02-01-2026 - MongoDB Queries

1. Selecting Database

Question: Switch to the Insurance database in MongoDB?

use Insurance

switched to db Insurance

2. Insert Operations

2.1 Insert Customers

Question: Insert multiple customer records into the customers collection?

```
db.customers.insertMany([
  {
    customerID: "CUST001",
    firstName: "Ravi",
    lastName: "Kumar",
    dateOfBirth: ISODate("1988-05-12"),
    phone: "9876543210",
    email: "ravi.kumar@gmail.com"
  },
  {
    customerID: "CUST002",
    firstName: "Anita",
    lastName: "Sharma",
    dateOfBirth: ISODate("1992-09-18"),
    phone: "9123456789",
    email: "anita.sharma@gmail.com"
  }
])
```

```
{
  acknowledged: true,
  insertedIds: {
    '0': ObjectId('6957555638c6bf62641e2621'),
    '1': ObjectId('6957555638c6bf62641e2622')
  }
}
```

2.2 Insert Agents

Question: insert multiple agents?

```
db.agents.insertMany([
  {
    agentID: "AGT001",
    agentName: "Suresh Rao",
    phone: "9000011111",
    city: "Bangalore"
  },
  {
    agentID: "AGT002",
    agentName: "Priya Mehta",
    phone: "9000022222",
    city: "Hyderabad"
  }
])
```

```
{
  acknowledged: true,
  insertedIds: {
    '0': ObjectId('6957556138c6bf62641e2623'),
    '1': ObjectId('6957556138c6bf62641e2624')
  }
}
```

2.3 Insert Policies

Question: insert insurance policies?

```
db.policies.insertMany([
  {
    policyID: "POL001",
    policyName: "Health Secure",
    policyType: "Health",
    premiumAmount: 12000,
    durationYears: 5
  },
  {
    policyID: "POL002",
    policyName: "Life Shield",
    policyType: "Life",
    premiumAmount: 18000,
```

```
    durationYears: 10
  }
])
```

```
{
  acknowledged: true,
  insertedIds: {
    '0': ObjectId('6957556a38c6bf62641e2625'),
    '1': ObjectId('6957556a38c6bf62641e2626')
  }
}
```

2.4 Insert Policy Assignments

Question: assign policies to customers?

```
db.policyAssignments.insertMany([
  {
    assignmentID: "ASN001",
    customerID: "CUST001",
    policyID: "POL001",
    agentID: "AGT001",
    startDate: ISODate("2023-01-01"),
    endDate: ISODate("2028-01-01")
  },
  {
    assignmentID: "ASN002",
    customerID: "CUST002",
    policyID: "POL002",
    agentID: "AGT002",
    startDate: ISODate("2022-06-15"),
    endDate: ISODate("2032-06-15")
  }
])
```

```
{
  acknowledged: true,
  insertedIds: {
    '0': ObjectId('6957557638c6bf62641e2627'),
    '1': ObjectId('6957557638c6bf62641e2628')
```

```
  }
}
```

2.5 Insert Claims

Question: How do you insert claim details?

```
db.claims.insertMany([
  {
    claimID: "CLM001",
    assignmentID: "ASN001",
    claimDate: ISODate("2024-03-10"),
    claimAmount: 45000,
    claimStatus: "Approved"
  },
  {
    claimID: "CLM002",
    assignmentID: "ASN002",
    claimDate: ISODate("2024-07-21"),
    claimAmount: 80000,
    claimStatus: "Pending"
  }
])
```

```
{
  acknowledged: true,
  insertedIds: {
    '0': ObjectId('6957558038c6bf62641e2629'),
    '1': ObjectId('6957558038c6bf62641e262a')
  }
}
```

3. Read Operations

3.1 Retrieve All Customers

Question: fetch all customers?

```
db.customers.find({})
```

```
[
```

```
  {
    _id: ObjectId('6957555638c6bf62641e2621'),
    customerID: 'CUST001',
    firstName: 'Ravi',
    lastName: 'Kumar',
    dateOfBirth: ISODate('1988-05-12T00:00:00.000Z'),
    phone: '9876543210',
    email: 'ravi.kumar@gmail.com'
  },
  {
    _id: ObjectId('6957555638c6bf62641e2622'),
    customerID: 'CUST002',
    firstName: 'Anita',
    lastName: 'Sharma',
    dateOfBirth: ISODate('1992-09-18T00:00:00.000Z'),
    phone: '9123456789',
    email: 'anita.sharma@gmail.com'
  }
]
```

