```sql
DROP DATABASE IF EXISTS Jungle_Book;

CREATE DATABASE Jungle_Book;

USE Jungle_Book;

-- CREATE TABLES
-- creating the customer table
CREATE TABLE Customer
(
    CustomerID INT(11) NOT NULL AUTO_INCREMENT,
    CustomerFirstName VARCHAR(32) NOT NULL,
    CustomerLastName VARCHAR(32) NOT NULL,
    BillingAddress VARCHAR(255) NOT NULL,
    ShippingAddress VARCHAR(255) NOT NULL,
    PhoneNum VARCHAR(11) NOT NULL,
    EmailAddress VARCHAR(80) NOT NULL,
    Username VARCHAR(32) NOT NULL,
    Password VARCHAR(32) NOT NULL,
    PRIMARY KEY (CustomerID)
);

-- creating the book table
CREATE TABLE Book
(
    ISBN VARCHAR(13) NOT NULL,
    BookTitle VARCHAR(80) NOT NULL,
    BriefSummary VARCHAR(255) NOT NULL,
    Price FLOAT NOT NULL,
    PRIMARY KEY (ISBN)
);

-- creating author table
CREATE TABLE Author
(
    AuthorID INT(11) NOT NULL AUTO_INCREMENT,
    AuthorFirstName VARCHAR(32) NOT NULL,
    AuthorLastName VARCHAR(32) NOT NULL,
    PRIMARY KEY (AuthorID)
);

-- create order table
CREATE TABLE CustomerOrder
(
    OrderNum INT(11) NOT NULL AUTO_INCREMENT,
    DateOrdered DATETIME NOT NULL,
    DateShipped DATETIME NULL,
    OrderStatus VARCHAR(32) NOT NULL,
    OrderSubtotal FLOAT NOT NULL,
    Taxes FLOAT NOT NULL,
    TotalCost FLOAT NOT NULL,
    CustomerID INT(11) NOT NULL,
    PRIMARY KEY (OrderNum)
);

-- create book category
CREATE TABLE BookCategory
(
    BookCategoryID INT(11) NOT NULL AUTO_INCREMENT,
    BookCategoryName VARCHAR(32) NOT NULL,
    PRIMARY KEY (BookCategoryID)
);

-- create OrderBook
CREATE TABLE Order_Book
(
    OrderNum INT(11) NOT NULL,
    ISBN VARCHAR(13) NOT NULL,
    Quantity INT(11) NOT NULL,
    Subtotal FLOAT NOT NULL,
    PRIMARY KEY (OrderNum, ISBN)
);

-- create Author_Book
CREATE TABLE Author_Book
(
    AuthorID INT(11) NOT NULL,
    ISBN VARCHAR(13) NOT NULL,
    PRIMARY KEY (AuthorID, ISBN)
);

-- create Book_BookCategory
CREATE TABLE Book_BookCategory
(
    ISBN VARCHAR(13) NOT NULL,
    BookCategoryID INT(11) NOT NULL,
    PRIMARY KEY (ISBN, BookCategoryID)
);

-- ALTERING TABLES
ALTER TABLE CustomerOrder
    ADD CONSTRAINT FK_CustomerOrder_CustomerID
    FOREIGN KEY (CustomerID)
    REFERENCES Customer (CustomerID);

ALTER TABLE Author_Book
    ADD CONSTRAINT FK_Author_Book_AuthorID
    FOREIGN KEY (AuthorID)
    REFERENCES Author (AuthorID);

ALTER TABLE Author_Book
    ADD CONSTRAINT FK_Author_Book_ISBN
    FOREIGN KEY (ISBN)
    REFERENCES Book (ISBN);

ALTER TABLE Order_Book
    ADD CONSTRAINT FK_Order_Book_OrderNum
    FOREIGN KEY (OrderNum)
    REFERENCES CustomerOrder (OrderNum);

ALTER TABLE Order_Book
    ADD CONSTRAINT FK_Order_Book_ISBN
    FOREIGN KEY (ISBN)
    REFERENCES Book (ISBN);

ALTER TABLE Book_BookCategory
    ADD CONSTRAINT FK_Book_BookCategory_ISBN
    FOREIGN KEY (ISBN)
    REFERENCES Book (ISBN);

ALTER TABLE Book_BookCategory
    ADD CONSTRAINT FK_Book_BookCategory_BookCategoryID
    FOREIGN KEY (BookCategoryID)
    REFERENCES BookCategory (BookCategoryID);

-- adding the categories to Book Category
INSERT INTO BookCategory
    (BookCategoryName)
VALUES
    ('Fantasy');

INSERT INTO BookCategory
    (BookCategoryName)
VALUES
    ('Mystery');

INSERT INTO BookCategory
    (BookCategoryName)
```

```sql
VALUES
    ('Horror');

INSERT INTO BookCategory
    (BookCategoryName)
VALUES
    ('Thriller');

INSERT INTO BookCategory
    (BookCategoryName)
VALUES
    ('Databases');

-- adding books to Book
INSERT INTO Book
    (ISBN, BookTitle, BriefSummary, Price)
VALUES
    ('9780133970777', 'Fundamentals of Database Systems (7th Edition)',
     'This book introduces the fundamental concepts necessary for designing,
      using, and implementing database systems and database applications. ' , 179.16);

INSERT INTO Book
    (ISBN, BookTitle, BriefSummary, Price)
VALUES
    ('9780307269751', 'The Girl with the Dragon Tattoo',
     'An exciting, stunningly intelligent novel about society at its most hidden,
     and about the intimate lives of a brilliantly realized cast of characters,
     all of them forced to face the darker aspects of their world and of their own lives.' , 35.00);

INSERT INTO Book
    (ISBN, BookTitle, BriefSummary, Price)
VALUES
    ('9780385121675', 'The Shining',
     'Jack Torrance gets a new job at the hotel and spends time with his family and his writing.
