## **Hands-on Lab: Building Multi-language Reports for Power BI**

**Overview**: In this lab, you will learn how to work with Power BI Desktop and [**Translations Builder**](https://github.com/PowerBiDevCamp/TranslationsBuilder) to build multi-language reports for Power BI. You will start by downloading a PBIX project file named **Product Sales.pbix** and opening it in Power BI Desktop. Once you have opened **Product Sales.pbix,** you will launch Translations Builder and move through the steps to add metadata translations and report label translations. At various milestones in this lab, you will be required to publish the **Product Sales.pbix** project from Power BI Desktop to the Power BI Service so you can test how the translations you’ve added to the PBIX project to ensure they are displayed to users when loaded using different languages and different cultures.

**Prerequisite 1**: To complete this lab, you will need a Power BI workspace where you have appropriate permissions to publish PBIX project files from Power BI Desktop to test your work. This workspace must also be associated with a Premium capacity as indicated by the diamond image (*see below*) displayed after the workspace name. A workspace in a premium capacity is required because Power BI translations do not load properly for reports in the shared capacity.



**Prerequisite 2**: This lab assumes you’ve already installed Translations Builder. If you haven’t installed Translations Builder yet, you must follow steps in the [Translations Builder Installation Guide](https://github.com/PowerBiDevCamp/TranslationsBuilder/blob/main/Docs/Installation%20Guide.md) before continuing with these lab exercises.

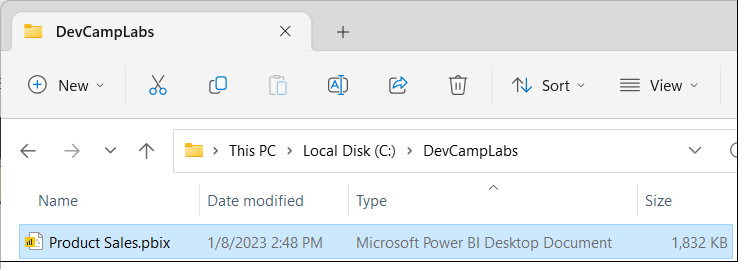
### Exercise 1: Creating and Testing Metadata Translations

In this exercise, you will begin by downloading a PBIX file with a simple data model and a single page report. You will publish the project to the Power BI Service to set up a workflow process to test how translations appear to report consumers. At the end of this exercise, you will use Translations Builder to add metadata translations for the column names and measure names in the dataset.

1. Download the PBIX starter file named **Product Sales.pbix** from the following link

<https://github.com/PowerBiDevCamp/TranslationsBuilder/raw/main/Labs/StarterFiles/Product%20Sales.pbix>

1. Create a new folder on your local hard drive for these lab exercises in a location such as **C:\DevCampLabs\**.

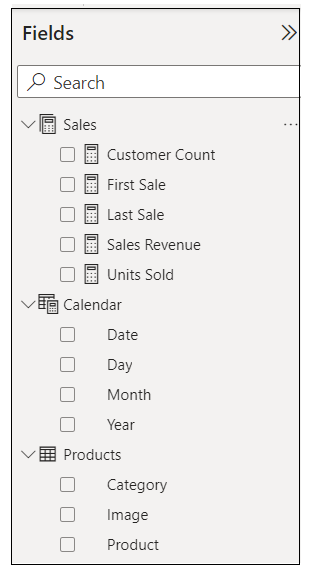


1. Copy **Product Sales.pbix** into the lab folder and then open it in Power BI Desktop to examine the report inside.

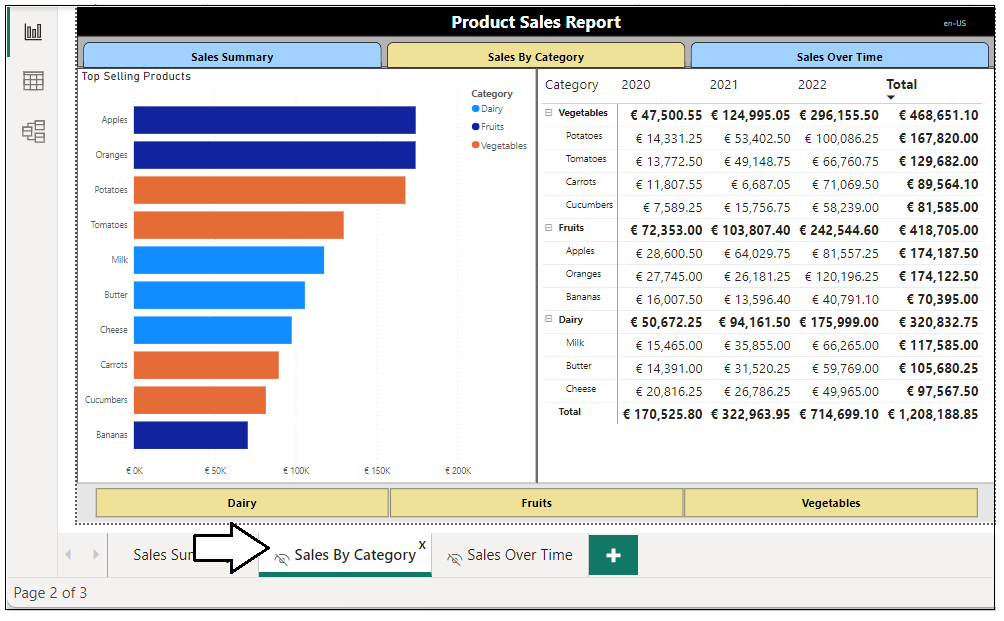
Graphical user interface, table

Description automatically generated

1. While in **Report view**, examine the **Fields** list to see the tables, columns and measure that are not hidden.

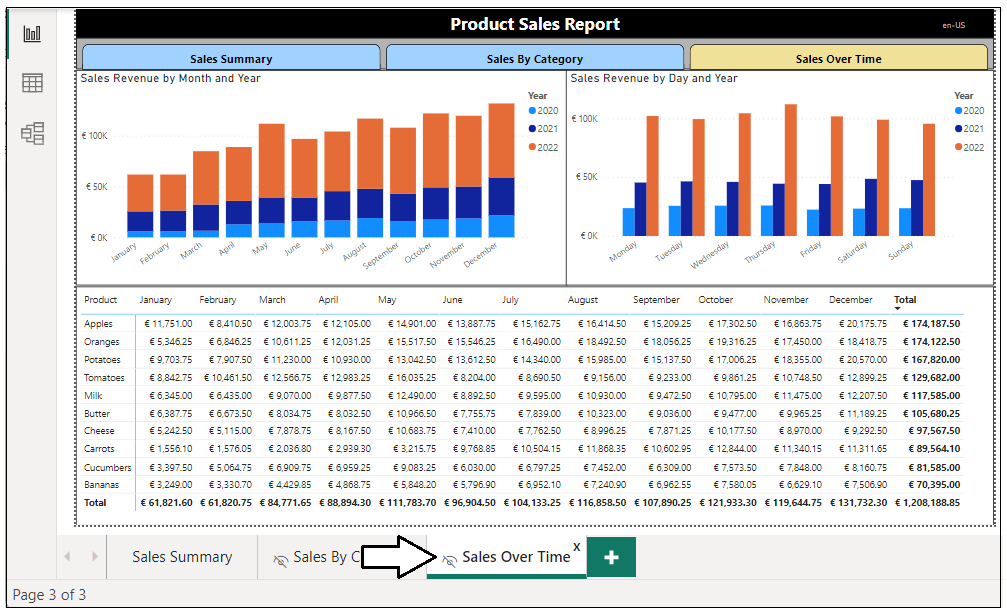


1. Look at the second page.



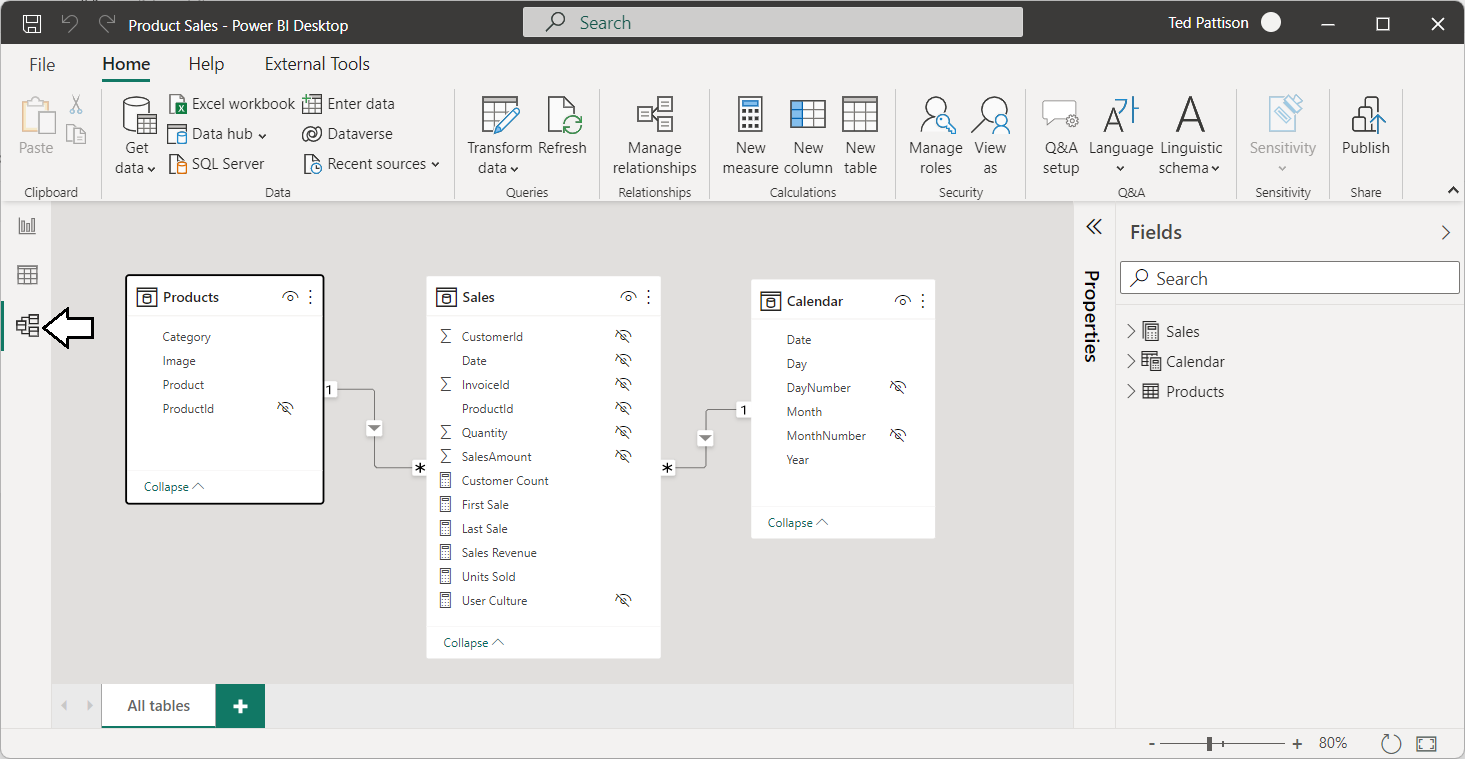
Data translations for product names are addressed in **Exercise 5: Implementing Data Translations using Field Parameters**.

1. Xxxxx

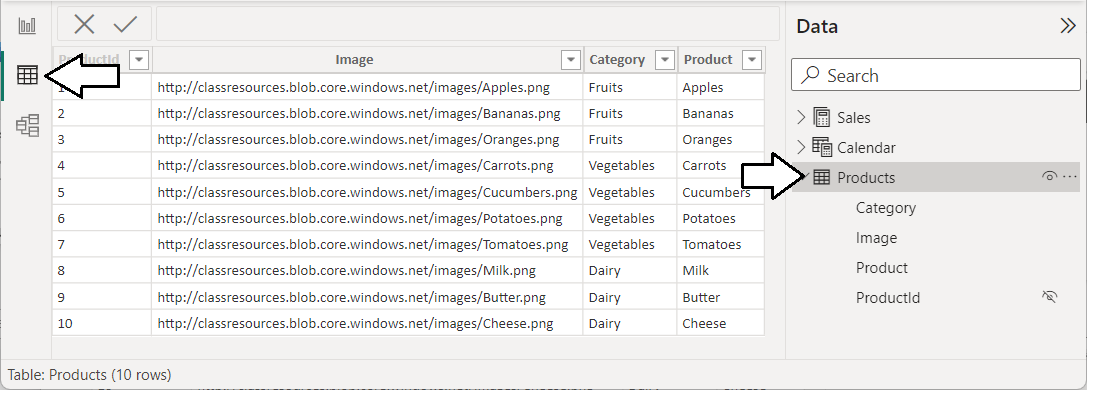


Data Translations for month and day names are addressed in **Exercise 6: Implementing Data Translations for a Calendar Table**.

1. Now, navigate to **Model view** so you can see the entire data model including the columns hidden from **Report view**.

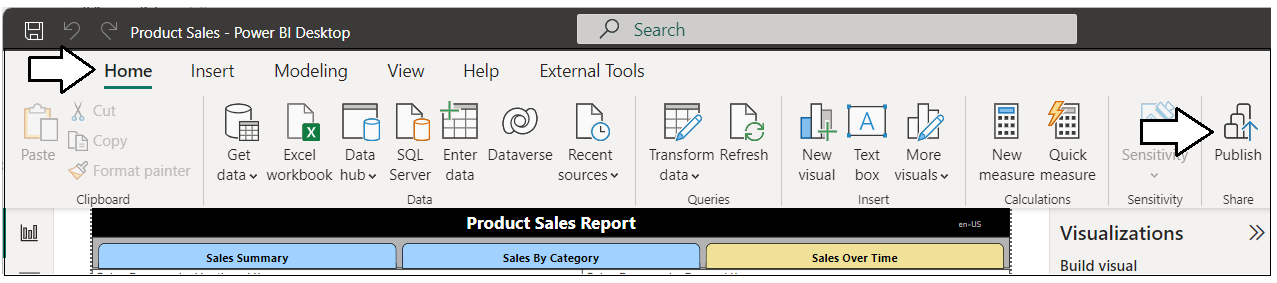


1. Navigate to **Data view** and examine the rows of the **Products** table.



Now you are going to publish the **Product Sales.pbix** project to a workspace in the Power BI Service.