3.2 Find a Specific Customer

Question: find a customer by customerID?

```
db.customers.findOne({ customerID: "CUST001" })
{
  _id: ObjectId('6957555638c6bf62641e2621'),
  customerID: 'CUST001',
  firstName: 'Ravi',
```

```
  lastName: 'Kumar',
  dateOfBirth: ISODate('1988-05-12T00:00:00.000Z'),
  phone: '9876543210',
  email: 'ravi.kumar@gmail.com'
}
```

4. Update Operations

4.1 Update a Single Field

Question: Update the first name of a customer?

```
db.customers.updateOne(
  { customerID: "CUST001" },
  { $set: { firstName: "Revi" } }
)
```

```
{
  acknowledged: true,
  insertedId: null,
  matchedCount: 1,
  modifiedCount: 1,
  upsertedCount: 0
}
```

4.2 Update Multiple Documents

Question: Update the last name of all customers?

```
db.customers.updateMany(
  {},
  { $set: { lastName: "Kumar" } }
)
```

```
{
  acknowledged: true,
  insertedId: null,
```

```
  matchedCount: 2,
  modifiedCount: 1,
  upsertedCount: 0
}
```

4.3 Increase Values

Question: Increase all claim amounts by 5,000?

```
db.claims.updateMany(
  {},
  { $inc: { claimAmount: 5000 } }
)
{
  acknowledged: true,
  insertedId: null,
  matchedCount: 2,
  modifiedCount: 2,
  upsertedCount: 0
}
```

5. Delete Operations

Question: Delete a customer record whose first name is Ramesh?

```
db.customers.deleteOne({ firstName: "Ramesh" })
```

Question: Delete customers whose first name is Ramesh?

```
db.customers.deleteMany({ firstName: "Ramesh" })
```

6. Query Operators

6.0 Basic Claim Amount Queries

Question: Find claims with claimAmount equal to 80,000.

```
db.claims.find({ claimAmount: { $eq: 80000 } }, { _id: 0 })
[
  {
    claimID: 'CLM002',
    assignmentID: 'ASN002',
    claimDate: ISODate('2024-07-21T00:00:00.000Z'),
    claimAmount: 80000,
    claimStatus: 'Pending'
  }
]
```

Question: Find claims with claimAmount greater than or equal to 80,000.

```
db.claims.find({ claimAmount: { $gte: 80000 } }, { _id: 0 })
[
  {
    claimID: 'CLM002',
    assignmentID: 'ASN002',
    claimDate: ISODate('2024-07-21T00:00:00.000Z'),
    claimAmount: 80000,
    claimStatus: 'Pending'
  }
]
```

Question: Find claims with claimAmount less than 80,000.

```
db.claims.find({ claimAmount: { $lt: 80000 } }, { _id: 0 })
[
  {
```

```
    claimID: 'CLM001',
    assignmentID: 'ASN001',
    claimDate: ISODate('2024-03-10T00:00:00.000Z'),
    claimAmount: 45000,
    claimStatus: 'Approved'
  }
]
```

Question: Find claims with claimAmount less than or equal to 80,000.

```
db.claims.find({ claimAmount: { $lte: 80000 } }, { _id: 0 })
[
  {
    claimID: 'CLM001',
    assignmentID: 'ASN001',
    claimDate: ISODate('2024-03-10T00:00:00.000Z'),
    claimAmount: 45000,
    claimStatus: 'Approved'
  },
  {
    claimID: 'CLM002',
    assignmentID: 'ASN002',
    claimDate: ISODate('2024-07-21T00:00:00.000Z'),
    claimAmount: 80000,
    claimStatus: 'Pending'
  }
]
```

6.1 Comparison Operators

Question: Find claims with amount greater than or equal to 80,000.

```
db.claims.find({ claimAmount: { $gte: 80000 } }, { _id: 0 })
```

