

## Apply-It #3, Beyond the Book

Download and open the **Apply-It 3 Start File** from Blackboard. Save the workbook as **FirstName.Lastname.Apply-It 3**. Save the workbook as a Macro-enabled workbook or you will lose all the Macros you are about to create.

### Part 1

Roberta Olson is a dispatch manager for Milwaukee Cheese, a large cheese and dairy supply company operating in the Midwest. One of Roberta's jobs is to provide the shipping assignments to the company drivers among 27 Wisconsin cities. It is company policy that no driver logs more than 350 miles in a single day driving from one distribution center to the next. You will help Roberta develop an Excel application for entering driving assignments that fulfill company policy.

Roberta's worksheet includes a Driving Form worksheet in which the user will enter commands to store each leg of a driving itinerary. The legs will then be stored in the Itinerary table of the Driving Itinerary worksheet. The distances between the 27 cities are stored in the Distance Table worksheet.

In this application, you will use validation tests to ensure that a driving leg is one of the 27 cities, and the total driving distance does not exceed 350 miles. You will create a macro to add driving legs from the Driving Form to the Driving Itinerary.

In the **Driving Form** worksheet, the user will enter individual legs of a driving itinerary. Notice that we already have 2 legs of our itinerary, listed in the Driving Itinerary worksheet. We will begin our 3<sup>rd</sup> leg in Eau Claire and end in Beloit, which is 40 miles as shown in the Distance Table worksheet.

1. Review the **Driving Itinerary** worksheet and see that you have already logged 2 legs of the driver's itinerary, ending the 2<sup>nd</sup> leg in the city of Eau Claire.
2. In cell C6 of the **Driving Form** worksheet, type the next leg number, **3**. In cell E6 type the Starting City for leg 3, **Eau Claire**. Type the Ending City, **Beloit**, in cell E6.

Enter Proposed Leg			
Leg	Starting City	Ending City	Distance
3	Eau Claire	Beloit	
	Appleton	Appleton	
	Beloit	Beloit	

3. Create a formula in cell F6 to calculate the mileage between starting and ending cities using information in the **Distance Table** worksheet. If the formula produces an error (such as when cells D6 or E6 are blank), display a blank text string in place of an error value. Use the named range **cities** and **distance\_table** in your formula. Verify that your formula calculates the distance between Eau Claire and Beloit as 40 miles. *Hint: Index, Match, IFError*

4. Create a formula in cell C6, replacing the number 3, which calculates the next leg needed to be added to the legs in column B of the **Driving Itinerary** worksheet. As the user adds legs, the number should increment by 1. *Hint: MAX formula*
5. The Driving Entry Form will use the AutoComplete feature to fill in the city names in cells D6 and E6. There is no need to display the city names in rows 7 through 33 for AutoComplete to work. Hide rows 7 through 33 in the worksheet so that content doesn't distract the user.
6. In this application, you will combine validation tests within a single cell using a custom validation formula. For the range D6:E6, create a custom validation rule ensuring the driving leg is valid. For a driving leg to be valid both of these rules must be true:
  - a. Start and end in one of the 27 cities. City names are listed in the **cities** named range. *Hint: use COUNTIF to check if the starting/ending cities in D6/E6 match your named range. Section EX 12-7e, Insight.*
  - b. The total driving distance, stored in the **dist** named range, should not exceed 350 miles.
7. If the validation rule is violated, display a warning box with the title **Invalid Data** and the message **You either mistyped the city name or adding this leg will result in a total driving distance exceeding 350 miles.**
8. Create a macro for this workbook named **Add\_Leg** with the description **Add a leg to the driving itinerary**. The macro should perform the following tasks:
  - a. Use Ctrl + G (or F5) to go to the travel\_end cell.
  - b. Insert a new sheet row directly above the travel\_end cell on the **Driving Itinerary** worksheet. *Hint: Insert the row in such a way that only one cell is active after the insert row is complete. Do not insert the row by selecting the row number to select the entire row.*
  - c. Go to the **Driving Form** worksheet and copy the values in the range C6:F6 to the clipboard.
  - d. Return to the **Driving Itinerary** worksheet and paste the values and number formats into the active cell of the worksheet.

