MTH785P-Project 2

1. Introduction

Nowadays, more and more young people are starting their own businesses and acting as brand managers to run their own commercial brands. In addition to opening offline stores, online shopping has also become the first choice for many customers. Customers often purchase and place orders through the brand's official website. Therefore, I also opened an online clothing store called XKG, which sells T-shirts, dresses, shirts, sweatshirts, jeans, jackets and other clothing for commercial operation.

To better understand the business of this clothing shop, I develop a small business analysis application to explore the sales of this clothing store in the first half of 2023. The app consists of a database, and an Excel frontend with some VBA middleware. The database stores all customer data. The Excel front end can display customer information, import new customer information and perform information query.

Almost all companies need to analyze the sales of their products, so this business project has a wide range of applications. Whether it is clothing, beauty, bags, or medicine, electrical appliances, this business platform is very useful for all industries.

2. Database

For the database, I created an online sales database myself, which contains the following data:

Online Cloth Shop	
Customer ID	Number uniquely identifying a customer
Customer Name	Name of the customer
Order ID	Number uniquely identifying an order
Order Date	Date the order was made
Product ID	Number uniquely identifying a product
Gender	Customer gender
Products	Name of the category to which the product belongs
SalePrice	The total sales revenue in dollars for all units of the product included in this order
Quantity	How many units of the product were included in this order
UnitCost	The unit cost of the product
Profit	The total profit achieved for all units of the product included in this order.
OrderStatus	The shipping status of the order
City	The destination of the order

When importing the data, I assigned specific data types to ensure accuracy and clarity. The columns **SalePrice**, **UnitCost**, and **Profit** were designated with the data type *Currency* to reflect their monetary nature. The **Quantity** column was set as an *Integer* to represent numerical values without decimal points, while **Order Date** was formatted as a *Date with Time*. Following the data import, I developed queries to create three distinct tables.

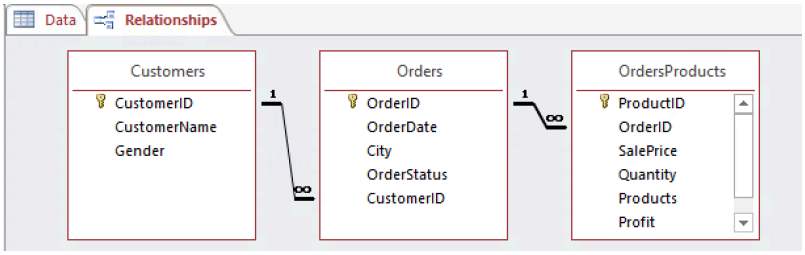
Table1: Create Customers
The Customers table includes Customer ID, Customer Name, Gender.

Table2: Create Orders
Orders table includes Order ID, OrderDate, Customer ID, OrderStatus, City.

Table3: Create OrdersProducts

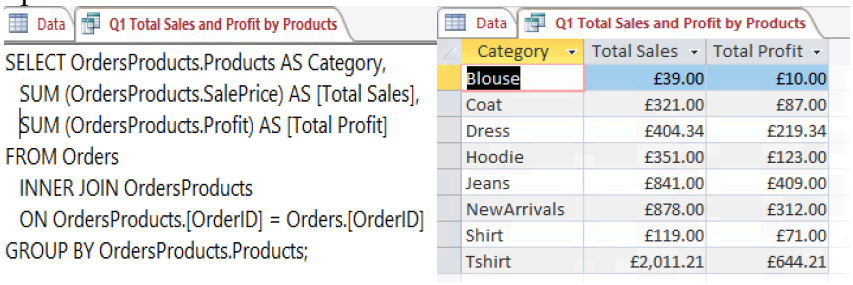
OrdersProducts table includes Product ID, Products, SalePrice, Quantity, UnitCost, Profit, Order ID.

Then I set the primary key for these tables: Customers (primary key: CustomerID), Orders (primary key: OrderID), and OrdersProducts (primary key: ProductID), and create the relationship between these three tables.

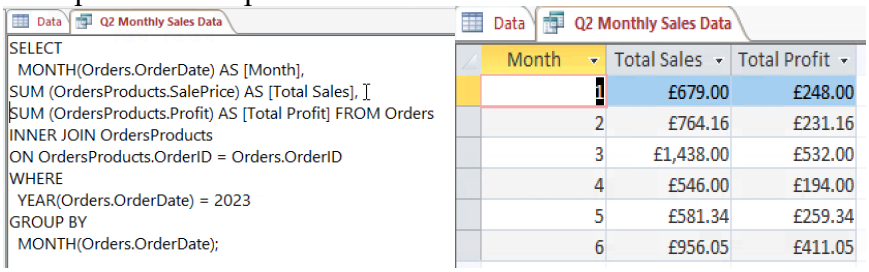


Next, I wrote some SQL Queries for Business Analysis.

First, sales data and profits are what all managers care about most, so I created the query to see the total sales and profit by products. From the query, we can know that T-shirts sold best and made the most profit.



Second, I would like to see the monthly sales data, therefore, I created the query to check the monthly sales. In the first half of 2023, March had the best sales, with sales reaching 1438 pounds and a total profit of 532 pounds.



