



MODULE-6

INDEXING AND AGGREGATION FRAMEWORK

→ **Module 1**

- » Design Goals, Architecture and Installation

→ **Module 2**

- » CRUD Operations

→ **Module 3**

- » Schema Design and Data Modelling

→ **Module 4**

- » Administration

→ **Module 5**

- » Scalability and Availability

→ **Module 6**

- » **Indexing and Aggregation Framework**

→ **Module 7**

- » Application Engineering and MongoDB Tools

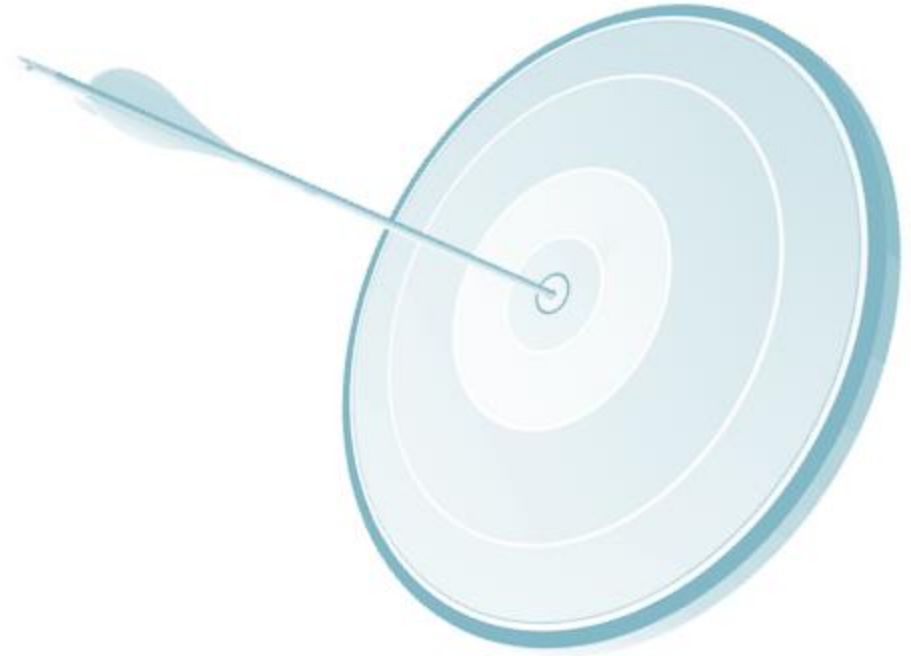
→ **Module 8**

- » Project, Additional Concepts and Case Studies

Objectives

At the end of this module, you will be able to

- Use various type of indexes in MongoDB®
- Use hint, explain plan of a query
- Work with Geospatial indexes
- Work with Aggregation Pipeline in MongoDB®
- Use MapReduce framework





What is Asynchronous Replication?



Secondary's apply operations from the primary asynchronously. By applying operations after the primary, sets can continue to function without some members. However, as a result secondary's may not return the most current data to clients.



What kinds of replication does MongoDB support?



MongoDB supports master-slave replication and a variation on master-slave replication known as replica sets. Replica sets are the recommended replication topology.



What is meaning of "secondary" and "slave"?



Secondary and slave nodes are read-only nodes that replicate from the primary.



What is Priority 0 Replica Set Members?



A priority 0 member is a secondary that cannot become primary. Priority 0 members cannot trigger elections.



Which members of a replica set vote in elections?



All members of a replica set, unless the value of votes is equal to 0, vote in elections. This includes all delayed, hidden and secondary-only members, as well as the arbiters.



How does sharding work with replication?



To use replication with sharding, deploy each shard as a replica set.



What happens to un-sharded collections in sharded databases?



In the current implementation, all databases in a sharded cluster have a "primary shard." All un-sharded collection within that database will reside on the same shard.

