OrderManager: Managing Products and Orders

1. Background and Overview

OrderManager is an e-commerce program that enables a business to manage information about products that can be sold to customers, to track current inventories of products, and to process orders for products from customers. The process involves placing the orders, processes and managing the customers, inventories and data connected to the order.

The development tool we use is Derby and mySQL.

The main process includes:

1.1 Order.

Customer places the order, there will produce several order record. If the inventory unit is available, the product will be shipped.

1.2 Inventory availability.

Before shipping, verify inventory is available as they review products.

1.3 Verification.

Verify with the customer that the order is placed and verify the data for the order, which includes name, address, ZIP codes and other data.

1.4 Update/Search/Add/Delete.

When a product is shipped, the units in the inventory need to be updated. Also it is able to retrieve the information, add or delete data from the database — inventory, product, or order information.

2.Tables

We built 5 tables. Includes Customer, Order, OrderRecord, Product, InventoryRecord table.

2.1 Product Table

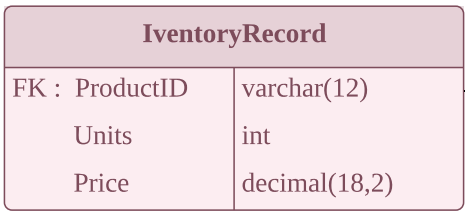
Product		
	Description	varchar(256)
	Name	varchar(32)
PK :	SKU	varchar(12)

field	introduction	constraints
Description	The information of the product	Not null
Name	Name of the product	Not null
SKU	12-character value of the form AA-NNNNNN-CC	Not null

Product table represents a product that can be purchased. It includes the name of the product, a product description, a product SKU that identifies the product. SKU is a 12-character value of the form AA-NNNNNN-CC.The primary key is SKU.

2.2 InventoryRecord Table

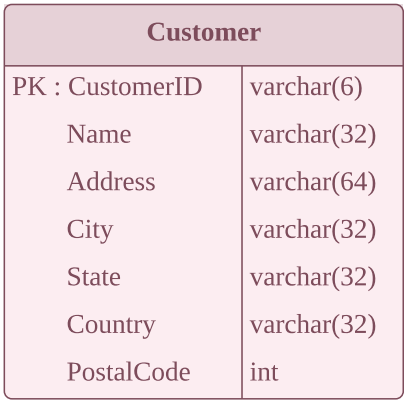
InventoryRecord is the number of units available for purchase and the price per unit for the current inventory. The primary key as well as the foreign key is the productID, which is SKU from Product table.



field	introduction	constraints
ProductID	12-character value of the form AA-NNNNNN-CC	Not null
Units	number of the product	Not null
Price	Price of the product	Not null

2.3 Customer Table

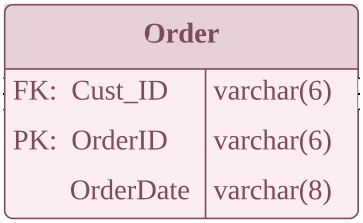
Customer table includes information of customer's name, address, city, state, country, postal code. The primary key is the 6-digit customerID.



field	introduction	constraints
Customer ID	6-character that identifies the customer	Not null
Name	number of the product	Not null
Adress	Price of the product	Not null
City	Cities around the world	Not null
State	If country is USA, judge whether it's a right state of USA, If country is not USA,	Not null
Country	USA/ Not USA	Not null
PostalCode	4-9digits	Not null

2.4 Order Table

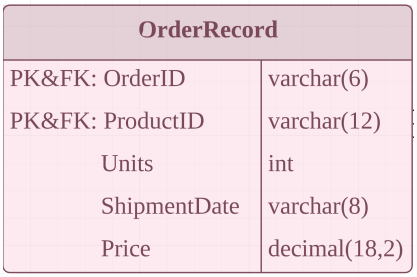
Order table is an order from a customer. It includes a customer ID, an order ID, and the order date. The primary key is the orderID, foreign key is Cust_ID from Customer table.



field	introduction	constraints
Cust_ID	6-character value that identifies the customer	Not null
OrderID	6-character value that identifies the order	Not null
OrderDate	8-character value of date	Not null

