SHEETHAL A
sheethala.ai2021@dce.edu.in

MONGO DB CODING CHALLENGE

Document Relationships

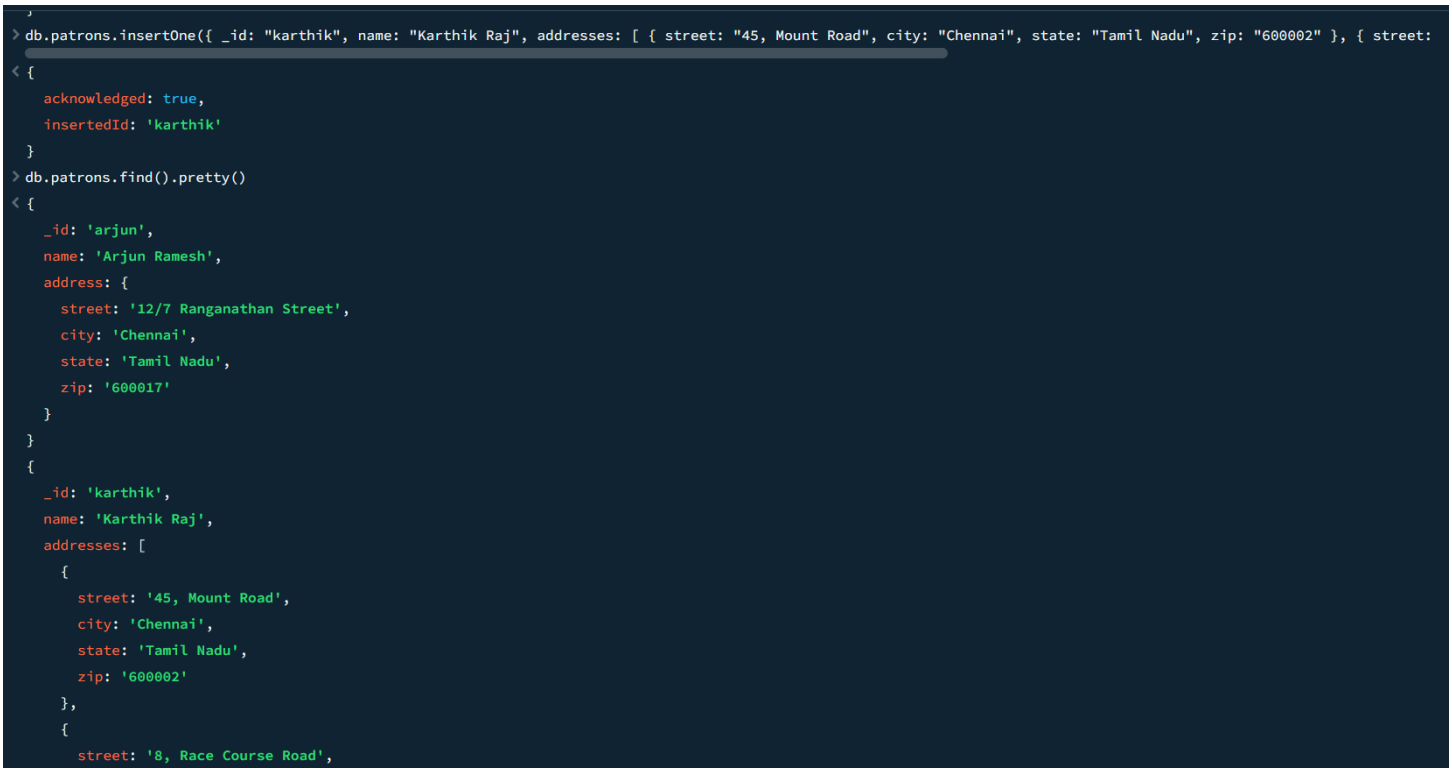
1. Model One-to-One Relationships with Embedded Documents

```
> use libraryDB
< switched to db libraryDB
> db.patrons.insertOne({ _id: "arjun", name: "Arjun Ramesh", address: {street: "12/7 Ranganathan Street", city: "Chennai", state: "Tamil Nadu", zip: "600017" } });
< {
    acknowledged: true,
    insertedId: 'arjun'
  }
> db.patrons.find().pretty()
< {
    _id: 'arjun',
    name: 'Arjun Ramesh',
    address: {
      street: '12/7 Ranganathan Street',
      city: 'Chennai',
      state: 'Tamil Nadu',
      zip: '600017'
    }
  }
```

