DAY-5: MongoDB ASSIGNMENT

Creating the database

test> use InsuranceDB

switched to db InsuranceDB

1.Inserting the data into Agents Collection:

InsuranceDB> db.agents.insertMany([

db.agents.insertMany([

```
  { _id: 1, agentname: "Ranjan Kumar", phone: "9876500001", city: "Nagpur" },
  { _id: 2, agentname: "Greeshmanth", phone: "9876500002", city: "Jaipur" },
  { _id: 3, agentname: "Siddartha Reddy", phone: "9876500003", city: "Delhi" },
  { _id: 4, agentname: "Pranouti", phone: "9876500004", city: "Mumbai" },
  { _id: 5, agentname: "Sarath Sai", phone: "9876500005", city: "Patna" }
]);
{
  acknowledged: true,
  insertedIds: { '0': 1, '1': 2, '2': 3, '3': 4, '4': 5 }
}
```

2.Reading the Data from Agents Collection:

InsuranceDB> db.agents.find();

```
[
  {
    _id: 1,
    agentname: 'Ranjan Kumar',
    phone: '9876500001',
    city: 'Nagpur'
  },
  {
```

```
  _id: 2,
  agentname: 'Greeshmanth',
  phone: '9876500002',
  city: 'Jaipur'
 },
 {
  _id: 3,
  agentname: 'Siddartha Reddy',
  phone: '9876500003',
  city: 'Delhi'
 },
 {
  _id: 4,
  agentname: 'Pranouti',
  phone: '9876500004',
  city: 'Mumbai'
 },
 {
  _id: 5,
  agentname: 'Sarath Sai',
  phone: '9876500005',
  city: 'Patna'
 }
]
```

1.Inserting the data into policies Collection:

```
InsuranceDB> db.policies.insertMany([

db.policies.insertMany([

  { _id: 1, policyname: "Life Secure", type: "Life Insurance", premium: 25000, duration: 20 },

  { _id: 2, policyname: "Health Plus", type: "Health Insurance", premium: 15000, duration: 1 },

  { _id: 3, policyname: "Car Protect", type: "Motor Insurance", premium: 12000, duration: 1 },

  { _id: 4, policyname: "Health Gold", type: "Health Insurance", premium: 20000, duration: 2 },

  { _id: 5, policyname: "Life Diamond", type: "Life Insurance", premium: 30000, duration: 25 }

]);
{

  acknowledged: true,

  insertedIds: { '0': 1, '1': 2, '2': 3, '3': 4, '4': 5 }

}
```

2.Reading the Data from policies Collection:

```
InsuranceDB> db.policies.find()

[

  {

    _id: 1,

    policyname: 'Life Secure',

    type: 'Life Insurance',

    premium: 25000,

    duration: 20

  },

  {

    _id: 2,
```

```
  policyname: 'Health Plus',
  type: 'Health Insurance',
  premium: 15000,
  duration: 1
 },
 {
  _id: 3,
  policyname: 'Car Protect',
  type: 'Motor Insurance',
  premium: 12000,
  duration: 1
 },
 {
  _id: 4,
  policyname: 'Health Gold',
  type: 'Health Insurance',
  premium: 20000,
  duration: 2
 },
 {
  _id: 5,
  policyname: 'Life Diamond',
  type: 'Life Insurance',
  premium: 30000,
  duration: 25
 }
]
```