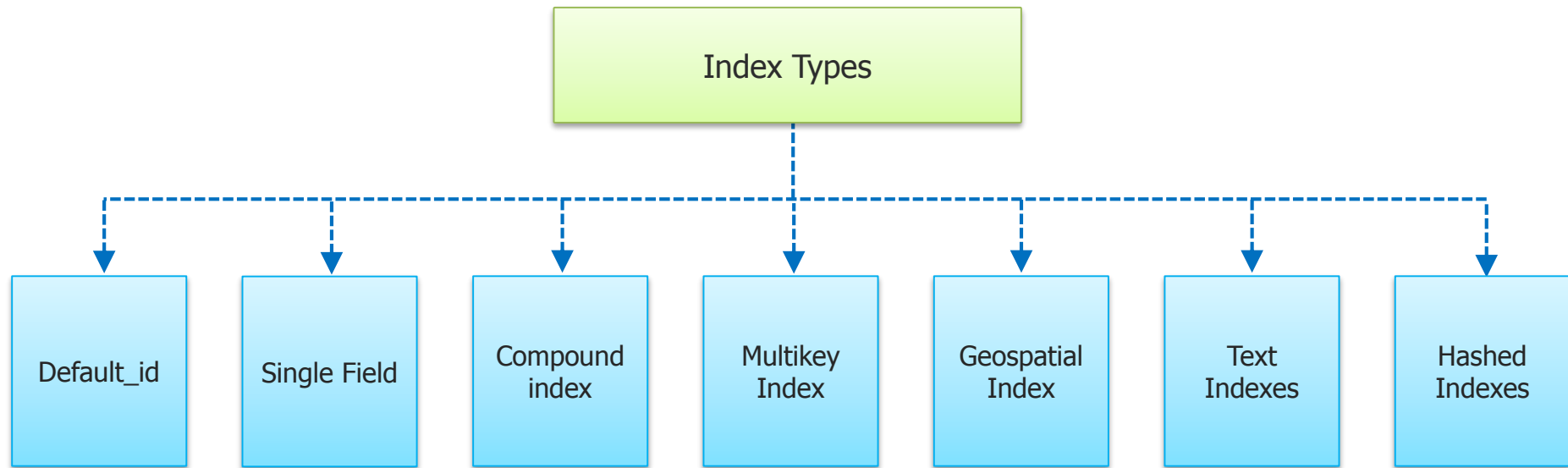


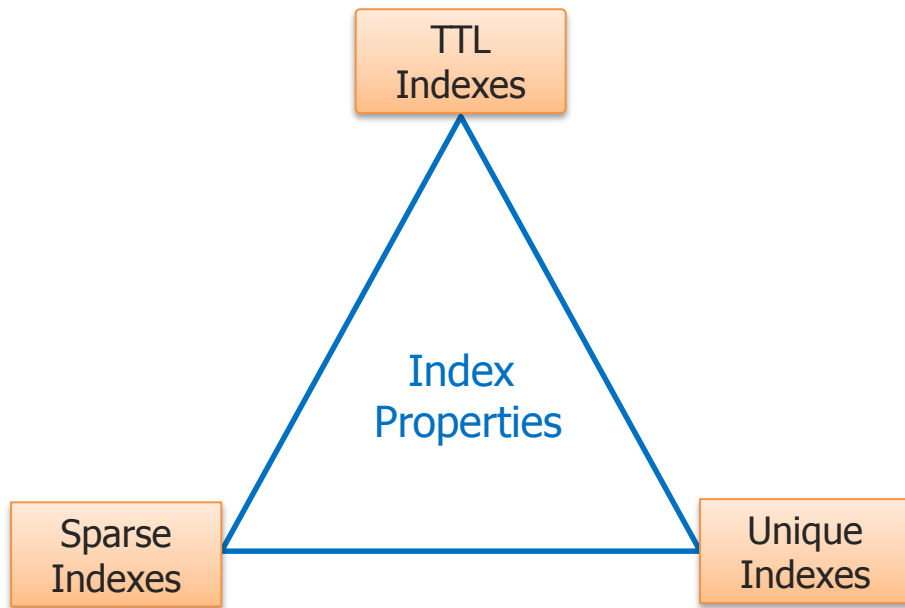
Is it safe to remove old files in the moveChunk directory?

