DATABASE DESIGN CONCEPTS

PROJECT

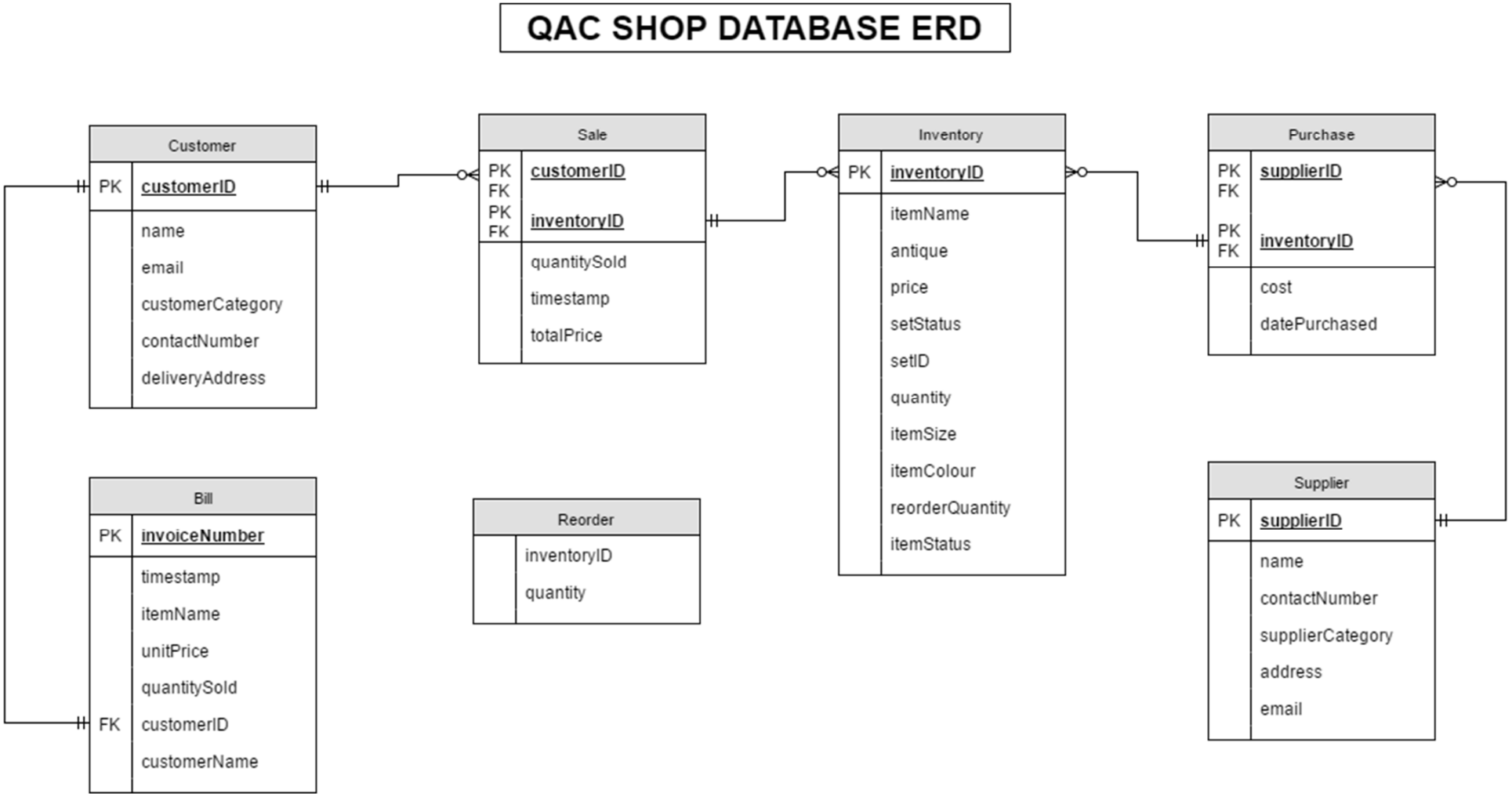
May 2017

Sydney Twigg
M8C3XRSN8

CONTENTS

DATABASE PLANNING

ERD



COLUMN PROPERTIES

Customer					
Column Name	Data Type (length)	Key	Required	Default Value	Remarks
customerID	int	Primary	Yes	-	Unique
name	varchar (50)	No	Yes	-	
email	varchar (50)	No	Yes	-	
customerCategory	varchar (100)	No	Yes	-	
contactNumber	varchar (13)	No	Yes	-	Accounts for international calling codes. E.g. +44 7983 205807
deliveryAddress	varchar (400)	No	Yes	-	

Supplier					
Column Name	Data Type(length)	Key	Required	Default Value	Remarks
supplierID	int	Primary	Yes	-	Unique
name	varchar (50)	No	Yes	-	
contactNumber	varchar(13)	No	Yes	-	
supplierCategory	varchar(100)	No	Yes	-	
address	varchar(400)	No	Yes	-	
email	varchar(50)	No	Yes	-	

Bill					
Column Name	Data Type(length)	Key	Required	Default Value	Remarks
invoiceNumber	int	Primary	Yes	-	Unique
timestamp	datetime	No	Yes	-	
itemName	varchar(500)	No	Yes	-	
unitPrice	double	No	Yes	-	
quantitySold	int	No	Yes	-	
customerID	int	Foreign Key	Yes	-	REF: customerID in customer
customerName	varchar(50)	No	Yes	-	

Inventory					
Column Name	Data Type(length)	Key	Required	Default Value	Remarks
inventoryID	int	Primary	Yes	-	Unique
itemName	varchar(100)	No	Yes	-	
antique	bit	No	Yes	0	Functions as a boolean field, 0 = false, 1 = true.
price	double	No	Yes	-	
setStatus	bit	No	Yes	0	Functions as a boolean field, 0 = false, 1 = true.
setID	int	No	No	-	unique
quantity	int	No	Yes	-	
itemSize	varchar(20)	No	No	-	
itemColour	varchar(20)	No	No	-	
reorderQuantity	int	No	No	-	
itemStatus	varchar(10)	No	Yes	-	determines if item is in stock or not

Purchase					
Column Name	Data Type(length)	Key	Required	Default Value	Remarks
supplierID	int	Primary/foreign	Yes	-	REF: supplierID in supplier
cost	double	No	Yes	-	
datePurchased	date	No	Yes	-	
inventoryID	int	Primary/foreign	Yes	-	REF: inventoryID in inventory

