|  |  |
| --- | --- |
| let's create some sample entries for the **sales** collection:   |  | | --- | | [  { "amount": 50 },  { "amount": 120 },  { "amount": 300 },  { "amount": 80 },  { "amount": 600 },  { "amount": 150 },  { "amount": 30 },  { "amount": 200 },  { "amount": 700 },  { "amount": 450 }  ] | |
| Now, let's run the aggregation query using **$bucket** to categorize sales into "Low", "Medium", and "High" buckets based on their amounts.   |  | | --- | | db.sales.aggregate([  {  $bucket: {  groupBy: "$amount",  boundaries: [0, 100, 300, Infinity],  default: "Other",  output: {  "count": { $sum: 1 },  "totalAmount": { $sum: "$amount" }  }  }  }  ]) | |
| let's create some sample entries for the **orders** and **customers** collections:   |  | | --- | | // Customers Collection  [  { "\_id": 1, "name": "John Doe", "email": "john@example.com" },  { "\_id": 2, "name": "Jane Smith", "email": "jane@example.com" },  { "\_id": 3, "name": "Alice Johnson", "email": "alice@example.com" }  ]  // Orders Collection  [  { "\_id": 101, "customerId": 1, "product": "Laptop", "quantity": 2 },  { "\_id": 102, "customerId": 1, "product": "Smartphone", "quantity": 1 },  { "\_id": 103, "customerId": 2, "product": "Tablet", "quantity": 1 },  { "\_id": 104, "customerId": 3, "product": "Headphones", "quantity": 2 },  { "\_id": 105, "customerId": 1, "product": "Monitor", "quantity": 1 }  ] | |
| In this sample data:   * The **orders** collection has orders with **customerId** referencing documents in the **customers** collection. * The **customers** collection contains information about customers.   Now, let's use **$lookup** to join these two collections and fetch customer information along with their orders:   |  | | --- | | db.orders.aggregate([  {  $lookup: {  from: "customers",  localField: "customerId",  foreignField: "\_id",  as: "customer"  }  },  {  $unwind: "$customer"  },  {  $project: {  "\_id": 1,  "product": 1,  "quantity": 1,  "customerName": "$customer.name",  "customerEmail": "$customer.email"  }  }  ]) |   This query will join the **orders** collection with the **customers** collection based on the **customerId** field and project the desired fields from both collections. |