1. Navigate to the **Home** tab and then click the **Publish** button.

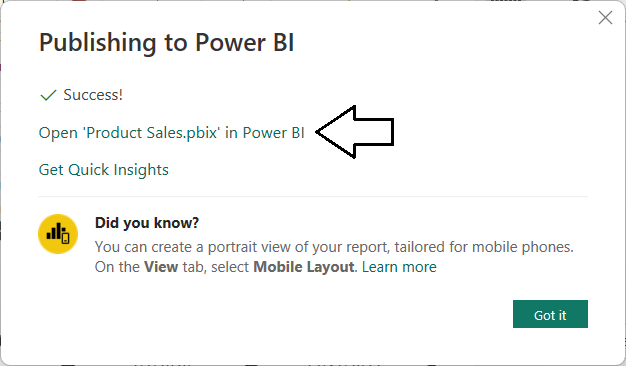


1. When prompted by the **Publish to Power BI** dialog, choose your test workspace and then click **Select**.

Graphical user interface, text, application, email, Teams

Description automatically generated

1. Once you see **Success!**, click **Open ‘Product Sales.pbx’ in Power BI** to view the report in the Power BI Service.

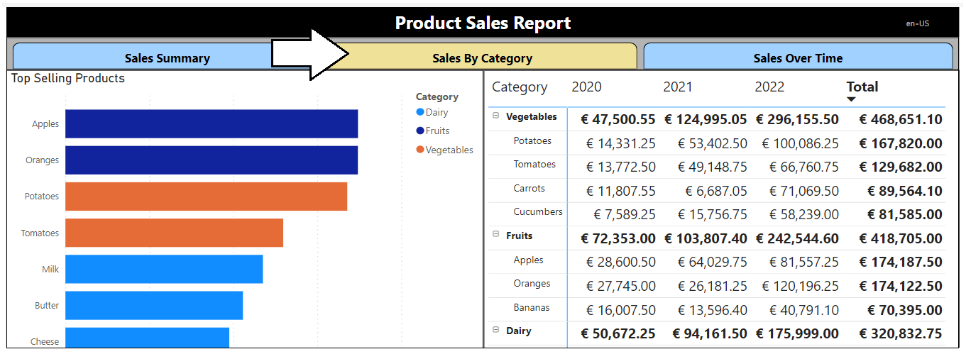


1. The report named **Product Sales** should appear like the report shown in the screenshot below.

Graphical user interface, chart

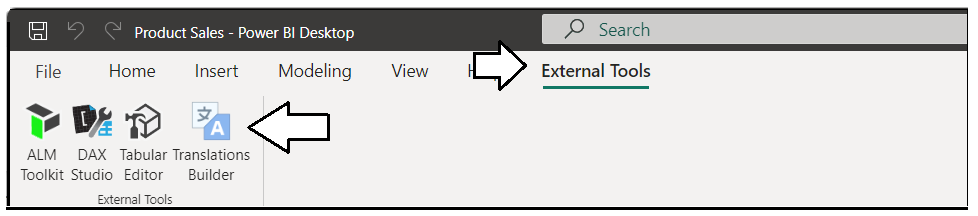
Description automatically generated

1. Navigate to the two other pages using the navigation button under the report title.



Now it’s time to begin adding translations. As you begin to add translations to a PBIX project, you will often follow this set of steps:   
**(1)** make changes in Power BI Desktop, **(2)** publish the project, **(3)** check your work in the Power BI Service, **(4)** repeat until happy

1. Return to Power BI Desktop, navigate to the **External Tools** tab and launch **Translations Builder**.

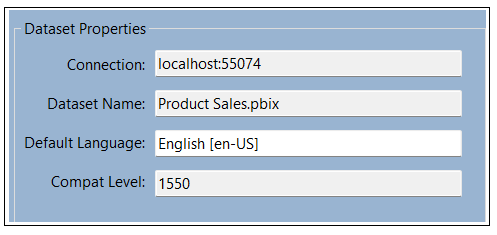


1. Translations Builder should start and load the data model for **Product Sales.pbix** as shown in this screenshot.

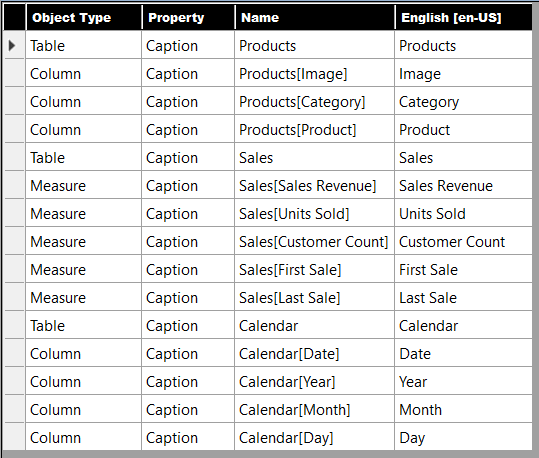
Graphical user interface

Description automatically generated

1. The **Dataset properties** section provides details about the dataset connection and the PBIX project file.

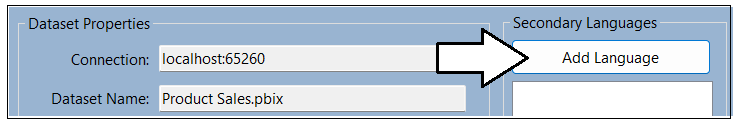


1. The translation grid at the bottom of the page displays one row for each non-hidden dataset object in the data model.

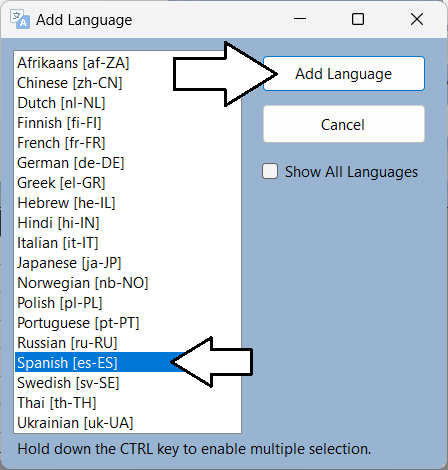


Tables, columns and measures that are hidden from report view in the data model are not displayed. You don’t need to translate them.

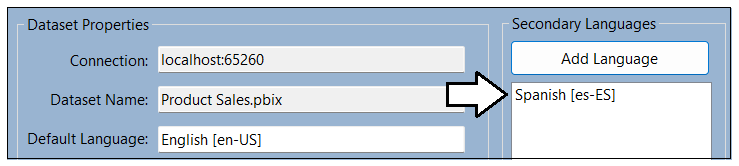
1. Click the **Add Language** button to add your first secondary language.



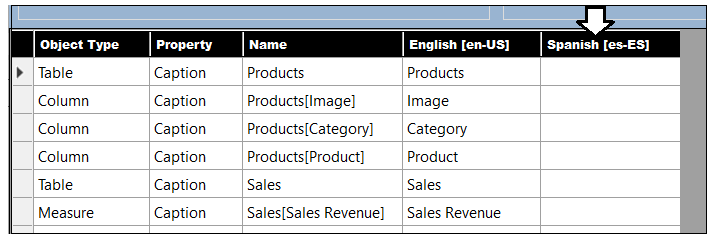
1. Select **Spanish [es-ES]** and click **Add Language**.



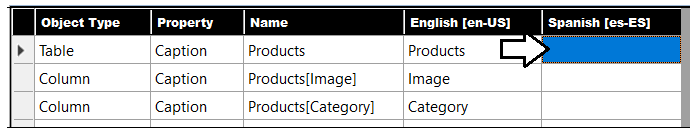
1. You should now see that **Spanish [es-ES]** appears as the first language in the **Secondary Languages** list.



1. You will also notice that a new column has been added for Spanish translations.



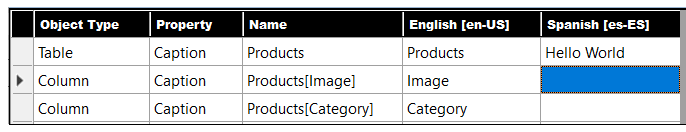
1. In the row with the **Products** table, click on the cell for the **Spanish** column. It should turn blue when selected.



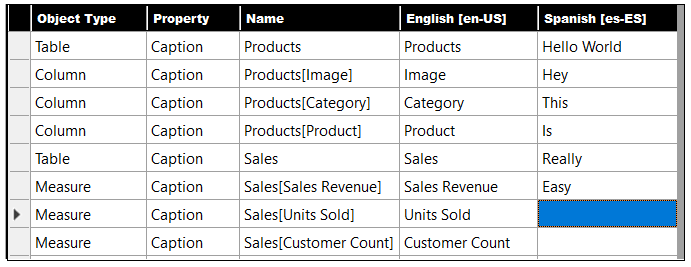
1. Type **Hello World**. You should see that you can just start typing in the selected cell to add or edit a translation.



1. Press the **ENTER** key to save your changes. Note that pressing **ENTER** will move the selection to the cell below.



1. Now, type more text and press **ENTER** repeatedly to experiment quickly adding text to all cells in the Spanish column.



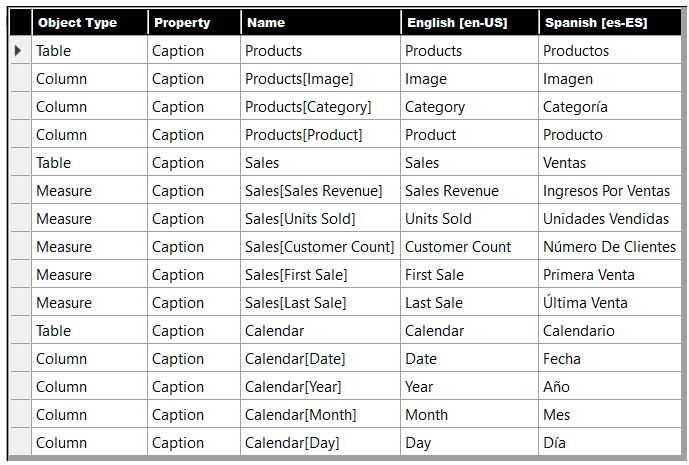
The point of the last few steps has been for you to become comfortable with the translation editing experience. You can see the grid provides an editing experience similar to working with Excel. You can even use the **{F2}** key to move a cell with content into edit mode.

1. Now add Spanish translations so that no columns remain empty. Use the following translations if you need them.

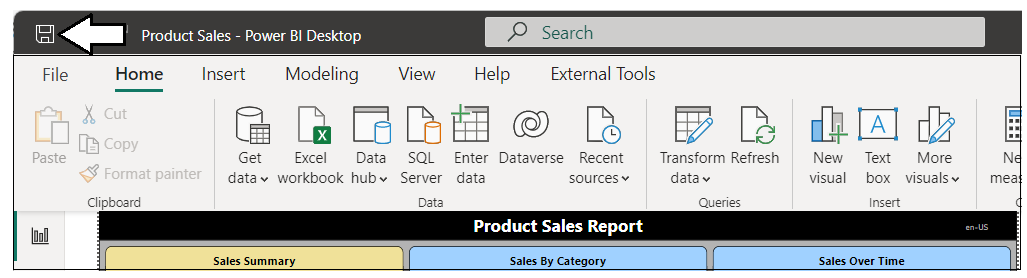
|  |  |
| --- | --- |
| **English** | **Spanish** |
| Products | Productos |
| Image | Imagen |
| Category | Categoría |
| Product | Producto |
| Sales | Ventas |
| Sales Revenue | Ingresos Por Ventas |
| Units Sold | Unidades Vendidas |
| Customer Count | Número De Clientes |
| First Sale | Primera Venta |
| Last Sale | Última Venta |
| Calendar | Calendario |
| Date | Fecha |
| Year | Año |
| Month | Mes |
| Day | Día |

It's OK if you don't speak Spanish and don't want to type in all the translations from the table. Just add some text for each translation.

1. When you are done with your edits, the Spanish translations should match the following screenshot.

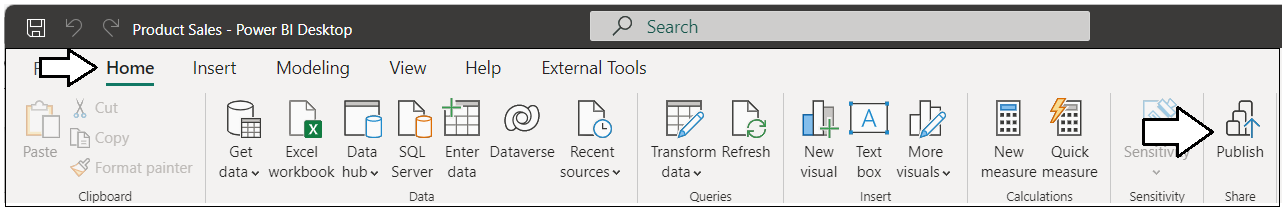


1. Return to the **Product Sales.pbix** project in Power BI Desktop and save your work by clicking the **Save** button.

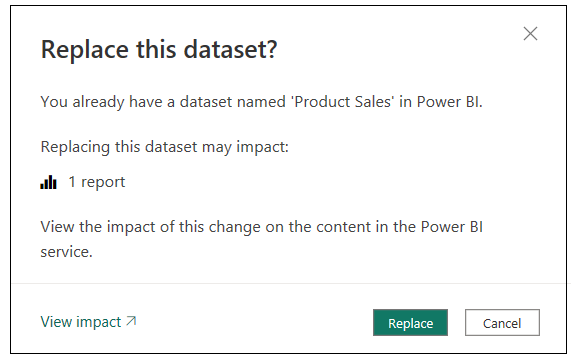


It’s easy to forget to save your changes in Power BI Desktop. Be aware that any changes made by Translations Builder are just made to the data model loaded in memory. None of your changes are saved back to the PBIX project file until you save in Power BI Desktop.

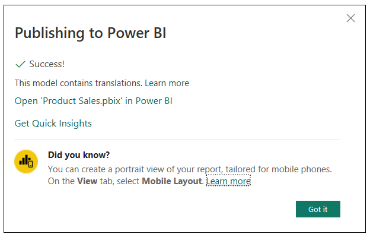
1. Publish the **Product Sales.pbix** project to push the changes to the project’s translations to the Power BI Service.



1. When prompted by the **Replace this dataset?** Dialog, click the **Replace** button to continue.



1. Once you see **Success!**, click **Open ‘Product Sales.pbx’ in Power BI** to view the report in the Power BI Service.



1. The report should load with its default behavior showing all text in English at first.

Graphical user interface, chart

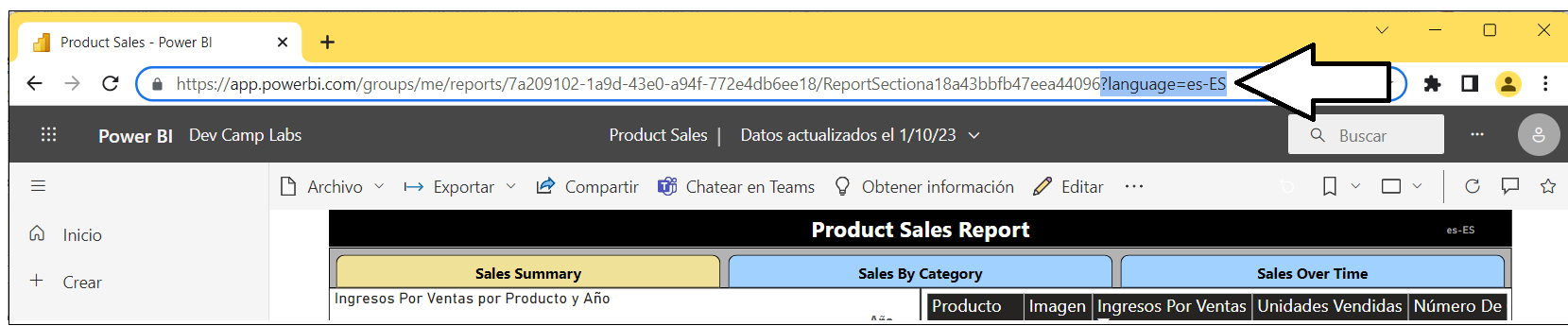
Description automatically generated

Now, it’s time to test your translations. You will accomplish this by using the **language** query string parameter to load the report.

1. Click the browser address bar and add the following **language** parameter to the end of the report URL.

/?language=es-ES

1. Press **ENTER.** You should see the **language** query string parameter accepted by the browser as it reloads the report.

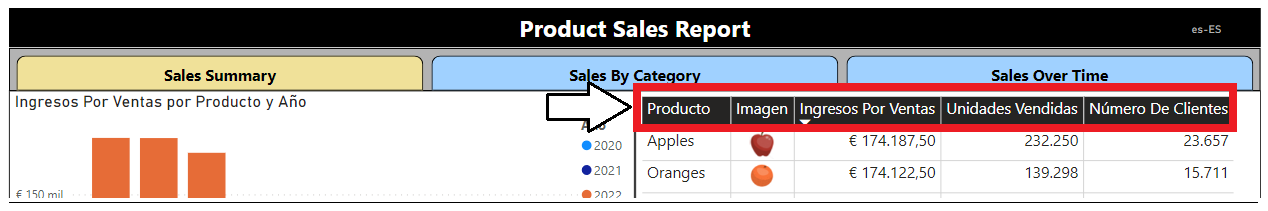


When the report reloads, you should see the UI experience for the Power BI Service UI switch from English to Spanish.

1. Inspect the Card visuals at the bottom of the Sales Summary page. Verify the column and measure names are displayed with the Spanish translations you added.



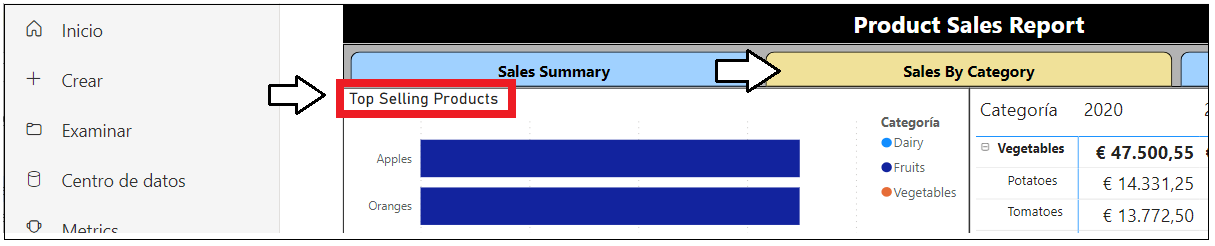
1. Inspect the column headers in the Table visual. Verify the column and measure names are displayed with the Spanish translations you added.



1. Inspect the Column Chart visual. Verify the title and legend title are displayed with the Spanish translations.



1. Navigate to the **Sales By Category** page.
2. You should see the title of the Bar Chart visual is not translated because the text for the title is hard-coded in the report.



