



# Python MongoDB Insert Document

[< Previous](#)[Next >](#)

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

## Insert Into Collection

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

The first parameter of the `insert_one()` method is a dictionary containing the name(s) and value(s) of each field in the document you want to insert.

## Example

Insert a record in the "customers" collection:

```
import pymongo

myclient = pymongo.MongoClient("mongodb://localhost:27017/")
mydb = myclient["mydatabase"]
mycol = mydb["customers"]

mydict = { "name": "John", "address": "Highway 37" }

x = mycol.insert_one(mydict)
```

[Run example »](#)

# Return the `_id` Field

The `insert_one()` method returns a `InsertOneResult` object, which has a property, `inserted_id`, that holds the id of the inserted document.

## Example

Insert another record in the "customers" collection, and return the value of the `_id` field:

```
mydict = { "name": "Peter", "address": "Lowstreet 27" }

x = mycol.insert_one(mydict)

print(x.inserted_id)
```

[Run example »](#)

If you do not specify an `_id` field, then MongoDB will add one for you and assign a unique id for each document.

In the example above no `_id` field was specified, so MongoDB assigned a unique `_id` for the record (document).

# Insert Multiple Documents

To insert multiple documents into a collection in MongoDB, we use the `insert_many()` method.

The first parameter of the `insert_many()` method is a list containing dictionaries with the data you want to insert:

## Example

```
import pymongo

myclient = pymongo.MongoClient("mongodb://localhost:27017/")
mydb = myclient["mydatabase"]
```

```
mycol = mydb["customers"]

mylist = [
    { "name": "Amy", "address": "Apple st 652"},
    { "name": "Hannah", "address": "Mountain 21"},
    { "name": "Michael", "address": "Valley 345"},
    { "name": "Sandy", "address": "Ocean blvd 2"},
    { "name": "Betty", "address": "Green Grass 1"},
    { "name": "Richard", "address": "Sky st 331"},
    { "name": "Susan", "address": "One way 98"},
    { "name": "Vicky", "address": "Yellow Garden 2"},
    { "name": "Ben", "address": "Park Lane 38"},
    { "name": "William", "address": "Central st 954"},
    { "name": "Chuck", "address": "Main Road 989"},
    { "name": "Viola", "address": "Sideway 1633"}
]

x = mycol.insert_many(mylist)

#print list of the _id values of the inserted documents:
print(x.inserted_ids)
```

Run example »

The `insert_many()` method returns a `InsertManyResult` object, which has a property, `inserted_ids`, that holds the ids of the inserted documents.

## Insert Multiple Documents, with Specified IDs

If you do not want MongoDB to assign unique ids for you document, you can specify the `_id` field when you insert the document(s).

Remember that the values has to be unique. Two documents cannot have the same `_id`.

### Example

```
import pymongo

myclient = pymongo.MongoClient("mongodb://localhost:27017/")
mydb = myclient["mydatabase"]
mycol = mydb["customers"]
```

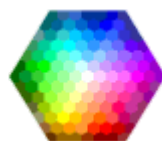
```
mylist = [  
  { "_id": 1, "name": "John", "address": "Highway 37"},  
  { "_id": 2, "name": "Peter", "address": "Lowstreet 27"},  
  { "_id": 3, "name": "Amy", "address": "Apple st 652"},  
  { "_id": 4, "name": "Hannah", "address": "Mountain 21"},  
  { "_id": 5, "name": "Michael", "address": "Valley 345"},  
  { "_id": 6, "name": "Sandy", "address": "Ocean blvd 2"},  
  { "_id": 7, "name": "Betty", "address": "Green Grass 1"},  
  { "_id": 8, "name": "Richard", "address": "Sky st 331"},  
  { "_id": 9, "name": "Susan", "address": "One way 98"},  
  { "_id": 10, "name": "Vicky", "address": "Yellow Garden 2"},  
  { "_id": 11, "name": "Ben", "address": "Park Lane 38"},  
  { "_id": 12, "name": "William", "address": "Central st 954"},  
  { "_id": 13, "name": "Chuck", "address": "Main Road 989"},  
  { "_id": 14, "name": "Viola", "address": "Sideway 1633"}  
]  
  
x = mycol.insert_many(mylist)  
  
#print list of the _id values of the inserted documents:  
print(x.inserted_ids)
```

[Run example »](#)

[< Previous](#)

[Next >](#)

COLOR PICKER



HOW TO

Tabs

Dropdowns

Accordions

Side Navigation

Top Navigation

Modal Boxes

Progress Bars

Parallax

Login Form

HTML Includes

Google Maps

Range Sliders

Tooltips

Slideshow

Filter List

Sort List

SHARE



CERTIFICATES

HTML

CSS

JavaScript

SQL

Python

PHP

jQuery

Bootstrap

XML

[Read More »](#)

---

REPORT ERROR

PRINT PAGE

[FORUM](#)[ABOUT](#)

## Top Tutorials

[HTML Tutorial](#)  
[CSS Tutorial](#)  
[JavaScript Tutorial](#)  
[How To Tutorial](#)  
[SQL Tutorial](#)  
[Python Tutorial](#)  
[W3.CSS Tutorial](#)  
[Bootstrap Tutorial](#)  
[PHP Tutorial](#)  
[jQuery Tutorial](#)  
[Java Tutorial](#)  
[C++ Tutorial](#)

## Top References

[HTML Reference](#)  
[CSS Reference](#)  
[JavaScript Reference](#)  
[SQL Reference](#)  
[Python Reference](#)  
[W3.CSS Reference](#)  
[Bootstrap Reference](#)  
[PHP Reference](#)  
[HTML Colors](#)  
[jQuery Reference](#)  
[Java Reference](#)  
[Angular Reference](#)

## Top Examples

[HTML Examples](#)  
[CSS Examples](#)  
[JavaScript Examples](#)  
[How To Examples](#)  
[SQL Examples](#)  
[Python Examples](#)  
[W3.CSS Examples](#)  
[Bootstrap Examples](#)  
[PHP Examples](#)  
[jQuery Examples](#)  
[Java Examples](#)  
[XML Examples](#)

## Web Certificates

[HTML Certificate](#)  
[CSS Certificate](#)  
[JavaScript Certificate](#)  
[SQL Certificate](#)

Python Certificate  
jQuery Certificate  
PHP Certificate  
Bootstrap Certificate  
XML Certificate

[Get Certified »](#)

---

W3Schools is optimized for learning, testing, and training. Examples might be simplified to improve reading and basic understanding. Tutorials, references, and examples are constantly reviewed to avoid errors, but we cannot warrant full correctness of all content. While using this site, you agree to have read and accepted our terms of use, cookie and privacy policy. Copyright 1999-2020 by Refsnes Data. All Rights Reserved.

Powered by W3.CSS.