Sale					
Column Name	Data Type(length)	Key	Required	Default Value	Remarks
customerID	int	Primary/foreign	Yes	-	REF: customerID in customer
inventoryID	int	Primary/foreign	Yes	-	REF: inventoryID in inventory
quantitySold	int	No	Yes	-	
timestamp	timestamp	No	Yes	-	
totalPrice	double	No	Yes	-	

Reorder					
Column Name	Data Type(length)	Key	Required	Default Value	Remarks
inventoryID	int	No	Yes	-	
quantity	int	No	Yes	-	

COMMAND LINE SCREENSHOTS

CREATE DATABASE & TABLES AND POPULATE TABLES

```
mysql> CREATE DATABASE IF NOT EXISTS QAC_SHOP_m8c3xrsn8;
Query OK, 1 row affected (0.01 sec)

mysql> USE QAC_SHOP_m8c3xrsn8;
Database changed
```

CUSTOMER

```
mysql> CREATE TABLE customer(
    -> customerID INT NOT NULL AUTO_INCREMENT,
    -> name VARCHAR(50) NOT NULL,
    -> email VARCHAR(50) NOT NULL,
    -> customerCategory VARCHAR(100) NOT NULL,
    -> contactNumber VARCHAR(13) NOT NULL,
    -> deliveryAddress VARCHAR(400) NOT NULL,
    -> PRIMARY KEY(customerID));
Query OK, 0 rows affected (0.36 sec)

mysql> INSERT INTO customer(name, email, customerCategory, contactNumber, deliveryAddress)
    -> VALUES('Sydney Twigg', 'sydneytwigg@gmail.com', 'individual', '+27832567264', '6 Mohr Road, Tokai, Cape Town, 7945');
Query OK, 1 row affected (0.05 sec)

mysql> INSERT INTO customer(name, email, customerCategory, contactNumber, deliveryAddress)
    -> VALUES('Alex Smith', 'asmith@interiors.co.za', 'interior designer', '+27839877624', '188 Main Road, Plumstead, Cape Town, 7945');
Query OK, 1 row affected (0.05 sec)

mysql> DESCRIBE customer;
+-----------------+-------------+------+-----+---------+----------------+
| Field           | Type        | Null | Key | Default | Extra          |
+-----------------+-------------+------+-----+---------+----------------+
| customerID      | int(11)     | NO   | PRI | NULL    | auto_increment |
| name            | varchar(50) | NO   |     | NULL    |                |
| email           | varchar(50) | NO   |     | NULL    |                |
| customerCategory| varchar(100)| NO   |     | NULL    |                |
| contactNumber   | varchar(13) | NO   |     | NULL    |                |
| deliveryAddress | varchar(400)| NO   |     | NULL    |                |
+-----------------+-------------+------+-----+---------+----------------+
6 rows in set (0.00 sec)

mysql> SELECT * FROM customer;
+------------+-------------+-----------------------+------------------+--------------+-----------------------------------------+
| customerID | name        | email                 | customerCategory | contactNumber | deliveryAddress                         |
+------------+-------------+-----------------------+------------------+--------------+-----------------------------------------+
|          1 | Sydney Twigg | sydneytwigg@gmail.com | individual       | +27832567264 | 6 Mohr Road, Tokai, Cape Town, 7945     |
|          2 | Alex Smith  | asmith@interiors.co.za | interior designer | +27839877624 | 188 Main Road, Plumstead, Cape Town, 7945 |
+------------+-------------+-----------------------+------------------+--------------+-----------------------------------------+
2 rows in set (0.00 sec)
```

SUPPLIER

```
mysql> CREATE TABLE supplier(
    -> supplierID INT NOT NULL AUTO_INCREMENT,
    -> name VARCHAR(50) NOT NULL,
    -> contactNumber VARCHAR(13) NOT NULL,
    -> supplierCategory VARCHAR(100) NOT NULL,
    -> address VARCHAR(400) NOT NULL,
    -> email VARCHAR(50) NOT NULL,
    -> PRIMARY KEY(supplierID));
Query OK, 0 rows affected (0.33 sec)

mysql> INSERT INTO supplier(name, contactNumber, supplierCategory, address, email)
    -> VALUES ('House & Home', '0218765967',
    -> 'distributor', '18 Business Park Road, Business Park, Milnerton, 7654', 'milnerton@houseandhome.co.za');
Query OK, 1 row affected (0.18 sec)

mysql> INSERT INTO supplier(name, contactNumber, supplierCategory, address, email)
    -> VALUES ('John Smith', '+27867567211','individual', '18 Ladies Mile Road, Bergvliet, Cape Town, 7945', 'jsmith@gmail.com');
Query OK, 1 row affected (0.03 sec)

mysql> DESCRIBE supplier;
+-----------------+-------------+------+-----+---------+----------------+
| Field           | Type        | Null | Key | Default | Extra          |
+-----------------+-------------+------+-----+---------+----------------+
| supplierID      | int(11)     | NO   | PRI | NULL    | auto_increment |
| name            | varchar(50) | NO   |     | NULL    |                |
| contactNumber   | varchar(13) | NO   |     | NULL    |                |
| supplierCategory| varchar(100)| NO   |     | NULL    |                |
| address         | varchar(400)| NO   |     | NULL    |                |
| email           | varchar(50) | NO   |     | NULL    |                |
+-----------------+-------------+------+-----+---------+----------------+
6 rows in set (0.00 sec)

mysql> SELECT * FROM supplier;
+------------+-------------+---------------+-----------------+-----------------------------------------------------+-----------------------------+
| supplierID | name        | contactNumber | supplierCategory | address                                             | email                       |
+------------+-------------+---------------+-----------------+-----------------------------------------------------+-----------------------------+
|          1 | House & Home | 0218765967    | distributor     | 18 Business Park Road, Business Park, Milnerton, 7654 | milnerton@houseandhome.co.za |
|          2 | John Smith  | +27867567211  | individual      | 18 Ladies Mile Road, Bergvliet, Cape Town, 7945     | jsmith@gmail.com            |
+------------+-------------+---------------+-----------------+-----------------------------------------------------+-----------------------------+
2 rows in set (0.00 sec)
```

