HITACHI
Inspire the Next

Lab Guide
Pentaho Data Integration (PDI) Fundamentals
PDI1000L / PDI1000S

HITACHI
Inspire the Next

Change log (if you want to use it):

Date	Courseware Version	Microcode Version	Notes
Oct-2020	3.1	8.x	New style, minor corrections
Nov-2020	3.2	9.1.0.1-360	Updates for change to Linux & Pentaho 9.1.01
Dec-2020	3.3	9.1.0.1-360	Corrected screenshot
Jan-2021	3.4	9.1.0.2-393	Corrections made for Windows/Linux and newer version UI differences
Feb-2022	3.5	9.1.0.2-393	Updated cover page to reflect this guide is used by ILT & self-paced

Contents

Guided Demo 1: Launching and Customizing PDI

In this guided demonstration, we launch Spoon, Pentaho Data Integration (PDI)'s graphical interface and customize some of its options and default behavior.

Objectives and Activities

After completing this guided demonstration, you will be able to:

- Launch Spoon
- Turn the Welcome Screen on and off
- Open Spoon's Options dialog
- Describe the common options and Look & Feel settings
- Change the grid settings

Launch and Customize PDI

To launch and customize PDI:

1. From the desktop, click the **Data Integration (Spoon) icon**. 
2. Scroll through the **Welcome Screen** to familiarize yourself with its contents.
3. To close the **Welcome Screen**, on the **Welcome!** tab, click **X**.
4. To view the **Kettle Options**, from the menu, select **Tools → Options**.
5. Review the options on the **General** tab, and then click the **Look & Feel** tab.
6. To modify the grid settings, on the **Look & Feel** tab:
7. Change the **Canvas Grid Size** to `32`.
8. Click to select the **Show Canvas Grid** checkbox.
9. Click **OK**.
10. To close the Info dialog, click **OK**.

Not all options require a restart for the changes to take effect.

11. Keep Spoon open for the next demonstration.

End of Guided Demo 1

Guided Demo 2: Creating a "Hello World!" Transformation

In this guided demonstration, we use Spoon to create a new transformation with rows containing "Hello World!" and a constant value. Although this is probably not a task you will be asked to do in the real world, the concepts learned in this guided demonstration help to build the foundation necessary for creating any transformation.

Objectives and Activities

After completing this guided demonstration, you will be able to:

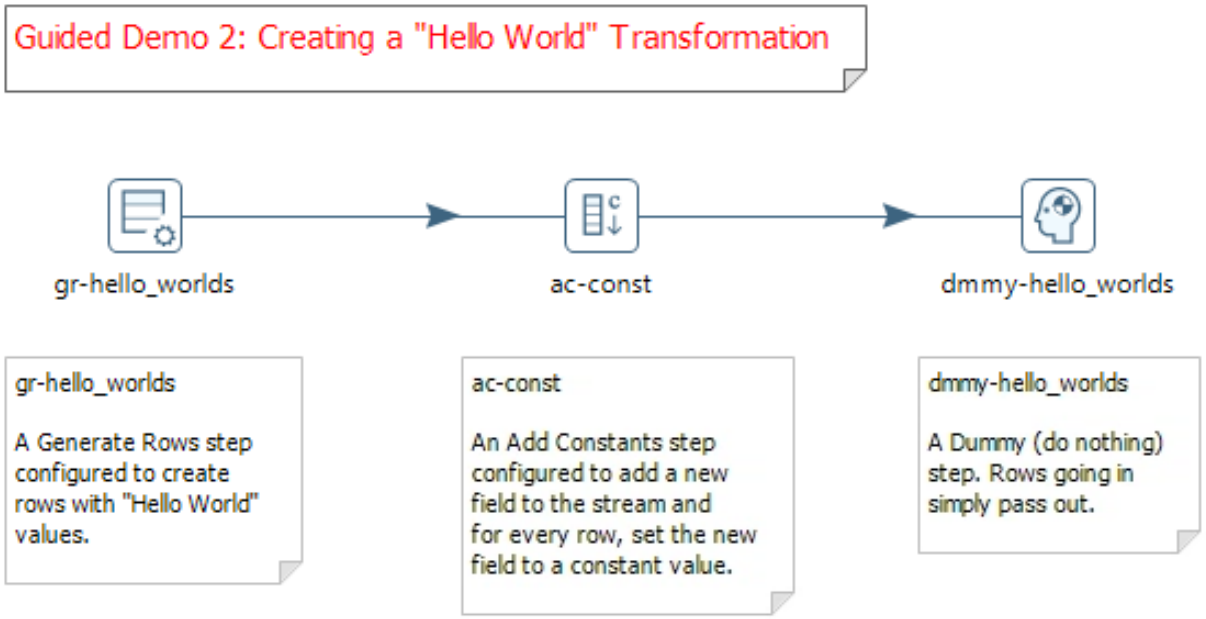
- Create a new transformation
- Add steps using best practice naming standards
- Create and split hops
- Configure and preview the **Generate rows** step
- Add a note to a transformation
- Use the **Add constants** step to add a new field to the stream and for every row, set it to a constant value
- Understand how fields get added and propagated to the stream
- Run the transformation and examine each step's fields and data

Steps Used

This guided demonstration uses the following steps:

- **Generate rows**
- **Add constants**
- **Dummy (do nothing)**

Transformation



Create Transformation and Add a Step

In this section, we create a new transformation and then add and configure the **Generate rows** step.

1. To create a new transformation, from the menu, select **File** → **New** → **Transformation**.

   Alternatively, on the toolbar, click the **New file** button, and then select **Transformation**.

2. To add a **Generate rows** step to the transformation:

   a. Click to view the **Design** tab.

   b. Expand the **Input** category.

   c. Drag the **Generate rows** step from the **Design** tab to the canvas.

3. To configure the **Generate rows** step, on the canvas, double-click the **Generate rows** step.

You might find it helpful to enlarge the step dialog windows.

4. Enter values in the properties of the **Generate rows** dialog as follows:

Property Name	Value
Step name	`gr-hello_worlds`
Limit	`15`

5. Complete the **Fields** grid of the **Generate rows** dialog as follows:

Column Name	Value
Name	`greeting_message`
Type	`String`
Value	`Hello World!`

6. To preview this step to verify it generates the data we expect:

    a. Click the **Preview** button.

    b. In the **Enter preview size** dialog, click **OK**.

    c. Verify that 15 rows of data with `Hello, World!` are displayed.

    d. In the **Examine preview data** dialog, click **Close**.

Previewing data and testing steps along the way can help to minimize errors and troubleshooting time later in the transformation creation process.

7. To close the **Generate rows** configuration dialog, click **OK**.

Adding a Dummy (Do Nothing) Step and Hop

Next, we add a **Dummy (do nothing)** step and create a hop from the **Generate rows** step. Then, add a note to the transformation. Notes help viewers understand how the transformation works.

1. To add a **Dummy (do nothing)** step to the transformation, on the **Design** tab:

a. Click to expand the **Flow** category.

b. Drag the **Dummy (do nothing)** step from the **Design** tab to the canvas. Drop it to the right of the **Generate rows** step.

2. To create a hop from the **gr-hello_worlds** step to the **Dummy (do nothing)** step, on the canvas:

   a. Hover the mouse pointer over the **Generate rows** step.

   b. Click the **output connector** icon.

   c. Click the **Dummy (do nothing)** step.

Other methods for creating hops include the following:

- Press and hold the shift key, then click and hold the source step, then point to the destination step, and then release.
- Using the middle mouse button, click and hold the first step, then point to the destination step, and then release.

3. To name the **Dummy (do nothing)** step, double-click the step, resize the dialog, and set its step name property to: `dmmy-hello_worlds`. Resizing the **Dummy (do nothing)** step's dialog is only necessary the first time you open this type of step. Subsequent openings of this step type will retain the last dialog size.

4. To add a note to the transformation, on the canvas, right-click near the top-right corner, and then select **New Note**.

5. In the **Notes** dialog:

   a. Click in the **Note** section.

   b. Type in any note that you'd like to add. For example: *This transformation creates 15 rows of Hello World*.

6. Click **OK**.

Adding a Step by Splitting a Hop

In this section, we learn how to split a hop. Splitting a hop is an easy way to insert a step between two other

steps that are already connected with a hop. The step that we'll insert is the **Add constants** step. We'll configure the step to add a new field to the stream and for every row coming into the step, set the new field's value to a constant. Our purpose for adding this step is to become acquainted with how new fields are added to the stream and how they propagate to other downstream steps.

1. To split the hop with the **Add constants** step:

    a. Use the **Design** tab's search box to find the **Add constants** step, and then drag it to an empty area of the canvas.

    b. On the canvas, drag the **Add constants** step up to the hop and touch the mouse pointer to the hop.

    c. When the hop turns bold, let go of the mouse button.

    d. At the **Split hop** dialog, click **Yes**.

*You must drop a step onto the canvas before you attempt to split a hop. You cannot drag a step from the **Design** tab's list of steps onto the hop.*

2. To configure the **Add constants** step, on the canvas, double-click the **Add constants** step.

3. To name the step, in the **Step name** property, type: `ac-const`

4. Define one row in the **Fields** grid as shown (leave any property names that are not in the table as their default values):

Property Name	Value
Name	`const`
Type	`Number`
Format	`0.00`
Value	`120`

5. To close the **Add constants** step dialog, click **OK**.

Saving and Running the Transformation

In this section of the guided demonstration, we save and then run the transformation.

1. To set the transformation properties:

a. On the canvas, double-click an empty area.

b. In the **Transformation name** property, type: `create_hello_worlds`

c. Optionally, provide a description and/or extended description.

d. Click **OK**.

2. To save the transformation:

a. From the menu, select **File → Save**, or on the toolbar, click the **Save** button.

b. Navigate to the `/home/pentaho/course_files/pdi1000l/my_work` folder.

c. In the **Name** property, type: `gd2_hello_world` (The filename for the transformation does not have to be the same as the transformation name.)

3. Click **OK**.

4. Notice the tab at the top of the canvas now reads **create_hello_worlds**.

5. To run the transformation:

a. In the subtoolbar, click the **Run** button.

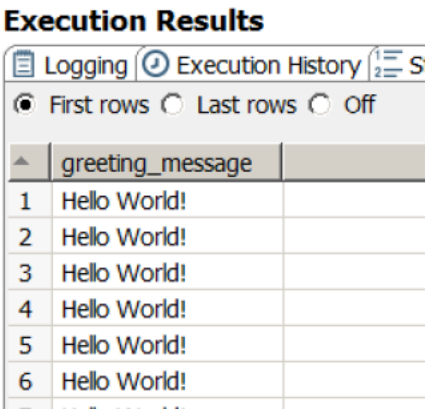
b. In the **Run Options** dialog, click **Run**.

Preview Each Step to Examine Field Layout and Steam

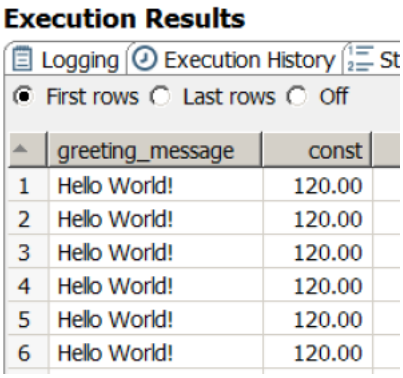
Understanding how fields are added to the stream and how they propagate through a transformation is vital to using PDI successfully. In this section, we'll use the **Preview** data tab in the **Execution Results** panel to examine the field layout and stream content for each step.

1. Click the **gr-hello_worlds** step to select it.

2. In the bottom **Execution Results** panel, click the **Preview data** tab. The stream field(s) and data will be displayed for the selected step. The **gr-hello_worlds** step's outgoing stream contains a single field, `greeting_message`, and 15 rows of data.



3. To see the outgoing stream for the next step, on the canvas, click the **ac-const** step. The **Preview data** tab's grid will update. This step's outgoing stream contains the field and data from the previous step plus the new field named **const** and its value for each row.



4. To see the outgoing stream for the final step, on the canvas, click the **dmmy-hello_worlds** step. This step's outgoing stream contains the field and data from the previous step. The **Dummy (do nothing)** step has no logic. Therefore, it does not change the fields or data that come into the step, so its outgoing stream is identical to the previous step.

*We'll discuss uses for the **Dummy (do nothing)** step later in this course.*

5. Keep the transformation open for the next guided demonstration.

Solution Details

The solution to this guided demonstration can be found at:

`/home/pentaho/course_files/pdi10001/solutions/guided_demos`

File:

`gd2_hello_world.ktr`

End of Guided Demo 2

Guided Demo 3: Viewing Errors

This guided demonstration introduces finding and viewing errors. We create an error condition, run the transformation, and view the log details. Then we correct the error and re-run the transformation.

Objectives and Activities

After completing this guided demonstration, you will be able to:

- Determine if there are errors
- View the log details
- Display only the error lines

Prerequisites

This guided demonstration uses the `gd2_hello_world` transformation created in Guided Demo 2. If you did not complete Guided Demo 2, you must have access to the course solution files.

Steps Used

This guided demonstration uses the following steps:

- **Generate rows**
- **Add constants**
- **Dummy (do nothing)**

Create an Error Condition

In this section, we purposely modify the Hello World transformation to run with an error.

1. If it is not already open, open the `gd2_hello_world` transformation.
2. Double-click the **gr-hello_worlds** step on the canvas to open it.
3. To create an error, in the **Fields** grid, change the **Type** to **Integer**, and then click **OK**.
4. Save the transformation under a different name:
    a. In the toolbar, click the **Save file with a different name (Save As)** button.

b. Type the new filename: `gd3_hello_world_with_error`

c. Click **OK**.

5. To run the transformation:

   a. Click the **Run** button on the subtoolbar.

   b. In the **Run Options** dialog, click **Run**.

6. To view the step metrics, click the Step Metrics tab.

*Notice the line highlighted in red on the **Step Metrics** tab. Also, if you click an empty area of the canvas to deselect everything, you will notice the **gr-hello_worlds** step is outlined in red.*

7. To view the **Logging** tab, in the **Execution Results**, click the **Logging** tab.

8. To view only the error lines in the log, on the **Logging** tab toolbar, click the **Show error lines** button.

9. After reviewing the error lines and noticing the specific error on lines 3 and 4, click **OK**.

10. Open the **gr-hello_worlds** step.

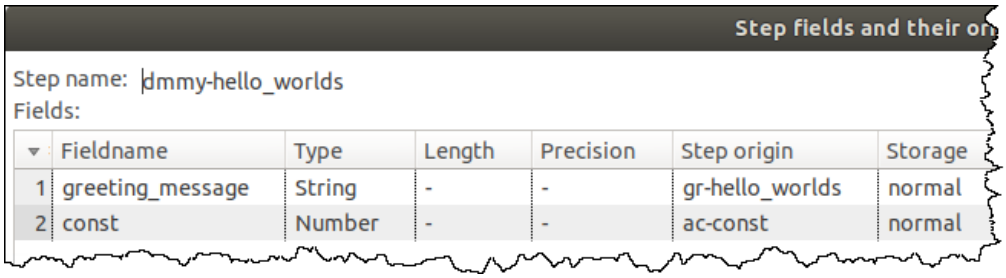
11. To correct the error, in the **Fields** grid, change the **Type** to **String**, and then click **OK**.

12. **Save** the transformation.

13. To run the transformation, on the subtoolbar, click the **Run** button and then click **Run** in **Run Options**.

14. Notice there are no errors, and all steps have a green checkmark indicating they ran successfully.

*When you troubleshoot a transformation, it can be helpful to know exactly where fields on a step originate from. To do this, right-click the step you want to learn more about and then select **Input Fields** or **Output Fields**. You will see a grid that shows the step each field originated from.*



If the transformation has a field misconfiguration such as inconsistent data type, Spoon cannot properly determine field origin and an error can result.

Solution Details

The solution to this guided demonstration can be found at:

`/home/pentaho/course_files/pdi1000l/solutions/guided_demos`

File:

`gd2_hello_world_with_error.ktr`

End of Guided Demo 3

Exercise 1: Generate Rows, Add Sequence, and Select Values

*In this exercise, you create a transformation using the **Generate rows**, **Add sequence**, and **Select values** steps. You add, configure, and preview each step to ensure the data previewed is what is expected before configuring the next step. This best practice technique helps to prevent configuration errors that would otherwise be difficult to troubleshoot if the entire transformation were created and configured prior to preview.*

Please concentrate on learning the techniques for creating the transformation and steps, rather than the logic of the steps being used. You will learn the details of these and many other steps throughout the remainder of this course.

Objectives

After completing this exercise, you will be able to:

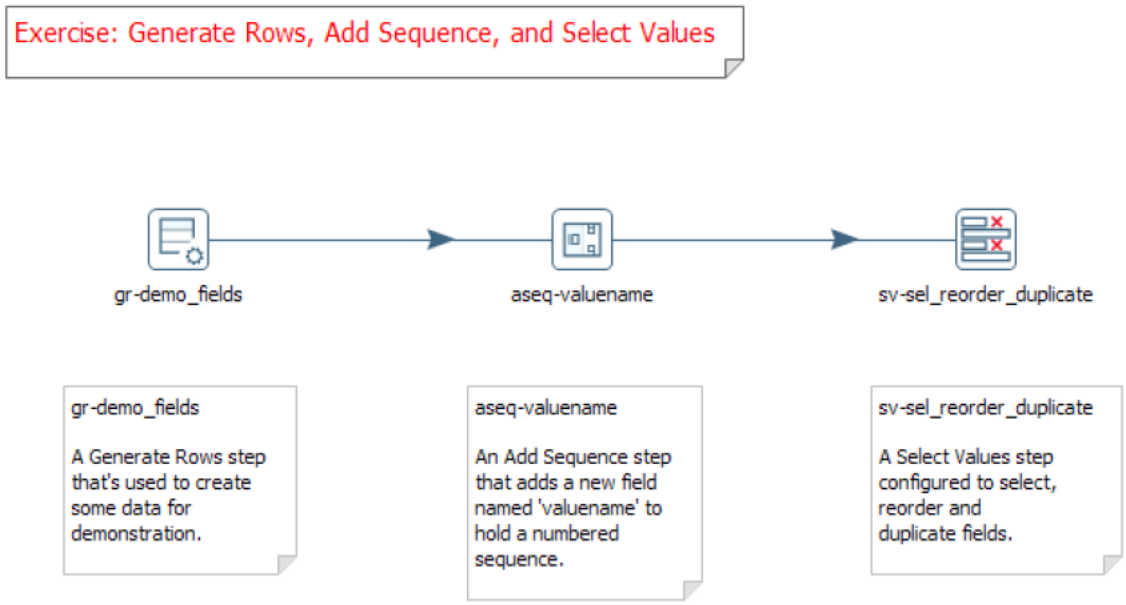
- Create a new transformation
- Add steps and hops
- Configure and preview the **Generate rows** step
- Configure and preview the **Add sequence** step
- Configure and preview the **Select values** step

Steps Used

This exercise uses the following steps:

- **Generate rows**
- **Add sequence**
- **Select values**

Transformation



Create the Transformation and Add the Generate Rows Step

In this section of the exercise, you create a new transformation, and then add, configure, and preview the **Generate rows** step to add 10 rows with three fields to the stream.

1. To create a new transformation, on the toolbar, click the **New file** button, and then select **Transformation**.

2. To add a **Generate rows** step to the transformation:

   a. Click to view the **Design** tab.

   b. Expand the **Input** category.

   c. Drag the **Generate rows** step from the **Design** tab to the canvas.

3. To configure the **Generate rows** step, on the canvas, double-click the **Generate rows** step.

4. Enter values in the properties of the **Generate rows** dialog as follows:

Property Name	Value
Step name	`gr-demo fields`
Limit	`10`

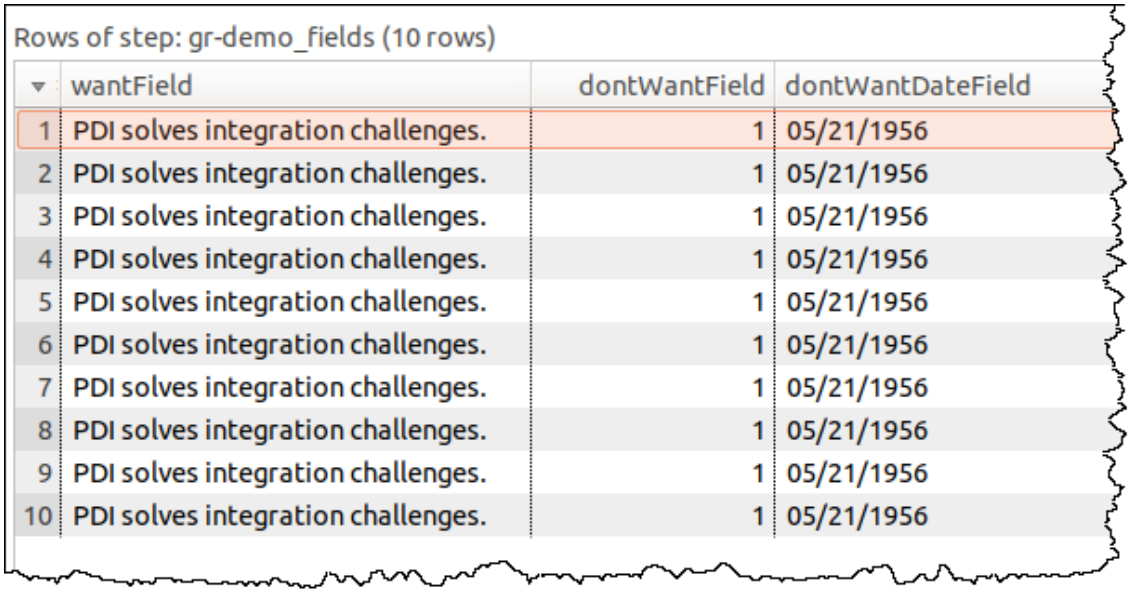
5. To configure the **Fields** grid, add three rows as shown:

Name	Type	Format	Value
wantField	`String`		`PDI solves integration challenges.`
dontWantField	`Integer`	`#`	`1`
dontWantDateField	`Date`	`MM/dd/yyyy`	`05/21/1956`

You must use the exact case and spelling as shown.

6. To preview the data and confirm it is configured properly, click **Preview**, and then in the **Enter preview size** dialog, click **OK**.

7. Verify the data generated is correct by comparing your data with the screenshot:


Rows of step: gr-demo_fields (10 rows)

	wantField	dontWantField	dontWantDateField
1	PDI solves integration challenges.	1	05/21/1956
2	PDI solves integration challenges.	1	05/21/1956
3	PDI solves integration challenges.	1	05/21/1956
4	PDI solves integration challenges.	1	05/21/1956
5	PDI solves integration challenges.	1	05/21/1956
6	PDI solves integration challenges.	1	05/21/1956
7	PDI solves integration challenges.	1	05/21/1956
8	PDI solves integration challenges.	1	05/21/1956
9	PDI solves integration challenges.	1	05/21/1956
10	PDI solves integration challenges.	1	05/21/1956

8. To close the **Examine preview data** dialog, click **Close**.

9. To close the **Generate rows** dialog, click **OK**.

10. To save the transformation, on the toolbar, click the **Save** button.

11. Navigate to the `/home/pentaho/course_files/pdi10001/my_work` folder, then in the **File name**, type ex`1_select_values` and then click **OK**.

Add and Configure the Add Sequence Step

In this section of the exercise, you add an **Add sequence** step to the transformation, and then add a hop from the **gr-demo_fields** step. The **Add sequence** step will add a field to the stream with a sequence number for each row. You then preview the transformation.

1. To add an **Add sequence** step to the transformation, on the **Design** tab:

    a. In the **Steps search** field, type: `add seq`

    b. Drag the **Add sequence** step from the Design tab to the canvas and drop it to the right of the **gr-demo_fields** step.

2. To create a hop from the **gr-demo_fields** step to the **Add sequence** step, on the canvas:

    a. Press and hold the **Shift** key.

    b. Click and hold the **gr-demo_fields** step.

    c. Point to the **Add sequence** step, and then release.

3. To rename the step using best practice naming standards, double-click the **Add sequence** step and change the **Step name** property to: `aseq-valuename`

    This step name was chosen because the step is an **Add sequence** step (**aseq**) and the step will add a new field named **valuename** to the stream.

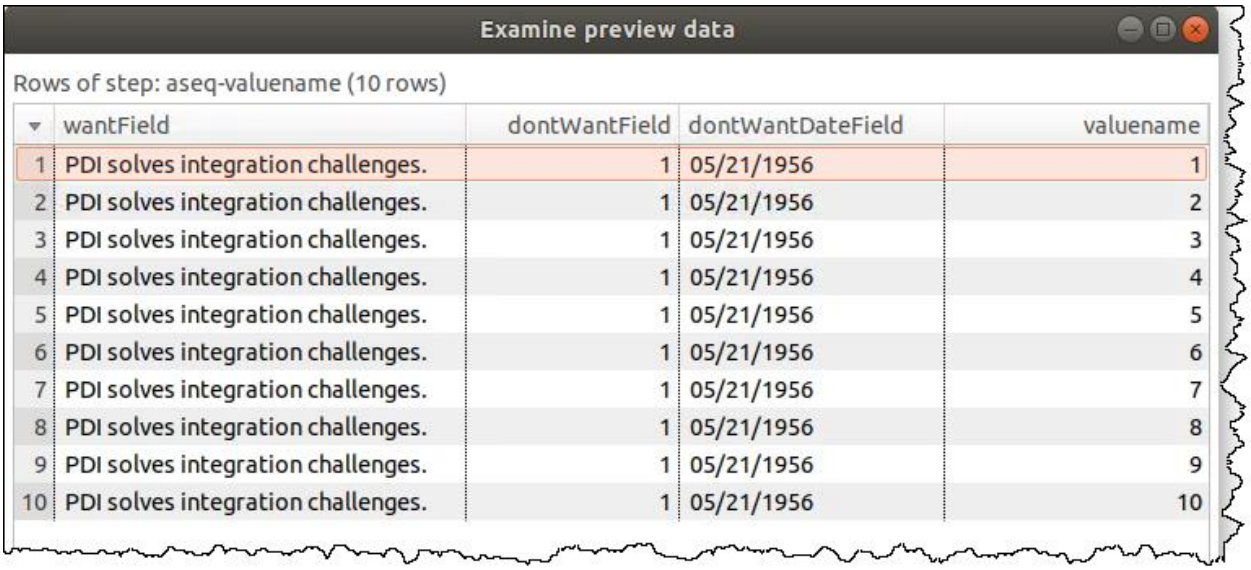
4. To preview the **aseq-valuename** step:

    a. On the canvas, click to select the **aseq-valuename** step.

    b. From the menu, click **Action** → **Preview**.

    c. Click the **Quick Launch** button.

