

NOTE: When copying and pasting, sometimes mariadb doesn't like the curly single quote so when you paste it in just delete both curly single quotes then type back in manually should work.

The following select statements will be used to display part information for the catalog interface.

The new system has to interface with this database (details provided later). A suitable database system has to be selected for additionally needed information: such as quantity on hand for each product, and customer orders.

Use the following select statement to display all parts and their information.

```
select * from Parts;
```

Use the following select statement to display the desired part information individually.

```
select * from parts where description = 'Camshaft';
```

```
select * from parts where description = 'Bolt';
```

```
select * from parts where description = 'Nut';
```

The following will be used to display cart information for a desired cart once items are selected.

```
select number, NumSelected from CustSelItems where CartID = 1;
```

The following select statements will allow the user to select parts.

```
INSERT INTO CustSelItems (CartID, number) VALUES  
(1, 1);
```

The following select statements will be used to further the transaction process.

Use the following to validate if an account has already been created and associated with information.

```
select Email, Password FROM Account JOIN AccInfo ON Account.AccID = AccInfo.AccID
```

where Email = 'helloimunderthe@gmail.com' AND Password = '1a2s3d4f5' and
AcclsCreated = True AND AcclsValid = True;

RECOMMENDATION REGARDING SELECT STATEMENT ABOVE

Direct to page that shows account information with associated cart and parts that were selected that shows cart cost along with shipping charge cost. This is just the transaction interface.

Use the following select statement to validate if an account hasn't been already created and does not have shipping or card information associated with the account.

```
select Email, Password FROM Account JOIN AccInfo ON Account.AccID = AccInfo.AccID  
where Email = 'helloimunderthe@gmail.com' AND Password = '1a2s3d4f5' and  
AcclsCreated = False AND AcclsValid = False;
```

RECOMMENDATION REGARDING THE SELECT STATEMENT ABOVE

Direct to a page that prompts the user to create an account, payment information, and shipping information. Use the following statements to collect account information.

```
INSERT INTO Account (Email, Password, PhoneNum, AcclsCreated) VALUES  
( 'helloimunderthe@gmail.com', '1a2s3d4f5', '111-111-1111', True);
```

```
INSERT INTO Payment (FullName, CardNum, ExpDate) VALUES  
( 'Potty the Parrot', '1111-2222-3333-4444', '4/17/2028');
```

```
INSERT INTO ShippingLocation (State, City, StreetAddr, AptSuiteUnitBuilding) VALUES  
( 'Illinois', 'Rockford', '1421 IL-2', NULL);
```

Once the user inputs all of the information use a button like a submit button to set the boolean value in account to true and as well as the boolean value in AccInfo and then move on to a transaction page but before you do this count the number of rows and increment by one to set the proper id value

```
Select COUNT(AccID) from AccInfo;
```

If accID = 2 increment by one and then perform the following

```
INSERT INTO AccInfo (AccID, PayID, ShipID, AcclsValid) VALUES  
(3, 3, 3, True);
```

This will automate the process of an account being in the system.

The following will be used for the transaction interface.

Credit card authorization is done via an interface to a credit card processing system which requires the credit card number, expiration date and purchase amount. The processing system confirms with an authorization number (details provided later).

The following join statement lists all orders for a desired email.

```
select SUM(price), SUM(HandleCharge), SUM(ShipCharge) from Account JOIN AccCart ON
Account.AccID = AccCart.AccID JOIN Cart ON AccCart.CartID = Cart.CartID JOIN
CustSeleItems ON Cart.CartID = CustSeleItems.CartID JOIN parts ON
CustSeleItems.number = parts.number JOIN SHFees ON parts.number = SHFees.number
JOIN Fees ON SHFees.FeeID = Fees.FeeID where Email = 'helloimunderthe@gmail.com';
```

The following join statement lists the shipping information for a desired email.

```
select State, City, StreetAddr, AptSuiteUnitBuilding from ShippingLocation join AccInfo ON
ShippingLocation.ShipID = AccInfo.ShipID JOIN Account ON AccInfo.AccID =
Account.AccID where Email = 'helloimunderthe@gmail.com';
```

The following join statement lists the payment information for a desired email.

```
select FullName, CardNum, ExpDate from Payment join AccInfo ON Payment.PayID =
AccInfo.PayID Join Account ON AccInfo.AccID = Account.AccID where Email =
'helloimunderthe@gmail.com';
```

The following will be used to display information regarding the second interface.

A second interface to the new system will run on workstations in the warehouse: there workers can print packing lists for completed orders, retrieve the items from the warehouse, package them up, add an invoice and shipping label (both printed with the new system). Successful packing and shipping completes the order and is recorded in the order status. An email is sent to the customer confirming that the order has shipped.