BILL

```
mysql> CREATE TABLE bill(
    -> invoiceNumber INT NOT NULL AUTO_INCREMENT,
    -> timestamp DATETIME NOT NULL,
    -> itemName VARCHAR(500) NOT NULL,
    -> unitPrice DOUBLE NOT NULL,
    -> quantitySold INT NOT NULL,
    -> customerID INT NOT NULL,
    -> PRIMARY KEY(invoiceNumber));
Query OK, 0 rows affected (0.32 sec)

mysql> DESCRIBE bill;
+---------------+--------------+------+-----+---------+----------------+
| Field         | Type         | Null | Key | Default | Extra          |
+---------------+--------------+------+-----+---------+----------------+
| invoiceNumber | int(11)      | NO   | PRI | NULL    | auto_increment |
| timestamp     | datetime     | NO   |     | NULL    |                |
| itemName      | varchar(500) | NO   |     | NULL    |                |
| unitPrice     | double       | NO   |     | NULL    |                |
| quantitySold  | int(11)      | NO   |     | NULL    |                |
| customerID    | int(11)      | NO   |     | NULL    |                |
+---------------+--------------+------+-----+---------+----------------+
6 rows in set (0.00 sec)
```

INVENTORY

```
mysql> CREATE TABLE inventory(
    -> inventoryID INT NOT NULL AUTO_INCREMENT,
    -> itemName VARCHAR(100) NOT NULL,
    -> antique BIT NOT NULL DEFAULT 0,
    -> price DOUBLE NOT NULL,
    -> setStatus BIT NOT NULL DEFAULT 0,
    -> setID INT,
    -> quantity INT NOT NULL,
    -> itemSize VARCHAR(100),
    -> itemColour VARCHAR(100),
    -> reorderQuantity INT,
    -> itemStatus VARCHAR(50) NOT NULL,
    -> PRIMARY KEY(inventoryID));
Query OK, 0 rows affected (0.23 sec)

mysql> INSERT INTO inventory(itemName, antique, price, setStatus, quantity, itemStatus)
    -> VALUES ('Oak Table', 1, '2600.00', 0, 1, 'available');
Query OK, 1 row affected (0.04 sec)

mysql> INSERT INTO inventory(itemName, antique, price, setStatus, quantity, itemSize, itemColour, reorderQuantity, itemStatus)
    -> VALUES ('Embroided Table Cloth', 0, 200.00, 0, 14, '300x200', 'cream/blue', '10', 'available');
Query OK, 1 row affected (0.07 sec)

mysql> INSERT INTO inventory(itemName, antique, price, setStatus, quantity, itemSize, itemColour, reorderQuantity, itemStatus)
    -> VALUES ('Embroided Table Cloth', 0, 200.00, 0, 0, '300x200', 'cream/pink', '30', 'out of stock');
Query OK, 1 row affected (0.04 sec)
```

```
mysql> INSERT INTO inventory(itemName, antique, price, setStatus, setID, quantity, itemStatus)
    -> VALUES ('Whicker Chair', 1, 600.00, 1, 1, 6, 'available');
Query OK, 1 row affected (0.07 sec)

mysql> DESCRIBE inventory
    -> ;
+-----------------+-------------+------+-----+---------+----------------+
| Field           | Type        | Null | Key | Default | Extra          |
+-----------------+-------------+------+-----+---------+----------------+
| inventoryID     | int(11)     | NO   | PRI | NULL    | auto_increment |
| itemName        | varchar(100)| NO   |     | NULL    |                |
| antique         | bit(1)      | NO   |     | b'0'    |                |
| price           | double      | NO   |     | NULL    |                |
| setStatus       | bit(1)      | NO   |     | b'0'    |                |
| setID           | int(11)     | YES  |     | NULL    |                |
| quantity        | int(11)     | NO   |     | NULL    |                |
| itemSize        | varchar(100)| YES  |     | NULL    |                |
| itemColour      | varchar(100)| YES  |     | NULL    |                |
| reorderQuantity | int(11)     | YES  |     | NULL    |                |
| itemStatus      | varchar(50) | NO   |     | NULL    |                |
+-----------------+-------------+------+-----+---------+----------------+
11 rows in set (0.00 sec)

mysql> SELECT * FROM inventory;
+-------------+----------------------+---------+-------+-----------+-------+----------+----------+------------+-----------------+--------------+
| inventoryID | itemName             | antique | price | setStatus | setID | quantity | itemSize | itemColour | reorderQuantity | itemStatus   |
+-------------+----------------------+---------+-------+-----------+-------+----------+----------+------------+-----------------+--------------+
|           1 | Oak Table            | ☒       |  2600 |           |  NULL |        1 | NULL     | NULL       |            NULL | available    |
|           2 | Embroided Table Cloth |         |   200 |           |  NULL |       14 | 300x200  | cream/blue |              10 | available    |
|           3 | Embroided Table Cloth |         |   200 |           |  NULL |        0 | 300x200  | cream/pink |              30 | out of stock |
|           4 | Whicker Chair        | ☒       |   600 | ☒         |     1 |        6 | NULL     | NULL       |            NULL | available    |
+-------------+----------------------+---------+-------+-----------+-------+----------+----------+------------+-----------------+--------------+
4 rows in set (0.00 sec)
```

PURCHASE

```
mysql> CREATE TABLE purchase(
    -> supplierID INT NOT NULL,
    -> inventoryID INT NOT NULL,
    -> cost DOUBLE NOT NULL,
    -> datePurchased DATE NOT NULL,
    -> PRIMARY KEY(supplierID, inventoryID));
Query OK, 0 rows affected (0.27 sec)

mysql> INSERT INTO purchase(supplierID, inventoryID, cost, datePurchased)
    -> VALUES(1, 2, 50.00, 2016-01-21);
ERROR 1292 (22007): Incorrect date value: '1994' for column 'datePurchased' at row 1
mysql> INSERT INTO purchase(supplierID, inventoryID, cost, datePurchased)
    -> VALUES(1, 2, 50.00, '2016-01-21');
Query OK, 1 row affected (0.10 sec)

mysql> INSERT INTO purchase(supplierID, inventoryID, cost, datePurchased)
    -> VALUES(2, 1, 1600.00, '2017-05-01');
Query OK, 1 row affected (0.07 sec)

mysql> DESCRIBE purchase;
+---------------+---------+------+-----+---------+-------+
| Field         | Type    | Null | Key | Default | Extra |
+---------------+---------+------+-----+---------+-------+
| supplierID    | int(11) | NO   | PRI | NULL    |       |
| inventoryID   | int(11) | NO   | PRI | NULL    |       |
| cost          | double  | NO   |     | NULL    |       |
| datePurchased | date    | NO   |     | NULL    |       |
+---------------+---------+------+-----+---------+-------+
4 rows in set (0.00 sec)

mysql> SELECT * FROM purchase;
+------------+-------------+------+---------------+
| supplierID | inventoryID | cost | datePurchased |
+------------+-------------+------+---------------+
|          1 |           2 |   50 | 2016-01-21    |
|          2 |           1 | 1600 | 2017-05-01    |
+------------+-------------+------+---------------+
2 rows in set (0.00 sec)

mysql>
```