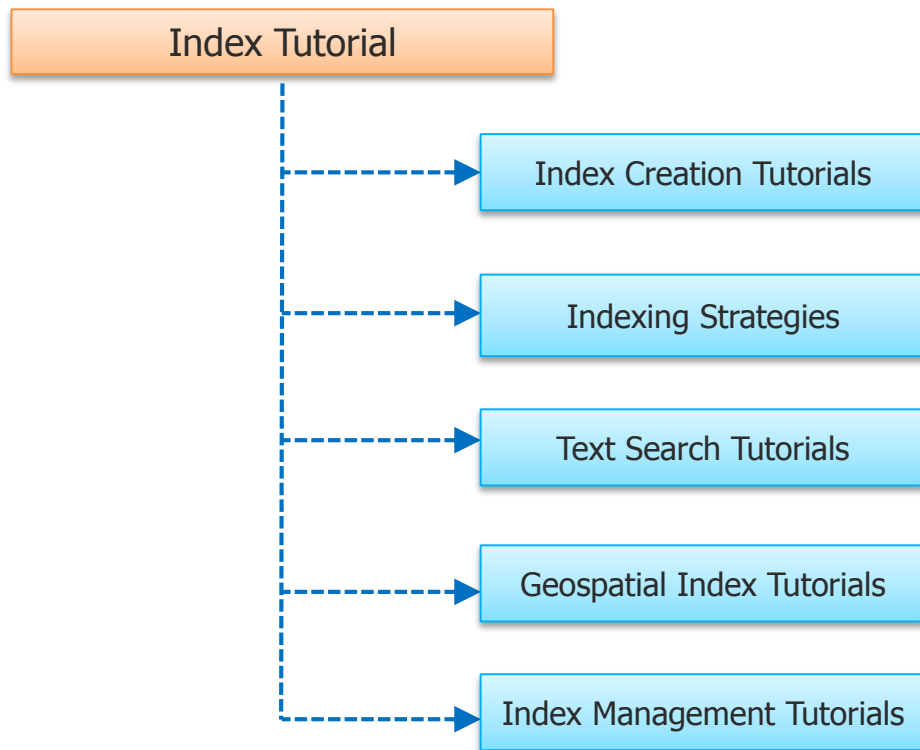


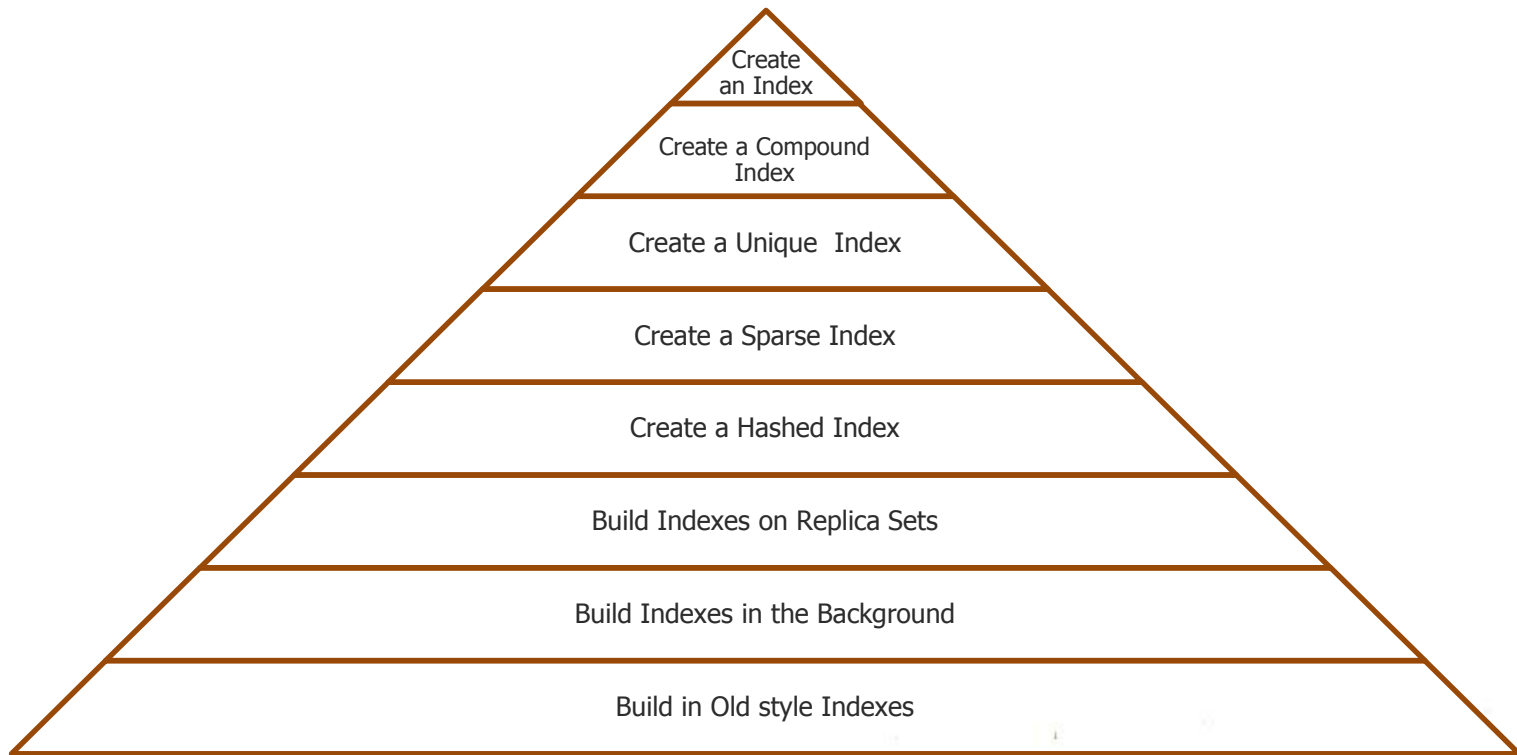
Yes. mongod creates these files as backups during normal shard balancing operations. Once these migrations are complete, you may delete these files.

- Indexes provide high performance read operations for frequently used queries.
- Without indexes, MongoDB must scan every document of a collection to select those documents that match the query statement.
- These collection scans are inefficient and require the mongod to process a large volume of data for each operation.
- Indexes are special data structures that store a small portion of the collection's data set in an easy to traverse form.
- The index stores the value of a specific field or set of fields, ordered by the value of the field.
- Fundamentally, indexes in MongoDB are similar to indexes in other database systems.
- MongoDB defines indexes at the collection level and supports indexes on any field or sub-field of the documents.

