|  |
| --- |
| **66SOFTWARE** |

Building a Database From Scratch

BY DAVID C. HAYES AND JAMES E. HUNTON

**Note:**

* This assignment is adapted from the Journal of Accountancy article above.
* You will be required to create a series of tables and establish the appropriate relationship(s) between tables.
* **The assignment will require you to implement several features that improve accuracy and completeness of data capture.**
* One of the tables will be created by importing data from an excel spreadsheet file.

1. Submit the completed file via the D2L dropbox.
   1. Files submitted to the wrong dropbox will receive a **grade of 0**
2. Assignment **due date: 09-14-2025, 11:30 P.M**
3. **This assignment is not a group project. Students should work independently**.
   1. **Copying/plagiarism** will result in a **grade of 0** for all parties involved.
4. **Remember, file must be virus free**. (**assignment received containing a Virus receives a grade of 0)**
5. **Additional Note: Make sure you submit the database file and not the temp file that is created when you are in access.** 
   1. **The temp file is erased when you exit an access database, but there may be a delay before that happens.**
   2. **Submitted database files will have the extension .accdb.** 
      1. **The temp file is a 1kb file with the extension .laccdb.**
      2. **Check the size of your submission that is on D2L.**
      3. **If it is only 1kb, you have submitted the wrong file**
   3. **It is your responsibility to make sure that you upload the correct file. Failure to do so will result in a grade of zero**
   4. **Submission to the wrong drop box will result in grade of zero**

**STARTING FROM SCRATCH**

The first, most important step in building a database is a user-needs analysis—that is, figuring out what information the user wants the database to supply. Although databases are quite flexible and can be adjusted as needs change, it helps to have a general goal in mind. This tutorial creates a fictional small clothing distributor seeking to expand the business. Because the distributor needs a better understanding of customers’ spending habits, this exercise will create six tables—one for **customers**, one for **products**, one for **sales**, one for **sales rep**, one for **sales-inventory** table and a **states** table.

**Step 1**:

Open Access. Create a blank database file and name it **AssignmentOne**.

**Step 2**

* Import the list of states from the Excel file
* Select external data
* Excel
* Import source data into new table
* Check first row contains column headings
* Check let access assign primary key
* On the import to table screen, **name the table state**

**Step 3**

Create a data table called **Customers**. This table should contain the following fields:

* Customer ID,
* Company Name,
* Contact First Name,
* Contact Last Name,
* Address 1,
* Address 2
* City,
* State
* Postal Code
* Phone Number

**Step 4**

In table design view define data types and field properties as follow:

|  |  |  |
| --- | --- | --- |
| Data Field | Data type | Field properties |
| **Customer ID** (set as the primary key) | Autonumber |  |
| Company Name | Text | Field size 40 |
| Contact First Name | Text | Field size 20 |
| Contact Last Name | Text | Field size 20 |
| Address 1 | Text | Field size 40 |
| Address 2 | Text | Field size 40 |
| City | Text | Field size 20 |
| State | Text | Field size 20 |
| Postal Code | Text | Field size 10 |
| Phone Number | Text | Field size 14 |