SALE

```
mysql> CREATE TABLE sale(
    -> customerID INT NOT NULL,
    -> inventoryID INT NOT NULL,
    -> quantitySold INT NOT NULL,
    -> totalPrice DOUBLE NOT NULL,
    -> timestamp DATETIME NOT NULL,
    -> PRIMARY KEY(customerID, inventoryID));
Query OK, 0 rows affected (0.26 sec)

mysql> DESCRIBE sale;
+--------------+----------+------+-----+---------+-------+
| Field        | Type     | Null | Key | Default | Extra |
+--------------+----------+------+-----+---------+-------+
| customerID   | int(11)  | NO   | PRI | NULL    |       |
| inventoryID  | int(11)  | NO   | PRI | NULL    |       |
| quantitySold | int(11)  | NO   |     | NULL    |       |
| totalPrice   | double   | NO   |     | NULL    |       |
| timestamp    | datetime | NO   |     | NULL    |       |
+--------------+----------+------+-----+---------+-------+
5 rows in set (0.00 sec)
```

CREATE RELATIONSHIPS AND SET FOREIGN KEYS

PURCHASE

```
mysql> ALTER TABLE purchase
    -> ADD FOREIGN KEY(supplierID)
    -> REFERENCES supplier(supplierID);
Query OK, 2 rows affected (0.92 sec)
Records: 2  Duplicates: 0  Warnings: 0

mysql> ALTER TABLE purchase
    -> ADD FOREIGN KEY(inventoryID)
    -> REFERENCES inventory(inventoryID);
Query OK, 2 rows affected (0.90 sec)
Records: 2  Duplicates: 0  Warnings: 0
```

SALE

```
mysql> ALTER TABLE sale
    -> ADD FOREIGN KEY(inventoryID)
    -> REFERENCES inventory(inventoryID);
Query OK, 0 rows affected (0.71 sec)
Records: 0  Duplicates: 0  Warnings: 0

mysql> ALTER TABLE sale
    -> ADD FOREIGN KEY(customerID)
    -> REFERENCES customer(customerID);
Query OK, 0 rows affected (0.68 sec)
Records: 0  Duplicates: 0  Warnings: 0
```

BILL

```
mysql> ALTER TABLE bill
    -> ADD FOREIGN KEY(customerID)
    -> REFERENCES customer(customerID);
Query OK, 0 rows affected (0.79 sec)
Records: 0  Duplicates: 0  Warnings: 0
```

TASK 1

THE SYSTEM SHOULD BE ABLE TO KEEP THE RECORDS OF THE SALES THAT HAVE BEEN MADE

The system keeps record of sales made in the table 'sale'. For every sale made, the customer ID (customerID) and item sold (inventoryID) is entered into the database, along with the quantity of the item, the time of the sale, and the total price of the sale.

The following screenshot shows the table data for the 'sale' table:

```
mysql> DESCRIBE sale;
+--------------+----------+------+-----+---------+-------+
| Field        | Type     | Null | Key | Default | Extra |
+--------------+----------+------+-----+---------+-------+
| customerID   | int(11)  | NO   | PRI | NULL    |       |
| inventoryID  | int(11)  | NO   | PRI | NULL    |       |
| quantitySold | int(11)  | NO   |     | NULL    |       |
| totalPrice   | double   | NO   |     | NULL    |       |
| timestamp    | datetime | NO   |     | NULL    |       |
+--------------+----------+------+-----+---------+-------+
5 rows in set (0.01 sec)
```

The following screenshot shows that sales information can be added to the database, using an INSERT INTO query:

```
mysql> INSERT INTO sale(customerID, inventoryID, quantitySold, totalPrice, timestamp)
    -> VALUES (1, 2, 4, 800.00, '2017-05-01 17:07:55');
Query OK, 1 row affected (0.07 sec)
```

The following screenshot shows that sales information is stored within the table, using a SELECT query:

```
mysql> SELECT * FROM sale;
+------------+-------------+--------------+------------+---------------------+
| customerID | inventoryID | quantitySold | totalPrice | timestamp           |
+------------+-------------+--------------+------------+---------------------+
|          1 |           1 |            1 |       2600 | 2017-04-30 15:37:18 |
|          1 |           2 |            4 |        800 | 2017-05-01 17:07:55 |
|          2 |           2 |            1 |        200 | 2017-05-09 09:13:12 |
|          2 |           4 |            6 |       3600 | 2017-05-09 09:13:12 |
+------------+-------------+--------------+------------+---------------------+
4 rows in set (0.00 sec)
```

From the data stored within the 'sale' table various calculations and data can be determined:

```
mysql> SELECT SUM(totalPrice) FROM sale WHERE inventoryID = 2;
+-----------------+
| SUM(totalPrice) |
+-----------------+
|            1000 |
+-----------------+
1 row in set (0.00 sec)
```

The total amount made in sales of Cream & Blue Embroided Table Cloth's (inventoryID 2)

```
mysql> SELECT SUM(totalPrice) FROM sale WHERE customerID = 2;
+-----------------+
| SUM(totalPrice) |
+-----------------+
|            3800 |
+-----------------+
1 row in set (0.00 sec)
```

The total amount Alex Smith (customerID 2) has spent at the QAC shop.

TASK 2:

CUSTOMERS SHOULD BE ABLE TO SEE THE AVAILABILITY OF THE PRODUCTS ON THE DATABASE

The table 'inventory' stores all products in the store, along with the quantity remaining, as well as the status of whether the item is available, out of stock, or reordered. This allows customers to see the availability of the products.