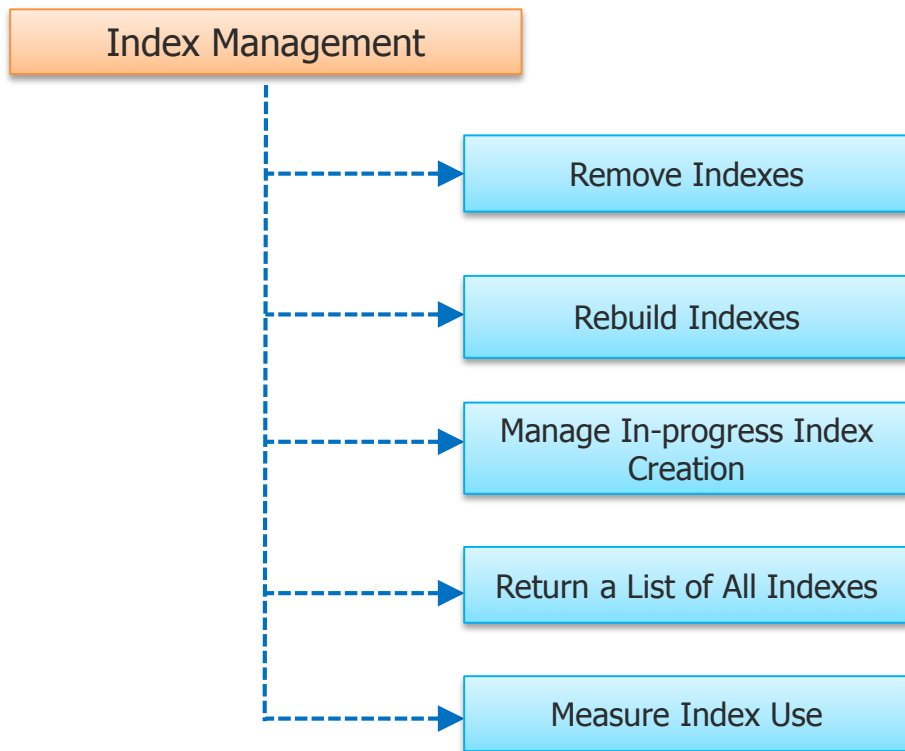


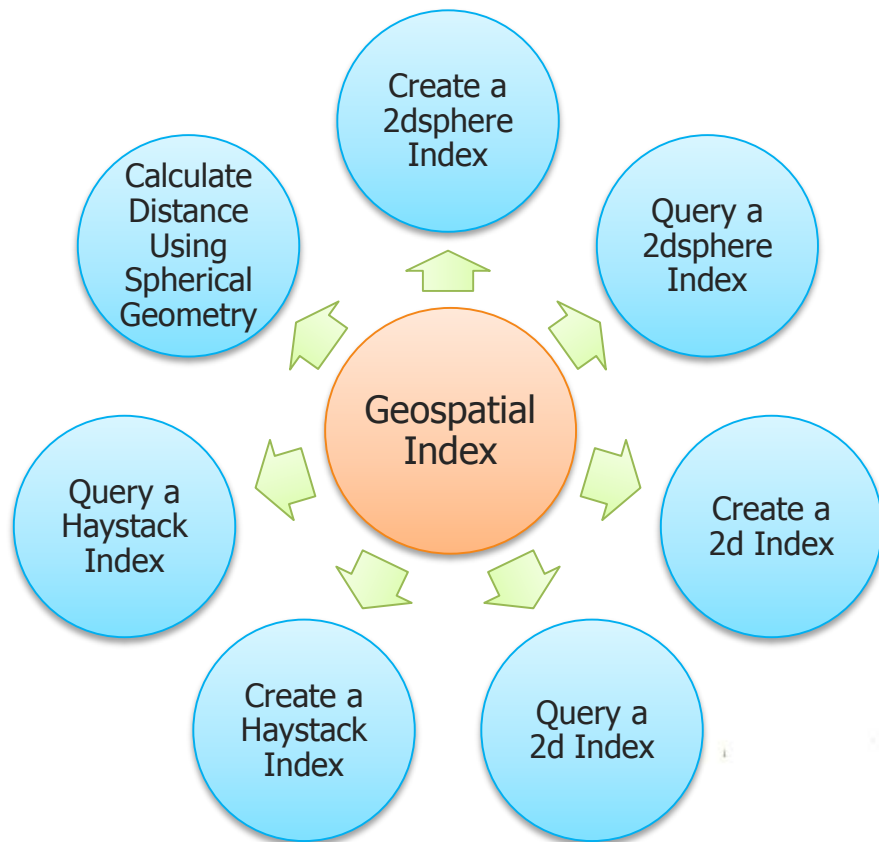


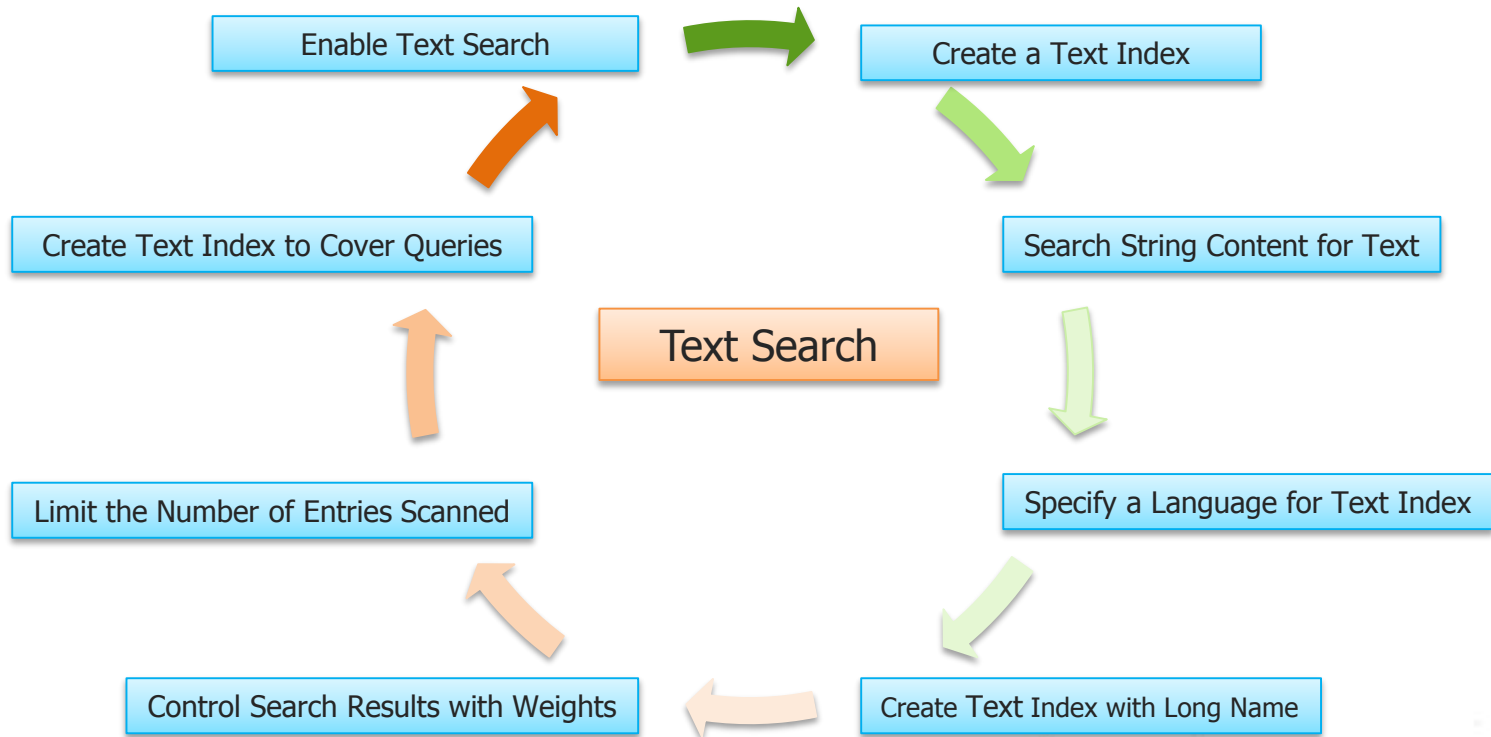


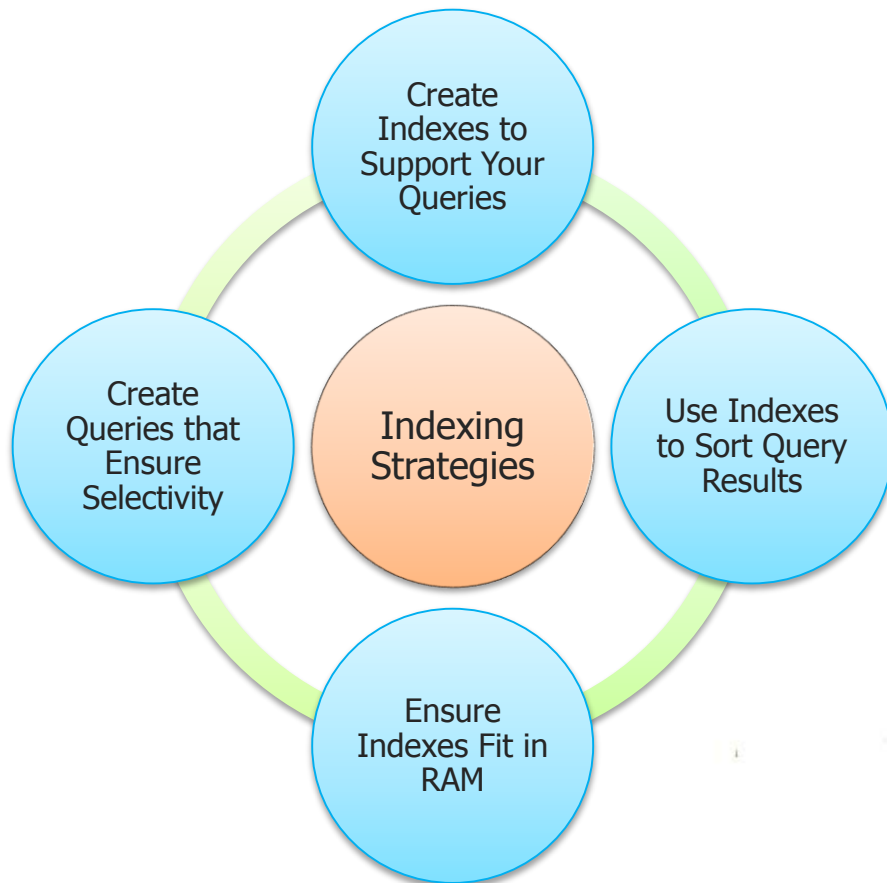


Index Creation Hands On











What is difference in single field and compound indexes?



Single field index will be there on only one field but on compound index it will be on more than one fields.



How can we create index on array?



Through Multikey Index.



Which Index do we create to support geospatial coordinate data?



Geospatial Index



What is sparse Index?



The sparse property of an index ensures that the index only contain entries for documents that have the indexed field.



What is TTL Index?



TTL indexes are special indexes that MongoDB can use to automatically remove documents from a collection after a certain amount of time.



Does TTL Indexes supports compound indexes?



No



How can we remove indexes?



`db.collection_Name.dropIndex({ "tax_id": 1 })` – pass the index key
Or
`Db.collection_name.dropIndex("tax_id_1")` – pass the index name



How can we rebuild indexes?