2. Model One-to-Many Relationships with Embedded Documents

To insert One-to-Many Embedded Document the code I used:

db.patrons.insertOne({ _id: "karthik", name: "Karthik Raj", addresses: [{ street: "45, Mount Road", city: "Chennai", state: "Tamil Nadu", zip: "600002" }, { street: "8, Race Course Road", city: "Coimbatore", state: "Tamil Nadu", zip: "641018" }] });



One-to-Many relationship using references that is using a school → students example. One school can have many students.

```
db.schools.insertOne({ _id: "school001", name: "Sunrise Public School", location: "Chennai" });
{
  acknowledged: true,
  insertedId: 'school001'
}
db.students.insertOne({ _id: 1, name: "Aara", age: 12, class: "6", school_id: "school001" });
db.students.insertOne({ _id: 2, name: "Mera", age: 13, class: "7", school_id: "school001" });
{
  acknowledged: true,
  insertedId: 2
}
```

```
> db.schools.find().pretty()
< {
    _id: 'school001',
    name: 'Sunrise Public School',
    location: 'Chennai'
  }
> db.students.find().pretty()
< {
    _id: 1,
    name: 'Aara',
    age: 12,
    class: '6',
    school_id: 'school001'
  }
  {
    _id: 2,
    name: 'Mera',
    age: 13,
    class: '7',
    school_id: 'school001'
  }
```

3. Model Many-to-One Relationships with Document References

```
> db.companies.insertOne({ _id: "hex001", name: "HEXAWARE", location: "Chennai" });
< {
    acknowledged: true,
    insertedId: 'hex001'
  }
> db.employees.insertOne({ _id: 1, name: "Priya", role: "Developer", company_id: "hex001" });
  db.employees.insertOne({ _id: 2, name: "Rahul", role: "Tester", company_id: "hex001" });
  db.employees.insertOne({ _id: 3, name: "Sneha", role: "Manager", company_id: "hex001" });
< {
    acknowledged: true,
    insertedId: 3
  }
```

```
> db.employees.find().pretty()
< {
    _id: 1,
    name: 'Priya',
    role: 'Developer',
    company_id: 'hex001'
  }
  {
    _id: 2,
    name: 'Rahul',
    role: 'Tester',
    company_id: 'hex001'
  }
  {
    _id: 3,
    name: 'Sneha',
    role: 'Manager',
    company_id: 'hex001'
  }
> db.companies.find().pretty()
< {
    _id: 'hex001',
    name: 'HEXAWARE',
    location: 'Chennai'
  }
```

4. Model Many-to-Many Relationships with Embedded Documents

```
> db.students.insertOne({ _id: "stu1", name: "Aana", classes: [ { class_id: "c1", name: "Math" }, { class_id: "c12", name: "Sci" } ] });
  db.students.insertOne({ _id: "stu2", name: "Diya", classes: [ { class_id: "c2", name: "Sci" }, { class_id: "c13", name: "Eng" } ] });
< {
    acknowledged: true,
    insertedId: 'stu2'
  }
```

```
{
  _id: 'stu1',
  name: 'Aana',
  classes: [
    {
      class_id: 'c1',
      name: 'Math'
    },
    {
      class_id: 'c12',
      name: 'Sci'
    }
  ]
}
{
  _id: 'stu2',
  name: 'Diya',
  classes: [
    {
      class_id: 'c2',
      name: 'Sci'
    },
    {
      class_id: 'c13',
      name: 'Eng'
    }
  ]
}
```