// Create a collection called 'orders'

db.createCollection("orders");

// Insert sample data into the 'orders' collection

db.orders.insertMany([

{ \_id: 0, name: "Pepperoni", size: "small", price: 19, quantity: 10, date: ISODate("2021-03-13T08:14:30Z") },

{ \_id: 1, name: "Pepperoni", size: "medium", price: 20, quantity: 20, date: ISODate("2021-03-13T09:13:24Z") },

{ \_id: 2, name: "Pepperoni", size: "large", price: 21, quantity: 30, date: ISODate("2021-03-17T09:22:12Z") },

{ \_id: 3, name: "Cheese", size: "small", price: 12, quantity: 15, date: ISODate("2021-03-13T11:21:39.736Z") },

{ \_id: 4, name: "Cheese", size: "medium", price: 13, quantity: 50, date: ISODate("2022-01-12T21:23:13.331Z") },

{ \_id: 5, name: "Cheese", size: "large", price: 14, quantity: 10, date: ISODate("2022-01-12T05:08:13Z") },

{ \_id: 6, name: "Vegan", size: "small", price: 17, quantity: 10, date: ISODate("2021-01-13T05:08:13Z") },

{ \_id: 7, name: "Vegan", size: "medium", price: 18, quantity: 10, date: ISODate("2021-01-13T05:10:13Z") }

]);

// Find all orders in the 'orders' collection

db.orders.find();

// Aggregate orders based on 'size' = 'medium' and group by 'name' with total quantity

db.orders.aggregate([

{ $match: { size: "medium" } },

{ $group: { \_id: "$name", totalQuantity: { $sum: "$quantity" } } }

]);

// Aggregate orders based on 'size' = 'medium' and count the total number of orders (documents)

db.orders.aggregate([

{ $match: { size: "medium" } },

{ $group: { \_id: "$name", totalCount: { $sum: 1 } } }

]);

// Count the total number of orders for each pizza type (group by 'name')

db.orders.aggregate([

{ $group: { \_id: "$name", pizzaCount: { $sum: 1 } } }

]);

// Calculate the average price of all pizzas

db.orders.aggregate([

{ $group: { \_id: "PIZZA", averagePrice: { $avg: "$price" } } }

]);

// Calculate the average price of pizzas and sort by average price in ascending order

db.orders.aggregate([

{ $group: { \_id: "PIZZAs", averagePrice: { $avg: "$price" } } },

{ $sort: { averagePrice: 1 } } // Corrected sort method

]);

// Calculate the average price based on size and sort by average price in descending order

db.orders.aggregate([

{ $group: { \_id: "$size", averagePrice: { $avg: "$price" } } },

{ $sort: { averagePrice: -1 } } // Corrected sort method

]);

// Calculate the average price based on size (ascending order of average price)

db.orders.aggregate([

{ $group: { \_id: "$size", averagePrice: { $avg: "$price" } } },

{ $sort: { averagePrice: 1 } } // Corrected sort method

]);