Name: Divya Jha
Roll No: L021
Sap Id: 40778230025
Practical : Indexing in MongoDB

Introduction

Indexing in MongoDB is crucial for improving query performance by allowing the database to locate documents more efficiently.

Types of Indexes

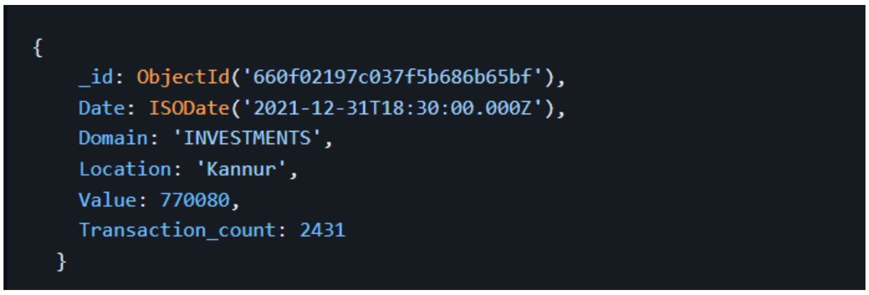
MongoDB supports various types of indexes, including:

- Single Field Indexes: Indexes created on a single field of a document.
- Compound Indexes: Indexes created on multiple fields within a document.
- Multikey Indexes: Indexes created on arrays of data.
- Text Indexes: Indexes designed for textual search.
- Geospatial Indexes: Indexes optimized for geospatial queries.
- Hashed Indexes: Indexes where MongoDB hashes the index keys' values.

Creating Indexes

Indexes can be created using the *createIndex()* method or by specifying indexes in a document's schema for collections using MongoDB's schema validation feature.

Suppose the document is like

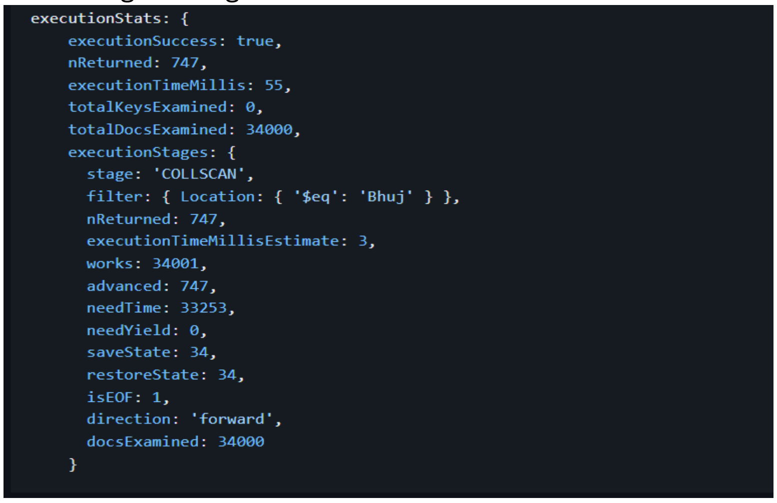


```
{
    _id: ObjectId('660f02197c037f5b686b65bf'),
    Date: ISODate('2021-12-31T18:30:00.000Z'),
    Domain: 'INVESTMENTS',
    Location: 'Kannur',
    Value: 770080,
    Transaction_count: 2431
  }
```

1. **Find documents where Location is 'Bhuj' without index:**

   db.sales.find({ Location: 'Bhuj' }).explain("executionStats");

Name: Divya Jha
Roll No: L021
Sap Id: 40778230025
Practical : Indexing in MongoDB



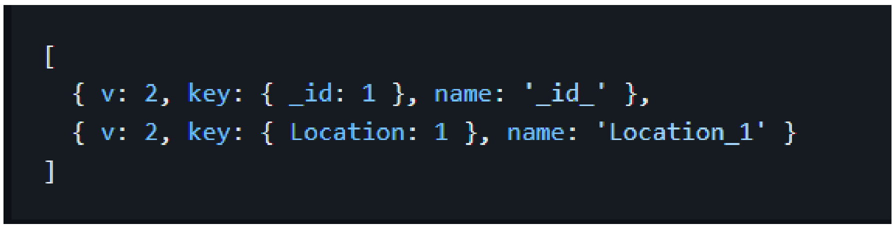
```
  executionStats: {
      executionSuccess: true,
      nReturned: 747,
      executionTimeMillis: 55,
      totalKeysExamined: 0,
      totalDocsExamined: 34000,
      executionStages: {
        stage: 'COLLSCAN',
        filter: { Location: { '$eq': 'Bhuj' } },
        nReturned: 747,
        executionTimeMillisEstimate: 3,
        works: 34001,
        advanced: 747,
        needTime: 33253,
        needYield: 0,
        saveState: 34,
        restoreState: 34,
        isEOF: 1,
        direction: 'forward',
        docsExamined: 34000
      }
```

2. **Now Creating a single field index on Location Filed:**

   db.sales.createIndex({ Location: 1 });