5. Verify the preview data has all the first step's fields as well as the new field named **valuename**.



	wantField	dontWantField	dontWantDateField	valuename
1	PDI solves integration challenges.	1	05/21/1956	1
2	PDI solves integration challenges.	1	05/21/1956	2
3	PDI solves integration challenges.	1	05/21/1956	3
4	PDI solves integration challenges.	1	05/21/1956	4
5	PDI solves integration challenges.	1	05/21/1956	5
6	PDI solves integration challenges.	1	05/21/1956	6
7	PDI solves integration challenges.	1	05/21/1956	7
8	PDI solves integration challenges.	1	05/21/1956	8
9	PDI solves integration challenges.	1	05/21/1956	9
10	PDI solves integration challenges.	1	05/21/1956	10

6. To close the **Examine preview data** dialog, click **Close**.

7. To close the **Select the preview step** dialog, click **Close**.

8. To close the **Add sequence step's** configuration dialog, click **OK**.

9. Save the transformation.

Add and Configure the Select Values Step

In this section of the exercise, you add and configure a **Select values** step to select fields that you want to keep in the stream, copy a field in the stream, including its data, and add it to the stream as a new field. You duplicate the **wantField** twice. You'll then preview the **Select values** step.

1. To add a **Select values** step to the transformation, on the **Design** tab:

    a. In the **Steps search** field, type: `select`

    b. Drag the **Select values** step from the **Design** tab to the canvas and drop it to the right of the **Add sequence** step.

2. To create a hop from the **aseq-valuename** step to the **Select values** step, on the canvas:

    a. Press and hold the **Shift** key.

    b. Click and hold the **aseq-valuename** step.

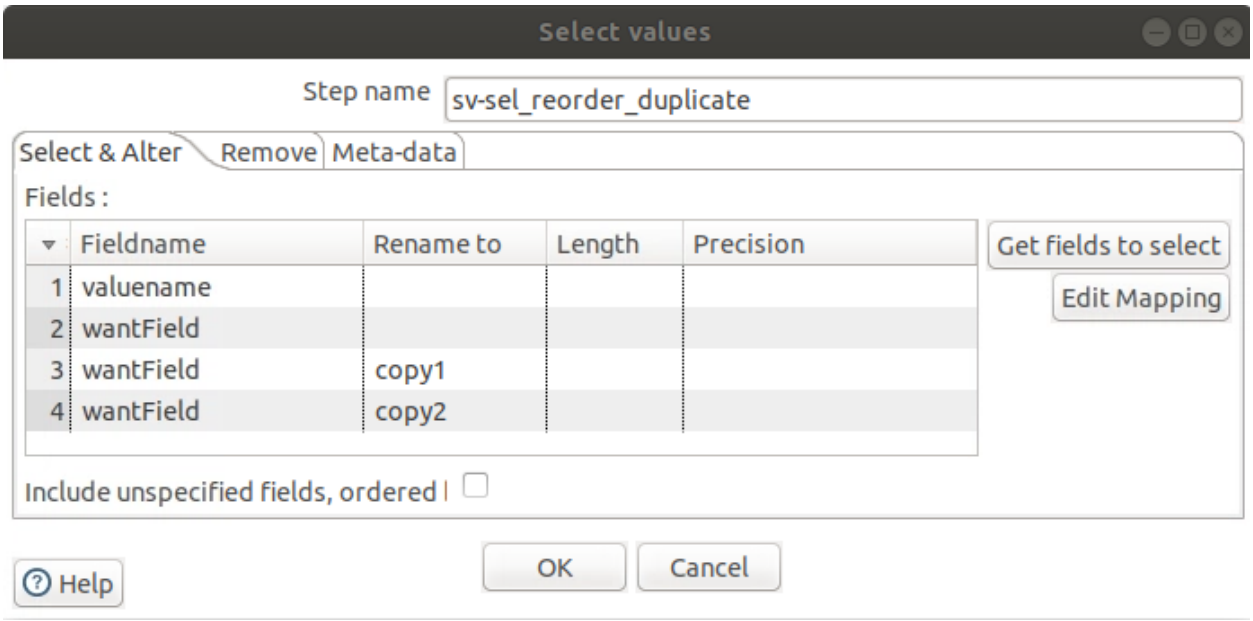
    c. Point to the **Select values** step, and then release.

3. Name the step: `sv-sel_reorder_duplicate`

4. Complete the Fields grid of the **Select & Alter** tab as follows:

Field Name	Rename to
valuename	`<empty>`
wantField	`<empty>`
wantField	`copy1`
wantField	`copy2`

5. Compare the step's configuration with the following screenshot and make any necessary changes. Then, click **OK**.



6. To preview the duplicate fields from the **sv-sel_reorder_duplicate** step:

   a. On the canvas, click to select the **sv-sel_reorder_duplicate** step.

   b. From the menu, click **Action Preview**.

   c. Click the **Quick Launch** button.

7. Verify the preview data has the **valuename**, **wantField**, and then two copies of the **wantField**.

8. To close the **Examine preview** data dialog, click **Close**.

9. To close the **Select the preview step** dialog, click **Close**.

10. Save the transformation.

11. Close the **ex1_select_values** tab.

Solution Details

The solution to this exercise can be found at:

`/home/pentaho/course_files/pdi10001/solutions/exercises`

File:

`ex1_select_values.ktr`

End of Exercise 1

Guided Demo 4: Saving to the Repository and Running Remotely

This guided demonstration briefly introduces you to the repository. You learn how to save an existing transformation to it. You can then use the repository throughout this course to save and organize your transformations and jobs. You'll see how to open a file from the file system while connected to the repository using import. In addition to using the repository, this guided demonstration has you create a run configuration to run a transformation remotely on the Pentaho Server.

Note: Throughout this course, you can choose to save objects to the file system or the repository.

Objectives

After completing this guided demonstration, you will be able to:

- Create a connection to an existing repository
- Save and import a transformation to the repository
- Create a new folder in the repository
- Create a run configuration
- Run a transformation on the Pentaho Server

Prerequisites

This guided demonstration uses the `gd2_hello_world` transformation created in Guided Demo 2. If you did not complete Guided Demo 2, you must have access to the course solution files.

You must also have access to a student environment where a repository has already been created.

Open a Transformation and Create a Repository Connection

In this section of the guided demonstration, we open the `gd2_hello_world` transformation and create a connection to an existing repository.

1. To open the `gd2_hello_world` transformation:
    a. From the menu, select **File → Open**.
    b. Navigate to the /home/pentaho/course_files/pdi1000l/my_work folder.

c. Click to select the `gd2_hello_world.ktr` transformation.

d. Click **OK**.

If necessary, you can open the transformation from the /home/pentaho/course_files/pdi1000l/solutions/guided_demos folder.

2. To create a connection to the repository, at the far right of the toolbar, click the **Connect** button.
3. To get started, in the **New Repository Connection** dialog, click **Get Started**.
4. To name the connection and set it to launch when Spoon starts:

   a. In the **Display Name**, type: `Class DI Repository`

   b. Click to select the **Launch connection on startup** checkbox.

   c. Click **Finish**.

5. To connect to the repository:

   a. Click **Connect Now**.

   b. In the **Username** field, type `admin`

   c. In the **Password** field, type `password`

   d. Click **Connect**.

6. To close all open files (tabs) and complete the connection, in the **Close Files** dialog, click **Yes**.

*If you wanted to keep the files open, you would have clicked **No**. The only reason you previously opened the `gd2_hello_world` transformation is so you could see and understand this dialog. If no tabs were open when connecting to the repository, the dialog would not have been displayed.*

Notice in the far right of the toolbar that you are now connected to the Class DI Repository as the admin user.

Save a Transformation in the Repository

In this section of the guided demonstration, we create a new transformation, and then save the transformation to a new folder in the repository.

1. To create a new transformation, on the toolbar, click the **New file** button, and then select **Transformation**.

2. To save the transformation, on the toolbar, click the **Save** button.
3. To create a new folder within the public folder, in the **Save** dialog:
    a. Click the **Public** folder.
    b. In the upper-right of the dialog, click the **Add folder** icon.
    c. In the **Name** field, type `PDI_Trn_Objects` and press **Tab**.
4. To select the **PDI_Trn_Objects** folder and name the transformation, in the **Save** dialog:
    a. If necessary, click **>** to expand the **Public** folder.
    b. Click to select the **PDI_Trn_Objects** folder.
    c. In the **Name**, type `gd4_blank_transformation`
    d. Click **OK**.
5. Close the **gd4_blank_transformation** tab.
6. To open a transformation, from the menu, select **File → Open**.
7. To open the `gd4_blank_transformation`, in the **Open** dialog:
    a. Expand the **Public** folder.
    b. Click to select the **PDI_Trn_Objects** folder.
    c. Click to select the `gd4_blank_transformation`.
    d. Click **Open**.
8. Notice in the top-left corner of Spoon that it identifies the name of the connection in addition to the name of the transformation.
9. Close the **gd4_blank_transformation** tab.

Importing a Transformation

In this section of the guided demonstration, we import the `gd2_hello_world` transformation to the repository, and then save it to the **PDI_Trn_Objects** folder.

1. To import (open) the `gd2_hello_world` transformation:

a. From the menu, select **File → Import From an XML file**.

b. Navigate to: `/home/pentaho/course_files/pdi1000l/my_work` folder

c. Click to select the `gd2_hello_world.ktr` transformation.

d. Click **OK**.

If necessary, you can import the transformation from:
`/home/pentaho/course_files/pdi1000l/solutions/guided_demos`

*The file menu's **Import From an XML file** feature does not save the file into the repository. It only opens it.*

2. To save the transformation to the repository, on the toolbar, click the **Save** button.

3. To access the repository's folders, in the **Transformation properties** dialog:

   a. Expand the **Public** folder.

   b. Click to select the **PDI_Trn_Objects** folder.

   c. Click **Save**.

4. Leave the transformation open for the next section.

Create Run Configuration and Run Remotely

Now that we have the `gd2_hello_world` transformation saved to the repository; we can run it remotely. It's a good idea to do this to make sure it runs correctly on the Pentaho Server; assuming that's where it will normally run. To do this, we will create a new run configuration and then use it when running the transformation.

1. To create a new run configuration:

   a. In the upper left of Spoon's window, click the **View** tab.

   b. Right-click **Run configurations** and then click **New**.

2. Configure the **Run configuration** dialog as shown:

Property Name	Value
Name	`Run on Pentaho Server`

Property Name	Value
Description	`This run configuration executes jobs and transformations on the Pentaho Server.`
Engine	`Pentaho` (default)
Settings	`Pentaho server` (enable radio button) *Note: This radio button is only displayed if you are connected to the repository.*

3. To close the **Run configuration** dialog, click **OK**.
4. Save the transformation.
5. To run the transformation on the Pentaho Server:
    a. In the subtoolbar, click the **Run** icon.
    b. In the **Run configuration** dropdown list, select **Run on Pentaho Server**.

    Note: The run configuration will only be listed in the dropdown list if you are connected to the repository.

    c. Click the **Run** button.
6. Scroll to the end of the **Logging** tab and find the line that reads: **"PurRepository – A request has been sent to the Pentaho Server..."**

When running a transformation or job, Spoon defaults to the run configuration used in the last run transformation or job. It's important to verify the run configuration that's selected when you execute a transformation or job to be sure it's the one you intend to use.

7. To close the transformation, click the **X** on the active transformation's tab.

    Note: Later in this course, you'll learn how to monitor transformations that have executed on the Pentaho Server.

End of Guided Demo 4

Guided Demo 5: Combining Several Inputs into One Output

*In this guided demonstration, we create a transformation that reads multiple text files using a regular expression. We then use the **Get system info** step to add the system date/time and transformation modified date/time to the stream. Finally, we use a **Text file output** step to output the stream to a delimited text file.*

Objectives

After completing this guided demonstration, you will be able to:

- Configure a **Text file input** step to read multiple text files based on a regular expression.
- • Add system and environment related data to the stream using the **Get system info** step.
- • Configure a **Text file output** step to create a single delimited output file (CSV).

Prerequisites

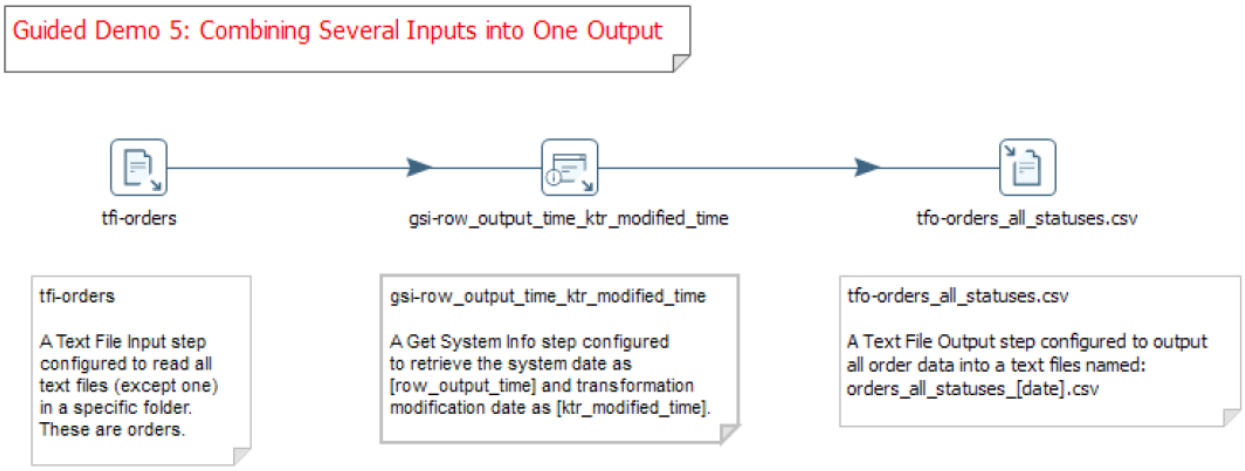
This guided demonstration requires access to the input files that reside on the course student environment.

Steps Used

This guided demonstration uses the following steps:

- **Text file input**
- **Get system info**
- **Text file output**

Transformation



Creating and Saving the Transformation

In this section of the guided demonstration, we create a new transformation and save it to the `PDI_Trn_Objects` folder on the repository.

1. To create a new transformation, on the toolbar, click the **New file** button, and then select **Transformation**.

   You can also press **Ctrl-N** to create a new transformation.

2. To save the transformation, on the toolbar, click the **Save** button.

3. To name the transformation and save it to the `PDI_Trn_Objects` folder, in the **Save** dialog:

   a. Expand the **Public** folder.

   b. Click to select the **PDI_Trn_Objects** folder.

   c. In the **File name**, type: `gd5_multi_input_one_output`

   d. Click **Save**.

Adding and Configuring the Text File Input Step

In this section, we add and configure a **Text file input** step to read several text files. We use a regular expression to define which files to include and which to exclude.

1. To add a **Text file input** step, on the **Design** tab, expand the **Input** category, and then drag the **Text file input** step to the canvas.

2. Open the new **Text file input** step.

3. To configure the step's properties, enter values in the properties as shown:

Property Name	Value
Step name	`tfi-orders`
File or directory	`/home/pentaho/course_files/pdi1000l/data_files/input/gd5_multi_input`
Regular Expression	`.*\.txt`
Exclude Regular Expression	`cancelled_orders_summary.txt`

4. To add the file and expression configuration to the **Selected files** grid, click the **Add** button.

5. To verify the directory and regular expressions are correct and the correct files will be read, click the **Show filename(s)** button.

6. Verify the correct files are shown, and then click **Close**.

7. To configure the fields for this step, click the **Fields** tab.

8. To obtain the fields via the hop:

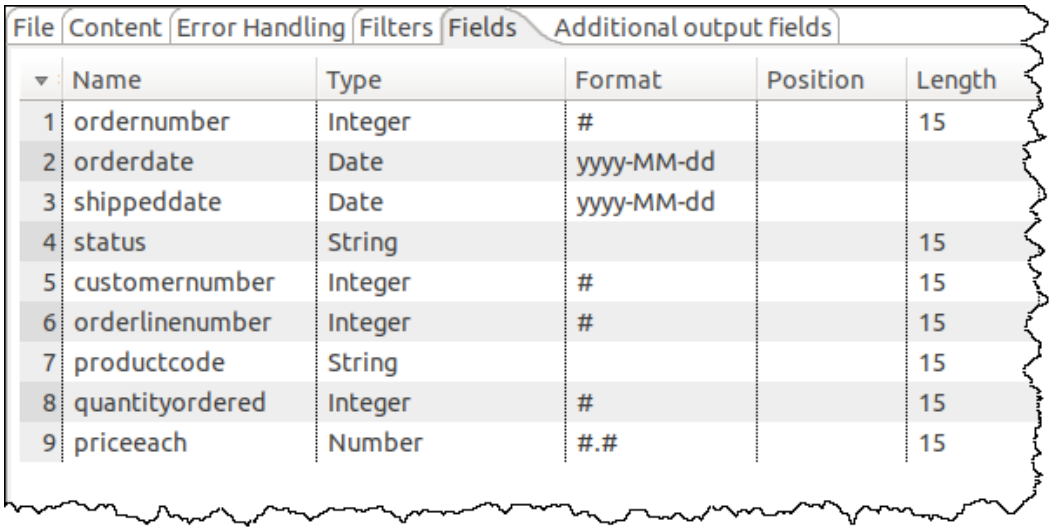
    a. Click the **Get Fields** button.

    b. In the Sample data dialog, **check** the Show sample summary checkbox.

    c. Click **OK**.

    d. Review the **Scan results**, and then click **Close**.

9. Verify the fields using the screenshot:

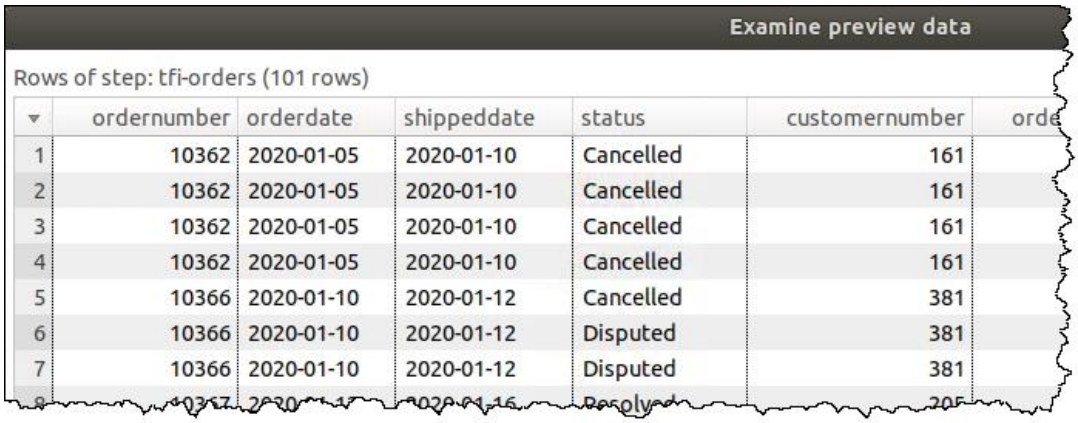

File | Content | Error Handling | Filters | Fields | Additional output fields

	Name	Type	Format	Position	Length
1	ordernumber	Integer	#		15
2	orderdate	Date	yyyy-MM-dd		
3	shippeddate	Date	yyyy-MM-dd		
4	status	String			15
5	customernumber	Integer	#		15
6	orderlinenumber	Integer	#		15
7	productcode	String			15
8	quantityordered	Integer	#		15
9	priceeach	Number	#.#		15

*The **Text file input** step must always be configured to read all the fields in the file.*

10. To preview the data and confirm it is configured properly, click **Preview rows**, and then click **OK**.

11. Verify the data generated is correct by comparing your data with the screenshot:


Examine preview data

Rows of step: tfi-orders (101 rows)

	ordernumber	orderdate	shippeddate	status	customernumber	orde
1	10362	2020-01-05	2020-01-10	Cancelled	161	
2	10362	2020-01-05	2020-01-10	Cancelled	161	
3	10362	2020-01-05	2020-01-10	Cancelled	161	
4	10362	2020-01-05	2020-01-10	Cancelled	161	
5	10366	2020-01-10	2020-01-12	Cancelled	381	
6	10366	2020-01-10	2020-01-12	Disputed	381	
7	10366	2020-01-10	2020-01-12	Disputed	381	
8	10367	2020-01-17	2020-01-16	Resolved	205	

12. To close the **Examine preview data** dialog, click **Close**.

13. To close the **Text file input** dialog, click **OK**.

14. Save the transformation.

Adding and Configuring the Get System Info Step

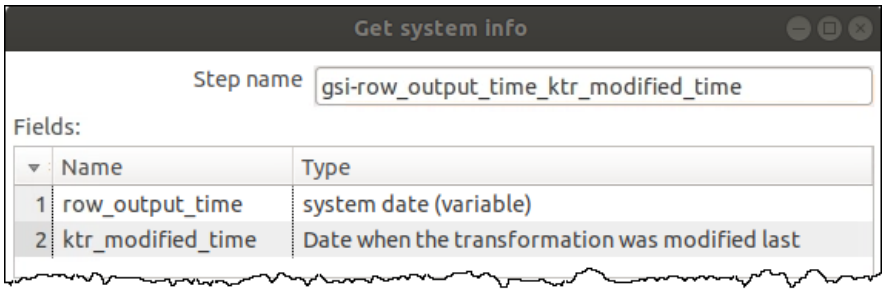
Now we add and configure the **Get system info** step to add the system date and the date the transformation

was last modified to the stream.

1. To add a **Get system info** step to the transformation, on the **Design** tab:
    a. In the **Steps** search field, type `get sys`
    b. Drag the **Get system info** step from the **Design** tab to the canvas and drop it to the right of the **tfi-orders** step.
2. To create a hop from the **tfi-orders** step to the **Get system info** step, on the canvas:
    a. Press and hold the **Shift** key.
    b. Click and hold the **tfi-orders** step.
    c. Point to the **Get system info** step, and then release.
3. Open the **Get system info** step.
4. To name the step, in the step name property, type:
   `gsi-row_output_time_ktr_modified_time`
5. To configure the **Fields** grid, add details for two rows as shown:

Name	Type
`row_output_time`	System date (variable)
`ktr_modified_time`	Date when the transformation was modified last

6. Verify the fields using the screenshot:



7. To close the **Get system info** dialog, click **OK**.
8. To preview the **Get system info** step:
    a. On the canvas, click to select the **gsi-row_output_time_ktr_modified_time** step.
    b. From the menu, click **Action** → **Preview**.

c. Click the **Quick Launch** button.

9. Verify the preview data has all the first step's fields as well as the two new fields from the **gsi-row_output_time_ktr_modified_time** step.
10. To close the **Examine preview data** dialog, click **Close**.
11. To close the **Select the preview** step dialog, click **Close**.
12. Save the transformation.

Adding and Configuring the Text File Output Step

In this section of the guided demonstration, we add and configure the **Text file output** step to output the data from the stream to a delimited text file.

1. To add a **Text file output** step to the transformation, on the **Design** tab:
   a. Click to expand the **Output** category.
   b. Drag the **Text file output** step from the **Design** tab to the canvas and drop it to the right of the **gsi-row_output_time_ktr_modified_time** step.
2. To create a hop from the **gsi-row_output_time_ktr_modified_time** step to the Text file output step, on the canvas:
   a. Press and hold the **Shift** key.
   b. Click and hold the **gsi-row_output_time_ktr_modified_time** step.
   c. Point to the **Text file output** step, and then release.
3. To configure the **Text file output** step, on the canvas, double-click the **Text file output** step.
4. To name the step, in the **Step name** property, type `tfo-orders_all_statuses.csv`
5. To provide a path and filename for the output file, in the **Filename**, type `/home/pentaho/course_files/pdi1000l/data_files/output/orders_all_statuses`
6. To set the file extension, change the **Extension** to **csv**.
7. To have the date included in the filename when the file is created, check the **Include date in filename?** property.

8. To verify the filename and path are correct and the correct file will be created, click the **Show filename(s)** button.
9. Verify the correct file is shown and then click **Close**.

   The `data_files` folder does not currently contain a folder called `output`. In addition to creating the file, Spoon will also create the folder because the Create Parent folder property is checked by default.
10. Click the **Content** tab.
11. Change the **Separator** property to **|** (a pipe symbol).
12. To configure the fields to include in the output file, click the **Fields** tab.
13. To obtain the fields from the stream, click the **Get Fields** button.
14. You may wish to have the output text file's fields in a different order than they are in the stream. To make the **row_output_time** field the first field in the output file, select that field's row and then press **Ctrl-Up Arrow** until the field is first in the list.
15. To configure the text file to have no trailing spaces in each column's data, click the **Minimal width** button.
16. To close the **Text file output** dialog, click **OK**.
17. Save the transformation.

Execute the Transformation

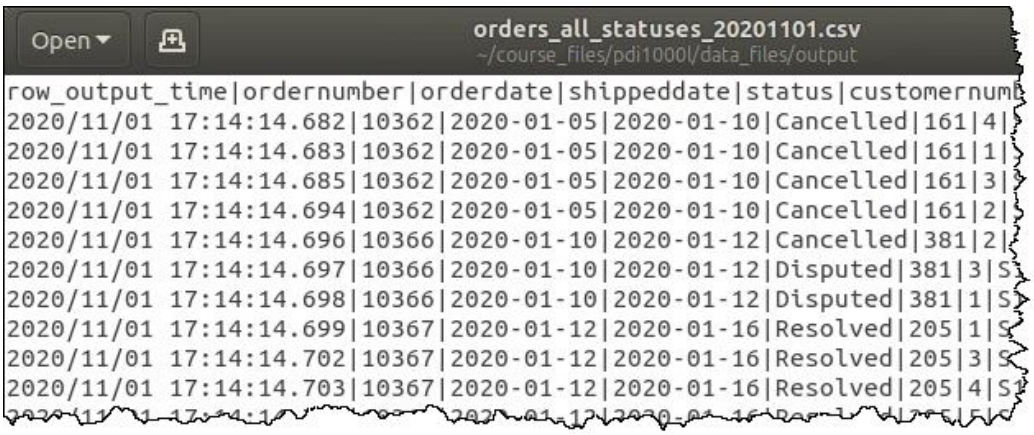
In the final section of the guided demonstration, we run the transformation and review the output file.

