



# Big Data Sandbox Workbook

**Contents**

Step 1: Fill the Data Lake ..... 3

What is it? ..... 3

Why do it? ..... 3

Value of Pentaho ..... 3

What You Will Accomplish..... 3

Fill the Data Lake Exercise 2: Explore how metadata injection can be used to automate data onboarding..... 7

Use Case 2: Create a Data Refinery ..... 13

What is it? ..... 13

Why do it? ..... 13

Value of Pentaho ..... 14

What You Will Accomplish..... 14

Create a Data Refinery Exercise 1: Transform CDR data and blend with geography data ... 15

Create a Data Refinery Exercise 2: Use Pentaho with Impala to blend CDR and IoT Data ... 27

Create a Data Refinery Exercise 3: Extend the PDI job to blend data and load to multiple locations..... 31

Create a Data Refinery Exercise 4: Explore data in PostgreSQL with Pentaho Analyzer ..... 37

Use Case 3: Self Service Data Preparation ..... 41

What is it? ..... 41

Why do it? ..... 41

Value of Pentaho .....	41
What You Will Accomplish.....	41
Self Service Data Preparation Exercise 1: Use Pentaho to prepare data for analytics .....	42
Use Case 4: Self Service Analytics .....	51
What is it? .....	51
Why do it? .....	51
Value of Pentaho .....	51
What You Will Accomplish.....	51
Self Service Analytics Exercise 1: Use Pentaho to visualize data.....	51

# Step 1: Fill the Data Lake

## What is it?

The data lake is the foundation for the modern data pipeline. The data pipeline involves many steps for going from raw data to business value, and the first step involves ingestion or onboarding the data. If data onboarding is not built and managed properly the data lake can become a data swamp: a disorganized, dumping ground of data. This first use case shows you how to ingest data into Hadoop using a variety of methods including file copy, metadata injection and Kafka stream processing.

## Why do it?

Onboarding data has always had challenges, and the challenges are greatly exacerbated in a big data context. To achieve maximum ROI on your data lake requires implementing efficient tools, processes, and architecture. Modern data onboarding challenges go beyond just 'connecting' to data sources or 'ingesting' data into a store of choice and introduce significant new challenges related to dealing with many more source of data that may changes over time. They also require a flexible, efficient, and governed process to be fully successful.

## Value of Pentaho

- Pentaho allows you define an ETL template for the overall data workflow without needing to specify any of the metadata detail.
- At run-time, metadata can be fed into the workflow, a process called metadata injection.
- This allows hundreds of data sources to be managed with a single, generic workflow template.
- All while reducing development time, risk, maintenance, and expense.
- We see our customers leverage this for a variety of use cases including:
  - Scalable data ingestion
  - Data migrations
  - Self-service data onboarding
  - And dynamic data discovery and parsing


## What You Will Accomplish

You will ingest the following data sources to Hadoop:

- Call Detail Records (CDR)
- Area Geography Master Records
- Hitachi Content Platform (HCP) logs

The sources used are data files, but sources could be through Kafka or other sources.

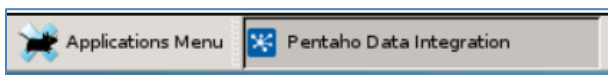
This first job loads the sources of raw data to HDFS so that it can be processed in Hadoop in a later exercise.

1. Launch the PDI client-based authoring tool (Spoon) from the launch menu icon  at the bottom of your screen. Click this icon just *once* to launch Spoon.

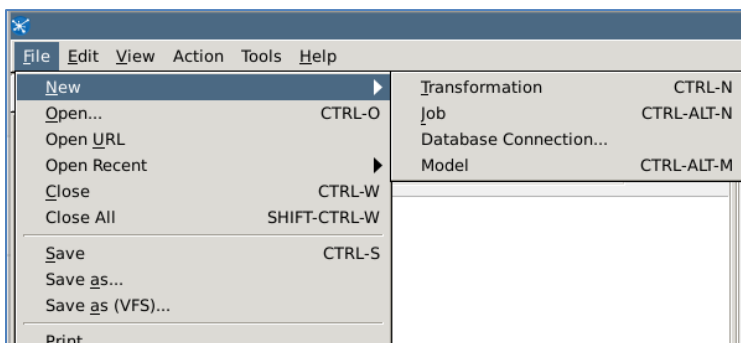


2. You should see the PDI splash screen appear while PDI loads.

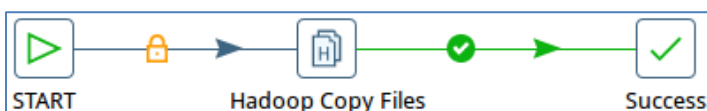
All open applications appear in the top left section of your screen. Once open, you will see Spoon as shown in the following screenshot.



3. From the main menu choose **File | New | Job**




4. From the **Design** tab on the left, expand the **General** folder and drag **START** and **Success** job steps onto the canvas.
5. Expand the **Big Data** folder and drag **Hadoop Copy Files** onto the canvas between the **START** and **Success** steps.
6. Connect the **Start** step and the **Hadoop Copy Files** step by creating a Hop. A Hop can be created by using the tool when highlighting over the step or by clicking the Shift key while moving the mouse from one step to the next.
7. Create a hop between the **Hadoop Copy Files** step and the **Success** step to match the following image:





To process data with our mapper transformation inside Hadoop, we need to define the data file that will be copied, and where it will be placed inside HDFS. Later in this exercise we will process data with a mapper transformation inside Hadoop, but before we can do that, we need to define the input file and place it inside the Hadoop cluster.

8. Double-click on **Hadoop Copy Files** to open its properties
9. On the **Settings** tab check the following two items: **Create destination folder** and **Replace existing files**.
10. In the **Files** tab select the cell beneath the “Source Environment” header on the file tab and select “Local” from the drop down.
11. Select the cell beneath the “Source File/Folder” step and select the  button.
12. Browse to the following file to select it, and then click the **OK** button to return to the **Copy Files** dialog box.

```
file:///home/demouser/pentaho/content/evaluation/01_Fill_the_data_lake/data/callrecords_all.csv
```

13. Click the **OK** button to return to the **Copy Files** dialog box
14. Select the cell beneath the “Destination Environment” header on the file tab and select “CDH” from the drop down.
15. Select the cell beneath the “Destination File/Folder” step and enter  
/BDO/callrecords/input
16. Repeat steps **8-13** to copy the “Source File/Folder”

```
file:///home/demouser/pentaho/content/evaluation/01_Fill_the_data_lake/data/callrecords_10years
```

to the “Destination File/Folder”  
/BDO/callrecords/input

and

```
file:///home/demouser/pentaho/content/evaluation/01_Fill_the_data_lake/data/areacodes.csv
```

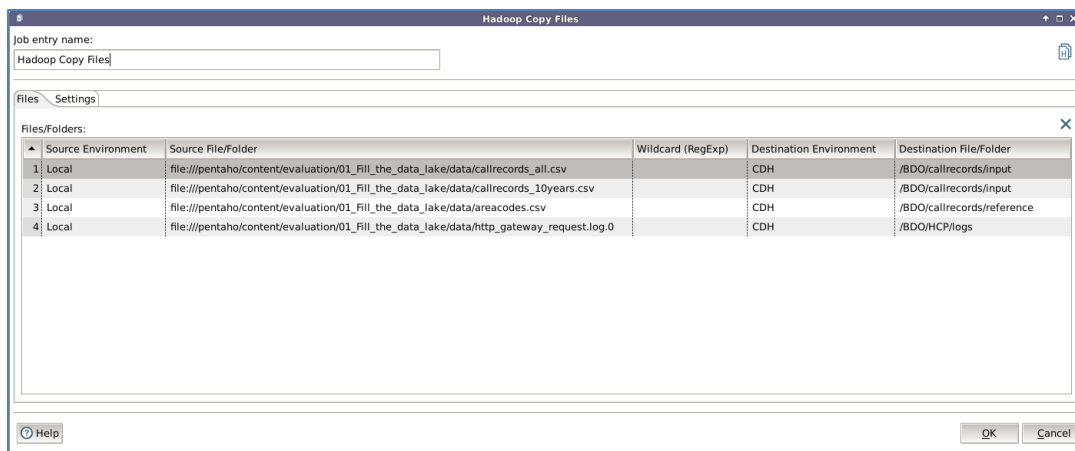
to the “Destination File/Folder”  
/BDO/callrecords/reference

and

```
file:///home/demouser/pentaho/content/evaluation/Fill_the_data_lake/data/http_gateway_request.log.0
```

to the “Destination File/Folder”  
/BDO/HCP/logs

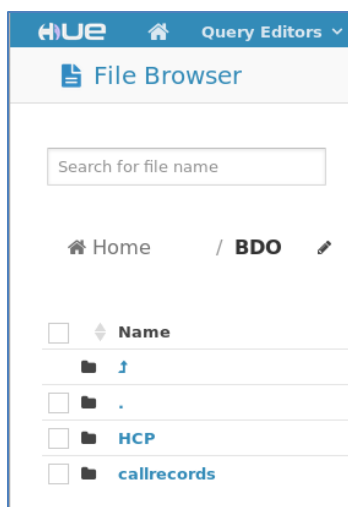
17. Your **Hadoop Copy Files** dialog box should match the following image (you can resize the window and table columns as needed).