2.5 OrderRecord Table

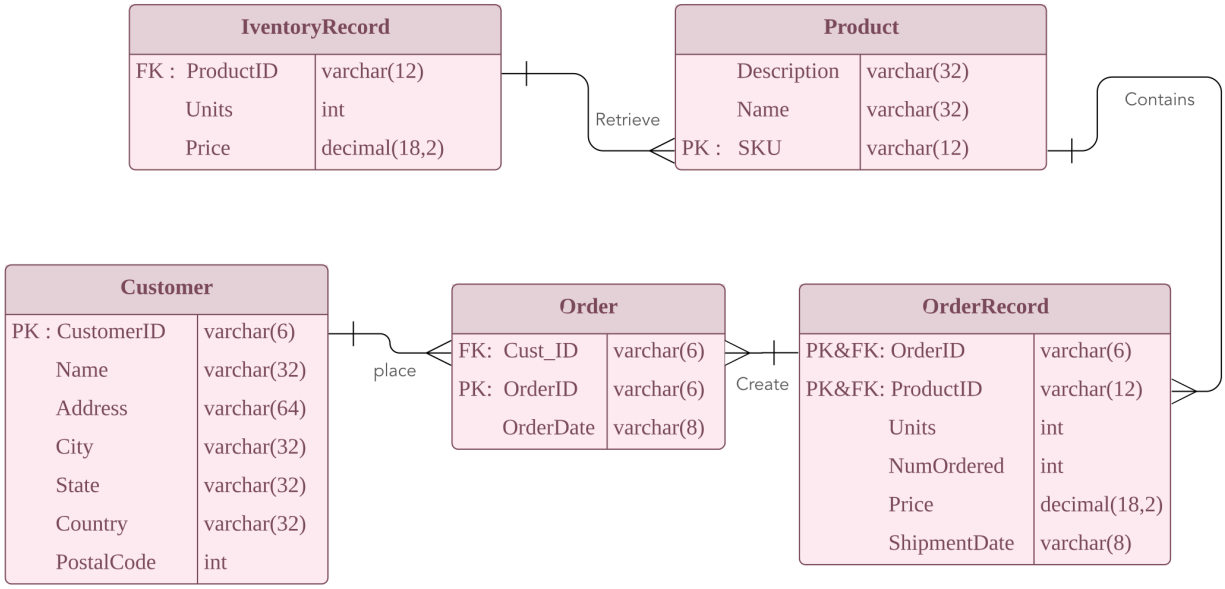
OrderRecord is the record for an item in the order. it includes the order ID, the number of units, shipment date and the unit price. When item is available, the inventory is automatically reduced, and the shipment date will be set to the order date, otherwise, the shipment date will be set to null. We put the shipment date in the orderRecord instead of order table because it is possible that not all the products in an order are in stock. We can only ship the products which are in stock. If not in stock, we cannot provide a shipment date. Therefore, the shipment date in one order may differer from each other.



field	introduction	constraints
ProductID	12-character value of the form AA-NNNNNN-CC that identifies the product	Not null
OrderID	6-character value that	Not null
Unites	Unites needed	Not null
Price	Price of the product	Not null
Shipment Date	8-character value of date, if inventory units is not available, will be set to null	Not null

2.6 Table Relationship

We have five tables. A customer can place an order, and then for each other, it may includes several products. We create an orderRecord for each product. In an orderRecord, it has a productID, which means it contains a kind of product. In the product table, we can use its SKU to retrieve the product information from the inventory table. At the same time, the inventoryRecord units will be reduced according to the order record.



3. Stored Function

Stored functions help us to check whether the input information is valid and correct. We have four stored functions.

Function	Description	Param	Return
isSKU	Judge whether input string is a valid SKU	String	Boolean
parseNum	Convert input string to a number	String	int
parsePrice	Convert input string to price (decimal)	String	BigDecimal
isState	If country is USA, judge whether it's a right state of USA, If country is not USA, return true	String	Boolean

4. Trigger

The Order Manager database includes four triggers.

4.1 Trigger UpdateOrderRecord

Purpose: The trigger is for initializing Price and ShipDate in the OrderRecord table

Implement:
Set Price equal to Price from InventoryRecord table according to the ProductID(SKU)
Set ShipDate equal to OrderDate from OrderTable table according to the OrderID

4.2 Trigger UpdateShipDate

Purpose: The trigger is for updating ShipDate in the OrderRecord table

Implement:
Set ShipDate to null if the Unit in OrderRecord table exceeds the Unit in InventoryRecord (null means pending)

4.3 Trigger UpdateInventoryRecord

Purpose: The trigger is for updating Unit in the InventoryRecord table

Implement:
Set Unit to Unit subtract Unit in OrderRecord table if the Unit in OrderRecord table doesn't exceed the Unit in InventoryRecord

4.4 Trigger DeleteOrderTable

Purpose: The trigger is for deleting an order in the OrderTable table

Implement:
The trigger will delete an order from OrderTable if this OrderID does not exist in OrderRecord anymore

5. Support API

The database includes some APIs for client to call.

5.1 Check API

Check API can be called to check the existence of an Customer, an Order or a Product

API	Description	input
containsCustomerID	check whether there is input customer in table	keyboard
containsOrderID	check whether there is input order in table	keyboard
containsProductID	check whether there is input product in table	keyboard

5.2 Insert API

API	Description	input
insertCustomer	Insert a new customer information into table	keyboard
insertOrder	Insert a new order information into table	keyboard
InsertProduct	Insert a new product information into table	keyboard

5.3 Delete API

API	Description	input
deleteCustomer	Delete the customer information from table	keyboard
deleteOrder	Delete the order information from table	keyboard
deleteProduct	Delete the product information from table	keyboard

6. Improvements

6.1 Check Validation

The present version can only check the validation of the state in America, for other countries' address, we just assume it valid. We plan to expand the validation check to cover all the countries.

6.2 Update InventoryRecord

The InventoryRecord table is static instead of dynamic, we haven't include the function to update InventoryRecord table if there are more inventory records. We plan to include it in the future version.

6.3 More Support APIs

Add more support Apis for client to call, and wrap them in a whole function for client to choose from.