**ITC6000 Database Management Systems CRN 22401**

**Signature Assignment #4: SQL Complex Query**

Submitted to: Professor John C. Chan

Northeastern University Silicon Valley Campus

**Submitted by: Sourabh D. Khot**

CPS Analytics

NUID 002754952

March 21, 2022

## Introduction

As a database designer and developer, I am helping a Donut shop create a mobile application to enable its customers to place orders. I had previously created four tables in Third Normal Form (3NF) form (Table 1), designed a normalized Entity-Relationship (E-R) model in crow's foot notation (Table 2), and created the database. In this report, I will insert dummy data and query the database using both simple and complex join queries using SQL Fiddle.

Table 1. Logical Table Schema

**Customer**

| Primary Key | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| Customer_ID | First_Name | Last_Name | Street_Address | Apt_No | City | State | ZIP | Mobile_Phone | Home_Phone | Other_Phone |
| 833 | John | Smith | 615 Third St | 302 | Lilburn | GA | 30047 | 1234567890 | 9876543210 | 6789054321 |

**Invoice**

| Primary Key | Foreign Key | | |
|---|---|---|---|
| Order_ID | Customer_ID | Order_Date | Special_Notes |
| 4532 | 833 | May 6, 2014 | Please include plates and napkins. |

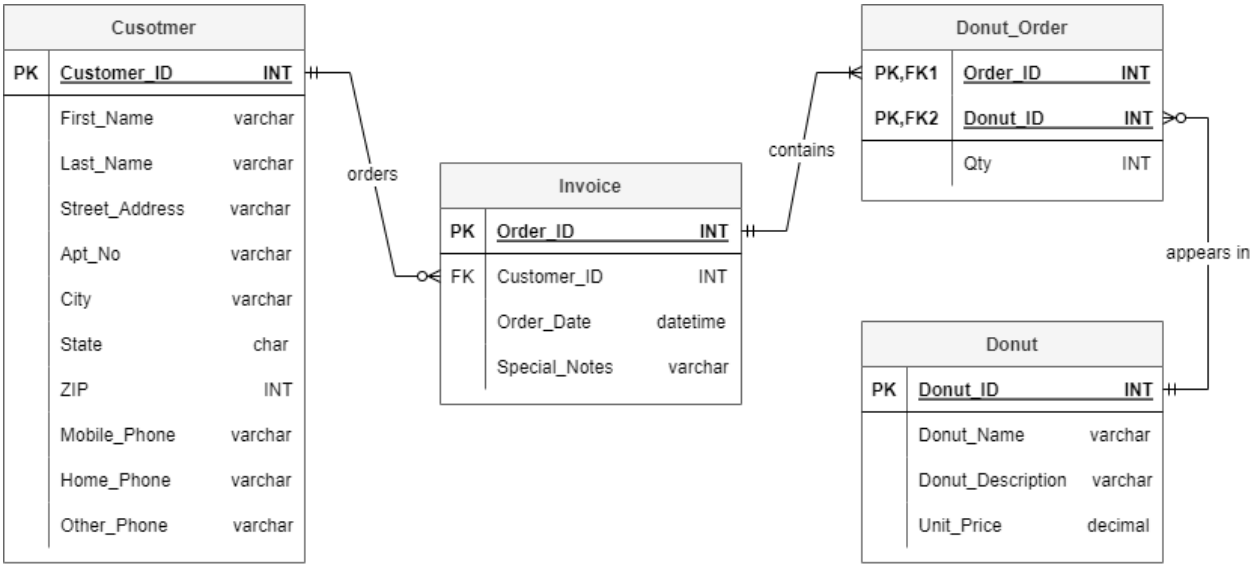
**Donut_Order**

| Composite Primary Key | | |
|---|---|---|
| Foreign Key | Foreign Key | |
| Order_ID | Donut_ID | Qty |
| 4532 | 1 | 1 |
| 4532 | 2 | 5 |
| 4532 | 3 | 12 |
| 4532 | 4 | 3 |
| 4532 | 5 | 4 |
| 4532 | 6 | 5 |

