// Step 1: Switch to or Create the Database

use booksdb

// Step 2: Create the Collection and Insert Dummy Data

db.books.insertMany([

{

title: 'The Psychology of Money',

author: 'Morgan Housel',

genre: 'Fiction',

price: 15.49,

published\_date: ISODate('2020-09-08T00:00:00.000Z')

},

{

title: 'Atomic Habits',

author: 'James Clear',

genre: 'Fiction',

price: 16.99,

published\_date: ISODate('2018-10-16T00:00:00.000Z')

},

{

title: 'mrityunjaya',

author: 'shivaji sawant',

genre: 'fiction',

price: 1000,

published\_date: ISODate('2004-07-07T18:30:00.000Z')

},

{

title: 'Musafir',

author: 'Achyut Godbole',

genre: 'Fiction',

price: 12.99,

published\_date: ISODate('1998-03-12T00:00:00.000Z')

},

{

title: 'Rich Dad Poor Dad',

author: 'Robert T. Kiyosaki',

genre: 'Fiction',

price: 14.99,

published\_date: ISODate('1997-04-01T00:00:00.000Z')

},

{

title: 'yugandhar',

author: 'shivaji sawant',

genre: 'fiction',

price: 198,

published\_date: ISODate('1989-12-31T18:30:00.000Z')

},

{

title: 'chaava',

author: 'shivaji sawant',

genre: 'fiction',

price: 890,

published\_date: ISODate('1968-02-04T18:30:00.000Z')

},

{

title: 'Mazi Janmthep',

author: 'V.D. Savarkar',

genre: 'Fiction',

price: 10.49,

published\_date: ISODate('1930-06-05T00:00:00.000Z')

},

{

title: 'Hindutva',

author: 'V.D. Savarkar',

genre: 'Fiction',

price: 8.99,

published\_date: ISODate('1923-02-01T00:00:00.000Z')

}

])

// Task 1: Find the Average Price of All Books

db.books.aggregate([

{

$group: {

\_id: null,

avgprice: { $avg: "$price" }

}

}

])

// Task 2: Find the Count of Books in Each Genre

db.books.aggregate([

{

$group: {

\_id: "$genre",

count: { $sum: 1 }

}

}

])

// Task 3: For Each Genre, Find the Most Expensive Book

db.books.aggregate([

{

$group: {

\_id: "$genre",

expensive: { $max: "$price" }

}

}

])

// Task 4: Find the Authors Who Have Written More Than 3 Books

db.books.aggregate([

{

$group: {

\_id: "$author",

bookCount: { $sum: 1 }

}

},

{

$match: {

bookCount: { $gt: 3 }

}

}

])

// Task 5: Sort the Books by Published Date in Descending Order, and Then by Price in Ascending Order

db.books.aggregate([

{

$sort: {

published\_date: -1,

price: 1

}

}

])

// Step 1: Create the Database and Insert Dummy Data for E-commerce Orders

// Use the e-commerce database for Problem Statement 2

use ecommerce

// Insert sample data for orders collection

db.orders.insertMany([

{

orderId: 1,

customerId: 'C001',

items: [

{ productId: 'P001', productName: 'Smartphone', category: 'Electronics', price: 500, quantity: 1 },

{ productId: 'P002', productName: 'Laptop', category: 'Electronics', price: 1200, quantity: 1 }

],

orderDate: ISODate('2023-08-15T10:20:00.000Z'),

shippingAddress: '123 Elm St, Apt 4, Springfield, ZIP12345',

status: 'Shipped'

},

{

orderId: 2,

customerId: 'C002',

items: [

{ productId: 'P003', productName: 'T-shirt', category: 'Clothing', price: 20, quantity: 3 },

{ productId: 'P004', productName: 'Jeans', category: 'Clothing', price: 40, quantity: 2 }

],

orderDate: ISODate('2023-08-18T15:45:00.000Z'),

shippingAddress: '456 Oak St, Springfield, ZIP67890',

status: 'Delivered'

},

{

orderId: 3,

customerId: 'C001',

items: [

{ productId: 'P001', productName: 'Smartphone', category: 'Electronics', price: 500, quantity: 2 },

