

Seidenberg Applied Data Sciences & Networking Lab

Student Lab #2: Customer Table Creation and Population in Microsoft Access and Microsoft SQL Server

Introduction

In this lab, Lab 2, you will be creating the “Customers” table, along with its supporting tables in Microsoft Access and Microsoft SQL Server. You will see the limitations of “SQL View” in Microsoft Access, and will be shown how to migrate tables from Microsoft Access to Microsoft SQL Server. You will create 10,000 total customers, 5,000 males and 5,000 females. You will accomplish this by having 3 foundational tables, one containing 50 male first names, one containing 50 female first names, and one containing 100 last names. The data sources for these can be found in the appendix at the end of this document. By using a Cartesian product query, every last name will be assigned to every male first name, and every female first name. This yields 10,000 total customers because $(50 \text{ male first names} \times 100 \text{ last names} = 5,000 \text{ full male names})$ and $(50 \text{ female first names} \times 100 \text{ last names} = 5,000 \text{ full female last names})$. Before starting this lab, you should already have gone through the documentation that explains how to install Microsoft SQL Server.


Tables to Create and Populate in Microsoft Access


Male_First_Names		
	Field Name	Data Type
	ID	Number
	Male_First_Name	Short Text

Female_First_Names		
	Field Name	Data Type
	ID	Number
	Female_First_Names	Short Text

Last_Names	
Field Name	Data Type
ID	Number
Last_Name	Short Text

- 1) Create the 3 tables as shown above using either “SQL View” or the graphical method in a new Microsoft Access database entitled “Customers”.
- 2) Import the proper datasets into each table. Just as in the earlier labs, the dataset, and its corresponding table share the same name.
- 3) Once the data has been imported into the tables, create the “Males” table and “Females” table as shown below:

 Male_ID	AutoNumber
First_Name	Short Text
Last_Name	Short Text
Sex	Short Text
Street_Number	Number
Street_Name	Short Text
Street_Type	Short Text
City	Short Text
State	Short Text
Zipcode	Number

 Female_ID	AutoNumber
First_Name	Short Text
Last_Name	Short Text
Sex	Short Text
Street_Number	Number
Street_Name	Short Text
Street_Type	Short Text
City	Short Text
State	Short Text
Zipcode	Number

- 4) Note #1: For those who wish to make the above tables using “SQL View”, the syntax for creating a primary key field with an AutoNumber datatype is as shown below:

[ID Field Name] INT PRIMARY KEY NOT NULL

- 5) Once you have the “Males” and “Females” tables made, run the two following queries in “SQL View” to utilize the Cartesian product, and to populate the “Males” and “Females” tables.

Males_Cartesian:

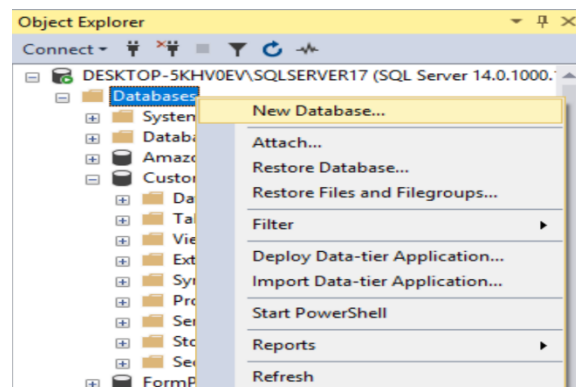
```
INSERT INTO Males ( First_Name, Last_Name, Sex )  
SELECT Male_First_Names.Male_First_Name, [Last_Names].[Last_Name], "M" AS Sex  
FROM Male_First_Names, Last_Names;
```

Females_Cartesian

```
INSERT INTO Females ( First_Name, Last_Name, Sex )  
SELECT Female_First_Names.Female_First_Names, [Last_Names].[Last_Name], "F" AS Sex  
FROM Female_First_Names, Last_Names;
```

Moving to SQL Server

- 6) Now, to go on with the further development and population of the “Customer” table, you must migrate the supporting tables you’ve made in the “Customer” Microsoft Access database to Microsoft SQL Server. To start, open up the “SQL Server Management Studio”.
- 7) In the left-hand side pane of “SQL Server Management Studio” (SSMS), right-click on “Databases”, and select “New Database” as shown below.