1. To run the transformation:
   a. On the subtoolbar, click the **Run** button.
   b. In the **Run configuration** dropdown, select **Pentaho local**. This step is necessary if the last transformation you ran used the run configuration for executing on the Pentaho Server.
   c. Click **Run**.
2. To verify the **Table output** step wrote data to a file:
   a. In the bottom **Execution Results** panel, click the **Step Metrics** tab.

b. Notice the Output column for the **tfo_orders_all_statuses.csv** step indicates the number of lines output to the file the transformation created.

3. To verify the file was created, navigate to `/home/pentaho/course_files/pdi10001/data_files/output` and verify the `orders_all_statuses.csv` file exists.

4. Open the file in a text editor and notice how orders from all the input files are included.



5. Close the file and return to Spoon.

6. Close the **gd5_multi_input_one_output** tab.

Solution Details

The solution to this guided demonstration can be found at:

`/home/pentaho/course_files/pdi10001/solutions/guided_demos`

File:

`gd5_multi_input_one_output.ktr`

Output:

`/home/pentaho/course_files/pdi10001/solutions/output_complete/orders_all_statuses_[date].csv`

End of Guided Demo 5

Guided Demo 6: Creating kettle.properties Variables

In this guided demonstration, we add two `kettle.properties` global variables for the folders containing the input and output files. We are then able to use those variables instead of typing the paths.

Note: When you create transformations more complex than this one, it is best practice to create and test the transformation using hard-coded values before attempting to parameterize it. This practice will save overall development time by reducing troubleshooting.

Objectives

After completing this guided demonstration, you will be able to:

- Edit the `kettle.properties` file
- Add global variables and values in the `kettle.properties` file
- Configure a **CSV file input** step using a variable to define the input file folder
- Configure a **Text file output** step using a variable to define the output file folder

Prerequisites

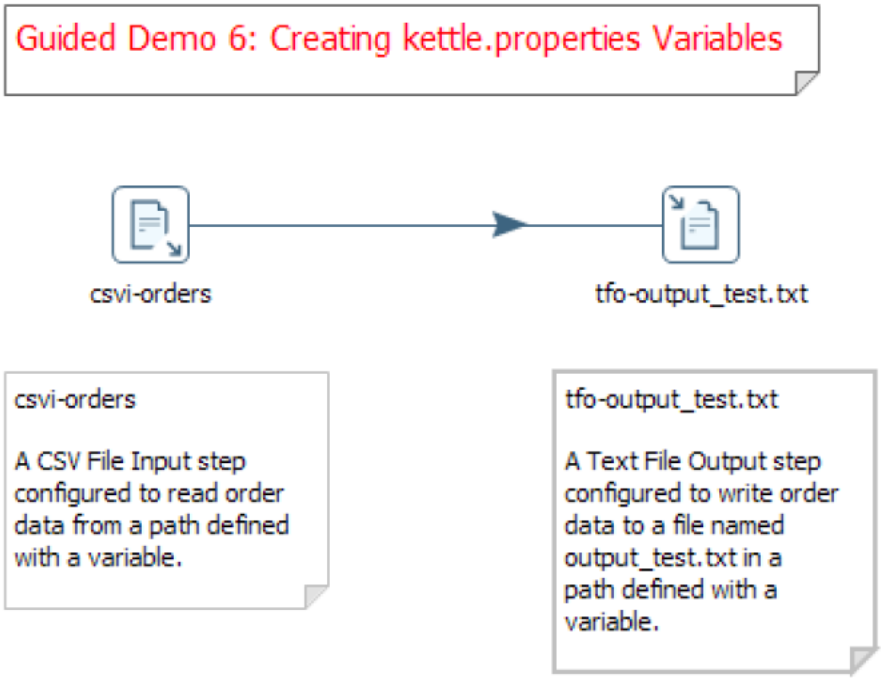
This guided demonstration requires access to the input files that reside on the course student environment.

Steps Used

This guided demonstration uses the following steps:

- **CSV file input**
- **Text file output**

Transformation



Creating the Transformation and Adding Variables for File Paths

In this section of the guided demonstration, we create a new transformation, and then edit the `kettle.properties` file to add global variables for the input and output folders.

1. To create a new transformation, on the toolbar, click the **New** file button, and then select **Transformation**.
2. To save the transformation, on the toolbar, click the **Save** button.
3. To name the transformation and save it to the `PDI_Trn_Objects` folder, in the **Save** dialog:
    a. Click **>** to expand the **Public** folder.
    b. Click to select the **PDI_Trn_Objects** folder.
    c. In the **File name**, type `gd6_kettle_variables`
    d. Click **Save**.
4. From the menu, click **Edit → Edit the kettle.properties file**.
5. Scroll down to the last row.

6. To add a new row, right-click on the last row, and then select **Insert after this row**.

7. Repeat the step to add a second row.

8. In the new lines, enter the following variables:

Variable name	Value
`DIR_INPUT`	`/home/pentaho/course_files/pdi1000l/data_files/input`
`DIR_OUTPUT`	`/home/pentaho/course_files/pdi1000l/data_files/output`

9. To close the `kettle.properties` file, click **OK**.

Adding a CSV File Input Step with a Variable

In this section of the guided demonstration, we add and configure a **CSV file input** step and use the `DIR_INPUT` variable to define the input folder.

1. To add a **CSV file input** step, on the **Design** tab, expand the **Input** category, and then drag the **CSV file input** step to the canvas.

2. Open the **CSV file input** step.

You might find it helpful to enlarge the step dialog window.

3. Name the step **csvi-orders**

4. To use the `DIR_INPUT` variable:

    a. Click in the **Filename** field.

    b. On the keyboard, press **Ctrl-Space**.

    c. From the list of variables, double-click `DIR_INPUT`.

*Note: Variables and parameters can be used in any field in Spoon that has a blue diamond dollar sign next to it. Pressing **Ctrl-Space** in the field opens a list of the variable and parameters loaded into memory.*

5. To append the filename to the variable, in the **Filename** field, after the variable, type `/order_file.csv`

6. To change the separator, in the **Delimiter**, type a semicolon `;`

7. To populate the **Fields** grid:

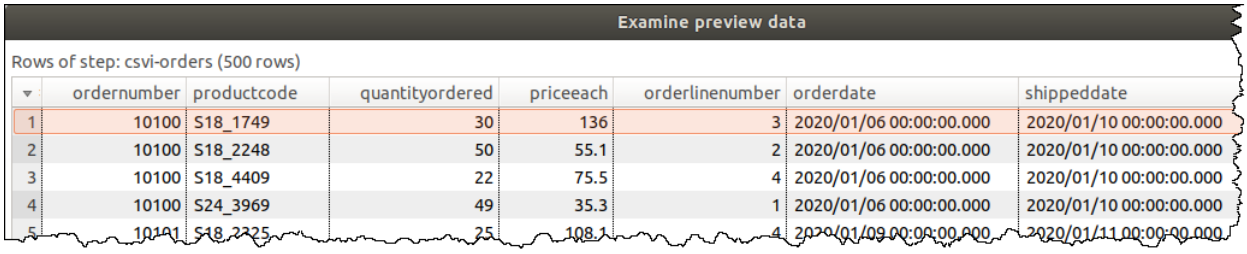
a. Click the **Get Fields** button.

b. In the **Sample data** dialog, change the value to `0`.

c. Click **OK**.

8. To preview the data and confirm it is configured properly, click **Preview**, and then click **OK**.

9. Verify the data generated is correct by comparing your data with the screenshot:



	ordernumber	productcode	quantityordered	priceeach	orderlinenumber	orderdate	shippeddate
1	10100	S18_1749	30	136	3	2020/01/06 00:00:00.000	2020/01/10 00:00:00.000
2	10100	S18_2248	50	55.1	2	2020/01/06 00:00:00.000	2020/01/10 00:00:00.000
3	10100	S18_4409	22	75.5	4	2020/01/06 00:00:00.000	2020/01/10 00:00:00.000
4	10100	S24_3969	49	35.3	1	2020/01/06 00:00:00.000	2020/01/10 00:00:00.000
5	10101	S18_2325	25	108.1	4	2020/01/09 00:00:00.000	2020/01/11 00:00:00.000

10. To close the **Examine preview data** dialog, click **Close**.

11. To close the **csvi-orders** dialog, click **OK**.

12. Save the transformation.

Adding a Text File Output Step with a Variable

In this section of the guided demonstration, we add and configure a **Text file output** step and use the `DIR_OUTPUT` variable to define the output folder.

1. To add a **Text file output** step to the transformation, on the **Design** tab:

   a. Click to expand the **Output** category.

   b. Drag the **Text file output** step from the **Design** tab to the canvas and drop it to the right of the **csvi-orders** step.

2. To create a hop from the **csvi-orders** step to the **Text file output** step, on the canvas:

   a. Press and hold the **Shift** key.

   b. Click and hold the **csvi-orders** step.

   c. Point to the **Text file output** step, and then release.

   d. From the **context** menu, select **Main output of step**.

3. Open the **Text file output** step.

4. Name the step **tfo-output_test.txt**
5. To use the `DIR_OUTPUT` variable:
    a. Click in the **Filename** field.
    b. Delete the word "file".
    c. On the keyboard, press **Ctrl-Space**.
    d. From the list of variables, double-click `DIR_OUTPUT`.
6. To append the filename to the variable, in the **Filename** field, after the variable, type `/output_test`
7. To configure the fields to include in the output file, click the **Fields** tab.
8. To obtain the fields from the stream, click the **Get Fields** button.
9. To close the Text file output dialog, click **OK**.
10. Save the transformation.

Run the Transformation

In this section of the guided demonstration, we run the transformation, and then review the output file.

1. To run the transformation, on the subtoolbar, click the **Run** button.
2. To view the variables, in the **Run Options** dialog, click the **Variables** tab.
3. To run the transformation, in the **Run Options** dialog, click **Run**.

    Notice the Output field indicates the number of lines output to the file the transformation created.
4. To verify the file was created, in Windows Explorer, navigate to `/home/pentaho/course_files/pdi1000l/data_files/output` and verify the `output_test.txt` file exists.
5. Open the file and notice how orders from all the input files are included.
6. Close the file and return to Spoon.
7. Close the **gd6_kettle_variables** tab.

Solution Details

The solution to this guided demonstration can be found at:

`/home/pentaho/course_files/pdi1000l/solutions/guided_demos`

File:

`gd6_kettle_variables.ktr`

Output:

`/home/pentaho/course_files/pdi1000l/solutions/output_complete/output_test.txt`

End of Guided Demo 6

Exercise 2: CSV Input to Multiple Text Output Using Switch/Case

*In this exercise, you create a transformation that reads a CSV file containing order data (including the country of origin). Then, it will route the orders from specific country's to text files using the **Switch/case** and **Text file output** steps. In the previous guided demonstration, you used `kettle.properties` variables, but in this exercise you use transformation parameters to specify the file input and output folder locations.*

Objectives

After completing this exercise, you will be able to:

- Create and use transformation parameters to define locations for input and output folder locations
- Configure a **CSV file input** step using a variable to define the input file location
- Configure a **Switch/case** step that sends data to specific steps depending on the data in the incoming stream
- Create hops from a **Switch/case** step based on specific case values
- Configure a **Text file output** step using a variable to define the output file location
- • Configure a **Dummy (do nothing)** step to collect metrics for all data that doesn't match a specific case value.

Prerequisites

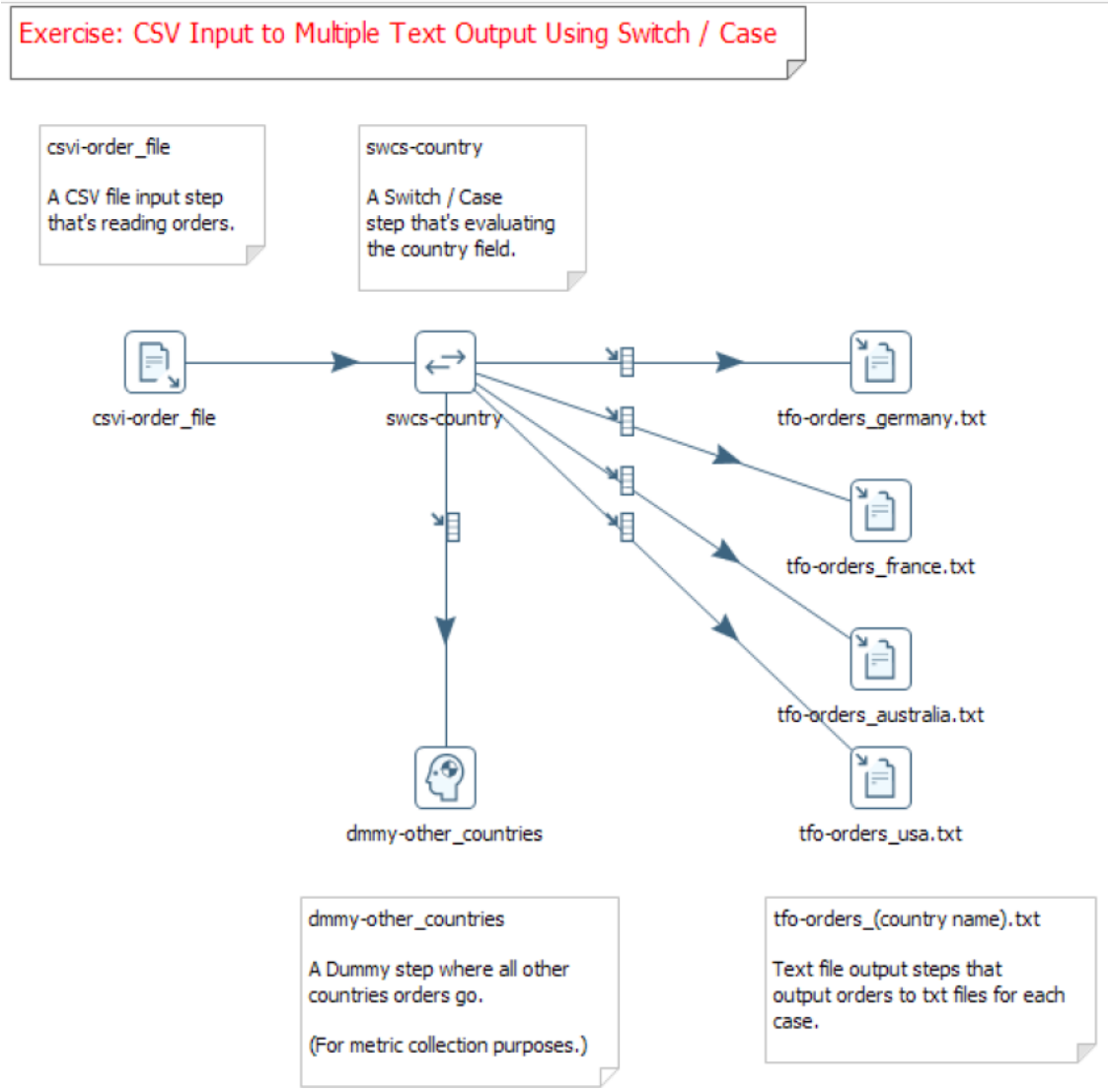
This exercise requires access to the input files that reside on the course student environment.

Steps Used

This exercise uses the following steps:

- **CSV file input**
- **Switch/case**
- **Text file output**
- **Dummy (do nothing)**

Transformation



Creating the Transformation and Adding Transformation Parameters

In this section of the exercise, you create a new transformation, and add transformation parameters to define the input and output folders.

1. To create a new transformation, on the toolbar, click the **New file** button, and then select **Transformation**.

2. To name the transformation and save it to the `PDI_Trn_Objects` folder, in the **Save** dialog:

    a. Click **>** to expand the **Public** folder.

b. Click to select the **PDI_Trn_Objects** folder.

c. In the **File name**, type `ex2_csv_input_text_output`

d. Click **Save**.

3. To open the **Transformation properties** dialog, on the canvas, double-click an empty area.

4. To create the transformation parameters, in the **Transformation properties** dialog:

a. Click the **Parameters** tab.

b. Create two new parameters according to the table below:

Parameter	Default Value
`KTR_DIR_INPUT`	`/home/pentaho/course_files/pdi1000l/data_files/input`
`KTR_DIR_OUTPUT`	`/home/pentaho/course_files/pdi1000l/data_files/output2` Note: This folder does not exist, but it will be created for you when the transformation executes. This parameter will override the variable with the same name.

c. Click **OK**.

5. Save the transformation.

Configure the CSV File Input Step with a Parameter

In this section of the exercise, you add and configure the CSV file input step with a parameter identifying the input file folder.

1. To add a **CSV file input** step, on the **Design** tab, expand the **Input** category, and drag the **CSV file input** step to the canvas.

2. To configure the **CSV file input** step, on the canvas, double-click the **CSV file input** step.

3. To name the step, in the **Step name**, type `csvi-order_file`

4. To use the `KTR_DIR_INPUT` parameter:

a. Click in the **Filename** field.

b. On the keyboard, press **Ctrl-Space**.

c. From the list of variables, double-click `KTR_DIR_INPUT`.

5. To append the filename to the variable, in the **Filename** field, after the variable, type `/order_file.csv`

6. To change the separator, in the **Delimiter**, type a semicolon `;`

7. To turn off lazy conversion, click to deselect the **Lazy conversion?** checkbox.

8. To populate the **Fields** grid:

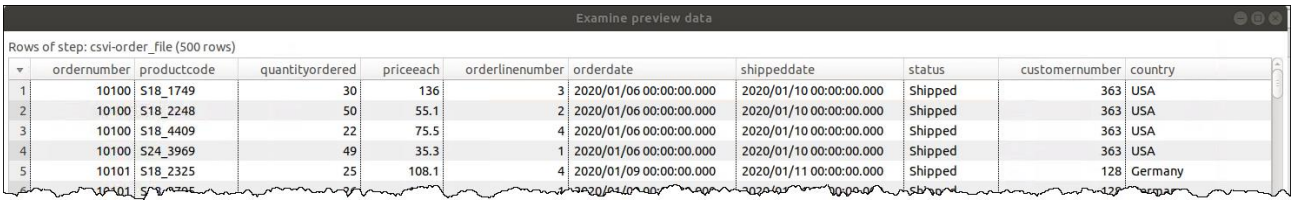
    a. Click the **Get Fields** button.

    b. In the **Sample data** dialog, change the value to `0`.

    c. Click **OK**.

9. To preview the data and confirm it is configured properly, click **Preview**, and then click **OK**.

10. Verify the data generated is correct by comparing your data with the screenshot:



	ordernumber	productcode	quantityordered	priceeach	orderlinenumber	orderdate	shippeddate	status	customernumber	country
1	10100	S18_1749	30	136	3	2020/01/06 00:00:00.000	2020/01/10 00:00:00.000	Shipped	363	USA
2	10100	S18_2248	50	55.1	2	2020/01/06 00:00:00.000	2020/01/10 00:00:00.000	Shipped	363	USA
3	10100	S18_4409	22	75.5	4	2020/01/06 00:00:00.000	2020/01/10 00:00:00.000	Shipped	363	USA
4	10100	S24_3969	49	35.3	1	2020/01/06 00:00:00.000	2020/01/10 00:00:00.000	Shipped	363	USA
5	10101	S18_2325	25	108.1	4	2020/01/09 00:00:00.000	2020/01/11 00:00:00.000	Shipped	128	Germany

11. To close the **Examine preview data** dialog, click **Close**.

12. To close the **csvi-order_file** dialog, click **OK**.

13. Save the transformation.

Configure the Switch/Case Step

In this section of the exercise, you add and configure the **Switch/case** step to define a condition where the Country equals Germany, France, Australia, or USA.

1. To add a **Switch/case** step to the transformation, on the **Design** tab:

    a. Click to expand the **Flow** category.

    b. Drag the **Switch/case** step from the **Design** tab to the canvas and drop it to the right of the **csvi-order_file** step.

2. To create a hop from the **csvi-order_file** step to the **Text file output** step, on the canvas:

    a. Press and hold the **Shift** key.

b. Click and hold the **csvi-order_file** step.

c. Point to the **Switch/case** step, and then release.

d. From the context menu, select **Main output of step**.

3. To configure the **Switch/case** step, on the canvas, double-click the **Switch/case** step.
4. To name the step, in the **Step name**, type `swcs-country`
5. To specify the field name to switch, from the **Field name to switch** dropdown list, and then select **country**.
6. To set the value comparison to contain the specified value (instead of exactly match), check the **Use string contains comparison** checkbox.
7. To set the data type of the case value, from the **Case value data type** dropdown list, select **String**.
8. To set the case values, in the **Case values** grid, add four values for `Germany`, `France`, `Australia`, and `USA`.

The remaining properties of this step will automatically get set as you configure the rest of the transformation.

9. To close the **Switch/case** dialog, click **OK**.
10. Save the transformation.

Configure the Germany Text File Output Step with a Conditional Hop

In this section of the exercise, you add and configure a **Text file output** step using a parameter to define the output file location. Since the output file will only contain results for Germany, you add a conditional hop from the **Switch/case** step.

1. To add a **Text file output** step to the transformation, on the **Design** tab:

   a. Click to expand the **Output** category.

   b. Drag the **Text file output** step from the **Design** tab to the canvas and drop it to the right of the **swcs-country** step.

2. To create a hop from the **swcs-country** step to the **Text file output** step, on the canvas:

a. Press and hold the **Shift** key.

b. Click and hold the **swcs-country** step.

c. Point to the **Text file output** step, and then release.

d. From the context menu, select the case target for value **Germany**.

3. To configure the **Text file output** step, on the canvas, double-click the **Text file output** step.

4. To name the step, in the **Step name**, type `tfo-orders_germany.txt`

5. To use the `KTR_DIR_OUTPUT` parameter:

a. Click in the **Filename** field.

b. Delete the word "file".

c. On the keyboard, press **Ctrl-Space**.

d. From the list of variables, double-click `KTR_DIR_OUTPUT`.

6. To append the filename to the variable, in the **Filename** field, after the variable, type `/orders_germany`

7. To configure the fields to include in the output file, click the **Fields** tab.

8. To obtain the fields via the hop, click the **Get Fields** button.

9. To close the **Text file output** dialog, click **OK**.

10. Save the transformation.

Configure the Remaining Text File Output Steps with Conditional Hops

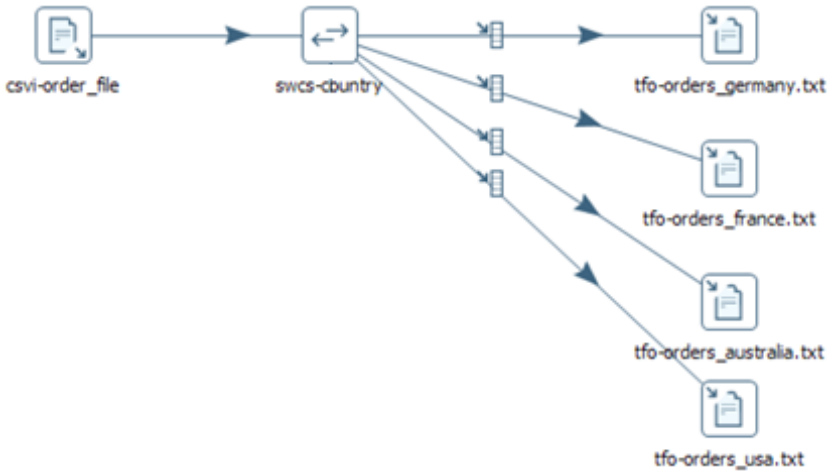
In this section of the exercise, you duplicate the Text file output step three times and then configure the outputs for France, Australia, and USA, and add conditional hops from the **Switch/case** step. Using this duplicate step technique saves you valuable time since the steps are configured nearly identically. Only the step names and the filenames they create are different.

11. To duplicate the tfo-orders_germany.txt step, right-click the step, and then use the **Duplicate** option from the context menu.

12. Duplicate it **two** more times.

13. Edit each of the duplicate steps and update the **step name** and the **Filename** properties. Use the screenshot below for guidance.

14. Create hops between the swcs-country and each of the remaining Text File output steps, making sure to choose the case target values from the pop-up menu. For example, the case target France goes to the tfo-orders_france.txt step.



15. Save the transformation.

Configure the Dummy (do nothing) Step with a Conditional Hop

In this section of the exercise, you add and configure a **Dummy (do nothing)** step for all the countries that do not match the **Switch/case** condition using a conditional hop from the **swcs-country** step.

1. To add a **Dummy (do nothing)** step to the transformation, on the **Design** tab:

    a. Click to expand the **Flow** category.

    b. Drag the **Dummy (do nothing)** step from the **Design** tab to the canvas and drop it below the **swcs-country** step.

2. To create a hop from the **swcs-country** step to the **Dummy (do nothing)** step, on the canvas:

    a. Press and hold the **Shift** key.

    b. Click and hold the **swcs-country** step.

    c. Point to the **Dummy (do nothing)** step, and then release.

    d. From the **context** menu, select the **default target step**.

3. To configure the **Dummy (do nothing)** step, on the canvas, double-click the **Dummy (do nothing)** step.

4. To name the step, in the **Step name** property, type `dmmy-other_countries`, and then click **OK**.

   To show the **OK** button, enlarge the **Dummy (do nothing)** dialog.

5. To see how the **swcs-country** step properties have been updated automatically when you created the hops and choose a target case, open the **swcs-country** step and notice the **Target step** column is filled out for you.

6. Save the transformation.

Run the Transformation

In this section of the exercise, you run the transformation, and then review the output files.

1. To run the transformation, on the active transformation's tab toolbar, click the **Run** button.

You can press F9 to run a transformation.

2. In the **Run Options** dialog, click **Run**.

In the bottom **Execution Results** pane, click the **Step Metrics** tab and notice the **Output** column indicates the number of rows output to the files the transformation created.

3. To verify the files were created, in Windows Explorer, navigate to `/home/pentaho/course_files/pdi1000l/data_files/output2` and verify the following files exist:

   a. `orders_australia.txt`

   b. `orders_france.txt`

   c. `orders_germany.txt`

   d. `orders_usa.txt`

4. Open each of the files and notice how the data for each country is written to the appropriate country text file.

5. Close the file and return to Spoon.

6. Close the **ex2_csv_input_text_output** tab.

Solution Details

The solution to this exercise can be found at:

`/home/pentaho/course_files/pdi1000l/solutions/exercises`

File:

`ex2_csv_input_text_output.ktr`

Output:

`/home/pentaho/course_files/pdi1000l/solutions/output_complete/`

`orders_australia.txt`

`orders_france.txt`

`orders_germany.txt`

`orders_usa.txt`

End of Exercise 2

Guided Demo 7: Connections and the Database Explorer

In this guided demonstration, we create a database connection and use the Database Explorer to view the database.

