MongoDB using Python

CRUD operation with MongoDB using python

SAVE IT

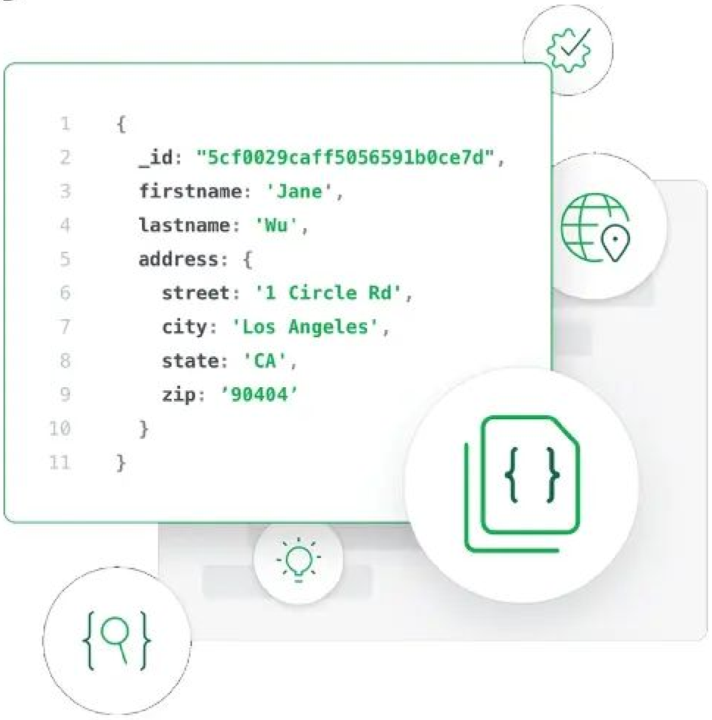
What Is MongoDB?

MongoDB is a document database with the scalability and flexibility that you want with the querying and in dexing that you need

- MongoDB stores data in flexible, JSON-like documents, meaning fields can vary from document to document and data structure can be changed over time
- The document model maps to the objects in your application code, making data easy to work with
- Ad hoc queries, indexing, and real time aggregation provide powerful ways to access and analyze your data
- MongoDB is a distributed database at its core, so high availability, horizontal scaling, and geographic distribution are built in and easy to use

Need to run MongoDB?

- High availability through built-in replication and failover
- Horizontal scalability with native sharding
- End-to-end security
- Native document validation and schema exploration with Compass
- Management tooling for automation, monitoring, and backup
- Fully elastic database as a service with built-in best practices

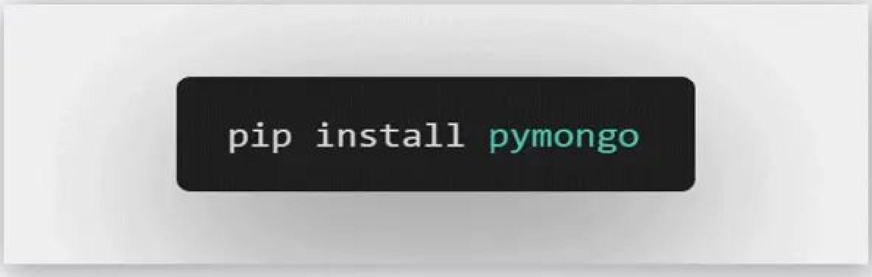


```
{
  _id: "5cf0029caff5056591b0ce7d",
  firstname: 'Jane',
  lastname: 'Wu',
  address: {
    street: '1 Circle Rd',
    city: 'Los Angeles',
    state: 'CA',
    zip: '90404'
  }
}
```

SAVE IT

How to perform CRUD operations in MongoDB using Python.

To install pymongo we need to type the below command in the command prompt.