You have now successfully added the metadata translations to display this report in both English and Spanish. Leave Power BI Desktop and Translations Builder open as you will continue using them in your next exercise.

### Exercise 2: Generating Machine Translations

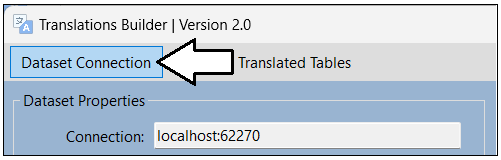
In this lab you will configure Translations Builder’s support to automatically generate machine translations using the Azure Translator Service. While machine translations might not prove good enough to use in all production scenarios, they do provide a great first step in generating translations for testing and getting something into production sooner.

To complete this lab you will require a **Key** and **Location** which provide access to the **Azure Translator Service**. If you do not have a Azure subscription and you cannot obtain your own Key, you can use the following Key and Location up through February 28, 2023.

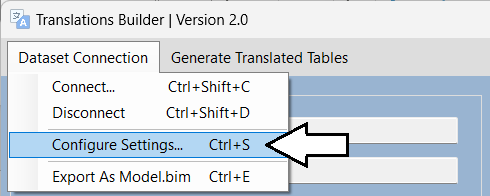
Key: **a75b371ce1fc402ca84a05732cfcff27**

Location: **eastus2**

1. Return to Translations Builder and drop down the **Dataset Connection** menu.



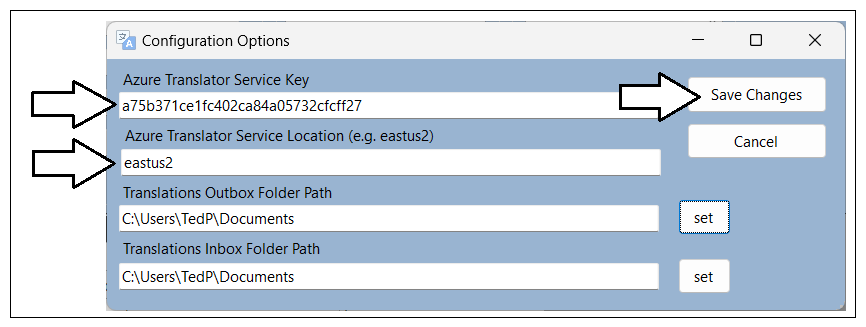
1. Select the **Configure Settings…** menu command to display the **Configuration Options** dialog.



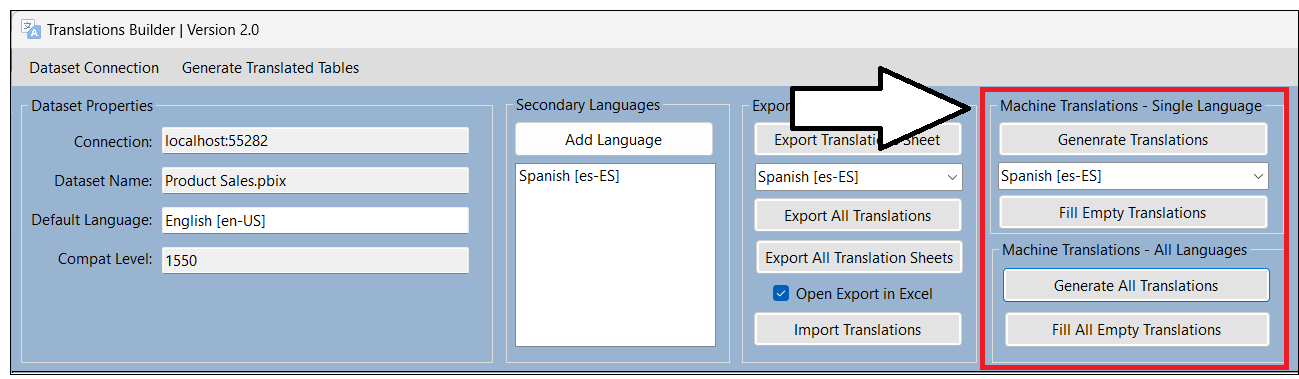
1. In the **Configuration Options** dialog, enter the **Key** and **Location** for the Azure Translator Service.



1. Once you have added the **Key** and **Location**, click **Save Changes**.

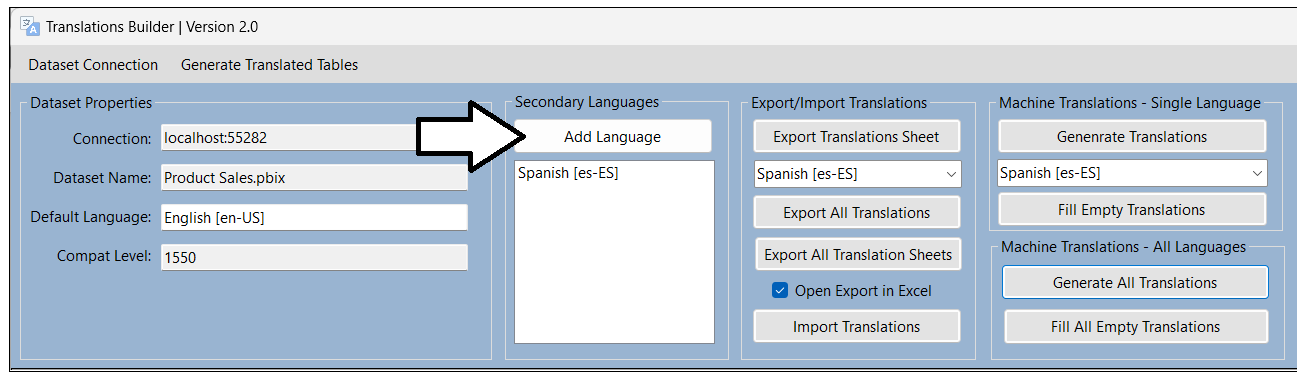


1. After you have configured the **Key** and **Location** for the Azure Translator Service, new controls will appear on the main page.



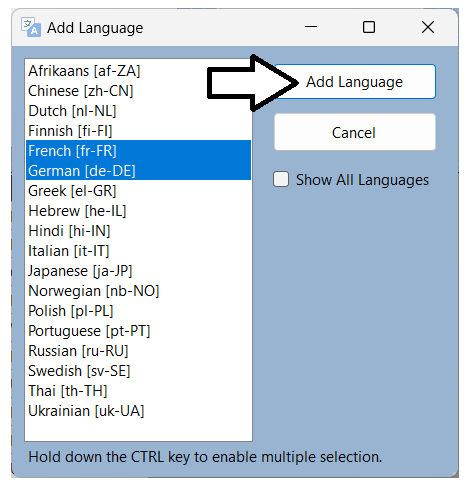
Now that you have added support for generating machine translations, it’s time to put that automatic translation support to work!

1. Click the **Add Language** button to add your second secondary language.

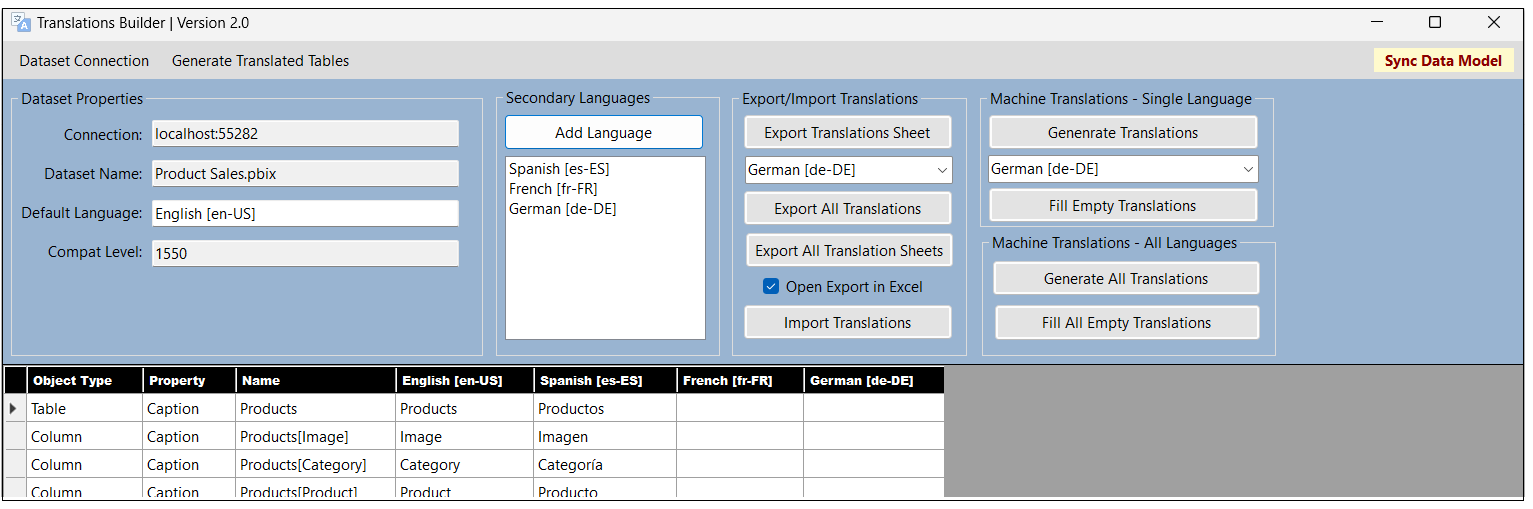


You can hold down the **CTRL** key in the **Add Language** dialog while selecting languages to enable multiple selection

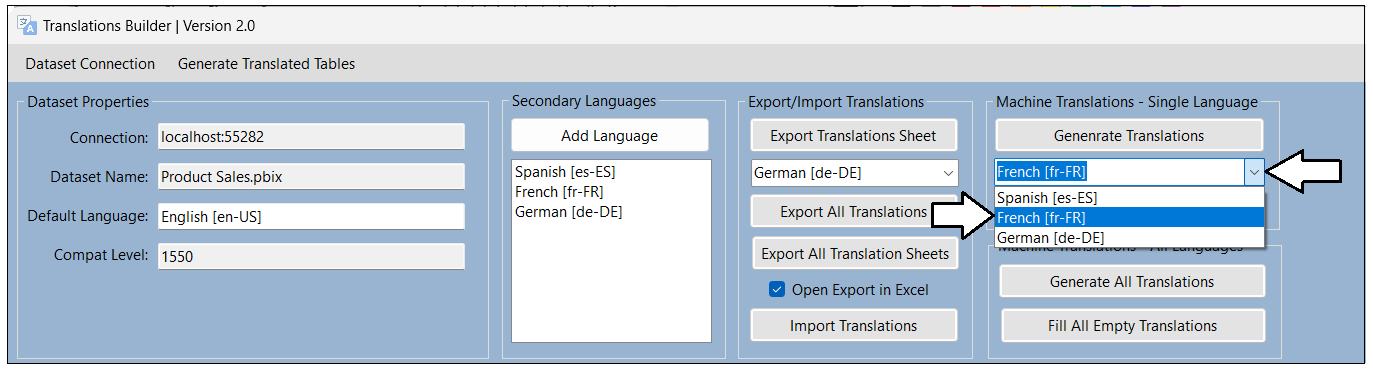
1. Hold down the **CTRL**, and select **French** and **German**. And then click **Add Language**.



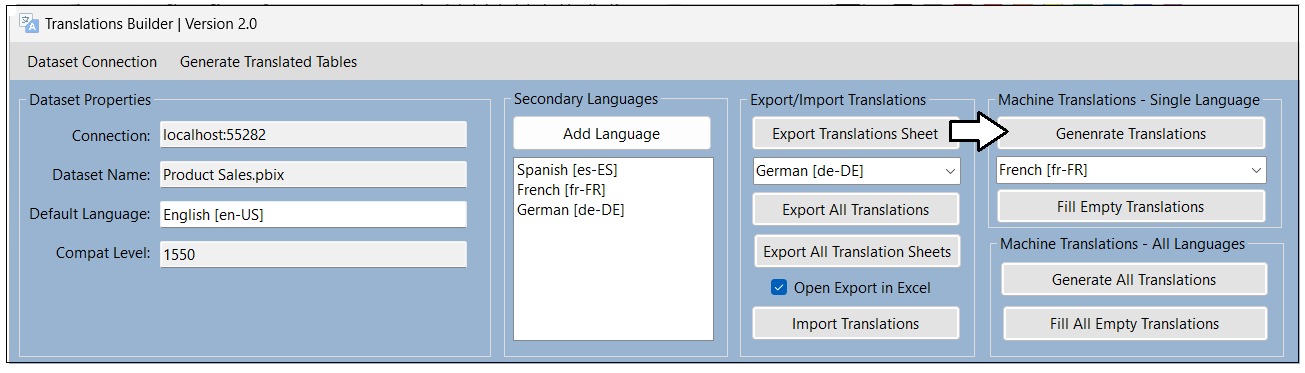
1. You should now see the two new languages appear in the **Secondary Languages** list.
2. You will also notice that new columns have been added to the translation grid for each new language.



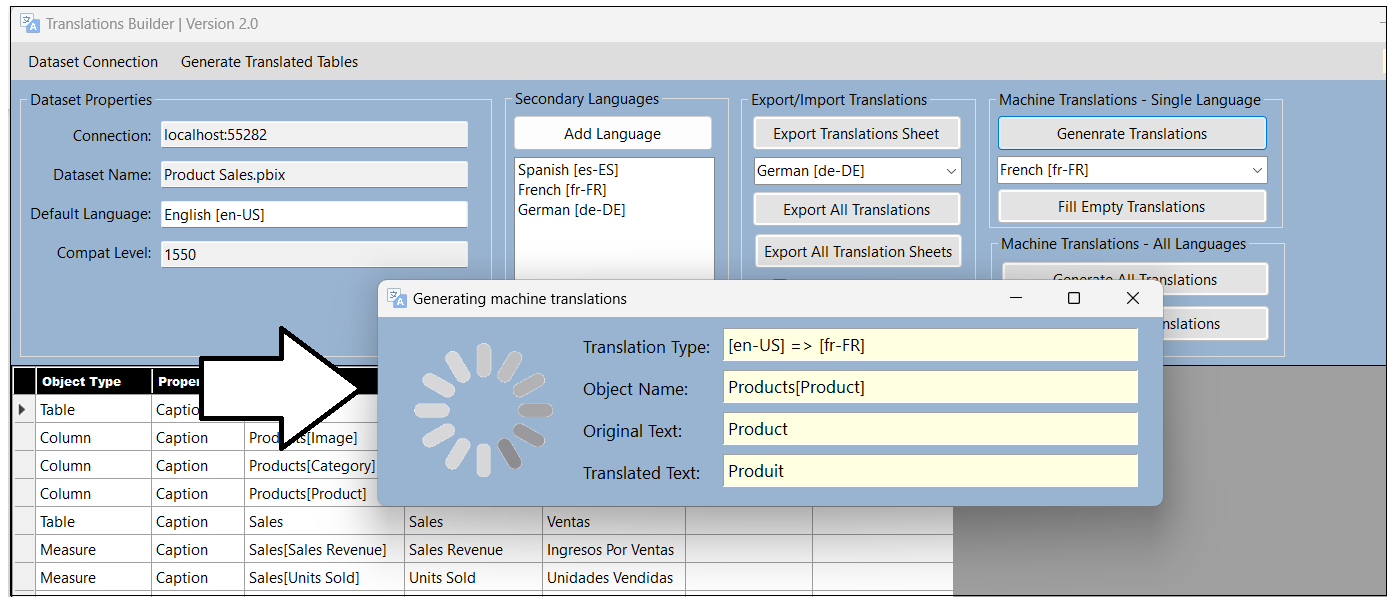
1. In the **Machine Translations – Single Language** section, select **French [fr-FR]** from the drop down menu.