Inserting data into Customers,Policyassignments,Claims (using Embeddings):

```
InsuranceDB> db.customers.insertMany([
  {
    _id: 1,
    firstname: "Jaya",
    lastname: "Prakash",
    DOB: new Date("2004-08-11"),
    phone: "7981655855",
    email: "jaya@gmail.com",
    policy_history: [
      {
        assignmentid: 1,
        policy_id: 1,
        agent_id: 1,
        startdate: new Date("2023-01-01"),
        enddate: new Date("2043-01-01"),
        claims: [
          { claimid: 1, date: new Date("2024-02-10"), amount: 50000, status: "Approved"
}
        ]
      }
    ]
  },
  {
    _id: 2,
    firstname: "Virat",
    lastname: "Kohli",
    DOB: new Date("2005-06-21"),
    email: "virat@gmail.com",
```

```
    policy_history: [
      {
        assignmentid: 2,
        policy_id: 2,
        agent_id: 2,
        startdate: new Date("2022-06-15"),
        enddate: new Date("2023-06-15"),
        claims: [
          { claimid: 2, date: new Date("2023-11-05"), amount: 30000, status: "Rejected"
}
        ]
      }
    ]
  },
  {
    _id: 3,
    firstname: "Masoom",
    lastname: "Baba",
    DOB: new Date("2001-12-15"),
    policy_history: [
      {
        assignmentid: 3,
        policy_id: 3,
        agent_id: 3,
        startdate: new Date("2021-03-10"),
        enddate: new Date("2022-03-10"),
        claims: [
          { claimid: 3, date: new Date("2024-07-18"), amount: 20000, status: "Rejected"
}
```

```
      ]
    }
  ]
 }
]);
{ acknowledged: true, insertedIds: { '0': 1, '1': 2, '2': 3 } }
```

2.Reading the data from Customers,Policyassignments,Claims

```
InsuranceDB> db.customers.find()
[
 {
  _id: 1,
  firstname: 'Jaya',
  lastname: 'Prakash',
  DOB: ISODate('2004-08-11T00:00:00.000Z'),
  phone: '7981655855',
  email: 'jaya@gmail.com',
  policy_history: [
   {
    assignmentid: 1,
    policy_id: 1,
    agent_id: 1,
    startdate: ISODate('2023-01-01T00:00:00.000Z'),
    enddate: ISODate('2043-01-01T00:00:00.000Z'),
    claims: [
     {
      claimid: 1,
      date: ISODate('2024-02-10T00:00:00.000Z'),
      amount: 50000,
      status: 'Approved'
```

```
      }
     ]
    }
  ]
},
{
  _id: 2,
  firstname: 'Virat',
  lastname: 'Kohli',
  DOB: ISODate('2005-06-21T00:00:00.000Z'),
  email: 'virat@gmail.com',
  policy_history: [
    {
      assignmentid: 2,
      policy_id: 2,
      agent_id: 2,
      startdate: ISODate('2022-06-15T00:00:00.000Z'),
      enddate: ISODate('2023-06-15T00:00:00.000Z'),
      claims: [
        {
          claimid: 2,
          date: ISODate('2023-11-05T00:00:00.000Z'),
          amount: 30000,
          status: 'Rejected'
        }
      ]
    }
  ]
},
```

```
  {
    _id: 3,
    firstname: 'Masoom',
    lastname: 'Baba',
    DOB: ISODate('2001-12-15T00:00:00.000Z'),
    policy_history: [
      {
        assignmentid: 3,
        policy_id: 3,
        agent_id: 3,
        startdate: ISODate('2021-03-10T00:00:00.000Z'),
        enddate: ISODate('2022-03-10T00:00:00.000Z'),
        claims: [
          {
            claimid: 3,
            date: ISODate('2024-07-18T00:00:00.000Z'),
            amount: 20000,
            status: 'Rejected'
          }
        ]
      }
    ]
  }
]
```

CRUD OPERATIONS,AGGREGATIONS,EMBEDDINGS:

1.Find all customers who have a Gmail account.