The following screenshot shows the table data for the 'inventory' table:

```
mysql> DESCRIBE inventory;
+-----------------+--------------+------+-----+---------+----------------+
| Field           | Type         | Null | Key | Default | Extra          |
+-----------------+--------------+------+-----+---------+----------------+
| inventoryID     | int(11)      | NO   | PRI | NULL    | auto_increment |
| itemName        | varchar(100) | NO   |     | NULL    |                |
| antique         | bit(1)       | NO   |     | b'0'    |                |
| price           | double       | NO   |     | NULL    |                |
| setStatus       | bit(1)       | NO   |     | b'0'    |                |
| setID           | int(11)      | YES  |     | NULL    |                |
| quantity        | int(11)      | NO   |     | NULL    |                |
| itemSize        | varchar(100) | YES  |     | NULL    |                |
| itemColour      | varchar(100) | YES  |     | NULL    |                |
| reorderQuantity | int(11)      | YES  |     | NULL    |                |
| itemStatus      | varchar(50)  | NO   |     | NULL    |                |
+-----------------+--------------+------+-----+---------+----------------+
```

The following screenshot shows the products stored in the table:

```
mysql> SELECT * FROM inventory;
+-------------+---------------------+---------+-------+-----------+-------+----------+-------------+-----------+-----------------+-------------+
| inventoryID | itemName            | antique | price | setStatus | setID | quantity | itemSize    | itemColour | reorderQuantity | itemStatus  |
+-------------+---------------------+---------+-------+-----------+-------+----------+-------------+-----------+-----------------+-------------+
|           1 | Oak Table           |         |  2600 |           |  NULL |        1 | NULL        | NULL      |            NULL | available   |
|           2 | Embroided Table Cloth |       |   200 |           |  NULL |       14 | 300x200     | cream/blue |              10 | available   |
|           3 | Embroided Table Cloth |       |   200 |           |  NULL |        0 | 300x200     | cream/pink |              30 | out of stock |
|           4 | Whicker Chair       |         |   600 |           |  NULL |        6 | NULL        | NULL      |            NULL | available   |
|           5 | Lampshade           |         |   400 |           |  NULL |        0 | Large       | Beige     |              15 | reordered   |
|           6 | Decorative Cushion  |         |   325 |           |  NULL |        0 | 40cm x 40cm | Navy/White |               0 | out of stock |
+-------------+---------------------+---------+-------+-----------+-------+----------+-------------+-----------+-----------------+-------------+
6 rows in set (0.00 sec)
```

The following screenshots show various queries to show the availability of items:

```
mysql> SELECT inventoryID,itemName,quantity FROM inventory WHERE itemStatus = 'available';
+-------------+---------------------+----------+
| inventoryID | itemName            | quantity |
+-------------+---------------------+----------+
|           1 | Oak Table           |        1 |
|           2 | Embroided Table Cloth |       14 |
|           4 | Whicker Chair       |        6 |
+-------------+---------------------+----------+
3 rows in set (0.00 sec)
```

This shows the products and how many are in stock that are currently available.

```
mysql> SELECT itemName, itemStatus FROM inventory WHERE inventoryID = 6;
+--------------------+--------------+
| itemName           | itemStatus   |
+--------------------+--------------+
| Decorative Cushion | out of stock |
+--------------------+--------------+
1 row in set (0.00 sec)
```

This shows how the availability of an item can be determined with the product's ID

```
mysql> SELECT itemName, reorderQuantity FROM inventory WHERE itemStatus = 'reordered';
+-----------+-----------------+
| itemName  | reorderQuantity |
+-----------+-----------------+
| Lampshade |              15 |
+-----------+-----------------+
1 row in set (0.00 sec)
```

This shows the amount of stock that has been reordered

```
mysql> SELECT itemName, quantity, itemStatus FROM inventory WHERE reorderQuantity > 0;
+----------------------+----------+--------------+
| itemName             | quantity | itemStatus   |
+----------------------+----------+--------------+
| Embroided Table Cloth |       14 | available    |
| Embroided Table Cloth |        0 | out of stock |
| Lampshade            |        0 | reordered    |
+----------------------+----------+--------------+
3 rows in set (0.00 sec)
```

This shows the items that have been reordered, as well as the current stock and status of the product.

```
mysql> SELECT inventoryID, itemName FROM inventory WHERE quantity > 0;
+-------------+------------------------+
| inventoryID | itemName               |
+-------------+------------------------+
|           1 | Oak Table              |
|           2 | Embroided Table Cloth  |
|           4 | Whicker Chair          |
+-------------+------------------------+
3 rows in set (0.00 sec)
```

This shows all products that are currently in stock at the shop.

TASK 3:

QAC SHOULD BE ABLE TO PURCHASE ANTIQUES FROM BOTH INDIVIDUALS, WHOLESALERS AND NEW ITEMS FROM DISTRIBUTORS

The table ‘supplier’ stores all information relating to the suppliers of the QAC Shop, including individuals, wholesalers and distributors - which are designated within the ‘supplierCategory’ column. The table ‘purchase’ stores all information relating to what products have been purchased, and from whom and how much for.

The following screenshot shows the table information for the ‘supplier’ and ‘purchase’ tables.

```
mysql> DESCRIBE supplier;
+------------------+--------------+------+-----+---------+----------------+
| Field            | Type         | Null | Key | Default | Extra          |
+------------------+--------------+------+-----+---------+----------------+
| supplierID       | int(11)      | NO   | PRI | NULL    | auto_increment |
| name             | varchar(50)  | NO   |     | NULL    |                |
| contactNumber    | varchar(13)  | NO   |     | NULL    |                |
| supplierCategory | varchar(100) | NO   |     | NULL    |                |
| address          | varchar(400) | NO   |     | NULL    |                |
| email            | varchar(50)  | NO   |     | NULL    |                |
+------------------+--------------+------+-----+---------+----------------+
6 rows in set (0.00 sec)

mysql> DESCRIBE purchase;
+---------------+---------+------+-----+---------+-------+
| Field         | Type    | Null | Key | Default | Extra |
+---------------+---------+------+-----+---------+-------+
| supplierID    | int(11) | NO   | PRI | NULL    |       |
| inventoryID   | int(11) | NO   | PRI | NULL    |       |
| cost          | double  | NO   |     | NULL    |       |
| datePurchased | date    | NO   |     | NULL    |       |
+---------------+---------+------+-----+---------+-------+
4 rows in set (0.00 sec)
```