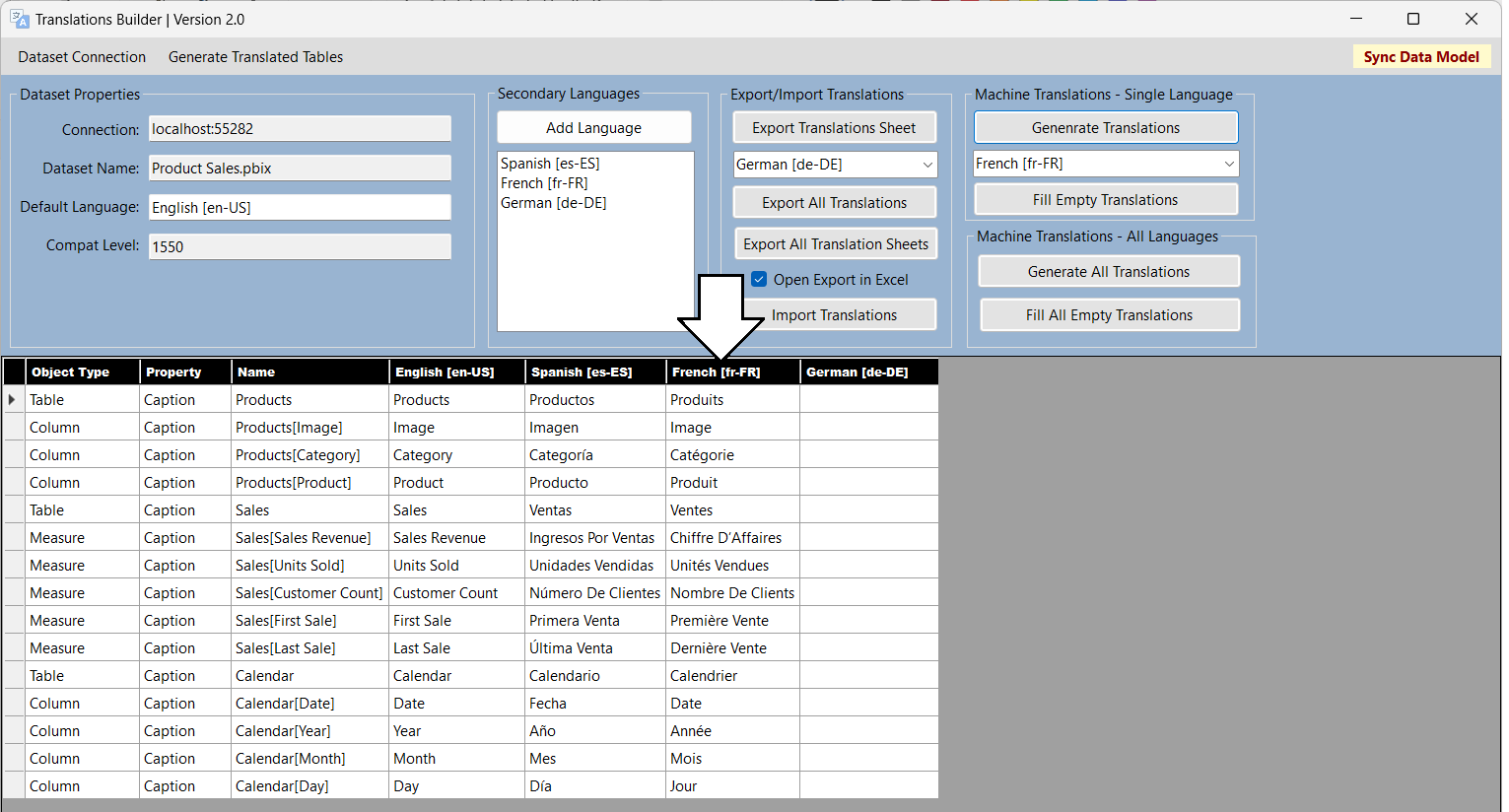
1. Once you have selected **French [fr-FR]**, click **Generate Translations** to create French translations for all rows in the grid.



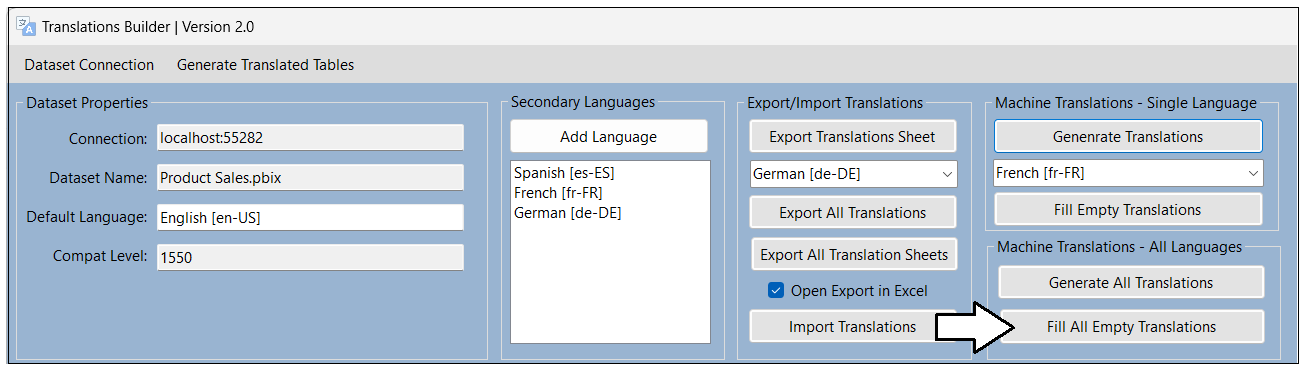
1. As the code runs to interact with the Azure Translator Service, the **Generating machine translation** dialog shows the progress.



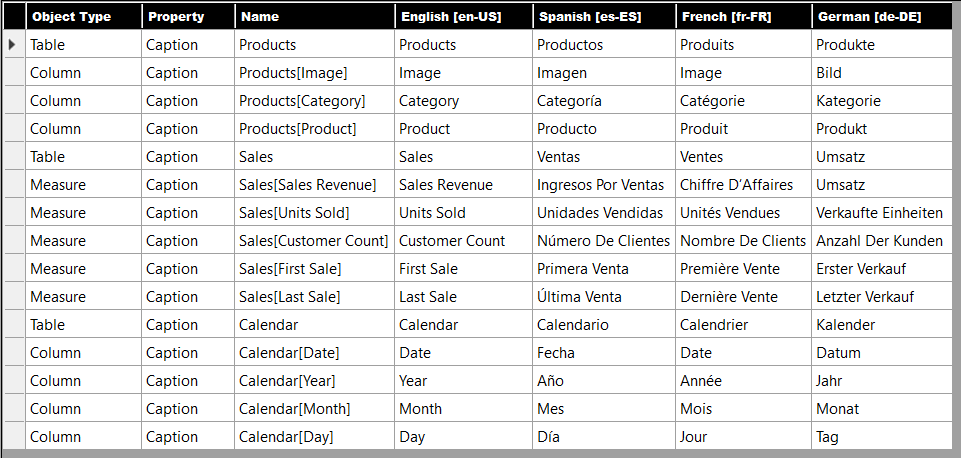
1. Once this dialog closes, you should see all cells the French column has been filled with machine-generated translations.



1. Click the **Fill All Empty Translation** button in the **Machine Translations - All Languages** section.



1. You should see that the empty cells for all languages have now been populated with machine-generated translations.



Now, it’s time once again to test your work in the Power BI Service,

1. Return to the **Product Sales.pbix** project in Power BI Desktop and save your work by clicking the **Save** button.

A screenshot of a computer

Description automatically generated

Don’t forget to save your work! Did we mention it’s easy to forget to save in Power BI Desktop and to lose your work.

1. Publish the **Product Sales.pbix** project to push the changes to the project’s translations to the Power BI Service.

Graphical user interface, application

Description automatically generated

1. When prompted by the **Replace this dataset?** Dialog, click the **Replace** button to continue.
2. Once you see **Success!**, click **Open ‘Product Sales.pbx’ in Power BI** to view the report in the Power BI Service.

Graphical user interface, text, application, email

Description automatically generated

1. The report should load as normal showing all text in English at first.

Graphical user interface, chart

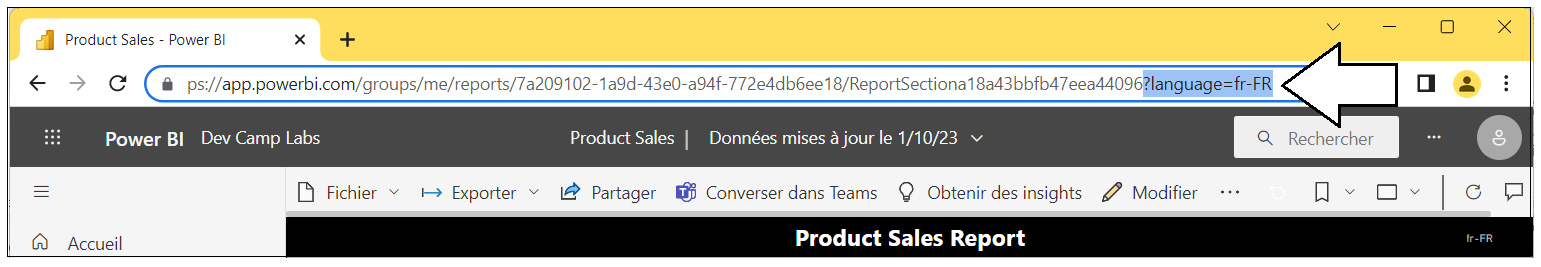
Description automatically generated

Now, it’s time to test your French and German translations using the **language** query string parameter to load the report.

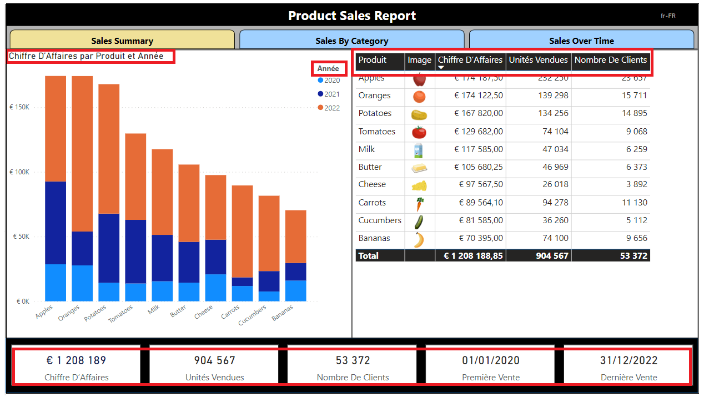
1. Click the browser address bar and add the **language** parameter value of **fr-FR** for French to the end of the report URL.

/?language=fr-FR

1. When the report reloads, you should see the UI experience for the Power BI Service UI switch from English to French.



1. Verify the column and measure names in columns headings of the table visual are displayed with French translations.

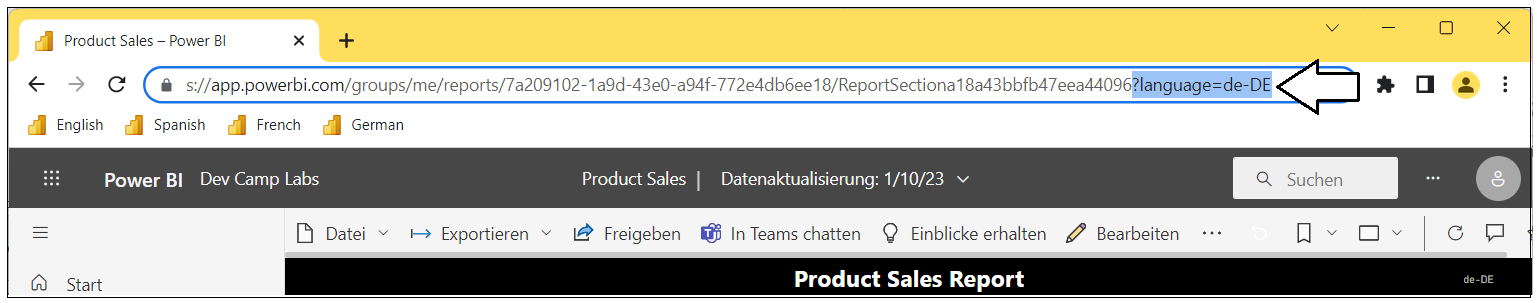


Now that you have tested the French translations, it’s time to test the two other new languages.

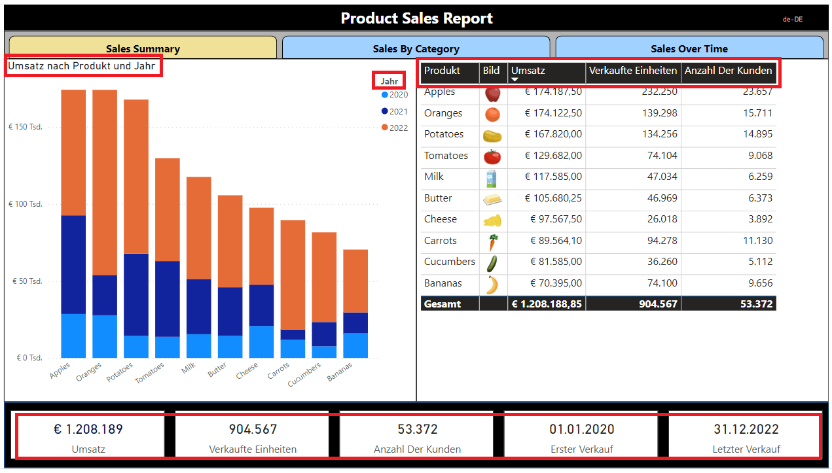
1. Click the browser address bar and add the **language** parameter value of **de-DE** for German to the end of the report URL.

/?language=de-DE

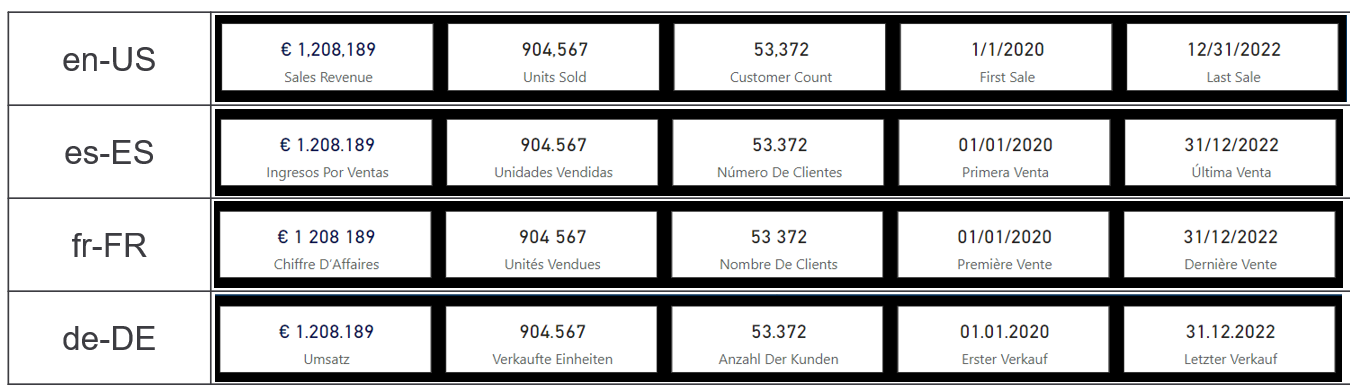
1. When the report reloads, you should see the UI experience for the Power BI Service UI switch to German.



1. Verify the column and measure names in columns headings are displayed with the German translations.



1. Take a moment to appreciate how the date and numeric formatting automatically changes across all four regional locales.



You have now successfully added the metadata translations to display this report in four different languages. Throughout these lab exercises, you will continue to test all four languages in the browser as you add additional translation support.

1. As a final step in this exercise, add a browser bookmark for each language with a **language** parameter at the end.



Creating a browser bookmark for each language might take a minute or two to set up at first. However, it will save lots of time in the long run as you continue to test the translations for this report in the lab work that remains ahead.

### Exercise 3: Creating and Testing Report Label Translations

In this exercise, you will work through the process of adding report label translations. You will add translations for the report title which is **Product Sales Report** and for the title of the table visual which is **Product List by Sales Revenue**. This will give you experience working with the localized labels table strategy that Translations Builder uses to quickly and easily implement report label translations.

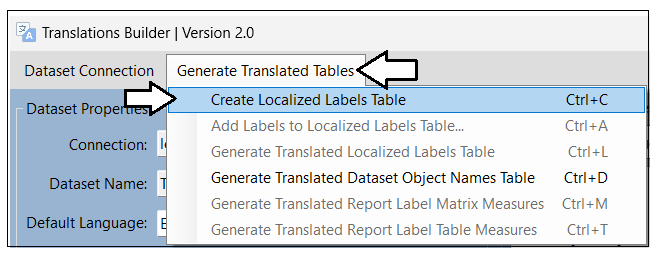
1. Return to the **Product Sales.pbix** project in Power BI Desktop and move to **Report view**.
2. The text **Product Sales Report** is displayed with a rectangle shape object.



