**1. Count Documents**

db.orders.aggregate([{ $count: "total\_orders" }]);

**2. Sum a Field**

db.orders.aggregate([{ $group: { \_id: null, totalAmount: { $sum: "$amount" } } }]);

**3. Average a Field**

db.orders.aggregate([{ $group: { \_id: null, avgAmount: { $avg: "$amount" } } }]);

**4. Find Maximum Value**

db.orders.aggregate([{ $group: { \_id: null, maxAmount: { $max: "$amount" } } }]);

**5. Find Minimum Value**

db.orders.aggregate([{ $group: { \_id: null, minAmount: { $min: "$amount" } } }]);

**6. Group by Field**

db.orders.aggregate([{ $group: { \_id: "$customerId", totalSpent: { $sum: "$amount" } } }]);

**7. Filter with $match**

db.orders.aggregate([{ $match: { status: "shipped" } }]);

**8. Project Specific Fields**

db.orders.aggregate([{ $project: { customerId: 1, amount: 1, \_id: 0 } }]);

**9. Sort in Ascending Order**

db.orders.aggregate([{ $sort: { amount: 1 } }]);

**10. Sort in Descending Order**

db.orders.aggregate([{ $sort: { amount: -1 } }]);

**11. Limit Results**

db.orders.aggregate([{ $limit: 5 }]);

**12. Skip Results**

db.orders.aggregate([{ $skip: 5 }]);

**13. Lookup (Join Collections)**

db.orders.aggregate([

{ $lookup: { from: "customers", localField: "customerId", foreignField: "\_id", as: "customerDetails" } }

]);

**14. Unwind an Array**

db.orders.aggregate([{ $unwind: "$items" }]);

**15. Count Unique Values**

db.orders.aggregate([{ $group: { \_id: "$customerId" } }, { $count: "uniqueCustomers" }]);

**16. Add a New Field**

db.orders.aggregate([{ $addFields: { discountApplied: true } }]);

**17. Compute New Fields**

db.orders.aggregate([{ $project: { amountWithTax: { $multiply: ["$amount", 1.1] } } }]);

**18. Conditional Fields with $cond**

db.orders.aggregate([

{ $project: { amount: 1, highValue: { $cond: { if: { $gte: ["$amount", 100] }, then: true, else: false } } } }

]);

**19. Convert Date to String**

db.orders.aggregate([{ $project: { orderDate: { $dateToString: { format: "%Y-%m-%d", date: "$orderDate" } } } }]);

**20. Extract Year from Date**

db.orders.aggregate([{ $project: { year: { $year: "$orderDate" } } }]);

**21. Extract Month from Date**

db.orders.aggregate([{ $project: { month: { $month: "$orderDate" } } }]);

**22. Extract Day from Date**

db.orders.aggregate([{ $project: { day: { $dayOfMonth: "$orderDate" } } }]);

**23. Extract Hour from Date**

db.orders.aggregate([{ $project: { hour: { $hour: "$orderDate" } } }]);

**24. Text Search**

db.products.aggregate([{ $match: { $text: { $search: "laptop" } } }]);

**25. Find Documents Without a Field**

db.orders.aggregate([{ $match: { deliveryDate: { $exists: false } } }]);

**26. Find Documents with a Field**

db.orders.aggregate([{ $match: { deliveryDate: { $exists: true } } }]);

**27. Group by Month**

db.orders.aggregate([{ $group: { \_id: { month: { $month: "$orderDate" } }, totalSales: { $sum: "$amount" } } }]);

**28. Group by Year**

db.orders.aggregate([{ $group: { \_id: { year: { $year: "$orderDate" } }, totalSales: { $sum: "$amount" } } }]);

**29. Match and Group Together**

db.orders.aggregate([

{ $match: { status: "shipped" } },

{ $group: { \_id: "$customerId", totalSpent: { $sum: "$amount" } } }

]);

**30. Bucket Documents**

db.orders.aggregate([

{

$bucket: {

groupBy: "$amount",

boundaries: [0, 50, 100, 500, 1000],

default: "other",

output: { count: { $sum: 1 } }

}

}

]);

**31. Bucket Documents by Date**

db.orders.aggregate([

{

$bucketAuto: {

groupBy: "$orderDate",

buckets: 5

}

}

]);

**32. Find Top N per Group**

db.orders.aggregate([

{ $sort: { amount: -1 } },

{ $group: { \_id: "$customerId", topOrder: { $first: "$$ROOT" } } }

]);

**33. Merge Aggregation Output to Another Collection**

db.orders.aggregate([{ $group: { \_id: "$customerId", totalSpent: { $sum: "$amount" } } }, { $merge: "customer\_spending" }]);

**34. Replace Root**

db.orders.aggregate([{ $replaceRoot: { newRoot: "$customerDetails" } }]);

**35. Set Operation (Union)**

db.collection1.aggregate([{ $unionWith: { coll: "collection2" } }]);

**36. Convert Object to Array**

db.orders.aggregate([{ $project: { itemsArray: { $objectToArray: "$items" } } }]);

**37. Find Distinct Values**

db.orders.aggregate([{ $group: { \_id: "$status" } }]);

**38. Find Top 3 by Amount**

db.orders.aggregate([{ $sort: { amount: -1 } }, { $limit: 3 }]);

**39. Filter Nested Arrays**

db.orders.aggregate([{ $project: { expensiveItems: { $filter: { input: "$items", as: "item", cond: { $gt: ["$$item.price", 100] } } } } }]);

**40. Add an Array Index**

db.orders.aggregate([{ $project: { items: { $arrayElemAt: ["$items", 0] } } }]);

**41. Conditional Aggregation**

db.orders.aggregate([{ $group: { \_id: "$status", count: { $sum: 1 }, totalAmount: { $sum: "$amount" } } }]);

**42. Regex Match**

db.orders.aggregate([{ $match: { customerName: { $regex: "^J", $options: "i" } } }]);

**43. Replace String**

db.orders.aggregate([{ $project: { newStatus: { $replaceAll: { input: "$status", find: "pending", replacement: "in-progress" } } } }]);

**44. Sort by Multiple Fields**

db.orders.aggregate([{ $sort: { status: 1, amount: -1 } }]);

**45. Merge Arrays**

db.orders.aggregate([{ $project: { allItems: { $concatArrays: ["$items", "$extraItems"] } } }]);