Case Study: BookHub Online Bookstore

Context:
BookHub is an online bookstore that has been operating for the past 5 years. It sells books across various genres, including fiction, non-fiction, academic, and children's books. BookHub also offers a platform for independent authors to sell their books. The company operates globally and has a diverse customer base.

Database Schema
The database for BookHub consists of the following tables:

```
create database BookHub
```

Books
BookID (INT, Primary Key)
Title (VARCHAR)
Author (VARCHAR)
Genre (VARCHAR)
Price (FLOAT)
PublishDate (DATE)

```
use BookHub;
create table Books (
    BookID int PRIMARY KEY,
    Title varchar(255),
    Author varchar(255),
    Genre varchar(50),
    Price float,
    PublishDate date)
```

```
INSERT INTO Books (BookID, Title, Author, Genre, Price, PublishDate) VALUES
(1, 'The Final Empire', 'Brandon Sanderson', 'Fantasy', 15.99, '2006-07-17'),
(2, 'Pride and Prejudice', 'Jane Austen', 'Classic', 9.99, '1813-01-28'),
(3, 'To Kill a Mockingbird', 'Harper Lee', 'Fiction', 12.99, '1960-07-11'),
(4, 'The Great Gatsby', 'F. Scott Fitzgerald', 'Classic', 14.99, '1925-04-10'),
(5, '1984', 'George Orwell', 'Dystopian', 13.99, '1949-06-08'),
(6, 'The Catcher in the Rye', 'J.D. Salinger', 'Fiction', 10.99, '1951-07-16'),
(7, 'Brave New World', 'Aldous Huxley', 'Dystopian', 11.99, '1932-01-01'),
(8, 'The Hobbit', 'J.R.R. Tolkien', 'Fantasy', 14.99, '1937-09-21'),
(9, 'Harry Potter and the Sorcerer's Stone', 'J.K. Rowling', 'Fantasy', 16.99, '1997-06-26'),
(10, 'Life of Pi', 'Yann Martel', 'Adventure Fiction', 13.99, '2001-09-11');
```

Sales
SaleID (INT, Primary Key)
BookID (INT, Foreign Key - References Books)
Quantity (INT)
SaleDate (DATE)
CustomerID (INT, Foreign Key - References Customers)

```
create table Sales(
    SaleID int PRIMARY KEY,
    BookID int,
    Quantity int,
    SaleDate date,
    CustomerID int,
    FOREIGN KEY (BookID) REFERENCES Books(BookID),
    FOREIGN KEY (CustomerID) REFERENCES
Customers(CustomerID)
);
```

```
INSERT INTO Sales (SaleID, BookID, Quantity, SaleDate, CustomerID) VALUES
(1, 1, 2, '2023-03-01', 1),
(2, 2, 1, '2023-03-02', 2),
(3, 3, 1, '2023-03-03', 3),
(4, 4, 3, '2023-03-04', 4),
(5, 5, 2, '2023-03-05', 5),
(6, 6, 1, '2023-03-06', 6),
(7, 7, 1, '2023-03-07', 7),
(8, 8, 1, '2023-03-08', 8),
(9, 9, 2, '2023-03-09', 9),
(10, 10, 3, '2023-03-10', 10);
```

```
INSERT INTO Sales (SaleID, BookID, Quantity, SaleDate, CustomerID) VALUES
(11, 1, 1, '2023-03-11', 1),
(12, 2, 2, '2023-03-12', 3),
(13, 3, 2, '2023-03-13', 3),
(14, 4, 2, '2023-03-14', 4),
(15, 5, 1, '2023-03-15', 4),
(16, 6, 2, '2023-03-16', 4),
(17, 7, 3, '2023-03-17', 4),
(18, 8, 2, '2023-03-18', 9),
(19, 9, 1, '2023-03-19', 9),
(20, 10, 1, '2023-03-20', 10);
```

Customers
CustomerID (INT, Primary Key)
FirstName (VARCHAR)
LastName (VARCHAR)
Email (VARCHAR)
JoinDate (DATE)
Country (VARCHAR)

```sql
create table Customers(
    CustomerID int PRIMARY KEY,
    FirstName varchar(30),
    LastName varchar(30),
    Email varchar(60),
    JoinDate date,
    Country varchar(30))
```

```sql
INSERT INTO Customers (CustomerID, FirstName, LastName, Email, JoinDate, Country) VALUES
(1, 'John', 'Doe', 'john.doe@example.com', '2020-01-15', 'USA'),
(2, 'Jane', 'Smith', 'jane.smith@example.com', '2020-02-20', 'Canada'),
(3, 'Emily', 'Jones', 'emily.jones@example.com', '2020-03-25', 'UK'),
(4, 'Michael', 'Brown', 'michael.brown@example.com', '2020-04-30', 'Australia'),
(5, 'Jessica', 'Taylor', 'jessica.taylor@example.com', '2020-05-05', 'New Zealand'),
(6, 'William', 'Davis', 'william.davis@example.com', '2020-06-10', 'Ireland'),
(7, 'Sophia', 'Miller', 'sophia.miller@example.com', '2020-07-15', 'South Africa'),
(8, 'James', 'Wilson', 'james.wilson@example.com', '2020-08-20', 'India'),
(9, 'Olivia', 'Moore', 'olivia.moore@example.com', '2020-09-25', 'Germany'),
(10, 'Liam', 'Taylor', 'liam.taylor@example.com', '2020-10-30', 'France');
```

Tasks

Basic Queries

a. List all book titles along with their authors.

```
select Title, Author from books
```

