Working with Databases in Python 3

Using a NoSQL Database - MongoDB and pymongo

Douglas Starnes
Author / Speaker

@poweredbyaltnet | linktr.ee/douglasstarnes

Overview

- Migrating from Mongita to MongoDB
- Visual Studio Code extension
- Refactor the Mongita demo to use MongoDB
- Modeling relationships with embedded documents
- New update operators

Connecting with pymongo

```
from pymongo import MongoClient
```

Connecting with pymongo

```
from pymongo import MongoClient

client = MongoClient() # connect to localhost and port 27017
```

Connecting with pymongo

```
from pymongo import MongoClient

client = MongoClient() # connect to localhost and port 27017

# same as Mongita
db = client.portfolio
investments = db.investments
```

Review of Filter Documents

```
investments.update_one({
    “coin”: “bitcoin”
}, {
    “$inc”, {“amount”: 1}
})
```

Filtering with Multiple Conditions

Mongita will not allow filter documents with multiple conditions

The $and operator will apply multiple filter documents

The $and operator accepts an array of filter documents and applied them all

In Python this is a list of dictionaries

Using the $and operator

```
{“coin”: “bitcoin”},
```

Using the $and operator

```
{“coin”: “bitcoin”},
{“amount”: {“$gt”: 2}}
```

Using the $and operator

```
{“$and”: [
    {“coin”: “bitcoin”},
    {“amount”: {“$gt”: 2}}
]}
```

Mongita violates the “Zen of Python”

“Errors should not pass silently”

Embedding JSON Objects

```
{
  “key_1”: “value_1”,
  “key_2”: {
    “embedded_key_1”: “embedded_value_1”,
    “embedded_key_2”: “embedded_value_2”,
    “embedded_key_3”: “embedded_value_3”,
  },
  “key_3”: “value_2”
}
```

A Watchlist Document

```
{
  “name”: “My Watchlist”,
  “description”: “It’s a watchlist”,
  “currency”: “USD”,
  “date_created”: “2023-04-29T14:41:05.701Z”
}
```

A Watchlist Document (with embedded metadata)

```
{
  “name”: “My Watchlist”,
  “metadata”: {
    “description”: “It’s a watchlist”,
    “currency”: “USD”,
    “date_created”: “2023-04-29T14:41:05.701Z”
  }
}
```

A pymongo Watchlist Document (with embedded metadata)

```
import datetime

metadata = {
    "description": "This is a watchlist",
    "currency": "USD",
    "date_created": datetime.datetime.now()
}
```

A pymongo Watchlist Document (with embedded metadata)

```
import datetime

metadata = {
    "description": "This is a watchlist",
    "currency": "USD",
    "date_created": datetime.datetime.now()
}

watchlist = {
    "name": "My Watchlist"
}
```

A pymongo Watchlist Document (with embedded metadata)

```
import datetime

metadata = {
    "description": "This is a watchlist",
    "currency": "USD",
    "date_created": datetime.datetime.now()
}

watchlist = {
    "name": "My Watchlist"
}

watchlist["metadata"] = metadata
```

A pymongo Watchlist Document (with embedded metadata)

```
import datetime

metadata = {
    "description": "This is a watchlist",
    "currency": "USD",
    "date_created": datetime.datetime.now()
}

watchlist = {
    "name": "My Watchlist"
}

watchlist["metadata"] = metadata

watchlists.insert_one(watchlist)
```

A pymongo Watchlist Document (with embedded metadata)

```
{
  "_id": {
    "$oid": "644d6bcb23a2367e59da32b4"
  },
  "name": "My Watchlist",
    "metadata": {
    "description": "This is a watchlist",
    "currency": "USD",
    "date_created": {
      "$date": "2023-04-29T14:11:07.535Z"
    }
  }
}
```

Filtering Embedded Documents

```
usd_watchlists = watchlists.find({
  “metadata.currency”: “USD”
})
```

Filtering Embedded Documents

```
usd_watchlists = watchlists.find({
  “metadata.currency”: “USD”
}, {
  “name”: 1, “metadata.currency”: 1
})
```

```
[
  {
    '_id': ObjectId('644d6bcb23a2367e59da32b4’),
    'name': 'My Watchlist’,
    'metadata’: {
      'currency': 'USD’
    }
  }
]
```

Embedding JSON Arrays

```
{
  “key_1”: “value_1”,
  “key_2”: [
    {“array_key_1”: “array_value_1”},
    {“array_key_2”: “array_value_2”},
    {“array_key_3”: “array_value_3”}
  ],
  “key_3”: “value_2”
}
```

