Janis Gonzalez

WEB 335 Introduction to NoSQL

Discussion 6.1 Indexes

April 20,2023

Bellevue University

MongoDB indexing is used to help the database scan various documents of a collection that match a query written. Indexes are special data structures that are able to store small portions of data sets in an easier form to be able to be scanned. An index stores the value of a specific set of fields as ordered by a value of a field that is specified in the index (MongoDB – indexing). When creating an index, the method on MongoDB would be to use createIndex(). The syntax for this would look like: db.NAME\_OF\_COLLECTION.createIndex.({KEY:1}) (MongoDB – indexing). The number 1 is used for an ascending order and –1 is used for a descending order. The key name is used as the name of the field where one would create the index. To create multiple fields, it would look like: db.mytitle.createIndex({“title”:1,”description”:-1}) (MongoDB – indexing). To drop a specific index, the dropIndex() method is used in MongoDB. The syntax for dropIndex() is db.NAME\_OF\_COLLECTION.dropIndex({KEY:1}) (MongoDB – indexing). Key is used for the name of the file where one wants to remove this existing index. Another way to use this dropIndex() is to just specify it as dropIndex(“index\_name”) (MongoDB – indexing). In order to drop multiple index is to use the syntax: db.NAME\_OF\_COLLECTION.dropIndexes() (MongoDB – indexing).

Cardinality in indexing MongoDB is defined by the number of unique elements that are in the sets (Understanding indexing and cardinality for mongodb). The more duplicated elements the lower the cardinality. Sparse indexes are a common type of index type in MongoDB. Sparse indexes allow to create an index where the value of a field is not undefined (Understanding indexing and cardinality for mongodb). This is also helpful when there are optional unique fields especially if the index is on a field where the majority of the field is undefined. Capped collections have fixed-sized collections that help retrieve documents based on insertion order (Capped Collections). Capped collections work by being able to overwrite the older documents in a collection when new documents are added in a collection (Capped Collections). To create a capped collection, it should look like: db.createCollection( “log”, { capped: true, size: 10000, max: 300 } ) (Capped Collections). Where max is the maximum number of documents for the collection to have. Capped collections are best used for storing log information or cache data as they have high volume in terms of data.

References

*Capped Collections*. MongoDB. (n.d.). Retrieved from https://www.mongodb.com/docs/manual/core/capped-collections/#:~:text=Capped%20collections%20are%20fixed-size,documents%20based%20on%20insertion%20order.

*MongoDB - indexing*. Tutorials Point. (n.d.). Retrieved from https://www.tutorialspoint.com/mongodb/mongodb\_indexing.htm

*Understanding indexing and cardinality for mongodb*. Bharat Kalluri. (n.d.). Retrieved from https://bharatkalluri.com/posts/cardinality-and-indexing-mongodb