Task Requirement for Bookstore Database

Goal

Create a structured Bookstore Database with two tables: **Authors** and **Books**. Populate these tables with sample data and perform various SQL queries to retrieve, update, and analyze data effectively.

Steps to Achieve Expected Output

1. **Database and Tables Setup**
   - **Create Database**: Set up a database named `bookstore`, only if it doesn't already exist.
   - **Switch to Bookstore Database**: Use the `USE` statement to set `bookstore` as the active database.
   - **Authors Table**: Create a table named `Authors` with columns for `AuthorID`, `Name`, `Bio`, and `DateOfBirth`.
     - Hint: Use `AUTO_INCREMENT` for `AuthorID` to ensure it generates unique values automatically.
   - **Books Table**: Create a table named `Books` with columns for `BookID`, `Title`, `AuthorID`, `Genre`, `Price`, and `PublishDate`.
     - Hint: Set up a **foreign key** on `AuthorID` to link it with the `Authors` table's `AuthorID` to maintain referential integrity.
2. **Data Insertion**
   - Insert sample data into the `Authors` table, including authors like J.K. Rowling and George R.R. Martin.
     - Hint: Use `ON DUPLICATE KEY UPDATE` to avoid duplicate entries if running this code multiple times.
   - Insert sample data into the `Books` table, adding books by these authors.
     - Hint: Use `ON DUPLICATE KEY UPDATE` here as well to prevent duplicate data on multiple insert attempts.
3. **Data Retrieval and Basic Queries**
   - **List all Books and Authors**:
     - Execute a `SELECT *` query on both tables to confirm data has been added successfully.
   - **Filter Books by Author**:
     - Write a query to find books by a specific author, e.g., J.K. Rowling (`AuthorID = 1`).
   - **Update Book Price**:
     - Update the price of "A Game of Thrones" to 26.99.
   - **Delete a Book**:

- Delete “A Game of Thrones” from the `Books` table.

4. **Data Analysis and Aggregation**
   - **Count Total Books**:
     - Use `COUNT()` to get the total number of books.
   - **Find Authors by Genre**:
     - Retrieve authors who have written books in the “Fantasy” genre using a `JOIN` on the `Books` and `Authors` tables.
   - **Most Recent Book**:
     - Find the most recently published book by ordering `PublishDate` in descending order and limiting the result to one.
   - **Average Book Price**:
     - Calculate the average price of all books.
5. **Complex Queries with Aggregations**
   - **Books per Author**:
     - Write a query to show each author’s name and the total number of books they have written.
     - Hint: Use `GROUP BY` with `COUNT()` for grouping and counting.
   - **Books and Authors Information**:
     - List all books along with their authors’ names and genres.
   - **Total Revenue by Genre**:
     - Calculate the total revenue generated by each genre using `SUM(Price)`.
   - **Highest-Priced Book per Author**:
     - Retrieve the title and price of the highest-priced book for each author.
     - Hint: Use a **correlated subquery** to find the maximum price within each author’s books.
   - **Authors with Multiple Books**:
     - List authors who have written more than one book.
     - Hint: Use `HAVING COUNT(BookID) > 1` to filter authors with multiple books.
   - **Average Price per Author**:
     - Calculate the average book price for each author using `AVG(Price)` and grouping by `AuthorID`.

Additional Hints

- **Use Aliases**: Use table aliases like `a` for `Authors` and `b` for `Books` in join queries for readability.
- **Practice with Aggregates**: Familiarize yourself with `COUNT()`, `SUM()`, `AVG()`, and `MAX()` to handle different types of data summarization tasks.

- **Understanding Constraints**: Make sure to understand how the `FOREIGN KEY` constraint helps maintain data integrity between the `Books` and `Authors` tables.

Expected Outcomes

After executing these steps, you should have:

1. A fully populated `Authors` and `Books` table in the `bookstore` database.
2. Accurate results from each of the queries, reflecting the data relationships, filtering, and aggregations specified above.