ITC6000 Database Management Systems CRN 22401

Signature Assignment #3: SQL Implementation

Submitted to: Professor John C. Chan

Northeastern University Silicon Valley Campus

Submitted by: Sourabh D. Khot

CPS Analytics

NUID: 002754952

Introduction

As a database designer and developer, I am helping a Donut shop create a mobile application to enable its customers to place orders. From their existing sales order form, I had previously created four tables in Third Normal Form (3NF) form, and designed a normalized Entity-Relationship (E-R) model in crow's foot notation as given below. In this report, I will implement the database using SQL in SQLFiddle using CREATE, VIEW and INDEX statements.

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.

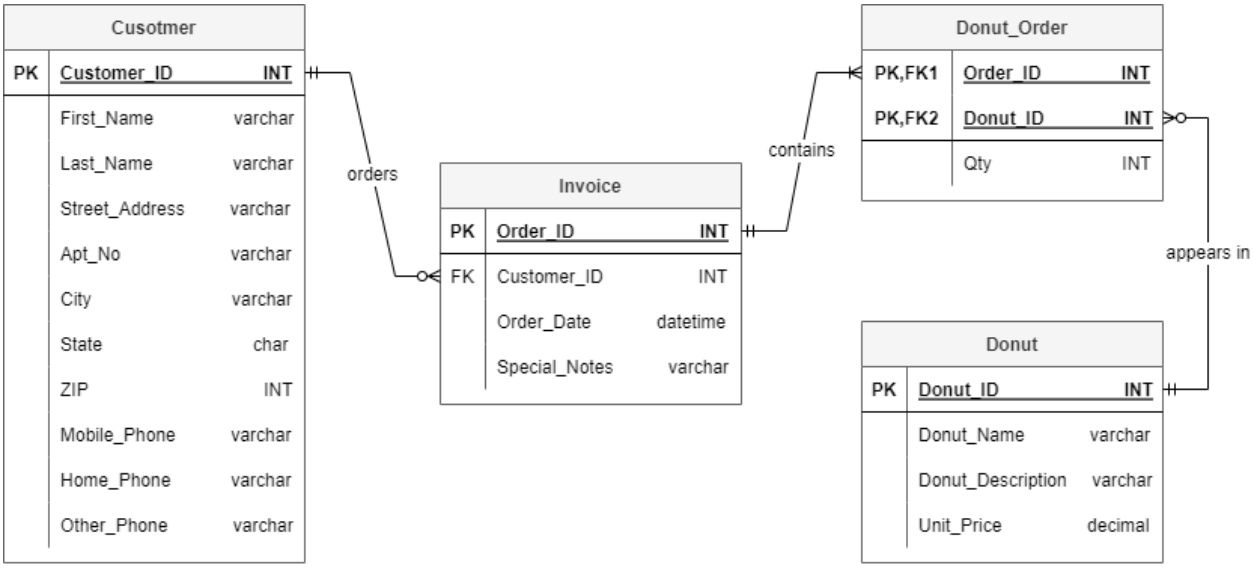
Donut_Order

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



Assumptions

I have considered additional assumptions in designing this database. The data type of 'Order_Date' is kept as *datetime* since a mobile application should record both the date and time of order. Instead of 'Home_Phone', 'Mobile_Phone' is considered mandatory (*NOT NULL*) since not everyone has a landline, but anyone using the mobile application must have a mobile number. 'Apt_No' is considered optional since those living in independent houses will not have one. As per US address format, 'ZIP' is kept five-digit *INT* and 'State' must be entered in the two-character short-form.