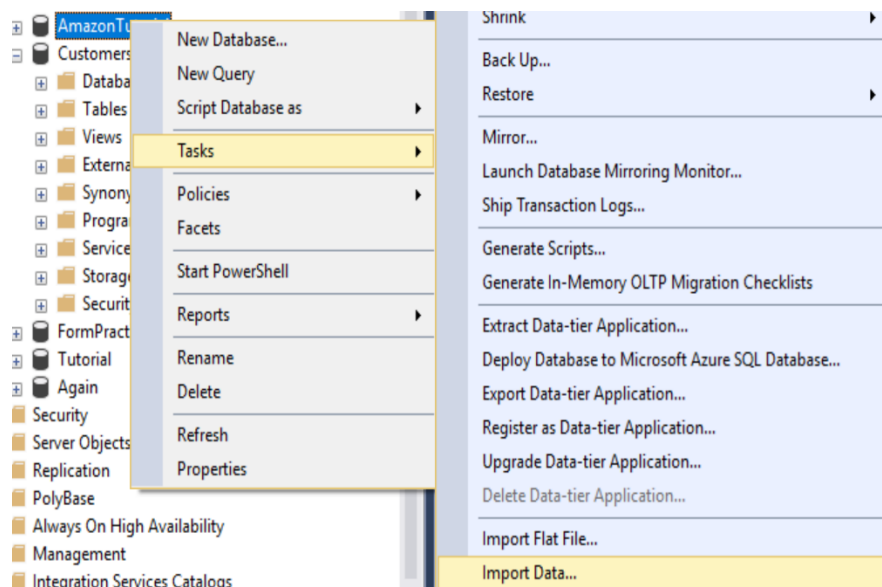


- 8) Entitle the database “Amazon”, as you will have all products tables, customer, and order tables in this database.

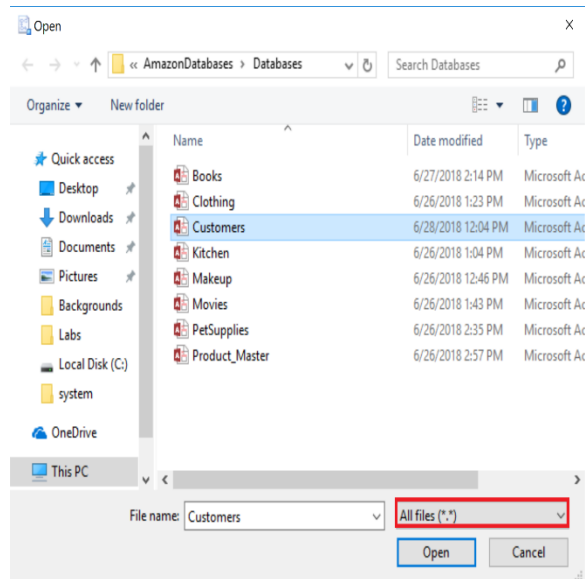
- 9) In Microsoft SQL Server, you will have a query window where you will save all of your queries. The query window is similar to a notepad editor, with the functionality of running the entire query, or subqueries at once. To create a new query, hit CTRL-N. Recall the “SQL View” method of creating tables in Microsoft Access. Simply type out the table to create the table, and then hit “Execute”. You can also highlight the chunk of code you wish to run, and hit “Execute” when you add more queries to your query window. Run the following query to create the “Males” table, which you will import data into:

```
CREATE TABLE Males (  
Male_ID int identity(1,1) PRIMARY KEY NOT NULL,  
First_Name text,  
Last_Name text,  
Sex text,  
Street_Number int,  
Street_Name text,  
Street_Type text,  
City text,  
State text,  
Zip int)
```

- 10) Now, to import your “Males” from Microsoft Access, right click on your new database (“Amazon”), then hover over “Tasks”, then in the pop-up menu, select “Import Data”.



- 11) Hit “Next” on the welcome page. On the next page that comes up, which asks you for your data source, select “Microsoft Access (Microsoft Access Database Engine)”. Browse to your Microsoft Access database entitled “Customers”. Ensure that you change the filetype to “All Files”.



12) Once you have selected your data source, hit “Open”, and then “Next”.

