```
// use("new_db");

// 1. display total number of orders placed by each customer
db.orders.aggregate([
    { $group: { _id: "$customerName", totalOrders: { $sum: 1 } } },
    { $project: { _id: 0, customerName: "$_id", totalOrders: 1 } }
]);

// 2. calculate average order value for each customer
db.orders.aggregate([
    { $group: { _id: "$customerName", Avg: { $avg: "$sales" } } },
    { $project: { customerName: "$_id", Avg: 1 } }
]);

// 3. display total revenue and profit generated by each product
db.orders.aggregate([
    { $group: { _id: "$productId", totalRevenue: { $sum: "$sales" }, totalProfit:
{ $sum: "$profit" } } }
])

// 4. identify the products which were sold more(more quantity)
db.orders.aggregate([
    // in grouping $productName can also be used to display product name(so using
both combinedly is best)
    { $group: { _id: "$productId", totalQty: { $sum: "$qty" } } },
    { $sort: { totalQty: -1 } }
]);

// 5. List the total number of products(means total quantity ) bought from
different regions
db.orders.aggregate([
    { $group: { _id: "$region", tot: { $sum: "$qty" } } }
]);

// 6. what is the average profit from east region from corportate segment
customers
db.orders.aggregate([
    {
        $match: {
            "region:east",
            "segment": "corportate"
        }
    },
    { $group: { _id: null, avgProfit: { $avg: "$profit" } } }
]);

// 7. List the details of a product its category and subcategory which has been
bought by maximum customers
db.orders.aggregate([
    {
        $group: {
            _id: {
```

```
                product: "$productId",
                name: "$productName",
                category: "$category",
                subcategory: "$subcategory"
            },
            customers: { $addToSet: { "$customerName"} }
        }
    },
    {
        $project: {
            _id: 0,
            productId: "$_id.productId",
            productName: "$_id.productName",
            category: "$_id.category",
            subcategory: "$_id.subcategory",
            customerCount: { $size: "$customers" }
        }
    },
    { $sort: { cnt: -1 } },
    { $limit: 1 }
]);

// 8. which region and segment are contributing to minimum profit
db.orders.aggregate([
    {
        $group: {
            _id: {
                region: "$region",
                segment: "$segment"
            },
            totalProfit: { $sum: "$profit" }
        }
    },
    { $sort: { totalProfit: 1 } },
    { $limit: 1 }
]);

// 9. display customer name and cities from which they have placed orders
db.orders.aggregate([
    // if duplicate cities
    {
        $group: {
            _id: "$customerName",
            cities: { $addToSet: "$city" } // if duplicates are allowed no need to
group
        }
    },
    { $project: { _id: 0, customerName: "$_id" cities: 1 } }
]);

// 10. select 1st 10 customer names,region,and sales sort region in ascending
order, and sales in descending order
db.orders.aggregate([
    {
```

```
        $project: {
            _id: 0,
            customerName: 1,
            region: 1,
            sales: 1
        }
    },
    {
        $sort: {
            region: 1,   // ascending
            sales: -1    // descending
        }
    },
    { $limit: 10 }
]);

// 11. display customer name of customers who have opted for 'L' prority
db.orders.aggregate([
    {
        $match: {
            priority: "L"
        }
    },
    {
        $project: {
            _id: 0,
            customerName: 1
        }
    }
]);

// 12. display customer name,region and city of customers who have not opted for
prority
db.orders.aggregate([
    {
        $match: {
            $or: [
                { priority: null },
                { priority: { $exists: false } },
                { priority: "" }
            ]
        }
    },
    {
        $project: {
            _id: 0,
            customerName: 1,
            region: 1,
            city: 1
        }
    }
]);

// 13. display the number of customers who have opted for H priroty and color is
```

```
red
db.orders.aggregate([
    {
        $match: {
            priority: "H",
            color: "Red"
        }
    },
    {
        $group: {
            _id: "$customerName"
        }
    },
    {
        $count: "numCustomers"
    }
]);

// 14. display number of unique orders placed
db.orders.aggregate([
    { $group: { _id: "$orderId" } },
    { $count: "SumOfUnique" }
]);
// or
db.orders.aggregate([
    {
        $group: {
            _id: null,
            uniqueOrders: { $addToSet: "$orderId" }
        }
    },
    {
        $project: {
            _id: 0,
            numOrders: { $size: "$uniqueOrders" }
        }
    }
]);

// 15. display customer name of all customers whose name starts with letter A
irrespective of case
db.orders.aggregate([
    {
        $match: {
            customerName: { $regex: /^A/i }  // ^A → starts with A, i → case-
insensitive
            // customerName:{$regex:"A",$options:"i"}
        }
    },
    {
        $project: {
            _id: 0,
            customerName: 1
        }
```

```
    }
]);

// 16. display customer name of all customers who are from texas or new york city
db.orders.aggregate([
    {
        $match: {
            "city": { $in: ["Texas", "New York"] }
        }
    },
    { $project: { _id: 0, customerName: 1 } }
])

// 17. display product id where quantity is between 2 and 5
db.orders.aggregate([
    {
        $match: {
            qty: { $gte: 2 , $lte: 5 }
        }
    },
    {
        $project: {
            _id: 0,
            customerName: 1
        }
    }
]);

// 18. how many orders have the product with all red,black & silver colors
db.orders.aggregate([
    {
        $match: { color: { $in: ["Red", "Black", "Silver"] } }
    },
    {
        $group: {
            _id: "$orderId",
            colorsInOrder: { $addToSet: "$color" }
        }
    },
    {
        $match: { colorsInOrder: { $all: ["Red", "Black", "Silver"] } }
    },
    {
        $project: {
            _id: 0,
            orderId: "$_id",
            colors: "$colorsInOrder"
        }
    },
    {
        $count: "totalOrders"
    }
]);
```

```
// 19. display the number of product names under home office segment

// if we want total product names(including duplicates -- no need to group)
// else group (if they ask unique product names )
db.orders.aggregate([
    { $match: { segment: "Home Office" } },
    { $group: { _id: "$productName" } },   // group by unique product names
    { $count: "numProducts" }
]);

// | Question Type | Use`$group` ? |
// | ---------------------------------------------- | ------------- |
// | Count total rows / sum / avg | No |
// | Count unique values(customers, products, etc.) | Yes |
// | Remove duplicates or collect items in array | Yes |

// 20. display total quantity of subcategory mountain bikes
db.orders.aggregate([
    {
        $match: {
            "subcategory": "mountain bikes"
        }
    },
    {
        $group: {
            _id: null,
            totalQty: "$qty"
        }
    },
    {
        $project: {
            _id: 0,
            totalQty: 1
        }
    }
]);
```