1. Creating Tables

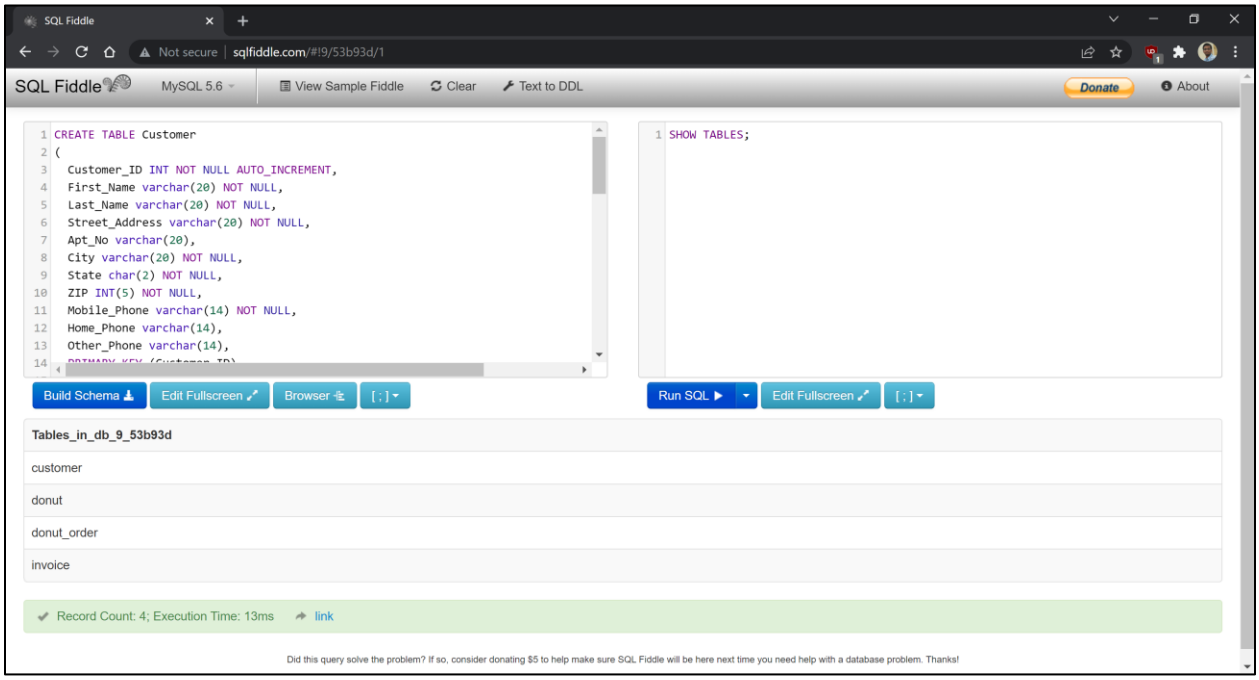
The above four tables are implemented using SQL Fiddle, assigning the primary keys, foreign keys, and attribute data types. The SQL code and demonstration are given below.

A. SQL code

```
CREATE TABLE Customer
(
 Customer_ID INT NOT NULL AUTO_INCREMENT,
 First_Name varchar(20) NOT NULL,
 Last_Name varchar(20),
 Street_Address varchar(20) NOT NULL,
 Apt_No varchar(20),
 City varchar(20) NOT NULL,
 State char(2) NOT NULL,
 ZIP INT(5) NOT NULL,
 Mobile_Phone varchar(14) NOT NULL,
 Home_Phone varchar(14),
 Other_Phone varchar(14),
 PRIMARY KEY (Customer_ID)
);

CREATE TABLE Invoice
(
 Order_ID INT NOT NULL AUTO_INCREMENT,
 Customer_ID INT NOT NULL,
 Order_Date Date NOT NULL,
 Special_Notes varchar(255),
 PRIMARY KEY (Order_ID),
 FOREIGN KEY (Customer_ID) REFERENCES Customer (Customer_ID)
);

CREATE TABLE Donut
(
 Donut_ID INT NOT NULL AUTO_INCREMENT,
 Donut_Name varchar(20) NOT NULL,
 Donut_Description varchar(30) NOT NULL,
 Unit_Price decimal(5,2) NOT NULL,
 PRIMARY KEY (Donut_ID)
);
```

```
CREATE TABLE Donut_Order
(
  Order_ID INT NOT NULL,
  Donut_ID INT NOT NULL,
  Qty INT NOT NULL,
  PRIMARY KEY (Order_ID,Donut_ID),
  FOREIGN KEY (Order_ID) REFERENCES Invoice (Order_ID),
  FOREIGN KEY (Donut_ID) REFERENCES Donut (Donut_ID)
);

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);
```