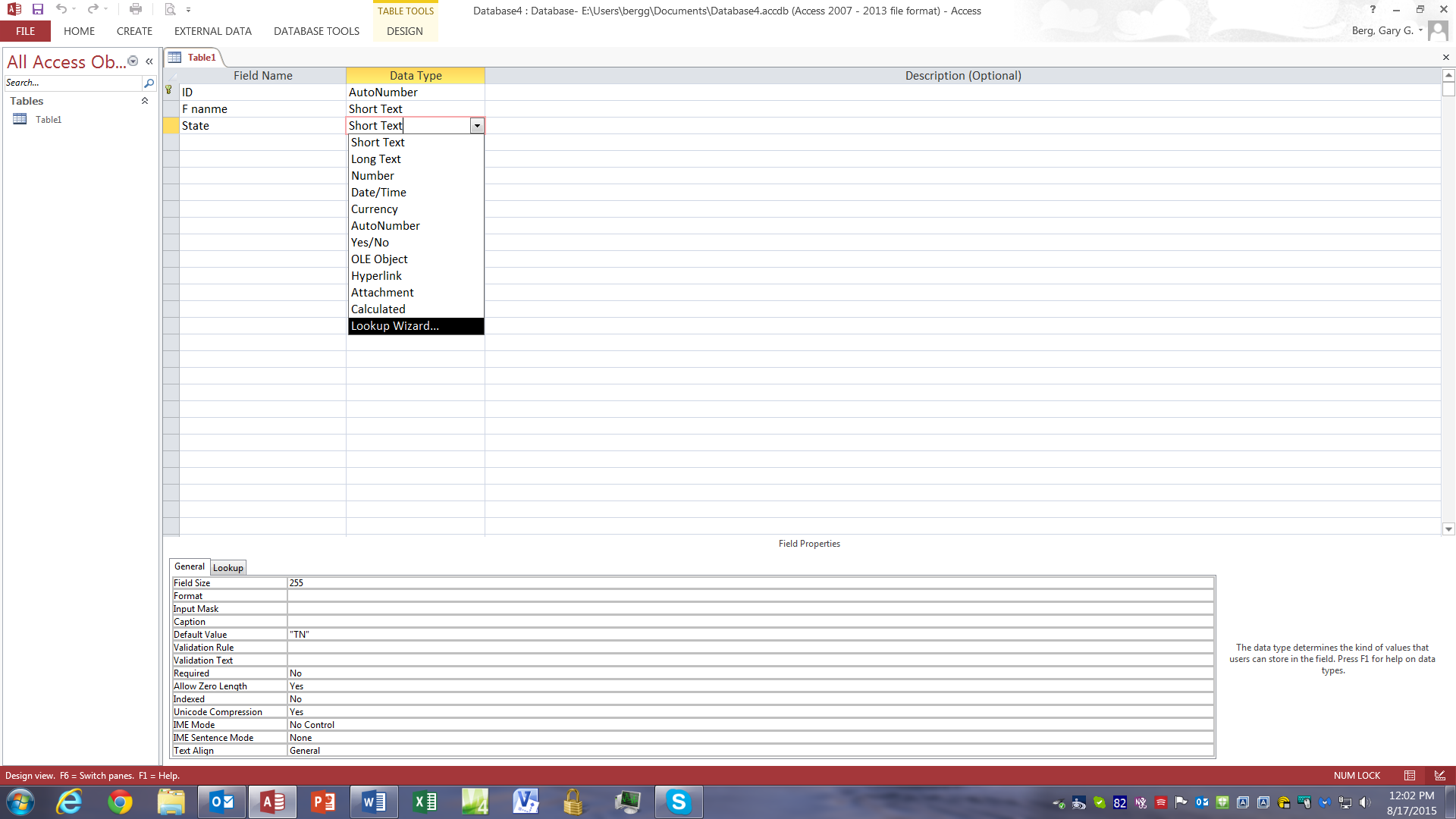
Setting the primary key (Customer ID), prevents duplication of Customer ID numbers. Each Customer ID is unique. (no duplicate Customer ID numbers can be entered). Setting a primary key does not stop you from entering the same company name twice. To prevent that, put the cursor in the **Company Name** cell and then change the **Indexed property** from No to **Yes (No Duplicates).**