1. The text **Product Sales List by Sales Revenue** is displayed using the **Title** property of the table visual.



1. Xxxx
2. Now, move back to Translations Builder and drop down the **Generate Translated Tables** menu.
3. Select the **Create Localized Labels Table** to create the **Localized Labels Table**.



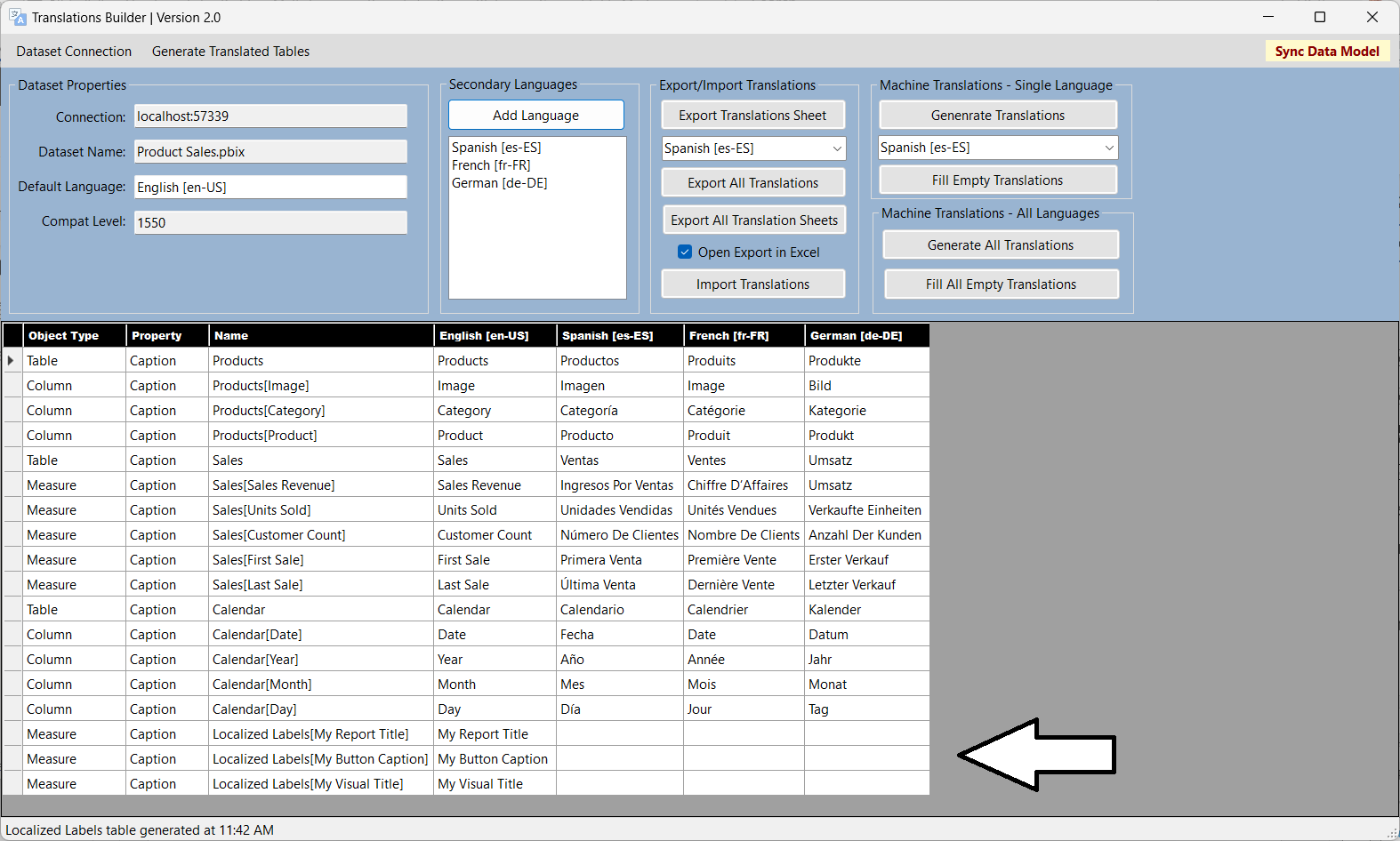
1. When you create the **Localized Labels** table, you will be prompted with the following dialog. Click **No** to continue.

Graphical user interface, application

Description automatically generated

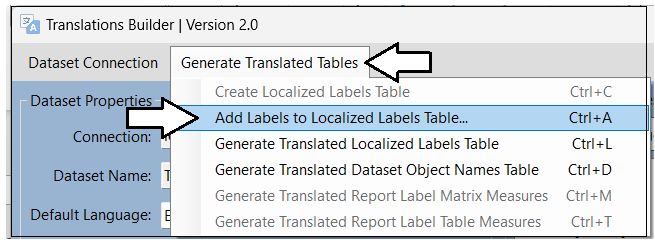
If you click **Yes**, Translations Builder will launch a browser and navigate to [this web page](https://github.com/PowerBiDevCamp/TranslationsBuilder/blob/main/Docs/Building%20Multi-language%20Reports%20in%20Power%20BI.md#understanding-the-localized-labels-table) which explains the localized labels strategy.

1. Once Translations Builder has created the **Localized Labels** table, it will also add three sample report labels.



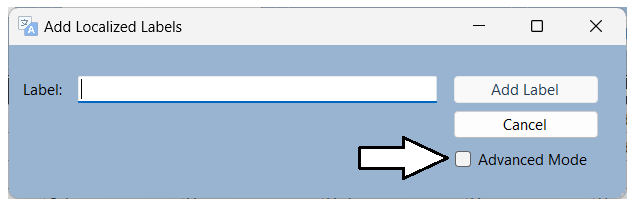
Over the next few steps, you will delete these three sample report labels and replace them by adding two of your own.

1. Drop down the **Generate Translated Tables** menu and select click **Add Labels to the Localized Labels Table**.

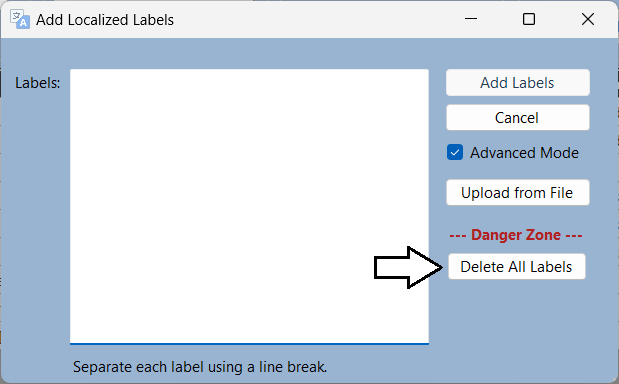


Note you can also execute the **Add Labels to the Localized Labels Table** command using the shortcut key of **Ctrl+A**.

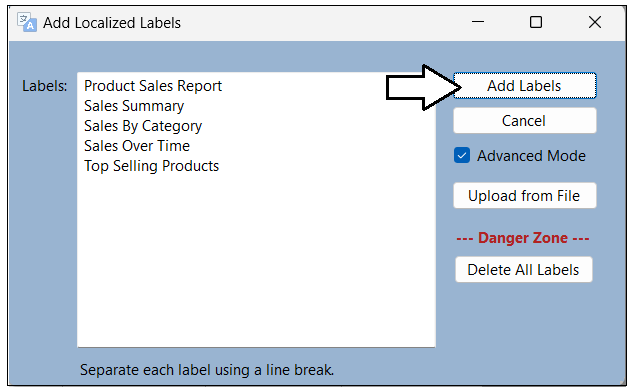
1. In the **Add Localized Labels** dialog, click the **Advanced Mode** checkbox.



1. Once the dialog is in **Advanced Mode**, click the **Delete All Labels** button to remove the sample report labels.



1. In the **Labels** textbox, type the following labels separated by line breaks.
   1. Product Sales Report
   2. Sales Summary
   3. Sales By Category
   4. Sales Over Time
   5. Top Selling Products
2. Click the **Add Labels** button to add the two new labels to your project.



1. You should now see five new rows have been added to the translations gird with the new report labels.

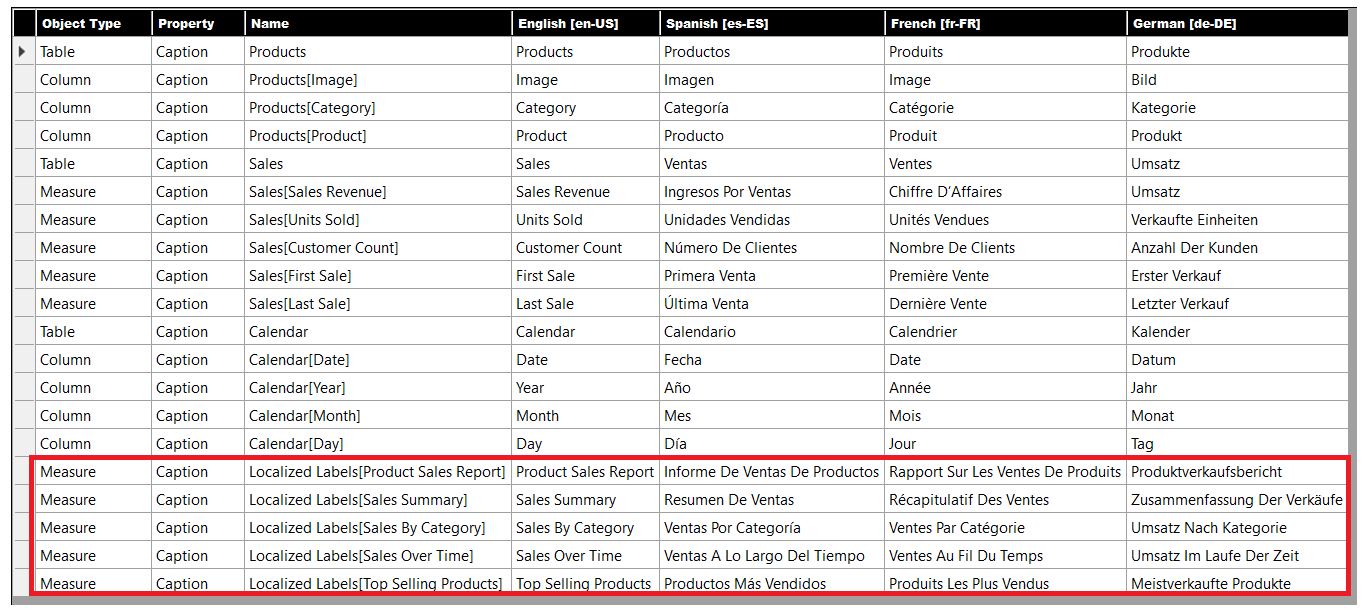


1. Click the **Fill All Empty Translations** button to create all the translations for both report labels.

Graphical user interface, application

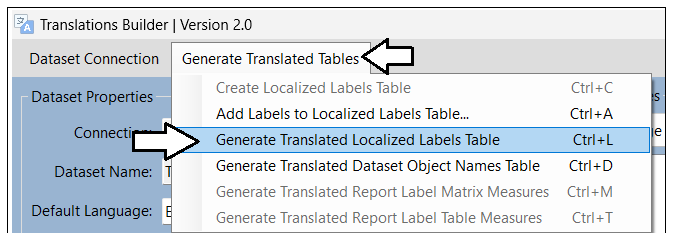
Description automatically generated

1. At this point, the translations grid should be completely filled with machine-generated translations.



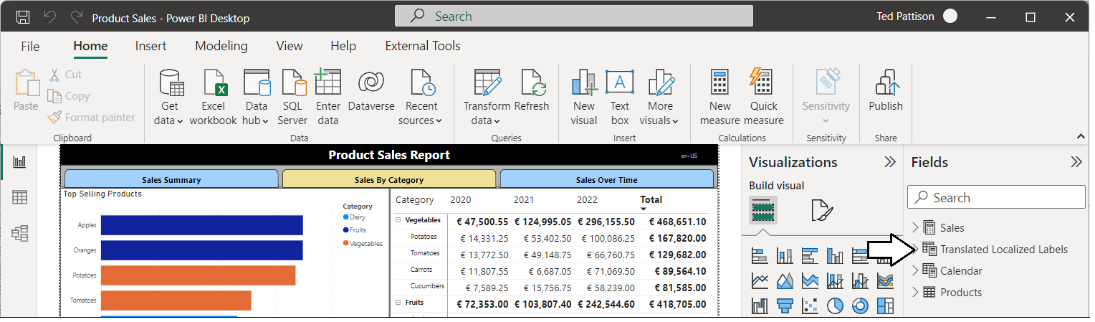
There is one critical step you must complete after modifying report labels in the **Localized Labels** table. More specifically, you must execute **Generate Translated Localized Labels Table** to create the measures that will be used to surface report labels on a report.

1. Drop down the **Generate Translated Tables** menu and select click **Generate Translated Localized Labels Table**.



Note you can also execute the **Generate Translated Localized Labels Table** command using the shortcut key of **Ctrl+L**.

1. Return to the **Product Sales.pbix** project in Power BI Desktop and navigate to **Report view**.
2. Locate the **Translated Localized Labels** table in the **Fields** list.



The measures in the **Translated Localized Labels** table are what you use to display report labels on a Power BI report.

1. Select the measure named **Product Sales Report Label** and examine the DAX expression behind this measure.

Graphical user interface

Description automatically generated

You should not edit the DAX expressions of measures in the **Translated Localized Labels** table. Any changes you make will be lost as all the measures in this table are deleted and recreated each time you execute **Generate Translated Localized Labels Table**.

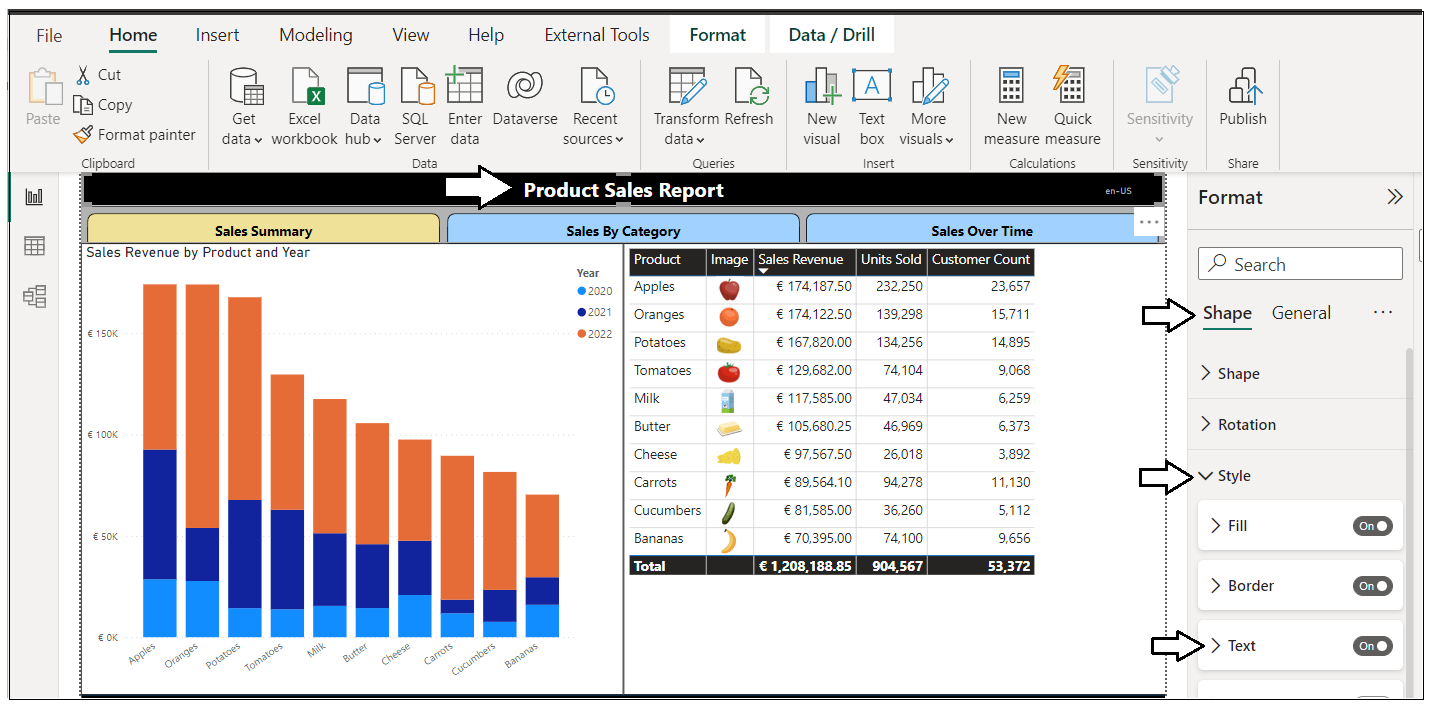
1. Examine the DAX expression behind each of the measures in the **Translated Localized Labels** table.