B. Testing in SQL Fiddle



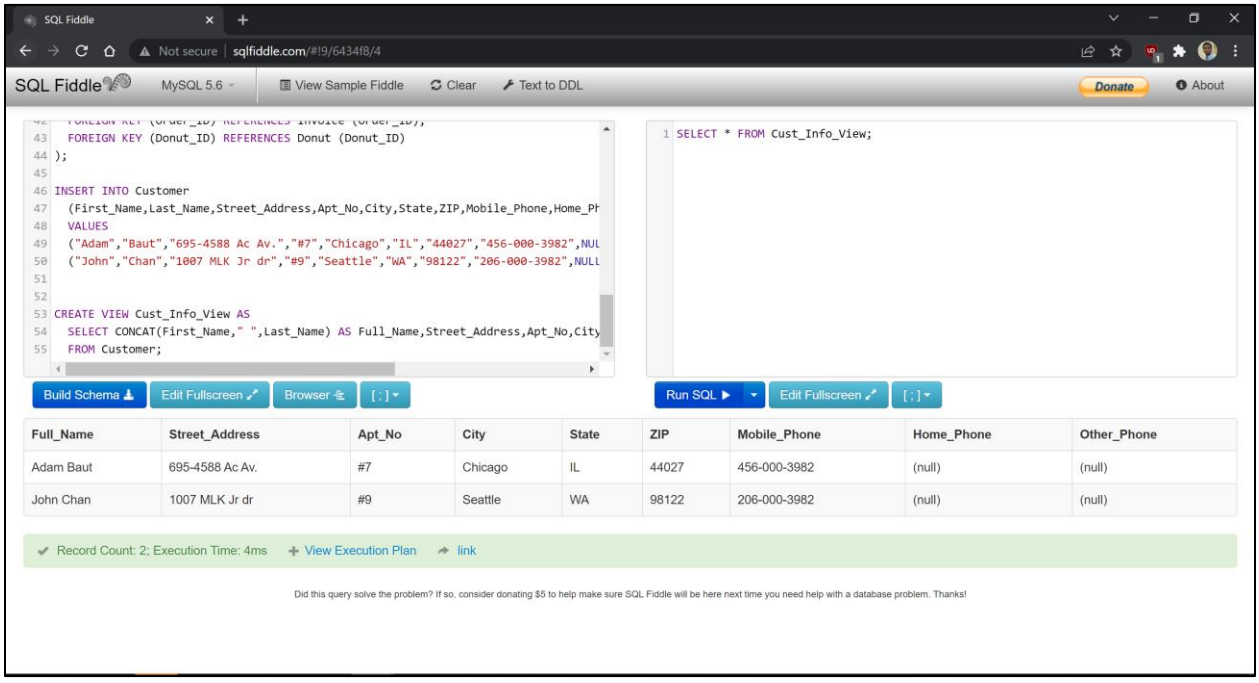
2. Creating View of Customer Information

A new view ‘Cust_Info_View’ of customer information is created by combining ‘First_Name’ and ‘Last_Name’ using the *CONCAT()* function and including all other details. Given next are the SQL code and the demonstration.

