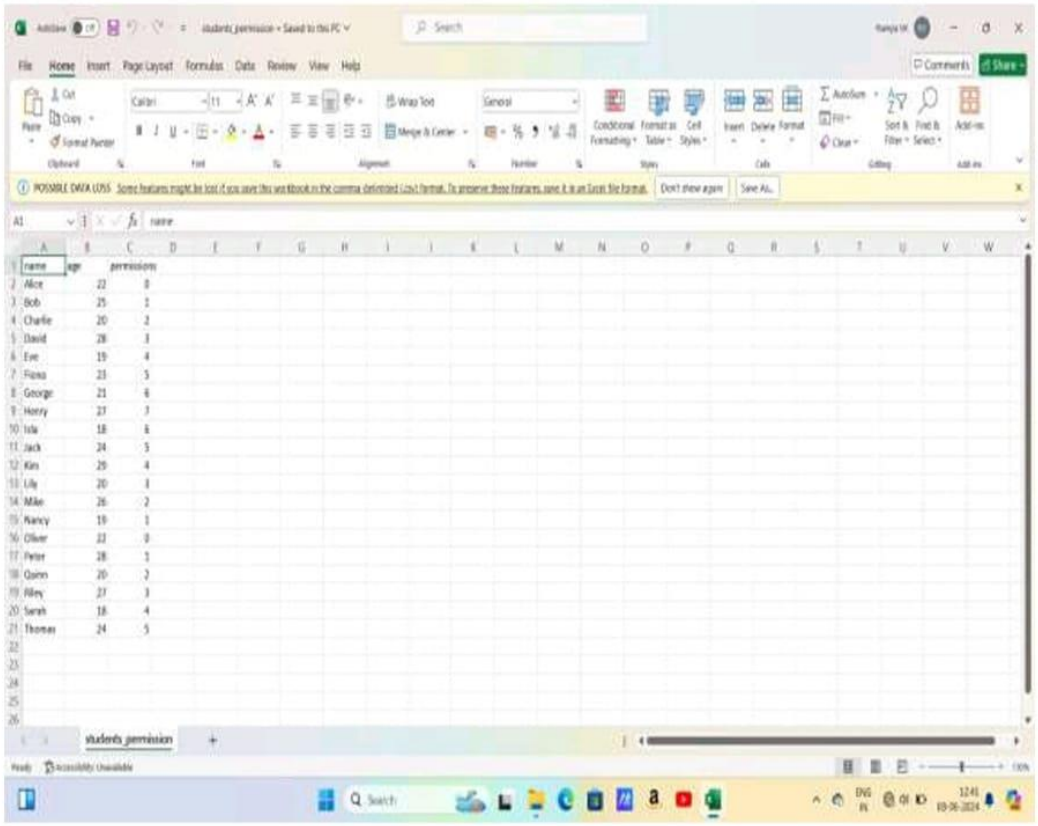
ADD,UPDATE &DELETE

FEW COMMANDS TO TEST:

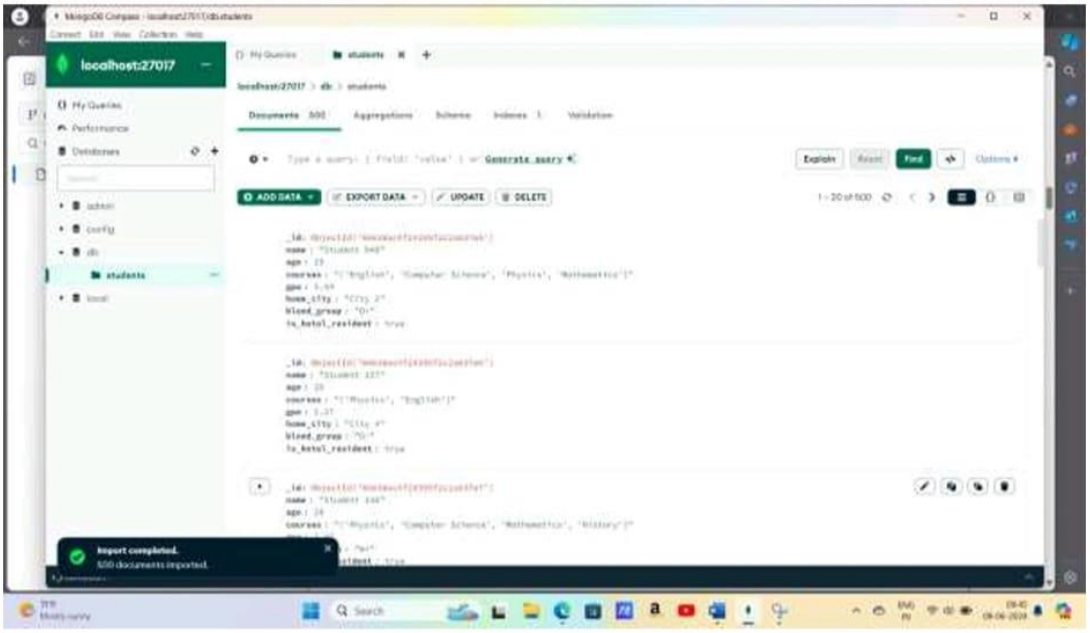
Command	Expected Output	Notes
show dbs	admin 40.00 KiB config 72.00 KiB db 128.00 KiB local 40.00 KiB	All Databases are shown
use db	switched to db db	Connect and use db
show collections	Students	Show all tables
db.foo.insert({"bar" : "baz"})		Insert a record to collection. Create Collection if not exists