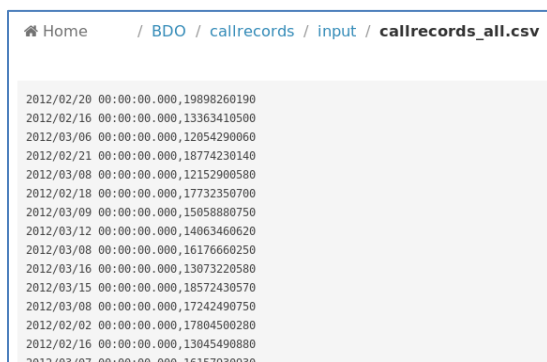
18. Click OK to return to the canvas.
19. From the **File** menu, choose **Save**.
20. In the Name field, specify `Fill Data Lake Exercise 1.kjb` and save to the following location:  
`/pentaho/content/evaluation/01_Fill_the_data_lake/student_files`
21. From the **Action** menu, choose **Run** and then click **Run**. On the **Job metrics** tab located in the bottom section of Spoon, you should see the job progress.
22. To view the newly copied CDR data in Hadoop, launch Firefox by single-clicking the Firefox icon at the bottom of your screen.



23. From the Firefox bookmarks bar click **Hue**, and on the sign in page, sign in with **demouser / demouser**
24. In the far-right corner, click **HDFS Browser** to locate the files
25. Navigate to the following directory: `/BDO`
26. You should see your files in each of the specified folders. For reference, you can open the **Hadoop Copy Files** step in your PDI job and check the destination.



27. If your job executes successfully you will see CDR records as shown in following screenshot:



## Fill the Data Lake Exercise 2: Explore how metadata injection can be used to automate data onboarding

This second exercise steps you through creating a transformations that uses the “metadata injection” method of ingestion. You will parse and ingest 3 different cell phone tower log CSV files into Hadoop. Each CSV file has a different format, so we leverage Metadata Injection to parse all 3 files and output to Hadoop in one common format using only **one** transformation template. Each log file name will be matched to a sheet in a spreadsheet that contains log file metadata (e.g. field names, delimiter, enclosure, etc.). The metadata injection step injects that metadata into a transformation template to dynamically configure the csv input and other steps.

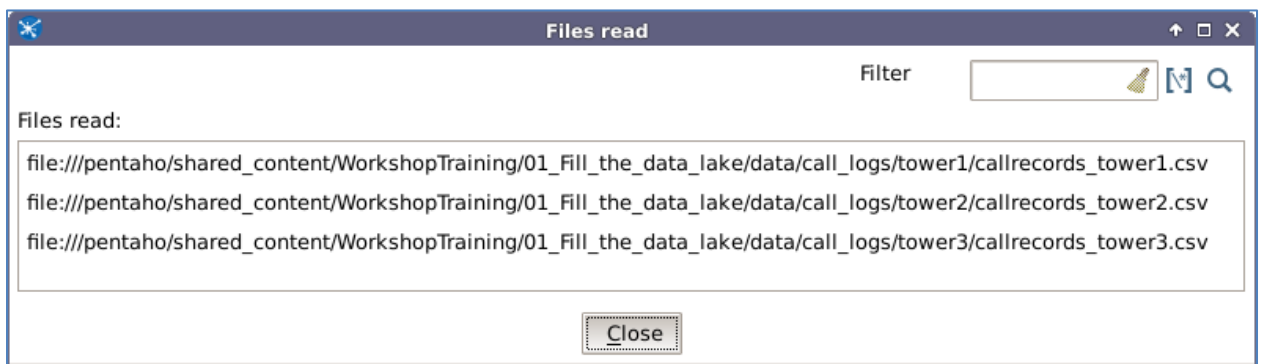
We will be creating two transformations in this exercise named: `t_process_tower_logs` and `t_process_single_log_via_metadata_injection`

1. In Spoon, Create a new transformation

2. From the **Input** folder, select and drag the **Get File Names** step onto the canvas
3. Double click the **Get File Names** step to update its properties.
4. In **File or directory**, specify the directory where our files are stored:  
File:///home/demouser/pentahobdvm/pentaho/content/evaluation/01\_Fill\_the\_data\_lake/data/call\_logs
5. Click **Add** to make sure it adds the directory under **Selected Files**.
6. In the **Selected Files** window, provide **Wildcard (RegExp)** = `.*csv*`
7. Ensure **Include Subfolders** is set to **Y**

Selected files:				
	File/Directory	Wildcard (RegExp)	Exclude wildcard	R
1	/home/demouser/pentaho-bigdatavm/content/evaluation/01_Fill_the_data_lake/data/call_logs/	.*csv		N

8. If everything is set up correctly, you should see a list of files when clicking on **Show filename(s)**



9. Next, we are going to add a **Transformation Executor** step in order to call another transformation we will be building. From the **Flow** folder in **Design** view, select and drag the **Transformation Executor** step onto the canvas.
10. Build a hop between the **Get File Names** and **Transformation Executor** steps.
11. Save this transform as `t_process_tower_logs`



Now we are going to build the transform which will be called by the Transformation Executor step above. In this transform, we will use a Spreadsheet to track the metadata for each log file and dynamically inject it. We can then use 1 transformation to load all log files.

12. Start a new transformation.



13. From the **Job** folder, select and drag **Get Rows from result** step onto the canvas. Double click on the step to modify its **properties**. Add two field names: `filename` and `short_filename`. Set the **type** to string. Set the **length** to 500. The **properties** should resemble the screenshot below.

Fieldname	Type	Length	Precision
1 filename	String	500	
2 short_filename	String	500	

14. Change the **Step Name** to Get Log File Name and Click **OK**
15. From the **Input** folder, select and drag **Microsoft Excel Input** step onto the canvas. Double-click the step to modify its **properties**.

Note: no need to create any hops here yet.

- Name the step Log File Metadata Definition.
- For the **Spread Sheet Type (engine)** set it to Excel 2007 XLSX (Apache POI)
- For **File or Directory** enter the following file and then click **Add**. The file is the spreadsheet that contains the metadata description of each log file.

File:///home/demouser/pentahobdvm/content/evaluation/01\_Fill\_the\_data\_lake/data/Tower\_logs\_metadata.xlsx

- Click the **Fields** tab, then click **Get fields from header row...** to fill the fields. It should look like the screenshot below. If there are duplicate fields created then you will need to remove them.

Name	Type	Length	Precision	Trim type	Repeat	Format	Currency	Decimal	Grouping
1 name	String	-1	-1	none	N				
2 type	String	-1	-1	none	N				
3 mask	String	-1	-1	none	N				
4 length	Integer	-1	-1	none	N	#			
5 precision	Integer	-1	-1	none	N	#			
6 delimiter	String	-1	-1	none	N				
7 enclosure	String	-1	-1	none	N				
8 header_line_present	String	-1	-1	none	N				

- e. Click the **Additional output fields** tab and enter `log_filename` for the field **Sheetname field**
- f. Click **OK**
16. From the **Joins** folder, select and drag the **Join Rows (cartesian product)** step onto the canvas.
17. Create a Hop between the **Log File Metadata Definition** step and the **Join Rows (cartesian product)** step. Create a Hop between the **Get Log File Name** step to the **Join Rows (cartesian product)** step. The result should look like the below.



18. Double-click the **Join Rows (cartesian product)** step to set its properties.
19. Under **The condition** click the field and select `short_filename`. Click on the second field and select `log_filename`. The result should look like the following screenshot.

Join rows

Step name: Join Rows (cartesian product)

Temp directory: %%java.io.tmpdir%% Browse...

TMP-file prefix: out

Max. cache size (in rows): 500

Main step to read from: ▼

The condition:

short\_filename = log\_filename

Help OK Cancel

20. Click **OK**
21. From the **Flow** folder, select and drag the **ETL Metadata Injection** step onto the canvas. Double-click the step to modify its properties. Note: you may get an error on this step. Please click OK if you get an error.
22. Create a **Hop** between **Get Log File Name** and **ETL Metadata Injection**. When you see the warning message click **Copy**.