- e. Go to the **Driving Form** worksheet and prepare the entry sheet for the next leg:
    - Copy the value from the Ending City, cell E6, into the Starting City, cell D6.
    - Delete the contents of the Ending City, cell E6.
  - f. End the macro
9. Your Driving Itinerary should now have 3 legs, ending with the city of Beloit. Insert a macro button in the range C35:D36 to play the **Add\_Leg** macro. Change the label of the macro button to **Add Leg to Itinerary**.
  10. Use the macro and the data form to enter the following legs into the driving itinerary:
    - Beloit to Madison
    - Madison to La Crosse

	A	B	C	D	E
1		<b>Milwaukee Cheese</b>			
2		Driving Itinerary			
3					
4		Total Distance	323	miles	
5					
6		Leg	Starting City	Ending City	Distance
7		1	Milwaukee	Brookfield	13
8		2	Brookfield	Eau Claire	87
9		3	Eau Claire	Beloit	40
10		4	Beloit	Madison	49
11		5	Madison	La Crosse	134
12		Travel End			

11. Verify that when you try to enter a new driving leg of La Crosse to Marshfield, the application warns you that you are about to exceed the allowed driving distance. Do not enter Marshfield as the last leg of the trip.
12. When a macro switches between worksheets, the quick jump from one sheet to another can be distracting. To correct this problem, edit the Add\_Leg sub procedure in the Visual Basic for Applications editor.
  - a. Directly after the initial comment section at the top of the sub procedure insert the following command to turn off screen updating while the macro is running:
 

```
Application.ScreenUpdating = False
```
  - b. Directly before the closing **End Sub** command at the bottom of the sub procedure, insert the following command to turn screen updating back on:
 

```
Application.ScreenUpdating = True
```

13. Insert a macro button next to your **Add Leg to Itinerary** button

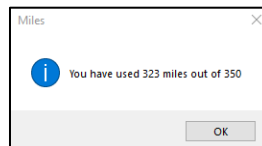
- a. The label of this button should be **Miles Used**
- b. When pressed, a message box should pop up saying **You have used XXX miles out of 350**, where XXX is the distance in the named range dist.  
*Hint: Create a new subroutine using the function **MsgBox** (prompt, [ buttons, ] [ title, ] ).*
  - i. For the *prompt* argument, use the **&** to combine text and the variable Range("dist") to display "You have used XXX miles out of 350".
  - ii. Use the **buttons** argument to display the *i* information icon.
  - iii. Use **Miles** as the title of the message box.
  - iv. Name the macro **Miles**.

**Enter Proposed Leg**

Leg	Starting City	Ending City	Distance
6	La Crosse		

Add Leg to Itinerary

Miles Used



14. In the **Driving itinerary** worksheet, create a macro named **PrintOrder**. Assign the shortcut key Ctrl+D. The macro should:

- a. Set the Order Form print area to the range B1:E20
- b. Center the print area horizontally.
- c. Add a footer in the right-hand side of the page showing the current date and time (the date/time stamp will be updated every time you run the macro).
- d. Notice that there is a comment in cell B2. Ensure the comment does not print when you run your macro.

Insert the printer icon or use a similar image (but not a form control button) and assign the **PrintOrder** macro to the image.

Milwaukee Cheese				
Driving Itinerary				
Total Distance	323	miles		
Leg	Starting City	Ending City	Distance	
1	Milwaukee	Brookfield	13	
2	Brookfield	Eau Claire	87	
3	Eau Claire	Beloit	40	
4	Beloit	Madison	49	
5	Madison	La Crosse	134	
Travel End				



15. You tried to set up protection so that no one would accidentally change the cities and miles on the **Distance Table** worksheet in cells B4:AC31. However, someone was able to change some cells so the worksheet protection setup isn't quite right.
- Unprotect the worksheet.
  - Create a conditional format rule to highlight all the unlocked cells, starting in cell B4 and ending in cell AC31. *Hint: use the CELL function with the "protect" argument & relative reference when creating your formula.*
  - Fix the cells that are set up incorrectly so they will be locked when you protect the sheet. Do not remove the conditional formatting.
  - Protect the worksheet with NO password.