The following screenshot shows the supplier details stored, including each category of supplier.

```
mysql> SELECT * FROM supplier;
+------------+---------------------+---------------+-----------------+-----------------------------------------------------+---------------------------+
| supplierID | name                | contactNumber | supplierCategory | address                                             | email                     |
+------------+---------------------+---------------+-----------------+-----------------------------------------------------+---------------------------+
|          1 | House & Home        | 0218765967    | distributor      | 18 Business Park Road, Business Park, Milnerton, 7654 | milnerton@houseandhome.co.za |
|          2 | John Smith          | +27867567211  | individual       | 18 Ladies Mile Road, Bergvliet, Cape Town, 7945     | jsmith@gmail.com          |
|          3 | Taylor & Sons Antiques | 0315646964    | wholesaler       | 38 Smith Street, Stellenbosch, Cape Town, 1080      | sales@taylorandsons.co.za |
+------------+---------------------+---------------+-----------------+-----------------------------------------------------+---------------------------+
3 rows in set (0.00 sec)
```

The following screenshots show purchases from each type of supplier being added:

```
mysql> INSERT INTO purchase(supplierID, inventoryID, cost, datePurchased)
    -> VALUES(3, 4, 1800.00, '2017-03-12')
    -> ;
Query OK, 1 row affected (0.07 sec)
```

Purchase from wholesale supplier 'Taylor & Sons Antiques' for a set of 6 antique wicker chairs (inventoryID 4)

```
mysql> INSERT INTO purchase(supplierID, inventoryID, cost, datePurchased)
    -> VALUES(1, 5, 200.00, '2016-08-20');
Query OK, 1 row affected (0.07 sec)
```

Purchase from distributor 'House & Home' for a lampshade (InventoryID 5)

```
mysql> INSERT INTO purchase(supplierID, inventoryID, cost, datePurchased)
    -> VALUES(2, 1, 1600.00, '2017-05-01');
Query OK, 1 row affected (0.07 sec)
```

Purchase from an individual, John Smith, for an antique oak table.

The following screenshot shows all purchases, one of which from each type of supplier.

```
mysql> SELECT * FROM purchase;
+------------+-------------+------+---------------+
| supplierID | inventoryID | cost | datePurchased |
+------------+-------------+------+---------------+
|          1 |           2 |   50 | 2016-01-21    |
|          1 |           5 |  200 | 2016-08-20    |
|          2 |           1 | 1600 | 2017-05-01    |
|          3 |           4 | 1800 | 2017-03-12    |
+------------+-------------+------+---------------+
4 rows in set (0.00 sec)
```

The following screenshots show the products bought by specific types of suppliers:

```
mysql> SELECT A.inventoryID, C.itemName, B.supplierCategory
    -> FROM ((purchase A
    -> INNER JOIN supplier B on A.supplierID = B.supplierID)
    -> INNER JOIN inventory C on A.inventoryID = C.inventoryID)
    -> WHERE supplierCategory = 'wholesaler';
+-------------+---------------+-----------------+
| inventoryID | itemName      | supplierCategory |
+-------------+---------------+-----------------+
|           4 | Whicker Chair | wholesaler      |
+-------------+---------------+-----------------+
1 row in set (0.00 sec)
```

This shows all products purchased from wholesalers.

```
mysql> SELECT A.inventoryID, C.itemName, B.supplierCategory
    -> FROM ((purchase A
    -> INNER JOIN supplier B on A.supplierID = B.supplierID)
    -> INNER JOIN inventory C on A.inventoryID = C.inventoryID)
    -> WHERE supplierCategory = 'individual';
+-------------+-----------+-----------------+
| inventoryID | itemName  | supplierCategory |
+-------------+-----------+-----------------+
|           1 | Oak Table | individual      |
+-------------+-----------+-----------------+
1 row in set (0.00 sec)
```

This shows all products purchased from individuals

```
mysql> SELECT A.inventoryID, C.itemName, B.supplierCategory
    -> FROM ((purchase A
    -> INNER JOIN supplier B on A.supplierID = B.supplierID)
    -> INNER JOIN inventory C on A.inventoryID = C.inventoryID)
    -> WHERE supplierCategory = 'distributor';
+-------------+----------------------+-----------------+
| inventoryID | itemName             | supplierCategory |
+-------------+----------------------+-----------------+
|           2 | Embroided Table Cloth | distributor     |
|           5 | Lampshade            | distributor     |
+-------------+----------------------+-----------------+
2 rows in set (0.00 sec)
```

This shows all products purchased from distributors

TASK 4:

THE SYSEM SHOULD BE ABLE TO REORDER ITEMS THAT ARE OUT OF STOCK

A table, 'reorder', must be created to hold the triggered reorder items. This table will store the product ID and amount to be reordered from the supplier, the amount to be reordered is predetermined in the 'reorderQuantity' column in the 'inventory' table. When a sale takes place, there is a trigger in place to decrease the stock quantity 'quantity' by the amount purchased, when this quantity reaches below 1 it sets off the reorder trigger, which then adds the reorder information into the table 'reorder'.

Create table to store reorder items

```
mysql> CREATE TABLE IF NOT EXISTS reorder(
    -> inventoryID INT NOT NULL,
    -> quantity INT NOT NULL,
    -> PRIMARY KEY (inventoryID));
    -> //
Query OK, 0 rows affected (0.46 sec)
```

Trigger to decrease stock of an item when an item in sold.

```
mysql> DELIMITER //
mysql> CREATE TRIGGER saleTrigger
    -> AFTER INSERT ON sale
    -> FOR EACH ROW
    -> UPDATE inventory
    -> SET quantity = quantity - NEW.quantitySold
    -> WHERE inventoryID = NEW.inventoryID;
    -> //
Query OK, 0 rows affected (0.14 sec)
```