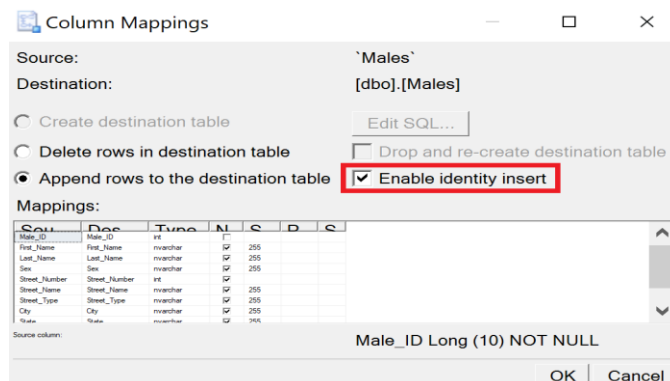
13) For the destination, select “Microsoft OLE DB Provider for SQL Server”. Hit “Next”.

14) On the next page, leave “Copy data from one or more tables or views” selected and hit “Next”.

15) Select only the box as shown below.



16) On that same page, select “Edit Mappings”. Turn on “Enable Identity Insert”, as shown below.



17) Click “Ok” and then “Next”. On the next page, hit “Finish”. The data will begin to transfer over into Microsoft SQL Server and populate. Then, hit “Close”.

- 18) Now that you have your “Males” table in Microsoft SQL Server, you must finish filling in the other fields (“Street_Number”, “Street_Name”, “Street_Type”, “City”, “State”, “Zip”). To start, create the following table entitled “Zipcodes” in Microsoft SQL Server using SQL code

Column Name	Data Type	Allow Nulls
ZipID	int	<input type="checkbox"/>
Zipcode	int	<input checked="" type="checkbox"/>
City	text	<input checked="" type="checkbox"/>
State	text	<input checked="" type="checkbox"/>

- 19) Import the text file entitled “Zipcodes” into this table. Remember, when performing the import, the data source in this case is “Flat File Source”.

- 20) Once the data has finished importing, run the following cursor and stored procedure to update the “City”, “State”, and “Zip” fields of the “Males” table. Copy and paste the code below into your query editor in SSMS, and then highlight all of it. Hit “Execute”. By doing this, you are only executing the highlighted code.

```
CREATE PROCEDURE zipcodeM_cursor
AS
BEGIN
--SET NOCOUNT ON
DECLARE @V_Males_id int
Declare @V_random_zipcode int
declare @V_zipcode int
declare @V_city nvarchar(max)
declare @V_state nvarchar(max)
declare @counter int
set @counter = 1
/* Declaring Cursor for the zipcode table for males*/
DECLARE Males_cursor CURSOR FOR
SELECT Male_ID
FROM [Males]
OPEN Males_cursor
FETCH NEXT FROM Males_cursor INTO @V_Males_id
WHILE @@FETCH_STATUS = 0
BEGIN
set @V_random_zipcode = (select top 1 abs(checksum(newid())) % 49 FROM Zipcodes)
set @V_zipcode = (select Zipcode from Zipcodes where ZipID = @V_random_zipcode)
set @V_city = (select City from Zipcodes where ZipID = @V_random_zipcode)
set @V_state = (select State from Zipcodes where ZipID = @V_random_zipcode)
update Males set Zip=@V_zipcode, City=@V_city, State=@V_state where Male_ID= @counter
set @counter = @counter + 1;
FETCH NEXT FROM Males_cursor INTO @V_Males_id
END
print @counter
CLOSE Males_cursor
DEALLOCATE Males_cursor END
```

21) By running the code above, you have created the procedure to update the “Males” table fields of “City”, “State”, and “Zip”. To execute the procedure, run the following line of code:

```
EXECUTE zipcodeM_cursor
```

22) To see your updated “Males” table, run the following:

```
SELECT * FROM Males
```

23) You will now move onto the population of the “Street_Number”, “Street_Name”, and “Street_Type” in the “Males” table. To start, create the following table entitled “Street_Number_Name”. Once you have the table created, import into it the document entitled “Street_Number_Name”. Ensure that you remove the check next to “Column names in the first data row”. Do this for all data importations.