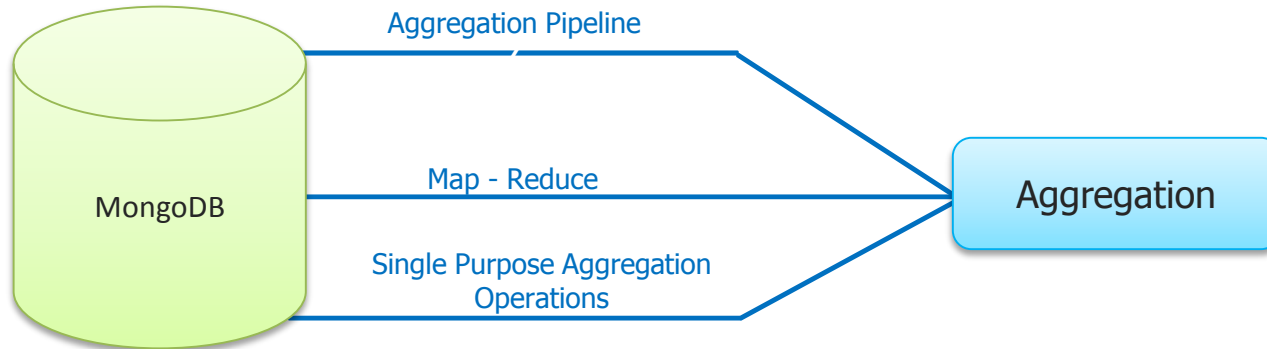
```
db.collection_Name.reIndex()
```

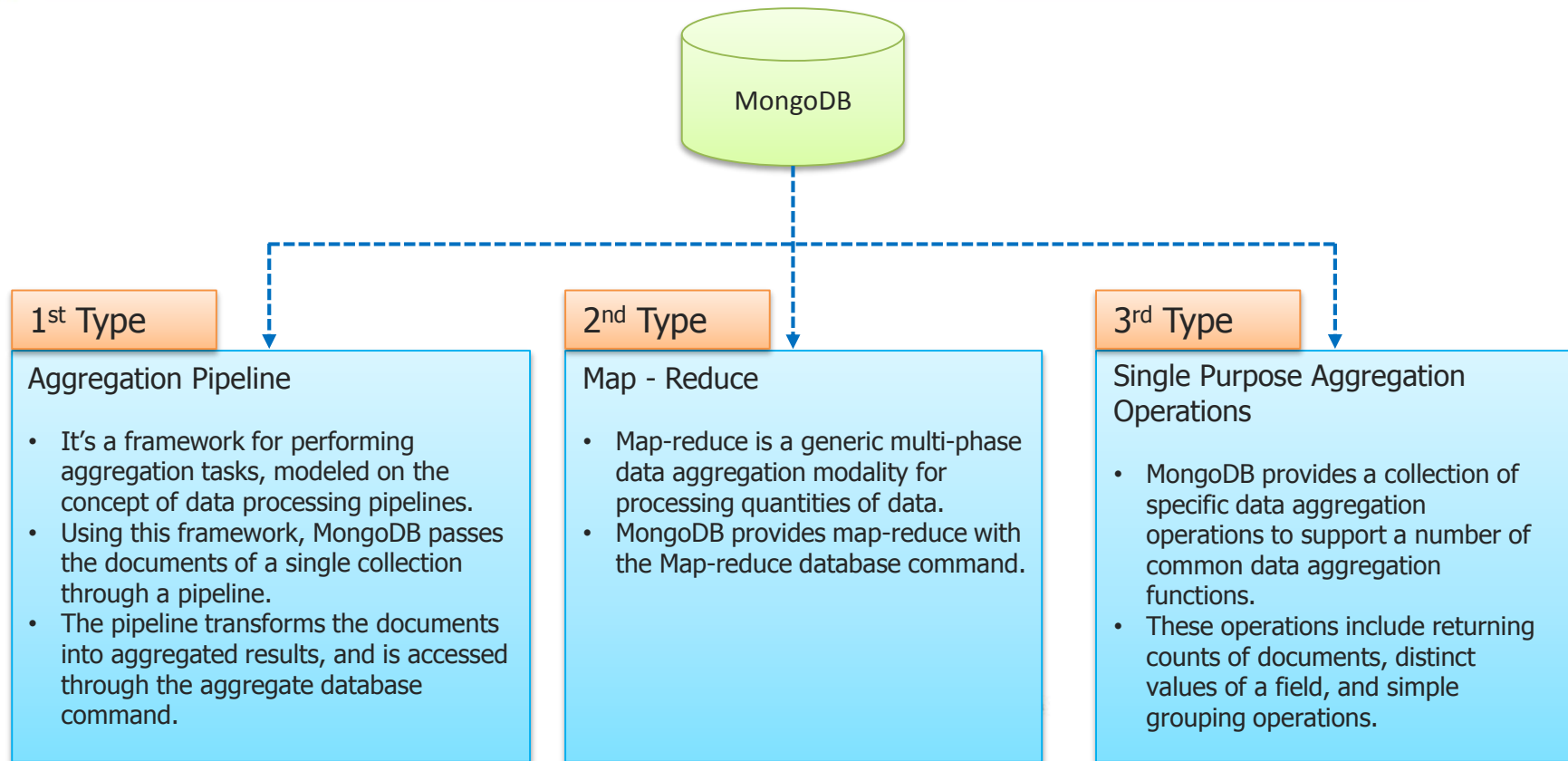
Indexing Tutorials

- Aggregations are operations that process data records and return computed results.
- MongoDB provides a rich set of aggregation operations that examine and perform calculations on the data sets.
- Running data aggregation on the mongod instance simplifies application code and limits resource requirements.
- Like queries, aggregation operations in MongoDB use collections of documents as an input and return results in the form of one or more documents.