Objectives

After completing this guided demonstration, you will be able to:

- Create a database connection to connect to a PostgreSQL database
- Share database connections
- Use Database Explorer to view the tables in a database connection

Prerequisites

This guided demonstration requires access to the input files that reside on the course student environment.

You must also have access to a student environment where a repository has already been created.

Creating Database Connections

In this section of the guided demonstration, we create database connections to the `pentaho_oltp` and `pentaho_olap` PostgreSQL databases. Then, they will be shared.

1. To create a new transformation, on the toolbar, click the **New file** button, and then select **Transformation**.

This transformation will be used in Exercise 3.

2. To save the transformation, on the toolbar, click the **Save** button.
3. To name the transformation and save it to the `PDI_Trn_Objects` folder, in the **Save** dialog:
    a. Click **>** to expand the **Public** folder.
    b. Click to select the **PDI_Trn_Objects** folder.
    c. In the **File name**, type `ex3_table_input_output`

d. Click **Save**.

4. To add a new database connection, in the left panel:

   a. Click the **View** tab.

   b. Right-click **Database connections**.

   c. Select **New**.

5. In the **Database Connection** dialog, type or choose:

Field	Value
Connection Name	`pentaho_oltp`
Connection Type	`PostgreSQL`
Access	`Native (JDBC)`
Host Name	`localhost`
Database Name	`pentaho_oltp`
Port Number	`5432`
User Name	`pentaho`
Password	`password123`

6. To verify the connection, in the **Database Connection** dialog, click **Test**.

7. To dismiss the **Database Connection Test** dialog, click **OK**, and then to close the **Database Connection** dialog, click **OK**.

8. To create a new database connection for the `pentaho_olap` database, repeat the previous steps using the following values:

Field	Value
Connection Name	`pentaho_olap`
Connection Type	`PostgreSQL`
Access	`Native (JDBC)`
Host Name	`localhost`

Field	Value
Database Name	`pentaho_olap`
Port Number	`5432`
User Name	`pentaho`
Password	`password123`

9. Alternately, you can duplicate an existing database connection by right-clicking the connection name on the View tab.

10. To share the database connections, in the **View** tab, right-click each database connection that you created, and then click **Share**.

11. Save the transformation.

Sharing the database connection is not required. It is included in this guided demonstration so that that if you did disconnect from the repository, the database connection would still be available for use without having to recreate it.

Using Database Explorer

In this section of the guided demonstration, we use Database Explorer to examine the `pentaho_oltp` database.

1. To view the database connections, on the **View** tab, click to expand **Database connections**.

2. Right-click **pentaho_oltp**, and then from the context menu, click **Explore**.

3. To preview the **customers** table, in the **Database Explorer** window:

    a. Click to expand **pentaho_oltp**.

    b. Click to expand **Tables**.

    c. Right-click **customers**.

    d. Select **Preview first 100**.

4. Examine the first 100 rows in the customers table, and then in the **Examine preview data** dialog, click **Close**.

5. To execute SQL to show the table ordered by the city column, in the **Database Explorer** window, right-click customers, and then select **View SQL**.

6. In the **Simple SQL editor** dialog, append the SQL with `ORDER BY city`, and then click **Execute**.

You might find it helpful to enlarge the Simple SQL editor dialog.

7. Examine the results, and then in the **Examine preview data** dialog, click **Close**.
8. To close the **Results of the SQL statements** dialog, click **OK**.
9. To close **Simple SQL editor** dialog, click **Close**.
10. To close **Database Explorer**, click **OK**.
11. Save the transformation.
12. Keep the transformation open for the next exercise.

End of Guided Demo 7

Exercise 3: Reading and Writing to Data Tables

*In the first part of this exercise, you complete the transformation started in Guided Demo 7 to read data from the `orderdetails` table in the `pentaho_oltp` database using a **Table input** step. You then configure a **Table output** step to write some of the data to a new table in the `pentaho_olap` database.*

*In the second part of the exercise, you modify the transformation to use a **Text file output** step instead of the **Table output** step to create a text file with the data.*

Objectives

After completing this exercise, you will be able to:

- Configure a **Table input** step to obtain data from a database table
- Configure a **Table output** step to write data to a database table
- Use the SQL helper button to generate SQL to create a database table
- Configure a **Text file output** to write data to a text file

Prerequisites

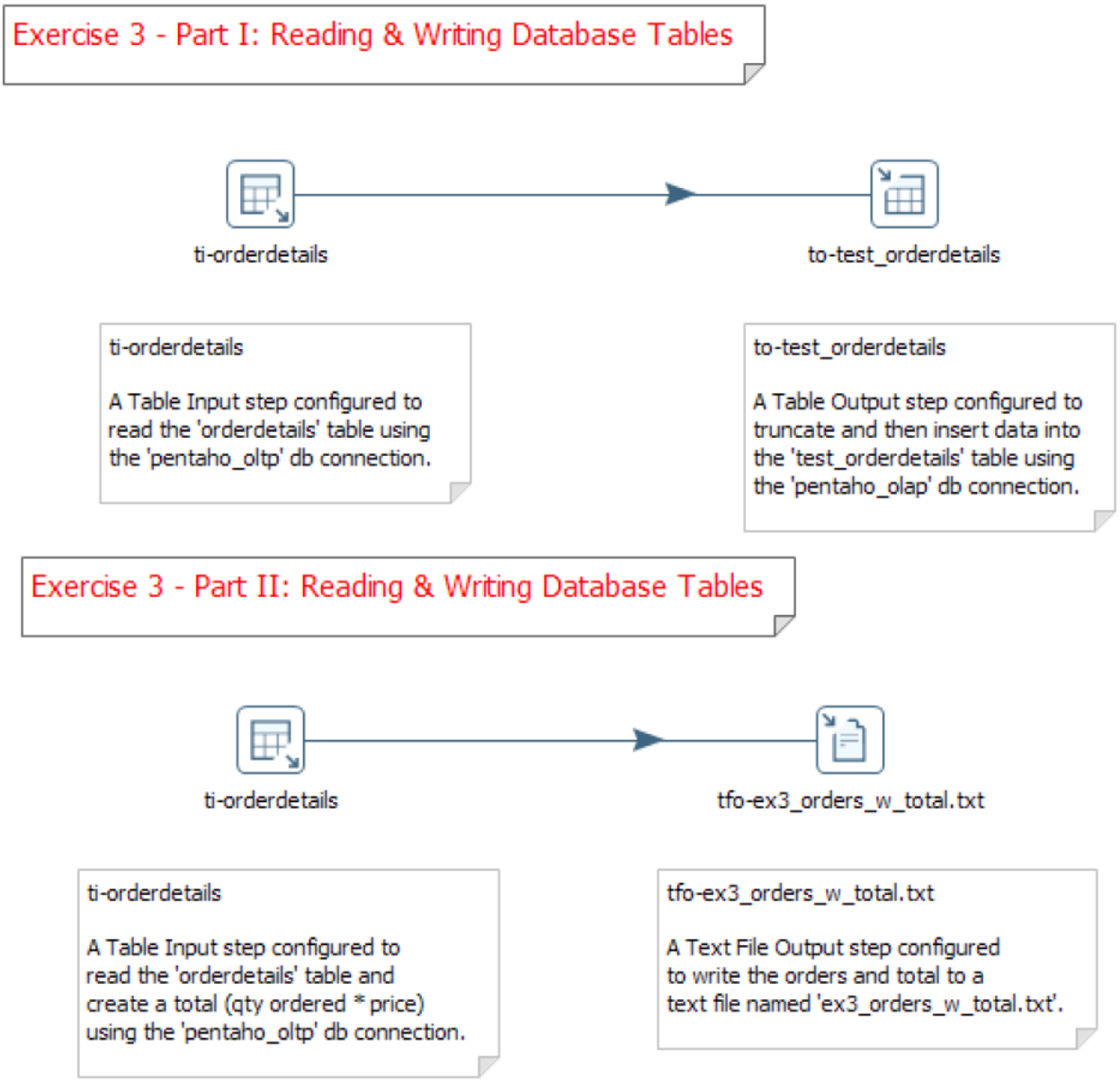
Prior to completing this exercise, you must complete Guided Demo 7: Connections and the Database Explorer.

Steps Used

This exercise uses the following steps:

- **Table input**
- **Table output**
- **Text file output**

Transformations



Part I

In Part I of this exercise, you create a transformation that gets data from a table in the `pentaho_oltp` database and then outputs the data into a table in the `pentaho_olap` database. Transferring data between databases is a common practice in data warehousing.

Configure the Table Input Step

In this section of the exercise, you configure a **Table input** step to obtain data from the `orderdetails` table in the `pentaho_oltp` database. You add a **Table output** step to the transformation and hop to the

transformation now, but you will configure the **Table output** step in the next section.

1. In the `ex3_table_input_output` transformation, click the **Design** tab.
2. From the **Input** category, drag a **Table input** step to the canvas.
3. From the **Output** category, drag a **Table output** step to the canvas, and drop it to the right of the **Table input** step
4. Create a hop from the **Table input** step to the **Table output** step.
5. To configure the **Table input** step, on the canvas, double-click the **Table input** step.
6. To name the step, in the Step name, type `ti-orderdetails`
7. To select the database connection, from the **Connection** dropdown list, select **pentaho_oltp**.
8. To generate the SQL that obtains the data, in the **Table input** dialog, click the **Get SQL select statement** button.
9. To select the `orderdetails` table from the `pentaho_oltp` database connection, in the **Database Explorer** dialog:
    a. Click to expand **pentaho_oltp**.
    b. Click to expand **Tables**.
    c. Click to select the **orderdetails** table.
    d. Click **OK**.
10. To include the field names in the SQL statement, in the **Question?** dialog, click **Yes**.
11. To preview the query:
    a. In the **Table input** dialog, click **Preview**.
    b. In the **Enter preview size** dialog, click **OK**.
    c. Review the results.
    d. In the **Examine preview data** dialog, click **Close**.
12. To close the **Table input** dialog, click **OK**.

Configure the Table Output Step and Run the Transformation

In this section of the exercise, you configure the **Table output** step to write the data to a new table in the `pentaho_olap` database. Since the table doesn't exist in the `penatho_olap` database, you use the **SQL helper** button to generate SQL to create the table. You then run the transformation.

1. To configure the **Table output** step, on the canvas, double-click the **Table output** step.
2. To name the step, in the **Step name**, type `to-test_orderdetails`
3. To configure the **Table output** step, in the **Table output** dialog, type or select the following:

Field	Value
Connection	`pentaho_olap`
Target schema	[leave blank]
Target table	`test_orderdetails`
Commit size	`1000`
Truncate table	[checked]

4. To generate the SQL statement to create the table, in the **Table output** dialog, click the **SQL** button.
5. Verify the syntax of the SQL statement, and then to execute the SQL statement, in the **Simple SQL editor** dialog, click **Execute**.
6. Verify the SQL statement executed, and then in the **Results of the SQL statements** dialog, click **OK**.
7. To close the **Simple SQL editor** dialog, click **Close**.
8. To close the **Table output** dialog, in the **Table output** dialog, click **OK**.
9. Save the transformation.
10. To run the transformation:
    a. On the subtoolbar, click the **Run** button.
    b. In the **Run Options** dialog, click **Run**.

Modify the Table Input and Table Output Steps

In this section of the exercise, you modify the **Table input** step to add a calculated value. You then modify the **Table output** step to generate a SQL statement that adds the new column to the output table.

1. To add a calculated column, double-click the **ti-orderdetails (Table input)** step.
2. To calculate the total, add the following line below the `, priceeach` line:

```
, quantityordered * priceeach as total
```

3. To preview the query:
    a. In the **Table input** dialog, click **Preview**.
    b. In the **Enter preview size** dialog, click **OK**.
    c. Review the results.
    d. In the **Examine preview data** dialog, click **Close**.
4. To close the **Table input** dialog, click **OK**.
5. To update the **to-test_orderdetails (Table output)** step, double-click the **to-test_orderdetails** step.
6. In the **Table output** dialog, click the **SQL** button.
7. Verify the syntax of the SQL statement, and then to execute the SQL statement, in the **Simple SQL editor** dialog, click **Execute**.

The SQL button generates any SQL that needs to be executed against the target table to make its layout match what you are trying to load into it, so in this case it only needs to alter the table to add the total column.

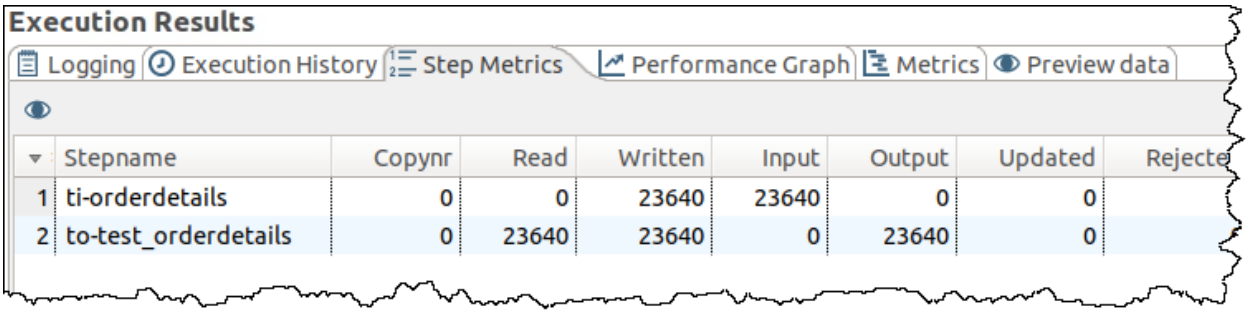
8. Verify the SQL statement executed, and then in the **Results of the SQL statements** dialog, click **OK**.
9. Close the Simple SQL editor dialog.
10. To close the **Table output** dialog, in the **Table output** dialog, click **OK**.
11. Save the transformation.

Run the Transformation

In this section of the exercise, you run the transformation and review the results in the `pentaho_olap` database.

1. To run the transformation:
    a. On the subtoolbar, click the **Run** button.

b. In the **Run Options** dialog, click **Run**.



	Stepname	Copynr	Read	Written	Input	Output	Updated	Rejecte
1	ti-orderdetails	0	0	23640	23640	0	0	
2	to-test_orderdetails	0	23640	23640	0	23640	0	

2. To view the database connections, on the **View** tab, click to expand **Database connections**.
3. Right-click **pentaho_olap**, and then from the context menu, click **Explore**.
4. To preview the `test_orderdetails` table, in the **Database Explorer** window:
   a. Click to expand **pentaho_olap**.
   b. Click to expand **Tables**.
   c. Right-click **test_orderdetails**.
   d. Select **Preview first 100**.
5. Examine the first 100 rows in the `customers` table, and then in the **Examine preview data** dialog, click **Close**.
6. To close **Database Explorer**, click **OK**.

Part II

In Part II of this exercise, you modify the transformation to write data to a text file instead of a database table.

Configure the Text File Output Step

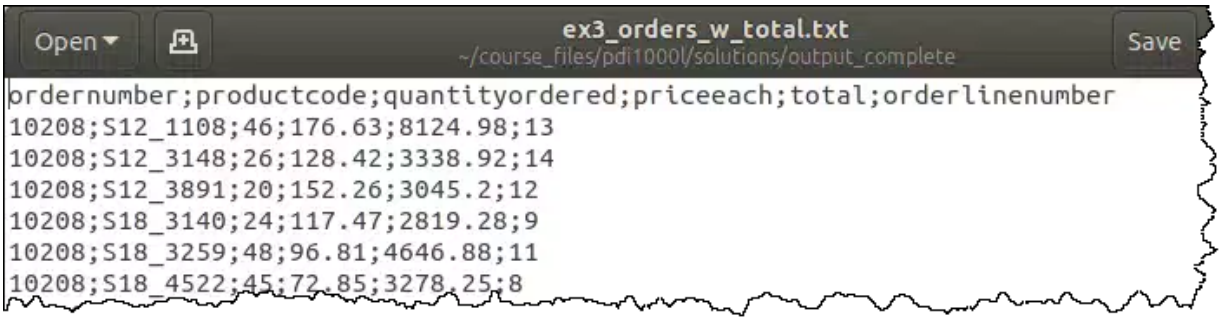
In this section of the exercise, you save the transformation with a new name, and then replace the **to-test_orderdetails (Table output)** step with a **Text file output** step.

1. To save the transformation with a new name, from the menu, select **File → Save as**.
2. Change the **File name** to `ex3_text_file_output`, and then click **OK**.
3. To delete the **to-test_orderdetails (Table output)** step, right-click **to-test_orderdetails**, and then select **Delete**.

4. To add a **Text file output** step to the transformation, on the **Design** tab:
    a. Click to expand the **Output** category.
    b. Drag the **Text file output** step from the **Design** tab to the canvas and drop it to the right of the **ti-orderdetails (Table input)** step.
5. Create a hop from the **ti-orderdetails** step to the **Text file output** step.
6. To configure the **Text file output** step, on the canvas, double-click the **Text file output** step.
7. To name the step, in the **Step name**, type: `Write order details to text file`
8. To use the `DIR_OUTPUT` variable:
    a. Click in the **Filename** field.
    b. Delete the word "file".
    c. On the keyboard, press **Ctrl-Space**.
    d. From the list of variables, double-click `DIR_OUTPUT`.
9. To append the filename to the variable, in the **Filename** field, after the variable, type `/ex3_orders_w_total`
10. To close the **Text file output** dialog, click **OK**.

*In this example, you do not need to configure the **Fields** tab. Leaving the **Fields** tab empty writes all fields in the stream to the file.*

11. Save the transformation.
12. To run the transformation:
    a. On the subtoolbar, click the **Run** button.
    b. In the **Run Options** dialog, click **Run**.
13. To verify the file was created, in Windows Explorer, navigate to `/home/pentaho/course_files/pdi1000l/data_files/output` and see that `ex3_orders_w_total.txt` exists.
14. Open the file and review the results.



15. Close the file and return to Spoon.

16. Close the `ex3_orders_w_total.txt` tab.

Solution Details

The solution to this exercise can be found at:

`/home/pentaho/course_files/pdi1000l/solutions/exercises`

Files:

`ex3_table_input_output.ktr`

`ex3_text_file_output.ktr`

Output (Part II):

`/home/pentaho/course_files/pdi1000l/solutions/output_complete/`

`ex3_orders_w_total.txt`

End of Exercise 3

Guided Demo 8: Filter Rows, Sort Rows, Excel Writer

In this guided demonstration, we create a transformation that reads customer-related information from a database table. Most of the customers are assigned a sales rep, but several customer records do not have a rep assigned. The transformation will find the customers that are missing an assigned sales rep and sort them by country and customer name. Then the customers are output to an Excel file.

Objectives

After completing this guided demonstration, you will be able to:

- Configure the **Filter rows** step with a condition that evaluates to `true` or `false`
- Configure the **Sort rows** step to sort the data in the stream
- Configure the **Microsoft Excel writer** step to output the data to an Excel file
- Recognize the usefulness of the **Dummy (do nothing)** step to capture data row metrics and clearly show the intention of the transformation's logic

Prerequisites

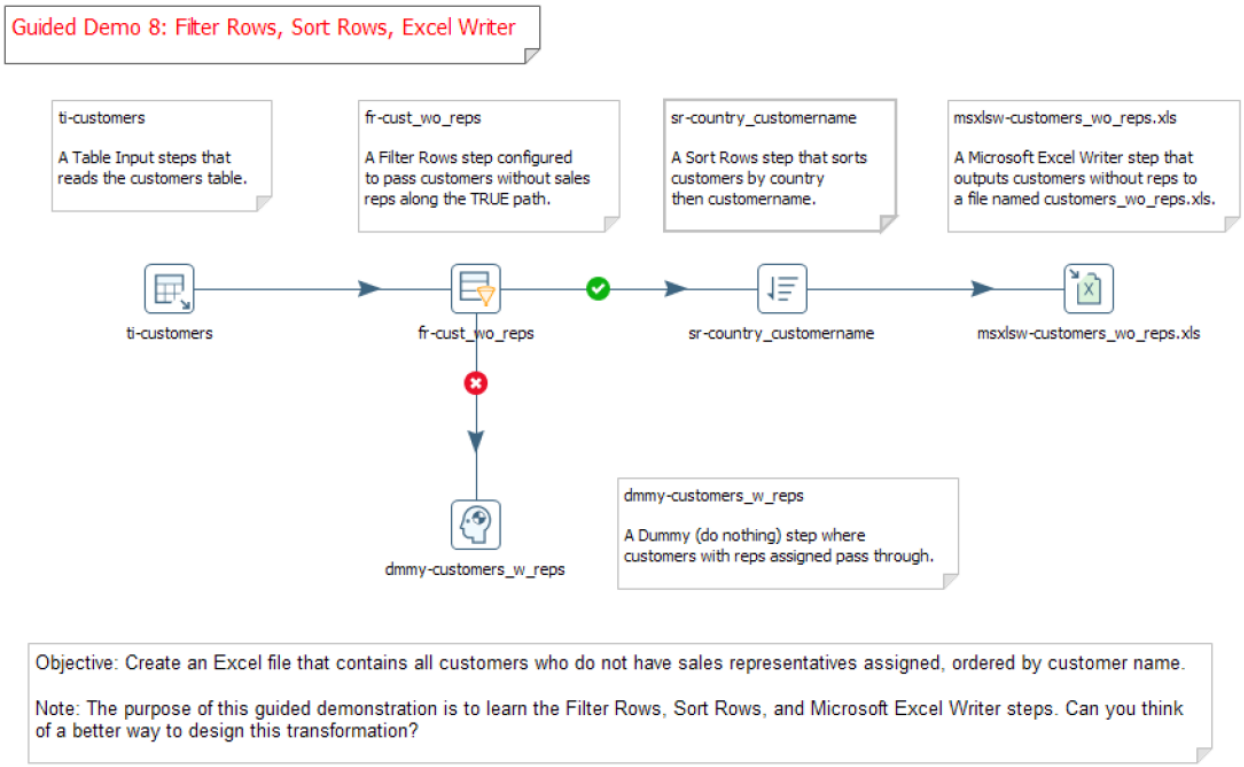
Prior to completing this exercise, you must complete Guided Demo 7: Connections and the Database Explorer. You must also have a `kettle.properties` variable defining the `DIR_OUTPUT` folder.

Steps Used

This guided demonstration uses the following steps:

- **Table input**
- **Filter rows**
- **Sort rows**
- **Microsoft Excel writer**
- **Dummy (do nothing)**

Transformation



Create a Transformation

In this section of the guided demonstration, we create a new transformation.

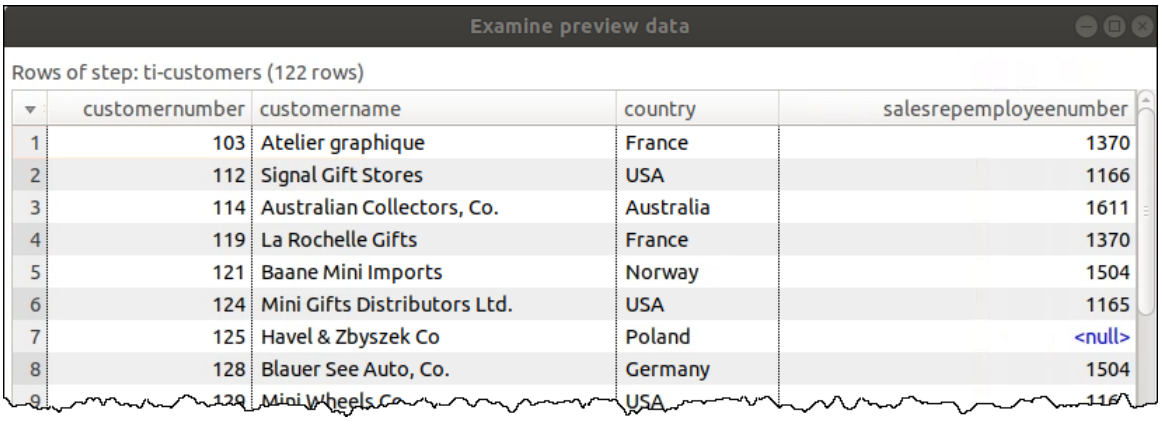
1. To create a new transformation, from the toolbar, click the **New file** button, and then select **Transformation**.
2. To save the transformation, on the toolbar, click the **Save** button.
3. To name the transformation and save it to the `PDI_Trn_Objects` folder, in the Save dialog:
    a. Expand the **Public** folder.
    b. Click to select the **PDI_Trn_Objects** folder.
    c. In the **File name**, type `gd8_filter_rows_sort_rows_excel_writer`
    d. Click Save.

Configuring the Table Input Step

In this section of the guided demonstration, we configure a **Table input** step to read in data from the `customers` table in the `pentaho_oltp` database.

1. From the **Input** category, drag a **Table input** step to the canvas.
2. Open the **Table input** step.
3. To name the step, in the **Step name**, type `ti-customers`
4. To select the database connection, from the **Connection** dropdown list, select **pentaho_oltp**.
5. To generate the SQL that obtains the data, in the **Table input** dialog, click the **Get SQL select statement** button.
6. To select the `customers` table from the `pentaho_oltp` database connection, in the **Database Explorer** dialog:
    a. Click to expand **pentaho_oltp**.
    b. Click to expand **Tables**.
    c. Click to select the **customers** table and click **OK**.
7. To include the field names in the SQL statement, in the **Question** dialog, click **Yes**.
8. To delete the columns not needed in this demonstration, in the SQL statement, remove all the columns except:
    a. `customernumber`
    b. `customername`
    c. `country`
    d. `salesrepemployeenumber`
9. To preview the data and confirm the SQL statement is correct, click **Preview**, and then click **OK**.

10. Verify the data generated is correct by comparing your data with the screenshot:



	customernumber	customername	country	salesrepemployeenumber
1	103	Atelier graphique	France	1370
2	112	Signal Gift Stores	USA	1166
3	114	Australian Collectors, Co.	Australia	1611
4	119	La Rochelle Gifts	France	1370
5	121	Baane Mini Imports	Norway	1504
6	124	Mini Gifts Distributors Ltd.	USA	1165
7	125	Havel & Zbyszek Co	Poland	<null>
8	128	Blauer See Auto, Co.	Germany	1504
9	129	Mini Wheels Co.	USA	1167

Notice that some of the customers do not have a `salesrepemployeenumber` *assigned.*

11. To close the **Examine preview data** dialog, click **Close**.

12. To close the **Table input** dialog, click **OK**.

13. Save the transformation.

Configuring the Filter Rows Step

In this section of the guided demonstration, we add and configure a **Filter rows** step to identify which customers do not have a sales rep assigned. Those that do not will continue through the stream.