     Winter comes, the hotel feels remote and sinister,
     the only one to notice the strange and terrible forces is Danny Torrance, a unique 5 year old.' , 49.99);

-- adding authors to Author
```

```sql
INSERT INTO Author
    (AuthorFirstName, AuthorLastName)
VALUES
    ('Ramez', 'Elmasri');

INSERT INTO Author
    (AuthorFirstName, AuthorLastName)
VALUES
    ('Shamkant', 'Navathe');

INSERT INTO Author
    (AuthorFirstName, AuthorLastName)
VALUES
    ('Stephen', 'King');

INSERT INTO Author
    (AuthorFirstName, AuthorLastName)
VALUES
    ('Stieg', 'Larsson');

-- adding the author to each book
INSERT INTO Author_Book
    (AuthorID, ISBN)
VALUES
    (4, 9780307269751);

INSERT INTO Author_Book
    (AuthorID, ISBN)
VALUES
    (3, 9780385121675);

INSERT INTO Author_Book
    (AuthorID, ISBN)
VALUES
    (2, 9780133970777);

INSERT INTO Author_Book
    (AuthorID, ISBN)
VALUES
    (1, 9780133970777);

-- adding categories to each book
INSERT INTO Book_BookCategory
    (ISBN, BookCategoryID)
VALUES
    ('9780307269751', '2');
```

```sql

INSERT INTO Book_BookCategory
    (ISBN, BookCategoryID)
VALUES
    ('9780307269751', '4');

INSERT INTO Book_BookCategory
    (ISBN, BookCategoryID)
VALUES
    ('9780385121675', '2');

INSERT INTO Book_BookCategory
    (ISBN, BookCategoryID)
VALUES
    ('9780385121675', '3');

INSERT INTO Book_BookCategory
    (ISBN, BookCategoryID)
VALUES
    ('9780133970777', '5');

-- adding customers into Customer
INSERT INTO Customer
    (CustomerFirstName, CustomerLastName, BillingAddress, ShippingAddress, PhoneNum, EmailAddress, Username, Password)
VALUES
    ('Monica', 'Geller',
     '77 Somewhere Lane', '77 Somewhere Lane',
     '9052177477', 'monica.geller@gmail.com',
     'monCleanFreak', 'monica77');

INSERT INTO Customer
    (CustomerFirstName, CustomerLastName, BillingAddress, ShippingAddress, PhoneNum, EmailAddress, Username, Password)
VALUES
    ('Chandler', 'Bing',
     '15 Area Road', '90 Anywhere Avenue',
     '4754801478', 'chandler.bing@yahoo.com',
     'missChanandelerBong', 'bing123');

INSERT INTO Customer
    (CustomerFirstName, CustomerLastName, BillingAddress, ShippingAddress, PhoneNum, EmailAddress, Username, Password)
```

```sql
VALUES
    ('Rachel', 'Green',
     '109 Fashion Avenue', '109 Fashion Avenue',
     '5559287541', 'rachel_green@hotmail.com',
     'greenMachine', 'emma234');

INSERT INTO Customer
    (CustomerFirstName, CustomerLastName, BillingAddress, ShippingAddress, PhoneNum, EmailAddress, Username, Password)
VALUES
    ('Joseph', 'Tribbiani',
     '44 Unknown Street', '15 Area Road',
     '8254457859', 'joey_tribbiani@icloud.com',
     'joeyTrib', 'joey22');

-- creating 3 new orders

-- customer one buys one copy of all three books
INSERT INTO CustomerOrder
    (DateOrdered, DateShipped, OrderStatus, OrderSubtotal, Taxes, TotalCost, CustomerID)
VALUES
    ('2019-11-23T09:07:17.000', NULL, 'pending confirmation', 264.15, 39.62, 303.77, 1);
-- adding the items in the customer's order
-- customer 1, order 1
INSERT INTO Order_Book
    (OrderNum, ISBN, Quantity, Subtotal)
VALUES
    (1, '9780307269751', 1, 35);

INSERT INTO Order_Book
    (OrderNum, ISBN, Quantity, Subtotal)
VALUES
    (1, '9780385121675', 1, 49.99);

INSERT INTO Order_Book
    (OrderNum, ISBN, Quantity, Subtotal)
VALUES
    (1, '9780133970777', 1, 179.16);


-- customer 2 buys just one of the books
INSERT INTO CustomerOrder
    (DateOrdered, DateShipped, OrderStatus, OrderSubtotal
, Taxes, TotalCost, CustomerID)
VALUES
    ('2019-11-22T10:04:02.000', NULL, 'pending confirmation', 49.99, 7.50, 57.49, 2);