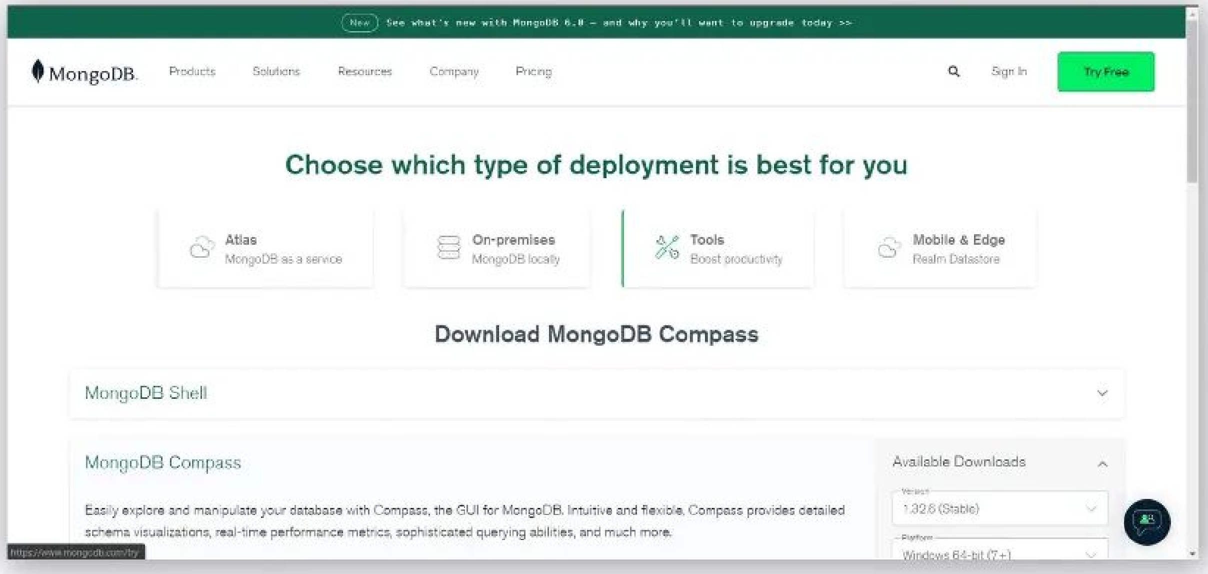


```
pip install pymongo
```

We will be using MongoDB compass, a GUI used to interact with data stored in MongoDB.

You can find the MongoDB compass installation here:

https://www.mongodb.com/docs/compass/master/install/



New
See what's new with MongoDB 6.0 – and why you'll want to upgrade today >>
MongoDB.
Products
Solutions
Resources
Company
Pricing
Sign In
Try Free

Choose which type of deployment is best for you

Atlas — MongoDB as a service
On-premises — MongoDB locally
Tools — Boost productivity
Mobile & Edge — Realm Datastore

Download MongoDB Compass

MongoDB Shell

MongoDB Compass

Easily explore and manipulate your database with Compass, the GUI for MongoDB. Intuitive and flexible, Compass provides detailed schema visualizations, real-time performance metrics, sophisticated querying abilities, and much more.

Available Downloads

Version: 1.32.6 (Stable)

Platform: Windows 64-bit (7+)

SAVE IT

Firstly, we need to import the pymongo package and To connect to the MongoDB server, we will use the MongoClient method.

```python
import pymongo

"""connect with MongoDB"""
connection_url = "mongodb://localhost:27017/"

# To connect to the MongoDB server, we will use the MongoClient method.
client = pymongo.MongoClient(connection_url)
print("client: ", client)

#output
# client:  MongoClient(host=['localhost:27017'], document_class=dict, tz_aware=False, connect=True)
```

Create Database

We can create a database using the below command.

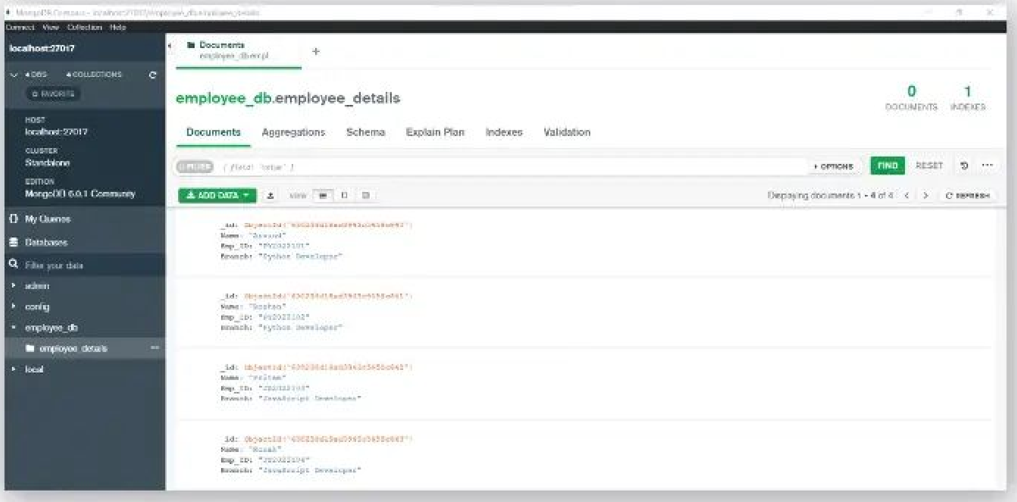
```python
# To create a database using the below command.
database_name = "employee_db"
employee_db = client[database_name]
```

