|  |  |
| --- | --- |
| **$bucket:**  Suppose you have a collection called **sales** with documents containing **amount** field. You want to categorize the sales based on their amounts into buckets like "Low", "Medium", and "High". Here's how you can use **$bucket** for this:   |  | | --- | | db.sales.aggregate([  {  $bucket: {  groupBy: "$amount",  boundaries: [0, 100, 500, Infinity],  default: "Other",  output: {  "count": { $sum: 1 },  "totalAmount": { $sum: "$amount" }  }  }  }  ]) |   This query will create buckets based on the ranges defined in **boundaries** and output the count and total amount for each bucket. |
| **$lookup:** join  Let's say you have two collections, **orders** and **customers**. Each order document in the **orders** collection has a **customerId** field referencing documents in the **customers** collection. You want to join these two collections to fetch customer information along with their orders. Here's how you can use **$lookup**:   |  | | --- | | db.orders.aggregate([  {  $lookup: {  from: "customers",  localField: "customerId",  foreignField: "\_id",  as: "customer"  }  },  {  $unwind: "$customer"  }  ]) |   This query will match each order with its corresponding customer document based on the **customerId** field. |
| **$project:**  Suppose you have a collection named **students** with documents containing various fields like **name**, **age**, **gender**, and **grades**. You want to fetch only the **name** and **age** fields for further processing. Here's how you can use **$project**:   |  | | --- | | db.students.aggregate([  {  $project: {  name: 1,  age: 1  }  }  ]) |   This query will project only the **name** and **age** fields from each document in the **students** collection. |
|  |