{ productId: 'P005', productName: 'Headphones', category: 'Electronics', price: 100, quantity: 1 }

],

orderDate: ISODate('2023-09-01T11:10:00.000Z'),

shippingAddress: '123 Elm St, Apt 4, Springfield, ZIP12345',

status: 'Pending'

},

{

orderId: 4,

customerId: 'C003',

items: [

{ productId: 'P006', productName: 'Coffee Maker', category: 'Home Appliances', price: 80, quantity: 1 }

],

orderDate: ISODate('2023-09-15T09:30:00.000Z'),

shippingAddress: '789 Pine St, Springfield, ZIP55555',

status: 'Cancelled'

}

]);

// Step 2: Perform Aggregation Operations

// 1. Identify the top 3 most popular product categories based on the number of items sold

db.orders.aggregate([

{ $unwind: "$items" },

{ $group: { \_id: "$items.category", totalQuantity: { $sum: "$items.quantity" } } },

{ $sort: { totalQuantity: -1 } },

{ $limit: 3 }

]);

// 2. Find out the total amount spent by each customer

db.orders.aggregate([

{ $unwind: "$items" },

{ $group: { \_id: "$customerId", totalSpent: { $sum: { $multiply: ["$items.price", "$items.quantity"] } } } }

]);

// 3. Identify the top 5 customers who've spent the most on the platform

db.orders.aggregate([

{ $unwind: "$items" },

{ $group: { \_id: "$customerId", totalSpent: { $sum: { $multiply: ["$items.price", "$items.quantity"] } } } },

{ $sort: { totalSpent: -1 } },

{ $limit: 5 }

]);

// 4. Order Insights for each month

// 4a. Calculate the total revenue

db.orders.aggregate([

{ $unwind: "$items" },

{ $group: { \_id: { month: { $month: "$orderDate" }, year: { $year: "$orderDate" } }, totalRevenue: { $sum: { $multiply: ["$items.price", "$items.quantity"] } } } }

]);

// 4b. Identify the most sold product for each month

db.orders.aggregate([

{ $unwind: "$items" },

{ $group: { \_id: { month: { $month: "$orderDate" }, year: { $year: "$orderDate" }, productId: "$items.productId" }, totalQuantity: { $sum: "$items.quantity" } } },

{ $sort: { "totalQuantity": -1 } },

{ $group: { \_id: "$\_id.month", mostSoldProduct: { $first: "$\_id.productId" }, quantity: { $first: "$totalQuantity" } } }

]);

// 4c. Determine the average number of items per order for each month

db.orders.aggregate([

{ $unwind: "$items" },

{ $group: { \_id: { month: { $month: "$orderDate" }, year: { $year: "$orderDate" }, orderId: "$orderId" }, totalItems: { $sum: "$items.quantity" } } },

{ $group: { \_id: "$\_id.month", avgItemsPerOrder: { $avg: "$totalItems" } } }

]);

// 4d. Find out the percentage of orders that were cancelled for each month

db.orders.aggregate([

{ $group: { \_id: { month: { $month: "$orderDate" }, year: { $year: "$orderDate" }, status: "$status" }, count: { $sum: 1 } } },

{ $group: { \_id: "$\_id.month", totalOrders: { $sum: "$count" }, cancelledOrders: { $sum: { $cond: [{ $eq: ["$\_id.status", "Cancelled"] }, "$count", 0] } } } },

{ $project: { \_id: 1, cancelPercentage: { $multiply: [{ $divide: ["$cancelledOrders", "$totalOrders"] }, 100] } } }

]);

// 5. Shipping Analysis by zip code

// 5a. Extract zip code and calculate the total number of orders and average order amount per zip code

db.orders.aggregate([

{ $unwind: "$items" },

{ $set: { zipCode: { $substr: ["$shippingAddress", -5, 5] } } },

{ $group: { \_id: "$zipCode", totalOrders: { $sum: 1 }, totalAmount: { $sum: { $multiply: ["$items.price", "$items.quantity"] } } } },

{ $project: { totalOrders: 1, avgOrderAmount: { $divide: ["$totalAmount", "$totalOrders"] } } }

]);