Along with some shipping information/account information to create an invoice/packing label.

```
select Email, PhoneNum, ValidState, City, StreetAddr, AptSuiteUnitBuilding FROM Account
JOIN AccInfo ON Account.AccID = AccInfo.AccID JOIN ShippingLocation ON
```

AccInfo.ShipID = ShippingLocation.ShipID;

The following will be used for the warehouse interface.

A third interface also runs in the warehouse, at the receiving desk. Whenever products are delivered they are added to the inventory: they can be recognized by their description or part number. Their quantity on hand is updated. Note that the legacy product database does not contain inventory information.

The description of this states that they(parts) can be recognized by their description or part number so im guessing its just gonna be the select statement below. Therefore, the user will use the select statement below to select a part name by the part number and description.

```
select * from parts where number = 2  
select * from parts where description = 'Nut';
```

Note: If user decides to look up part by description use the following to acquire the number. This will be done in the background.

```
select number from parts where description = 'Nut';
```

Then use the update statement below to allow the user to update the quantity on hand.

```
UPDATE ProdInventory SET QInStock = 70 WHERE InventoryID = 2;
```

The following will be used for the admin interface.

There will be an administrative interface that allows to set the shipping and handling charges, as well as view all orders. Shipping and handling charges are based on the weight of a complete order. This interface allows to set the weight brackets and their charges. Orders can be searched based on date range, status (authorized, shipped) or prize range. The complete order detail is displayed for a selected order.

The following join statement lists all orders regarding an associated email.

```
select description, price, HandleCharge, ShipCharge from Account JOIN AccCart ON  
Account.AccID = AccCart.AccID JOIN Cart ON AccCart.CartID = Cart.CartID JOIN
```

CustSeleltems ON Cart.CartID = CustSeleltems.CartID JOIN parts ON
CustSeleltems.number = parts.number JOIN SHFees ON parts.number = SHFees.number
JOIN Fees ON SHFees.FeeID = Fees.FeeID where Email = 'helloimunderthe@gmail.com';

The following join statement will list all orders regardless of email.

select Email, description, price, HandleCharge, ShipCharge from Account JOIN AccCart
ON Account.AccID = AccCart.AccID JOIN Cart ON AccCart.CartID = Cart.CartID JOIN
CustSeleltems ON Cart.CartID = CustSeleltems.CartID JOIN parts ON
CustSeleltems.number = parts.number JOIN SHFees ON parts.number = SHFees.number
JOIN Fees ON SHFees.FeeID = Fees.FeeID;

The following join statement will list shipping and handling charges for each part along with their associated weight.

select description, price, Weight, HandleCharge, ShipCharge from parts JOIN SHFees ON
parts.number = SHFees.number JOIN Fees ON SHFees.FeeID = Fees.FeeID;

The following will display orders based on date range, status, (authorized or shipped) or price range for a specified email.

select TransactionTime, CurDate, Authorized, Shipped, SUM(price), SUM(HandleCharge),
SUM(ShipCharge) from OrderStatus JOIN Account ON OrderStatus.AccID = Account.AccID
JOIN AccCart ON Account.AccID = AccCart.AccID JOIN Cart ON AccCart.CartID =
Cart.CartID JOIN CustSeleltems ON Cart.CartID = CustSeleltems.CartID JOIN parts ON
parts.number = CustSeleltems.number JOIN SHFees ON parts.number = SHFees.number
JOIN Fees ON SHFees.FeeID = Fees.FeeID where Email = helloimunderthe@gmail.com;

The following will display all orders based on date range, status, weight, (authorized or shipped) or price range for all emails.

select Email, TransactionTime, CurDate, Authorized, Shipped, SUM(Weight), SUM(price),
SUM(HandleCharge), SUM(ShipCharge) from OrderStatus JOIN Account ON
OrderStatus.AccID = Account.AccID JOIN AccCart ON Account.AccID = AccCart.AccID
JOIN Cart ON AccCart.CartID = Cart.CartID JOIN CustSeleltems ON Cart.CartID =
CustSeleltems.CartID JOIN parts ON parts.number = CustSeleltems.number JOIN SHFees
ON parts.number = SHFees.number JOIN Fees ON SHFees.FeeID = Fees.FeeID GROUP BY
Email ORDER BY SUM(Weight), SUM(price), SUM(HandleCharge), SUM(ShipCharge) DESC;

The following will allow the admin to set the shipping and handling charges to a specified part.

update Fees set HandleCharge = 101.00 where FeeID = 1;