SAVE IT

05

Inserting documents in the collection

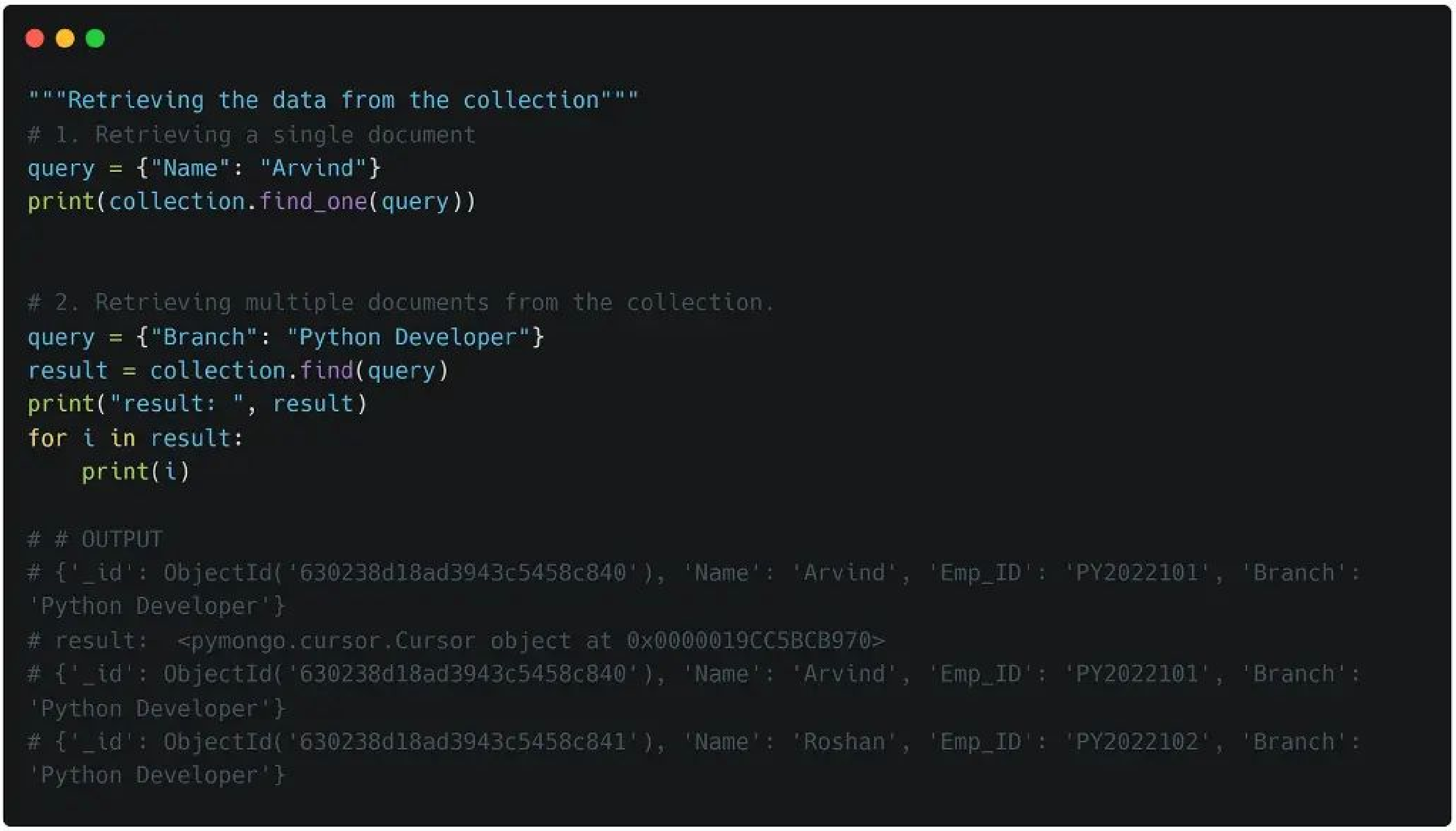
```python
# To create a database using the below command.
database_name = "employee_db"
employee_db = client[database_name]

# collection name
collection_name = "employee_details"
collection = employee_db[collection_name]

# To list the available collections in a database we can use the below command.
print("collection: ", employee_db.list_collection_names())

"""Inserting documents in the collection"""
# To insert a single document, we use insert_one() method.
document = {"Name": "Arvind", "Emp_ID": "PY2022101", "Branch": "Python Developer"}
collection.insert_one(document)


#  To insert multiple documents, we will use insert_many() meth  od.
documents = [
    {"Name": "Roshan", "Emp_ID": "PY2022102", "Branch": "Python Developer"},
    {"Name": "Pritam", "Emp_ID": "JA2022103", "Branch": "JavaScript Developer"},
    {"Name": "Ronak", "Emp_ID": "JS2022104", "Branch": "JavaScript Developer"},
]
collection.insert_many(documents)
```



```
_id: ObjectId('[illegible]')
Name: "Arvind"
Emp_ID: "PY2022101"
Branch: "Python Developer"

_id: ObjectId('[illegible]')
Name: "Roshan"
Emp_ID: "PY2022102"
Branch: "Python Developer"

_id: ObjectId('[illegible]')
Name: "Pritam"
Emp_ID: "JA2022103"
Branch: "JavaScript Developer"

_id: ObjectId('[illegible]')
Name: "Ronak"
Emp_ID: "JS2022104"
Branch: "JavaScript Developer"
```