Column Name	Data Type	Allow Nulls
ID	int	<input checked="" type="checkbox"/>
Street_Number	int	<input checked="" type="checkbox"/>
Street_Name	text	<input checked="" type="checkbox"/>

24) Now, create a table entitled “Street_Temp_Male”, as shown below. (*Hint: for this table, create an ID field with an AutoNumber datatype. To do this in SSMS, the syntax is: *[Field Name] int identity(1,1)*)

Column Name	Data Type	Allow Nulls
ID	int	<input type="checkbox"/>
Street_Number_Temp	int	<input checked="" type="checkbox"/>
Street_Name_Temp	text	<input checked="" type="checkbox"/>

25) Run the following query to populate “Street_Temp_Male”:

```
DECLARE @value INT
SET @value = 1
WHILE @value<=5000
BEGIN
    INSERT INTO Street_Temp_Male(Street_Number_Temp,Street_Name_Temp)
    SELECT top 1 Street_Number,Street_Name from Street_Number_Name order by newid()
    SET @value=@value+1
END
```

26) Now, you must finish the “Males” table by populating the “Street_Type” column. The method to do so is similar to the method you followed to fill in “Street_Number” and “Street_Name”. First, start by creating a table entitled “Street_Type”, and importing the dataset “Street_Type” into it.

Column Name	Data Type	Allow Nulls
ID	int	<input type="checkbox"/>
Street_Type	text	<input checked="" type="checkbox"/>

27) Now, create the following table, “Street_Type_Temp_Male”:

Column Name	Data Type	Allow Nulls
ID	int	<input type="checkbox"/>
Street_Type_Temp	text	<input checked="" type="checkbox"/>

28) To populate “Street_Type_Temp_Male”, run the following query:


```
DECLARE @value INT
SET @value = 1
while @value<=5000
begin
insert into Street_Type_Temp_Male(Street_Type_Temp) select top 1 Street_Type from
Street_Type order by newid()
set @value=@value+1
end
```

29) Now, to finish populating the “Males” table with the “Street_Type”, run the following query:

```
UPDATE Males
SET Street_Type=b.Street_Type_Temp
From (select top 5000 Street_Type_Temp, id from Street_Type_Temp_Male ) AS b
where males.male_id=b.id;
```

30) Now, go back to Step 9, and repeat the process for the “Females” table. Use the same query editor window, and be sure to save the query. You do not need to recreate “Zipcodes”, “Street_Number_Name”, or “Street_Type”, but you will need to create “Street_Temp_Female” and “Street_Type_Temp_Female”. Be sure to make the appropriate name changes in the cursors and procedures when populating the “Females” table.

31) Once you have finished creating the “Females” table, you must create the “Customers” table, as shown below:

	Column Name	Data Type	Allow Nulls
	ID	int	<input type="checkbox"/>
	First_Name	text	<input checked="" type="checkbox"/>
	Last_Name	text	<input checked="" type="checkbox"/>
	Sex	nvarchar(255)	<input checked="" type="checkbox"/>
	Street_Number	int	<input checked="" type="checkbox"/>
	Street_Name	text	<input checked="" type="checkbox"/>
	Street_Type	text	<input checked="" type="checkbox"/>
	City	text	<input checked="" type="checkbox"/>
	State	text	<input checked="" type="checkbox"/>
	Zipcode	int	<input checked="" type="checkbox"/>
	Date_Of_Birth	date	<input checked="" type="checkbox"/>
	Marital_Status_ID	int	<input checked="" type="checkbox"/>
	Credit_Card_ID	int	<input checked="" type="checkbox"/>
	Education_ID	int	<input checked="" type="checkbox"/>
	Salary	money	<input checked="" type="checkbox"/>

32) Now, you must create the table (“Random_Dates”) that will store the birthdays of the customers.

