimeformat)

    # Insert Data into Database
    document = { "MeasurementValue": MeasurementValue, "MeasurementDateTime": 
MeasurementDateTime }
    x = collection.insert_one(document)

    # Wait
    time.sleep(Ts)
```

Logged Data



HOST
localhost:27017

CLUSTER
Standalone

EDITION
MongoDB 4.4.5 Community

MeasurementSystem.MeasurementData

DOCUMENTS 10 TOTAL SIZE 890B AVG. SIZE 89B INDEXES 1 TOTAL SIZE 36.0KB AVG. SIZE 36.0KB

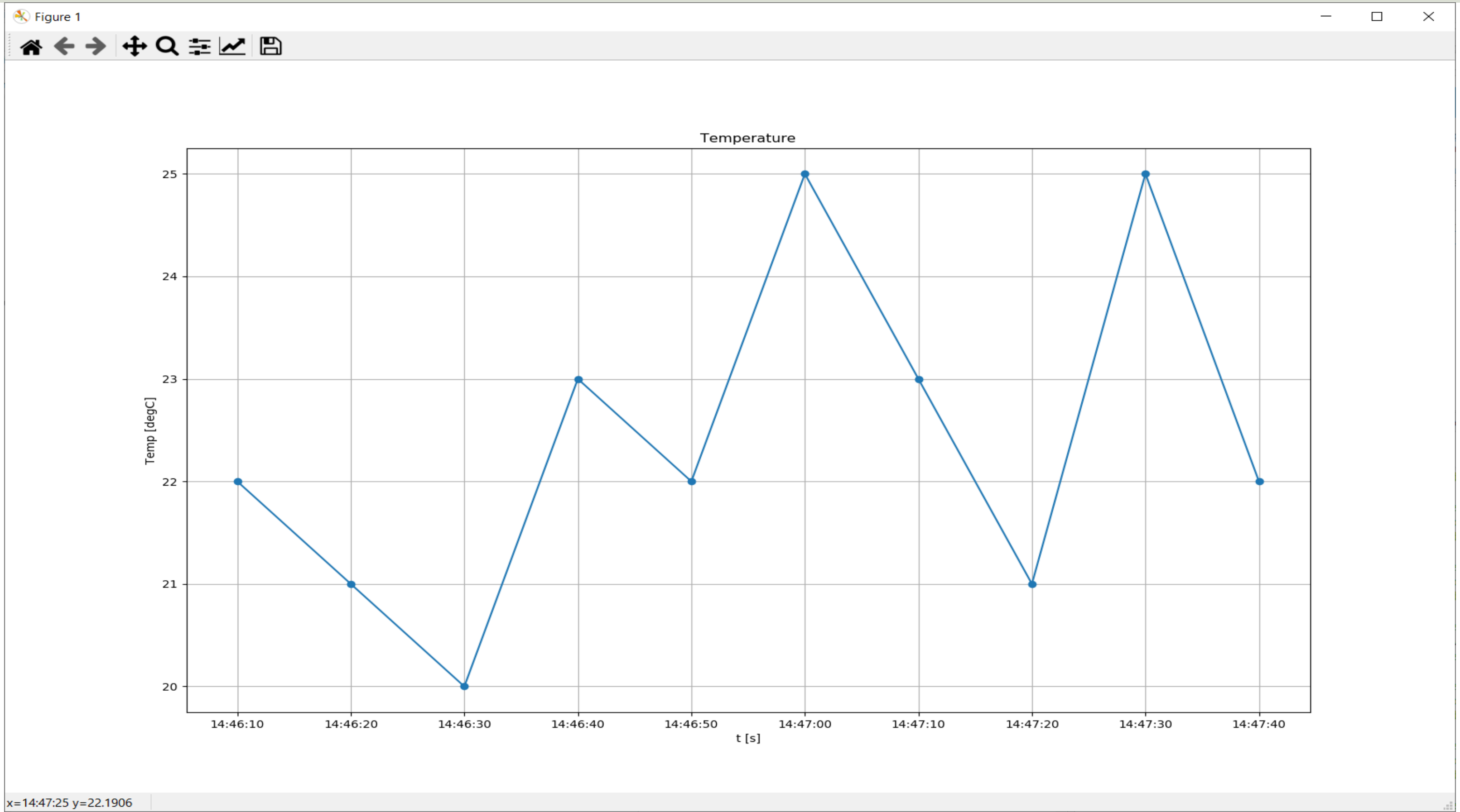
Displaying documents 1 - 10 of 10

```
_id: ObjectId("60817012c0cb31cdf06169dd")
MeasurementValue: 22
MeasurementDateTime: "2021-04-22 14:46:10"

_id: ObjectId("6081701cc0cb31cdf06169de")
MeasurementValue: 21
MeasurementDateTime: "2021-04-22 14:46:20"