**Ways to improve Completeness and Accuracy**

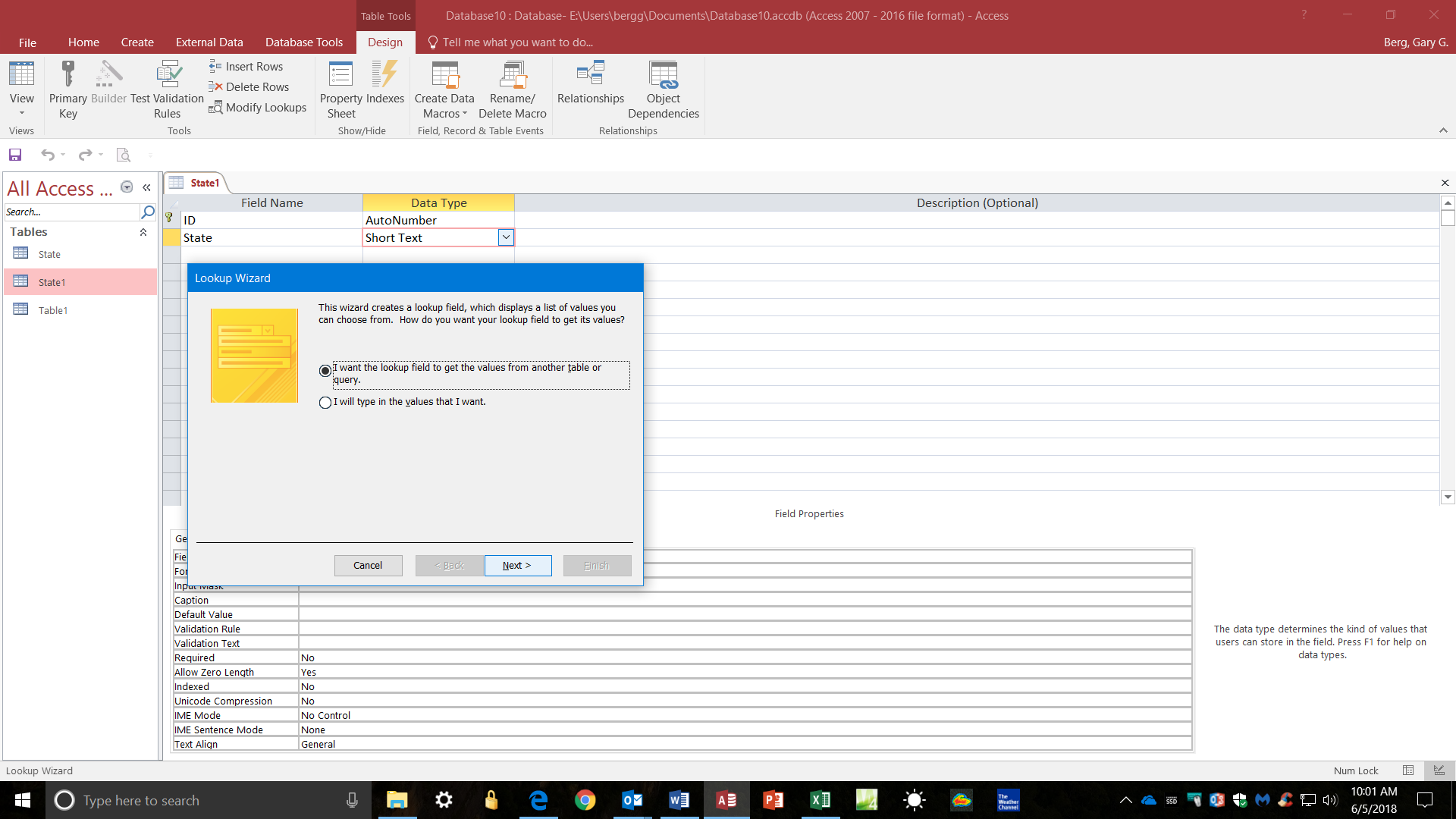
There are ways for completeness and accuracy of the database to be improved. To improve completeness, change the **Required** property from **No** to **Yes for all non-key fields except for one.( I will not tell you which non key field should not be required)**  This prevents a record from being created if any values are missing.

Another way to improve accuracy and save time is to fill cells with default values and to use a