23. Create a **Hop** between **Join Rows (cartesian product)** and **ETL Metadata Injection**

24. In this step, we will be referencing a pre-defined template. Under **Use a file for the transformation template**, click on **Browse** to select the following file:

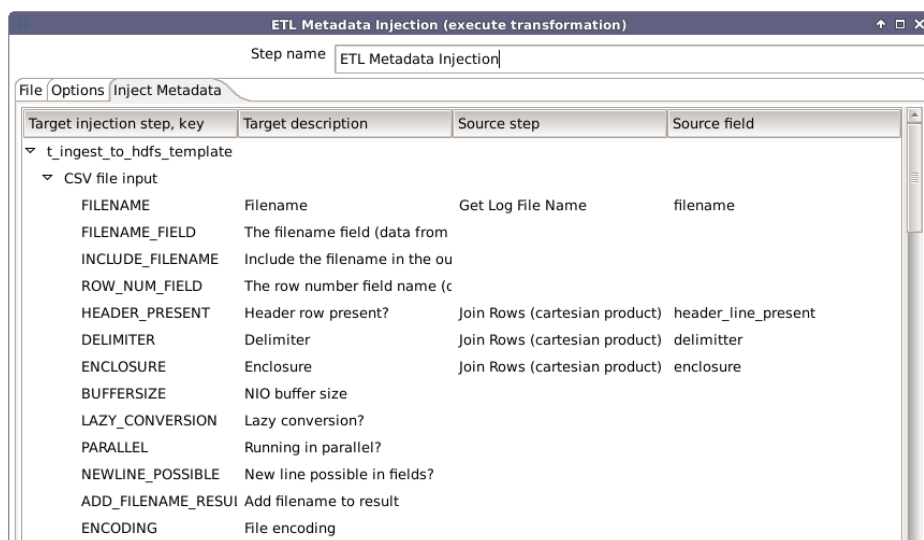
[file:///home/demouser/pentahobdvm/pentaho/content/evaluation/01\\_Fill\\_the\\_data\\_lake/Solutions/t\\_ingest\\_to\\_hdfs\\_template.ktr](file:///home/demouser/pentahobdvm/pentaho/content/evaluation/01_Fill_the_data_lake/Solutions/t_ingest_to_hdfs_template.ktr)

The template transform inputs the incoming file and writes it into a standard comma separated format (without headers) in HDFS

25. Under the **Inject Metadata** tab inside the properties window, set the following properties. This will set the configuration for the CSV file input step of the transformation being called with fields that are dynamically injected by this metadata injection step.

- Click **FILENAME** and select **Get Log File Name : filename**
- Click **HEADER\_PRESENT** and select **Join Rows (cartesian product) : header\_line\_present**
- Click **DELIMITER** and select **Join Rows (cartesian product) : delimiter**
- Click **ENCLOSURE** and select **Join Rows (cartesian product) : enclosure**

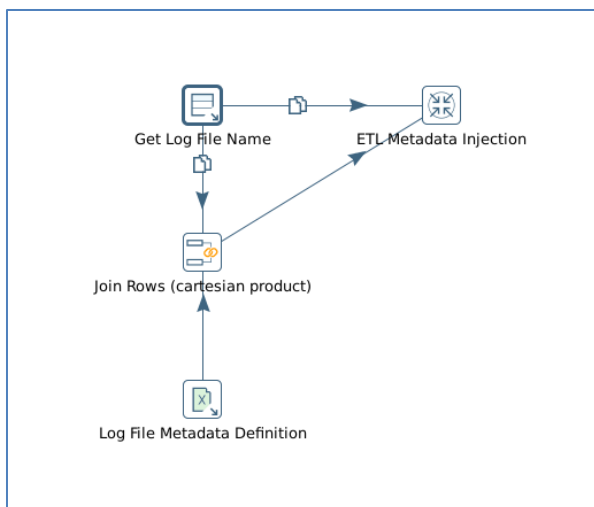
26. If everything is set up correctly, you should see the metadata as in the image below:



27. Click **OK** to close the **properties** window.

28. Save this transform to

`file:///home/pentahobdvm/pentaho/content/evaluation/01_Fill_the_data_lake/student_files` as `t_process_single_log_via_metadata_injection`



29. We are done with this transform. Note: **Please don't run this transform.** It will be executed by the `t_process_tower_logs` transform we built earlier.
30. Let's get back to the `t_process_tower_logs`. We need to modify the **Transformation Executor** step in that transform to execute the `t_process_single_log_via_metadata_injection` transformation.
31. In the `t_process_tower_logs` transformation double-click on **Transformation Executor** step to update the properties.
  - a. **File Name:**  
`file:///home/pentahobdvm/pentaho/content/evaluation/01_Fill_the_data_lake/student_files/t_process_single_log_via_metdata_injection.ktr`
  - b. **Parameters:**

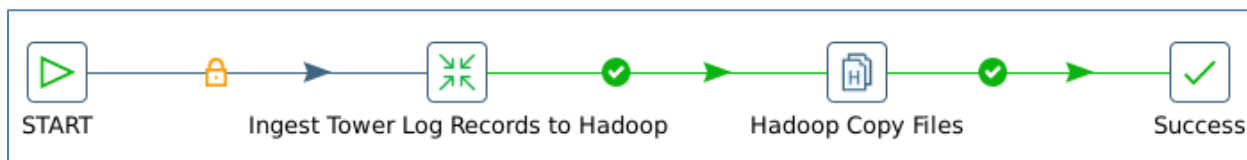
Parameters			
Row grouping			
Execution results			
Result rows			
Result files			
	Variable / Parameter name	Field to use	Static input value
1	filename	filename	
2	short_filename	short_filename	

- c. Make sure that **Inherit all variables from the transformation** is checked
32. Save and Execute this transformation



33. Using Hue, navigate to `/BDO/callrecords/input` to see the files that have been copied. You can click to view each file to see that they now have a common format.

34. Last thing we have to do in this exercise is modify the Job we built in exercise 1 to take advantage of the metadata injection process we created.
35. Open the **Fill Data Lake Exercise 1** job from your student files.
36. From the **General** folder select and drag the **Transformation** step onto the canvas.
37. Set that step between **Start** and **Hadoop Copy files** steps. To insert a step, you will need to delete the original hop and then create a hop between **Start** and **Transformation** as well as **Transformation** to **Hadoop Copy Files**.
38. Double-Click the **Transformation** Step. Rename it to `Ingest Tower Log Records to Hadoop`.
39. Transformation filename should be `t_process_tower_logs.ktr`. Make sure to browse to select the transformation you built earlier in this exercise. You will need to provide the full path.
40. Your job should look like this now:



41. Save this job as **Fill Data Lake Exercise 1 – MI\_Updated**.
42. Execute the job

## Use Case 2: Create a Data Refinery

### What is it?

A data refinery is an enterprise solution for processing and blended data that is governed, analytics-ready, and on-demand. The data refinery is powered by on-demand orchestration for blending traditional data and Big Data, and it is a first step toward Governed Data Delivery. Governed Data Delivery is defined as the delivery of blended, trusted and timely data to power analytics at scale regardless of data source, environment, or user role. It lays the groundwork for seamless end user exploration and analysis of validated data blends from across the organization.

### Why do it?

These three core data delivery needs that are only being met on a limited basis in the market today:

- Orchestrate on-demand processing, blending, and modeling of user requested data sets to accelerate time to value in complex analytics initiatives.
- Ensure proper data governance during the delivery process, such that risk is minimized and confidence is increased in data-driven decisions.
- Provide blended and enriched data in the end user format of choice, so that business users can be more productive in deriving insight from diverse data.

## Value of Pentaho

Pentaho's highly scalable data integration engine, managed through its intuitive end user interface, provides the 'glue' between the different data sources and big data stores in this architecture. The entire process outlined can be triggered on-demand with the following key capabilities: blending & orchestration, automatic modeling and publishing, and governance.

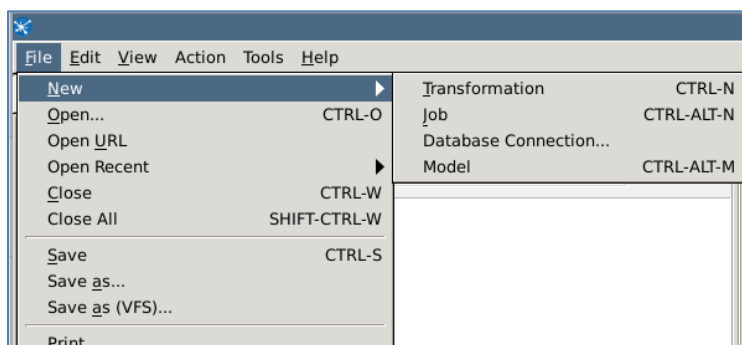
## What You Will Accomplish

You will complete **4 exercises** to create a data refinery that processes and blends a combination of data sources from previous exercises.