Now that you have created the measures for translation in the **Translated Localized Labels** table, it’s time to use them in the report.

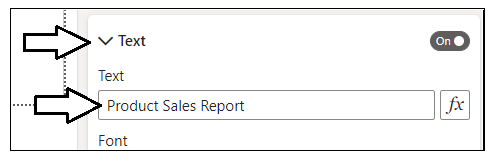
1. S



1. In the report layout, select the large yellow rectangle shape that displays the report title **Product Sales Report**.
2. With the rectangle shaped selected, move to the **Format** pane and locate the **Text** section inside the **Style** selection.

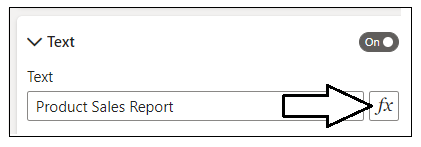


1. Expand the **Text** section to see the **Text** property is configured with the literal string value of **Product Sales Report**.

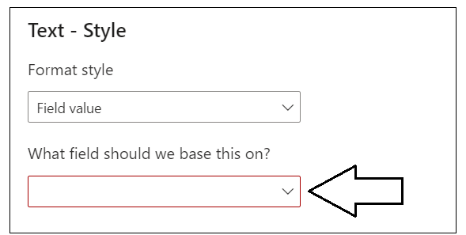


Literal string values in a report layout cannot be localized. Therefore, you will replace this literal string with a measure with translations.

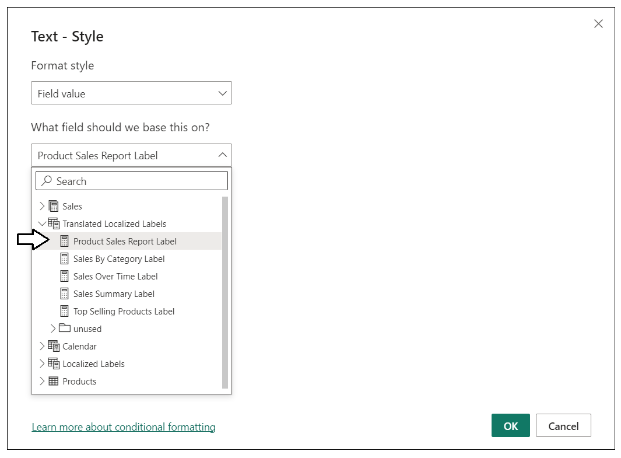
1. Click on the ***fx*** button to the right of the textbox to replace the literal string value.

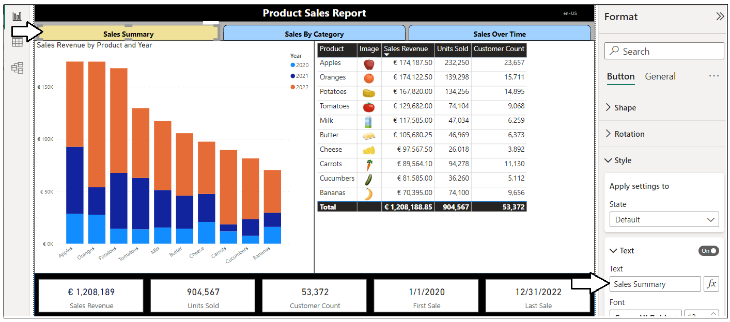


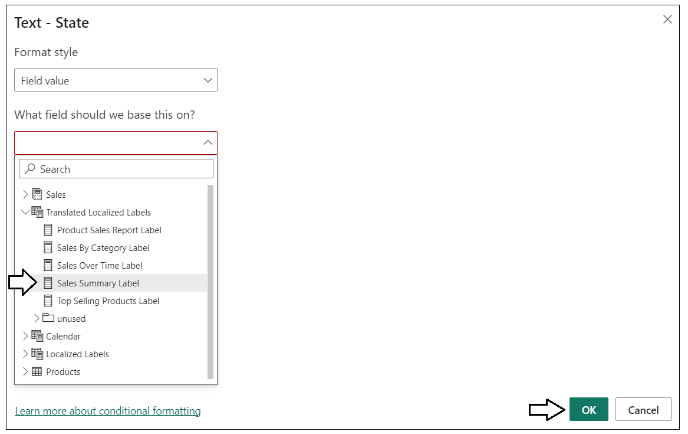
1. In the **Text – Style** dialog, select **Field value** as the **Format style**.
2. Drop down the select control with the caption of **What field should we base this on?**

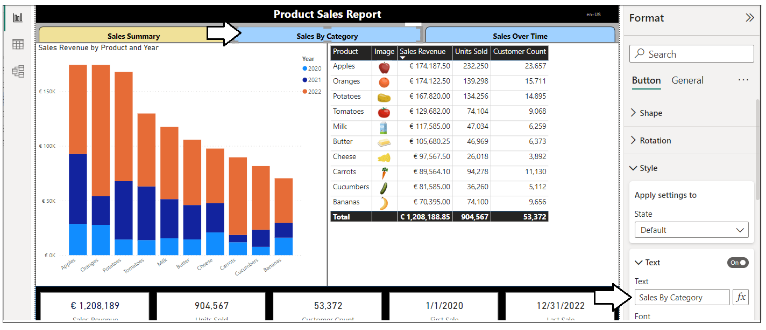


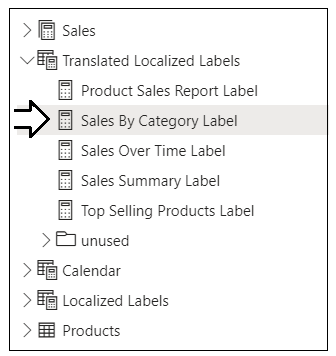
1. Select the **Product Sales Report Label** measure from the **Translated Localized Labels** table.

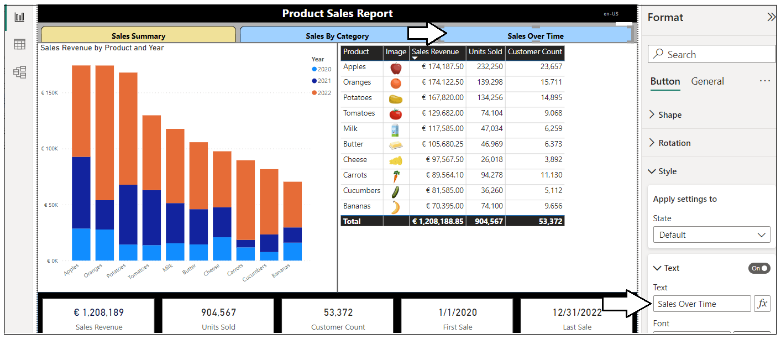


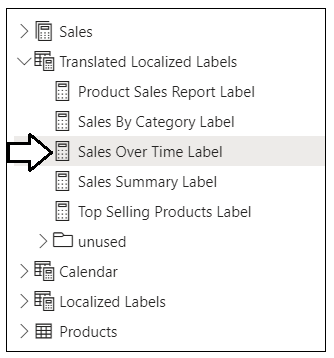


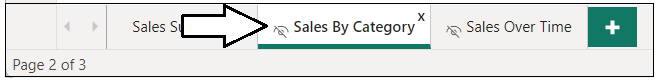






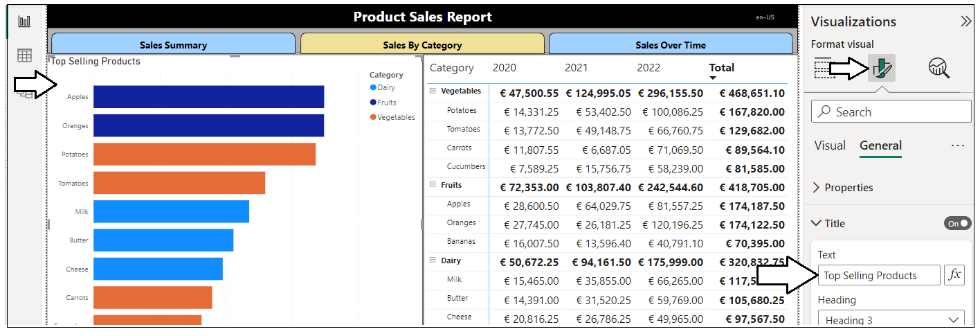


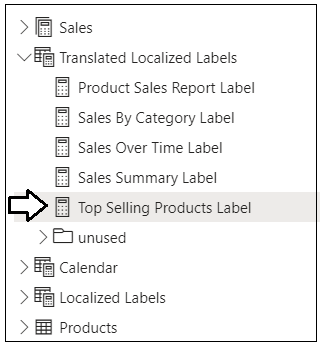




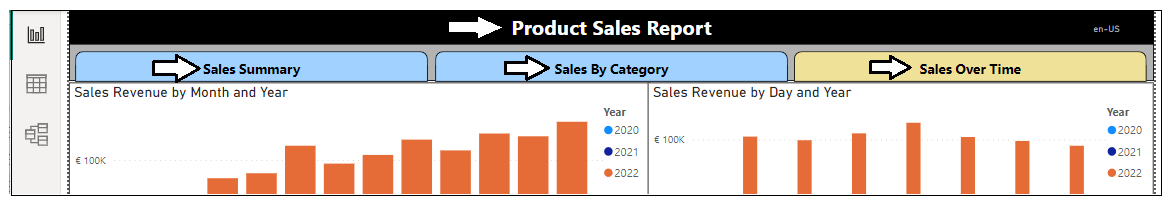


1. Select the table visual that display the product list.
2. With the table visual selected, move to the **Format** pane and click the **General** tab.
3. Locate the **Title** section and the **Text** property inside with the literal text value of **Product List by Sales Revenue**.









Graphical user interface, text, application, Word

Description automatically generated

Now, it’s time once again to test your work in the Power BI Service,

1. Save your work by clicking the **Save** button.

A screenshot of a computer

Description automatically generated

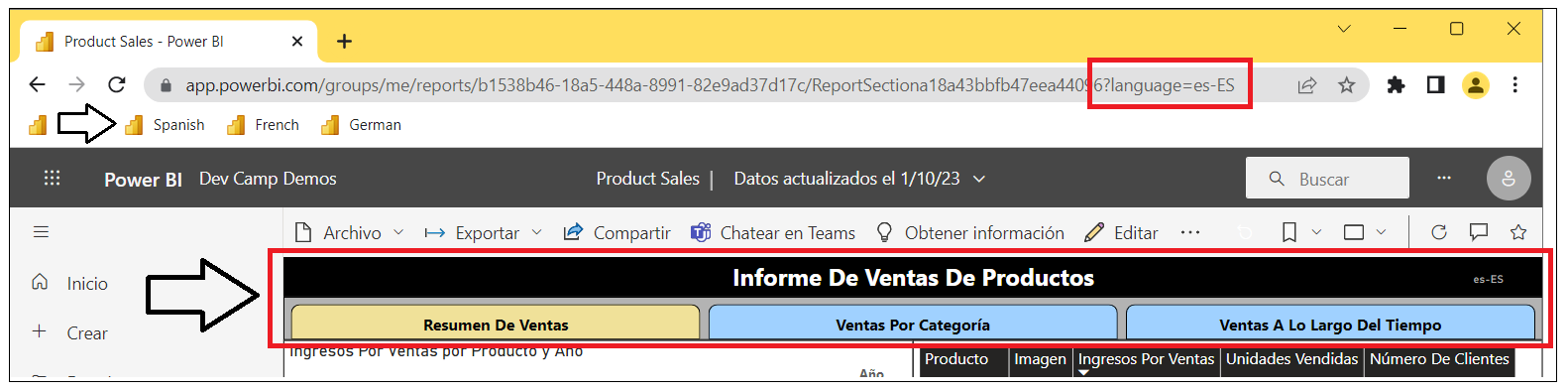
Don’t forget to save your work! Did we mention it’s easy to forget and to lose your work.

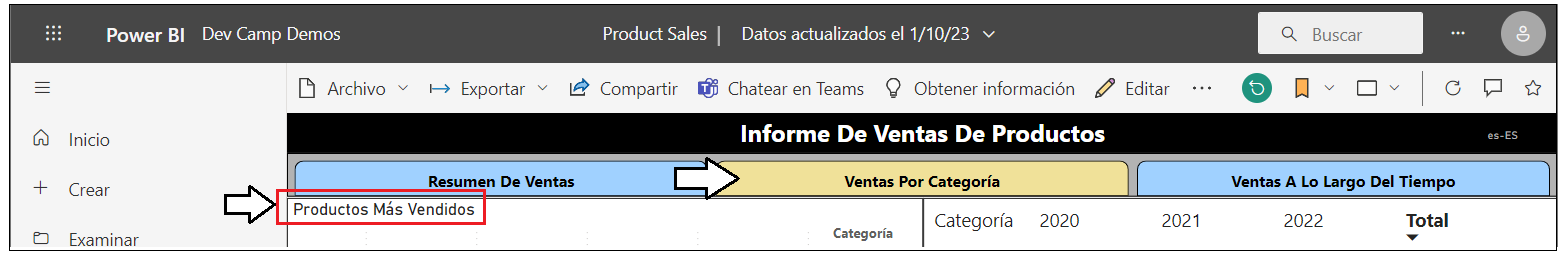
1. Publish the **Product Sales.pbix** project to push your changes to the project’s translations to the Power BI Service.

Graphical user interface, application

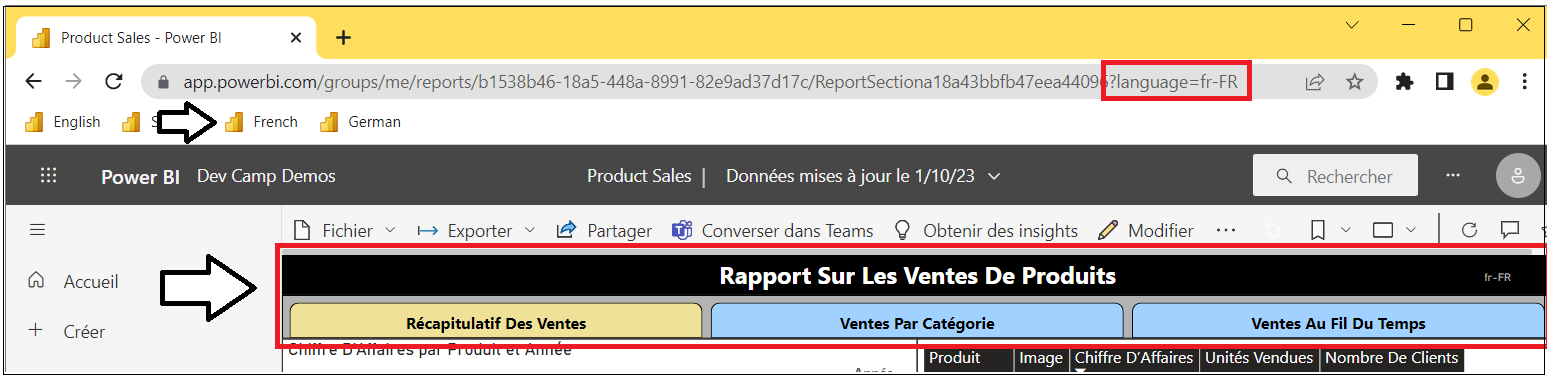
Description automatically generated

1. When prompted by the **Replace this dataset?** Dialog, click the **Replace** button to continue.
2. Once you see **Success!**, click **Open ‘Product Sales.pbx’ in Power BI** to view the report in the Power BI Service.
3. The report should load as normal showing all text in English at first.
4. Use the bookmark created earlier to load the report in Spanish. Verify the report labels show Spanish translations.