The following screenshot shows an INSERT INTO 'sale', and the subsequent quantity decrease in the 'inventory' table:

```
mysql> SELECT * FROM inventory;
+-------------+----------------------+---------+-------+-----------+-------+----------+-------------+------------+-----------------+--------------+
| inventoryID | itemName             | antique | price | setStatus | setID | quantity | itemSize    | itemColour | reorderQuantity | itemStatus   |
+-------------+----------------------+---------+-------+-----------+-------+----------+-------------+------------+-----------------+--------------+
|           1 | Oak Table            |         |  2600 |           |  NULL |        1 | NULL        | NULL       |            NULL | available    |
|           2 | Embroided Table Cloth |         |   200 |           |  NULL |       14 | 300x200     | cream/blue |              10 | available    |
|           3 | Embroided Table Cloth |         |   200 |           |  NULL |        0 | 300x200     | cream/pink |              30 | out of stock |
|           4 | Whicker Chair        |         |   600 |           |     1 |        6 | NULL        | NULL       |            NULL | available    |
|           5 | Lampshade            |         |   400 |           |  NULL |        0 | Large       | Beige      |              15 | reordered    |
|           6 | Decorative Cushion   |         |   325 |           |  NULL |        0 | 40cm x 40cm | Navy/White |               0 | out of stock |
+-------------+----------------------+---------+-------+-----------+-------+----------+-------------+------------+-----------------+--------------+
6 rows in set (0.00 sec)
```

Before sale

```
mysql> INSERT INTO sale(customerID, inventoryID, quantitySold, totalPrice, timestamp)
    -> VALUES(3, 2, 2, 400.00, '2017-09-07 20:00:08');
    ->
    -> //
Query OK, 1 row affected (0.12 sec)
```

```
mysql> SELECT * FROM inventory;
    -> //
+-------------+----------------------+---------+-------+-----------+-------+----------+-------------+------------+-----------------+-------------+
| inventoryID | itemName             | antique | price | setStatus | setID | quantity | itemSize    | itemColour | reorderQuantity | itemStatus  |
+-------------+----------------------+---------+-------+-----------+-------+----------+-------------+------------+-----------------+-------------+
|           1 | Oak Table            | ☑       |  2600 |           |  NULL |        1 | NULL        | NULL       |            NULL | available   |
|           2 | Embroided Table Cloth |         |   200 |           |  NULL |       12 | 300x200     | cream/blue |              10 | available   |
|           3 | Embroided Table Cloth |         |   200 |           |  NULL |        0 | 300x200     | cream/pink |              30 | out of stock |
|           4 | Whicker Chair        | ☑       |   600 | ☑         |     1 |        6 | NULL        | NULL       |            NULL | available   |
|           5 | Lampshade            |         |   400 |           |  NULL |        0 | Large       | Beige      |              15 | reordered   |
|           6 | Decorative Cushion   |         |   325 |           |  NULL |        0 | 40cm x 40cm | Navy/White |               0 | out of stock |
+-------------+----------------------+---------+-------+-----------+-------+----------+-------------+------------+-----------------+-------------+
6 rows in set (0.00 sec)
```

After sale

Trigger to add items to reorder table

```
mysql> CREATE TRIGGER reorder_trigger
    -> AFTER UPDATE ON inventory
    -> FOR EACH ROW
    -> BEGIN
    -> IF NEW.quantity < 1 THEN
    -> INSERT INTO reorder(inventoryID, quantity)
    -> SELECT inventoryID, reorderQuantity
    -> FROM inventory
    -> WHERE inventoryID = NEW.inventoryID;
    -> END IF;
    -> END//
Query OK, 0 rows affected (0.12 sec)
```

The following screenshots show the reorder_trigger adding ‘Drawer Handles’ to the ‘reorder’ table after all 10 in stock had been purchased.

```
+-------------+----------------------+---------+-------+-----------+-------+----------+-------------+------------+-----------------+-------------+
| inventoryID | itemName             | antique | price | setStatus | setID | quantity | itemSize    | itemColour | reorderQuantity | itemStatus  |
+-------------+----------------------+---------+-------+-----------+-------+----------+-------------+------------+-----------------+-------------+
|           1 | Oak Table            | ☑       |  2600 |           |  NULL |        1 | NULL        | NULL       |            NULL | available   |
|           2 | Embroided Table Cloth |         |   200 |           |  NULL |       12 | 300x200     | cream/blue |              10 | available   |
|           3 | Embroided Table Cloth |         |   200 |           |  NULL |        0 | 300x200     | cream/pink |              30 | out of stock |
|           4 | Whicker Chair        | ☑       |   600 | ☑         |     1 |        6 | NULL        | NULL       |            NULL | available   |
|           5 | Lampshade            |         |   400 |           |  NULL |        0 | Large       | Beige      |              15 | reordered   |
|           6 | Decorative Cushion   |         |   325 |           |  NULL |        0 | 40cm x 40cm | Navy/White |               0 | out of stock |
|           7 | Drawer Handles       |         |    20 |           |  NULL |       10 | NULL        | NULL       |              10 | available   |
+-------------+----------------------+---------+-------+-----------+-------+----------+-------------+------------+-----------------+-------------+
7 rows in set (0.00 sec)
```

Inventory before the sale

```
mysql> INSERT INTO sale(customerID, inventoryID, quantitySold, totalPrice, timestamp)
    -> VALUES(1, 7, 10, 200.00, '2016-05-10 10:00:00');
    -> //
Query OK, 1 row affected (0.15 sec)
```

```
mysql> SELECT * FROM inventory
    -> //
+-------------+----------------------+---------+-------+-----------+-------+----------+-------------+------------+-----------------+-------------+
| inventoryID | itemName             | antique | price | setStatus | setID | quantity | itemSize    | itemColour | reorderQuantity | itemStatus  |
+-------------+----------------------+---------+-------+-----------+-------+----------+-------------+------------+-----------------+-------------+
|           1 | Oak Table            | ☑       |  2600 |           |  NULL |        1 | NULL        | NULL       |            NULL | available   |
|           2 | Embroided Table Cloth |         |   200 |           |  NULL |       12 | 300x200     | cream/blue |              10 | available   |
|           3 | Embroided Table Cloth |         |   200 |           |  NULL |        0 | 300x200     | cream/pink |              30 | out of stock |
|           4 | Whicker Chair        | ☑       |   600 | ☑         |     1 |        6 | NULL        | NULL       |            NULL | available   |
|           5 | Lampshade            |         |   400 |           |  NULL |        0 | Large       | Beige      |              15 | reordered   |
|           6 | Decorative Cushion   |         |   325 |           |  NULL |        0 | 40cm x 40cm | Navy/White |               0 | out of stock |
|           7 | Drawer Handles       |         |    20 |           |  NULL |        0 | NULL        | NULL       |              10 | available   |
+-------------+----------------------+---------+-------+-----------+-------+----------+-------------+------------+-----------------+-------------+
7 rows in set (0.00 sec)
```

Inventory after the sale

```
mysql> SELECT * FROM reorder;
    -> //
+-------------+----------+
| inventoryID | quantity |
+-------------+----------+
|           7 |       10 |
|           7 |       10 |
+-------------+----------+
2 rows in set (0.00 sec)
```

Reorder table after the sale

TASK 5:

THE SYSTEM SHOULD BE ABLE TO GENERATE A BILL FOR THE CUSTOMER

The table 'bill' can store details related to a customer's bill, and can be populated from the tables 'sale', 'inventory', and 'customer'. This can be done using an INSERT INTO and INNER JOIN query to combine columns from the three tables into one. The bill can be printed for the customer using a SELECT statement, and their customerID, or from the invoice number.