## Create a Data Refinery Exercise 1: Transform CDR data and blend with geography data

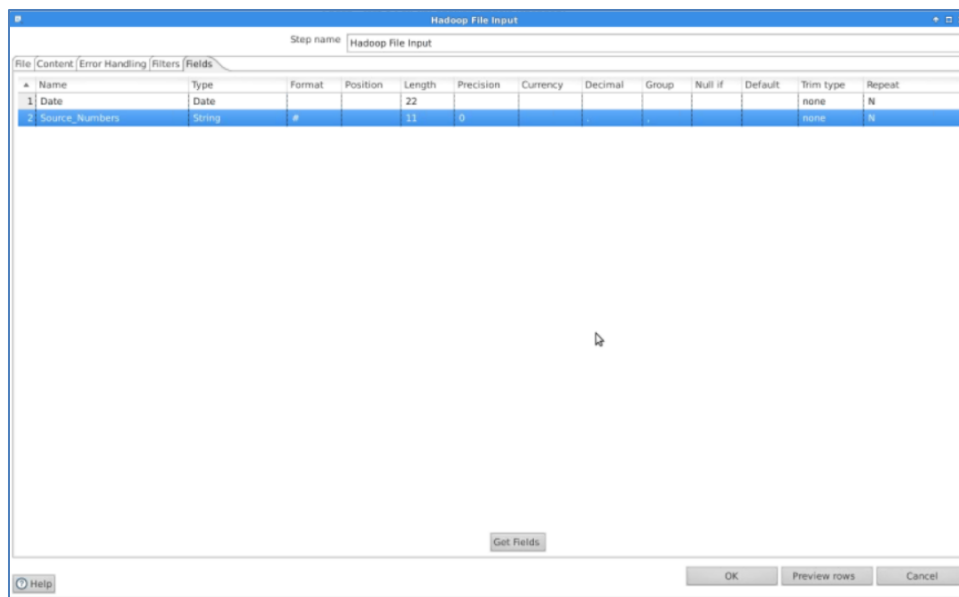
This exercise steps you through creating an advanced transformation to process and blend CDR data. You will process and blend file data and save it back to HDFS. The transform will include working with the **Stream Lookup** step to blend master geographic data. You will also add additional steps to enrich, filter, and sort the data as needed to complete the transformation. You will execute this transformation on Spark.

1. From the main Spoon menu choose **File | New | Transformation**

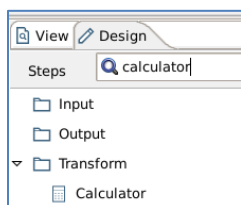


2. From the **Design** tab on the left, expand the **Big Data** folder; then, select and drag **Hadoop File Input** onto the **canvas**
3. Double-click on **Hadoop file Input** step to update its properties.
4. Under the **File** tab create 1 row with the following values:
  - a. **Environment:** CDH
  - b. **File:** /BDO/callrecords/input/callrecords\_10years.csv
  - c. Our source file does not include a header row, so **Content** tab make sure to leave the **Header** property unchecked
  - d. At the bottom of the **Fields** tab, click **Get Fields**. When prompted for a **sample size**, type 0. Click **OK**

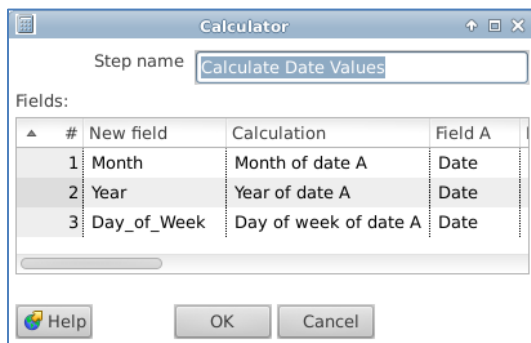
Note: You will notice that both fields are coming in as String. We will need to change that.
  - e. In the first row change the **Name** to Date, **Type** to Date and the **Format** to yyyy/MM/dd HH:mm:ss:SSS
  - f. In the second row, change the **Name** to Source\_Number, **Type** to String and **Length** to 11
  - g. The properties window should look like the screenshot below



5. Click **OK** to return to the canvas.
6. Select and drag the **Calculator** step onto the canvas. In order to locate the **Calculator** step, you can search for it in the **Steps** search box



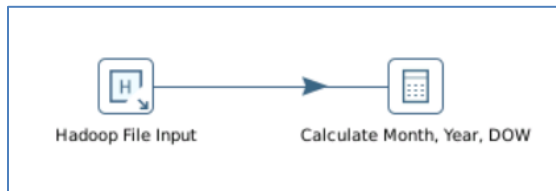
7. Double-click on the **Calculator** step to open its properties.
8. Change the **Step name** to Calculate Month, Year, DOW
9. In the **Fields** section add Month, Year and Day\_of\_Week as **New Fields**. In **Field A**, select Date field from the drop down. The properties for these fields should match the following screenshot:



10. Click **OK** to return to the canvas.



11. Create a hop from the **Hadoop file Input** step to the **Calculate Month, Year, DOW** step.
12. Click **OK** to return to the canvas. The first part of your transformation should now match the following image:



Next, we'll need the location information. To derive location information from the data, we must know the Area Code within the phone number.

13. Select and drag the **Strings Cut** step onto the canvas.
14. Create a hop between the **Calculate Month, Year, DOW** step and the **Strings Cut** step.
15. Double-click on the **Strings Cut** step to open its properties.
16. In the **Step name** field, type `Extract Area Codes`. In the **Fields to cut** section select `Source_Number` as the **In Stream field** and type `Area_Code` as the **Out Stream field**. The properties for these fields should match the following screenshot:

String Cut

Step name:

The fields to cut:

#	In stream field	Out stream field	Cut from	Cut to
1	Source_Number	Area_Code	1	4

Buttons: Help, OK, Get fields, Cancel

Note: To select the **In Stream field**, you can click on the drop down and select it from the list of fields available. If no fields are available, click **Get Fields** to get the list.

String Cut

Step name:

The fields to cut:

#	In stream field	Out stream field	Cut from	Cut to
1	Date			

Buttons: Get fields, Cancel

17. Click **OK** to return to the canvas.

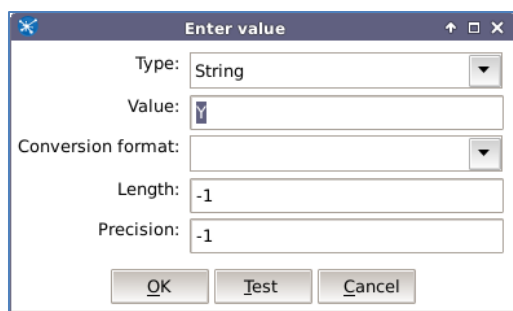


Now that we know the Area Code, we will use a lookup file to map the Area Code to State, Country, and Time Zone. Before we do that, let's verify our Area Codes are numeric and discard the records where they are not.

18. From the **Scripting** Folder, select and drag **Regex Evaluation** step onto the canvas
19. Create a **hop** from **Extract Area Code** step to the **Regex Evaluation** step.
20. Rename the step to **Check AC**. In **Field to evaluate** property, select **AreaCode** from the drop down. The Result field should be named **AreaCodeNumeric**. In the **Regular Expression** window, type the following expression: `^[0-9]+$`
21. The properties of the
22. **Check AC** step should look like this:

	New field	Type	Length	Precision	Format	Group	Decimal	Currency	Null If	Default	Trim
1											

23. From the **Flow** folder, select and drag the **Filter Rows** step onto the canvas
24. Create a hop between **Check AC** step and **Filter rows** step.
25. Double-click the **Filter rows** step to update its properties.
26. Rename this step to **Trash non-numeric**.
  - a. In **The Condition** window, choose **AreaCodeNumeric** as the field.
  - b. Click on **value**. It will bring up a pop-up window where you can set the value and the type. Keep **Type** as **String** and **Y** as the **value**. The value properties should look like this:

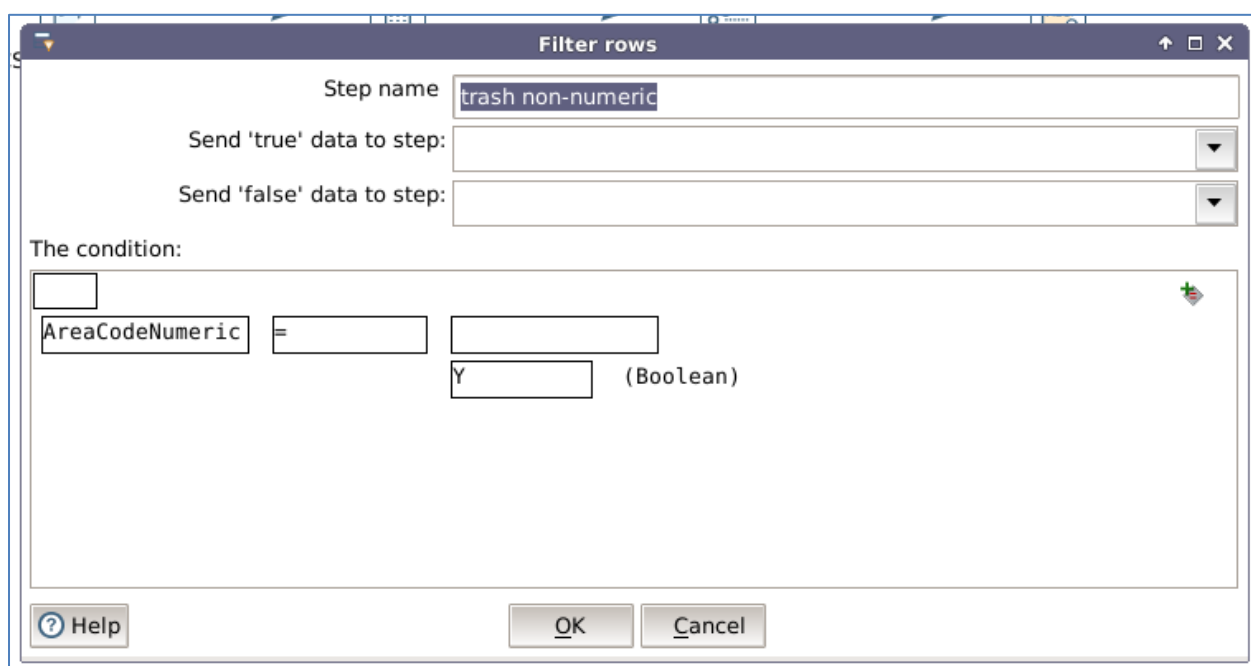


The 'Enter value' dialog box is shown with the following fields:

- Type: String
- Value: Y
- Conversion format: (empty)
- Length: -1
- Precision: -1

Buttons at the bottom: OK, Test, Cancel.

- c. Click **OK**
- d. The properties of Trash non-numeric step will now look like this:



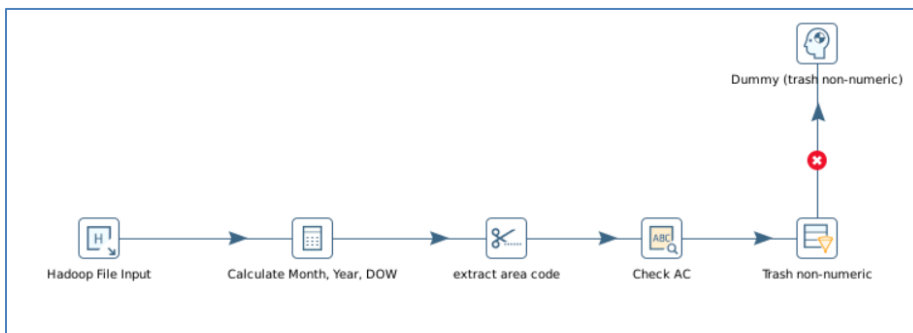
The 'Filter rows' dialog box is shown with the following fields:

- Step name: trash non-numeric
- Send 'true' data to step: (empty)
- Send 'false' data to step: (empty)
- The condition:
  - AreaCodeNumeric = Y (Boolean)

Buttons at the bottom: Help, OK, Cancel.

27. Expand the **Flow** folder; then, select and drag **Dummy(do nothing)** onto the canvas, directly above the **Trash non-numeric** step. Double-click to open the properties of this step. Rename the step to **Dummy(Trash non-numeric)**
28. Create a hop between the **Trash non-numeric** step and the **Dummy (Trash non-numeric)** step. When prompted, select **Result if FALSE** option

29. Your transform should look similar to this image now:

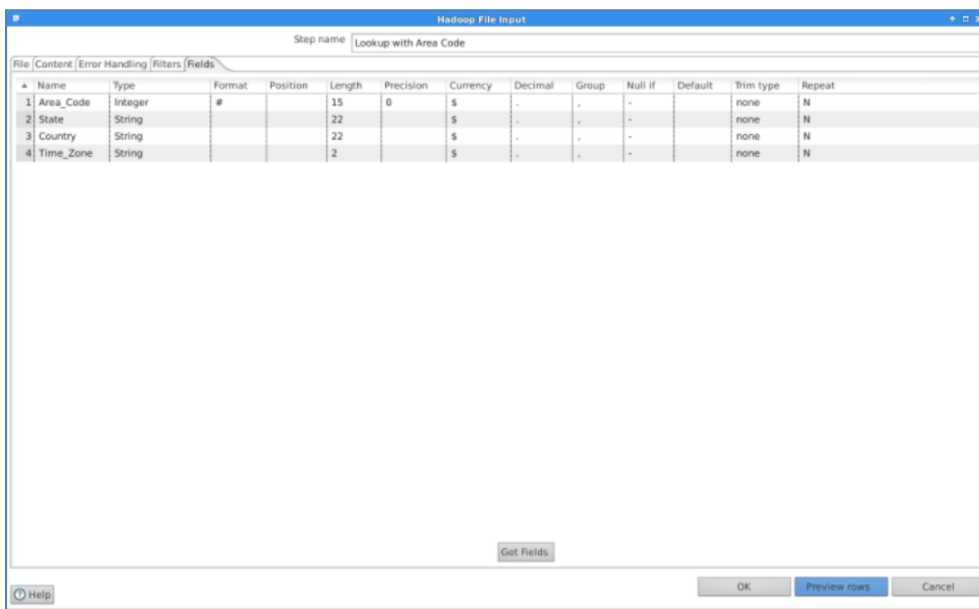


30. Expand the **Lookup** folder; then, select and drag **Stream Lookup** onto the canvas.

31. Expand the **Input** folder and select and drag the **Hadoop File Input** step onto the canvas directly above the Stream Lookup step.

32. Double-click on the **Hadoop File Input** step to open its properties. Update the properties as follows:

- Step Name:** Lookup Area Codes
- Environment:** CDH
- File:** /BDO/callrecords/reference/areacodes.csv
- This file does contain the header row, so make sure to check the box next to **Header** on the **Content** tab.
- Click the **Fields** tab and click on **Get Fields** to get the field names and data types. In the sample size, type 0.
- The Properties for this lookup file should resemble the image below:



- g. Click **OK** to return to the canvas.
33. Create a hop from the **Trash non-numeric** step to the **Stream Lookup** step. When prompted choose, **Main Output of Step**, to complete the hop connection.
34. Create a hop from the **Look Up Area Codes** step to the **Stream Lookup** step and choose **Main Output of Step**, to complete the hop connection.
35. Double-click on the **Stream Lookup** step to open its properties.
- In the **Lookup step** select **Lookup Area Codes**.
  - In the **Key(s) to Lookup Value(s)** section select **AreaCode** in the **Field** column and select **Area\_Code** as the **LookupField** column.
  - Click the **Get Lookup Fields** button (bottom right) to populate the **Fields to retrieve** section at the bottom.
  - Rename your Stream Lookup to **Lookup Country/State**. Highlight and delete **Area\_Code** from the fields to retrieve section. Your **Stream Lookup** dialog box should now look like this:

Stream Value Lookup

Step name:

Lookup step:

The key(s) to look up the value(s):

	Field	LookupField
1	AreaCode	Area_Code

Specify the fields to retrieve :

	Field	New name	Default	Type
1	State			String
2	Country			String
3	Time_Zone			String

- e. Click **OK** to return to the canvas.



Now let's convert the numeric values of day of week from our data set to the actual Day Of Week values.

36. Expand the Transform folder and drag the **Value Mapper** step onto the canvas.
37. Create a hop between **Lookup Country/State** step and the **Value Mapper**
38. Double Click the **Value Mapper Step**. Rename it to **Day of Week**

39. Update the properties of this step as follows:

- a. Fieldname to use: `DayOfWeek`
- b. Target Field name: `Weekday`

The Field Values should look like this:

Field values:		
	Source value	Target value
1	1	Sunday
2	2	Monday
3	3	Tuesday
4	4	Wednesday
5	5	Thursday
6	6	Friday
7	7	Saturday



We need to apply a filter to the data to ensure we only get calls placed in the USA and calls made on weekends only, Saturday and Sunday. Calls placed outside of the US and on days Monday through Friday will be discarded.