DOCUMENTS:

Documents are stored in collections, which are analogous to tables in a relational database.



name	age	permissions
Alice	22	0
Bob	25	1
Charlie	20	2
David	28	3
Eve	19	4
Fiona	23	5
George	21	6
Henry	27	7
Isla	18	6
Jack	24	5
Kim	29	4
Lily	20	3
Mike	26	2
Nancy	19	1
Oliver	22	0
Peter	28	1
Quinn	20	2
Riley	27	3
Sarah	18	4
Thomas	24	5



COLLECTIONS:

• Collections A collection is a group of documents.

```
db> db.students.find({});
[
  {
    _id: ObjectId('6663dac4f24355f2c2a837e5'),
    name: 'Student 948',
    age: 19,
    courses: "['English', 'Computer Science', 'Physics', 'Mathematics']",
    gpa: 3.44,
    home_city: 'City 2',
    blood_group: 'O+',
    is_hotel_resident: true
  },
  {
    _id: ObjectId('6663dac4f24355f2c2a837e6'),
    name: 'Student 157',
    age: 20,
    courses: "['Physics', 'English']",
    gpa: 2.27,
    home_city: 'City 4',
    blood_group: 'O-',
    is_hotel_resident: true
  },
  {
    _id: ObjectId('6663dac4f24355f2c2a837e7'),
    name: 'Student 316',
    age: 20,
    courses: "['Physics', 'Computer Science', 'Mathematics', 'History']",
    gpa: 2.32,
    blood_group: 'B+',
    is_hotel_resident: true
  },
```

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.

Ex: db.students.find({ gpa: {$lt: 2.5}});

```
db> db.students.find({gpa:{$lt:4.5}});
[
  {
    _id: ObjectId('6645f14f8419dc976a376b76'),
    name: 'Student 948',
    age: 19,
    courses: "['English', 'Computer Science', 'Physics', 'Mathematics']",
    gpa: 3.44,
    home_city: 'City 2',
    blood_group: 'O+',
    is_hotel_resident: true
  },
  {
    _id: ObjectId('6645f14f8419dc976a376b77'),
    name: 'Student 157',
    age: 20,
    courses: "['Physics', 'English']",
    gpa: 2.77,
    home_city: 'City 4',
    blood_group: 'O-',
    is_hotel_resident: true
  },
  {
    _id: ObjectId('6645f14f8419dc976a376b78'),
    name: 'Student 316',
    age: 20,
    courses: "['Physics', 'Computer Science', 'Mathematics', 'History']",
    gpa: 2.82,
    blood_group: 'B+',
```

OR :

The $or operator is used to specify a compound query with multiple conditions, where at least one condition must be satisfied for a document to match.
db.students.find({$or:[{home_city:"City 4"},{gpa:{$gt:3.0}}]});

```
db> db.students.find({$or:[{home_city:"City 4"},{gpa:{$gt:3.0}}]});
[
  {
    _id: ObjectId('6645f14f8419dc976a376b76'),
    name: 'Student 948',
    age: 19,
    courses: "['English', 'Computer Science', 'Physics', 'Mathematics']",
    gpa: 3.44,
    home_city: 'City 2',
    blood_group: 'O+',
    is_hotel_resident: true
  },
  {
    _id: ObjectId('6645f14f8419dc976a376b77'),
    name: 'Student 157',
    age: 20,
    courses: "['Physics', 'English']",
    gpa: 2.77,
    home_city: 'City 4',
    blood_group: 'O-',
    is_hotel_resident: true
  },
  {
    _id: ObjectId('6645f14f8419dc976a376b79'),
    name: 'Student 346',
    age: 25,
    courses: "['Mathematics', 'History', 'English']",
    gpa: 3.31,
    home_city: 'City 8',
    blood_group: 'O-',
```

AND:

The $and operator allows you to specify multiple conditions that documents must satisfy to match the query.