1. To add a **Filter rows** step to the transformation, on the **Design** tab:

    a. Click to expand the **Flow** category.

    b. Drag the **Filter rows** step to the canvas and drop it to the right of the **ti-customers** step.

2. Create a hop from the **ti-customers** step to the **Filter rows** step.

3. Open the **Filter rows** step.

4. Set its **step name** property to `fr-cust_wo_reps`

5. To identify the field to be filtered on, in the **Filter rows** dialog:

    a. Click the first **<field>** box.

    b. In the **Fields** dialog, click **salesrepemployeenumber**.

c. Click **OK**.

6. To specify the function (condition), in the **Filter rows** dialog:

   a. Click the **Function** box.

   b. In the **Functions** dialog, click **IS NULL**.

   c. Click **OK**.

The `Send true data to step` *and* `Send false data to step` *values will automatically populate when we add hops from this step.*

7. To close the **Filter rows** dialog, click **OK**.

8. Save the transformation.

Configuring the Sort Rows Step

In this section of the guided demonstration, we add and configure a **Sort rows** step to sort the stream by country and customer name. We configure the hop from the **Filter rows** step to only send rows where the filter evaluates to `TRUE` to the **Sort rows** step.

1. To add a **Sort rows** step to the transformation, on the **Design** tab:

   a. Click to expand the **Transform** category.

   b. Drag the **Sort rows** step to the canvas and drop it to the right of the **Filter rows** step.

2. To create a hop that only sends data with a `TRUE` result, add a hop from the **Filter rows** step to the **Sort rows** step, and then from the context menu select **Result is TRUE**.

3. Open the **Sort rows** step.

4. Name the step `sr-country_customername`

5. To specify the first sort order, create a row in the **Fields** grid:

   a. From the **Fieldname** dropdown list, select **country**.

   b. From the **Ascending** dropdown list, select **Y**.

6. Create a second row in the **Fields** grid and configure it:

   a. From the **Fieldname** dropdown list select **customername**.

b. From the **Ascending** dropdown list, select **Y**.

For this guided demonstration, we use the default values for the other properties.

7. To close the **Sort rows** dialog, click **OK**.
8. Save the transformation.

Configuring the Microsoft Excel Writer Step

In this section of the guided demonstration, we configure the **Microsoft Excel writer** step to create an Excel (`.xls`) file that contains the customers without sales reps, ordered by customer name.

1. To add a **Microsoft Excel writer** step to the transformation, on the **Design** tab:
   a. Click to expand the **Output** category.
   b. Drag the **Microsoft Excel writer** step to the canvas and drop it to the right of the **Sort rows** step.
2. Create a hop from the **Sort rows** step to the **Microsoft Excel writer** step.
3. Open the **Microsoft Excel writer** step.
4. Name the step `msxlsw-customers_wo_reps.xls`
5. To specify the filename using the `DIR_OUTPUT` variable:
   a. Click in the **Filename** field.
   b. Delete the word "file".
   c. On the keyboard, press **Ctrl-Space**.
   d. From the list of variables, double-click **DIR_OUTPUT**.
6. To append the filename to the variable, in the **Filename** field, after the variable, type `/customers_wo_reps`
7. To configure the content to place into the Excel file, click the **Content** tab.
8. To automatically size the width of the Excel files columns to fit their contents, check the **Auto size columns** property.

9. To obtain the fields from the stream, click the **Get Fields** button.

*Scroll down to see the **Get Fields** button.*

10. To close the **Excel Writer Step** dialog, click **OK**.

11. Save the transformation.

Configuring the Dummy (do nothing) Step

In this section of the guided demonstration, we complete the transformation with a **Dummy (do nothing)** step with a hop from the **Filter rows** step, sending the data in the stream for which the filter evaluates to `FALSE`. Having the **Dummy (do nothing)** step allows us to easily capture how many customers do have sales reps assigned, by viewing its step metrics.

1. To add a **Dummy (do nothing)** step to the transformation, on the Design tab:

    a. Click to expand the **Flow** category.

    b. Drag the **Dummy (do nothing)** step to the canvas and drop it below the **Filter rows** step.

2. Name the step `dmmy-customers_w_reps`

3. To create a hop that only sends data with a `FALSE` result, add a hop from the **Filter rows** step to the **Dummy (do nothing)** step, and then from the context menu select **Result is FALSE**.

Notice the `True` and `False` indicators on the hops.

4. Save the transformation.

Run the Transformation

In this section of the guided demonstration, we run the transformation, and then review the output file.

1. To run the transformation:

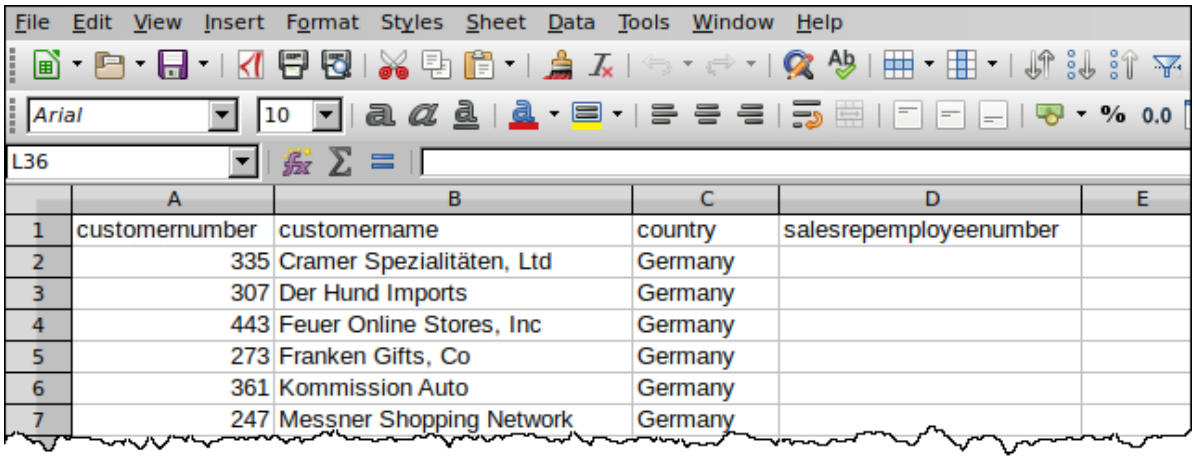
    a. On the subtoolbar, click the **Run** button.

    b. In the **Run Options** dialog, click **Run**.

2. On the **Step Metrics** tab, notice the **Output** field for the **Microsoft Excel writer** step indicates 22 lines were output to the Excel file.

3. To verify the file was created, in Windows Explorer, navigate to `/home/pentaho/course_files/pdi1000l/data_files/output` and see that the `customers_wo_reps.xls` file exists.

4. Open the file and review the data.



	A	B	C	D	E
1	customernumber	customername	country	salesrepemployeenumber	
2	335	Cramer Spezialitäten, Ltd	Germany		
3	307	Der Hund Imports	Germany		
4	443	Feuer Online Stores, Inc	Germany		
5	273	Franken Gifts, Co	Germany		
6	361	Kommission Auto	Germany		
7	247	Messner Shopping Network	Germany		

5. Close the file and return to Spoon.

6. Close the **gd8_filter_rows_sort_rows_excel_writer** tab.

Solution Details

The solution to this guided demo can be found at:

`/home/pentaho/course_files/pdi1000l/solutions/guided_demos`

File:

`gd8_filter_rows_sort_rows_excel_writer.ktr`

Output:

`/home/pentaho/course_files/pdi1000l/solutions/output_complete/customers_wo_reps.xls`

End of Guided Demo 8

Exercise 4: Insert/Update and Input with Parameters

*In the first part of this exercise, you create a transformation that gets data from the `ex3_orders_w_total.txt` file created in Exercise 3. Then you use the **Insert/update** step to load the data into a new table in the `pentaho_olap` database. Next, you modify the `ex3_orders_w_total.txt` file and re-run the transformation to update the database table with the new data.*

*In the second part of this exercise, you create a transformation that uses two parameters in a **Table input** step.*

Objectives

After completing this exercise, you will be able to:

- Configure the **Insert/update** step to either insert rows or update rows in a database table
- Configure the **Table input** step to load data based on parameter values

Prerequisites

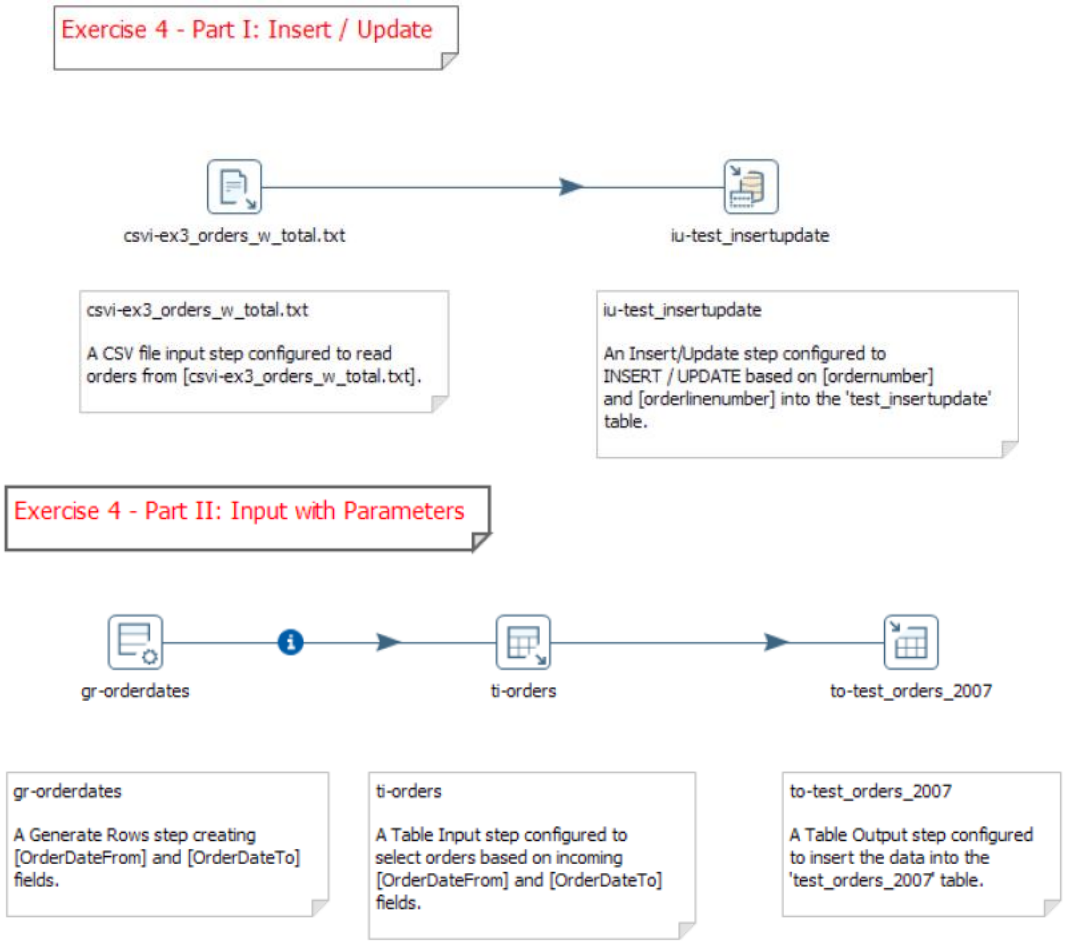
Prior to completing this exercise, you must complete Guided Demo 7: Connections and the Database Explorer. This exercise uses the text file created in Exercise 3.

Steps Used

This exercise uses the following steps:

- **CSV file input**
- **Insert/update**
- **Generate rows**
- **Table input**
- **Table output**

Transformations



Part I

In Part I of this exercise, you create a transformation that uses the **CSV file input** and **Insert/update** steps to capture updated data in the source file.

Create and Save the Transformation

In this section of the exercise, you create and save a new transformation.

1. To create a new transformation, on the toolbar, click the **New** file button, and then select **Transformation**.

2. To save the transformation, on the toolbar, click **Save**.

3. To name the transformation and save it to the `PDI_Trn_Objects` folder, in the **Save** dialog:

    a. Click **>** to expand the **Public** folder.

b. Click to select the **PDI_Trn_Objects** folder.

c. In the **File name**, type `ex4_csv_input_insert_update`

d. Click **Save**.

Configure the CSV File Input Step

In this section of the exercise, you add and configure the **CSV file input** step to bring in the data from the `ex3_text_file_output.txt` file.

1. To add a **CSV file input** step, on the **Design** tab, expand the **Input** category, and then drag the **CSV file input** step to the canvas.
2. To configure the **CSV file input** step, on the canvas, double-click the **CSV file input** step.
3. To use the `DIR_OUTPUT` variable:
   a. Click in the **Filename** field.
   b. On the keyboard, press **Ctrl-Space**.
   c. From the list of variables, double-click **DIR_OUTPUT**.
4. To append the filename to the variable, in the **Filename** field, after the variable, type `/ex3_orders_w_total.txt`
5. To change the separator, in the **Delimiter**, type a semicolon `;`
6. To populate the **Fields** grid:
   a. Click the **Get Fields** button.
   b. In the **Sample data** dialog, click **OK**.
7. To close the **CSV Input** dialog, click **OK**.

Configure the Insert/Update Step and Run the Transformation

In this section of the exercise, you configure an **Insert/update** step to insert the data from the stream into a new table in the `pentaho_olap` database. Since the table does not exist in the `pentaho_olap` database, you use the SQL helper button to generate SQL to create the table. You then run the transformation.

1. To add an **Insert/update** step to the transformation, on the **Design** tab:
   a. Click to expand the **Output** category.

b. Drag the **Insert/update** step to the canvas and drop it to the right of the **CSV file input** step.

2. Create a hop from the **CSV file input** step to the **Insert/update** step and select **Main output of step**.
3. To configure the **Insert/update** step, on the canvas, double-click the **Insert/update** step.
4. In the **Insert/update** dialog, type or choose:

Field	Value
Step name	`iu-test_insertupdate`
Connection	`pentaho_olap`
Target schema	[leave blank]
Target table	`test_insertupdate`
Commit size	`100`

5. To get the list of fields, click the **Get fields** button.
6. To include only the key fields needed, delete all but the following key fields:
   a. `ordernumber`
   b. `orderlinenumber`

You can use multi-select to select multiple lines.

7. To get the list of update fields, click the **Get update fields** button. Keep all the default values for this exercise.
8. To generate the SQL statement to create the table, in the **Insert/update** dialog, click the **SQL** button.
9. Verify the syntax of the SQL statement, and then to execute the SQL statement, in the **Simple SQL editor** dialog, click **Execute**.
10. Verify the SQL statement executed, and then in the **Results of the SQL statements** dialog, click **OK**.
11. To close **Simple SQL editor** dialog, click **Close**.
12. To close the **Insert/update** dialog, in the **Insert/update** dialog, click OK.
13. Save the transformation.

14. To run the transformation:

    a. On the subtoolbar, click the **Run** button.

    b. In the **Run Options** dialog, click **Run**.

15. On the **Step Metrics** tab, notice that 23,640 rows were written to the table.

16. Run the transformation again and notice that no rows were output because there were no updates.

Edit the Text File and Run the Transformation

In this section of the exercise, you modify the `ex3_orders_w_total.txt` file, and then re-run the transformation to see that the changed rows get updated.

1. To edit the `ex3_orders_w_total.txt` file:

    a. In Windows Explorer, navigate to `/home/pentaho/course_files/pdi1000l/data_files/output`

    b. Double-click the `ex3_orders_w_total.txt` file.

2. To add a new row:

    a. Click at the end of the header row.

    b. Press **Enter**.

    c. Type `10208;s12_1108;48;176.6;8125;99`

The data you type is identical to the first row except for the last two digits, so you can copy and paste the first row.

3. To edit the third entry, change the `quantityordered` from `26` to `36`.

4. Save the changes, close the `ex3_orders_w_total.txt` file, and then return to Spoon.

5. To run the transformation:

    a. On the subtoolbar, click the **Run** button.

    b. In the **Run Options** dialog, click **Run**.

6. On the **Step Metrics** tab, notice that one new row was output to the table for the row you added to the text file, and one row was updated for the row you changed in the text file.

7. Run the transformation again and notice that no rows were output or updated.

8. Close the **ex4_csv_input_insert_update.ktr** tab.

Part II

In Part II of this exercise, you create a transformation that uses the **Table input** step to load data based on a parameter value. This procedure is commonly used to load only changed (or delta) data into a data warehouse.

Create the Transformation and Configure the Generate Rows Step

In this section of the exercise, you create a new transformation and configure a **Generate rows** step with dates that will be used as parameter values.

1. Create a new transformation named `ex4_table_input_parameter` and save it to the `PDI_Trn_Objects` folder on the repository.

2. From the **Input** category, drag a **Generate rows** step to the canvas.

3. To configure the **Generate rows** step, on the canvas, double-click the **Generate rows** step and name it `gr-orderdates`

4. To only add one row, in the **Limit** field, type `1`

5. To configure the **Fields** grid, add two rows as shown:

Name	Type	Format	Value
OrderDateFrom	`Date`	`yyyy-MM-dd`	`2007-01-01`
OrderDateTo	`Date`	`yyyy-MM-dd`	`2007-12-31`

6. To close the **Generate rows** dialog, click OK.

7. Save the transformation.

Configure the Table Input Step

In this section of the exercise, you configure the **Table input** step to read data from the orders table in the `pentaho_oltp` database using parameters to specify a date range.

1. From the **Input** category, drag a **Table input** step to the canvas, and drop it to the right of the **Generate rows** step.

2. Create a hop from the **Generate rows** step to the **Table input** step.

3. To configure the **Table input** step, on the canvas, double-click the **Table input** step and name it `ti-orders`

4. To select the database connection, from the **Connection** dropdown list, select **pentaho_oltp**.

5. To generate the SQL that obtains the data, in the **Table input** dialog, click the **Get SQL select statement** button.

6. To select the customers table from the `pentaho_oltp` database connection, in the **Database Explorer** dialog:

    a. Click to expand **pentaho_oltp**.

    b. Click to expand **Tables**.

    c. Click to select the **orders** table.

    d. Click **OK**.

7. To include the field names in the SQL statement, in the **Question?** dialog, click **Yes**.

8. To add a `where` clause with parameters, at the end of the SQL statement, add:
`WHERE orderdate>=? AND orderdate<=?`

9. To specify the step containing the parameter values, from the **Insert data from step** dropdown list, select **gr-orderdates**.

10. To close the **Table input** dialog, click **OK**.

11. Save the transformation.

Configure the Table Output Step and Run the Transformation

In this section of the exercise, you configure the **Table output** step to write the data to a new table in the `pentaho_olap` database. Since the table does not exist in the `pentaho_olap` database, you use the SQL helper button to generate SQL to create the table. You then run the transformation and view the results in Database Explorer.

1. From the **Output** category, drag a **Table output** step to the canvas, and drop it to the right of the **Table input** step.

2. Create a hop from the **Table input** step to the **Table output** step.

3. To configure the **Table output** step, on the canvas, double-click the **Table output** step.

4. In the **Table output** dialog, type or select the following:

Field	Value
Step name	`to-test_orders_2007`
Connection	`pentaho_olap`
Target schema	[leave blank]
Target table	`test_orders_2007`
Commit size	`1000`
Truncate table	[checked]

5. To generate the SQL statement to create the table, in the **Table output** dialog, click the SQL button.
6. Verify the syntax of the SQL statement, and then to execute the SQL statement, in the **Simple SQL editor** dialog, click **Execute**.
7. Verify the SQL statement executed, and then in the **Results of the SQL statements** dialog, click **OK**.
8. To close **Simple SQL editor** dialog, click **Close**.
9. To close the **Table output** dialog, click **OK**.
10. Save the transformation.
11. To run the transformation:
    a. On the subtoolbar, click the **Run** button.
    b. In the **Run Options** dialog, click **Run**.
12. To view the results, on the **View** tab, click to expand **Database connections**.
13. Right-click **pentaho_olap**, and then from the context menu, click **Explore**.
14. To preview the `customers` table, in the **Database Explorer** window:
    a. Click to expand **pentaho_olap**.
    b. Click to expand **Tables**.
    c. Right-click **test_orders_2007**.
    d. Select **Preview first 100**.

15. Examine the first 100 rows in the table (specifically the `orderdate`), and then in the **Examine preview data** dialog, click **Close**.
16. To close **Database Explorer**, click **OK**.
17. Close the **ex4_table_input_parameter** tab.

Solution Details

The solution to this exercise can be found at:

`/home/pentaho/course_files/pdi10001/solutions/exercises`

File:

`ex4_csv_file_input_insert_update.ktr`

`ex4_table_input_parameter.ktr`

End of Exercise 4

Exercise 5: Parallel Processing

*In the first part of this exercise, you modify the Hello World transformation (from Guided Demo 2) to generate 1,000,000 rows. You then create a version of the transformation with five copies of the **Generate rows** step to evaluate the impact on performance. Finally, you create a version of the transformation that distributes the rows to evaluate the impact on performance.*

In the second part of this exercise, you create and run a transformation that uses parallel processing.

This exercise is used to help you understand concepts by creating demonstration transformations. Therefore, best practice naming conventions are not used.

Objectives

After completing this exercise, you will be able to:

- Copy and distribute data
- Start multiple copies of a step
- Send data to multiple output steps

Prerequisites

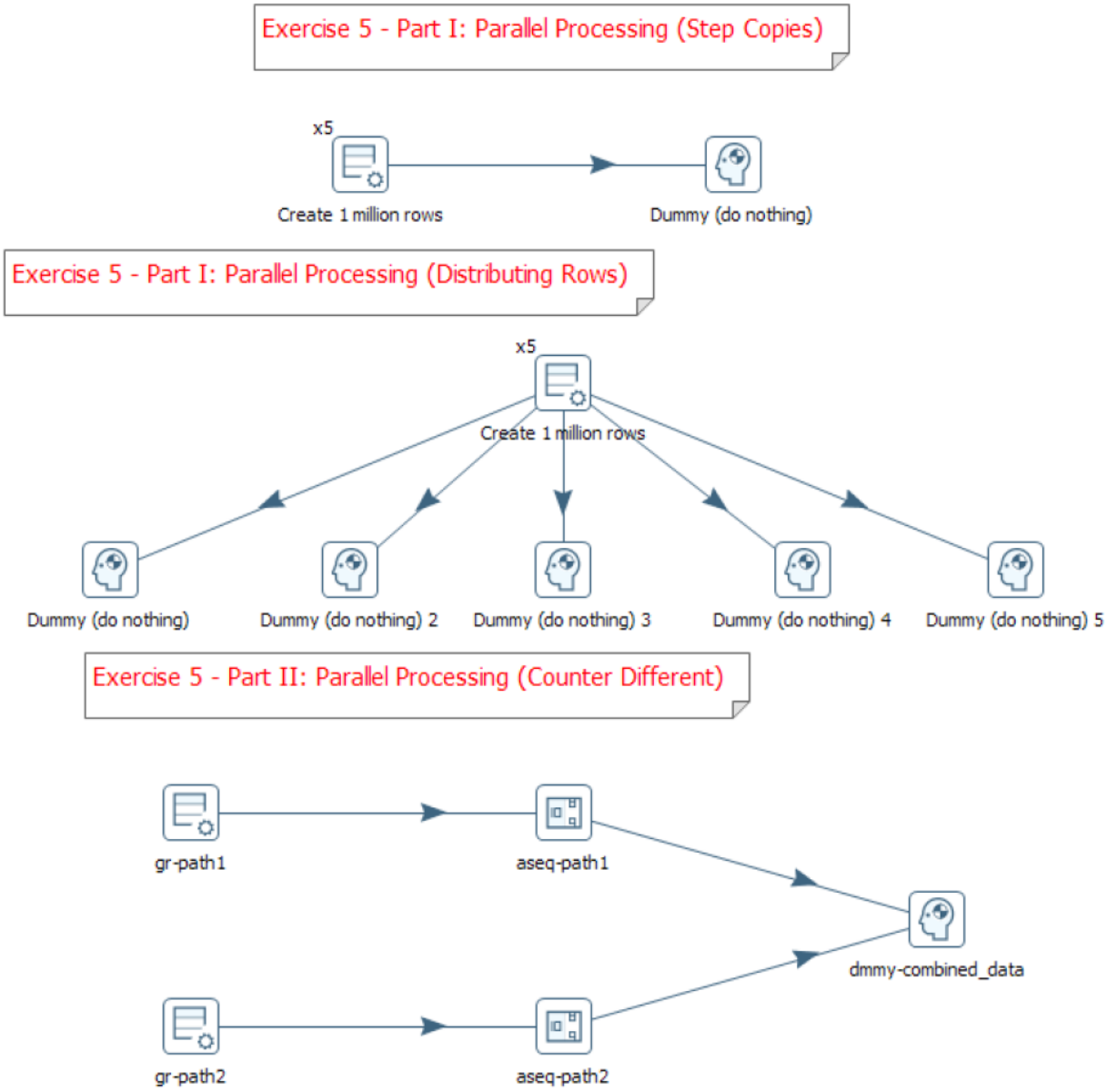
This guided demonstration uses the `gd2_hello_world` transformation created in Guided Demo 2. If you did not complete Guided Demo 2, you must have access to the course solution files.