-- adding the items in the customer's order
INSERT INTO Order_Book
    (OrderNum, ISBN, Quantity, Subtotal)
VALUES
    (2, '9780385121675', 1, 49.99);


-- customer 3 buys five copies of one book, two copies of the second book, and ten copies of the third
INSERT INTO CustomerOrder
    (DateOrdered, DateShipped, OrderStatus, OrderSubtotal, Taxes, TotalCost, CustomerID)
VALUES
    ('2019-11-25T05:12:02.000', NULL, 'pending confirmation', 1465.70, 73.29, 1685.56, 3);

-- adding the item's in the customer's order
INSERT INTO Order_Book
    (OrderNum, ISBN, Quantity, Subtotal)
VALUES
    (3, '9780133970777', 5, 895.80);

INSERT INTO Order_Book
    (OrderNum, ISBN, Quantity, Subtotal)
VALUES
    (3, '9780307269751', 2, 70);

INSERT INTO Order_Book
    (OrderNum, ISBN, Quantity, Subtotal)
VALUES
    (3, '9780385121675', 10, 499.90);

-- delete the second order
DELETE FROM Order_Book
    WHERE OrderNum = 2;
DELETE FROM CustomerOrder
    WHERE CustomerID = 2;

-- change the status of the first order to complete
UPDATE CustomerOrder
SET DateShipped = '2019-11-26T04:12:02.000', OrderStatus
```

```sql
 = 'complete'
WHERE CustomerID = 1 AND OrderNum = 1;

-- change the third order to add another copy of the second book
UPDATE Order_Book
SET Quantity = 3, Subtotal = 105
WHERE OrderNum = 3 AND ISBN = '9780307269751';
UPDATE CustomerOrder
SET OrderSubtotal = 1500.70, Taxes = 75.04, TotalCost = 1725.81
WHERE CustomerID = 3 AND OrderNum = 3;

-- display all customer information for customers with no orders
SELECT C.CustomerID, CustomerFirstName, CustomerLastName, BillingAddress, ShippingAddress, PhoneNum, EmailAddress, Username, Password
FROM Customer C  LEFT OUTER JOIN CustomerOrder CO
ON C.CustomerID = CO.CustomerID
WHERE OrderNum is NULL;

-- display the title, author, ISBN, and price of all books
SELECT B.ISBN, BookTitle, Price, A.AuthorID, AuthorLastName, AuthorFirstName
FROM Book B INNER JOIN Author_Book AB ON B.ISBN = AB.ISBN
            INNER JOIN Author A ON AB.AuthorID = A.AuthorID
            INNER JOIN Book_BookCategory BBC ON AB.ISBN = BBC.ISBN
            INNER JOIN BookCategory BC ON BBC.BookCategoryID = BC.BookCategoryID
            WHERE BookTitle LIKE '%Database%' OR BookCategoryName LIKE '%Database%';

-- display the email addresses of customers who have outstanding orders
SELECT EmailAddress
FROM Customer C  RIGHT OUTER JOIN CustomerOrder CO
ON C.CustomerID = CO.CustomerID
WHERE DateShipped is NULL;

-- display just the order information on all orders that have purchased more than one copy of any books
SELECT CustomerOrder.OrderNum, CustomerOrder.DateOrdered, CustomerOrder.DateShipped, CustomerOrder.OrderStatus,
       CustomerOrder.OrderSubtotal, CustomerOrder.Taxes, CustomerOrder.TotalCost, CustomerOrder.CustomerID
FROM CustomerOrder  RIGHT OUTER JOIN Order_Book OB
ON CustomerOrder.OrderNum = OB.OrderNum
WHERE Quantity > 1
GROUP BY OB.OrderNum
ORDER BY OB.OrderNum;

-- display the order number and total number of books in each order
SELECT CustomerOrder.OrderNum, SUM(Quantity)
FROM CustomerOrder
INNER JOIN Order_Book OB
ON CustomerOrder.OrderNum = OB.OrderNum
GROUP BY CustomerOrder.OrderNum;


-- display each order number, customer name, and total cost of the order
SELECT CO.OrderNum, CustomerLastName, CustomerFirstName, SUM(Quantity * Price * 1.15)
FROM CustomerOrder CO INNER JOIN Customer C ON CO.CustomerID = C.CustomerID
                      INNER JOIN Order_Book OB ON CO.OrderNum = OB.OrderNum
                      INNER JOIN Book B ON OB.ISBN = B.ISBN
GROUP BY CO.OrderNum;
```