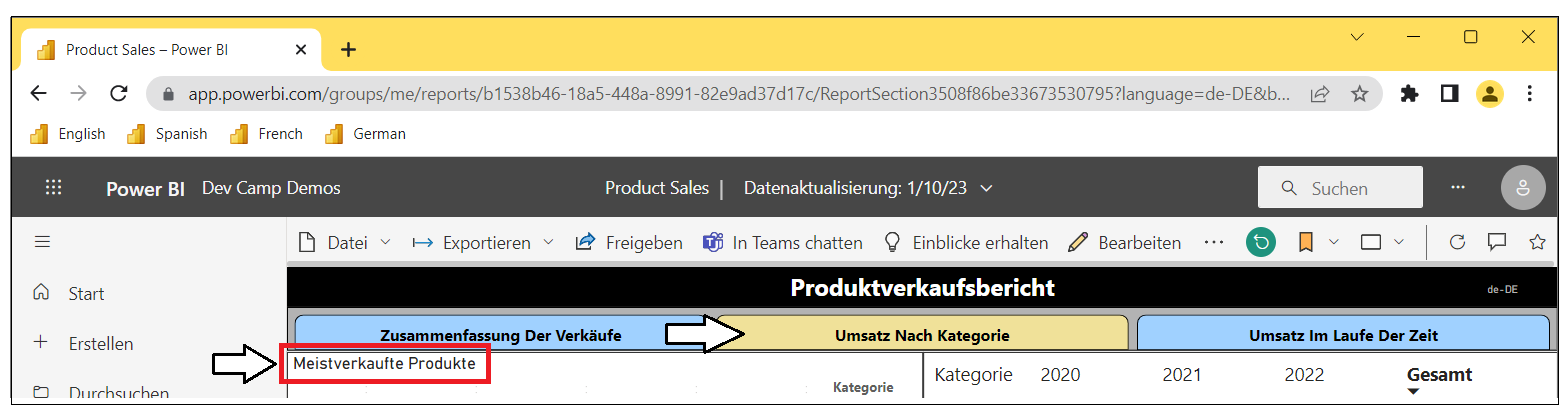
1. Use the bookmark created earlier to load the report in French. Verify the report labels show French translations.





1. Use the bookmark created earlier to load the report in German. Verify the report labels show German translations.





You have now implemented report label translations using the Translations Builder localized label strategy. You should be able to see that this will add a significant level of productivity to your future efforts to build Power BI reports that support multiple languages.

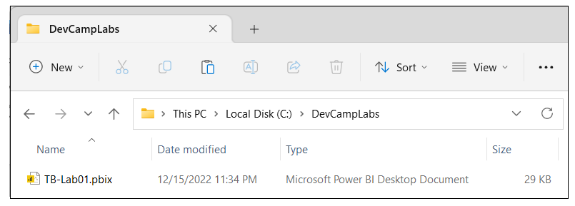
### Exercise 4: Creating a Workflow Process To Gather & Integrate Human Translations

Up to this point, you have done the work required to get the report and its underlying dataset into a structure to support translations for secondary languages. You were able to complete this work in a quick and efficient manner using Translations Builder together with machine-generated translations. However. It’s import to acknowledge that machine-generated translations alone will not be adequate for many production scenarios. You will need a way to integrate other people acting as translators into a human workflow process.

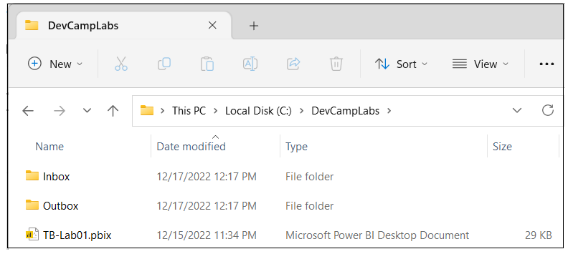
In this lab exercise, you will work with the Translations Builder features to export and import translations using a CSV file format. This will provide a quick way to generate translations sheets that can be sent to human translators. As you will see, translators can make their edits to a translation sheets using Microsoft Excel. Once you’ve received an updated translation sheet back from a translator, Translations Builder provides an import operation to integrate those updated translations back into the dataset for the current project.

**Prerequisite**: To complete this exercise, you will need Microsoft Excel installed on the same PC running Translations Builder.

1. Launch Windows Explorer and navigate to the folder where you copied the project file **Product Sales.pbix**.

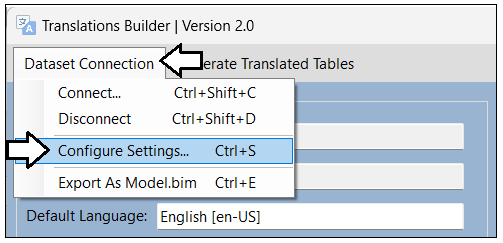


1. Create two new folders inside the lab folder named **Outbox** and **Inbox**.

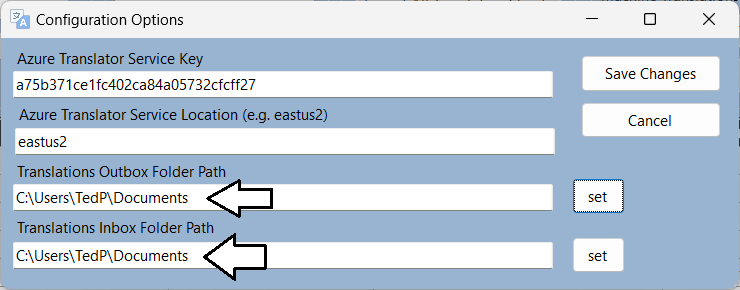


Next, you need to configure settings in Translations Builder so that these folders are used as targets for export and import operations.

1. Return to Translations Builder and drop down the **Dataset Connection** menu.
2. Click **Configure Settings…** to display the **Configuration Options** dialog.

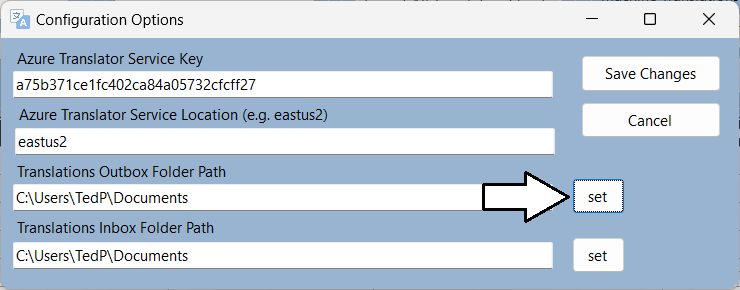


1. By default, folder paths for the **Outbox** and **Inbox** are configured to target the current user’s **Documents** folder.

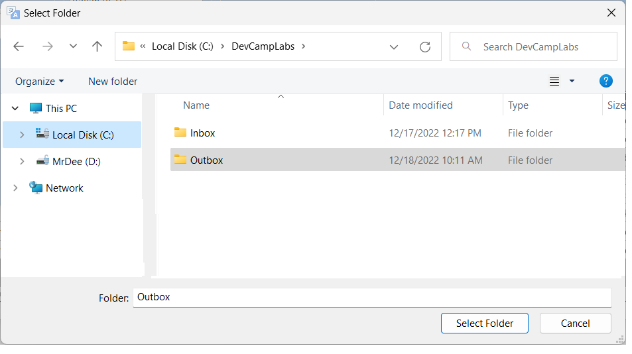


Why does **Outbox** come before **Inbox**? That’s because you generally work with the **Outbox** first when you export translation sheets that you will send to translators. Once you get updated translations sheets back from translators, you add them to the **Inbox** for import.

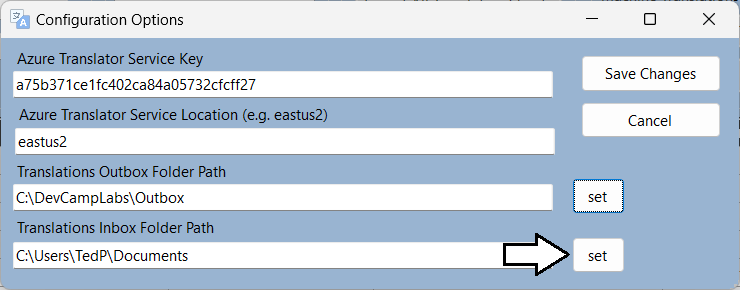
1. Click the **set** button to update the setting for **Translations Outbox Folder Path**.



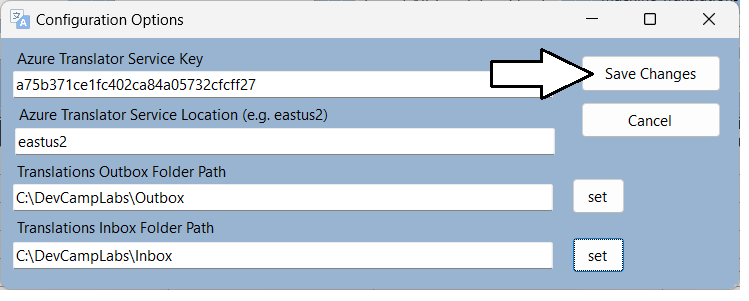
1. Select the **Outbox** folder you created earlier in this exercise.



1. Click the **set** button for **Translations Inbox Folder Path** and select the **Inbox** folder you created earlier

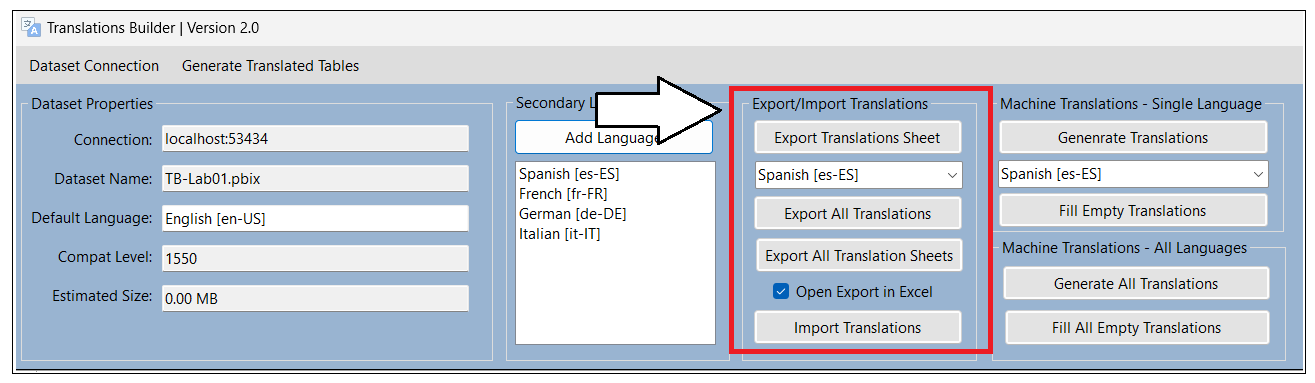


1. Click **Save Changes**.



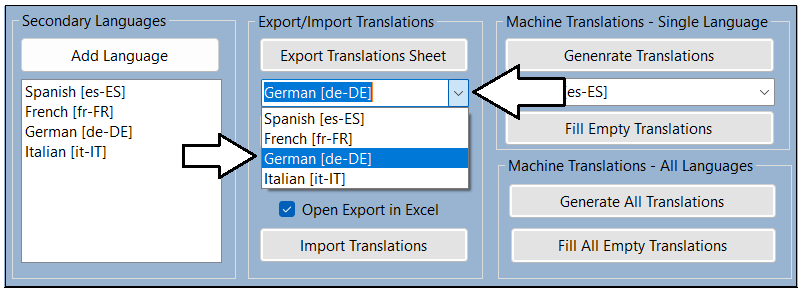
Now that you have configured the folder paths for the Outbox and Inbox, it’s time to begin exporting translation sheets.

1. Examine what’s inside the Export/Import Translations section.

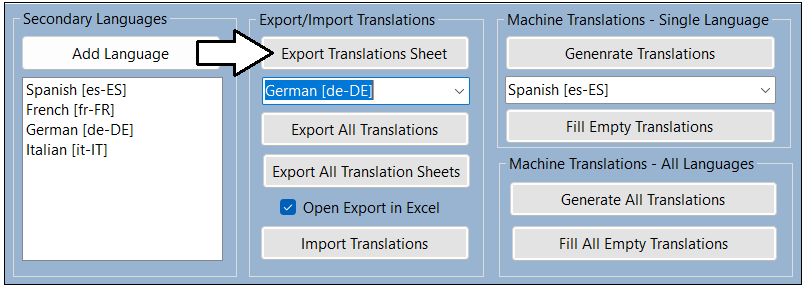


Let’s start by creating a translation sheet for a single language.

1. Drop down the selection menu under the **Export Translations Sheet** button and select **German [de-DE]**.



1. After selecting **German [de-DE]**, click the **Export Translations Sheet** button.



Translations Builder should create a translation sheet named **Product Sales-Translations-German.csv** and open this CSV file in Excel

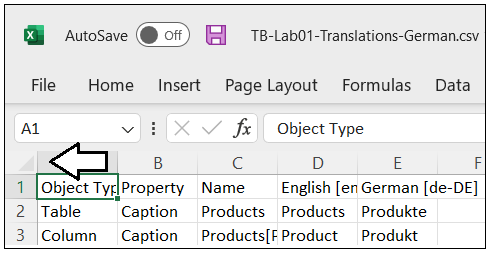
1. Examine the contents of **Product Sales-Translations-German.csv**.

Graphical user interface, application, table, Excel

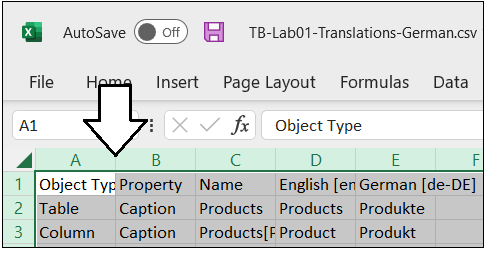
Description automatically generated

Over the next two steps you will use a trick in Excel to widen the columns so ou can see all their contents.

1. Click on the top left corner where the row headers and the column headers meet. This should select all columns and rows.



1. Double-click on the column heading divider between the column headers showing **A** and **B**.

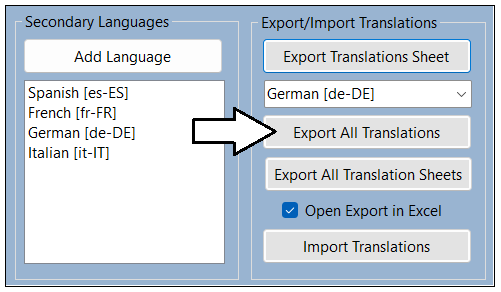


1. You should now be able to see all the text from each column.



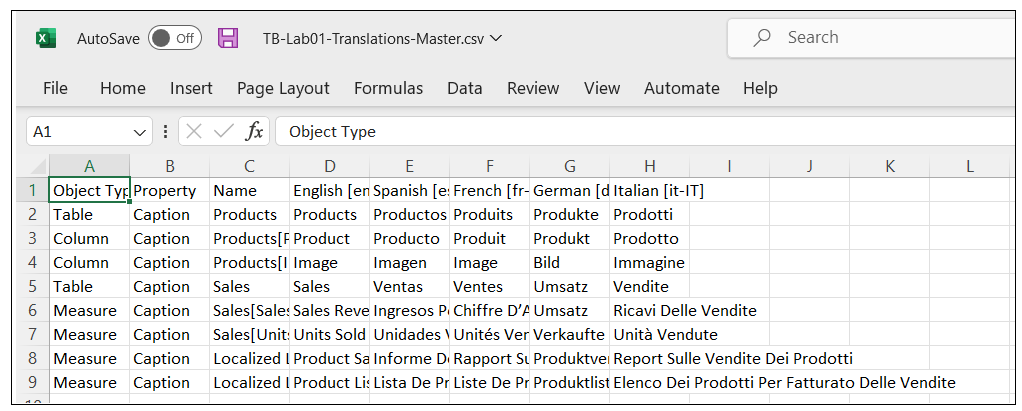
This translation sheet is what you will send to translators. They can then use Excel to review the machine translations and make changes wherever they are required.

1. Close **Product Sales-Translations-German.csv** and return to Translations Builder.
2. Click the **Export All Translations** button to export a master translation sheet with the translations for all languages.

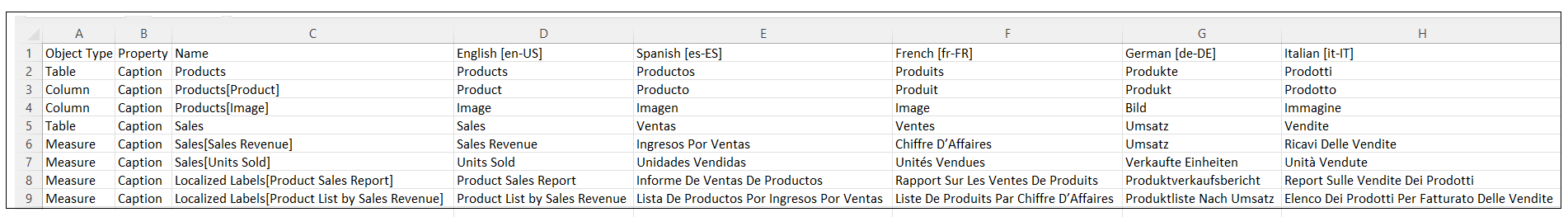


Translations Builder creates a translation sheet named **Product Sales-Translations-Master.csv** and opens this CSV file in Excel

1. When **Product Sales-Translations-Master.csv** open in Microsoft Excel, you cannot see the contents of all columns at first.