Ans: db.customers.find({ email: /@gmail\.com$/ });
o/p:

```
[
  {
    _id: 1,
    firstname: 'Jaya',
    lastname: 'Prakash',
    DOB: ISODate('2004-08-11T00:00:00.000Z'),
    phone: '7981655855',
    email: 'jaya@gmail.com',
    policy_history: [
      {
        assignmentid: 1,
        policy_id: 1,
        agent_id: 1,
        startdate: ISODate('2023-01-01T00:00:00.000Z'),
        enddate: ISODate('2043-01-01T00:00:00.000Z'),
        claims: [
          {
            claimid: 1,
            date: ISODate('2024-02-10T00:00:00.000Z'),
            amount: 50000,
            status: 'Approved'
          }
        ]
      }
```

```
    ]
  },
  {
    _id: 2,
    firstname: 'Virat',
    lastname: 'Kohli',
    DOB: ISODate('2005-06-21T00:00:00.000Z'),
    email: 'virat@gmail.com',
    policy_history: [
      {
        assignmentid: 2,
        policy_id: 2,
        agent_id: 2,
        startdate: ISODate('2022-06-15T00:00:00.000Z'),
        enddate: ISODate('2023-06-15T00:00:00.000Z'),
        claims: [
          {
            claimid: 2,
            date: ISODate('2023-11-05T00:00:00.000Z'),
            amount: 30000,
            status: 'Rejected'
          }
        ]
      }
    ]
  }
]
```

2. Delete a customer by their ID.

Ans: db.customers.deleteOne({ _id: 3 });
o/p:Customer id with 3 is deleted successfully.

3. Find policies where the premium is between 15,000 and 25,000.

Ans:

```
db.policies.find({
  $and: [
    { premium: { $gte: 15000 } },
   { premium: { $lte: 25000 } }
  ]
});
[
  {
   _id: 1,
   policyname: 'Life Secure',
   type: 'Life Insurance',
   premium: 25000,
   duration: 20
  },
  {
   _id: 2,
   policyname: 'Health Plus',
   type: 'Health Insurance',
   premium: 15000,
   duration: 1
  },
  {
   _id: 4,
```

```
    policyname: 'Health Gold',
    type: 'Health Insurance',
    premium: 20000,
    duration: 2
  }
]
```

4. Find agents located in either 'Nagpur' or 'Mumbai'

Ans: db.agents.find({ city: { $in: ["Nagpur", "Mumbai"] } });

o/p:

```
[
  {
    _id: 1,
    agentname: 'Ranjan Kumar',
    phone: '9876500001',
    city: 'Nagpur'
  },
  {
    _id: 4,
    agentname: 'Pranouti',
    phone: '9876500004',
    city: 'Mumbai'
  }
```

5. Update a customer's phone number.

Ans: db.customers.updateOne(

```
  { _id: 1 },
  { $set: { phone: "9000000000" } });
```

6. Add a new claim to an existing policy for Customer 1

Ans: db.customers.updateOne(

{ _id: 1, "policy_history.assignmentid": 1 },

{ $push: { "policy_history.$.claims": { claimid: 101, amount: 5000, status: "Pending" } } }

);
o/p:New claim is added successfully

7. Find all customers who have at least one "Approved" claim.

Ans: db.customers.find({ "policy_history.claims.status": "Approved" });

o/p:

```
[
  {
    _id: 1,
    firstname: 'Jaya',
    lastname: 'Prakash',
    DOB: ISODate('2004-08-11T00:00:00.000Z'),
    phone: '9000000000',
    email: 'jaya@gmail.com',
    policy_history: [
      {
        assignmentid: 1,
        policy_id: 1,
        agent_id: 1,
        startdate: ISODate('2023-01-01T00:00:00.000Z'),
        enddate: ISODate('2043-01-01T00:00:00.000Z'),
        claims: [
          {
            claimid: 1,
```

```
        date: ISODate('2024-02-10T00:00:00.000Z'),
        amount: 50000,
        status: 'Approved'
      }
    ]
  }
 ]
}
]
```

8. Calculate the total amount of all "Approved" claims across the entire company.

Ans: db.customers.aggregate([

```
  { $unwind: "$policy_history" },
  { $unwind: "$policy_history.claims" },
  { $match: { "policy_history.claims.status": "Approved" } },
  { $group: { _id: null, totalClaimsPaid: { $sum: "$policy_history.claims.amount" } }
}
```