SAVE IT

Retrieving the data from the collection

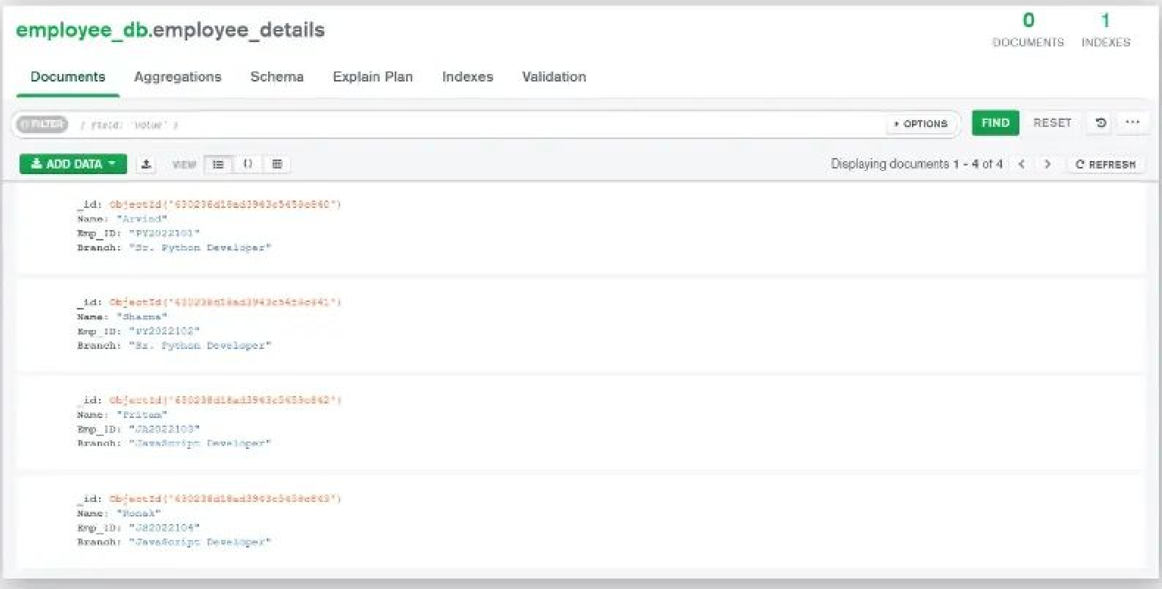


```python
"""Retrieving the data from the collection"""
# 1. Retrieving a single document
query = {"Name": "Arvind"}
print(collection.find_one(query))


# 2. Retrieving multiple documents from the collection.
query = {"Branch": "Python Developer"}
result = collection.find(query)
print("result: ", result)
for i in result:
    print(i)

# # OUTPUT
# {'_id': ObjectId('630238d18ad3943c5458c840'), 'Name': 'Arvind', 'Emp_ID': 'PY2022101', 'Branch':
'Python Developer'}
# result:  <pymongo.cursor.Cursor object at 0x0000019CC5BCB970>
# {'_id': ObjectId('630238d18ad3943c5458c840'), 'Name': 'Arvind', 'Emp_ID': 'PY2022101', 'Branch':
'Python Developer'}
# {'_id': ObjectId('630238d18ad3943c5458c841'), 'Name': 'Roshan', 'Emp_ID': 'PY2022102', 'Branch':
'Python Developer'}
```