**Donut**

| Primary Key | | | |
|---|---|---|---|
| Donut_ID | Donut_Name | Donut_Description | Unit_Price |
| 1 | Plain | Plain Donut | $1.50 |
| 2 | Glazed | Glazed Donut | $1.75 |
| 3 | Cinnamon | Cinnamon Donut | $1.75 |
| 4 | Chocolate | Chocolate Donut | $1.75 |
| 5 | Sprinkle | Sprinkle Donut | $1.75 |
| 6 | Gluten-Free | Gluten-Free Donut | $2.00 |

Table 2. Entity-Relationship (E-R) model



## 1. Inserting Data

Database represented by schema and model was created in SQL Fiddle in the earlier work. Now, I will insert dummy data (satisfying data validations) into each table.

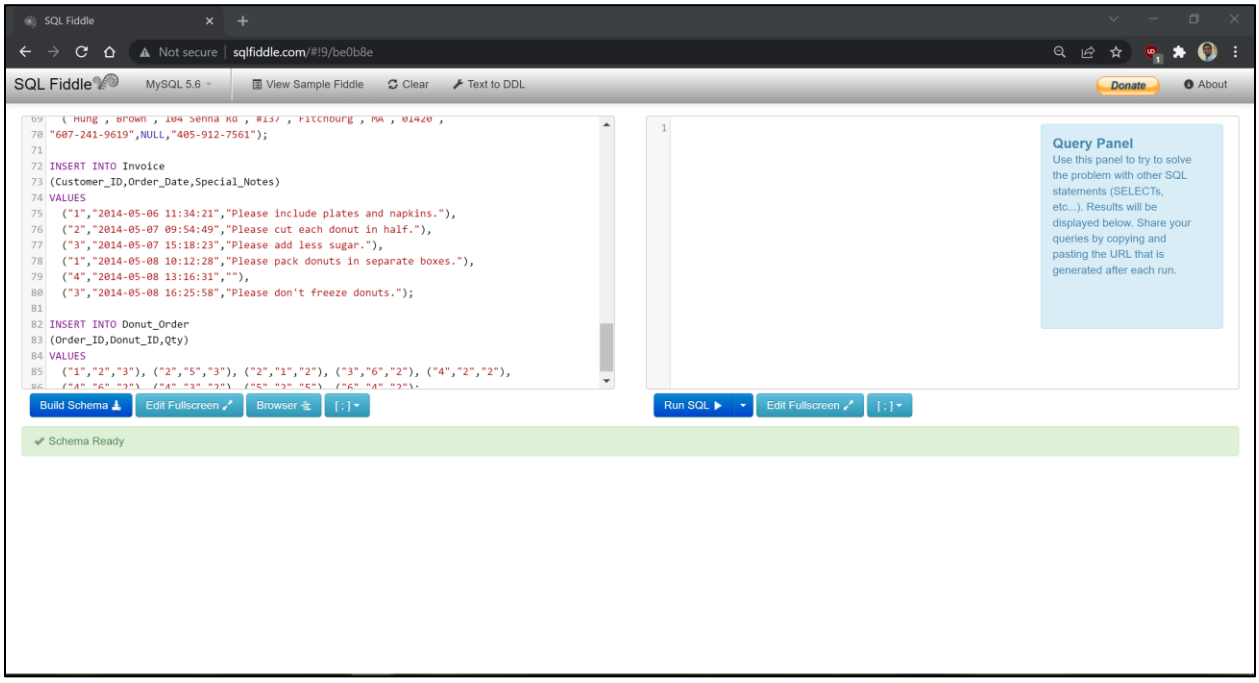
### A. SQL Code for Inserting Data

```
INSERT INTO Donut
(Donut_Name,Donut_Description,Unit_Price)
VALUES
 ("Plain","Plain Donut","1.50"),
 ("Glazed","Glazed Donut","1.75"),
 ("Cinnamon","Cinnamon Donut","1.75"),
 ("Chocolate","Chocolate Donut","1.75"),
 ("Sprinkle","Sprinkle Donut","1.75"),
 ("Gluten-Free","Gluten-Free Donut","2.00");

INSERT INTO Customer
(First_Name,Last_Name,Street_Address,Apt_No,City,State,ZIP,
Mobile_Phone,Home_Phone,Other_Phone)
VALUES
 ("Adam","Baut","695-4588 Ac Av.","#7","Chicago","IL","44027",
"456-000-3982",NULL,NULL),
 ("John","Chan","1007 MLK Jr dr","#9","Seattle","WA","98122",
"206-000-3982",NULL,NULL),
 ("Jeraldine","Lee","275 Gay St","#409","Ripley","TN","38063",
"920-776-8013","918-724-9024",NULL),
 ("Kermit","Smith","109 Oak St","#881","Onamia","MN","56359",
"904-355-9854","903-687-7243","615-906-2930"),
 ("Hung","Brown","104 Senna Rd","#137","Fitchburg","MA","01420",
"607-241-9619",NULL,"405-912-7561");

INSERT INTO Invoice
(Customer_ID,Order_Date,Special_Notes)
VALUES
 ("1","2014-05-06 11:34:21","Please include plates and napkins."),
 ("2","2014-05-07 09:54:49","Please cut each donut in half."),
 ("3","2014-05-07 15:18:23","Please add less sugar."),
 ("1","2014-05-08 10:12:28","Please pack donuts in separate boxes."),
 ("4","2014-05-08 13:16:31",""),
 ("3","2014-05-08 16:25:58","Please don't freeze donuts.");
```

```
INSERT INTO Donut_Order
(Order_ID,Donut_ID,Qty)
VALUES
 ("1","2","3"),
 ("2","5","3"),
 ("2","1","2"),
 ("3","6","2"),
 ("4","2","2"),
 ("4","6","2"),
 ("4","3","2"),
 ("5","2","5"),
 ("6","4","2");
```

