Name : Harsh Chugh Reg.No. : 23bce11666

1. Find the total revenue (price × quantity) for each item, sorted from highest to lowest.
2. Calculate the total quantity sold per month in 2022.
3. Find all items where price is greater than 10 and size is not 'Short'.
4. Get all Cappuccino sales with quantity between 10 and 20
5. Query to find items where the item name starts with "A".
6. Find all records that do not have the field size.
7. Find all sales that are either "Grande" or "Tall" but not "Americanos".
8. List all items sold in February 2022.
9. Find sales where the quantity is more than twice the price
10. Find all sales where the price is greater than the average price of their respective size.
11. . Filter sales where the total revenue is even and exceeds 100.
12. 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]
13. . 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]

A1

db.sales.aggregate([

{

$group: {

\_id: "$item",

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

}

},

{ $sort: { totalRevenue: -1 } }

])

A2

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" }

}

},

{ $sort: { \_id: 1 } }

])

A3.

db.sales.find({

price: { $gt: 10 },

size: { $ne: "Short" }

})

A4.

db.sales.find({

item: "Cappuccino",

quantity: { $gte: 10, $lte: 20 }

})

A5.

db.sales.find({

item: { $regex: /^A/ }

})

A6.

db.sales.find({

size: { $exists: false }

})

A7.

db.sales.find({

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

item: { $ne: "Americanos" }

})

A8.

db.sales.find({

date: {

$gte: ISODate("2022-02-01T00:00:00Z"),

$lt: ISODate("2022-03-01T00:00:00Z")

}

}, {

item: 1,

\_id: 0

})

A9..

db.sales.find({

$where: function() {

return this.quantity > this.price \* 2;

}

})

A10.

db.sales.find({

$where: function() {

const avgMap = {

"Short": 5.5,

"Grande": 11.67,

"Tall": 16.67

};

return this.price > avgMap[this.size];

}

})

A11.

db.sales.find({

$where: function() {

const revenue = this.price \* this.quantity;

return revenue > 100 && revenue % 2 === 0;

}

})

A12.

db.sales.find({

$where: function () {

const dayOfWeek = this.date.getDay();

const lastDigit = this.quantity % 10;

return dayOfWeek === lastDigit;

}

})

A13

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;

}

})

A14. Find Sales with "Suspicious Quantities" (Divisible by 5 or 7) [Filter sales where quantity is divisible by 5 or 7]

db.sales.find({

$where: function () {

return this.quantity % 5 === 0 || this.quantity % 7 === 0;

}

})