Anand balachandran 22BCE7494

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

db.sales.aggregate([

{

$group: {

\_id: "$item",

totalRevenue: {

$sum: { $multiply: ["$price", "$quantity"] }

}

}

},

{

$sort: { totalRevenue: -1 }

  }

]);  
  
  
2.Calculate the total quantity sold per month in 2022.  
  
  
db.sales.aggregate([

{

$match: {

date: {

$gte: ISODate("2022-01-01"),

$lt: ISODate("2023-01-01")

}

}

},

{

$group: {

\_id: { $dateToString: { format: "%Y-%m", date: "$date" } },

totalQuantity: { $sum: "$quantity" }

}

},

{ $sort: { \_id: 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. Get all Cappuccino sales with quantity between 10 and 20.  
  
  
db.sales.find({

item: { $regex: /^A/, $options: "i" }

});

6. Find all records that do not have the field size.  
  
db.sales.find({

size: { $exists: false }

});  
  
6. List all items sold in February 2022.

db.sales.find({

date: {

$gte: ISODate("2022-02-01"),

$lt: ISODate("2022-03-01")

}

});  
  
  
7. Find all sales that are either "Grande" or "Tall" but not "Americanos".

db.sales.find({

size: { $in: ["Grande", "Tall"] },

item: { $ne: "Americanos" }

});  
  
  
8. List all items sold in February 2022.

db.sales.find(

{

date: {

$gte: ISODate("2022-02-01"),

$lt: ISODate("2022-03-01")

}

},

{

item: 1, \_id: 0

}

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

db.sales.aggregate([

{

$match: {

$expr: { $gt: ["$quantity", { $multiply: ["$price", 2] }] }

    }

  }

]);  
  
  
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" }

}

},

{

$lookup: {

from: "sales",

localField: "\_id",

foreignField: "size",

as: "salesData"

}

},

{ $unwind: "$salesData" },

{

$match: {

$expr: { $gt: ["$salesData.price", "$avgPrice"] }

}

},

{

$replaceRoot: { newRoot: "$salesData" }

}

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

db.sales.find({ $where: function() { const total = this.price \* this.quantity; return total > 100 && total % 2 === 0; }})  
  
  
11. Find Sales Where the Day of Week Matches Quantity's Last Digit [Filter sales where the day of the week (0=Sunday, 1=Monday, etc.) matches the last digit of quantity]  
  
  
db.sales.find({

$where: function() {

const dayOfWeek = this.date.getDay(); // 0 = Sunday, ..., 6 = Saturday

const lastDigit = this.quantity % 10;

return dayOfWeek === lastDigit;

  }

});  
  
  
12. Find Sales Where the Month is Prime and Quantity is Odd [Filter sales where the month (1-12) is a prime number (2,3,5,7,11) AND quantity is odd]  
  
db.sales.find({

$where: function() {

const month = this.date.getMonth() + 1; // getMonth() returns 0-11, so +1 for 1-12

const primeMonths = [2, 3, 5, 7, 11];

const isPrimeMonth = primeMonths.includes(month);

const isOddQuantity = (this.quantity % 2) === 1;

return isPrimeMonth && isOddQuantity;

}

})  
  
  
13. Find Sales with "Suspicious Quantities" (Divisible by 5 or 7) [Filter sales where quantity is divisible by 5 or 7]  
  
db.sales.find({

$expr: {

$or: [

{ $eq: [{ $mod: ["$quantity", 5] }, 0] },

{ $eq: [{ $mod: ["$quantity", 7] }, 0] }

]

}

})