SAVE IT

Updating the documents in the collection

```python
"""Updating the documents in the collection"""
# 1. Updating a single document
query = {"Emp_ID": {"$eq": "PY2022102"}}
present_data = collection.find_one(query)

new_data = {"$set": {"Name": "Sharma"}}
collection.update_one(present_data, new_data)

# 2. To update multiple documents we use update_many() method.
present_data = {"Branch": "Python Developer"}
new_data = {"$set": {"Branch": "Sr. Python Developer"}}
collection.update_many(present_data, new_data)
```

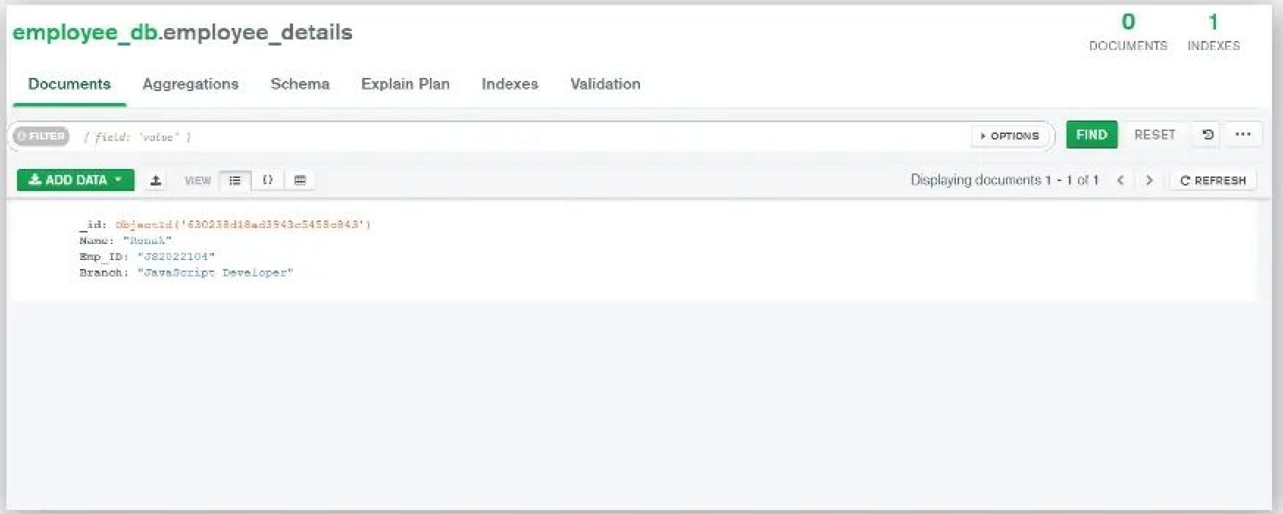


```
_id: ObjectId('630238d18ad3943c5453e840')
Name: "Arvind"
Emp_ID: "PY2022101"
Branch: "Sr. Python Developer"

_id: ObjectId('630238d18ad3943c5453e841')
Name: "Sharma"
Emp_ID: "PY2022102"
Branch: "Sr. Python Developer"

_id: ObjectId('630238d18ad3943c5453e842')
Name: "Pritam"
Emp_ID: "JA2022103"
Branch: "JavaScript Developer"

_id: ObjectId('630238d18ad3943c5453e843')
Name: "Ronak"
Emp_ID: "JS2022104"
Branch: "JavaScript Developer"
```

SAVE IT

Deleting documents from the collectio

```
"""Deleting documents from the collection"""
# 1. Deleting a single document

query = {"Emp_ID": "JA2022103"}
collection.delete_one(query)

# 2. Deleting multiple documents

query = {"Branch": "Sr. Python Developer"}
collection.delete_many(query)
```



Displaying documents 1 - 1 of 1

```
_id: ObjectId('630238d18ad3943c5458c843')
Name: "Ronak"
Emp_ID: "JS2022104"
Branch: "JavaScript Developer"
```

SAVE IT