# MongoDB Queries for Online Shopping Platform

## 1. Find High-Spending Users

Query:

db.users.aggregate([  
 {  
 $lookup: {  
 from: "orders",  
 localField: "userId",  
 foreignField: "userId",  
 as: "orders"  
 }  
 },  
 {  
 $unwind: "$orders"  
 },  
 {  
 $group: {  
 \_id: "$userId",  
 name: { $first: "$name" },  
 totalSpent: { $sum: "$orders.totalAmount" }  
 }  
 },  
 {  
 $match: {  
 totalSpent: { $gt: 500 }  
 }  
 }  
])

### Sample Output:

[  
 { "\_id": "U001", "name": "John Doe", "totalSpent": 750 },  
 { "\_id": "U002", "name": "Jane Smith", "totalSpent": 600 }  
]

## 2. List Popular Products by Average Rating

Query:

db.products.aggregate([  
 {  
 $unwind: "$ratings"  
 },  
 {  
 $group: {  
 \_id: "$productId",  
 name: { $first: "$name" },  
 averageRating: { $avg: "$ratings.rating" }  
 }  
 },  
 {  
 $match: {  
 averageRating: { $gte: 4 }  
 }  
 }  
])

### Sample Output:

[  
 { "\_id": "P001", "name": "Wireless Mouse", "averageRating": 4.25 },  
 { "\_id": "P003", "name": "Bluetooth Headphones", "averageRating": 4.5 }  
]

## 3. Search for Orders in a Specific Time Range

Query:

db.orders.aggregate([  
 {  
 $match: {  
 orderDate: { $gte: ISODate("2024-12-01T00:00:00Z"), $lte: ISODate("2024-12-31T23:59:59Z") }  
 }  
 },  
 {  
 $lookup: {  
 from: "users",  
 localField: "userId",  
 foreignField: "userId",  
 as: "userDetails"  
 }  
 },  
 {  
 $unwind: "$userDetails"  
 },  
 {  
 $project: {  
 orderId: 1,  
 orderDate: 1,  
 totalAmount: 1,  
 status: 1,  
 "userDetails.name": 1  
 }  
 }  
])

### Sample Output:

[  
 { "orderId": "ORD001", "orderDate": "2024-12-10T14:32:00Z", "totalAmount": 250, "status": "Delivered", "userDetails": { "name": "John Doe" } },  
 { "orderId": "ORD002", "orderDate": "2024-12-15T10:20:00Z", "totalAmount": 300, "status": "Processing", "userDetails": { "name": "Jane Smith" } }  
]

## 4. Update Stock After Order Completion

Query:

db.orders.find({}).forEach(order => {  
 order.items.forEach(item => {  
 db.products.updateOne(  
 { productId: item.productId },  
 { $inc: { stock: -item.quantity } }  
 );  
 });  
});

## 5. Find Nearest Warehouse

Query:

db.warehouses.aggregate([  
 {  
 $geoNear: {  
 near: { type: "Point", coordinates: [-74.006, 40.7128] },  
 distanceField: "distance",  
 maxDistance: 50000,  
 spherical: true,  
 query: { products: "P001" }  
 }  
 }  
])

Sample Output:  
[  
 { "warehouseId": "W001", "distance": 15000 }  
]