```
---Create Books Table
Drop table if exists books
Create table books(
Book_ID INT Primary key,
Title varchar(100),
Authorvarchar(100),
Genre  varchar(50),
Published_Year int,
Price numeric(10,2),
Stock int )
---Create Books Customers
Drop table if exists Customers
Create table Customers(
Customer_ID int primary key ,
Name varchar(100),
Email varchar(100),
Phone varchar(15),
City varchar(50),
Country varchar(150))
---Create Orders Table
Drop table if exists orders
Create table orders(
Order_ID        int,
Customer_ID int References customers(Customer_ID),
Book_ID int References books(Book_ID),
Order_Date date,
Quantity int,
Total_Amount numeric(10,2))
```

```
--Copy data into Book table using Bulk
BULK INSERT books
FROM 'C:\Users\anshu\Downloads\Books.csv'
WITH (
   FORMAT = 'CSV',
   FIRSTROW = 2,    -- Skip the header row
   FIELDTERMINATOR = ',',
   ROWTERMINATOR = '0x0A',
   TABLOCK );
--Copy data into Customers table using Bulk
BULK INSERT Customers
FROM "C:\Users\anshu\Downloads\Customers.csv"
WITH (
   FORMAT = 'CSV',
   FIRSTROW = 2,    -- Skip the header row
   FIELDTERMINATOR = ',',
   ROWTERMINATOR = '0x0A',
   TABLOCK );
--Copy data into Orders table using Bulk
BULK INSERT orders
FROM "C:\Users\anshu\Downloads\Orders.csv"
WITH (
   FORMAT = 'CSV',
   FIRSTROW = 2,    -- Skip the header row
   FIELDTERMINATOR = ',',
   ROWTERMINATOR = '0x0A',
   TABLOCK);
```

#Basic Query

--1.Retrieve all books in the ‘friction’ genre

Select * from books where genre='Fiction'

--2.Find books published after the year 1950

Select * from books where Published_Year>'1950'

---3.List of all customers from canada

Select * from Customers

where Country='Canada'

---4.Show orders placed in November 2023

where Order_Date between '2023-11-01' and '2023-11-30'

--5.Retrieve total stock of books available

select sum(stock) as Total_Book_stock from books

--6.Find the details of the most expensive book

SELECT TOP 1 * FROM books ORDER BY Price DESC;

--7.Show all the customers who ordered more than 1 quantity of book

Select * from orders where Quantity>1

--8.Retrieve all orders where the total amount exceeds $20

Select * from orders

Select * from orders Where Total_Amount> 20

---9.List all genres available in the book table

SELECT Distinct(Genre) FROM BOOKS

--10.Find the book with the lowest stock

Select top 1 * from books order by stock asc

--11.Calculate the total revenue generated from all orders

Select sum(Total_Amount) as revenue_generated from orders

```
--Advanced Queries

--1.retrieve the total number of books sold for each  genre

 select b.genre,sum(o.total_amount) as Total_book_sold from books as b   join  orders as o on b.Book_ID=o.Book_ID group by Genre

--2.find the avg price  of books in ‘fantastic’ genre

Select genre,avg(price) as avg_price from books  group by genre having Genre ='fantasy'

--3.ist the customers who have placed at least 2 orders

Select c.Customer_ID,c.Name,COUNT(o.Order_ID) as order_count

from orders as o

join  Customers as c on c.Customer_ID=o.customer_id

group by c.Customer_ID,c.Name

having  count(Order_ID)>=2

--4.find the most frequently order book

Select book_id,count(order_id) As order_count

from orders

Group by Book_ID

order by  order_count DESC

--5.show the top 3  most expensive books  of  ‘fantasy’ genre

select * from books

Select TOP 3 * from books where Genre='Fantasy'

order by Price DESC

--6.Retrive  the total quantity  of books sold by each author

Select b.author , SUM(O.Quantity)AS Total_quantity from orders as o join books as b

on o.Book_ID=b.Book_ID group by b.author

---7.list the cities where customers who spent  over $30 are located

Select * from Customers

select * from orders

select Distinct c.city,total_amount from orders as  o
```

```
join
customers as c on c.Customer_ID=o.Customer_ID
where o.Total_Amount> 30
--8.find the customer who spent the most  on orders
Select top 1 c.customer_id ,c.name,sum(o.total_amount) as total_spent
from orders as o join
customers as c on c.customer_id=o.Customer_ID
group by c.customer_id,c.Name
order by total_spent DESC
---------------------------alternative way
SELECT TOP 1
WITH TIES c.customer_id, c.name,  SUM(o.total_amount) AS total_spent
FROM orders o
JOIN customers c ON c.customer_id = o.Customer_ID
GROUP BY c.customer_id, c.name
ORDER BY SUM(o.total_amount) DESC;
--9.calculate the stock remaining  after fulfilling all orders
SELECT DISTINCT b.Book_ID, b.Title,b.Stock AS Initial_Stock,
ISNULL(SUM(o.Quantity) OVER (PARTITION BY b.Book_ID), 0) AS Total_Ordered,
(b.Stock - ISNULL(SUM(o.Quantity) OVER (PARTITION BY b.Book_ID), 0)) AS
Remaining_Stock
FROM Books b LEFT JOIN Orders o ON b.Book_ID = o.Book_ID;
-------Alternative way
SELECT b.book_Id,  b.title, b.Stock,
COALESCE(SUM(o.Quantity), 0) AS order_quantity,
b.Stock - COALESCE(SUM(o.Quantity), 0) AS Remaining_quantity
FROM books AS b  LEFT JOIN orders AS o ON b.Book_ID = o.Book_ID
GROUP BY b.Book_ID, b.title, b.Stock
```