Embedding MongoDB Arrays with pymongo

```
bitcoin = {
    “coin”: “bitcoin”,
    “note”: “Bitcoin is number one!”,
    “date_added”: datetime.datetime.now()
}
```

Embedding MongoDB Arrays with pymongo

```
bitcoin = {
    “coin”: “bitcoin”,
    “note”: “Bitcoin is number one!”,
    “date_added”: datetime.datetime.now()
}

watchlist = {
    “name”: “My Watchlist”,
    “coins”: [bitcoin]
}
```

Embedding MongoDB Arrays with pymongo

```
bitcoin = {
    “coin”: “bitcoin”,
    “note”: “Bitcoin is number one!”,
    “date_added”: datetime.datetime.now()
}

watchlist = {
    “name”: “My Watchlist”,
    “coins”: [bitcoin]
}

watchlist["coins"].append({
    "coin": "ethereum",
    "note": "Ethereum is number two!",
    "date_added": datetime.datetime.now()
})
```

Embedding MongoDB Arrays with pymongo

```
bitcoin = {
    “coin”: “bitcoin”,
    “note”: “Bitcoin is number one!”,
    “date_added”: datetime.datetime.now()
}

watchlist = {
    “name”: “My Watchlist”,
    “coins”: [bitcoin]
}

watchlist["coins"].append({
    "coin": "ethereum",
    "note": "Ethereum is number two!",
    "date_added": datetime.datetime.now()
})

watchlists.insert_one(watchlist)
```

Embedding MongoDB Arrays with pymongo


```
{
  "_id": {
    "$oid": "644d6bcb23a2367e59da32b4"
  },
  "name": "My Watchlist",
  "coins": [
    {
      "coin": "bitcoin",
      "note": "Bitcoin is number one!",
      "date_added": {
        "$date": "2023-04-29T14:11:07.535Z"
      }
    },
    {
      "coin": "ethereum",
      "note": "Ethereum is number two!",
      "date_added": {
        "$date": "2023-04-29T14:11:07.535Z"
      }
    }
  ],
  "metadata": {
    "description": "This is a watchlist",
    "currency": "USD",
    "date_created": {
      "$date": "2023-04-29T14:11:07.535Z"
    }
  }
}
```


Filtering Embedded Arrays

```
bitcoin_watchlist = find_one({“coins”: {“coin”: “bitcoin”}}) # this won’t work!
```

Filtering Embedded Arrays

```
bitcoin_watchlist = find_one({“coins”: {“coin”: “bitcoin”}}) # this won’t work!

bitcoin_watchlist = find_one({“coins.coin”: “bitcoin”})
```

Filtering Embedded Arrays

```
bitcoin_watchlist = find_one({“coins”: {“coin”: “bitcoin”}}) # this won’t work!

bitcoin_watchlist = find_one({“coins.coin”: “bitcoin”})

bitcoin_watchlist = find_one({“coins.coin”: “bitcoin”}, {“name”: 1})
```

Filtering Embedded Arrays

```
bitcoin_watchlist = find_one({“coins”: {“coin”: “bitcoin”}}) # this won’t work!

bitcoin_watchlist = find_one({“coins.coin”: “bitcoin”})

bitcoin_watchlist = find_one({“coins.coin”: “bitcoin”}, {“name”: 1})

bitcoin_usd_watchlist = find_one({
    “$and”: [
        {“metadata.currency”: “USD”},
        {“coins.coin”: “bitcoin”}
    ]
}, {“name”: 1})
```

The $push update operator

```
import datetime

from bson import ObjectId

watchlists.update_one(
    {"_id", ObjectId("644d6bcb23a2367e59da32b4")},
    {
        "$push":
        {
            "coins": {
                "coin": "dogecoin",
                "note": "It's a joke!",
                "date_added": datetime.datetime.now()
            }
        }
    }
)
```

The $pull update operator

```
from bson import ObjectId

watchlists.update_one(
    {"_id", ObjectId("644d6bcb23a2367e59da32b4")},
    {“$pull”, {“coins”: {“coin”: “dogecoin”}}}
)
```

Array Operators in Mongita

The $push operator is supported

The $pull operator is not supported

Unlike $and, which fails silently, the $pull operator will raise an error

Summary

The pymongo package is the official Python package for MongoDB

Mongita to pymongo migration

The $and filter operator

Embedded documents and arrays

The $push and $pull update operators

Visual Studio Code tooling

Stay tuned for MongoEngine!