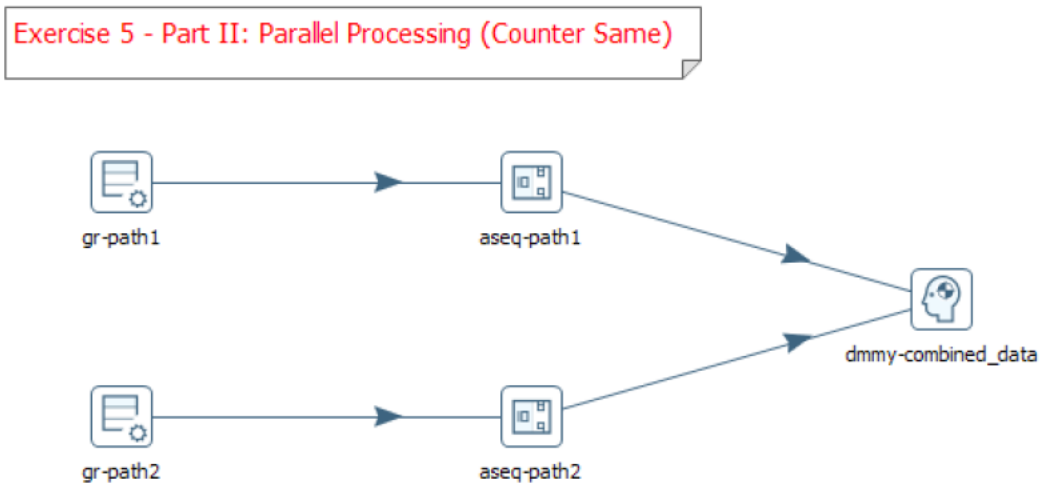
Steps Used

This exercise uses the following steps:

- **Generate rows**
- **Dummy (do nothing)**
- **Add sequence**

Transformations





Part I

In Part I of this exercise, you modify the Hello World transformation (from Guided Demo 2) to generate 1,000,000 rows. You then create a version of the transformation with five copies of the **Generate rows** step to evaluate the impact on performance. Finally, you create a version of the transformation that distributes the rows to evaluate the impact on performance.

Modify the Hello World Transformation

In this section of the exercise, you modify the `gd2_hello_world` transformation to generate 1,000,000 rows, and then run it.

1. To open the `gd2_hello_world` transformation:
    a. From the menu, select **File → Open**.
    b. Navigate to the **Public → PDI_Trn_Objects** folder.
    c. Click to select the **gd2_hello_world** transformation.
    d. Click **Open**.

If necessary, you can import the transformation from the /home/pentaho/course_files/pdi1000l/solutions/guided_demos folder.

2. Delete the Add constants step.
3. Create a hop between the Generate rows and Dummy (do-nothing) steps.
4. Save the transformation as `ex5_helloworld_parallel_processing` in the `PDI_Trn_Objects` folder.

5. To modify the **gr-hello_worlds** step:
    a. Double-click the **gr-hello_worlds** step.
    b. Change the **Step name** to `Create 1 million rows`.
    c. Change the **Limit** to `1000000`.
    d. Click **OK**.
6. Save the transformation.
7. To run the transformation:
    a. On the subtoolbar, click the **Run** button.
    b. In the **Run Options** dialog, click **Run**.

Create Five Copies of the Generate Rows Step

In this section of the exercise, we create five copies of the **Create 1 million rows** step.

1. Save the transformation as `ex5_helloworld_parallel_processing_5copies` in the `PDI_Trn_Objects` folder.
2. To change the number of copies of the **Create 1 million rows** step, right-click the **Create 1 million rows** step, and then from the context menu, select **Change Number of Copies to Start**.
3. To set the number of copies to `5`, in the **Nr of copies of step** dialog, change the **Number of Copies** to `5`, and then click **OK**.

*Enlarge the **Nr of copies of step** dialog to see the **OK** button.*

4. Notice the `x5` indicator at the top left of the **Create 1 million rows** step.
5. Save the transformation.
6. To run the transformation:
    a. On the subtoolbar, click the **Run** button.
    b. In the **Run Options** dialog, click **Run**.

7. Click the **Step Metrics** tab and notice that 5,000,000 rows were written to the stream: 1,000,000 rows for each step copy.

Distribute the Rows in the Stream

In this section of the exercise, you distribute the rows from the **Create 1 million rows** step to five copies of the **Dummy (do nothing)** step.

1. Save the transformation as `ex5_helloworld_parallel_processing_distribute` in the `PDI_Trn_Objects` folder.
2. To create four additional **Dummy (do nothing)** steps:
    a. Right-click the **Dummy (do nothing)** step.
    b. From the context menu, select **Copy**.
    c. Right-click the canvas.
    d. From the context menu, select **Paste**.
    e. Repeat the previous two steps to create a total of five **Dummy (do nothing)** steps.
3. Create a hop from the **Create 1 million rows** step to one of the new **Dummy (do nothing)** steps.
4. To distribute the rows among the five **Dummy (do nothing)** steps, in the **Warning** dialog, click the **Distribute** button.
5. Create hops from the **Create 1 million rows** step to the remaining **Dummy (do nothing)** steps.
6. Save the transformation.
7. To run the transformation:
    a. On the subtoolbar, click the **Run** button.
    b. In the **Run Options** dialog, click **Run**.

    Notice that each of the **Dummy (do nothing)** steps reads **1,000,000 rows**.
8. Close the **ex5_helloworld_parallel_processing_distribute** tab.

Part II

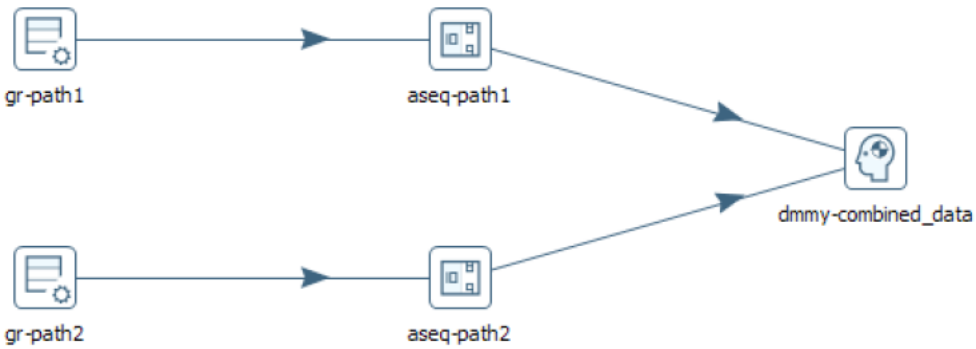
In Part II of this exercise, you create and run a transformation that demonstrates how parallel processing

behaves when using a named counter in the **Add sequence** step under different scenarios.

Create the Transformation and Add the Steps and Hops

In this section of the exercise, you create a new transformation, add two **Generate rows** steps, two **Add sequence** steps, and a **Dummy (do nothing)** step, and create all the hops.

1. Create a new transformation named `ex5_parallel_processing_aseq_counter_different` and save it to the `PDI_Trn_Objects` folder on the repository.
2. From the **Input** category, drag two **Generate rows** steps to the canvas.
3. From the **Transform** category, drag two **Add sequence** steps to the canvas.
4. From the **Flow** category, drag a **Dummy (do nothing)** step to the canvas.
5. Add hops as follows:
    a. From the **Generate rows** step to the **Add sequence** step.
    b. From the **Generate rows 2** step to the **Add sequence 2** step.
    c. From the **Add sequence** step to the **Dummy (do nothing)** step.
    d. From the **Add sequence 2** step to the **Dummy (do nothing)** step.
    e. Then, name the steps as shown:



6. Save the transformation.

Configure the Steps and Preview the Results

In this section of the exercise, you configure each **Generate rows** step to add 1000 rows, and then configure each **Add sequence** step to add a unique counter. You then preview the stream to see that there are duplicate values for the counter. Finally, you change each **Add sequence** step to assign a common counter name and

verify the data in the stream.

1. Configure the **gr-path1** and **gr-path2** steps and set the **Limit** to `1000`. For this exercise, it is not necessary to add fields.
2. To configure the **Add sequence** step, on the canvas, double-click the **aseq-path1** step.
3. To set the Counter name, in the **Counter name (optional)**, type: `counter1` and then click **OK**.

*The **Name of value** property adds a new field to the stream named* `valuename`

4. Repeat the process to set the **Counter name (optional)** in the **aseq-path2** step to `counter2`.
5. Save the transformation.
6. To preview the **dmmy-combined_data** step to see the stream:
    a. Right-click the **dmmy-combined_data** step.
    b. From the context menu, select **Preview**.
    c. Change the **Number of rows to retrieve** to `2000`.
    d. In the **Transformation debug** dialog, click **Quick Launch**.
7. To sort the results, click the **valuename** column header.

    Notice the duplicate values in the **valuename** column. This is caused by assigning different counter names in the **Add sequence** steps. Using different counter names within a transformation's **Add sequence** steps provides the capability to keep your sequences separate (and automatically tracked by name) from one another.
8. To close the **Examine preview data** dialog, click **Close**.
9. To close the **Select the preview step** dialog, click **Close**.
10. To save the transformation with another name, click **File → Save as...** and then type `ex5_parallel_processing_aseq_counter_same` and click **Save**.
11. To update the **aseq-path1** step, on the canvas, double-click the first **aseq-path1** step.
12. To set the Counter name, in the **Counter name (optional)**, type `counter_same` and then click **OK**.
13. Repeat the steps to set the **Counter name (optional)** in the **aseq-path2** step to `counter_same`.

14. Save the transformation.
15. To preview the **dmmy-combined_data** step to see the stream:
    a. Right-click the **dmmy-combined_data** step.
    b. From the context menu, select **Preview**.
    c. Change the **Number of rows to retrieve** to `2000`.
    d. In the **Transformation debug** dialog, click **Quick Launch**.
16. To sort the results, click the **valuename** column header.

    Notice the **valuename** column is now consecutive from 1 to 2000. This is because there is now only one sequence counter for the entire transformation. This feature provides the capability to assign a single sequence across a transformation regardless of the stream it is in.
17. To close the **Examine preview data** dialog, click **Close**.
18. To close the **Select the preview step** dialog, click **Close**.
19. Close the **ex5_parallel_processing_aseq_counter_same** tab.

Solution Details

The solution to this exercise can be found at:

`/home/pentaho/course_files/pdi1000l/solutions/exercises`

Files:

`ex5_helloworld_parallel_processing_5copies.ktr`

`ex5_helloworld_parallel_processing_distribute.ktr`

`ex5_paralell_processing_aseq_counter_different.ktr`

`ex5_paralell_processing_aseq_counter_same.ktr`

End of Exercise 5

Guided Demo 9: Choosing Adequate Sample Size for Get Fields

*In this guided demonstration, we create a transformation that loads data from a file and outputs it to a database table. However, when we configure the **CSV file input** step, we use an inadequate sample size to demonstrate the impact incorrect metadata has when using Spoon to create database table DDL.*

Note: Transformation step naming standards are not used in this guided demonstration because the objective is to quickly demonstrate development technique instead of creating a transformation for actual use.

Objectives

After completing this guided demonstration, you will be able to:

- Describe the importance of properly selecting the sample size for **CSV file input** and **Text file input** steps
- Identify transformation errors caused by an improper sample size

Prerequisites

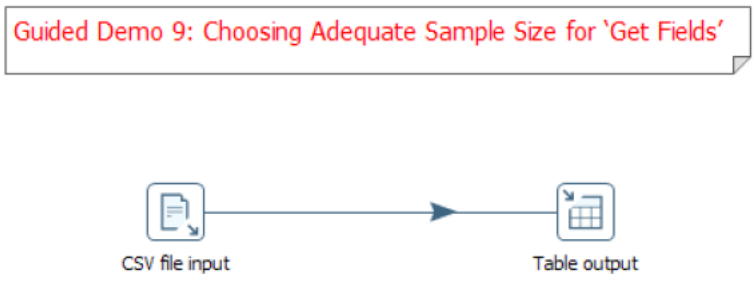
Prior to completing this exercise, complete Guided Demo 7: Connections and the Database Explorer. You must also have access to the sample files within the Pentaho program files.

Steps Used

This guided demonstration uses the following steps:

- **CSV file input**
- **Table output**

Transformation



Creating the Transformation

In this section of the guided demonstration, we create a new transformation and save it to the repository.

1. Create a new transformation named `gd9_preview_sample_size`.
2. Save it to the `PDI_Trn_Objects` folder on the repository.

Configuring the CSV File Input Step

In this section of the guided demonstration, configure the **CSV file input** step to read the `sales_data.csv` sample file, and configure the fields using the proper sample size.

1. To add a **CSV file input** step, from the **Input** category, drag the **CSV file input** step to the canvas.
2. To configure the **CSV file input** step, on the canvas, double-click the **CSV file input** step.
3. To use the sample `sales_data` file:
    a. To the right of the **Filename** field, click **Browse**.
    b. Navigate to the folder `/home/pentaho/Pentaho/design-tools/data-integration/samples/transformations/files`
    c. Double-click to select the `sales_data.csv` file.
4. To populate the **Fields** grid:
    a. Click the **Get Fields** button.
    b. In the **Sample size** dialog, change the value to `0`.
    c. Check the **Show sample summary** checkbox.
    d. In the **Sample size** dialog, click **OK**.

If you are reading a very large file, it may take a long time to read all its contents. In that case, it would be better to determine the proper field lengths beforehand or determine a sample size that is smaller than the entire file, but large enough to encapsulate the largest field values.

5. In the **Scan results** dialog, notice that PDI scanned 2823 lines, and then click **Close**.
6. In the **Fields** grid, notice the length for the **PRODUCTLINE** field is correctly set to `16`.

Using an Improper Sample Size

In this section of the guided demonstration, we configure the fields for the **CSV file input** step again, but this time using an inadequate sample size.

1. To populate the **Fields** grid:
    a. Click the **Get Fields** button.
    b. Click **OK** to close the No new fields were found dialog.
    c. In the **Sample size** dialog, keep the default value of `100`.
    d. Check the **Show sample summary** checkbox.
    e. In the **Sample data** dialog, click **OK**.
2. In the **Scan results** dialog, notice that PDI scanned 100 lines, and then click **Close**.
3. In the **Fields** grid, notice the length for the **PRODUCTLINE** field is now set to `12`, which is incorrect.

    In addition to **productline** being incorrect, several other fields are incorrect. We will keep these incorrect values and continue creating the transformation to see the impact of these errors.
4. To close the **CSV Input** dialog, click **OK**.

Configure the Table Output Step

In this section of the guided demonstration, we configure the **Table output** step using the metadata from the stream (which is incorrect for **productline** and several other fields).

1. From the **Output** category, drag a **Table output** step to the canvas, and drop it to the right of the **CSV file input** step.
2. Create a hop from the **CSV file input** step to the **Table output** step and select **Main output of step**.
3. To configure the **Table output** step, on the canvas, double-click the **Table output** step.
4. To configure the **Table output** step, in the **Table output** dialog, type or select the following:

Field	Value
Connection	`pentaho_olap`
Target table	`SampleSizeTest`

5. To generate the SQL statement to create the table, in the **Table Output** dialog, click the **SQL** button.
6. Notice the field lengths used to create the new table are based on the metadata coming from the **CSV file input** step.
7. To execute the SQL statement, in the **Simple SQL editor** dialog, click **Execute**.
8. Verify the SQL statement executed, and then in the **Results of the SQL statements** dialog, click **OK**.
9. To close the **Simple SQL editor** dialog, click **Close**.
10. To close the **Table Output** dialog, click **OK**.
11. Save the transformation.

Running the Transformation

In this section of the guided demonstration, we run the transformation and view the errors that occur when the transformation attempts to load data into the database table.

1. To run the transformation:
    a. On the subtoolbar, click the **Run** button.
    b. In the **Run Options** dialog, click **Run**, and then click the **Step Metrics** tab.

    Notice the **Table Output** step is red, indicating there are errors.
2. To view only the error lines in the log, on the **Logging** tab toolbar, click the **Show error lines** button.
3. Review the error lines and notice the error stating **ERROR: value too long for type character varying(30)**. This means data for one of the fields is longer than the table column is wide.
4. To close the **Error lines** dialog, click **OK**.
5. Close the **gd9_preview_sample_size** tab.

Solution Details

The solution to this guided demonstration can be found at:

`/home/pentaho/course_files/pdi1000l/solutions/guided_demos`

File:

`gd9_preview_sample_size.ktr`

End of Guided Demo 9

Exercise 6: Lookups and Field Layout Formatting

In this exercise, you create a transformation that identifies customers that are missing postal codes, obtains the missing postal codes from a file, and then loads the complete customer data into a database table that can be used to generate mailing labels. You use a Stream Lookup step to obtain the missing postal codes from another data stream, and a Select Values step to align the updated customer fields with the ones that already had a postal code.

Objectives

After completing this exercise, you will be able to:

- Configure the **Stream lookup** step to obtain data from a file
- Merge data from different streams
- Configure the **Select values** step to change the name and order of a field

Prerequisites

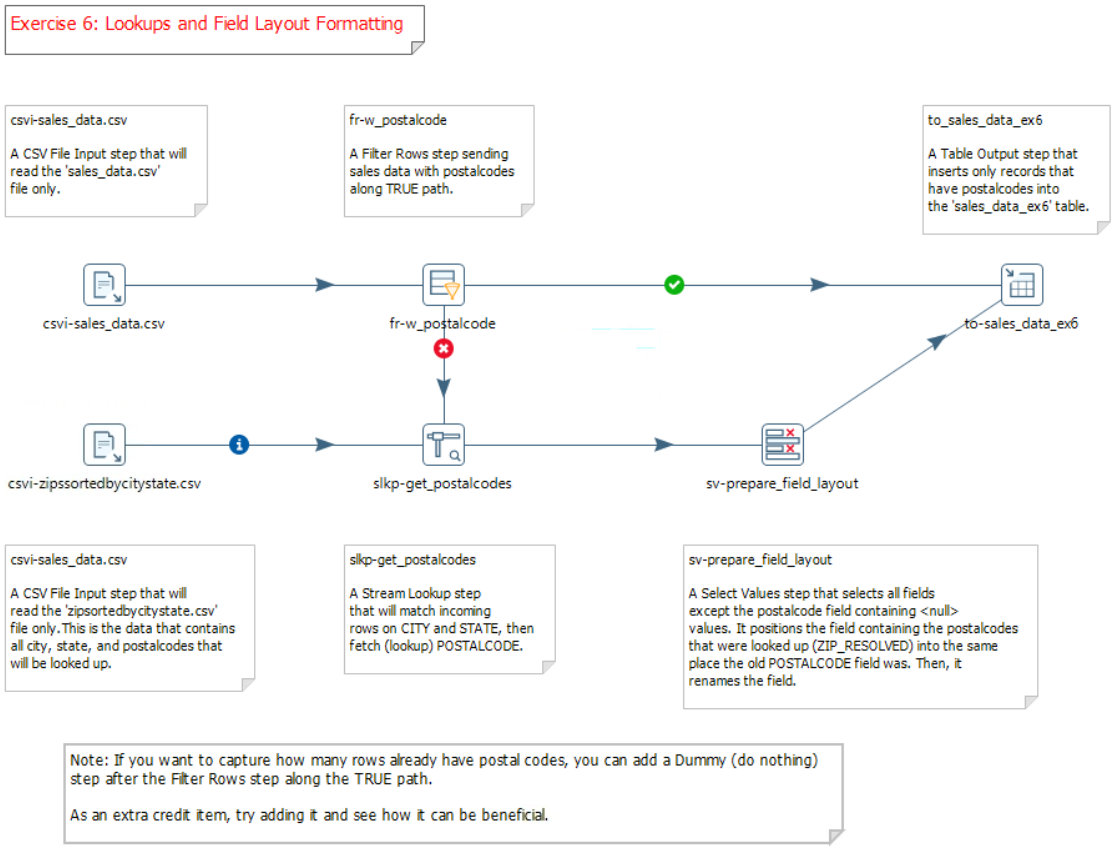
Prior to completing this exercise, you must complete Guided Demo 7: Connections and the Database Explorer. You must also have access to the samples in the Pentaho Data Integration client program files.

Steps Used

This exercise uses the following steps:

- **CSV file input**
- **Filter rows**
- **Table output**
- **Stream lookup**
- **Select values**

Transformation



Create a New Transformation and Configure the CSV File Input Step

In this section of the exercise, you create a new transformation and configure the **CSV input** step to obtain data from the `sales_data.csv sample` file.

1. Create a new transformation named `ex6_update_zip` and save it to the `PDI_Trn_Objects` folder on the repository.
2. To add a **CSV file input** step, from the **Input** category, drag the **CSV file input** step to the canvas.
3. To configure the **CSV file input** step, on the canvas, double-click the **CSV file input** step.
4. To name the step, in the **Step name**, type `csvi-sales_data.csv`
5. To use the sample `sales_data.csv` file:
    a. To the right of the **Filename** field, click **Browse**.

b. Navigate to:
`/home/pentaho/Pentaho/design-tools/data-integration/samples/transformations/files`

c. Double-click to select the `sales_data.csv` file.

6. To populate the **Fields** grid:

a. Click the **Get Fields** button.

b. In the **Sample data** dialog, change the value to `0`.

c. Click **OK**.

7. To preview the data and confirm it is configured properly, click **Preview**, and then click **OK**.

8. Verify the file is being read and notice that many customers do not have a **POSTALCODE**. Then, click **Close**.

9. To close the **CSV file input** dialog, click **OK**.

10. Save the transformation.

Configure the Filter Rows Step

In this section of the exercise, you configure the **Filter rows** step to identify the customers who have a postal code.

1. To add a **Filter rows** step to the transformation, from the **Flow** category, drag the **Filter rows** step to the canvas, and drop it to the right of the **csvi-sales_data.csv** step.

2. Create a hop from the **csvi-sales_data.csv** step to the **Filter rows** step and select **Main output of step**.

3. To configure the **Filter rows** step, on the canvas, double-click the **Filter rows** step.

4. To name the step, in the **Step name**, type `fr-w_postalcode`

5. To identify the field to be filtered on, in the **Filter rows** dialog:

a. Click the first **<field>** box.

b. In the **Fields** dialog, click **POSTALCODE**.

c. Click **OK**.

6. To specify the **function** (condition), in the **Filter rows** dialog:

   a. Click the **function** box.

   b. In the **Functions** dialog, click **IS NOT NULL**.

   c. Click **OK**.

7. To close the **Filter rows** dialog, click **OK**.

8. Save the transformation.

Configure the Table Output Step

In this section of the exercise, you configure the **Table output** step to add a new table to the `pentaho_olap` database containing the sales data for customers who have a postal code (the **Filter rows** step evaluated to `TRUE`).

1. To add a **Table output** step to the transformation, from the **Output** category, drag the **Table output** step to the canvas, and drop it to the right of the **fr-w_postalcode** step.

2. To create a hop that only sends data with a `TRUE` result, add a hop from the **fr-w_postalcode** step to the **Table output** step, and then from the context menu select **Result is TRUE**.

3. To configure the **Table output** step, on the canvas, double-click the **Table output** step.

4. To name the step, in the **Step name**, type `to-sales_data_ex6`

5. To configure the **Table output** step, in the **Table output** dialog, type or select the following:

Field	Value
Connection	`pentaho_olap`
Target table	`sales_data_ex6`
Truncate table	[checked]

6. To generate the SQL statement to create the table, in the **Table output** dialog, click the **SQL** button.

7. To execute the SQL statement, in the **Simple SQL editor** dialog, click **Execute**.

8. Verify the SQL statement executed, and then in the **Results of the SQL statements** dialog, click **OK**.

9. To close the **Simple SQL editor** dialog, click **Close**.
10. To close the **Table output** dialog, click **OK**.
11. Save the transformation.

Configure the Second CSV File Input Step

In this section of the exercise, you configure a second **CSV file input** step to read in a file containing all postal codes.

1. To add a **CSV file input** step, from the **Input** category, drag the **CSV file input** step to the canvas, and drop it below the **csvi-sales_data.csv** step.
2. To configure the **CSV file input** step, on the canvas, double-click the **CSV file input** step.
3. To name the step, in the **Step name**, type `csvi-zipsortedbycitystate.csv`
4. To use configure the file to read:
    a. To the right of the **Filename** field, click **Browse**.
    b. Navigate to: `/home/pentaho/Pentaho/design-tools/data-integration/samples/transformations/files`
    c. Double-click to select the `Zipssortedbycitystate.csv` file.
5. To populate the **Fields** grid:
    a. Click the **Get Fields** button.
    b. In the **Sample size** dialog, change the value to `0`.
    c. Click **OK**.
6. In the **Fields** grid, rename the **POSTALCODE** field to **ZIP_RESOLVED**, and change the **Type** to **String**.
7. To preview the data and confirm it is configured properly, click **Preview**, and then click **OK**.
8. Verify the file is being read, and then to close the **Examine preview data** dialog, click **Close**.
9. To close the **CSV file input** dialog, click **OK**.

10. Save the transformation.

Configure the Stream Lookup Step

In this section of the exercise, you configure the **Stream lookup** step to obtain/look up the missing postal codes from the **csvi-zipsortedbycitystate.csv** step and add them to the stream.

1. To add a **Stream lookup** step to the transformation, from the **Lookup** category, drag the **Stream lookup** step to the canvas, and drop it to the right of the **csvi-zipsortedbycitystate.csv** step.
2. Create a hop from the **csvi-zipsortedbycitystate.csv** step to the **Stream lookup** step and select **Main output of step**.
3. To create a hop that only sends data with a `FALSE` result, add a hop from the **fr-w_postalcode** step to the **Stream lookup** step, and then from the context menu select **Result is FALSE**.
4. To configure the **Stream lookup** step, on the canvas, double-click the **Stream lookup** step.
5. To name the step, in the **Step name**, type `slkp-get_postalcodes`
6. From the **Lookup** step dropdown list, select **csvi-zipsortedbycitystate.csv**.
7. To define the key fields to lookup, in the **Field** and **Lookup Field** fields, select the following from the dropdown lists:

Field	LookupField
CITY	CITY
STATE	STATE

8. To obtain the lookup fields from the stream, click the **Get lookup fields** button.
9. To delete **CITY** and **STATE**, right-click on each line number and then select **Delete selected lines**.
10. To close the **Stream Lookup** dialog, click **OK**.
11. To preview the **slkp-get_postalcodes** step:
    a. On the canvas, right-click the **slkp-get_postalcodes** step.
    b. From the context menu, select **Preview**.
    c. Click the **Quick Launch** button.

12. Scroll to the right to see the new field, **ZIP_RESOLVED**, has been added to the stream.
13. To close the **Examine preview data** dialog, click **Close**.
14. To close the **Select the preview step** dialog, click **Close**.
15. Save the transformation.

Configure the Select Values Step and Run the Transformation

In this section of the exercise, you configure the **Select values** step to ensure that the data in the stream coming out of the **slkp-get_postalcodes** step matches the structure of the data in the stream coming out of the **fr-w_postalcode** step. The two streams must match in order to write the data to a database table. You then run the transformation.

