MongoDB Assignment Questions

This document contains practical MongoDB assignment questions based on two real-world scenarios. Each assignment encourages applying CRUD operations, complex queries, sorting, and indexing.

Scenario 1: Library Management System

You are designing a MongoDB database to manage a library's book inventory and borrowing records. The database must store book details such as title, author, publication year, genre, and availability.

Assignment Tasks

1. Create a database named `libraryDB` and a collection named `books`.
2. Insert at least 5 book documents with fields: title, author, year, genre (array), and available (boolean).
3. Write a query to find all books published after 2015.
4. Retrieve only the `title` and `author` of books where `available` is true.
5. Update the availability of a specific book to false when it is borrowed.
6. Delete books of a particular genre (e.g., 'Magazine').
7. Find books where genre contains 'Science' or publication year is greater than 2020.
8. Sort the books by publication year in descending order.
9. Create an index on the `author` field .

Scenario 2: E-commerce Product Catalog

You are tasked to manage an e-commerce platform's product catalog using MongoDB. Products can have varying attributes such as name, category, price, stock quantity, and ratings.

Assignment Tasks

1. Create a database named `ecommerceDB` and a collection named `products`.
2. Insert at least 6 product documents with fields: name, category, price, stock, and ratings (array of numbers).
3. Write a query to find all products with a price greater than 1000 and stock less than 20.
4. Retrieve only the name and price of products in the category 'Electronics'.
5. Update the stock of a product after a purchase by reducing it by a given quantity.
6. Delete products with no ratings or where stock equals zero.
7. Find products where the average rating is greater than or equal to 4.5.
8. Sort products by price in ascending order and limit the result to the top 3 cheapest products.

```
db.products.find().sort({ price: 1 }).limit(3)
```

9. Create a index on `category` .