]);

o/p: [{ _id: null, totalClaimsPaid: 50000 }]

9. Join Customers with Agents to see who sold which policy.

Ans: db.customers.aggregate([

```
  { $unwind: "$policy_history" },
  {
   $lookup: {
    from: "agents",
    localField: "policy_history.agent_id",
    foreignField: "_id",
```

```
      as: "agentDetails"
    }
  },
  { $project: { firstname: 1, "agentDetails.agentname": 1, _id: 0 } }]);
o/p: [
  {
    firstname: 'Jaya',
    agentDetails: [ { agentname: 'Ranjan Kumar' } ]
  },
  {
    firstname: 'Virat',
    agentDetails: [ { agentname: 'Greeshmanth' } ]
  }
]
```

10.Displaying only policyname and Premium amount

Ans: InsuranceDB> db.policies.find({}, { policyname: 1, premium: 1 })

```
o/p:
[
  { _id: 1, policyname: 'Life Secure', premium: 25000 },
  { _id: 2, policyname: 'Health Plus', premium: 15000 },
  { _id: 3, policyname: 'Car Protect', premium: 12000 },
  { _id: 4, policyname: 'Health Gold', premium: 20000 },
  { _id: 5, policyname: 'Life Diamond', premium: 30000 }
]
```

11. Policies sorted by premium (descending)

Ans: db.policies.find().sort({ premium: -1 })

o/p:

```
[
  {
    _id: 5,
    policyname: 'Life Diamond',
    type: 'Life Insurance',
    premium: 30000,
    duration: 25
  },
  {
    _id: 1,
    policyname: 'Life Secure',
    type: 'Life Insurance',
    premium: 25000,
    duration: 20
  },
  {
    _id: 4,
    policyname: 'Health Gold',
    type: 'Health Insurance',
    premium: 20000,
    duration: 2
  },
  {
    _id: 2,
    policyname: 'Health Plus',
```

```
    type: 'Health Insurance',
    premium: 15000,
    duration: 1
  },
  {
    _id: 3,
    policyname: 'Car Protect',
    type: 'Motor Insurance',
    premium: 12000,
    duration: 1
  }
]
```

12. Top 2 highest premium policies

Ans: InsuranceDB> db.policies.find().sort({ premium: -1 }).limit(2)

o/p:

```
[
  {
    _id: 5,
    policyname: 'Life Diamond',
    type: 'Life Insurance',
    premium: 30000,
    duration: 25
  },
  {
    _id: 1,
    policyname: 'Life Secure',
    type: 'Life Insurance',
    premium: 25000,
```

duration: 20 }

13. Show policyname in uppercase

Ans: db.policies.aggregate([{ $project: { policyname: { $toUpper: "$policyname" } } }])

o/p:

[

{ _id: 1, policyname: 'LIFE SECURE' },

{ _id: 2, policyname: 'HEALTH PLUS' },

{ _id: 3, policyname: 'CAR PROTECT' },

{ _id: 4, policyname: 'HEALTH GOLD' },

{ _id: 5, policyname: 'LIFE DIAMOND' }

14. Average premium per policyType

Ans: InsuranceDB> db.policies.aggregate([

{ $group: { _id: "$policyType", avgPremium: { $avg: "$premium" } } }

])

o/p:

[{ _id: null, avgPremium: 20400 }]

15. Skip first 2 records

Ans: InsuranceDB> db.policies.find().skip(2)

o/p:

[

{

_id: 3,

policyname: 'Car Protect',

type: 'Motor Insurance',

premium: 12000,

duration: 1

},

```
  {
    _id: 4,
    policyname: 'Health Gold',
    type: 'Health Insurance',
    premium: 20000,
    duration: 2
  },
  {
    _id: 5,
    policyname: 'Life Diamond',
    type: 'Life Insurance',
    premium: 30000,
    duration: 25
  }
]
```