look-up field if the default is not acceptable. In design view, use the look up wizard to **make the state field a look up field**. In addition since most customers are located in Tennessee, set the **default state** value to the primary key numeric value for “Tennessee”. Should there be an exception to the default value, you can use the drop down list created in the look-up wizard to enter to correct state**.**

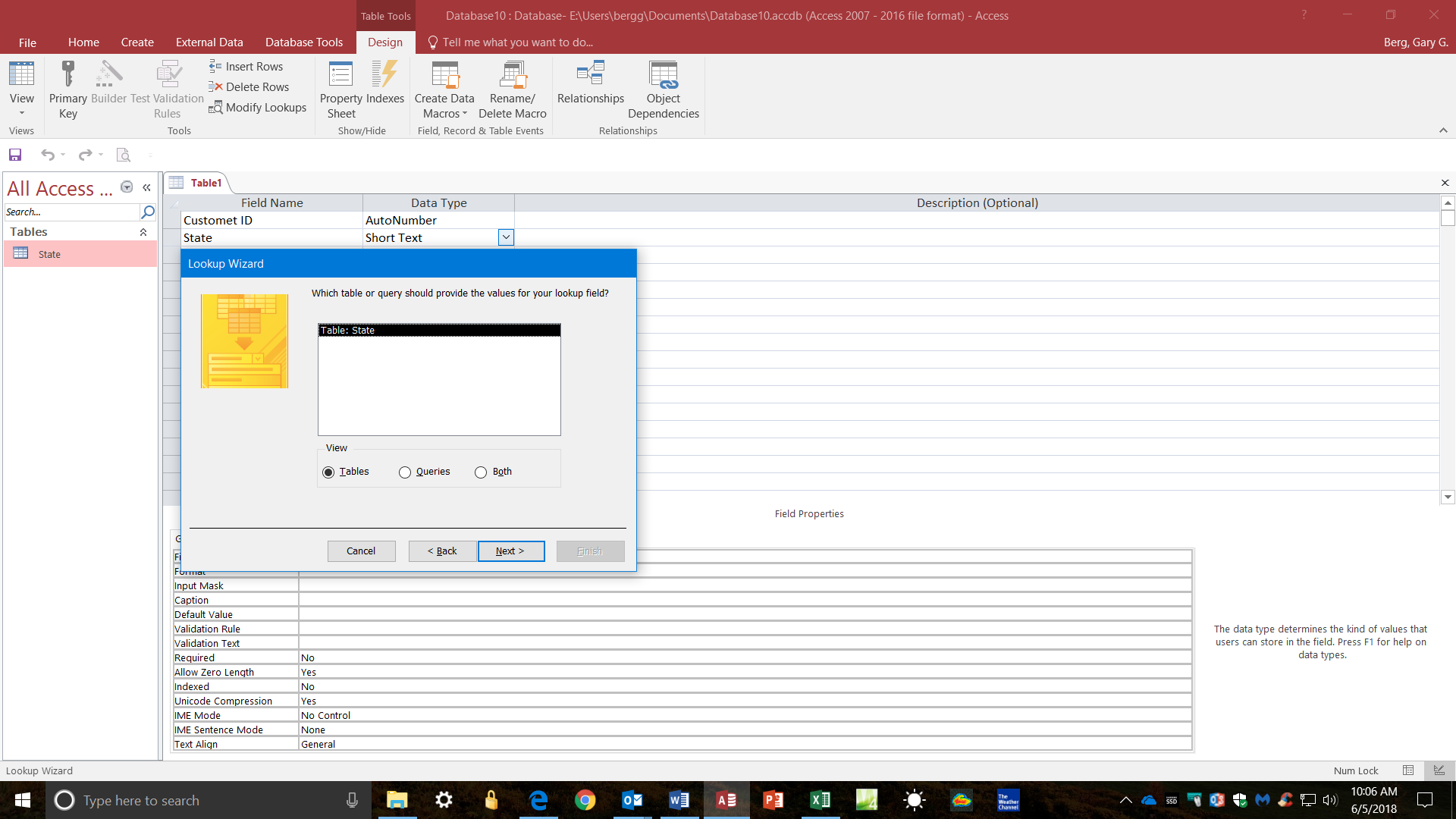
**For the state field, select data type look-up wizard**