## B. Demonstration for Inserting Data

All the values were inserted, and the schema was successfully built.



## 2. Simple & Complex Queries

### A. SQL Code for Simple Query

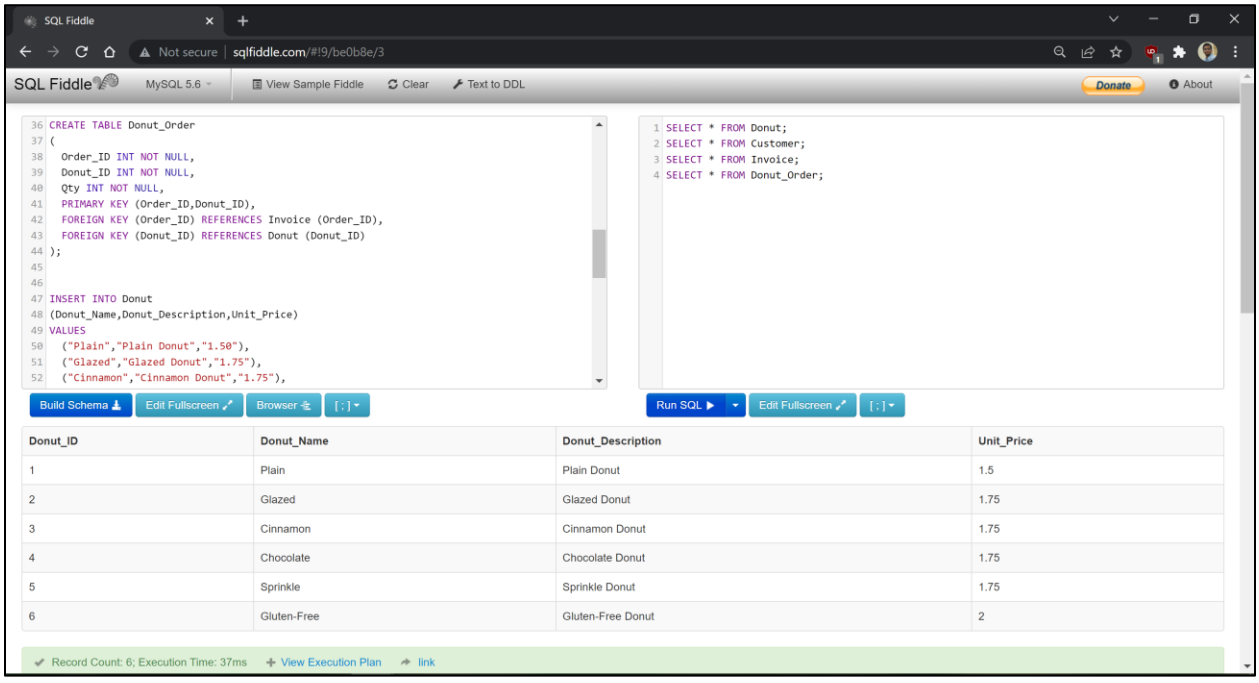
Simple SELECT-FROM queries are implemented to display all the data present in the four tables. Since the entire data is to be displayed unfiltered, no 'WHERE' clause is used.

```
SELECT * FROM Donut;
SELECT * FROM Customer;
SELECT * FROM Invoice;
SELECT * FROM Donut_Order;
```