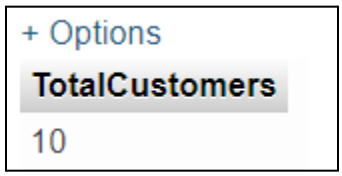
	Title	Author
Edit Copy Delete	The Final Empire	Brandon Sanderson
Edit Copy Delete	Pride and Prejudice	Jane Austen
Edit Copy Delete	To Kill a Mockingbird	Harper Lee
Edit Copy Delete	The Great Gatsby	F. Scott Fitzgerald
Edit Copy Delete	1984	George Orwell
Edit Copy Delete	The Catcher in the Rye	J.D. Salinger
Edit Copy Delete	Brave New World	Aldous Huxley
Edit Copy Delete	The Hobbit	J.R.R. Tolkien
Edit Copy Delete	Harry Potter and the Sorcerer's Stone	J.K. Rowling
Edit Copy Delete	Life of Pi	Yann Martel

b. Find the total number of customers.

```
select DISTINCT COUNT(*) as TotalCustomers from
Customers
```

or

```
select COUNT(*) as TotalCustomers from Customers
```



+ Options

TotalCustomers
10

c. Show the top 5 bestselling books by quantity sold.

```
select B.title, sum(S.quantity) as TotalSold
from Sales S
join Books B on S.BookID = B.BookID
group by B.BookID
order by TotalSold Desc
LIMIT 5
```

+ Options

title	TotalSold ▼ 1
Life of Pi	3
The Great Gatsby	3
1984	2
Harry Potter and the Sorcerer's Stone	2
The Final Empire	2

Intermediate Queries

a. Calculate the total revenue generated from sales.

```
select SUM(S.quantity*B.price) as TotalRevenue
from Sales S
join Books B on S.BookID = B.BookID
```

TotalRevenue
241.8299961090088

b. Identify the best-selling genre.

```
select B.Genre, SUM(S.quantity) as TotalSold
from Sales S
join Books B on S.BookID = B.BookID
group by B.Genre
order by TotalSold DESC
LIMIT 1
```

Genre	TotalSold
Fantasy	5

c. List customers who have made more than 5 purchases.

```
SELECT C.CustomerID, CONCAT(C.FirstName, ' ', C.LastName) AS FullName, COUNT(*) AS Purchases
FROM Sales S
JOIN Customers C ON S.CustomerID = C.CustomerID
GROUP BY C.CustomerID
HAVING Purchases > 3;
```

CustomerID	FullName	Purchases
4	Michael Brown	5

Advanced Queries

a. For each country, find the most popular book based on the quantity sold.

```
select C.country, B.title, sum(S.Quantity) as TotalSold
From Sales S
join books B on S.BookId = B.BookID
join customers C on S.CustomerID = C.CustomerID
group by C.country, B.BookID
order by C.country, TotalSold Desc
```

country ▲ 1	title	TotalSold ▼ 2
Australia	The Great Gatsby	5
Australia	Brave New World	3
Australia	The Catcher in the Rye	2
Australia	1984	1
Canada	Pride and Prejudice	1
France	Life of Pi	4
Germany	Harry Potter and the Sorcerer's Stone	3
Germany	The Hobbit	2
India	The Hobbit	1
Ireland	The Catcher in the Rye	1
New Zealand	1984	2
South Africa	Brave New World	1
UK	To Kill a Mockingbird	3
UK	Pride and Prejudice	2
USA	The Final Empire	3

b. Identify the month with the highest sales in terms of revenue.

```
SELECT EXTRACT(YEAR FROM SaleDate) AS SaleYear,
EXTRACT(MONTH FROM SaleDate) AS SaleMonth,
SUM(S.Quantity * B.Price) AS Revenue
FROM Sales S
JOIN Books B ON S.BookID = B.BookID
GROUP BY SaleYear, SaleMonth
ORDER BY Revenue DESC
LIMIT 1;
```

SaleYear	SaleMonth	Revenue
2023	3	466.6599922180176

c. Determine the average sale value per book genre.

```
select B.genre, AVG(S.quantity*B.price) as AvgSaleValue
from Sales S
join Books B on S.bookId = B.BookID
group by B.Genre
```

genre	AvgSaleValue
Adventure Fiction	27.979999542236328
Classic	26.229999542236328
Dystopian	22.482499599456787
Fantasy	23.984999656677246
Fiction	17.984999656677246

Analytical Queries

a. Analyze the sales trend over the years.

```
SELECT EXTRACT(YEAR FROM SaleDate) AS Year,
SUM(S.Quantity * B.Price) AS Revenue
FROM Sales S
JOIN Books B ON S.BookID = B.BookID
GROUP BY Year
ORDER BY Year;
```

Year	Revenue
2023	466.6599922180176

b. Predict the genres that will see growth in sales based on historical data.

```
SELECT B.Genre, EXTRACT(YEAR FROM SaleDate) AS Year,
SUM(S.Quantity) AS TotalSold
FROM Sales S
JOIN Books B ON S.BookID = B.BookID
GROUP BY B.Genre, Year
ORDER BY B.Genre, Year;
```

Genre ▲ 1	Year ▲ 2	TotalSold
Adventure Fiction	2023	4
Classic	2023	8
Dystopian	2023	7
Fantasy	2023	9
Fiction	2023	6

c. Segment customers based on their purchasing behavior (e.g., frequent buyers, high spenders, etc.).

```
SELECT CustomerID, COUNT(*) AS PurchaseCount
FROM Sales
GROUP BY CustomerID
HAVING PurchaseCount >= 2;
```

CustomerID	PurchaseCount
1	2
3	3
4	5
9	3
10	2