# MONGO DB TEST (06.06.25)

db.sales.insertMany([

{ "\_id" : 1, "item" : "Americanos", "price" : 5, "size": "Short", "quantity" : 22, "date" : ISODate("2022-01-15T08:00:00Z") },

{ "\_id" : 2, "item" : "Cappuccino", "price" : 6, "size": "Short","quantity" : 12, "date" : ISODate("2022-01-16T09:00:00Z") },

{ "\_id" : 3, "item" : "Lattes", "price" : 15, "size": "Grande","quantity" : 25, "date" : ISODate("2022-01-16T09:05:00Z") },

{ "\_id" : 4, "item" : "Mochas", "price" : 25,"size": "Tall", "quantity" : 11, "date" : ISODate("2022-02-17T08:00:00Z") },

{ "\_id" : 5, "item" : "Americanos", "price" : 10, "size": "Grande","quantity" : 12, "date" : ISODate("2022-02-18T21:06:00Z") },

{ "\_id" : 6, "item" : "Cappuccino", "price" : 7, "size": "Tall","quantity" : 20, "date" : ISODate("2022-02-20T10:07:00Z") },

{ "\_id" : 7, "item" : "Lattes", "price" : 25,"size": "Tall", "quantity" : 30, "date" : ISODate("2022-02-21T10:08:00Z") },

{ "\_id" : 8, "item" : "Americanos", "price" : 10, "size": "Grande","quantity" : 21, "date" : ISODate("2022-02-22T14:09:00Z") },

{ "\_id" : 9, "item" : "Cappuccino", "price" : 10, "size": "Grande","quantity" : 17, "date" : ISODate("2022-02-23T14:09:00Z") },

{ "\_id" : 10, "item" : "Americanos", "price" : 8, "size": "Tall","quantity" : 15, "date" : ISODate("2022-02-25T14:09:00Z")}

]);

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}  
}  
]);

**//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: { $regex: /^A/ }  
});

**//6. List all items sold in February 2022.**

db.sales.find({  
 date: {  
 $gte: ISODate("2022-02-01T00:00:00Z"),  
 $lt: ISODate("2022-03-01T00:00:00Z")  
 }  
}, {  
 item: 1,  
 \_id: 0  
});

**// 7. Find all sales that are either "Grande" or "Tall" but not "Americanos".**

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

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

db.sales.find({  
 $expr: {  
 $gt: ["$quantity", { $multiply: [2, "$price"] }]  
 }  
});

**//8.Find all records that do not have the field size.**

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

**//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"}}}])

**// 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.**

db.sales.find({  
$where: function() {  
const dayoftheweek = this.date.getDay();  
const lastdigit = this.quantity % 10;  
return dayoftheweek == 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 = 1+ this.date.getMonth();  
const isPrime = [2,3,5,7,11].includes(month);  
const isOdd = this.quantity % 2 ==1;  
return isPrime && isOdd; }})

**//13. 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)}})