db.students.find({ $and: [{ is_hotel_resident:true} , { home_city:"City 5"}]});

```
db> db.students.find({$and:[{is_hotel_resident:true},{home_city:"City 5"}]});
[
  {
    _id: ObjectId('6645f14f8419dc976a376bb4'),
    name: 'Student 219',
    age: 18,
    courses: "['Physics', 'Computer Science', 'Mathematics', 'History']",
    gpa: 3.4,
    home_city: 'City 5',
    blood_group: 'B-',
    is_hotel_resident: true
  },
  {
    _id: ObjectId('6645f14f8419dc976a376bb9'),
    name: 'Student 632',
    age: 23,
    courses: "['Physics', 'English', 'Computer Science']",
    gpa: 3.76,
    home_city: 'City 5',
    blood_group: 'AB-',
    is_hotel_resident: true
  },
  {
    _id: ObjectId('6645f14f8419dc976a376bcf'),
    name: 'Student 989',
    age: 24,
    courses: "['Computer Science', 'Mathematics']",
    gpa: 3.32,
```

CRUD :

- C - Create / Insert
- R – Remove
- U – update
- 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 ,and a collection is a group of documents.

Const studentsData={

“name”:“Alice Smith”,

“age”:12,,

“coueses”:[“Mathematics”,“computer science”,“ English”],

“gpa”:3.5

“home_city”:“New York”,

“blood_group”:“A+”,

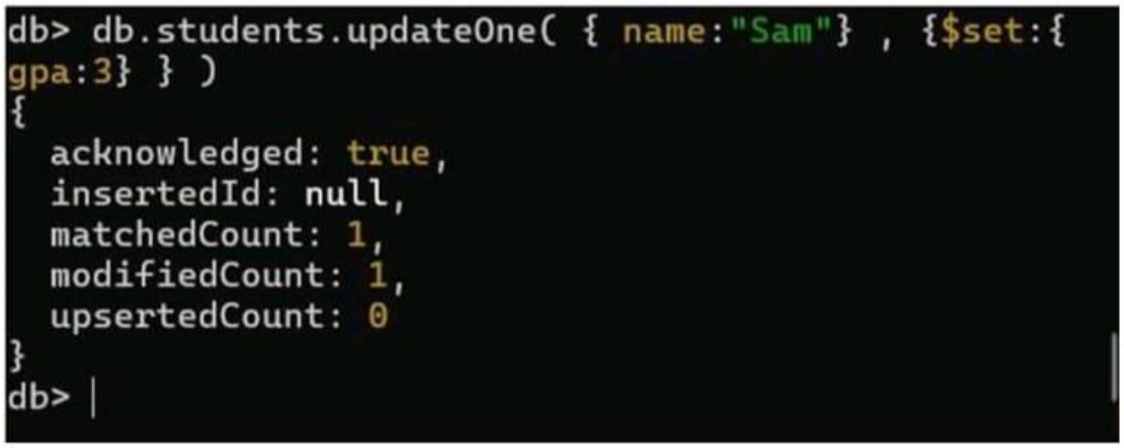
“is_hotel_resident”:false };

```
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>
```

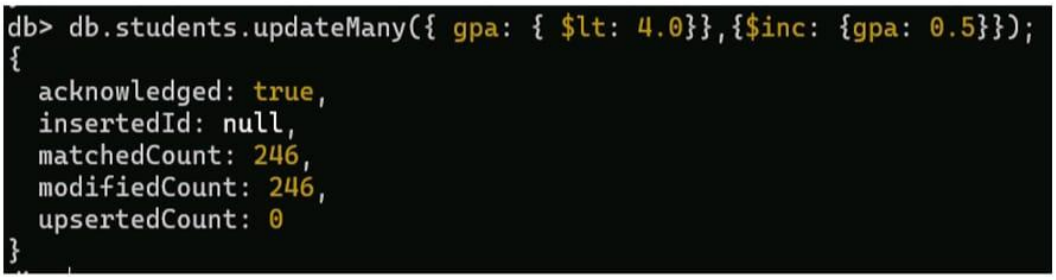
UPDATE:

db.students.updateOne({ name: “Sam”} , {$set: {gpa:3.5} });



UpdateMany():

db.students.updateMany({ gpa: { $lt: 4.0}},{$inc: {gpa: 0.5}});



DeleteMany():

db.students.deleteMany({ is_hotel_resident: true});

PROJECTION:

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

```
db> db.students.deleteOne({ name:"Sam" })
{ acknowledged: true, deletedCount: 1 }
db> db.students.find({} , {name:1 , gpa:1 })
[
  {
    _id: ObjectId('66587b4a0a3749dfd07d78a0'),
    name: 'Student 948',
    gpa: 3.44
  },
  {
    _id: ObjectId('66587b4a0a3749dfd07d78a1'),
    name: 'Student 157',
    gpa: 2.27
  },
  {
    _id: ObjectId('66587b4a0a3749dfd07d78a2'),
    name: 'Student 316',
    gpa: 2.32
```