M5 - Aggregation and Indexing

Problem Statement: Design and Develop MongoDB Queries using Aggregation operations: Create Employee collection by considering following Fields: i. Emp_id : Number ii. Name: Embedded Doc (FName, LName) iii. Company Name: String iv. Salary: Number v. Designation: String vi. Age: Number vii. Expertise: Array viii. DOB: String or Date ix. Email id: String x. Contact: String xi. Address: Array of Embedded Doc (PAddr, LAddr) Insert at least 5 documents in collection by considering above attribute and execute following: 1. Using aggregation Return separates value in the Expertise array and return sum of each element of array. 2. Using Aggregate method return Max and Min Salary for each company. 3. Using Aggregate method find Employee with Total Salary for Each City with Designation="DBA". 4. Using aggregation method Return separates value in the Expertise array for employee name where Swapnil Jadhav 5. To Create Compound Indexes on Name: 1, Age: -1 6. Create an Index on Emp_id field, compare the time require to search Emp_id before and after creating an index. (Hint Add at least 10000 Documents) 7. Return a List of Indexes on created on employee Collection.

Creating database & collection:

```
use empDB3
db.createCollection("Employee")
```

Inserting data:

```
db.Employee.insertMany([
{
  Name: {FName: "Ayush", LName: "Kalaskar"},
  Company: "TCS",
  Salary: 45000,
  Designation: "Programmer",
  Age: 24,
  Expertise: ['Docker', 'Linux', 'Networking', 'Politics'],
  DOB: new Date("1998-03-12"),
  Email: "ayush.k@tcs.com",
  Contact: 9972410427,
  Address: [{PAddr: "Kokan, Maharashtra"}, {LAddr: "Lohegaon, Pune", Pin_code: 411014}]
},
{
  Name: {FName: "Mehul", LName: "Patil"},
  Company: "MEPA",
  Salary: 55000,
  Designation: "Tester",
  Age: 20,
  Expertise: ['HTML', 'CSS', 'Javascript', 'Teaching'],
  DOB: new Date("1964-06-22"),
  Email: "mehul.p@mepa.com",
  Contact: 9972410426,
  Address: [{PAddr: "NDB, Maharashtra"}, {LAddr: "Camp, Pune", Pin_code: 411001}]
},
{
  Name: {FName: "Himanshu", LName: "Patil"},
  Company: "Infosys",
  Salary: 85000,
  Designation: "Developer",
  Age: 67,
  Expertise: ['Mongodb', 'Mysql', 'Cassandra', 'Farming'],
  DOB: new Date("1957-04-28"),
  Email: "himanshu.p@infosys.com",
  Contact: 9972410425,
  Address: [{PAddr: "NDB, Maharashtra"}, {LAddr: "Camp, Pune", Pin_code: 411001}]
},
{
  Name: {FName: "Swapnil", LName: "Jadhav"},
  Company: "Wayne Industries",
  Salary: 95000,
  Designation: "DBA",
  Age: 75,
  Expertise: ['Blockchain', 'Hashing', 'Encryption', 'Nerd'],
  DOB: new Date("1949-12-28"),
  Email: "swapnil.j@wayne.com",
  Contact: 9972410427,
  Address: [{PAddr: "Viman Nagar, Pune"}, {LAddr: "Viman Nagar, Pune", Pin_code: 411001}]
}
])
```

Queries

1. Using aggregation Return separates value in the Expertise array and return sum of each element of array.

```
db.Employee.aggregate([
  {
    $unwind: "$Expertise"
  },
  {
    $group: {
      _id: "$Expertise",
      count: { $sum: 1 }
    }
  }
])
```

2. Using Aggregate method return Max and Min Salary for each company.

```
db.Employee.aggregate([
  {
    $group: {
      _id: "$Company",
      MIN: { $min: "$Salary" },
      MAX: { $max: "$Salary" }
    }
  }
])
```

3. Using Aggregate method find Employee with Total Salary for Each City with Designation=“DBA”.

```
db.Employee.aggregate([
  {
    $match: {
      Designation: "DBA"
    }
  },
  {
    $group: {
      _id: "$Address.PAddr",
      Total: { $sum: "$Salary" }
    }
  }
])
```

4. Using aggregation method Return separates value in the Expertise array for employee name where Swapnil Jadhav

```
db.Employee.aggregate([
  {
    $match: {
    "Name.FName": "Swapnil",
    "Name.LName": "Jadhav"
    }
  },
  {
    $unwind: "$Expertise"
  },
  {
    $group: {
      _id: "$Expertise"
    }
  }
])
```

5. To Create Compound Indexes on Name: 1, Age: -1

```
db.Employee.createIndex({Name: 1, Age: -1})
```

6. Create an Index on Emp_id field, compare the time require to search Emp_id before and after creating an index. (Hint Add at least 10000 Documents)

```
// Creating 10000 documents
for (let i=1; i<=10000; i++) {
  db.Employee.insertOne({
    Emp_id: i,
    Name: `Employee ${i}`,
    Designation: `Work ${i*5}`
  });
}
// Wait for it to insert 10000 documents!

// Time without index
let startTime = new Date();
db.Employee.find( { Emp_id: 7500 } );
let endTime = new Date();
print("Time taken to search before index: " + (endTime - startTime) + "ms");

// Creating index on Emp_id
db.Employee.createIndex( { Emp_id: 1 } )

// Time with index
startTime = new Date();
db.Employee.find( { Emp_id: 7500 } );
endTime = new Date();
print("Time taken to search after index: " + (endTime - startTime) + "ms")
```

Output for query 6:

Time taken to search before index: 57ms Time taken to search after index: 35ms

7. Return a List of Indexes on created on employee Collection.

```
db.Employee.getIndexes();
```