Save the workbook as a macro-enabled workbook

## PART 2

Selene Marados is an account executive with Talcma, a tech company that specializes in custom software for hospitals and clinics. Selene wants to create a report on software sales to hospital and clinics in the previous year. Selene has stored sales data in a workbook containing a Data Model with 4 tables:

- an **Agents** table containing information about the sales staff
- a **Hospitals** table containing data on the hospitals and clinics that purchase software licenses from Talcma
- a **Products** table containing data on the software products sold by the company
- a **Sales** table that contains information on each sales transaction.

You will use these tables to generate an analysis of the sales data.

Sales agents working at Talcma are paid a base salary plus a commission based on the percentage of revenue they generate beyond \$350,000. Commission rates can vary from 4% to 6%. Selene wants the report to include the total compensation paid to the sales staff from the base salary and their commissions. Complete the following:

- Review the Tables and Measures worksheet and the relationship between the tables in the Data Model.
- In the Revenue Report worksheet, in cell **B4**, create the PivotTable seen in the screenshot below. Points will be taken off if the pivot tables are not placed in the

cells as requested (it breaks my grading macro if the data isn't in the expected cells).

Date (Month)	Rank	Total Revenue	Running Total	Percentage
Jan				
Feb				
Mar				
Apr				
May				
Jun				
Jul				
Aug				
Sep				
Oct				
Nov				
Dec				

- a. Use the **Date** field from the **Sales** table.
  - b. Use the **Revenue** field from the **Sales** table to display sales revenue in four different ways in your table (Rank, Total Revenue, Running Total & Percentage of Running Total).
  - c. Display the **rank** values from largest to smallest, meaning the month with the highest Total Revenue will be given the rank of 1.
  - d. The **Running Total** should track the increase in revenue over the year.
  - e. Display the **Percentage** as a percentage of the running total. The percentage for December should be 100%
  - f. Set the report layout so that grid lines show on the pivot table.
3. Examine how sales vary by region and agent. Go to the **Region Report** worksheet and create a PivotTable in cell **B4** as shown below.

Sales Region	Name	Total Revenue	Percent
☺ Southeast			
☺ West			
☺ Midwest			
☺ Northeast			
☺ Southwest			
Grand Total			

- a. Use the **Sales Region** and the **Name** fields from the **Agents** table.



Name	Base Salary	Commission Rate	Revenue Generated	Earned Commission	Total Compensation

- a. Use the **Name** field from the **Agents** table.
- b. Use the **Base Salary** field from the **Agents** table.
- c. Use the **Commission** field from the **Agents** table.
- d. Use the **Revenue** field from the **Sales** table.
- e. Use the **Sales Minimum** field from the **Agents** table.
- f. Use **Earned Commission** from the **Sales** table
- g. Use **Base Salary** from the **Agents** table
- h. Set the report layout so that grid lines show on the pivot table.
- i. Create a measure named **Earned Commission** in the **Sales** table with the description **Commission earned by revenue generated above a minimum amount** that calculates the amount of commission earned by a sales agent for revenue generated above a minimum level.
  - i. Use a DAX formula to create a formula where  
 $(\text{Revenue} - \text{Sales Minimum}) * \text{Commission}$
  - ii. *Hint: Insert an equal sign to start the formula and match all closing and opening parentheses. Use the SUM function for each of your fields in your calculation. See EX11-14 Creating a Measure*
- j. Create a measure named **Total Compensation** in the **Sales** table with the description **Total compensation from base salary and earned commission** that calculates the total compensation for each sales agent by adding the base salary and the earned commission.
  - i. Use a DAX formula to create a formula where  
 $\text{Base Salary} + \text{Earned Commission}$

ii. *Hint: You don't need a function for the **Earned Commission** field because it is a measure*

- k. Sort the agent names in descending order of the **Total Compensation** column.

Congratulations, you have finished your last Apply-It assignment!

Submit to Blackboard. Celebrate your accomplishments and all you have learned this semester.