Third, sometimes customers need to check their order information, and brand managers also want to see the repurchase rate of different customers. Hence, I created a query to list of all the order information associated with a given CustomerName.

Q3 Orders for CustomerName

```
PARAMETERS [Enter CustomerName] CHAR;
SELECT DISTINCT
  Orders.OrderID, Orders.OrderDate, Orders.OrderStatus, Customers.CustomerID,
Customers.CustomerName, OrdersProducts.Products, OrdersProducts.Quantity, OrdersProducts.SalePrice
FROM (Orders
INNER JOIN
  Customers
ON Orders.CustomerID = Customers.CustomerID)
INNER JOIN OrdersProducts
ON OrdersProducts.OrderID = Orders.OrderID
```

```
WHERE Customers.CustomerName=[Enter CustomerName];
```

Fourth, due to many reasons, there may be situations where orders are not delivered in time. Therefore, I created a query to know how many orders have not been delivered in the first quarter of 2023.

Q4 OrderStatus are not delivered in the first quarter of 2023

```
SELECT DISTINCT Orders.OrderID, Orders.CustomerID, Orders.OrderStatus, Orders.OrderDate,
Orders.City
FROM Orders
WHERE ((Orders.OrderStatus <> "Delivered") AND ((Orders.OrderDate)>=#1/1/2023# And
(Orders.OrderDate)<#3/31/2023#));
```

3. Front-end

This Excel sheet is the front end of this online clothing shop, which contains three sheets: OnlineClothShop, Sales Pivot Table and CustomerDetails.

Sheet OnlineClothShop shows all the data in the system. When clicking the button Load From Database, it imports data from the Access database and displays it in the sheet. When clicking the button Color OrderStatus, it will colour the orders to visually show the order status. Green means Delivered, yellow means Pending and blue means shipped. We can also import new customers information. The manager can enter the information in the table New Customer Entry form. When clicking the button Save Data To Database, we can take a new record from the spreadsheet and inserts it in the database.

Sheet Sales Pivot Table shows the total sales by product. The managers can check the sales of all products at different times and they can also select the specific product to see its sales.

Sheet CustomerDetails shows the outstanding orders check system, which includes a VBA subroutine that provides details for a particular customer and any outstanding orders. The outstanding order refers the order with sale price exceeding 100 pounds. When clicking the button select customers and outstanding orders, you can enter the name of the customer you want to query, and the table will return the outstanding order of this customer. If the customer does not have the order in question, the system will pop up “No outstanding orders found for customer”.

4. VBA Middleware

In the excel sheet, we have 4 VBA subroutines in total. The first subroutine is LoadFromDatabase. Through this subroutine, we connect Access and Excel using the codes:

```
Set conn = CreateObject("ADODB.Connection")
conn.Open "Provider=Microsoft.ACE.OLEDB.12.0;Data Source=" & dbPath & ";"
```

Then we write SQL query to select the data and by using a do until loop statement, we import the information from the Access database into Excel Sheet OnlineClothShop using codes such as:

```
strSQL = “SELECT OrderID, CustomerID, CustomerName, Gender, ProductID, Products, SalePrice, UnitCost,
OrderDate, Quantity, Profit, City, OrderStatus FROM Data”
Set rs = conn.Execute(strSQL)
```

The second subroutine is ColorOrderStatus. First we set the worksheet range. Then we mark different status orders with different colors through the For Next loop and if statement.

The third subroutine is GetCustomerDetailsAndOutstandingOrders. In this subroutine, we connect Excel and Access databases here, and pop up MsgBox through VBA code to enter the name and query outstanding orders. Here I set outstanding orders to orders with sales amount exceeding 100 pounds. If there is an outstanding order for the specific customer name entered, the customer information and order information will be returned. If not, the MsgBox "No outstanding orders found for customer" will be displayed.

The fourth subroutine is SaveDataToDatabase. This subroutine enters the new customer information under the New Customer Entry form into the Access database. When we run the LoadFromDatabase subroutine again, we will see the new customer information displayed.

5. Conclusion

In today's highly competitive business environment, the amount of data faced by enterprises is growing exponentially, efficient data processing capabilities are particularly critical. This application can upgrade the database or deploy cloud platform to expand batch processing capabilities, so as to help support the large-scale input and batch processing of data. With the batch processing function of the application, enterprises can quickly process orders, customer information and other business data through SQL or VBAcodes, effectively reducing operating costs and improving overall efficiency.

In addition to batch processing, the application can also upgrade the query system function and expand the real-time query processing function, so that the latest data can be queried at any time according to personalized business needs. For example, in the order management scenario, users can quickly query outstanding orders for timely follow-up. In real business, based on this application, through tools such as VBA, enterprises can design flexible and diverse query modules according to their own operational needs and create a more intelligent front-end query system. This expansion capability enables enterprises to cope with diverse business scenarios and improves operational convenience.

Finally, the application's front-end Excel interface can be further upgraded and optimized to have powerful data visualization capabilities. By using advanced visualization tools, users can transform complex data into intuitive charts and dashboards, track key indicators in real time, and discover potential trends.

In conclusion, this application is a prototype of an online store, and it has great potential for application in the real business world and business scenarios.

Link to github:

https://github.com/Cathy-zhao1007/MTH785P-Project2-OnlineClothesShop