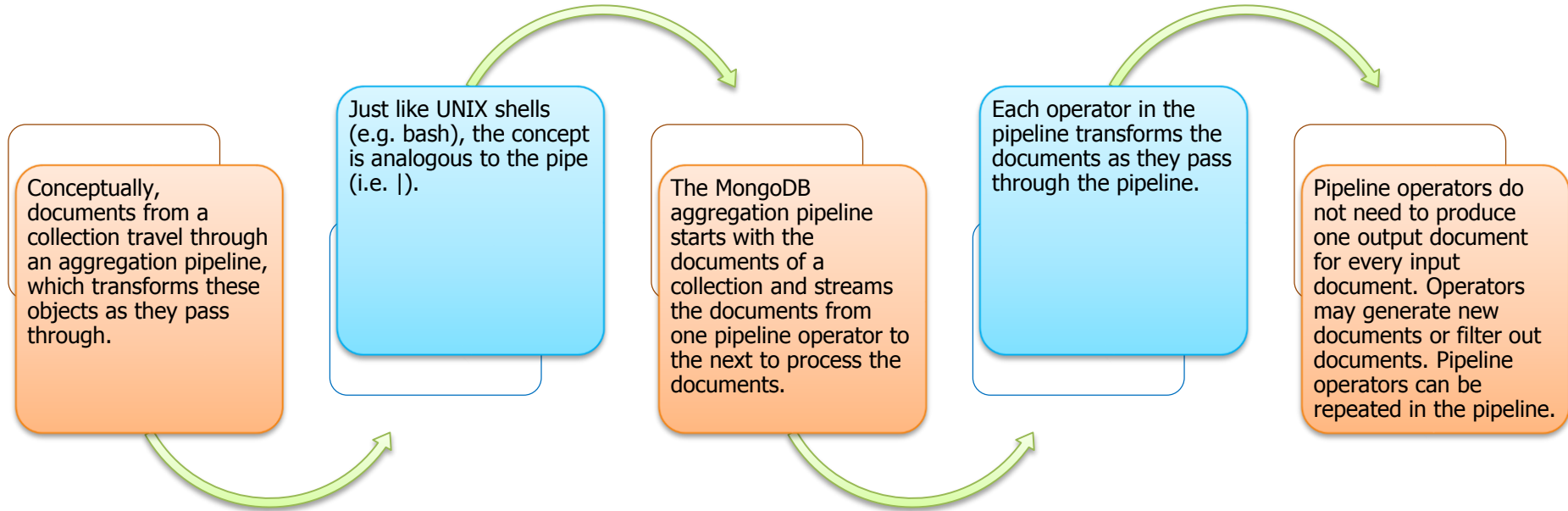
Introduction Aggregation (Contd.)

→ MongoDB provides the 3 approaches to aggregation, each with its own strengths and purposes for a given situation.






- The aggregation pipeline is a framework for data aggregation modelled on the concept of data processing pipelines.
- Documents enter a multi-stage pipeline that transforms the documents into an aggregated result.
- The aggregation pipeline provides an alternative to map-reduce and may be the preferred solution for many aggregation tasks where the complexity of map-reduce may be unwarranted.
- The aggregation pipeline has two phases: `$match` and `$group`.
- Aggregation pipeline have some limitations on value types and result size.



- Each pipeline operator takes a pipeline expression as its operand.
- Pipeline expressions specify the transformation to apply to the input documents.
- Expressions have a document structure and can contain fields, values, and operators.
- Pipeline expressions can only operate on the current document in the pipeline and cannot refer to data from other documents.
- Expression operations provide in-memory transformation of documents.

- In MongoDB, the aggregate command operates on a single collection, logically passing the entire collection into the aggregation pipeline.
- To optimize the operation, wherever possible, use the Pipeline Operators and Indexes to avoid scanning the entire collection.



- The \$match, \$sort, \$limit, and \$skip pipeline operators can take advantage of an index when they occur at the beginning of the pipeline before any of the following aggregation operators: \$project, \$unwind, and \$group.
-  New in version 2.4: The \$geoNear pipeline operator takes advantage of a geospatial index. When using \$geoNear, the \$geoNear pipeline operation must appear as the first stage in an aggregation pipeline.
- For un-sharded collections, when the aggregation pipeline only needs to access the indexed fields to fulfil its operations, an index can cover the pipeline.

Map-reduce is a data processing paradigm for condensing large volumes of data into useful aggregated results.

For map-reduce operations, MongoDB provides the mapReduce database command.

In this map-reduce operation, MongoDB applies the map phase to each input document (i.e. the documents in the collection that match the query condition).

The map function emits key-value pairs. For those keys that have multiple values, MongoDB applies the reduce phase, which collects and condenses the aggregated data.

MongoDB then stores the results in a collection. Optionally, the output of the reduce function may pass through a finalize function to further condense or process the results of the aggregation.

All map-reduce functions in MongoDB are JavaScript and run within the mongod process. Map-reduce operations take the documents of a single collection as the input and can perform any arbitrary sorting and limiting before beginning the map stage.

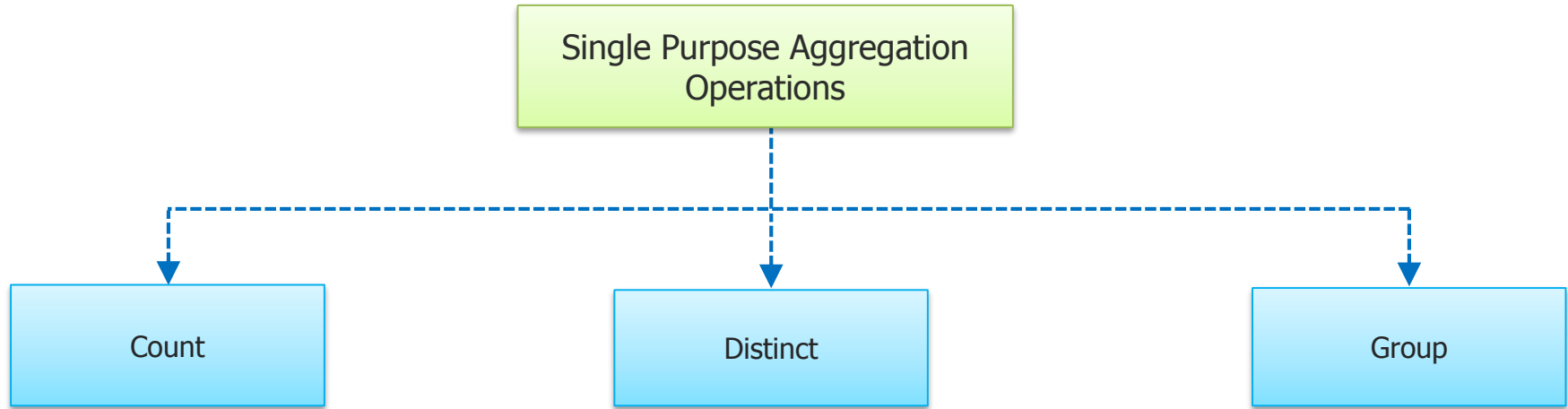
Map-reduce can return the results of a map-reduce operation as a document, or may write the results to collections.