	Column Name	Data Type	Allow Nulls
	ID	int	<input type="checkbox"/>
	Dates	date	<input checked="" type="checkbox"/>

33) Run the following cursor and procedure to generate random birthdays:

```

/* Cursor to create and store random dates*/
DECLARE @startLoopID INT = 1
DECLARE @endLoopID INT = 10000
DECLARE @FromDate DATETIME2(0)
DECLARE @ToDate DATETIME2(0)
WHILE @startLoopID<=@endLoopID
BEGIN
SET @FromDate = '1950-01-01'
SET @ToDate = '2000-01-01'
DECLARE @Seconds INT = DATEDIFF(SECOND, @FromDate, @ToDate)
DECLARE @Random INT = ROUND((((@Seconds-1) * RAND()), 0)
INSERT INTO Random_Date ([Dates])
VALUES (DATEADD(SECOND, @Random, @FromDate))
SET @startLoopID=@startLoopID+1;
END

```

34) Create a table entitled “Salaries”, as shown below. Import the dataset entitled “Salaries” into it.

Column Name	Data Type	Allow Nulls
ID	int	<input checked="" type="checkbox"/>
Salary	money	<input checked="" type="checkbox"/>

35) Create a table entitled “Credit_Card_Lookup”, and populate it as shown below:

Column Name	Data Type	Allow Nulls
ID	int	<input type="checkbox"/>
Credit_Card	text	<input checked="" type="checkbox"/>

ID	Credit_Card
1	American Express
2	Discover
3	MasterCard
4	Visa

36) Create a table entitled “Credit_Card_Temp”, as shown below. Once completed, import the dataset entitled “Credit_Card_Temp” into the table.


Column Name	Data Type	Allow Nulls
ID	int	<input checked="" type="checkbox"/>
Credit_Card_ID	int	<input checked="" type="checkbox"/>

37) Create the following table entitled “Education_Lookup” and populate the table as shown below.


Column Name	Data Type	Allow Nulls
ID	int	<input type="checkbox"/>
Education	text	<input checked="" type="checkbox"/>

ID	Education
1	High School Diploma
2	Associate's Degree
3	Bachelor's Degree
4	Master's Degree
5	Ph.D.
6	Post-Doctorate Degree

38) Create the following table entitled “Education_Temp” and import the dataset entitled “Education_Temp” into the table.

Column Name	Data Type	Allow Nulls
 ID	int	<input checked="" type="checkbox"/>
Education_ID	int	<input checked="" type="checkbox"/>

39) Create the following table entitled “Marital_Status_Lookup”, and populate it as shown below.

Column Name	Data Type	Allow Nulls
 ID	int	<input type="checkbox"/>
Marital_Status	text	<input checked="" type="checkbox"/>

ID	Marital_Status
1	Single
2	Married
3	Widowed
4	Divorced

40) Create the following table entitled “Marital_Status_Temp”, and import the dataset entitled “Marital_Status_Temp” into the table.

Column Name	Data Type	Allow Nulls
 ID	int	<input checked="" type="checkbox"/>
Marital_Status_ID	int	<input checked="" type="checkbox"/>

41) Now, to populate the “Customers” table, run the following query.

```
INSERT INTO Customers
(First_Name,Last_Name,Sex,Street_Number,Street_Name,Street_Type,City,State,Zipcode)
SELECT First_Name, Last_Name, Sex, Street_Number, Street_Name, Street_Type, City, State,
Zip FROM Males
INSERT INTO Customers
(First_Name,Last_Name,Sex,Street_Number,Street_Name,Street_Type,City,State,Zipcode)
SELECT First_Name, Last_Name, Sex, Street_Number, Street_Name, Street_Type, City, State,
Zip FROM Females
UPDATE Customers
SET Birthday=c.Dates
From (select top 10000 Dates, id from Random_Date ) AS c
where Customers.id=c.id;
UPDATE Customers
SET Salary=d.Salary
From (select top 10000 Salary, id from Salaries ) AS d
where Customers.id=d.id;
UPDATE Customers
SET Credit_Card_ID=e.Credit_Card_ID
From (select top 10000 Credit_Card_ID, id from Credit_Card_Temp ) AS e
where Customers.id=e.id;
UPDATE Customers
SET Marital_Status_ID=f.Marital_Status_ID
From (select top 10000 Marital_Status_ID, id from Marital_Status_Temp ) AS f
where Customers.id=f.id;
SELECT * FROM Customers
UPDATE Customers
SET Education_ID=g.Education_ID
From (select top 10000 Education_ID, id from Education_Temp ) AS g
where Customers.id=g.id;
```

Review

In this lab, you used Microsoft Access and Microsoft SQL Server to create and populate a “Customers” table with 10,000 records and various fields.