40. Expand the **Flow** folder; then, select and drag **Filter Rows** onto the canvas.
41. Create a hop between the **Day of Week** step and the **Filter Rows** step.
42. Double-click on the **Filter Rows** step to open its properties.
  - a. Rename this step to **U.S.-only calls**.
  - b. Click **OK** to return to the canvas.
43. From the **Flow** folder, select and drag **Dummy (do nothing)** onto the canvas above the **U.S.-only calls** step.
44. Create a hop from the **U.S.-only calls** step to the **Dummy (do nothing)** step. Select `Result is False`.
45. Double click the **U.S.-only calls** step to open its properties.
  - a. Under **The condition** section select the **<field>** on the left. From the pop-up window select `Country` and then click **OK**.
46. Add another **Filter Rows** Step to the canvas. Name it **Weekend Only**. Create a Hop between **US Calls only** and **Weekend Only**. Create a Hop between **Weekend Only** and **Dummy (do nothing)** step used above. Double click the **Weekend Only** step to open its properties.
  - a. Click in the middle box and select `"="` as the function and then click **OK**.

- b. Click the bottom right **<value>** box and type UNITED STATES as the **Value**.



The text comparison is case sensitive. Be sure to enter all caps.

- c. Click the **null = [ ]** area to add the condition.
- d. Select the **<field>** on the left. From the pop-up window select Day\_of\_Week and then click **OK**.
- e. Click in the middle box and select **>=** as the function.
- f. Click the bottom right **<value>** box and type 6 as the **Value**.

- g. Click **OK** to return to the canvas.



There are some Area Code values of 000. We will change those to Null values.

47. Expand the **Utility** folder and drag **Null if...** step onto the canvas.
48. Create a hop between the **U.S.-only calls** step and the **Null if...** step. Select **Result is true** to complete the hop.

49. Double click the **Null if...** step to open its properties.
  - a. Rename this step to **Replace Nulls**
  - b. Under the **Fields** section select **Area\_Code** and type **000** in the **Value to turn to NULL** field, then click **OK**.
50. From the **Transform** folder, select and drag the **Select Values** step. Create a hop between **Replace Nulls** and the **Select Values** step.



To generate a call count measure, we will add a constant value of 1 to each record so we can aggregate the number of calls with analysis reports.

51. Expand the **Transform** folder; then, select and drag the **Add Constants** step onto the canvas.
52. Create a hop between the **Select Columns** step and the **Add Constants** step.
53. Double click the **Add Constants** step to open its properties.
  - a. In the **Step Name** field, type **Add Call Count**.
  - b. Add a **Field** named **Calls** as a **Type** **Integer** with a **Value** of **1** as illustrated here:

Add constant values										
Step name <input type="text" value="Add Call Count"/>										
Fields :										
▲	Name	Type	Format	Length	Precision	Currency	Decimal	Group	Value	Set empty string?
1	Calls	Integer							1	N

54. From the **Big Data** folder, select and drag the **Hadoop File Output** step onto the canvas.
55. Double-click the **Hadoop File Output** step to update its properties
  - a. Under the **File** tab, enter the following folder name for the **Folder/File** field.  
`/demo/data_refinery/callrecords_10years`
  - b. Be sure that **Create Parent** folder is checked.
  - c. Under the **Fields** tab, click **Get Fields**. The fields should look like the following.



Hadoop File Output

Step name: Hadoop File Output

File Content Fields

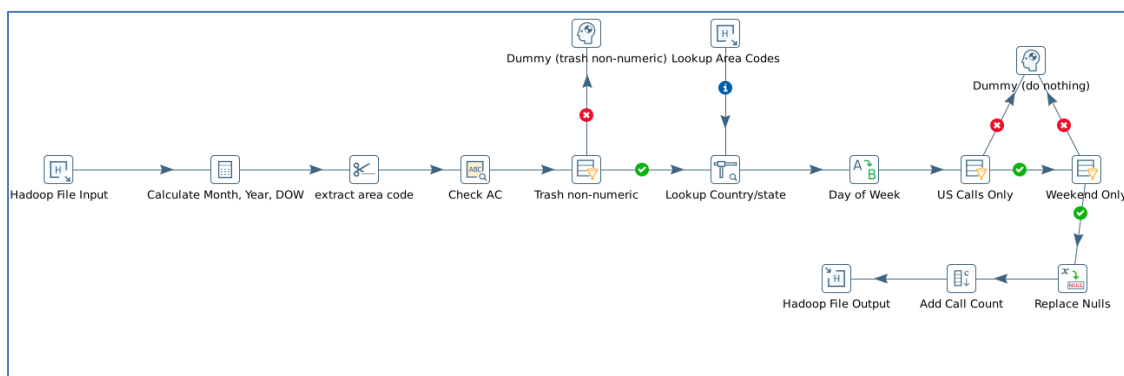
#	Name	Type	Format	Length	Precision	Currency	Decimal	Group	Trim Type
1	Date	String		22					none
2	Source_Number	Integer		15	0				none
3	Month	Integer			0				none
4	Year	Integer			0				none
5	DayofWeek	Integer			0				none
6	AreaCode	String		100					none
7	AreaCodeNumeric	Boolean							none
8	State	String		22					none
9	Country	String		22					none
1..	Time_Zone	String		2					none
1..	Weekday	String		9					none
1..	Calls	Integer			0				none

Get Fields Minimal width

Help OK Cancel

56. Create a hop between the **Add Call Count** step and the **Hadoop File Output** step.

57. Your completed transformation should match the following image:



58. From the **File** menu, choose **Save As**. Save this transformation as `t_call_vol_analysis_spark` in the following directory:  
`file:///home/demouser/pentahobdvm/pentaho/content/evaluation/02_create_data_refinery/student_files`

59. Click the play icon  and run this transformation using Pentaho Local.

- From the Firefox bookmarks click **Hue**. On the sign in page, and sign-in with **demouser / demouser** if you are not already signed in
- In the far right corner, click **HDFS Browser** to locate the files
- Navigate the output directory `/demo/data_refinery` to see the output file.

60. Click the play icon  and run this transformation using Spark

- Select **Spark** from the run configuration dropdown list. Click **Run**.

61. Create another tab in the firefox browser, click **Hue** and click on **Job Browser** to see that the job is accepted. You can monitor the job's progress from here or select the **Yarn Application** bookmark to see the jobs progress.



The job may take some time to run. There is an initial amount of time to package any job and send it to Spark. Larger jobs will see a significant performance benefit from running on Spark. Small jobs like this one will take longer to run on Spark than if it were run on the Pentaho Server.

62. When the job is complete, click back to the browser tab with **Hue** and the **HDFS Browser**
63. Navigate to the following directory: `/demo/data_refinery` and notice the directory with the name of the output file. Within the directory, you see that there are 2 files since the job was distributed and the output file was broken into more than 1 file and placed in a directory.

## Create a Data Refinery Exercise 2: Use Pentaho with Impala to blend CDR and IoT Data

This first exercise steps you through the process of creating a PDI job to execute a transformation for loading data to HDFS and creating an Impala table for querying with SQL.

1. From the main menu choose **File | New | Job**
2. From the **File** menu, choose **Save**.
3. In the Name field, specify `SDR_GeoLocation_Impala_Job` and save to this directory:  
`file:///home/demouser/pentahobdvm/pentaho/content/evaluation/02_create_data_refinery/student_files`.
4. From the **Design** tab on the left, expand the **General** folder and drag the following three steps on the canvas: **START**, **Success** and **Transformation**.



We need to create a directory on the Hadoop File System (HDFS) to house our geolocation data file.

5. Expand the **File management** folder and drag the **Create a folder** step onto the canvas.
6. Create a hop between the **START** and **Create a folder** steps.
7. Double click the **Create a folder** step to edit its properties. In the **Folder name** field type `hdfs://pentahobdvm.localdomain:8020/demo/data_refinery`
8. Uncheck the **Fail if folder exists** box.
9. Click **OK** to return to the canvas.



Next, we need to set HDFS permissions to allow Impala write privileges.

10. Expand the **Scripting** folder and drag the **Shell** step onto the canvas to the right of **Create a folder** and double click to open.
11. In **Job entry name** enter **Set HDFS Directory Permissions**.
12. Check the box next to **Insert script**.
13. In **Working directory**, enter `/tmp`

14. Switch to the **Script** tab and copy/paste the following:

```
echo 'demouser' | sudo -Sk -u hdfs hadoop fs -chmod -R 777 /demo/data_refinery
```

15. Click **OK** to return to the canvas.

16. Create a hop between the **Create a folder** and **Set HDFS Directory Permissions** steps

17. Create a hop between the **Set HDFS Directory Permissions** and **Transformation** steps



Once the target directory is created and permissions for Impala access is granted, we need to create and load a text file with geolocation data to HDFS.

18. Double Click the **Transformation** step to edit its properties.

19. In the **Name of job entry** box, type Load Geolocation Data.

20. On **Transformation Specification** tab click the browse icon for the **Transformation filename** field and browse to select the following file:

```
/pentaho/content/evaluation/02_create_data_refinery/solutions/SDR_GeoLocation_HDFS.ktr
```

Note: to save time and reduce redundant work, you are using an *existing* transformation to load the geolocation data to HDFS.

