Chapter 2: MongoDB Indexes

Lab 2.1: Using Indexes to Sort

Back to the Question

The key to this lab is to identify the prefixes for the given index, and to take your time and think about each query one by one.

Here's an explanation for each query:

- `db.people.find({ "first_name": { $gt: "J" } }).sort({ "address.city": -1 })`

  No, this query doesn't use equality on the index prefix. When using an index for filtering and sorting the query must include equality conditions on all the prefix keys that precede the sort keys. Moreover, on the sort predicate it skipped the next key in the prefix `"address.state"`.

- `db.people.find({ "first_name": "Jessica" }).sort({ "address.state": 1, "address.city": 1 })`

  Yes, this query matches with equality on the query predicate with an index prefix, and continues the prefix in the sort predicate by walking the index backward.

- `db.people.find({ "first_name": "Jessica", "address.state": { $lt: "S"} }).sort({ "address.state": 1 })`

  Yes, while this query fails to use equality on the `"address.state"` field of the index prefix, it uses the same field for sorting.

- `db.people.find({ "address.city": "West Cindy" }).sort({ "address.city": -1 })`

  No, this query does not use an index prefix.

- `db.people.find({ "address.state": "South Dakota", "first_name": "Jessica" }).sort({ "address.city": -1 })`

  Yes, this query is able to use the index prefix. The order of the fields in the query predicate does not matter.

Proceed to next section