A. SQL Code

```
CREATE VIEW Cust_Info_View AS
  SELECT CONCAT(First_Name," ",Last_Name) AS Full_Name,Street_Address,
  Apt_No,City,State,ZIP,Mobile_Phone,Home_Phone,Other_Phone
  FROM Customer;
```

B. Testing in SQL Fiddle



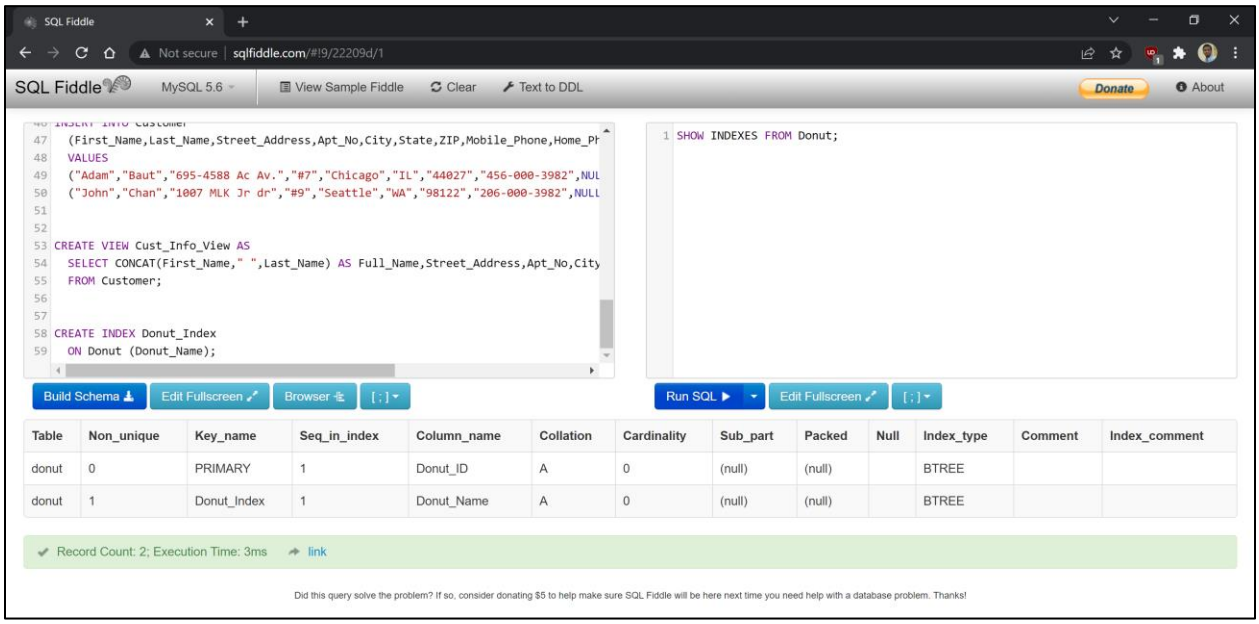
3. Creating Index for Donut Name

The 'Donut_Name' in 'Donut' table is indexed for faster querying using *CREATE INDEX* command. The SQL code for implementation and actual demonstration in SQL Fiddle follows next.

A. SQL Code

```
CREATE INDEX Donut_Index
  ON Donut (Donut_Name);
```

B. Testing in SQL Fiddle



References

(n.d.). Retrieved from SQL Fiddle MySQL 5.6: http://sqlfiddle.com/#!9/22209d/1

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

How to Create a Composite Primary Key in SQL Server? (n.d.). Retrieved from GeeksforGeeks.org: https://www.geeksforgeeks.org/how-to-create-a-composite-primary-key-in-sql-server/

SQL CREATE INDEX Statement. (n.d.). Retrieved from W3 Schools: https://www.w3schools.com/sql/sql_create_index.asp

SQL CREATE VIEW, REPLACE VIEW, DROP VIEW Statements. (n.d.). Retrieved from W3 Schools: https://www.w3schools.com/sql/sql_view.asp

SQL List All Tables. (n.d.). Retrieved from sqltutorial.org: https://www.sqltutorial.org/sql-list-all-tables/

tsqpl - List of all index & index columns in SQL Server DB - Stack Overflow. (2009). Retrieved from stackoverflow.com: https://stackoverflow.com/questions/765867/list-of-all-index-index-columns-in-sql-server-db