The input and the output collections may be sharded.

- In MongoDB, map-reduce operations use custom JavaScript functions to map, or associate, values to a key.
- If a key has multiple values mapped to it, the operation reduces the values for the key to a single object.
- The use of custom JavaScript functions provide flexibility to map-reduce operations.
- For instance, when processing a document, the map function can create more than one key and value mapping or no mapping.
- Map-reduce operations can also use a custom JavaScript function to make final modifications to the results at the end of the map and reduce operation, such as perform additional calculations.

- In MongoDB, the map-reduce operation can write results to a collection or return the results inline.
- If you write map-reduce output to a collection, you can perform subsequent map-reduce operations on the same input collection that merge replace, merge, or reduce new results with previous results.
- When returning the results of a map reduce operation inline, the result documents must be within the BSON Document Size limit, which is currently 16 megabytes.
- MongoDB supports map-reduce operations on sharded collections. Map-reduce operations can also output the results to a sharded collection.

- Aggregation refers to a broad class of data manipulation operations that compute a result based on an input and a specific procedure.
- MongoDB provides a number of aggregation operations that perform specific aggregation operations on a set of data.
- Although limited in scope, particularly compared to the aggregation pipeline and map-reduce, these operations provide straightforward semantics for common data processing options.



Name	Description
\$addToSet	Returns an array of all the unique values for the selected field among for each document in that group.
\$first	Returns the first value in a group.
\$last	Returns the last value in a group.
\$max	Returns the highest value in a group.
\$min	Returns the lowest value in a group.
\$avg	Returns an average of all the values in a group.
\$push	Returns an array of all values for the selected field among for each document in that group.
\$sum	Returns the sum of all the values in a group.



How many type of aggregation is there in MongoDB?



3 Types :
Aggregation Pipeline
Map – Reduce
Single Purpose Aggregation



Does group aggregation support sharded cluster?



It does not support sharded collection.



What is Aggregation Pipeline?



The aggregation pipeline is a framework for data aggregation modelled on the concept of data processing pipelines.



Can we apply Aggregation Pipeline on multiple collection?



No, In MongoDB, the aggregate command operates on a single collection.



What is Map-Reduce in MongoDB?







Map-reduce is a data processing paradigm for condensing large volumes of data into useful aggregated results.

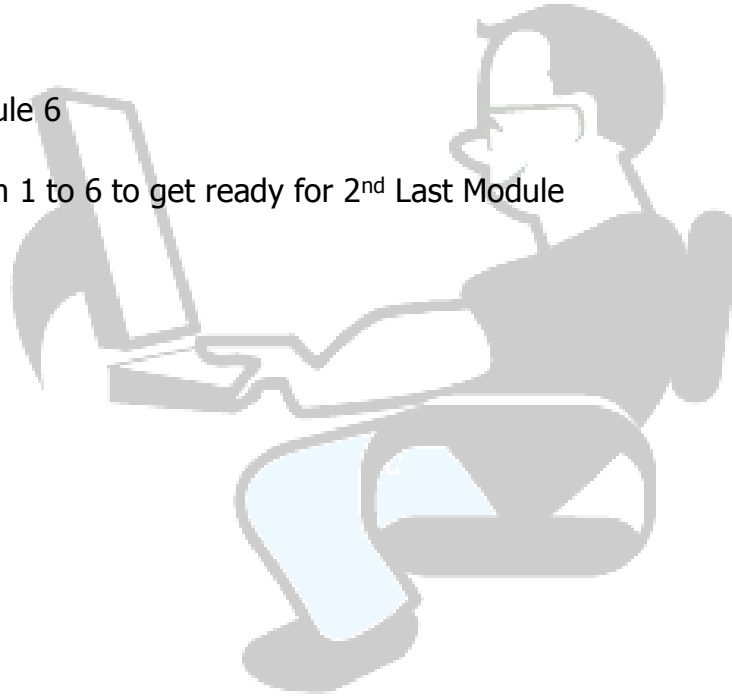
Aggregation Framework Hands On

Assignment

→ Attempt the assignment using document present in LMS.



-  Read Module 6 FAQ
-  Attempt Module 6 Quiz
-  Do all assignments for Module 6
-  Summarize all modules from 1 to 6 to get ready for 2nd Last Module



- MongoDB Package Components
- Configuration File Options
- MongoDB Limits and Thresholds
- Connection String URI Format
- API and Drivers for MongoDB
- MMS (MongoDB Monitoring Service)
- HTTP Interface
- Integration of Analytics and Reporting Tool with MongoDB
- MongoDB with Hadoop



Your feedback is important to us, be it a compliment, a suggestion or a complaint. It helps us to make the course better!

Please spare few seconds to take the survey after the webinar.

Thank you!