**Select I want the lookup field to get the values from another table or query**



**Select the State table as your source**



**Select the State field and select next**

Graphical user interface, application

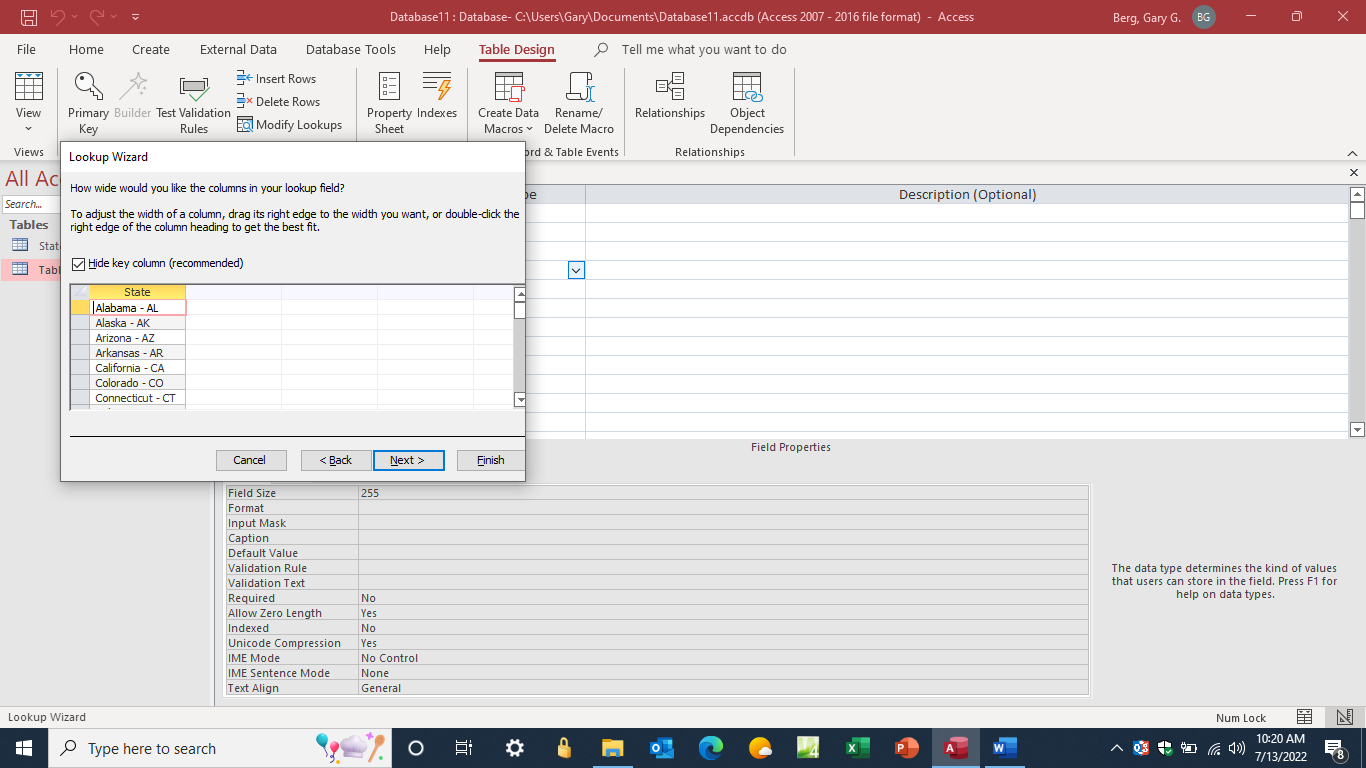
Description automatically generated

**Sort on ID ascending order and select next**

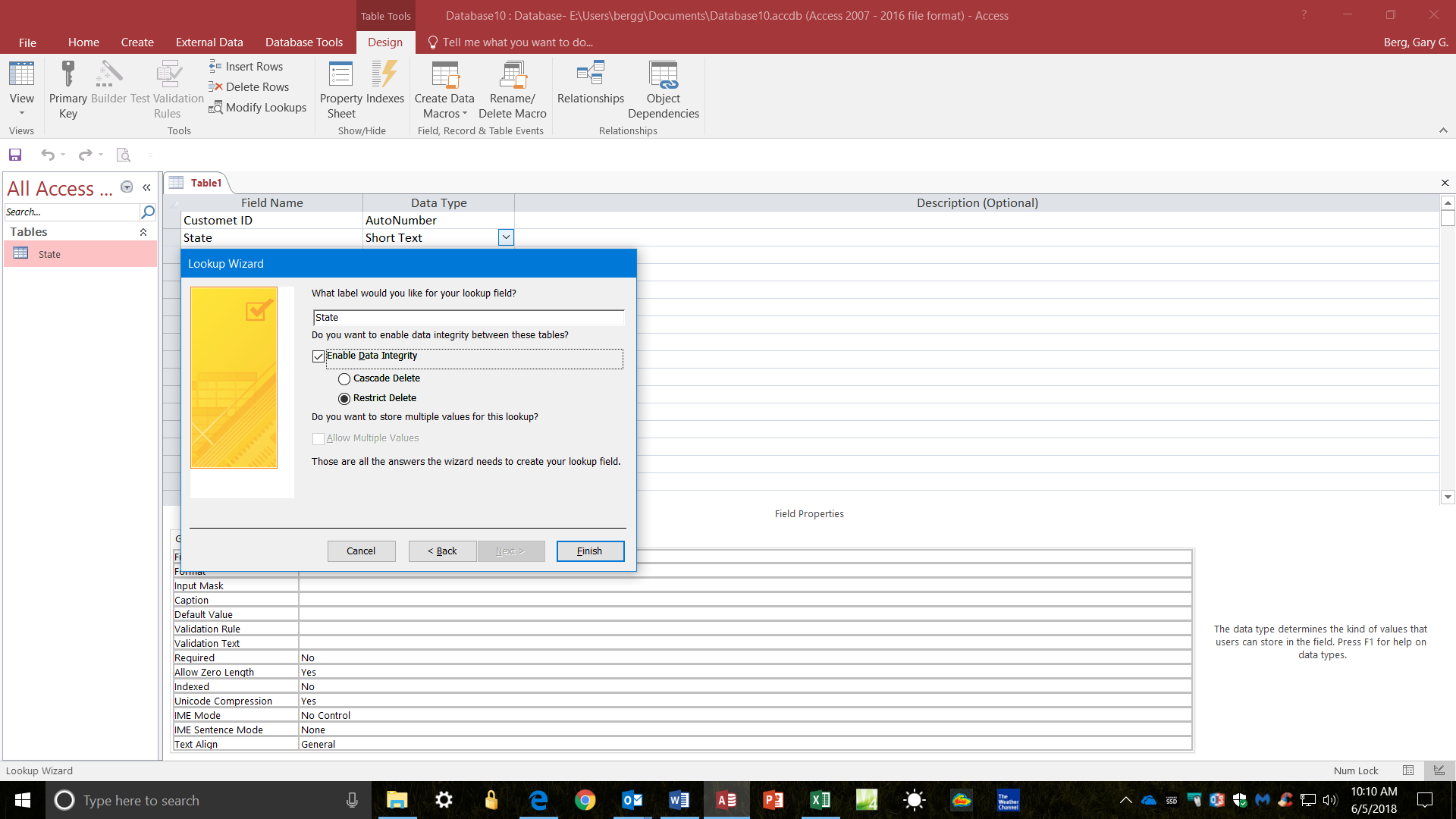
A screenshot of a computer

Description automatically generated

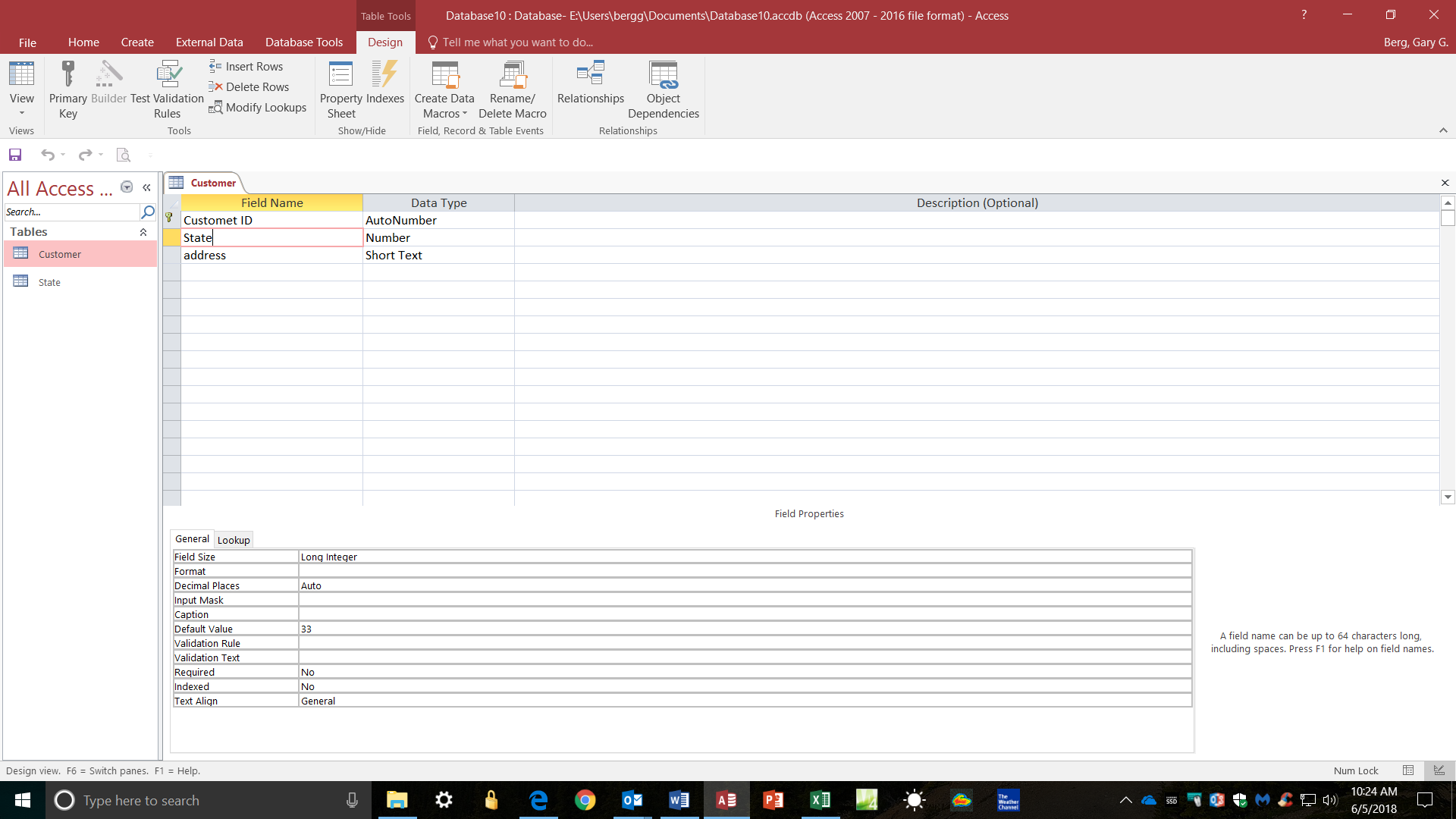
**Hide key column and select next**



**Name lookup field state**



**Select the state field and set the default value**



In design view, enter the primary key numeric value for “Tennessee” to set the default value for state to Tennessee.(**master the data**)

**Yet another way to improve accuracy is to use the input mask feature. Use the appropriate input mask feature for the phone number and the postal code. The “0s” in the mask means a numeric value is required and the “9s” indicate that a numeric value is optional.**

Graphical user interface, application

Description automatically generated

**Step 5a**

Create a simple data entry form for the customer table using the form wizard

**Step 5b**

Enter the following data to the customer table. In addition **add a third record** with ETSU as the company, your first and last name as the contact, address 220 Sam Wilson Hall, Box 70710, Johnson City Tennessee 37614-1710, phone 423-439-4000.