### B. Demonstration for Simple Query

After running the above queries, screenshots of the one-screen view and the full output are given below.

#### (i) One-Screen View



## (ii) Full Output

| Donut_ID | Donut_Name | Donut_Description | Unit_Price |
|---|---|---|---|
| 1 | Plain | Plain Donut | 1.5 |
| 2 | Glazed | Glazed Donut | 1.75 |
| 3 | Cinnamon | Cinnamon Donut | 1.75 |
| 4 | Chocolate | Chocolate Donut | 1.75 |
| 5 | Sprinkle | Sprinkle Donut | 1.75 |
| 6 | Gluten-Free | Gluten-Free Donut | 2 |

Record Count: 6; Execution Time: 37ms View Execution Plan link

| Customer_ID | First_Name | Last_Name | Street_Address | Apt_No | City | State | ZIP | Mobile_Phone | Home_Pho |
|---|---|---|---|---|---|---|---|---|---|
| 1 | Adam | Baut | 695-4588 Ac Av. | #7 | Chicago | IL | 44027 | 456-000-3982 | (null) |
| 2 | John | Chan | 1007 MLK Jr dr | #9 | Seattle | WA | 98122 | 206-000-3982 | (null) |
| 3 | Jeraldine | Lee | 275 Gay St | #409 | Ripley | TN | 38063 | 920-776-8013 | 918-724-90 |
| 4 | Kermit | Smith | 109 Oak St | #881 | Onamia | MN | 56359 | 904-355-9854 | 903-687-72 |
| 5 | Hung | Brown | 104 Senna Rd | #137 | Fitchburg | MA | 1420 | 607-241-9619 | (null) |

Record Count: 5; Execution Time: 0ms View Execution Plan link

| Order_ID | Customer_ID | Order_Date | Special_Notes |
|---|---|---|---|
| 1 | 1 | 2014-05-06T11:34:21Z | Please include plates and napkins. |
| 2 | 2 | 2014-05-07T09:54:49Z | Please cut each donut in half. |
| 3 | 3 | 2014-05-07T15:18:23Z | Please add less sugar. |
| 4 | 1 | 2014-05-08T10:12:28Z | Please pack donuts in separate boxes. |
| 5 | 4 | 2014-05-08T13:16:31Z | |
| 6 | 3 | 2014-05-08T16:25:58Z | Please don't freeze donuts. |

Record Count: 6; Execution Time: 1ms View Execution Plan link

| Order_ID | Donut_ID | Qty |
|---|---|---|
| 1 | 2 | 3 |
| 2 | 1 | 2 |
| 2 | 5 | 3 |
| 3 | 6 | 2 |
| 4 | 2 | 2 |
| 4 | 3 | 2 |
| 4 | 6 | 2 |
| 5 | 2 | 5 |
| 6 | 4 | 2 |

Record Count: 9; Execution Time: 1ms View Execution Plan link

### C. SQL Code for Complex Query

To display all the information from the original ‘Sales Order Form’ as closely as possible and in a condensed view:

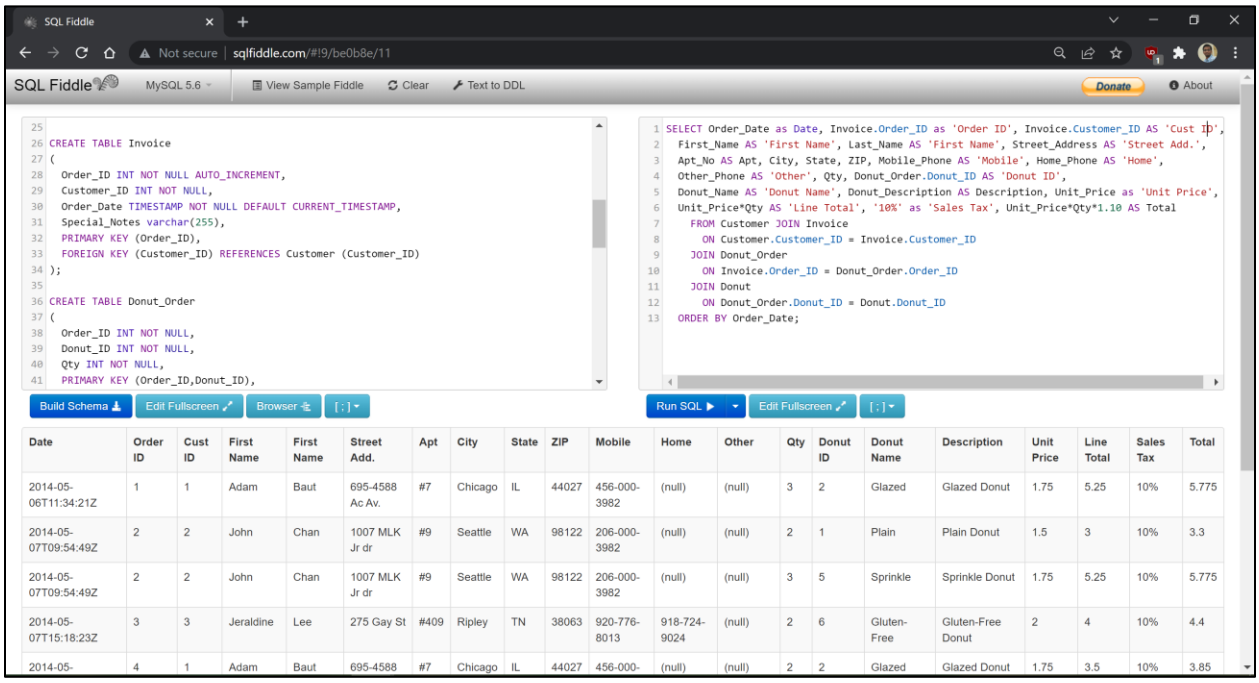
- I have used the ‘AS’ command to create aliases for column names.
- The ‘Line Total’ column is computed by multiplying ‘Qty’ with ‘Unit Price’
- The original form shows ‘Total’ including 10% sales tax totaled at the ‘Order ID’ level. However, since I am joining all tables in a single view, I am computing the tax-inclusive ‘Total’ at each ‘Donut ID’–‘Order ID’ level. This view can easily be further combined to get Total at the ‘Order ID’ level.

```
SELECT Order_Date as Date,
    Invoice.Order_ID as 'Order ID',
    Invoice.Customer_ID AS 'Cust ID',
    First_Name AS 'First Name',
    Last_Name AS 'First Name',
    Street_Address AS 'Street Add.',
    Apt_No AS Apt,
    City,
    State,
    ZIP,
    Mobile_Phone AS 'Mobile',
    Home_Phone AS 'Home',
    Other_Phone AS 'Other',
    Qty,
    Donut_Order.Donut_ID AS 'Donut ID',
    Donut_Name AS 'Donut Name',
    Donut_Description AS Description,
    Unit_Price as 'Unit Price',
    Unit_Price*Qty AS 'Line Total',
    '10%' as 'Sales Tax',
    Unit_Price*Qty*1.10 AS Total
  FROM Customer JOIN Invoice
      ON Customer.Customer_ID = Invoice.Customer_ID
  JOIN Donut_Order
      ON Invoice.Order_ID = Donut_Order.Order_ID
  JOIN Donut
      ON Donut_Order.Donut_ID = Donut.Donut_ID
  ORDER BY Order_Date;
```

## D. Demonstration for Complex Query

After running the above queries, screenshots of the one-screen view and the full output are given below.