16. Policies with missing active field

Ans: db.customers.find({ phone: { $exists: false } });

```
o/p: [
  {
    _id: 2,
    firstname: 'Virat',
    lastname: 'Kohli',
    DOB: ISODate('2005-06-21T00:00:00.000Z'),
    email: 'virat@gmail.com',
    policy_history: [
      {
        assignmentid: 2,
        policy_id: 2,
```

```
    agent_id: 2,
    startdate: ISODate('2022-06-15T00:00:00.000Z'),
    enddate: ISODate('2023-06-15T00:00:00.000Z'),
    claims: [
     {
      claimid: 2,
      date: ISODate('2023-11-05T00:00:00.000Z'),
      amount: 30000,
      status: 'Rejected'
     }
    ]
   }
 ]
}

{
 _id: 2,
 agentname: 'Greeshmanth',
 phone: '9876500002',
 city: 'Jaipur'
},
{
 _id: 3,
 agentname: 'Siddartha Reddy',
 phone: '9876500003',
 city: 'Delhi'
},
{
 _id: 4,
```

```
    agentname: 'Pranouti',
    phone: '9876500004',
    city: 'Mumbai'
  },
  {
    _id: 5,
    agentname: 'Sarath Sai',
    phone: '9876500005',
    city: 'Patna'
  }
]
```

17. Find Customers who are NOT from 'Nagpur' (including those where city is missing)

Ans: InsuranceDB> db.agents.find({ city: { $ne: "Nagpur" } });

o/p:[

```
  {
    _id: 2,
    agentname: 'Greeshmanth',
    phone: '9876500002',
    city: 'Jaipur'
  },
  {
    _id: 3,
    agentname: 'Siddartha Reddy',
    phone: '9876500003',
    city: 'Delhi'
  },
  {
```

```
    _id: 4,
    agentname: 'Pranouti',
    phone: '9876500004',
    city: 'Mumbai'
  },
  {
    _id: 5,
    agentname: 'Sarath Sai',
    phone: '9876500005',
    city: 'Patna'
  }
]
```

18. Find Claims where the status is "Pending" OR "Rejected"

Ans:
```
db.customers.find({
  "policy_history.claims.status": { $in: ["Pending", "Rejected"] }
});
```

o/p:
```
[
  {
    _id: 2,
    firstname: 'Virat',
    lastname: 'Kohli',
    DOB: ISODate('2005-06-21T00:00:00.000Z'),
    email: 'virat@gmail.com',
    policy_history: [
      {
        assignmentid: 2,
        policy_id: 2,
```

```
      agent_id: 2,
      startdate: ISODate('2022-06-15T00:00:00.000Z'),
      enddate: ISODate('2023-06-15T00:00:00.000Z'),
      claims: [
        {
          claimid: 2,
          date: ISODate('2023-11-05T00:00:00.000Z'),
          amount: 30000,
          status: 'Rejected'
        }
      ]
    }
  ]
 }
]
```

19. Find customers who have a phone number listed

Ans: db.customers.find({ phone: { $exists: true } });

o/p:

```
[
  {
    _id: 1,
    firstname: 'Jaya',
    lastname: 'Prakash',
    DOB: ISODate('2004-08-11T00:00:00.000Z'),
    phone: '9000000000',
    email: 'jaya@gmail.com',
    policy_history: [
      {
```

```
    assignmentid: 1,
    policy_id: 1,
    agent_id: 1,
    startdate: ISODate('2023-01-01T00:00:00.000Z'),
    enddate: ISODate('2043-01-01T00:00:00.000Z'),
    claims: [
      {
        claimid: 1,
        date: ISODate('2024-02-10T00:00:00.000Z'),
        amount: 50000,
        status: 'Approved'
      }
    ]
  }
 ]
}
]
```

20. Find policies where the 'duration' field is missing

Ans:db.policies.find({ duration: { $exists: false } });

o/p:There are no such entries with duration null.