ASSIGNMENT ON AGGREGATION

Aggregation involves combining information from multiple data points into a single value. This is often done using aggregate functions such as count, sum, average, maximum, or minimum.

Examples:

* Counting the number of customers in a database.
* Calculating the average age of people in a city.
* Finding the highest temperature recorded in a weather dataset.

Aggregation in MongoDB is a powerful way to analyze and summarize your data by combining information from multiple documents into a single value or set of values. It allows you to perform calculations, group data, and extract meaningful insights from your collections.

Here's an overview of aggregation with MongoDB:

1. Aggregation Pipelines:

* Aggregation in MongoDB is performed through aggregation pipelines. These pipelines are sequences of stages that process documents in a step-by-step manner.
* Each stage in the pipeline performs a specific operation on the documents, such as filtering, grouping, calculating values, or modifying the document structure.
* You can combine multiple stages to achieve complex aggregations and extract deeper insights from your data.

2. Common Aggregation Stages:

* $match: Filters documents based on a specific condition.
* $group: Groups documents based on common values in a field and performs calculations on the grouped documents.
* $sort: Sorts the documents in a specific order.
* $project: Selects specific fields from the documents and transforms them.
* $sum: Calculates the sum of a field across all documents.
* $avg: Calculates the average of a field across all documents.
* $max: Finds the maximum value of a field across all documents.
* $min: Finds the minimum value of a field across all documents.
* Many other stages exist for specific operations like lookup, unwind, and accumulator.

3. Benefits of Aggregation:

* Efficiently analyze large datasets: Aggregation allows you to perform complex calculations on your data without retrieving all documents at once, making it efficient for large datasets.
* Gain insights from data: By grouping and calculating values, you can extract meaningful trends, patterns, and summary statistics from your data.
* Flexible and powerful: You can combine different stages in various ways to achieve complex aggregations and answer specific questions about your data.

4. Getting Started:

* To get started with aggregation, you can use the db.collection.aggregate() method in your MongoDB client.
* There are also various libraries and frameworks available for different programming languages that make working with aggregation pipelines easier.

5. Example:

Imagine you have a collection of "orders" documents, each containing information like product, price, and customer. Here's an example aggregation pipeline to calculate the total sales per product:

db.orders.aggregate([

{

$group: {

\_id: "$product",

totalSales: { $sum: "$price" }

}

}

])