_id: ObjectId("60817026c0cb31cdf06169df")
MeasurementValue: 20
MeasurementDateTime: "2021-04-22 14:46:30"

_id: ObjectId("60817030c0cb31cdf06169e0")
MeasurementValue: 23
MeasurementDateTime: "2021-04-22 14:46:40"

_id: ObjectId("6081703ac0cb31cdf06169e1")
MeasurementValue: 22
MeasurementDateTime: "2021-04-22 14:46:50"

_id: ObjectId("60817044c0cb31cdf06169e2")
MeasurementValue: 25
MeasurementDateTime: "2021-04-22 14:47:00"
```

Plotting Data

```python
import pymongo
import matplotlib.pyplot as plt
from datetime import datetime

# Connect to Database
client = pymongo.MongoClient("mongodb://localhost:27017/")
database = client["MeasurementSystem"]
collection = database["MeasurementData"]

t = []
data = []

# Retrieving and Formatting Data
for document in collection.find():
    MeasurementValue = document["MeasurementValue"]
    MeasurementDateTime = document["MeasurementDateTime"]

    timeformat = "%Y-%m-%d %H:%M:%S"
    MeasurementDateTime = datetime.strptime(MeasurementDateTime, timeformat)

    data.append(MeasurementValue)
    t.append(MeasurementDateTime)

# Plotting
plt.plot(t, data, 'o-')
plt.title('Temperature')
plt.xlabel('t [s]')
plt.ylabel('Temp [degC]')
plt.grid()
plt.show()
```

Plotted Data

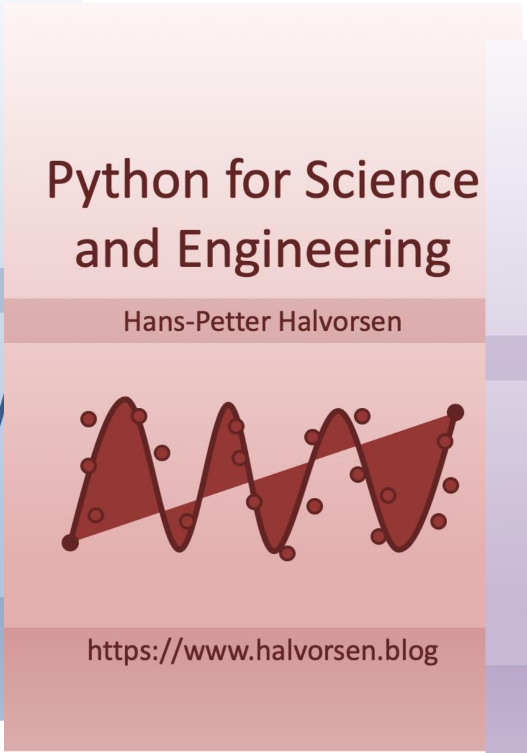


Additional Python Resources

Python Programming

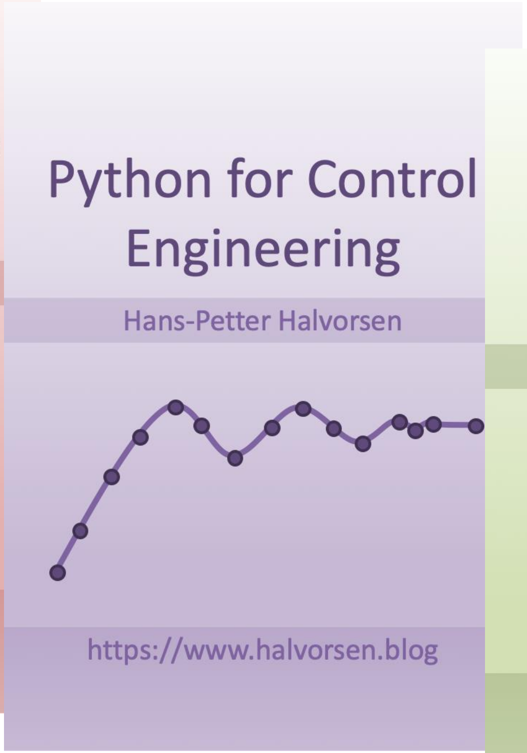
Hans-Petter Halvorsen

https://www.halvorsen.blog


Python for Science and Engineering

Hans-Petter Halvorsen

https://www.halvorsen.blog


Python for Control Engineering

Hans-Petter Halvorsen

https://www.halvorsen.blog

Python for Software Development

Hans-Petter Halvorsen

Python Software Development

Do you want to learn Software Development?

OK Cancel

https://www.halvorsen.blog

https://www.halvorsen.blog/documents/programming/python/

Hans-Petter Halvorsen

University of South-Eastern Norway

www.usn.no

E-mail: hans.p.halvorsen@usn.no

Web: https://www.halvorsen.blog