Indexing and Aggregation Basics

(Group-7)

Name : Movva Venkatesh

Regd.No : 211FA04084

Section : CSE-4G

1. Create an index on the category field of the products collection to speed up queries filtering by category.

```
> db.products.createIndex({ category: 1 });
< category_1
```

2. Write an aggregation query to count the number of products in each category in the products collection.

```
> db.products.aggregate([
    { $group: { _id: "$category", totalProducts: { $sum: 1 } } }
  ]);
< {
    _id: 'Electronics',
    totalProducts: 3
  }
  {
    _id: 'Appliances',
    totalProducts: 1
  }
```

3. Create a compound index on price and stock fields in the products collection.

```
> db.products.createIndex({ price: 1, stock: 1 });
< price_1_stock_1
```

4. Write an aggregation query to calculate the total revenue generated by each productId in the sales collection.

```
> db.sales.aggregate([
    { $group: { _id: "$productId", totalRevenue: { $sum: "$revenue" } } }
  ]);
< {
    _id: 3,
    totalRevenue: 640
  }
  {
    _id: 1,
    totalRevenue: 18000
  }
  {
    _id: 2,
    totalRevenue: 3000
  }
```

5. Create a unique index on the email field of the users collection.

```
> db.users.createIndex({ email: 1 }, { unique: true });
< email_1
```

6. Write an aggregation query to find the average rating of products in the reviews collection.

```
> db.reviews.aggregate([
    { $group: { _id: "$productId", averageRating: { $avg: "$rating" } } }
  ]);
< {
    _id: 1,
    averageRating: 4.75
  }
  {
    _id: 2,
    averageRating: 4
  }
  {
    _id: 3,
    averageRating: 3.5
  }
```

7. Create a text index on the description field of the products collection for text search.

```
> db.products.createIndex({ description: "text" });
< description_text
```

8. Write an aggregation query to find the total number of orders placed by each customer in the orders collection.

```
> db.orders.aggregate([
    { $group: { _id: "$customerId", totalOrders: { $sum: 1 } } }
  ]);
< {
    _id: 1,
    totalOrders: 2
  }
  {
    _id: 2,
    totalOrders: 1
  }
  {
    _id: 3,
    totalOrders: 1
  }
```

9.Create an index on the createdAt field of the orders collection to optimize sorting by date.

```
> db.orders.createIndex({ createdAt: 1 });
< createdAt_1
```

10.Write an aggregation query to find the top 5 products by quantity sold in the sales collection.

```
> db.sales.aggregate([
    { $group: { _id: "$productId", totalQuantity: { $sum: "$quantity" } } },
    { $sort: { totalQuantity: -1 } },
    { $limit: 5 }
  ]);
< {
    _id: 1,
    totalQuantity: 15
  }
  {
    _id: 2,
    totalQuantity: 15
  }
  {
    _id: 3,
    totalQuantity: 8
```