1. To add a **Select values** step to the transformation, from the **Transform** category, drag the **Select values** step to the canvas, and drop it to the right of the **slkp-get_postalcodes** step.
2. Create a hop from the **slkp-get_postalcodes** step to the **Select values** step.
3. To configure the **Select values** step, on the canvas, double-click the **Select values** step.
4. To name the step, in the **Step name**, type `sv-prepare_field_layout`
5. To obtain the fields from the stream, at the right side of the **Fields** grid, click the **Get fields to select** button.
6. To delete the **POSTALCODE** field, scroll to the **POSTALCODE** (line 20), and then select and delete the line.
7. To move the **ZIP_RESOLVED** field, scroll to the **ZIP_RESOLVED** field, select the line, and then press **Ctrl-Up Arrow** repeatedly until **ZIP_RESOLVED** is in line 20.
8. To configure the metadata of the **ZIP_RESOLVED** field to match the original **POSTALCODE** field, click the **Meta-data** tab.
9. To select the **ZIP_RESOLVED** field, from the first row's **Fieldname** dropdown list, select **ZIP_RESOLVED**.
10. Configure the remainder of this row as follows:
    a. In the **Rename to** column, type `POSTALCODE`
    b. In the **Length** column, type `9`

11. To close the **Select/Rename values** dialog, click **OK**.
12. Create a hop from the **sv-prepare_field_layout** step to the **to-sales_data_ex6** step, and then select **Main output of step**.
13. To preview the **to-sales_data_ex6** step:
    a. On the canvas, right-click the **to-sales_data_ex6** step.
    b. From the context menu, select **Preview**.
    c. Change the **Number of rows to retrieve** to `3000`.
    d. Click the **Quick Launch** button.
14. Review the results and notice all rows have a **POSTALCODE**.
15. To close the **Examine preview data** dialog, click **Close**.
16. To close the **Select the preview step** dialog, click **Close**.
17. Save the transformation.
18. To run the transformation:
    a. On the subtoolbar, click the **Run** button.
    b. In the **Run Options** dialog, click **Run**.
19. Close the **ex6_update_zip** tab.

Solution Details

The solution to this exercise can be found at:

`/home/pentaho/course_files/pdi1000l/solutions/exercises`

File:

`ex6_update_zip.ktr`

End of Exercise 6

Guided Demo 10: Creating Summary Fields Using Group By

In this guided demonstration, we create a transformation that reads in product data from the `pentaho_oltp` *database, sorts the data by product line, and then sums the quantity in stock for each product line.*

Objectives

After completing this guided demonstration, you will be able to:

- Configure the **Group by** step to summarize data based on a field in the stream

Prerequisites

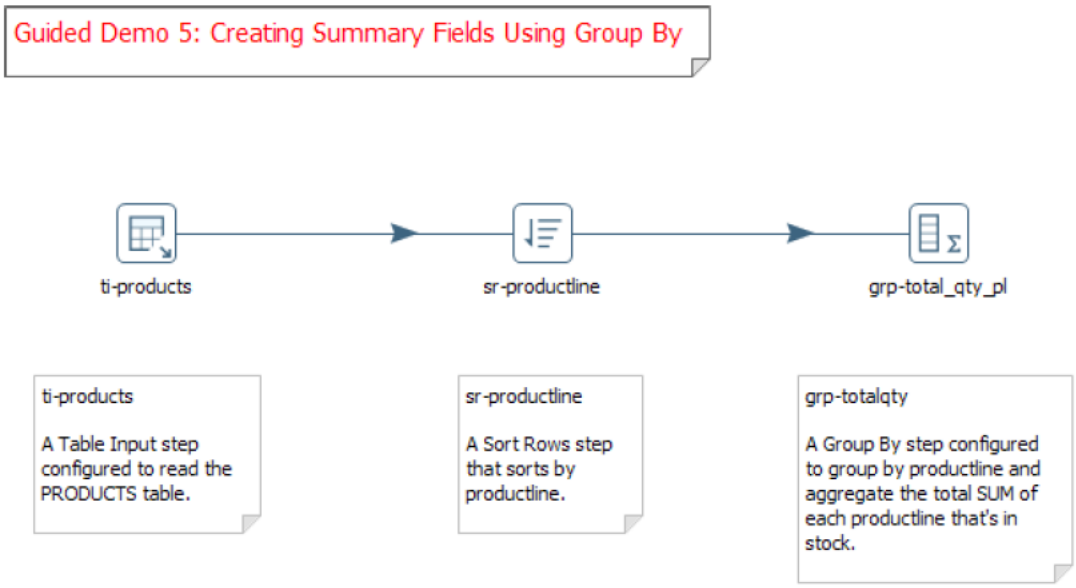
Prior to completing this guided demonstration, you must complete Guided Demo 7: Connections and the Database Explorer.

Steps Used

This guided demonstration uses the following steps:

- **Table input**
- **Sort rows**
- **Group by**

Transformation



Creating the Transformation and Configuring the Table Input Step

In this section of the guided demo, we create a new transformation and configure the **Table input** step to read the products table from the `pentaho_oltp` database.

1. Create a new transformation named `gd10_sort_group_by` and save it to the `PDI_Trn_Objects` folder on the repository.
2. From the **Input** category, drag a **Table input** step to the canvas.
3. To configure the **Table input** step, on the canvas, double-click the **Table input** step.
4. To name the step, in the **Step name**, type `ti-products`
5. To select the database connection, from the **Connection** dropdown list, select **pentaho_oltp**.
6. To generate the SQL that obtains the data, in the **Table input** dialog, click the **Get SQL select statement** button.
7. To select the `products` table from the `pentaho_oltp` database connection, in the **Database Explorer** dialog:
    a. Click to expand **pentaho_oltp**.
    b. Click to expand **Tables**.

c. Click to select the **products** table.

d. Click **OK**.

8. To include the field names in the SQL statement, in the **Question?** dialog, click **Yes**.

9. In the SQL statement, delete the **productdescription** line (including the leading comma).

10. To preview the data and confirm the SQL statement is correct, click **Preview**, and then click **OK**.

    Notice the **productline** and the **quantityinstock**. In this demonstration we sort the results by **productline** and then sum the **quantityinstock** for each **productline**.

11. To close the **Examine preview data** dialog, click **Close**.

12. To close the **Table input** dialog, click **OK**.

13. Save the transformation.

Configure the Sort Rows Step

In this section of the guided demonstration, we configure the **Sort rows** step to sort the stream by `productline`.

1. To add a **Sort rows** step to the transformation, from the **Transform** category, drag the **Sort rows** step to the canvas, and drop it to the right of the **ti-products** step.

2. Add a hop from the **ti-products** step to the **Sort rows** step.

3. To configure the **Sort rows** step, on the canvas, double-click the **Sort rows** step.

4. To name the step, in the **Step name**, type `sr-productline`

5. To specify the sort order, in **the** Fields grid:

    a. From the **Fieldname** dropdown list, select **productline**.

    b. From the **Ascending** dropdown list, select **Y**.

6. To close the **Sort rows** dialog, click **OK**.

7. Save the transformation.

Configure the Group by Step and Preview the Results

In this section of the guided demonstration, we configure the **Group by** step to sum the `quantityinstock` for each `productline`.

1. To add a **Group by** step to the transformation, from the **Statistics** category, drag the **Group by** step to the canvas, and drop it to the right of the **sr-productline** step.
2. Add a hop from the **sr-productline** step to the **Group by** step.
3. To configure the **Group by** step, on the canvas, double-click the **Group by** step.
4. To name the step, in the **Step name**, type `grp-total_qty_pl`
5. To group by `productline`, from the **Group field** dropdown list, select **productline**.
6. To name the **Group by** field, in the **Name** field, type `total_qty_pl`
7. To select which field to summarize, from the **Subject** dropdown list, select **quantityinstock**.
8. To select the aggregation type, from the **Type** dropdown list, select **Sum**.
9. To close the **Group by** dialog, click **OK**.
10. To close the **Notice** message, click **Close**.
11. To preview the **grp-total_qty_pl** step:
    a. On the canvas, right-click the **grp-total_qty_pl** step.
    b. From the context menu, select **Preview**.
    c. Click the **Quick Launch** button.
12. Review the results and notice the grouped data.
13. To close the **Examine preview data** dialog, click **Close**.
14. To close the **Select the preview step** dialog, click **Close**.
15. To modify the **grp-total_qty_pl** step, on the canvas, double-click the **grp-total_qty_pl** step.
16. To show the summed `quantityinstock` with each row of data, in the **Group By** dialog, click to enable the **Include all rows?** checkbox.
17. To close the **Group By** dialog, click **OK**.

18. To close the **Notice** message, click **Close**.
19. To preview the **grp-total_qty_pl** step:
    a. On the canvas, right-click the **grp-total_qty_pl** step.
    b. From the context menu, select **Preview**.
    c. Click the **Quick Launch** button.
20. Review the results and notice the grouped data appears for each row.
21. To close the **Examine preview data** dialog, click **Close**.
22. To close the **Select the preview step** dialog, click **Close**.
23. Save the transformation.
24. Close the **gd10_sort_group_by** tab.

Solution Details

The solution to this guided demonstration can be found at:

`/home/pentaho/course_files/pdi10001/solutions/guided_demos`

File:

`gd10_sort_group_by.ktr`

End of Guided Demo 10

Exercise 7: Calculating and Aggregating Order Quantity

In this exercise, you create a transformation that reads in order data from a file. You keep only some of the fields in the stream and sort the data by country. You then aggregate the quantity ordered for each country and include the aggregated value for each row. Finally, you calculate the percentage of the total quantity for the country that was ordered in each row.

Objectives

After completing this exercise, you will be able to:

- Configure the **Calculator** step to create a calculated value
- Use the **Group by** step to create an aggregation

Prerequisites

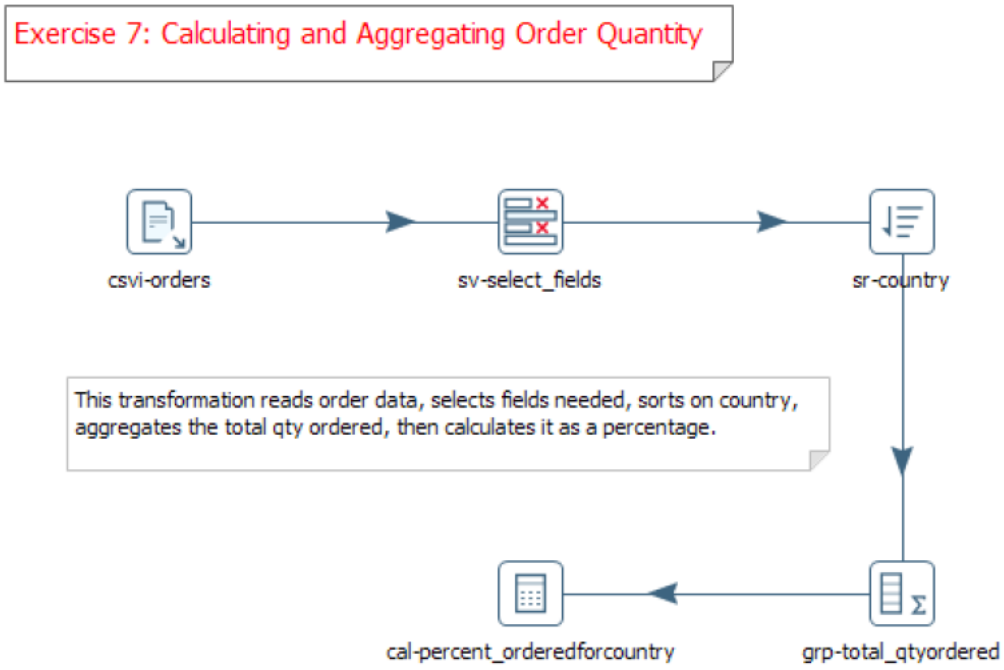
Prior to completing this exercise, you must complete Guided Demo 7: Connections and the Database Explorer. This exercise requires access to the input files that reside on the course student environment.

Steps Used

This exercise uses the following steps:

- **CSV file input**
- **Select values**
- **Sort rows**
- **Group by**
- **Calculator**

Transformation



Create the Transformation and Configure the CSV File Input Step

In this section of the exercise, you create and save the new transformation and configure the **CSV file input** step to read in order data.

1. Create a new transformation named `ex7_sorting_data_grouping` and save it to the `PDI_Trn_Objects` folder on the repository.
2. To add a **CSV file input** step, on the **Design** tab, expand the **Input** category, and then drag the **CSV file input** step to the canvas.
3. To configure the **CSV file input** step, on the canvas, double-click the **CSV file input** step.
4. To name the step, in the **Step name**, type `csvi-orders`
5. To use the `DIR_INPUT` variable:
    a. Click in the **Filename** field.
    b. On the keyboard, press **Ctrl-Space**.
    c. From the list of variables, double-click **DIR_INPUT**.

6. To append the filename to the variable, in the **Filename** field, after the variable, type `/order_file.csv`
7. To change the separator, in the **Delimiter**, type a semicolon `;`
8. To turn off lazy conversion, click to deselect the **Lazy conversion?** checkbox.
9. To populate the **Fields** grid:
    a. Click the **Get Fields** button.
    b. In the **Sample size** dialog, change the value to `0`.
    c. Click **OK**.
10. To preview the data and confirm it is configured properly, click **Preview**, and then click **OK**.
11. Review the preview data, and then, to close the **Examine preview data** dialog, click **Close**.
12. To close the **csvi-orders** dialog, click **OK**.
13. Save the transformation.

Configure the Select Values Step

In this section of the exercise, you configure the **Select values** step to only include some of the fields in the stream because this transformation does not need everything from the `Order_Data` file.

1. To add a **Select values** step, from the **Transform** category, drag the **Select values** step to the canvas, and drop it to the right of the **csvi-orders** step.
2. Create a hop from the **csvi-orders** step to the **Select values** step and select **Main output of step**.
3. To configure the **Select values** step, on the canvas, double-click the **Select values** step.
4. To name the step, in the **Step name**, type `sv-select_fields`
5. Complete the **Fields** grid of the **Select & Alter** tab as follows:

Fieldname	Rename to
`ordernumber`	<empty>
`quantityordered`	<empty>
`priceeach`	<empty>

Fieldname	Rename to
customernumber	<empty>
country	<empty>

6. To close the **Select values** dialog, click **OK**.
7. Save the transformation.

Configure the Sort Rows Step

In this section of the exercise, you configure the **Sort rows** step to sort the data in the stream by country.

1. To add a **Sort rows** step, from the **Transform** category, drag the **Sort rows** step to the canvas, and drop it to the right of the **sv-select_fields** step.
2. Create a hop from the **sv-select_fields** step to the **Sort rows** step and select **Main output of step**.
3. To configure the **Sort rows** step, on the canvas, double-click the **Sort rows** step.
4. To name the step, in the **Step name**, type `sr-country`
5. To specify the sort order, in the **Fields** grid:
    a. From the **Fieldname** dropdown list, select **country**.
    b. From the **Ascending** dropdown list, select **Y**.
6. To close the **Sort rows** dialog, click **OK**.
7. Save the transformation.

Configure the Group by Step

In this section of the exercise, you configure the **Group by** step to group the data by country and calculate the total quantity ordered for each country. The values will be added to each row in a new field called `total_qtyordered`.

1. To add a **Group by** step to the transformation, from the **Statistics** category, drag the **Group by** step to the canvas, and drop it to the right of the **sr-country** step.
2. Add a hop from the **sr-country step** to the **Group by** step.
3. To configure the **Group by** step, on the canvas, double-click the **Group by** step.

4. To name the step, in the **Step name**, type `grp-total_qtyordered`
5. To show the summed `quantityordered` with each row of data, in the **Group by** dialog, click to select the **Include all rows?** checkbox.
6. To group by country, from the **Group field** dropdown list, select **country**.
7. To name the group by field, in the **Name** field, type `total_qtyordered`
8. To select which field to summarize, from the **Subject** dropdown list, select **quantityordered**.
9. To select the aggregation type, from the **Type** dropdown list, select **Sum**.
10. To close the **Group by** dialog, click **OK**.
11. To close the **Notice** message, click **Close**.
12. Save the Transformation.

Configure the Calculator Step

In this section of the exercise, you configure the Calculator step to calculate the percentage of the total quantity for the country that was ordered in each row.

1. To add a **Calculator** step to the transformation, from the **Transform** category, drag the **Calculator** step to the canvas, and drop it to the right of the **grp-total_qtyordered** step.
2. Add a hop from the **grp-total_qtyordered** step to the **Calculator** step.
3. To configure the **Calculator** step, on the canvas, double-click the **Calculator** step.
4. To name the step, in the **Step name**, type `cal-percent_orderedforcountry`
5. To configure the **Fields** grid:
    a. In the **New field** property, type `percent_orderedforcountry`
    b. From the **Calculation** dropdown list, select **100 * A / B**
    c. From the **Field A** dropdown list, select **quantityordered**
    d. From the **Field B** dropdown list, select **total_qtyordered**

*When configuring the **Calculator** step's field grid, each new field you define is added to the list of available fields for subsequent rows. This allows you to use the field you just defined in other*

calculations in the same step.

6. To close the **Calculator** dialog, click **OK**.

7. Save the Transformation.

Preview the Results of Each Step

In this section of the exercise, you preview the results of each step to see how the data in the stream changes throughout the transformation.

In the real world, it's best to preview each step as you build the transformation to make sure your configuration is correct before moving on.

1. To preview the **csvi-orders** step:

    a. On the canvas, right-click the **csvi-orders** step.

    b. From the context menu, select **Preview**.

    c. Click the **Quick Launch** button.

2. Review the results and notice the step read in all the fields in the file.

3. To close the **Examine preview data** dialog, click **Close**.

4. To close the **Select the preview step** dialog, click **Close**.

5. To preview the **sv-select_fields** step:

    a. On the canvas, right-click the **sv-select_fields** step.

    b. From the context menu, select **Preview**.

    c. Click the **Quick Launch** button.

6. Notice the results only included the selected values.

7. To close the **Examine preview data** dialog, click **Close**.

8. To close the **sv-select_fields** dialog, click **Close**.

9. To preview the **sr-country** step:

    a. On the canvas, right-click the **sv-select_fields** step.

b. From the context menu, select **Preview**.

c. Click the **Quick Launch** button.

10. Notice the results are sorted by country.
11. To close the **Examine preview data** dialog, click **Close**.
12. To close the **Select the preview step** dialog, click **Close**.
13. To preview the **grp_total_qtyordered** step:

    a. On the canvas, right-click the **grp_total_qtyordered** step.

    b. From the context menu, select **Preview**.

    c. Click the **Quick Launch** button.

14. Notice each row includes the aggregated value called **total_qtyordered**.
15. Close the **ex7_sorting_data_grouping** tab.

Solution Details

The solution to this exercise can be found at:

`/home/pentaho/course_files/pdi1000l/solutions/exercises`

File:

`ex7_sorting_data_grouping.ktr`

End of Exercise 7

Guided Demo 11: Onboarding Data with a Job

Your organization's ETL processes are critical to its success. Orchestrating those processes using jobs ensures they run successfully. If something does go wrong, errors are logged, and perhaps other people are notified. In this guided demonstration, we create a transformation that loads item data into a table from a delimited file. Then, we create a job that runs the transformation, making certain everything that is needed is present. We will design the job to create the database table and log the progress at important areas of execution. Both the transformation and job will be parameterized and documented.

Objectives

After completing this guided demonstration, you will be able to:

- Determine the conditions for running a transformation successfully
- Create a new job that runs the transformation only if those conditions exist
- Log the results when conditions are met or not met
- Demonstrate failure by running job when the requirements are not met
- Demonstrate success by running job when requirements are met

Prerequisites

Prior to completing this guided demonstration, you must complete Guided Demo 7: Connections and the Database Explorer.

You must also have access to the DI1000 course files that reside on the course student environment.

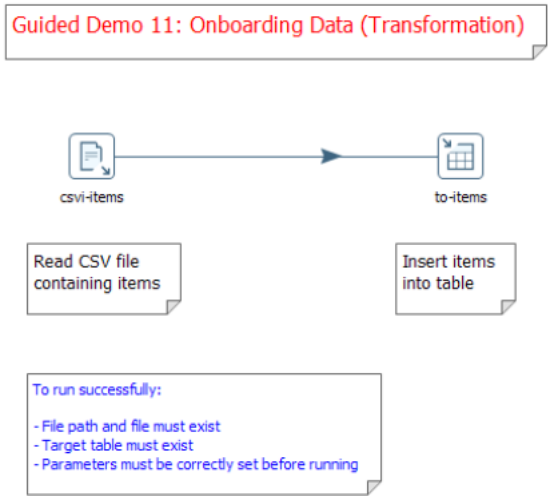
Job Entries Used

This guided demonstration uses the following job entries:

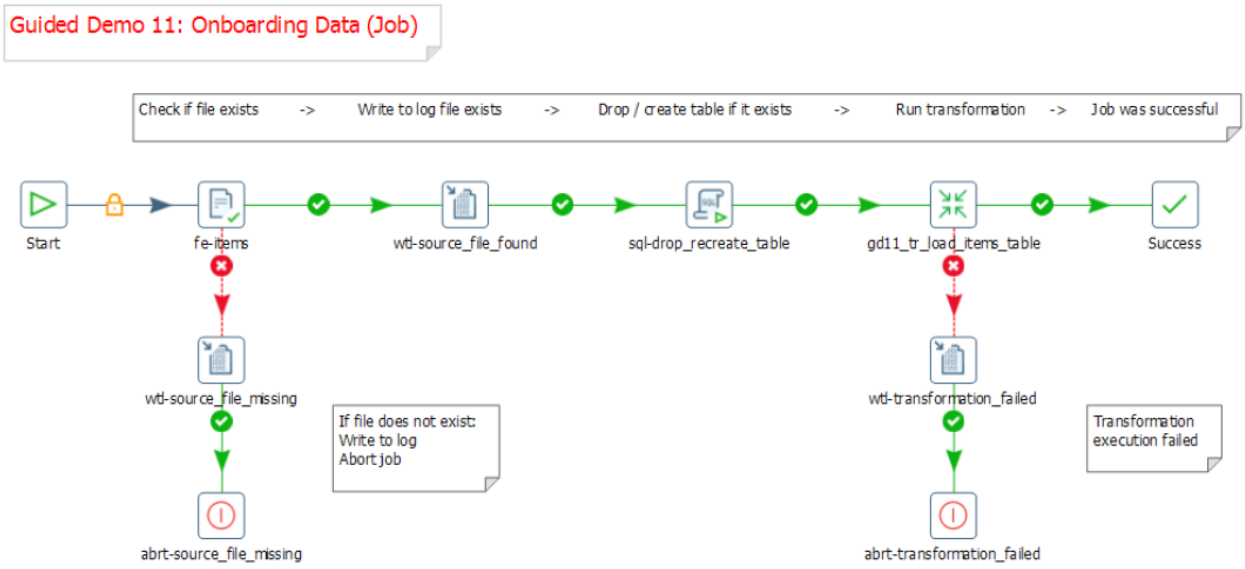
- Start
- File exists
- Write to log
- SQL
- Transformation

- Success
- Abort

Transformation



Job



Create and Verify Transformation

In this section, we create a parameterized transformation that reads item data from a delimited file and loads it into a database table. We will start with hardcoded configuration values so we can verify it works correctly. Later, we will transition them to parameters. As you create the transformation, think about all the things that

must be set up for it to run successfully. For example, the input data must exist, database table must exist, and so on. This knowledge will help us design the job later in the guided demonstration.

1. Create a new transformation.
2. Open the transformation properties dialog using **CTRL-T** or double-clicking an empty area of the canvas.
3. Click on the **Parameters** tab.
4. Add three parameters and their descriptions as shown. Leave the **Default Value** column blank.

Parameter name	Default value	Description
`delimiter`	(blank)	The input file delimiter.
`input_file`	(blank)	The input file path and filename.
`table_name`	(blank)	The target table name.

Note: The values for these parameters will be passed to the transformation by the job.

5. To close the transformation properties dialog, click **OK**.
6. Add the **CSV file input** step to the canvas.
7. Open the step and configure its properties as shown:

Property name	Value
Step name	`csvi-items`
Filename	`/home/pentaho/course_files/pdi1000l/data_files/input/items.csv`
Delimiter	`|` (pipe symbol)

8. To configure the **Fields to insert** grid:
    a. Click the **Get Fields** button.
    b. Accept the default sample size (we know this is adequate for our data) by clicking **OK**.
9. To check the configuration of this step:
    a. Click **Preview**.
    b. Click **OK**.

c. Verify the data from the file is displayed correctly, then click **Close**.

10. Close the **CSV file input** step dialog by clicking **OK**.

11. Add a **Table output** step to the canvas.

12. Create a hop from the **CSV file input** step to the **Table output** step.

13. Open the **Table output** step and configure its properties as shown:

Property name	Value
Step name	`to-items`
Connection	`pentaho_oltp`
Target table	`ITEMS`
Specify database fields	(checked)

14. Click the **Database fields** tab.

15. To define the table column to stream field mapping, configure the **Fields to insert grid** as shown:

Table field	Stream field
`item_id`	`item_id`
`status_id`	`status_id`
`serial_number`	`serial_number`
`warehouse_location_id`	`warehouse_location_id`
`expiration_date`	`exp_dt`
`total_value`	`value`

*Note: Alternatively, you could have clicked the **Get fields** button and then updated the last two rows.*

16. To create the target table:

a. Click the **SQL** button.

b. Click **Execute**.

c. To close the **Results of the SQL statements** dialog, click **OK**.

d. To close the **Simple SQL editor** dialog, click **Close**.

17. Close the **Table output** step by clicking **OK**.

18. Save the transformation in the repository's `Public/PDI_Trn_Objects` folder and name it: `gd11_tr_load_items_table`

19. Run the transformation to make sure it runs correctly with the hardcoded configuration values by verifying there are no errors and the data is loaded into the table. If there are errors, double-check the configuration, make any corrections, and run it again.

Parameterize the Transformation

Now that the transformation has been created and verified, it is ready to be parameterized. In this section, we replace the hardcoded file path and name, delimiter, and target table with the parameters we created earlier. The values for these parameters will be provided by the job in which this transformation runs.

1. Double-click the **CSV file input** step to open it.

2. Replace the current property values with those shown:

Property name	Value
Filename	`${input_file}`
Delimiter	`${delimiter}`

3. Close the step by clicking **OK**.

4. Open the **Table output** step.

5. Replace the current **Target table property value** with the one shown:

Property name	Value
Target table	`${table_name}`

6. Close the step by clicking **OK**.

Document the Transformation

Documenting transformations helps you and others easily understand how it works. In this section, we add notes to the canvas explaining what the transformation does and what is required for it to run successfully.