21. Click **OK** to return to the canvas.



We will want to load data to an Impala table. But first we should validate that the target table exists within Impala. The table is named 'call\_detail\_geo'.

22. Expand the **Conditions** folder and drag the **Table Exists** step onto the canvas.

23. Create a hop between the **Load Geolocation Data** and **Table Exists** steps.

24. Double Click the **Tables Exists** step to edit its properties.

25. In the **Job entry name** field type Does Impala table exist?

26. In the **Connection** field select Impala.

Note: this connection to Impala has already been established and is available for use.

27. In the **Table Name** field type call\_detail\_geo.

28. Click **OK** to return to the canvas.



If the Impala table does exist, we will truncate the table data, and load the geolocation data that we previously loaded to HDFS. To load the data, we will execute a SQL script.

29. Expand the **Scripting** folder and drag a **SQL** step to the right of the **Does Impala table exist?** step.

30. Rename this **SQL** step to Load Impala Table



If the Impala table does not exist, we will create the table before loading the geolocation data using a SQL script.

31. Drag a second **SQL** step from the **Scripting** folder and place it below the **Does Impala table exist?** step.

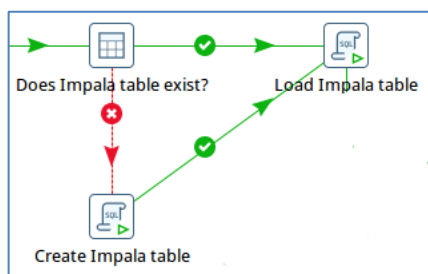
32. Rename this **SQL** step to Create Impala Table

33. Create a hop from the **Does Impala table exist?** step to the **Load Impala table** step to the right. This hop should be green in color indicating this path will be taken if the previous step returns a true evaluation.

Tip: To change the hop color, click the green arrow.

34. Create a hop from the **Does Impala table exist?** step to the **Create Impala table** step below. This hop should be red in color indicating this path will be taken if the previous step returns a false evaluation.

35. Create a hop from the **Create Impala table** step to the **Load Impala table** step. The hops should match the following image:

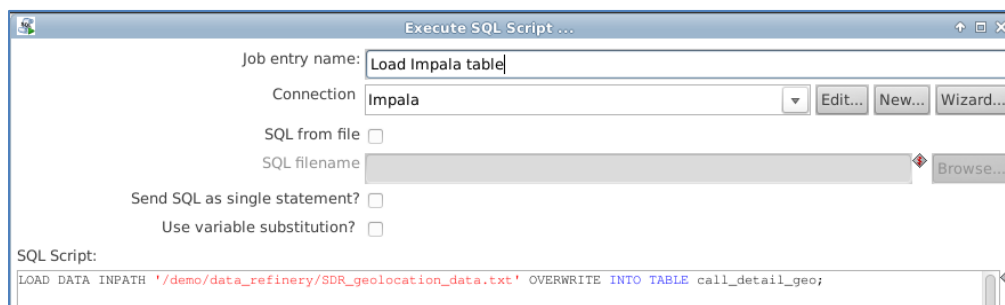


36. Double click the **Load Impala table** step to edit its properties.

37. In the **Connection** field select Impala.

38. In the **SQL Script** section type the following: `LOAD DATA INPATH '/demo/data_refinery/SDR_geolocation_data.txt' OVERWRITE INTO TABLE call_detail_geo;`

39. Your SQL script step should now look like this:



40. Click **OK** to return to the canvas.

41. Double click the **Create Impala table** step to edit its properties.

42. In the **Connection** field select Impala.

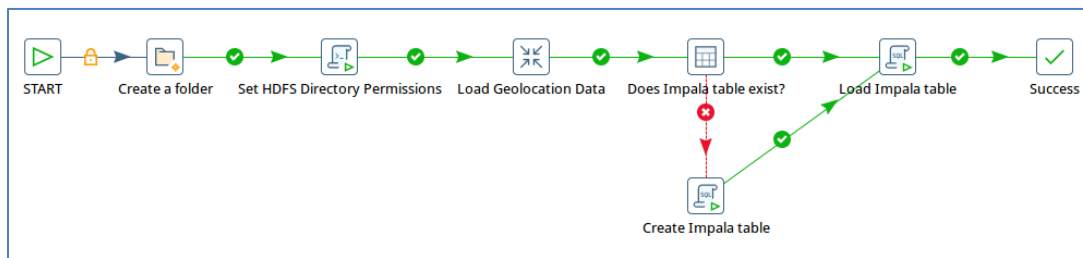
43. In the **SQL Script** section type or copy/paste the following SQL command:


```
CREATE TABLE call_detail_geo
(
    calldate VARCHAR(26),
    source_number VARCHAR(11),
    home_latitude DOUBLE,
    home_longitude DOUBLE,
    distance DOUBLE,
    direction VARCHAR(4),
    location_category VARCHAR(20),
    new_lat DOUBLE,
    new_long DOUBLE
)
row format delimited
fields terminated by '|'
STORED AS TEXTFILE;
```

44. Click **OK** to return to the canvas.

45. Create a hop from the **Load Impala table** step to the **Success** step.

46. Your PDI job should now look like this:



47. From the **File** menu, choose **Save**.
48. Execute the job by choosing **Action** → **Run** from the main menu or by clicking the run icon .
49. The **Execute a job** dialog will appear. Click the **Launch** button at the bottom.
50. A green check will appear on the **Success** step when the job finishes without errors.
51. Keep this job open as it will be used in the next exercise.

## Create a Data Refinery Exercise 3: Extend the PDI job to blend data and load to multiple locations

Pentaho Data Integration (PDI) allows you to join data from multiple tables, transform it, and load it to multiple locations. We can take advantage of Impala queries to join two large tables into a combined data set. This combined set is loaded into a new combined Impala table and to a PostgreSQL database to enable high performance OLAP analysis.

1. Make sure the job, `SDR_GeoLocation_Impala_Job`, created in the previous exercise is open.
2. To save time we will use an existing job to load call detail records into Impala. From the **General** folder drag the **Job** step onto the canvas.
3. Drag the Job between steps **Set HDFS Directory Permissions** and **Load Geolocation Data**
4. Double click the **Job** step to configure it
  - a. Set **Entry Name** to `Load Call Detail Records`
  - b. Navigate and select the following **Job**

```
file:///home/demouser/pentahobdvm/pentaho/content/evaluation/02_Create_data_refinery/setup/SDR_CallDetailRecords_Setup.kjb
```
  - c. Under the **Options** tab we can select **Local** since we will run this job locally
  - d. Click **OK**
5. Select the **Success** step and delete it.



Now we need to blend our geolocation data with our call data records, and place the combined data set into a new Impala table using a SQL script.

6. From the **Design** tab on the left, expand the **Scripting** folder and drag the **SQL** step onto the canvas below the **Load Impala Table** step.
7. Create a hop between the last step, the **Load Impala Table** step, and the **SQL** step.
8. Double click the **SQL** step to edit its properties. In the **Step name** field type Join CDR to Geo Location Data.
9. In the **Connection** field select Impala.
10. In the **SQL** field type or copy/paste in the following SQL Command:

```
DROP TABLE IF EXISTS call_detail_combined;
CREATE TABLE call_detail_combined
(
    key INT
    , source_number VARCHAR(11)
    , call_date VARCHAR(26)
    , call_month INT
    , call_year INT
    , day_of_week INT
    , area_code INT
    , state VARCHAR(11)
    , country VARCHAR(13)
    , time_zone VARCHAR(8)
    , weekday VARCHAR(8)
    , num_calls INT
    , home_latitude DOUBLE
    , home_longitude DOUBLE
    , distance DOUBLE
    , direction VARCHAR(4)
    , location_category VARCHAR(20)
    , new_lat DOUBLE
    , new_long DOUBLE
);
```

```
INSERT INTO TABLE call_detail_combined
SELECT
    cdr.key
    , cdr.source_number
    , cdr.call_date
    , cdr.call_month
    , cdr.call_year
    , cdr.day_of_week
    , cdr.area_code
```

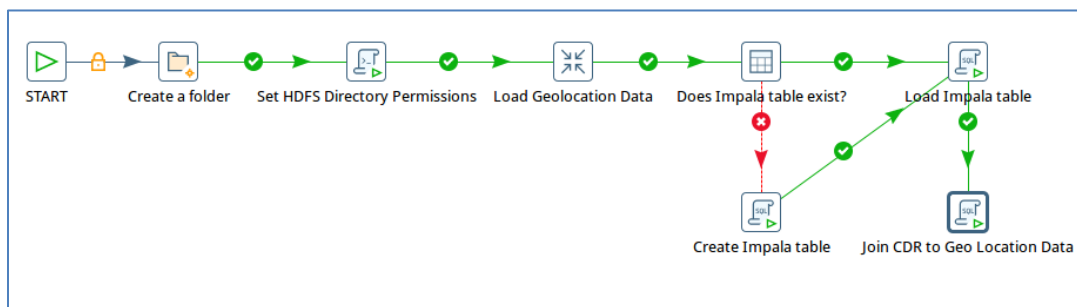


```


, cdr.state
, cdr.country
, cdr.time_zone
, cdr.weekday
, cdr.num_calls
, cdg.home_latitude
, cdg.home_longitude
, cdg.distance
, cdg.direction
, cdg.location_category
, cdg.new_lat
, cdg.new_long
FROM
call_detail_records cdr JOIN call_detail_geo cdg ON
(cdr.source_number = cdg.source_number);

```

11. Click **OK** to return to the canvas. Your job should match the following screenshot.



To enable high-performance OLAP analysis and reporting, we would also like to load the combined data set to a PostgreSQL database.

