

Where, AND , OR & CRUD

Where:

Given a Collection you want to FILTER a subset based on a condition.

That is the place WHERE is used

```
test> db.stu.find({gpa:{$gt:3.5}}).count();  
124
```

```
test> db.stu.find({home_city:"City 3"}).count();  
34
```

AND:

Given a Collection you want to FILTER a subset based on multiple conditions.

```
test> db.stu.find({  
... $and:[  
...   {home_city:"City 5"},  
...   {blood_group:"A+"}  
... ]  
... });  
[  
  {  
    _id: ObjectId('6655e91deedc73e7398db'),  
    name: 'Student 142',  
    age: 24,  
    courses: "['History', 'English', 'Physics', 'Computer Science']",  
    gpa: 3.41,  
    home_city: 'City 5',  
    blood_group: 'A+',  
    is_hotel_resident: false  
  },  
  {  
    _id: ObjectId('6655e91eeedc73e7399fb'),  
    name: 'Student 947',  
    age: 20,  
    courses: "['Physics', 'History', 'English', 'Computer Science']",  
    gpa: 2.86,  
    home_city: 'City 5',  
    blood_group: 'A+',  
    is_hotel_resident: true  
  },  
  {  
    _id: ObjectId('6655e91eeedc73e739a6d'),  
    name: 'Student 567',
```

```

    gpa: 2.86,
    home_city: 'City 5',
    blood_group: 'A+',
    is_hotel_resident: true
  },
  {
    _id: ObjectId('6655e91eee1dcfb73e739a6d'),
    name: 'Student 567',
    age: 22,
    courses: "['Computer Science', 'History', 'English', 'Mathematics']",
    gpa: 2.01,
    home_city: 'City 5',
    blood_group: 'A+',
    is_hotel_resident: true
  }
]

```

OR:

Given a Collection you want to FILTER a subset based on multiple conditions but Any One is Sufficient

```

test> db.stu.find({ $or: [ { is_hostel_resident: true }, { gpa: { $lt: 3.0 } } ] }).count();
261
test> |

```

CRUD:

1. C - Create/Insert
2. R - Remove
3. U - update
4. D – Delete

CREATE/INSERT:

The insert function in MongoDB is used to add documents to a collection.

In the context of MongoDB, a document is a set of key-value pairs (similar to a JSON object), and a collection is a group of documents.

```
test> const studentData = {  
...   "name": "Alice Smith",  
...   "age": 22,  
...   "courses": ["Mathematics", "Computer Science", "English"],  
...   "gpa": 3.8,  
...   "home_city": "New York",  
...   "blood_group": "A+",  
...   "is_hotel_resident": false  
... };  
  
test> db.stu.insertOne(studentData);  
{  
  acknowledged: true,  
  insertedId: ObjectId('665b529e49389824aecdcdf7')  
}  
test> |
```

UPDATE:

```
test> db.stu.updateOne({name: "Alice Smith"}, {$set: {gpa: 3.8}});  
{  
  acknowledged: true,  
  insertedId: null,  
  matchedCount: 1,  
  modifiedCount: 0,  
  upsertedCount: 0  
}  
test> |
```

UPDATE MANY:

The update Many function in MongoDB is used to update multiple documents in a collection that match a given filter. This function is particularly useful when you need to modify several documents at once based on a specific condition.

```
test> db.stu.updateMany({gpa:{$lt:3.0}},{$inc:{gpa:0.5}});
{
  acknowledged: true,
  insertedId: null,
  matchedCount: 261,
  modifiedCount: 261,
  upsertedCount: 0
}
```

DELETE MANY:

```
test> db.stu.deleteMany({is_hostel_resident:false});
{ acknowledged: true, deletedCount: 1 }
test> |
```

PROJECTION:

This is used when we don't need all columns/attributes

```
test> db.stu.find({}, {name:1, gpa:1});
```