**Create a Collection**

Create a collection stores with the following documents:

|  |
| --- |
| db.stores.insertMany( |
| [ |
| { \_id: 1, name: **"Java Hut"**, description: **"Coffee and cakes"** }, |
| { \_id: 2, name: **"Burger Buns"**, description: **"Gourmet hamburgers"** }, |
| { \_id: 3, name: **"Coffee Shop"**, description: **"Just coffee"** }, |
| { \_id: 4, name: **"Clothes Clothes Clothes"**, description: **"Discount clothing"** }, |
| { \_id: 5, name: **"Java Shopping"**, description: **"Indonesian goods"** } |
| ] |
| ) |

**Create a Text Index**

Run the following in [mongosh](https://www.mongodb.com/docs/mongodb-shell/" \l "mongodb-binary-bin.mongosh" \t "_self) to allow text search over the name and description fields:

|  |
| --- |
| db.stores.createIndex( { name: **"text"**, description: **"text"** } ) |

**Search for an Exact Phrase**

You can also search for exact phrases by wrapping them in double-quotes. If the $search string includes a phrase and individual terms, text search will only match documents that include the phrase.

For example, the following will find all documents containing "coffee shop":

|  |
| --- |
| db.stores.find( { $text: { $search: **"\"coffee shop\""** } } ) |

For more information, see [Phrases.](https://www.mongodb.com/docs/manual/reference/operator/query/text/#std-label-text-operator-phrases)

**Exclude a Term**

To exclude a word, you can prepend a "-" character. For example, to find all stores containing "java" or "shop" but not "coffee", use the following:

|  |
| --- |
| db.stores.find( { $text: { $search: **"java shop -coffee"** } } ) |

**Sort the Results**

MongoDB will return its results in unsorted order by default. However, text search queries will compute a relevance score for each document that specifies how well a document matches the query.

To sort the results in order of relevance score, you must explicitly project the [$meta](https://www.mongodb.com/docs/manual/reference/operator/aggregation/meta/#mongodb-expression-exp.-meta) textScore field and sort on it:

|  |
| --- |
| db.stores.find( |
| { $text: { $search: **"java coffee shop"** } }, |
| { score: { $meta: **"textScore"** } } |
| ).sort( { score: { $meta: **"textScore"** } } ) |

Text search is also available in the aggregation pipeline.

Sorting:

db.COLLECTION\_NAME.find().sort({KEY:1})