Q1. Which are different aggregation commands and methods?

Ans. → Aggregation commands:

| | Name | Description |
|---|---|---|
| 1. | aggregate | Performs aggregate tasks such as $group using an aggregation pipeline |
| 2. | count | Counts the number of documents in a collection or a view |
| 3. | Distinct | Displays the distinct values found for a specified key in a collection or a view. |
| 4. | mapReduce | Performs map-reduce aggregation for large data sets. |

→ Aggregation methods:

| | Name | Description |
|---|---|---|
| 1. | db.collection.aggregate() | Provides access to the aggregation pipeline |
| 2. | db.collection.mapReduce() | Performs map-reduce aggregation for large data sets |

Q2. Enlist user-defined and system variables in aggregation?

Ans. → User-defined:

- User variable names can contain the ascii characters [_a-zA-Z0-9] and any non-ascii character.
- User variable names must begin with a lowercase ascii letter [a-z] or a non-ascii character.

→ System variables:

1. NOW
2. CLUSTER_TIME
3. ROOT
4. CURRENT
5. REMOVE
6. DESCEND
7. PRUNE
8. KEEP
9. SEARCH-META
10. USER_ROLES

Q3. ~~Explain~~ Describe SQL to aggregation mapping chart.

Ans. The following table provides an overview of common SQL ~~terms~~ aggregation terms, functions, and concepts and the corresponding MongoDB aggregation operators.

| | SQL terms, functions, concepts | MongoDB aggregation operators |
|---|---|---|
| 1. | WHERE | $match |
| 2. | GROUP BY | $group |
| 3. | HAVING | $match |
| 4. | SELECT | $project |
| 5. | ORDER BY | $sort |
| 6. | LIMIT | $limit |
| 7. | SUM() | $sum |
| 8. | COUNT() | $sum<br>~~$sort by cor~~ $sort By count |
| 9. | Select INTO NEW_TABLE | $out |
| 10. | MERGE INTO TABLE | $merge |
| 11. | UNION ALL | ~~$pro~~ $unionWith |

Q4. Explain indexing methods on the mongo shell.

Ans. The different indexing methods are:-

1. Creating an Index
   - MongoDB provides a method called createIndex() that allows users to create an index.
   - Syntax:
   → db.COLLECTION_NAME.createIndex({KEY:1})

2. Drop an Index
   - In order to drop an index, MongoDB provides the dropIndex() method.
   - Syntax:
   → db.NAME_OF_COLLECTION.dropIndex({KEY:1})

3. Get Description of all Indexes
   - The getIndexes() method of MongoDB gives a description of all the indexes that exists in the given collection.
   - Syntax:
   → db.NAME_OF_COLLECTION.getIndexes()

Q5. What is different option for indexing?

Ans. The different options for indexing are:-

1. Background:
   - Builds the index in the background so that building an index does not block other database activities.

2. Unique:

- Creates a unique index so that the collection will not accept insertion of documents where the index key or keys match an existing value in the index.

3. Sparse:
- If true, the index only references documents with the specified field.

4. expireAfterSeconds:
- Specifies a value, in seconds, as a TTL to control how long MongoDB retains documents in this collection.

Q6. Enlist different Pipeline operators, Expression operators and Comparison operators.

Ans. → Pipeline operators:

1. $project
2. $match
3. $redact
4. $limit
5. $skip
6. $unwind
7. $group
8. $sort
9. $geoNear
10. $out

→ Expression operators

1. $addToSet
2. $first
3. $last

4. $max
5. $min
6. $avg
7. $push

→ Comparison Operators:

1. $emp
2. $eq
3. $gt
4. $gte
5. $Lt
6. $Lte
7. $ne

Q7. What is use of Drop Duplicates option in ondexing?

Ans. In MongoDB, to drop duplicates from a collection, you can use the ~~aggr~~ dropDups option.

- Force MongoDB to create a unique index, use the db.collection.ensureIndex() method with the unique option set to true.

→ Syntax:

- db.collection.ensureIndex({a:1}, {unique:true})

→ steps

1. Identify and Remove duplicates:
- Write a script to find and delete duplicates based on your criteria.

2. Create unique index:
- Use the createIndex method with the unique option:

db.collection.createIndex({field: 1}, {unique: true});

Q3. Write method to return a list of all indexes on a collection and ~~data~~ databases.

Ans. Steps:

1. Open your mongo DB shell or Mongo DB client like Mongo DB Compass.
2. Use the following commands:

→ use your Database

db.yourCollection.getIndexes()

- This will return an array of documents, each representing an index, including details like the index_name, the fields indexed, and whether its unique.

→ db.yourcollection.getIndexes()

- Fetches and returns all indexes for the specified collection.