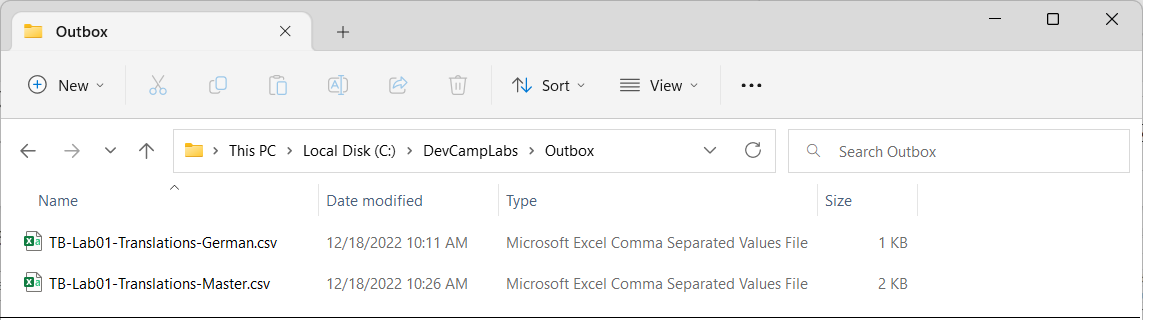


1. Use the Excel trick you learned earlier to expand all columns so you can see the entire contents of all cells.

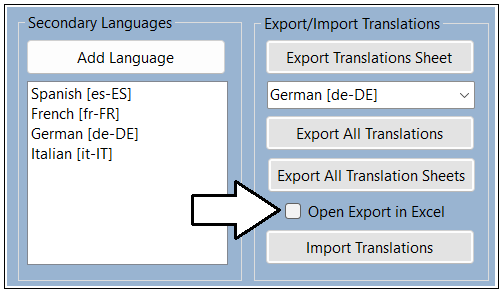


Now that you have learned to export translations sheets, it’s time to examine how to manage translation sheet files.

1. In Windows Explorer, navigate to the **Outbox** folder. You should see the two files you generated using export operations.

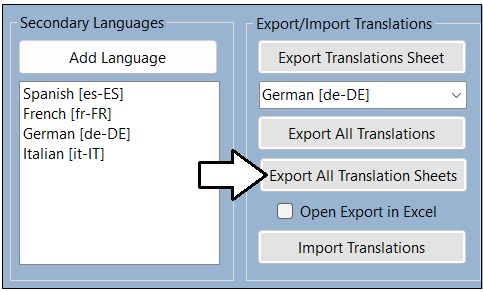


1. Return to Translations Builder and uncheck the checkbox with the caption **Open Export in Excel**.



All three export commands use this checkbox to decide whether to open a translation sheet in Excel after it’s generated. In some cases, it’s handy to have the translation sheet open in Excel. In other scenarios like the one ahead, it’s unnecessary and distracting.

1. Click the **Export All Translation Sheets** button.



The **Export All Translation Sheets** command generates the complete set of translation sheets to be sent to translators.

1. Return to the **Outbox** folder in Windows Explorer.
2. You should see that a sperate translation sheet has been generated for each secondary language.

Graphical user interface, text, application, email

Description automatically generated

Now that you have experienced how to export translation sheets, you will now learn how to import translation sheets.

1. In the **Outbox** folder in Windows Explorer, select **Product Sales-Translations-Master.csv** and **Product Sales-Translations-Spanish.csv**.

Graphical user interface, text, application, email

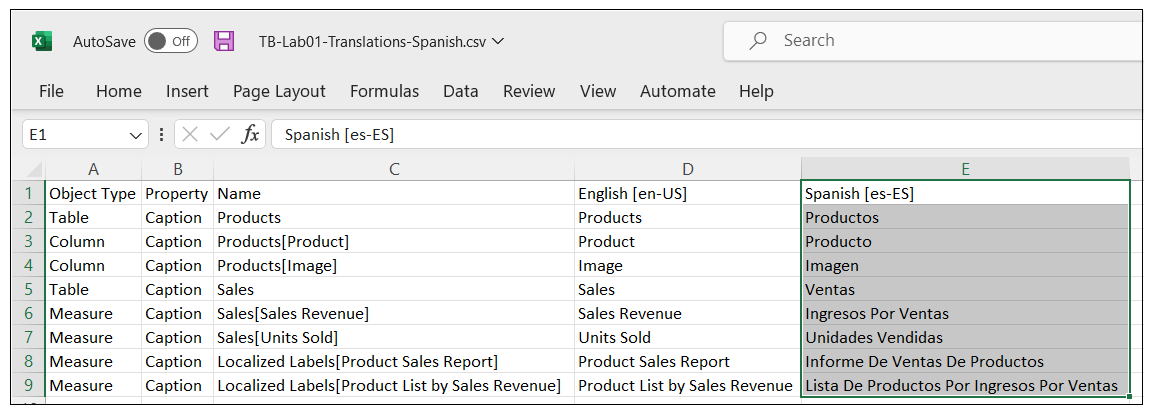
Description automatically generated

1. Copy the two selected translation sheet files to the Windows clipboard.
2. Navigate from the **Outbox** folder to the **Inbox** folder.
3. Paste the two translation sheet files from the Windows clipboard into the **Inbox** folder.

Graphical user interface, text, application, email

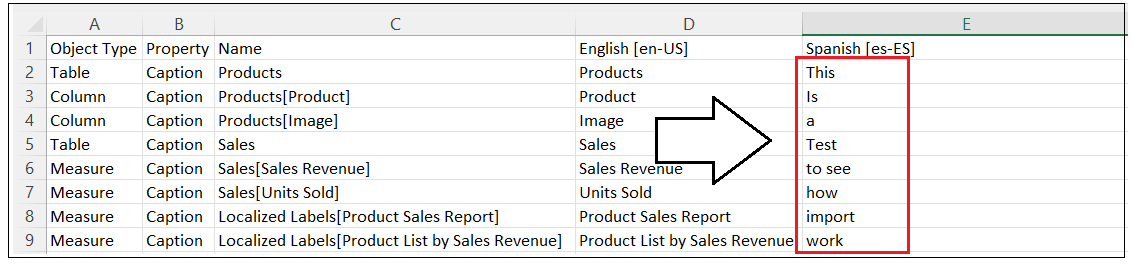
Description automatically generated

1. Open the translation sheet named **Inbox\Product Sales-Translations-Spanish.csv** in Microsoft Excel.



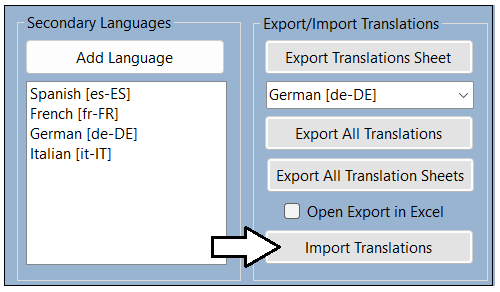
The job of the translator is to review all translations in the fifth column and to make updates where appropriate. From the perspective of the translator, the top row with column headers and the first four columns should be treated as read-only values.

1. Enter new values for each of the Spanish translations in the fifth column.



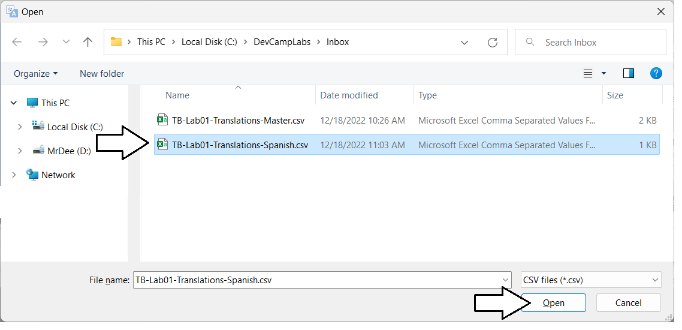
Don’t worry whether the values you are valid translations. You just need to add any text so you can test the import process.

1. Save your changes to **Product Sales-Translations-Spanish.csv** and then close the file in Microsoft Excel.
2. Return to Translations Builder and click the **Import Translations** button.

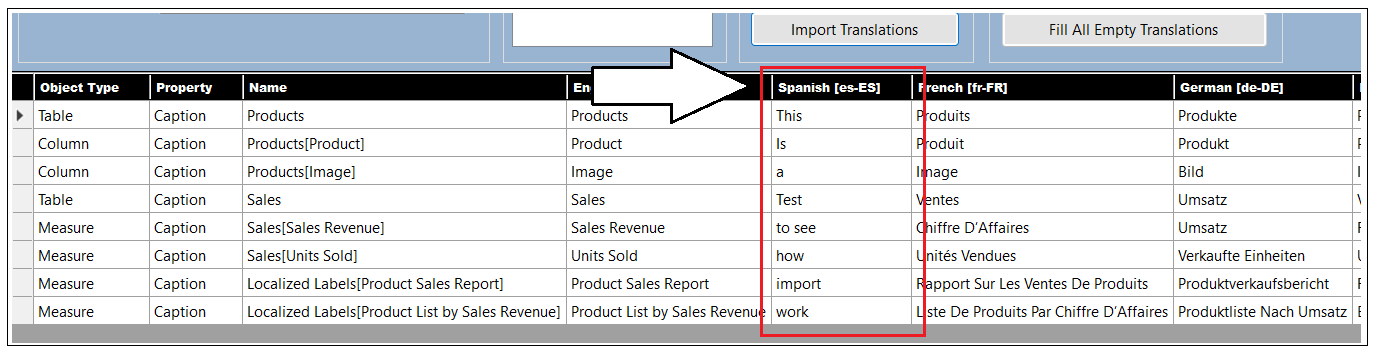


Remember to close translation sheet files in Microsoft Excel before importing them with Translations Builder to prevent errors.

1. In the **Open** file dialog, select **Product Sales-Translations-Spanish.csv** and click **Open**.

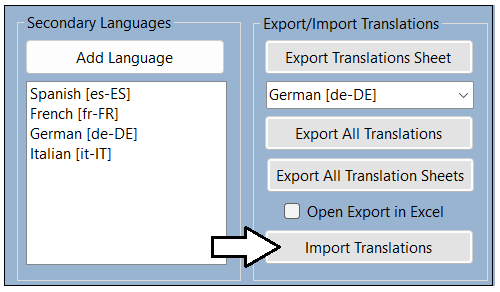


1. You should see that your updates to the Spanish translation sheet now appear in the translation grid.

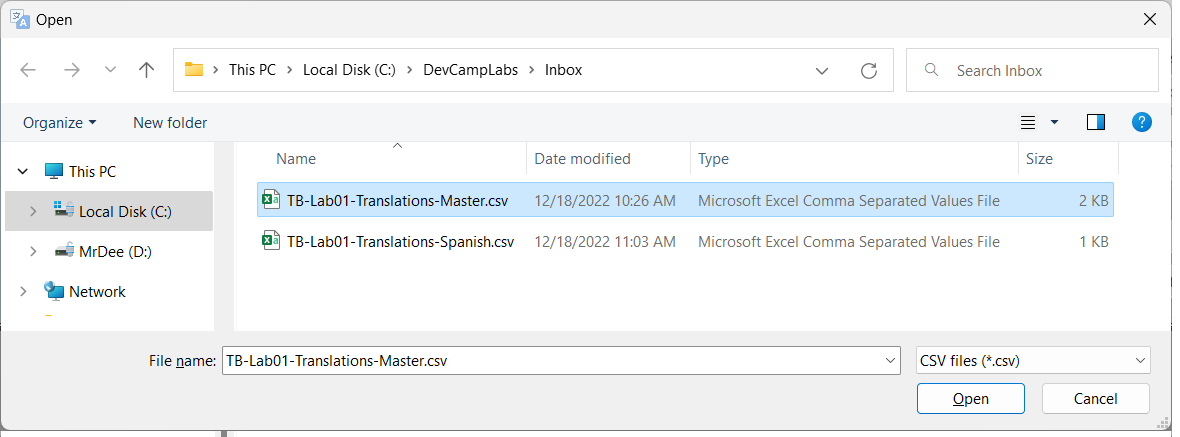


Now that you have seen how to how to import translations from an updated translations sheet with a single language, it’s time to move ahead and import translations from all languages at once by importing the master translation sheet.

1. click the **Import Translations** button.



1. In the **Open** file dialog, select **Product Sales-Translations-Master.csv** and click **Open**.



1. You should see that the original, machine-generated Spanish translations now appear in the translation grid.



You can see that the master translation sheet can also serve as a great way to backup and restore your translations work. To make this point, you are now going to delete the column for French. Deleting a column like this will delete all translations for that language. As you will see, Translations Builder will automatically add the column back if it finds the column when importing a translation sheet.

1. Right-click on the **French [fr-FR]** column header and click **Delete Secondary Language**.



1. When prompted by the **Confirm Delete Secondary Language Operation** dialog, click **OK** to continue.

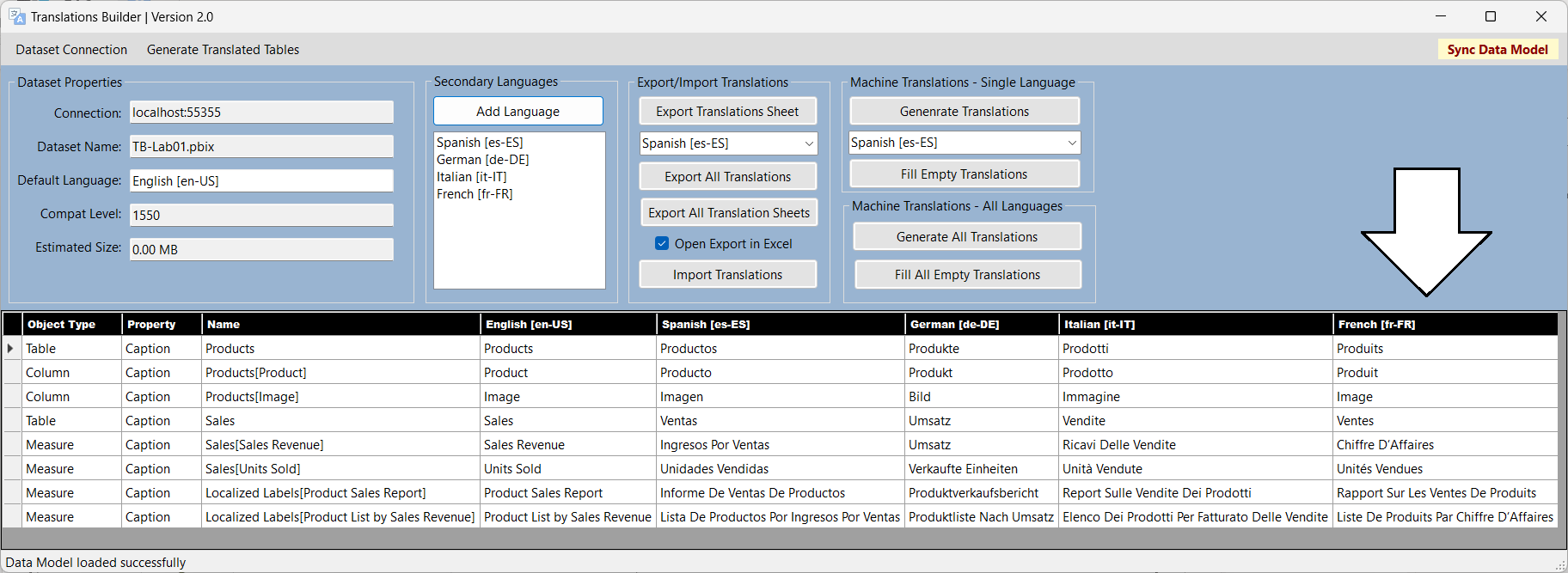
Graphical user interface, application

Description automatically generated

1. You should see that the column for French has been removed from the translations grid.



1. click the **Import Translations** button.
2. In the **Open** file dialog, select **Product Sales-Translations-Master.csv** and click **Open**.
3. After the import operation competes, the **French [fr-FR]** column should reappear as the last column on the right.



### Exercise 5: Implementing Data Translations using Field Parameters

xxxx.

xxx

### Exercise 6: Implementing Data Translations for a Calendar Table

xxxx.

Congratulations. You have now completed this lab.