1. Add a note to the canvas under the **CSV file input** step by right-clicking the canvas and clicking **New Note**. The note's text is:

Read CSV file

containing items

---

2. Add a second note to the canvas under the **Table output** step. The note text is:

Insert items

into table

---

3. Add a third note to the canvas that explains the high-level criteria needed for the transformation to run successfully. This helps us know how to design the job. The note text is:

To run successfully:

- File path and file must exist

- Target table must exist

- Parameters must be correctly set before running

---

4. Save the transformation.

Create the Job and Parameters

The transformation has been created and verified. It is time to create the job that will run it. The job will be designed to run the transformation only when the requirements for it to execute successfully are in place. These requirements can be determined easily from the last note that was added to the transformation. In this section, we will begin creating the job, its parameters, and add the **Start job** entry.

1. To create a new job, on the toolbar, click the **New file** button, and then select **Job**.

2. Open the job properties dialog by double-clicking an empty area of the canvas.

3. Click on the **Parameters** tab.

4. Add four parameters, default values, and descriptions as shown:

Parameter name	Default value	Description
`delimiter`	`|` (pipe symbol)	The input file delimiter.

Parameter name	Default value	Description
`input_file`	`/home/pentaho/course_files/pdi10001/data_files/input/items_not_here.csv` Note: We are deliberately configuring for a file that does not exist.	The input file path and file name.
`sql_file`	`/home/pentaho/course_files/pdi10001/data_files/input/gd11_drop_create_table.sql`	The SQL script that drops and creates the target table.
`table_name`	`ITEMS`	The target table name.

The values will be passed to the transformation by the job.

5. To close the job properties dialog, click **OK**.
6. In the **Design** tab, expand the **General** category, and drag the **Start job** entry to the canvas.
7. Save the job in the repository's `Public/PDI_Trn_Objects` folder and name it: `gd11_jb_load_items_table`

Add and Configure File Exists Job Entry

After the job starts, it will use the **File exists** job entry to check for the existence of the input file that the transformation will read. In this section, we will add and configure the **File exists** job entry.

1. Using the **Design** tab search box, find the **File exists** job entry, and drag it to the right of the **Start job** entry.
2. Create a hop between the **Start** and **File exists** job entries.
3. Open the **File exists** job entry and configure it as shown:

Property name	Value
Job entry name	`fe-items`
File name	`${input_file}`

4. Click **OK** to close the **File exists** job entry.

Add and Configure File Exists Destination Job Entries

In this section, we define success and failure paths for the **File exists** job entry. Both paths will lead to **Write to log** job entries that will write messages to the log indicating if the source file was found or not. In the event

the file is not found, the job will be aborted with the **Abort job** entry after writing to the log.

1. Using the **Design** tab search box, find the **Write to log** job entry, and drag it to the right of the **fe-items** job entry.
2. Create a hop between the **fe-items** and **Write to log** job entries. The hop should be green and have a green checkmark icon on it to indicate this is the success path.
3. Open the **Write to log** job entry and configure it as shown:

Property name	Value
Job entry name	`wtl-source_file_found`
Log level	`Basic`
Log subject	`***SOURCE FILE FOUND***`
Log message	`The "${input_file}" file will be loaded to the "${table_name}" table.`

4. Close the **Write to log** job entry by clicking **OK**.
5. Add a second **Write to log** job entry directly below the **fe-items** job entry.
6. Create a hop between the **fe-items** and the second **Write to log** job entries. The hop should be red and have a red X icon on it to indicate this is the failure path.
7. Open the second **Write to log** job entry and configure it as shown:

Property name	Value
Job entry name	`wtl-source_file_missing`
Log level	`Error`
Log subject	`***SOURCE FILE MISSING***`
Log message	`***WARNING: THE SOURCE FILE "${input_file}" IS NOT PRESENT.***`

8. Close the **wtl-source_file_missing** job entry.
9. Add an **Abort job** entry directly below the **wtl-source_file_missing** job entry.
10. Create a hop between the **wtl-source_file_missing** and **Abort job** entries.
11. Open the **Abort job** entry and configure it as shown:

Property name	Value
Abort job	`abrt-source_file_missing`
Message	`source file is missing.`

12. Close the **Abort job** entry by clicking **OK**.

13. Save the job.

Add and Configure SQL Job Entry

The target table must exist before the transformation tries to load data into it. In this section, we add a SQL job entry that will execute SQL script that drops the target table if it already exists and then creates a new one. This technique ensures a new, empty table exists and the data that is contained in the table after it is loaded is the result of a single run of the job and transformation. The SQL script is in a file that we will point to.

1. Using the **Design** tab search box, find the SQL job entry, and drag it to the right of the **wtl-souce_file_found** job entry.

2. Create a hop between the **wtl-souce_file_found** and SQL job entries. The hop should be green and have a green checkmark icon on it to indicate this is the success path.

3. Open the SQL job entry and configure it as shown:

Property name	Value
Job entry name	`sql-drop_recreate_table`
Connection	`pentaho_oltp`
SQL from file	(checked)
SQL filename	`${sql_file}`
Send SQL as a single statement?	(unchecked)
Use variable substitution?	(checked)

You may want to locate and open the SQL script file in a text editor to become acquainted with it.

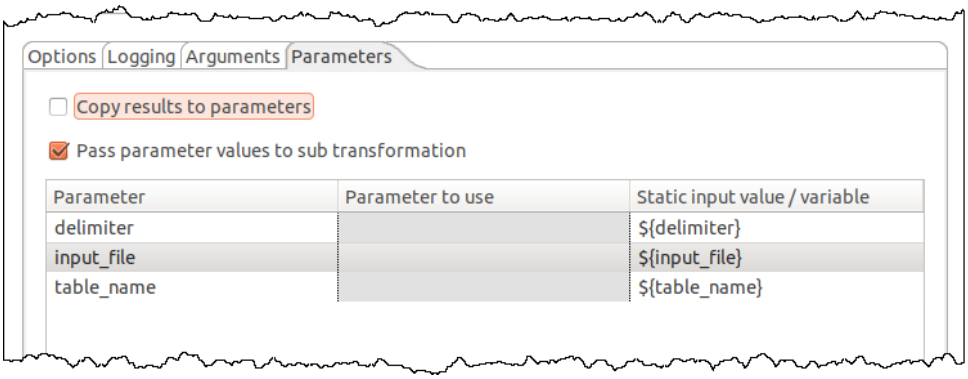
4. Close the **sql drop_recreate_table** job entry by clicking **OK**.

5. Save the job.

Add and Configure Transformation Job Entry

We're finished creating the portions of the job that ensure the file and target table exist. Now, the transformation can be executed. In this section, we add the **Transformation** job entry and configure it to run the transformation we created earlier. Part of this configuration includes passing the job parameter values down to the transformation.

1. Using the **Design** tab search box, find the **Transformation** job entry, and drag it to the right of the **sql drop_recreate_table** job entry.
2. Create a hop between the **sql drop_recreate_table** and **Transformation** job entries. The hop should be green and have a green checkmark icon on it to indicate this is the success path.
3. Open the **Transformation** job entry.
4. Name the job entry `gd11_tr_load_items_table`
5. To configure the transformation to run:
    a. Click the **Browse** button.
    b. Navigate to the **Public/PDI_Trn_Objects** folder.
    c. Select the **gd11_tr_load_items_table** and click **Open**.
6. Click the **Parameters** tab.
7. Verify the **Pass parameter values to sub transformation** checkbox is checked.
8. To automatically populate the grid with the configured transformation's parameters, click the **Get Parameters** button. This configuration will cause the job's parameter values to be passed to the transformation's parameters with the same name.
9. Map the parameters as shown:



10. Close the **gd11_tr_load_items_table** job entry by clicking **OK**.

11. Save the job.

Add and Configure Transformation Destination Job Entries

In this section, we define the success and failure paths from the **gd11_tr_load_items_table** job entry. If the transformation completes successfully, we mark the job successful using the **Success** job entry. If the transformation fails, we write to the log and abort the job.

1. Add a **Success** job entry directly to the right of the **gd11_tr_load_items_table** job entry.

2. Create a hop between the **gd11_tr_load_items_table** and **Success** job entries. The hop should be green and have a green checkmark icon on it to indicate this is the success path. The **Success** job entry doesn't require any configuration.

3. Add a **Write to log** job entry directly below the **gd11_tr_load_items_table** job entry.

4. Create a hop between the **gd11_tr_load_items_table** and new **Write to log** job entries. The hop should be red and have a red X icon on it to indicate this is the failure path.

5. Open the new **Write to log** job entry and configure it as shown:

Property name	Value
Job entry name	`wtl-transformation_failed`
Log level	`Error`
Log subject	`***TRANSFORMATION FAILED***`
Log message	`***WARNING: THE TRANSFORMATION FAILED TO RUN SUCCESSFULLY.***`

6. Close the **wtl_transformation_failed** job entry by clicking **OK**.

7. Add an **Abort job** entry directly below the **wtl_transformation_failed** job entry.

8. Create a hop between the **wtl_transformation_failed** and new **Abort job** entries.

9. Open the new **Abort job** entry and configure it as shown:

Property name	Value
Abort job	`abrt-transformation_failed`

Property name	Value
Message	`Transformation failed to execute successfully`

10. Close the **abrt-transformation_failed** job entry by clicking **OK**.

11. Save the job.

Document the Job

It is as important to document jobs as it is transformations. In this section, we add notes to the canvas explaining what each branch of the job does.

1. Add a note that describes what happens along the path that leads to the **Success** job entry. Place it all on one line and centered above the main success branch of the job. The note's text is:

   Check if file exists -> Write to log file exists -> Drop / create table -> Run transformation -> Job was successful

   ---

2. Add a note that describes what happens along the branch that leads to the **abrt-source_file_missing** job entry. Place it near the **abrt-source_file_missing** job entry. The note's text is:

   If file does not exist:

   Write to log

   Abort job

   ---

3. Add a note that describes what happens along the branch that leads to the **abrt-transformation_failed** job entry. Place it near the **abrt-source_file_missing** job entry. The note's text is:

   Transformation

   execution failed

   ---

4. Save the job.

Test Job Failure Path

In this section, we verify the failure path works properly by causing the job to fail. In fact, it should fail already because it is currently configured to look for a file that does not exist. The job will not find the file, log the

failure, and abort. Then, we examine the log and metrics.

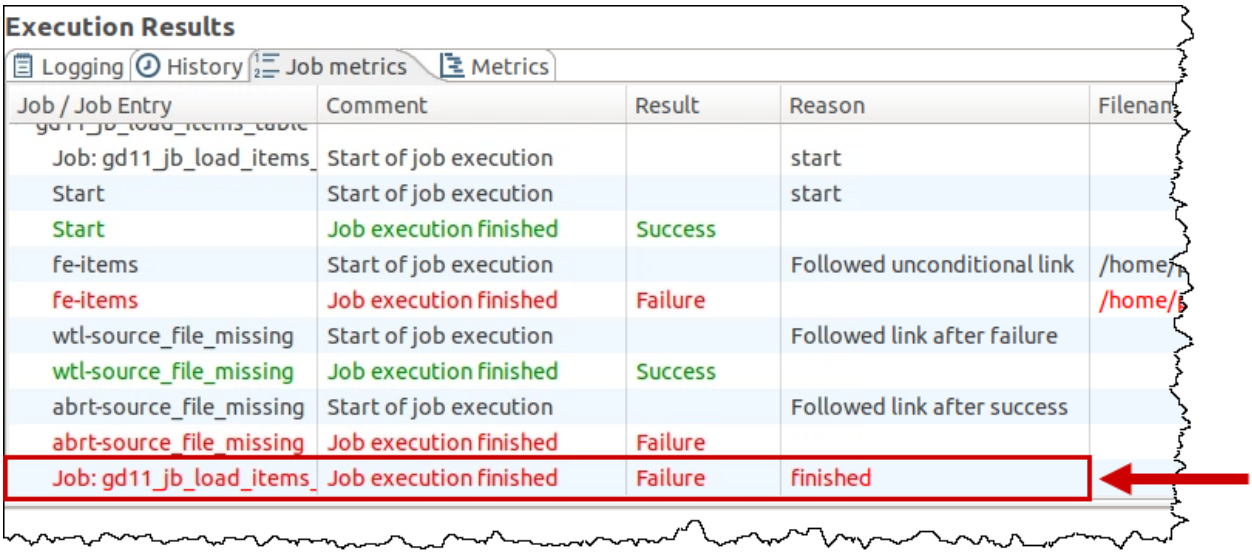
1. To run the job:

    a. On the subtoolbar, click **Run**.

    b. In the **Run Options** dialog, click **Run**.

2. Examine the **Logging** tab and find the entry that indicates the source file was not found. It begins with `*** SOURCE FILE MISSING ***`.

3. Review the **Job Metrics** tab and confirm the job finished with a failure result.



Test Job Success Path

In this section, we "fix" the issue by changing the input_file parameter to point to a file that exists. Then, rerun the job when all the requirements for its success are met.

1. Click on the **Parameters** tab.

2. Change the input_file parameter's, default values, as shown:

Parameter name	Default value	Description
`input_file`	`/home/pentaho/course_files/pdi10001/data_files/input/items.csv`	The input file path and file name.
`sql_file`	`/home/pentaho/course_files/pdi10001/data_files/input/gd11_drop_create_table.sql`	The SQL script that drops and creates the target table.

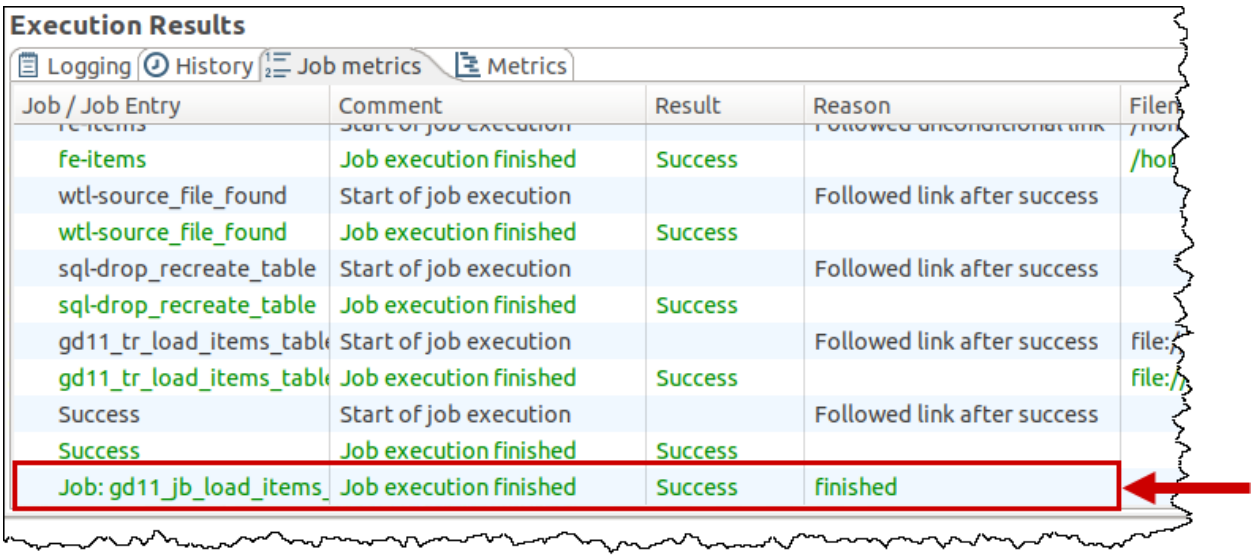
Parameter name	Default value	Description
table_name	ITEMS	The target table name.

3. To close the job properties dialog, click **OK**.

4. Run the job.

5. Examine the **Logging** tab and find the entry that indicates the source file was found. It begins with `*** SOURCE FILE FOUND ***`.

6. Review the **Job Metrics** tab and confirm the job ran successfully with a success result.



Verify Data is Loaded

We finish this guided demonstration by using the Database Explorer to verify the table has been loaded with the item data.

1. To explore the database connection:

    a. On the **View** tab, expand **Database Connections**.

    b. Right-click **pentaho_oltp** and click **Explore**.

2. To verify the item data has been loaded into the table:

    a. In the Database Explorer, expand **pentaho_oltp**.

    b. Click to expand **Tables**.

    c. Right-click the **items** table and click **Preview first 100**.

3. Examine the data in the table.
4. Close the preview dialog.
5. To close Database Explorer, click **OK**.
6. Close the job and transformation tabs.

Solution Details

The solution to this guided demonstration can be found at:

`/home/pentaho/course_files/pdi1000l/solutions/guided_demos`

Files:

`gd11_tr_load_items_table.ktr`

`gd11_jb_load_items_table.kjb`

End of Guided Demo 11

Guided Demo 12: Using the Pentaho Enterprise Repository

In this guided demonstration, we explore the repository, add folders and move a file between folders.

Objectives

After completing this guided demonstration, you will be able to:

- Access the Repository Explorer
- Add folders to the repository
- Import a file to the repository
- Move a file between folders on the repository

Prerequisites

Prior to completing this guided demonstration, you must complete Guided Demo 7: Connections and the Database Explorer.

You must also have access to the sample files that reside on the course student environment.

Exploring the Repository Structure

In this section of the guided demonstration, we explore the repository, add folders, and move a file between folders.

1. To explore the repository, from the menu, select **Tools → Repository → Explore**. Alternatively, you can click the **Explore Repository** button on the toolbar.
2. To view the contents of the `PDI_Trn_Objects` folder, in the left-side tree, expand the **public** folder and then click to select the **PDI_Trn_Objects** folder.
3. To add a new folder, right-click **PDI_Trn_Objects**, and then select **New Folder**.
4. To name the folder, in the **Enter name for New Folder** field, type `development`
5. Repeat the previous steps to add two more folders under the `PDI_Trn_Objects` folder: `testing` and `production`.

6. To close the **Repository** explorer, click **Close**.
7. To import a transformation to the `development` folder:
    a. From the menu, select **File → Import from an XML** file.
    b. Navigate to: `/home/pentaho/course_files/pdi1000l/demo`
    c. Click to select the `fact_sales_basic.ktr` transformation.
    d. Click **OK**.
8. To save the transformation, on the toolbar, click **Save**.
9. To save the transformation to the `development` folder, in the **Transformation properties** dialog:
    a. Navigate to the `Public/PDI_Trn_Objects/development` repository folder
    b. Click **Save**.
10. To explore the repository, on the toolbar, click the **Explore Repository** button.
11. To access the `development` folder:
    a. Click to expand the **Public** folder.
    b. Click to expand the **PDI_Trn_Objects** folder.
    c. Click to select the **development** folder.

    Notice the **File Name**, **Type**, and **Date Modified**.
12. To move the transformation to the `/home/admin` repository folder, click to select the `fact_sales_basic.ktr` file, and drag it to the `/home/admin` repository folder.
13. To verify the file was moved, click to expand the **home** folder, and then click to select the **admin** folder.
14. Close the Repository Explorer.

End of Guided Demo 12

Guided Demo 13: Scheduling and Monitoring

In this guided demonstration, we configure scheduling and monitoring using PDI.

Objectives

After completing this guided demonstration, you will be able to:

- Schedule the execution of a transformation/job using Spoon
- Monitor job and transformation execution on the Pentaho server

Prerequisites

Prior to completing this guided demonstration, you must complete Guided Demo 7: Connections and the Database Explorer, and Guided Demo 12: Using the Pentaho Repository.

Scheduling a Transformation and Viewing the Scheduler Perspective

In this section of the guided demonstration, we schedule the `fact_sales_basic` transformation to run every two minutes. We then look at the scheduled transformation on the Scheduler Perspective.

1. To open the `fact_sales_basic` transformation:
    a. From the menu, select **File → Open**.
    b. Navigate to the `/home/admin` repository folder.
    c. Click to select the `fact_sales_basic` transformation.
    d. Click **Open**.

Feel free to use a different job or transformation instead of `fact_sales_basic` if you like. Whichever one you choose, it must be saved in the Pentaho Repository in order to schedule it.

2. To schedule the transformation, from the menu, select **Action → Schedule**.
3. To schedule the transformation to run every two minutes, in the **Schedule** dialog, set the following options, and then click **OK**.

Option	Setting
Start	`Now`
Repeat	`Minutes`
Every minute(s)	`2`

4. To view the Scheduler Perspective, from the menu, select **View → Perspectives → Scheduler**.
5. Review the scheduling options. Notice the **Next Run** and **Last Run (duration)** options.
6. To return to the Data Integration perspective, from the menu, select **View → Perspectives → Data Integration**.

Do not click the X in the top right corner to close the Scheduler. That closes Spoon.

Monitoring a Slave Server (Carte)

In this section of the guided demonstration, we create a connection to a slave server (Pentaho Server in this case) and monitor transformation activity.

1. To create a connection to a slave server:
    a. Click the **View** tab.
    b. Right-click **Slave server**.
    c. Select **New**.
2. To configure the slave server connection, in the **Slave Server** dialog, type the following values, and then click **OK**.

Field	Value
Server name	`Class PS Server`
Hostname or IP address	`localhost`
Port (empty is port 80)	`8080`
Web App Name (optional)	`pentaho`
Username	`admin`
Password	`password`

Field	Value
Is the master	[unchecked]

3. To view the slave server, on the **View** tab:
    a. Click to expand **Slave server**.
    b. Right-click **Class PS Server**.
    c. Select **Monitor**.
4. To view the transformation, click to expand **Transformations**, and then click to expand one of the **fact_sales_basic** transformations.
5. Close the **Slave server: Class PS Server** tab.
6. Close the **fact_sales_basic** tab.

End of Guided Demo 13

Guided Demo 14: Logging Execution Metrics to a Database

In this guided demonstration, we configure logging for a job and its job entries, so their execution metrics are stored in database tables.

Objectives

After completing this guided demonstration, you will be able to:

- Create database tables to store job and job entry execution metrics
- Configure logging for jobs and job entries
- View logging information in database tables

PDI comes with a database connection, `live_logging_info`, used to automatically log transformation and job metrics with Kettle logging variables in `kettle.properties`. This guided demo overrides those variables by hardcoding values, showing how to use different values than those defined in the variables. If you wish, explore `live_logging_info` to see how metrics logging has been automatically performed throughout this course.

Prerequisites

Prior to completing this guided demonstration, you must complete Guided Demo 7: Connections and the Database Explorer, and Guided Demo 11: Onboarding Data with a Job.

Configuring Job Metrics Logging

In this section of the guided demonstration, we configure job logging for the **gd11_jb_load_items_table** job and then use Database Explorer to view the log table.

1. To open the **gd11_jb_load_items_table** job:
    a. From the menu, select **File** → **Open**.
    b. Navigate to the **PDI_Trn_Objects** folder.
    c. Click to select the **gd11_jb_load_items_table** job.
    d. Click **OK**.
2. To access the job properties, on the canvas, double-click an empty area.

3. Click the **Log** tab.
4. To set the job logging properties, in the left panel, click **Job log table**.
5. To specify the database connection, from the **Log Connection** dropdown list, select **pentaho_oltp**.
6. To specify the log table name, in the **Log table** property, type `pdi_log_job`
7. Set the **Log schema** property to `public`
8. In the **Log table fields** grid, review the default selections, and verify the **LOG_FIELD** checkbox is enabled.

*Monitoring the **LOG_FIELD** field can negatively impact server performance, especially when logging transformations. However, if you do not select all fields, including **LOG_FIELD**, when configuring job logging, you will not see log details logged.*

9. To generate the SQL statement to create the log table, click the **SQL** button. Two Simple SQL Editor dialogs open. One over top of the other.
10. **Close** the **top** Simple SQL Editor dialog. This is the dialog with the create index statements. The underlying Simple SQL Editor dialog is displayed with a create table statement.
11. Verify the syntax of the create table SQL statement, and then to execute in the **Simple SQL editor** dialog, click **Execute**.
12. Verify the SQL statement executed, and then in the **Results of the SQL statements** dialog, click **OK**.

    Notice the SQL created indexes.
13. To close the **Simple SQL editor** dialog, click **Close**.
14. To close the **job properties** dialog, click **OK**.
15. Save the job.
16. Run the job.
17. To view the database connections, on the **View** tab, click to expand **Database connections**.
18. Right-click **pentaho_oltp**, and then from the context menu, click **Explore**.
19. To preview the **pdi_log_job** table, in the **Database Explorer** window:
    a. Click to expand **pentaho_oltp**.

b. Click to expand **Tables**.

c. Right-click **pdi_log_job**.

d. Select **Preview first 100**.

20. Examine the log data, and then in the **Examine preview data** dialog, click **Close**.
21. To close Database Explorer, click **OK**.
22. Run the job again.
23. Repeat the steps to view the `pdi_log_job` table, and then close Database Explorer.

Configuring Job Entry Logging

In this section of the guided demonstration, we configure job entry logging for the `gd11_jb_load_items_table` job, and then use Database Explorer to view the log table.

1. To access the job properties, on the canvas, double-click an empty area.
2. To view the **Log** tab, click the **Log** tab.
3. To specify the database connection, from the **Log Connection** dropdown list, select **pentaho_oltp**.
4. To set the job entry logging properties, in the left panel, click **Job entry log table**.
5. To specify the log table name, in the **Log table name**, type `pdi_log_entry`
6. In the **Log table fields** grid, review the default selections, and then to include the **LOG_FIELD**, click to select the **LOG_FIELD** checkbox.
7. To generate the SQL statement to create the log table, click the **SQL** button. Two Simple SQL Editor dialogs open. One over top of the other.
8. **Close** the **top** Simple SQL Editor dialog. This is the dialog with the create index statements. The underlying Simple SQL Editor dialog is displayed with a create table statement.
9. Verify the syntax of the create table SQL statement, and then to execute in the **Simple SQL editor** dialog, click **Execute**.
10. Verify the syntax of the SQL statement, and then to execute the SQL statement, in the **Simple SQL editor** dialog, click **Execute**.

11. Verify the SQL statement executed, and then in the **Results of the SQL statements** dialog, click **OK**.
12. To close the **Simple SQL editor** dialog, click **Close**.
13. To close the **job properties** dialog, click **OK**.
14. Save the job.
15. Run the job.
16. To view the database connections, on the **View** tab, click to expand **Database connections**.
17. Right-click **pentaho_olap**, and then from the context menu, click **Explore**.
18. To preview the **pdi_log_entry** table, in the Database Explorer window:
    a. Click to expand **pentaho_olap**.
    b. Click to expand **Tables**.
    c. Right-click **pdi_log_entry**.
    d. Select **Preview first 100**.
19. Examine the log data, and then in the **Examine preview data** dialog, click **Close**.
20. To close Database Explorer, click **OK**.
21. Close the job.

End of Guided Demo 14