The following screenshot shows the table information for the table 'bill':

```
mysql> DESCRIBE bill;
+---------------+--------------+------+-----+---------+----------------+
| Field         | Type         | Null | Key | Default | Extra          |
+---------------+--------------+------+-----+---------+----------------+
| invoiceNumber | int(11)      | NO   | PRI | NULL    | auto_increment |
| timestamp     | datetime     | NO   |     | NULL    |                |
| itemName      | varchar(500) | NO   |     | NULL    |                |
| unitPrice     | double       | NO   |     | NULL    |                |
| quantitySold  | int(11)      | NO   |     | NULL    |                |
| customerID    | int(11)      | NO   | MUL | NULL    |                |
| customerName  | varchar(50)  | NO   |     | NULL    |                |
+---------------+--------------+------+-----+---------+----------------+
7 rows in set (0.00 sec)
```

The following screenshot shows the INSERT INTO and INNER JOIN query to populate the table.

```
mysql> INSERT INTO bill(timestamp, itemName, unitPrice, quantitySold, customerID, customerName)
    -> SELECT A.timestamp, B.itemName, B.price, A.quantitySold, A.customerID, C.name
    -> FROM ((sale A
    -> INNER JOIN inventory B ON A.inventoryID = B.inventoryID)
    -> INNER JOIN customer C ON A.customerID = C.customerID);
Query OK, 4 rows affected (0.07 sec)
Records: 4  Duplicates: 0  Warnings: 0
```

The following screenshot shows the information stored in the table 'bill':

```
mysql> SELECT * FROM bill;
+---------------+---------------------+----------------------+-----------+--------------+------------+--------------+
| invoiceNumber | timestamp           | itemName             | unitPrice | quantitySold | customerID | customerName |
+---------------+---------------------+----------------------+-----------+--------------+------------+--------------+
|             1 | 2017-04-30 15:37:18 | Oak Table            |      2600 |            1 |          1 | Sydney Twigg |
|             2 | 2017-05-01 17:07:55 | Embroided Table Cloth |       200 |            4 |          1 | Sydney Twigg |
|             3 | 2017-05-09 09:13:12 | Embroided Table Cloth |       200 |            1 |          2 | Alex Smith   |
|             4 | 2017-05-09 09:13:12 | Whicker Chair        |       600 |            6 |          2 | Alex Smith   |
+---------------+---------------------+----------------------+-----------+--------------+------------+--------------+
4 rows in set (0.00 sec)
```

The following screenshot shows a generated bill for customer Sydney Twigg's purchase of an oak table.

```
mysql> SELECT * FROM bill WHERE customerID = 1 && itemName = 'Oak Table';
+---------------+---------------------+-----------+-----------+--------------+------------+--------------+
| invoiceNumber | timestamp           | itemName  | unitPrice | quantitySold | customerID | customerName |
+---------------+---------------------+-----------+-----------+--------------+------------+--------------+
|             1 | 2017-04-30 15:37:18 | Oak Table |      2600 |            1 |          1 | Sydney Twigg |
+---------------+---------------------+-----------+-----------+--------------+------------+--------------+
1 row in set (0.00 sec)
```

REFERENCES

IBM, 2017. *Specifying when a trigger fires (BEFORE, AFTER, and INSTEAD OF clauses).* [Online]
Available at:
https://www.ibm.com/support/knowledgecenter/SSEPGG_10.1.0/com.ibm.db2.luw.admin.dbobj.doc/doc/t0020229.html
[Accessed May 2017].

Java2S, 2017. *Dictionary « Trigger « SQL / MySQL.* [Online]
Available at: http://www.java2s.com/Code/SQL/Trigger/ViewingTriggers.htm
[Accessed May 2017].

Markus, 2008. *Which MySQL data type to use for storing boolean values.* [Online]
Available at: http://stackoverflow.com/questions/289727/which-mysql-data-type-to-use-for-storing-boolean-values
[Accessed May 2017].

Microsoft, 2016. *Create Foreign Key Relationships.* [Online]
Available at: https://docs.microsoft.com/en-us/sql/relational-databases/tables/create-foreign-key-relationships
[Accessed May 2017].

M, P., 2013. *How can I update inventory using TRIGGER.* [Online]
Available at: http://stackoverflow.com/questions/16875415/how-can-i-update-inventory-using-trigger
[Accessed May 2017].

My SQL Tutorial, 2017. *Create Trigger in MySQL.* [Online]
Available at: http://www.mysqltutorial.org/create-the-first-trigger-in-mysql.aspx
[Accessed May 2017].

My SQL Tutorial, 2017. *MySQL Foreign Key.* [Online]
Available at: http://www.mysqltutorial.org/mysql-foreign-key/
[Accessed May 2017].

Narnian [screen name], 2011. *Multiple inner joins with multiple tables.* [Online]
Available at: http://stackoverflow.com/questions/7150088/multiple-inner-joins-with-multiple-tables
[Accessed May 2017].

Oracle, 2017. *Using Triggers.* [Online]
Available at: https://dev.mysql.com/doc/refman/5.7/en/triggers.html
[Accessed May 2017].

Sheldon, R., 2014. *Questions about Primary and Foreign Keys You Were Too Shy to Ask.* [Online]
Available at: https://www.simple-talk.com/sql/t-sql-programming/questions-about-primary-and-foreign-keys-you-were-too-shy-to-ask/
[Accessed May 2017].

W3 Schools, 2017. *SQL Tutorial.* [Online]
Available at: https://www.w3schools.com/sql/default.asp
[Accessed May 2017].