1. Find the total revenue (price × quantity) for each item, sorted from highest to lowest.

```
db.sales.aggregate([
 {
  $project: {
   _id: 1,
   item: 1,
   revenue: { $multiply: ["$price", "$quantity"] }
  } },  {  $sort: { revenue: -1 }
 }])
```

2. Calculate the total quantity sold per month in 2022.

```
db.sales.aggregate([
 {
  $match: {
   date: {
    $gte: ISODate("2022-01-01T00:00:00Z"),
    $lt: ISODate("2023-01-01T00:00:00Z")
   }}  }, { $group: {
   _id: { $month: "$date" },
   totalQuantity: { $sum: "$quantity" },
   month: { $first: { $month: "$date" } }
  }  }, {
  $project: {
   _id: 0,
   month: 1,
   totalQuantity: 1
  } },{
  $sort: { month: 1 }
 }])
```

3. Find all items where price is greater than 10 and size is not 'Short'.

```
db.sales.find({
  price: { $gt: 10 },
  size: { $ne: "Short" }
})
```

4. Get all Cappuccino sales with quantity between 10 and 20.

```
db.sales.find({
  item: "Cappuccino",
  quantity: { $gte: 10, $lte: 20 }
})
```

5. Query to find items where the item name starts with "A".

```
db.sales.find({
  item: /^A/
})
```

6. Find all records that do not have the field size.

```
db.sales.find({
  size: { $exists: false }
})
```

7. List all items sold in February 2022.

```
db.sales.distinct("item", {
  date: {  $gte: ISODate("2022-02-01T00:00:00Z"),
    $lt: ISODate("2022-03-01T00:00:00Z")}})
```

8. Find all sales that are either "Grande" or "Tall" but not "Americanos".

```
db.sales.find({
  $and: [
    { size: { $in: ["Grande", "Tall"] } },
    { item: { $ne: "Americanos" } }
  ]})
```

9. Find sales where the quantity is more than twice the price.

```
db.sales.find({
  $where: "this.quantity > (2 * this.price)"
})
```

10. Find all sales where the price is greater than the average price of their respective size.

```
db.sales.aggregate([
  {
    $group: {
      _id: "$size",
      avgPrice: { $avg: "$price" },
      docs: { $push: "$$ROOT" } } },
  {  $unwind: "$docs" },  {
    $replaceRoot: {
      newRoot: {$mergeObjects: [
          "$docs",   { avgPrice: "$avgPrice" } ]  } } },
  {  $match: {
      $expr: { $gt: ["$price", "$avgPrice"] }   }
  },{ $project: { avgPrice: 0 }  }])
```

11. Filter sales where the total revenue is even and exceeds 100.

```
db.sales.find({
  $where: function() {
    const revenue = this.price * this.quantity;
    return revenue > 100 && revenue % 2 === 0;
  }})
```

11. Find Sales Where the Day of Week Matches Quantity's Last Digit.

```
db.sales.find({
  $where: function() {
    const date = this.date;
    const day = date.getDay();
    const lastDigit = this.quantity % 10;
    return day === lastDigit;
  }
})
```

12. Find Sales Where the Month is Prime and Quantity is Odd .

```
db.sales.find({
  $where: function() {
    const primeMonths = [2, 3, 5, 7, 11];
    const month = this.date.getMonth() + 1
    return primeMonths.includes(month) && this.quantity % 2 === 1;  }})
```

13. Find Sales with "Suspicious Quantities" (Divisible by 5 or 7) .

```
db.sales.find({
  $or: [
    { quantity: { $mod: [5, 0] } },
    { quantity: { $mod: [7, 0] } }  ]})
```