Experiment: 9

Develop a query to demonstrate Text search using catalog data collection for a given word

Suppose your `catalog_data` collection has a field named `description` where you want to perform the text search.

```
db.catalog_data.insertMany([
{
item: "Laptop",
description: "High-performance laptop for gaming and productivity tasks",
category: "Electronics" ,
price:1200,
details:
{brand: " apple",size:14}
},
{
item: "Smartphone",
description: "Latest smartphone with advanced camera features and technology",
category: "Electronics" ,
price:3400,
details:
{brand: " lg",size:14}
},
{
item: "Book",
description: "Classic novel set in the Victorian era",
category: "Books" ,
price:6780
},
{
item: "Television",
description: "Ultra HD television with smart features",
category: "Electronics" ,
price:5460,
details:
{brand: " lg",size:14}
},
{
item: "Headphones",
description: "Wireless headphones with noise-cancellation technology",
category: "Electronics" ,
price:8900,
details:
{brand: " boat",size:12}
},
{
item: "Smartphone",
description: "Latest smartphone with latest technology ",
category: "Electronics" ,
price:7632,
details:
{brand: " samsung",size:13}
}
])
> db.catalog_data.find({ $text: { $search: "given_word" } }).explain("executionStats")
```

1. Creating Text Index:

First, you need to create a text index on the field where you want to perform the text search. In this case, let's create a text index on the `description` field:

```
> db.catalog_data.createIndex({ description: "text" })
```

2. Executing Text Search Query:

Now, you can perform a text search using the `$text` operator along with the `$search` parameter:

```
> db.catalog_data.find({ $text: { $search: "given_word" } })
```

Replace `"given_word"` with the word you want to search for in the `description` field.

3. Query Explain:

As before, you can use the `explain()` method to get information about how MongoDB executes the text search query:

```
> db.catalog_data.find({ $text: { $search: "given_word" } }).explain("executionStats")
```

This command provides execution statistics for the text search query.

4. Additional Text Search Options:

MongoDB's text search also supports additional options such as case insensitivity, language- specific stemming, and weighting of search terms. You can specify these options using the `$text` operator.

```
>db.catalog_data.find({ $text: { $search: "given_word", $caseSensitive: false, $language:"english" } })
```

Adjust the options (`$caseSensitive`, `$language`, etc.) according to your requirements.

b.Develop queries to illustrate excluding documents with certain words and phrases

The `$not` operator in conjunction with regular expressions or the `$regex` operator. Here's how you can develop queries to illustrate excluding documents with certain words and phrases:

Suppose you have a collection named `documents` and you want to exclude documents containing specific words or phrases from a field named `text`.

1. Using Regular Expressions:

You can use regular expressions to match documents that do not contain certain words or phrases. For example, to exclude documents containing the word "example", you can use the following query:

```
> db.documents.find({ text: { $not: /example/ } })
```

This query returns all documents from the `documents` collection where the `text` field does not contain the word "example".

2. Using the $regex Operator:

Alternatively, you can use the `$regex` operator to achieve the same result. For example, to exclude documents containing the phrase "example phrase", you can use the following query:

```
> db.documents.find({ text: { $not: { $regex: "example phrase" } } })
```

This query returns all documents from the `documents` collection where the `text` field does not contain the phrase "example phrase".

3. Query Explain:

As always, you can use the `explain()` method to get information about how MongoDB executes the

query:

```
>db.documents.find({ text: { $not: /example/ } }).explain("executionStats")
```

This command provides execution statistics for the query.vv

By using the `$not` operator with regular expressions or the `$regex` operator, you can exclude documents with certain words or phrases from your MongoDB queries. Adjust the regular expressions or regex patterns according to your specific exclusion criteria.