



# Pymongo

Commands



In any database there are four fundamental operations which represents **CRUD**.

**C** → Create

**R** → Read

**U** → Update

**D** → Delete



# Create

MongoDB Atlas, PyMongo

S.No	Operation	MongoDB Shell (mongosh)	PyMongo (Python)
1	Show all databases	<code>show dbs</code>	<code>client.list_database_names()</code>
2	Use / switch database	<code>use school</code>	<code>db = client["school"]</code>
3	Show current database	<code>db</code>	<code>db.name</code>
4	Show collections	<code>show collections</code>	<code>db.list_collection_names()</code>
5	Create collection	<code>db.students.insertOne({})</code>	<code>db.students.insert_one({})</code>
6	Insert one document	<code>db.students.insertOne({name:"Amit"})</code>	<code>db.students.insert_one({"name":"Amit"})</code>
7	Insert many documents	<code>db.students.insertMany([     {name:"A"},     {name:"B"} ])</code>	<code>db.students.insert_many([     {name:"A"},     {name:"B"} ])</code>



# Read

MongoDB Atlas, PyMongo

```
db.collection.find(
  <filter>,
  <projection>
)
```

S.No	Operation	MongoDB Shell (mongosh)	PyMongo (Python)
1	Show all databases	<code>show dbs</code>	<code>client.list_database_names()</code>
2	Show collections	<code>show collections</code>	<code>db.list_collection_names()</code>
3	Find all documents	<code>db.students.find()</code>	<code>db.students.find()</code>
4	Find one document	<code>db.students.findOne({name:"Amit"})</code>	<code>db.students.find_one({"name":"Amit"})</code>
5	Filter documents	<code>db.students.find({age: 20})</code>	<code>db.students.find({"age":20})</code>
6	Count documents	<code>db.students.countDocuments({})</code>	<code>db.students.count_documents({})</code>
7	Sort documents	<code>db.students.find().sort({age:1})</code> 1 → ascending order, -1 → descending order	<code>db.students.find().sort({age:1})</code> 1 → ascending order, -1 → descending order
8	Limit documents	<code>db.students.find().limit(5)</code>	<code>db.students.find().limit(5)</code>
9	Projection Select on specific fields	<code>db.students.find({}, {name: 1, age: 1})</code>	<code>db.students.find({}, {name: 1, age: 1})</code>



Read

Comparative Operators

```
db.collection.find(  
  {  
    field: { $operator : value}  
  }  
)
```



Operator	Meaning	Example Query
\$eq	Equal to	<code>db.emp.find({ age: { \$eq: 35 } })</code>
\$ne	Not equal to	<code>db.emp.find({ salary: { \$ne: 75000 } })</code>
\$gt	Greater than	<code>db.emp.find({ salary: { \$gt: 40000 } })</code>
\$gte	Greater than or equal to	<code>db.emp.find({ age: { \$gte: 30 } })</code>
\$lt	Less than	<code>db.emp.find({ age: { \$lt: 30 } })</code>
\$lte	Less than or equal to	<code>db.emp.find({ salary: { \$lte: 50000 } })</code>
\$in	In	<code>db.emp.find({ name: { \$in: ["Ram", "Sita"] } })</code>
\$nin	Not in	<code>db.emp.find({ name: { \$nin: ["Ram", "Sita"] } })</code>



Read

Logical Operators

```
db.collection.find(  
  {  
    $operator:[  
      condition1,  
      condition2,  
      ...  
    ]  
  }  
)
```

Operator	Meaning	Description	Example Query
<b>\$and</b>	Logical AND	Matches documents where <b>all conditions</b> are true	<pre>db.emp.find({ \$and: [   { age: { \$gt: 30 } },   { salary: { \$gte: 50000 } } ]})</pre>
<b>\$or</b>	Logical OR	Matches documents where <b>at least one condition</b> is true	<pre>db.emp.find({ \$or: [   { age: { \$lt: 25 } },   { salary: { \$gt: 80000 } } ]})</pre>
<b>\$nor</b>	Logical NOR	Matches documents where <b>none of the conditions</b> are true	<pre>db.emp.find({ \$nor: [   { age: { \$lt: 25 } },   { salary: { \$gt: 80000 } } ]})</pre>
<b>\$not</b>	Logical NOT	Matches documents where the condition is <b>not true</b>	<pre>db.emp.find({ age: { \$not: { \$gte: 30 } } })</pre>



# Update commands

Modify existing documents in a collection

# Update Commands in MongoDB

Update operations are used to **modify existing documents** in collection.

MongoDB mainly provides three update methods:

```
db.collection.update_one()
```

```
db.collection.update_many()
```

```
db.collection.replace_one()
```

```
db.collection.update_one(  
    <filter>,  
    <update>,  
    <options>...  
)
```

Operator	Purpose	Example Query
<b>\$set</b>	Set or update a field	<pre>db.emp.update_one(   { name: { \$eq: "Amit" } },   { \$set: { age: 21 } } )</pre>
<b>\$unset</b>	Remove a field	<pre>db.emp.update_one(   { name: "Amit"},   { \$unset: {phone: ""} } )</pre>
<b>\$inc</b>	Increment a numeric value	<pre>db.emp.update_one(   { name: "Amit"},   { \$inc: {age: 1} } )</pre>
<b>\$push</b>	Add value to array	<pre>db.enm.update_one(   { name: "Amit" },   { \$push: { skills: "Python" } } )</pre>
<b>\$pull</b>	Remove value from array	<pre>db.enm.update_one(   { name: "Amit" },   { \$pull: { skills: "Python" } } )</pre>



<code>\$set</code>	→ add / update field
<code>\$unset</code>	→ remove field
<code>\$inc</code>	→ increment number
<code>\$push</code>	→ add to array
<code>\$pull</code>	→ remove from array



# update\_many, replace\_one

Modify existing documents in a collection

# Update Commands in MongoDB

Update operations are used to **modify existing documents** in collection.

MongoDB mainly provides three update methods:

```
db.collection.update_one()
```

```
db.collection.update_many()
```

```
db.collection.replace_one()
```



# Delete Commands

Remove documents or entire collection/databases from MongoDB

# Delete Commands in MongoDB

Delete operations are used to **remove documents or entire collections/databases** in collection.

MongoDB mainly provides two main methods:

```
db.collection.delete_one()
```

```
db.collection.delete_many()
```

```
db.collection.delete_one(  
    <filter>,  
    <options>...  
)
```

S.No	Operation	PyMongo
1	Delete one document	<code>db.emp.delete_one({ "name": "Rohit" })</code>
2	Delete many documents	<code>db.emp.delete_many({ "currency": "inr" })</code>
3	Delete by condition	<code>db.emp.delete_many({ "salary": { "\$lt": 30000 } })</code>
4	Delete all documents	<code>db.emp.delete_many({})</code>
5	Delete using _id	<code>db.emp.delete_one({ "_id": ObjectId("...") })</code>
6	Drop collection	<code>db.emp.drop()</code>
7	Drop database	<code>client.drop_database("employee")</code>

vedicskill.