6.2 Logical Operators

Question: Find claims with amount 80,000 and status Pending.

```
db.claims.find({
  $and: [
    { claimAmount: 80000 },
    { claimStatus: "Pending" }
  ]
})
```

```
[
  {
    claimID: 'CLM002',
    assignmentID: 'ASN002',
    claimDate: ISODate('2024-07-21T00:00:00.000Z'),
    claimAmount: 80000,
    claimStatus: 'Pending'
  }
]
```

6.3 Pattern Matching

Question: Find customers whose phone number starts with 9.

```
db.customers.find({ phone: { $regex: "^9" } })
```

```
[
  {
    _id: ObjectId('6957555638c6bf62641e2621'),
    customerID: 'CUST001',
```

```
    firstName: 'Ravi',
    lastName: 'Kumar',
    dateOfBirth: ISODate('1988-05-12T00:00:00.000Z'),
    phone: '9876543210',
    email: 'ravi.kumar@gmail.com'
  },
  {
    _id: ObjectId('6957555638c6bf62641e2622'),
    customerID: 'CUST002',
    firstName: 'Anita',
    lastName: 'Kumar',
    dateOfBirth: ISODate('1992-09-18T00:00:00.000Z'),
    phone: '9123456789',
    email: 'anita.sharma@gmail.com'
  }
]
```

7. Aggregation Framework

7.0 Total Number of Claims by Claim Status

Question: Find the total number of claims grouped by claim status.

```
db.claims.aggregate([
  {
    $group: {
      _id: "$claimStatus",
      totalClaims: { $sum: 1 }
    }
  }
])
```

```
[
  { _id: 'Pending', totalClaims: 1 },
  { _id: 'Approved', totalClaims: 1 }
]
```

7.1 Total Claim Amount per Customer

Question: Calculate total claim amount per customer?

```
db.claims.aggregate([
  {
    $lookup: {
      from: "policyAssignments",
      localField: "assignmentID",
      foreignField: "assignmentID",
      as: "assignment"
    }
  },
  { $unwind: "$assignment" },
  {
    $group: {
      _id: "$assignment.customerID",
      totalClaimAmount: { $sum: "$claimAmount" }
    }
  }
])
```

```
[
  { _id: 'CUST001', totalClaimAmount: 50000 },
  { _id: 'CUST002', totalClaimAmount: 85000 }
]
```

7.2 Number of Policies per Customer

Question: find how many policies each customer has?

```
db.policyAssignments.aggregate([
  {
    $group: {
      _id: "$customerID",
      totalPolicies: { $sum: 1 }
    }
  }
])
```

```
[

  { _id: 'CUST001', totalPolicies: 1 },

  { _id: 'CUST002', totalPolicies: 1 }

]
```

7.3 Customers with Total Claims Above 50,000

Question: find customers whose total claim amount exceeds 50,000?

```
db.claims.aggregate([
  {
    $lookup: {
      from: "policyAssignments",
      localField: "assignmentID",
      foreignField: "assignmentID",
      as: "assignment"
    }
  },
  { $unwind: "$assignment" },
  {
    $group: {
      _id: "$assignment.customerID",
      totalClaimAmount: { $sum: "$claimAmount" }
    }
  },
  {
    $match: {
      totalClaimAmount: { $gt: 50000 }
    }
  }
])
```

```
[ { _id: 'CUST002', totalClaimAmount: 85000 } ]
```

7.4 Sort Customers by Total Claim Amount

Question: How do you sort customers by total claim amount in descending order?

```
db.claims.aggregate([
  {
    $lookup: {
      from: "policyAssignments",
      localField: "assignmentID",
      foreignField: "assignmentID",
      as: "assignment"
    }
  },
  { $unwind: "$assignment" },
  {
    $group: {
      _id: "$assignment.customerID",
      totalClaimAmount: { $sum: "$claimAmount" }
    }
  },
  { $sort: { totalClaimAmount: -1 } }
])
```

```
[

  { _id: 'CUST002', totalClaimAmount: 85000 },

  { _id: 'CUST001', totalClaimAmount: 50000 }

]
```