Exhibit 5b

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Customer ID** | **Company Name** | **Contact First Name** | **Contact Last Name** | **Address 1** | **Address 2** | **City** | **State** | **Postal Code** | **Phone Number** |
| 1 | Fly-by-Night  Clothes | Mark | Fly | 210 Sam-Wilson Hall | Box 707109 | Johnson City | Tennessee - TN | 37614-1710 | (423) 439-4332 |
| 2 | Master Semesters | Lora | Masters | 222 Any Street |  | Fayetteville | Arkansas - AR | 77777-7777 | (777) 777-7777 |
|  |  |  |  |  |  |  |  |  |  |

**Step 6**

Create a **Sales Rep table**. The table should contain the following fields:

* Sales Rep ID
* First name
* Last Name
* Commission

In table design view define data types and field properties as follow:

|  |  |  |
| --- | --- | --- |
| Data Field | Data type | Field properties |
| **Sales Rep ID** (set as the primary key) | Autonumber |  |
| First Name | Text | Field size 20 |
| Last Name | Text | Field size 20 |
| Commission % | Number | Field size Single  Format %  Decimal places 2 |

To improve completeness, change the **Required** property for each of the field names from **No** to **Yes**; this prevents a record from being created if any values are missing.

You can now enter data into the Sales Rep table. Enter all the data that are shown in **exhibit 6**.

Exhibit 6

| **Sales Rep** | | | |
| --- | --- | --- | --- |
| **Sales Rep ID** | **First Name** | **Last Name** | **Commission %** |
| 1 | John | Smith | 10 |
| 2 | Jane | Smith | 12 |

**Step 7**

Create a data table called **Products**. This table should conation the following fields:

* Product ID,
* Product Name,
* Product Description
* Unit Cost

In table design view define data types and field properties as follow:

|  |  |  |
| --- | --- | --- |
| Data Field | Data type | Field properties |
| **Product ID** (set as the primary key) | Autonumber |  |
| Product Name | Text | Field size 20, |
| Product Description | Text | Field size 20 |
| Unit Cost | Currency | Format 2 decimal places |

To improve completeness, change the **Required** property for each of the field names from **No** to **Yes**; this prevents a record from being created if any values are missing.

Another accuracy feature is the Validation Rule (**this rule is a range check specifying a minimum and maximum values**). In design view, click on the Cost field name, click on the Validation Rule cell. We want the database to reflect the fact that all items will cost at least $20 and to reflect the fact that no items cost more than $375 per item to be purchased. This can be accomplished by entering the following **range check** **“>= $20 and <= $375**” in the Validation Rule cell. Validation Text is the message that will be displayed if the validation rule is broken. In the Validation Text, type the text you want displayed, such as “There must be an error. All items cost at least $20 and we do not buy anything that costs more than $375.” **Once you enter the data in step 7, you should test your rule to make sure it works.**

**Step 8**

You can now enter product data into the product table. Enter the data that is shown in **exhibit 8**.

Exhibit 8

| **Products** | | | |
| --- | --- | --- | --- |
| **Product ID** | **Prodcut Name** | **Product Description** | **Unit Cost** |
| 1 | Jeans | 905 Blue Jeans | $35.00 |
| 2 | Shirts | Old Grungy Shirts | $20.00 |
| 3 | Designer Jeans | Overpriced Designer Jeans | $325.00 |

**Step 9**

Create a data table called **Sales**. This table should conation the following fields:

* Sale ID,
* Date,
* Time
* Customer ID
* SalesRepID

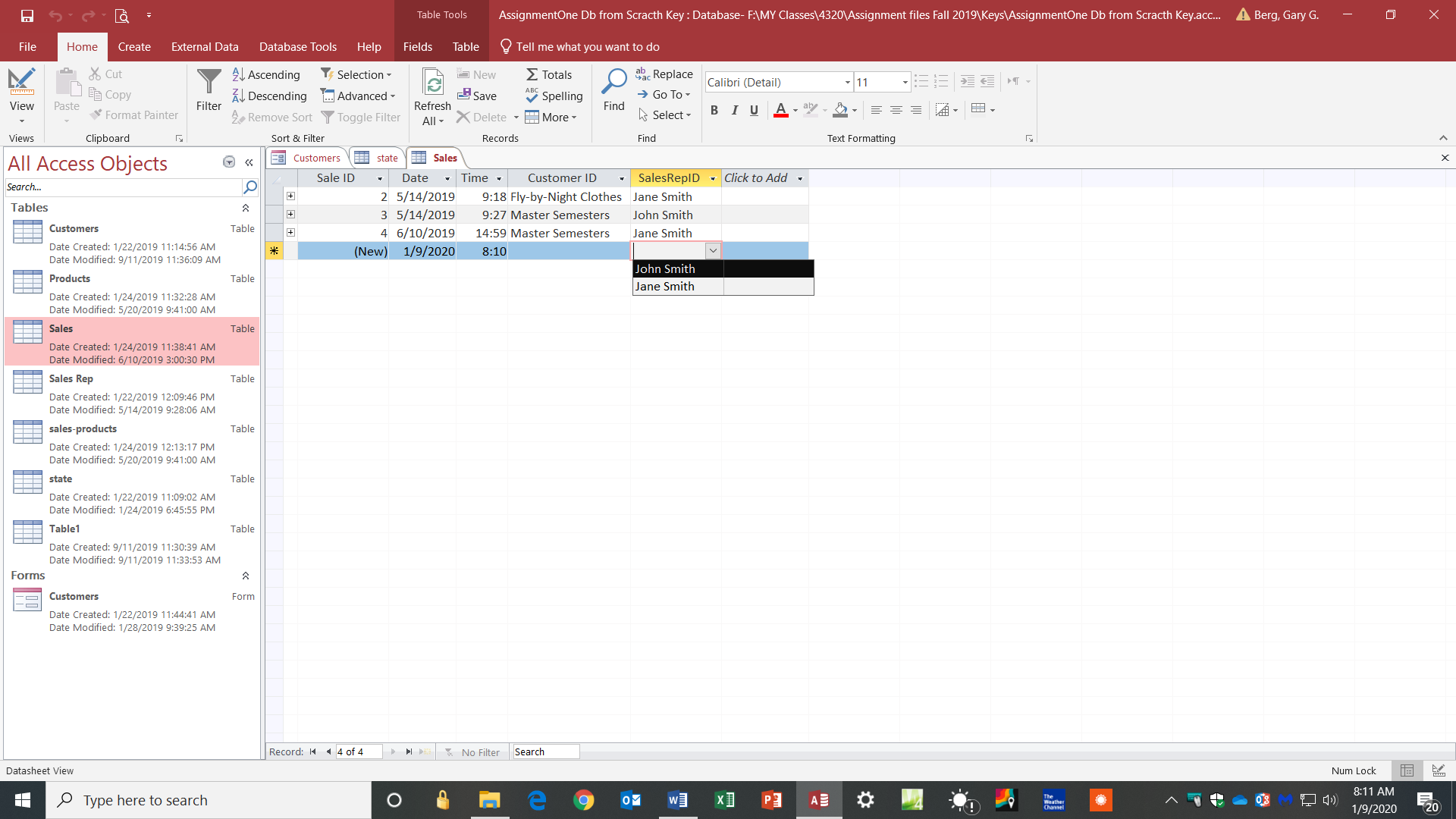
In table design view define data types and field properties as follow:

|  |  |  |
| --- | --- | --- |
| Data Field | Data type | Field properties |
| **Sale ID** (set as the primary key) | Autonumber |  |
| Date | Date/time | Format Short Date |
| Time | Date/time | Format Short Time |
| Customer ID | ?? | Use access defaults for data type you select |
| SalesRepID | ?? | Use access defaults for data type you select |

Electronic sales forms will enter the date and in many cases the time automatically. This can easily be done by setting the default value for date and time. **Enter Date() and Time() for default values**. This will populate the date and time field automatically as sales records are entered.

To improve data entry for the Customer ID and Sale Rep Fields, **use the data type look-up wizard to create a drop down list to enter the customer name and sales rep name.** The Sale Rep field should show both first and last name and will require you to make a minor change to the look up row source.

**SELECT [Sales Rep].[Sales Rep ID], [Sales Rep].[First Name] &" "& [Sales Rep].[Last Name] FROM [Sales Rep];**



To improve completeness, change the **Required** property for each of the field names from **No** to **Yes**; this prevents a record from being created if any values are missing.

**For this table, you do not need to enter any data into this table**

**Step 10**

Create a relationship table called **sales-products. This table should contain the following fields.**

* Sale ID
* Product ID
* QTY sold
* Selling Price

|  |  |  |
| --- | --- | --- |
| Data Field | Data type | Field properties |
| **Sale ID** (set as the primary key) | ? |  |
| **Product ID** (set as the primary key) | ? |  |
| Qty Sold | Number | Use access default |
| Selling Price | Currency | Format 2 decimal places |

**For this table, you do not need to enter any data into this table**

To improve completeness, change the **Required** property for each of the field names from **No** to **Yes**; this prevents a record from being created if any values are missing.

**Step 11**

**There is a relationship between the sales table and three of the other tables and a relationship between the products table and one other table (I will not tell you which ones). Establish the relationships. Make sure you enforce referential integrity for all relatiosips.**

**Primary key/Foreign Key**

* **Establishing relationship**
* **Primary key/Foreign key must be like type data types**
  + **Auto# cannot be FK because it can only be used one time**

****