### (i) One-Screen View



## (ii) Full Output

| Date | Order ID | Cust ID | First Name | First Name | Street Add. | Apt | City | State | ZIP | Mobile | Home | Other | Qty | Donut ID | Donut Name | Description | Unit Price | Line Total | Sales Tax | Total |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 2014-05-06T11:34:21Z | 1 | 1 | Adam | Baut | 695-4588 Ac Av. | #7 | Chicago | IL | 44027 | 456-000-3982 | (null) | (null) | 3 | 2 | Glazed | Glazed Donut | 1.75 | 5.25 | 10% | 5.775 |
| 2014-05-07T09:54:49Z | 2 | 2 | John | Chan | 1007 MLK Jr dr | #9 | Seattle | WA | 98122 | 206-000-3982 | (null) | (null) | 2 | 1 | Plain | Plain Donut | 1.5 | 3 | 10% | 3.3 |
| 2014-05-07T09:54:49Z | 2 | 2 | John | Chan | 1007 MLK Jr dr | #9 | Seattle | WA | 98122 | 206-000-3982 | (null) | (null) | 3 | 5 | Sprinkle | Sprinkle Donut | 1.75 | 5.25 | 10% | 5.775 |
| 2014-05-07T15:18:23Z | 3 | 3 | Jeraldine | Lee | 275 Gay St | #409 | Ripley | TN | 38063 | 920-776-8013 | 918-724-9024 | (null) | 2 | 6 | Gluten-Free | Gluten-Free Donut | 2 | 4 | 10% | 4.4 |
| 2014-05-08T10:12:28Z | 4 | 1 | Adam | Baut | 695-4588 Ac Av. | #7 | Chicago | IL | 44027 | 456-000-3982 | (null) | (null) | 2 | 3 | Cinnamon | Cinnamon Donut | 1.75 | 3.5 | 10% | 3.85 |
| 2014-05-08T10:12:28Z | 4 | 1 | Adam | Baut | 695-4588 Ac Av. | #7 | Chicago | IL | 44027 | 456-000-3982 | (null) | (null) | 2 | 2 | Glazed | Glazed Donut | 1.75 | 3.5 | 10% | 3.85 |
| 2014-05-08T10:12:28Z | 4 | 1 | Adam | Baut | 695-4588 Ac Av. | #7 | Chicago | IL | 44027 | 456-000-3982 | (null) | (null) | 2 | 6 | Gluten-Free | Gluten-Free Donut | 2 | 4 | 10% | 4.4 |
| 2014-05-08T13:16:31Z | 5 | 4 | Kermit | Smith | 109 Oak St | #881 | Onamia | MN | 56359 | 904-355-9854 | 903-687-7243 | 615-906-2930 | 5 | 2 | Glazed | Glazed Donut | 1.75 | 8.75 | 10% | 9.625 |
| 2014-05-08T16:25:58Z | 6 | 3 | Jeraldine | Lee | 275 Gay St | #409 | Ripley | TN | 38063 | 920-776-8013 | 918-724-9024 | (null) | 2 | 4 | Chocolate | Chocolate Donut | 1.75 | 3.5 | 10% | 3.85 |

**References**

(n.d.). Retrieved from SQL Fiddle MySQL 5.6: http://sqlfiddle.com/#!9/0c6fc9

Donut Assignment. (n.d.). Northeastern ITC6000 Resources.

*How to Multiply Two Columns in SQL*. (n.d.). Retrieved from LearnSQL.com: learnsql.com/cookbook/how-to-multiply-two-columns-in-sql/

*Random US Address | Best Random Tools*. (n.d.). Retrieved from BestRandoms.com: bestrandoms.com/random-address

*Random US, UK, CA Phone Numbers*. (n.d.). Retrieved from RandomPhoneNumbers.com: randomphonenumbers.com/

*SQL SELECT Statement*. (n.d.). Retrieved from W3Schools: w3schools.com/sql/sql_select.asp