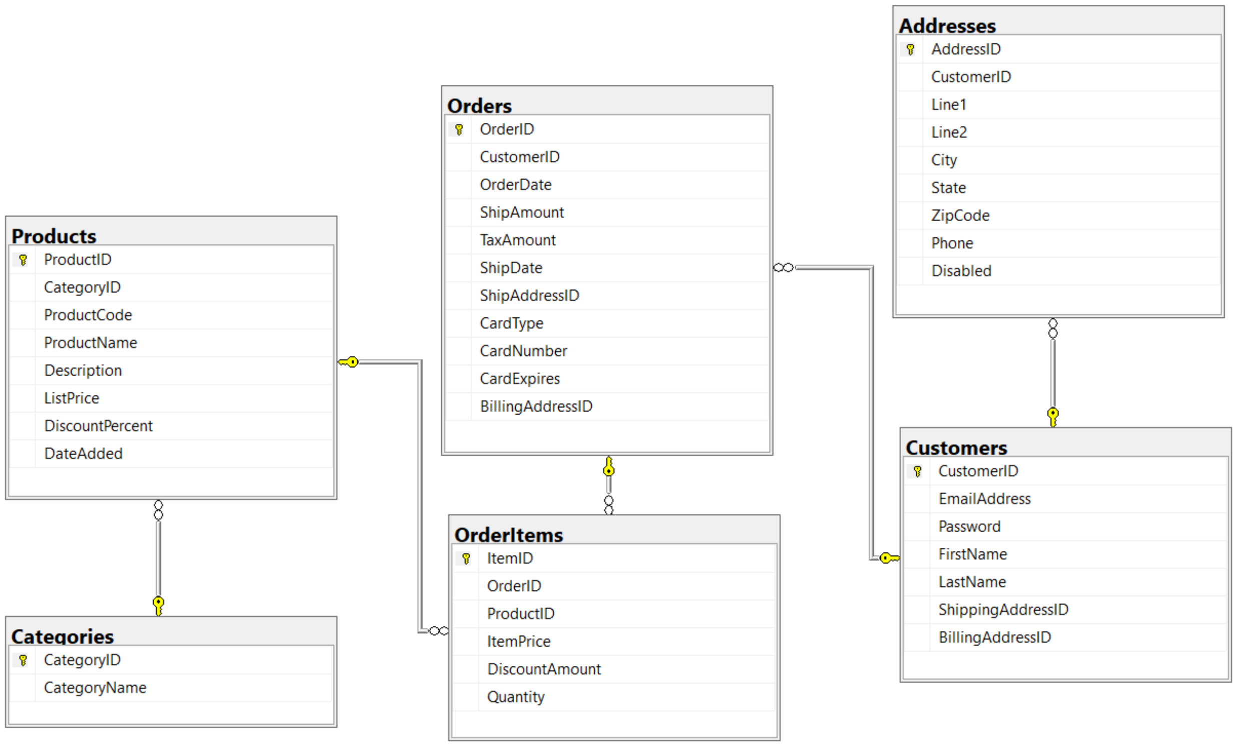
Use MyGuitarShop DB



1. Write a SELECT statement that returns the same result set as this SELECT statement, but don't use a join. Instead, use a subquery in a WHERE clause that uses the IN operator.

```
SELECT DISTINCT CategoryName
FROM Categories c JOIN Products p
  ON c.CategoryID = p.CategoryID
ORDER BY CategoryName
```

2. Write a SELECT statement that answers this question: Which products have a list price that's greater than the average list price for all products?

   Return the ProductName and ListPrice columns for each product.

   Sort the results by the ListPrice column in descending sequence.

3. Write a SELECT statement that returns the CategoryName column from the Categories table.

   Return one row for each category that has never been assigned to any product in the Products table. To do that, use a subquery introduced with NOT EXISTS.

4. Write a SELECT statement that returns three columns: EmailAddress, OrderID, and the order total for each order. To do this, you can group the result set by the EmailAddress and OrderID columns. Then, you can calculate the order total from the columns in the OrderItems table.

   Write a second SELECT statement that uses the first SELECT statement in its FROM clause. The main query should return two columns: the customer's email address and the largest order for that customer. To do this, you can group the result set by the EmailAddress column.

5. Write a SELECT statement that returns the name and discount percent for each product that has a unique discount percent. In other words, don't include products that have the same discount percent as another product.

   Sort the results by the ProductName column.

6. Use a correlated subquery to return one row per customer with each customer's oldest order (the one with the earliest date). Each row should include these three columns: EmailAddress, OrderID, and OrderDate.