3. **Retrieve the list of created indexes**

   db.sales.getIndexes()

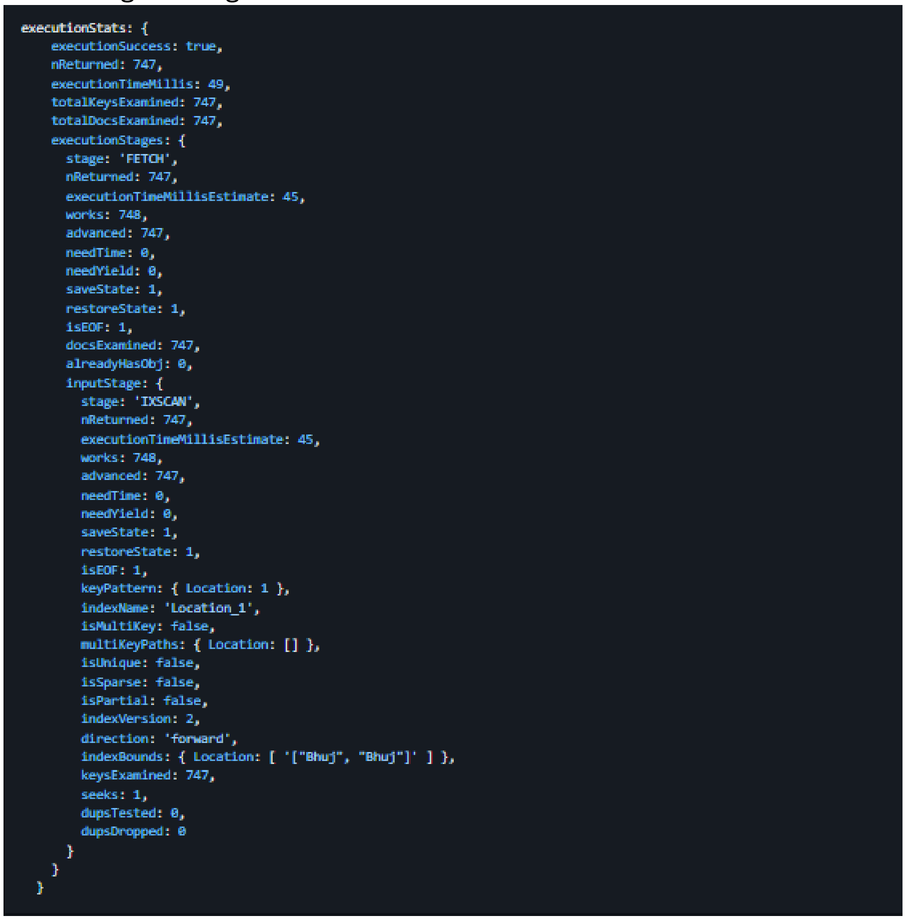


```
[
  { v: 2, key: { _id: 1 }, name: '_id_' },
  { v: 2, key: { Location: 1 }, name: 'Location_1' }
]
```

4. **Find documents where Location is 'Bhuj' with index**

   db.sales.find({ Location: 'Bhuj' }).explain("executionStats");

Name: Divya Jha
Roll No: L021
Sap Id: 40778230025
Practical : Indexing in MongoDB



```
executionStats: {
    executionSuccess: true,
    nReturned: 747,
    executionTimeMillis: 49,
    totalKeysExamined: 747,
    totalDocsExamined: 747,
    executionStages: {
      stage: 'FETCH',
      nReturned: 747,
      executionTimeMillisEstimate: 45,
      works: 748,
      advanced: 747,
      needTime: 0,
      needYield: 0,
      saveState: 1,
      restoreState: 1,
      isEOF: 1,
      docsExamined: 747,
      alreadyHasObj: 0,
      inputStage: {
        stage: 'IXSCAN',
        nReturned: 747,
        executionTimeMillisEstimate: 45,
        works: 748,
        advanced: 747,
        needTime: 0,
        needYield: 0,
        saveState: 1,
        restoreState: 1,
        isEOF: 1,
        keyPattern: { Location: 1 },
        indexName: 'Location_1',
        isMultiKey: false,
        multiKeyPaths: { Location: [] },
        isUnique: false,
        isSparse: false,
        isPartial: false,
        indexVersion: 2,
        direction: 'forward',
        indexBounds: { Location: [ '["Bhuj", "Bhuj"]' ] },
        keysExamined: 747,
        seeks: 1,
        dupsTested: 0,
        dupsDropped: 0
      }
    }
  }
```

5. **Deleting the index on Location field**

   db.sales.dropIndex('Location_1')

   Conclusion:
   Before index creation, the query to find documents where the Location is 'Bhuj' took approximately 55 milliseconds, involving a collection scan (COLLSCAN) with a total of 34000 documents examined. After index creation, the same query utilized the created index (Location_1) resulting in a more efficient index scan (IXSCAN) which took approximately 49 milliseconds, with significantly fewer keys and documents examined, thus improving the query performance.