12. From the **Design** tab on the left, expand the **General** folder and drag the **Transformation** step onto the canvas to the right of the **Load Impala Table** step.
13. Create a hop between the **Join CDR to Geo Location Data** step and the **Transformation** step.
14. Double click the **Transformation** step to edit its properties. In the **Name of job entry** field type Transfer Blended Data to PostgreSQL.
15. On **Transformation Specification** tab click the browse icon  for the **Transformation filename** field and browse to select the following file:

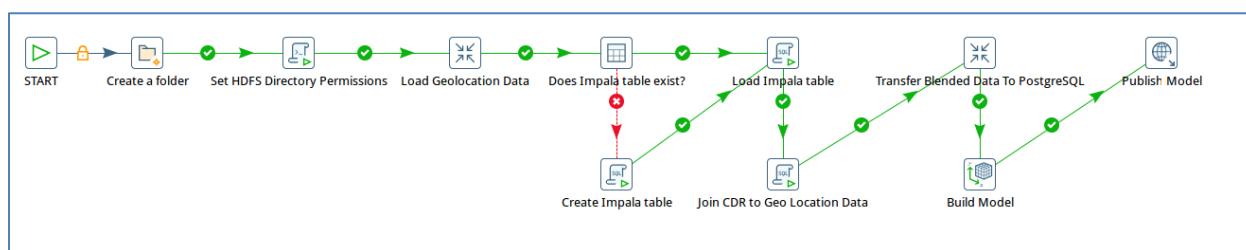
```
file:///home/demouser/pentahobdvm/pentaho/content/evaluation/02_create_data_refinery/solutions/SDR_GeoCDR_Transfer_To_postgres.ktr
```

Note: to save time and reduce redundant work, you are using an *existing* transformation to load the blended data to PostgreSQL.



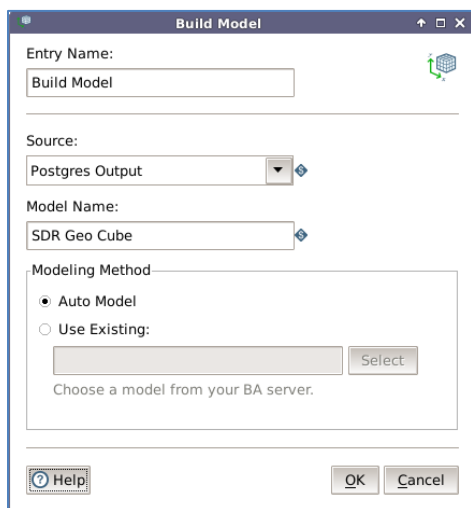
Now that our data is blended, we need to create and publish a model to the Pentaho Analytics server for web-based dimensional analysis and reporting.

16. From the **Design** tab on the left, expand the **Modeling** folder and drag the **Build Model** step onto the canvas below the **Transfer Blended Data to PostgreSQL** step.
17. Also from the **Modeling** folder drag the **Publish Model** step onto the canvas to the right of the **Transfer Blended Data to PostgreSQL** step.
18. Create a hop between the **Transfer Blended Data to PostgreSQL** step and the **Build Model** step.
19. Create a hop between the **Build Model** step and the **Publish Model** step to match the following screenshot.

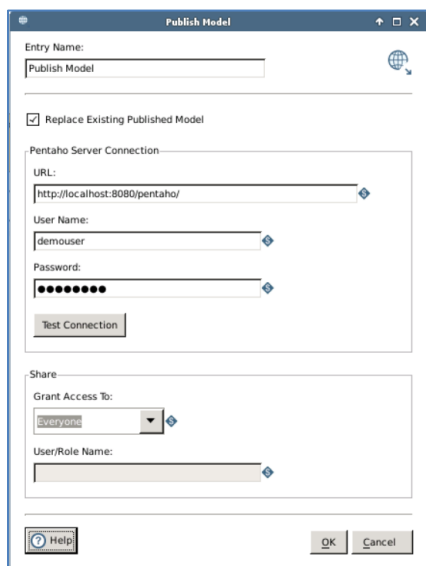


Note: the build model and publish model steps will automatically create and publish a metadata model to the analytics server for web-based dimensional analysis on the blended data created by this job.

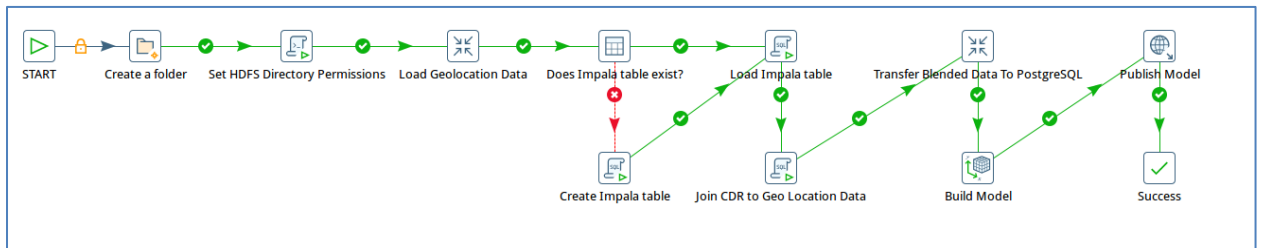
20. Double click the **Build Model** step to edit its properties. In the **Model Name** field type SDR Geo Cube and confirm that the **Source** is set to Postgres Output.



21. Double click the **Publish Model** step to edit its properties.
22. Check **Replace Existing Published Model**.
23. For the **URL** enter: <http://localhost:8080/pentaho/>
24. Login to the Pentaho User Console
25. The remaining fields can be left as the default
26. Click **Test Connection**.



27. From the **Design** tab on the left, expand the **General** folder and drag the **Success** step onto the canvas below the **Publish Model** step.
28. Create a hop between the **Publish Model** step and the **Success** step to match the following screenshot.



29. From the **File** menu, choose **Save**.
30. Execute the job by choosing **Action** → **Run** from the main menu or by clicking the run icon ▶.
31. The **Execute a job** dialog will appear. Click the **Launch** button at the bottom.
32. A green check will appear on the **Success** step when the job finishes without errors.

Congratulations, you have completed the data integration work for implementing the streamlined data refinery (SDR) use case! A later section contains exercises for analyzing the data you loaded.

## Create a Data Refinery Exercise 4: Explore data in PostgreSQL with Pentaho Analyzer

Exercise 6 has two parts showcasing how to use Pentaho Analyzer for analysis against a PostgreSQL database. Pentaho Analyzer offers an easy to use, graphical drag-and-drop design environment that can be used by anyone who wants to dynamically explore data to discover anomalies or trends and create visualizations.

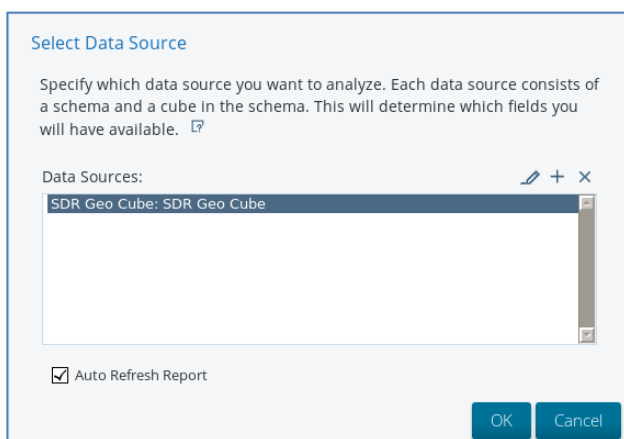
A telecom company is considering introducing a new VOIP service. You need to analyze the geo-location and calling data to determine customer calling patterns to determine the best markets to launch a VOIP pilot service.

In Part One you will analyze data to determine where calls originate: from the house, the neighborhood, in town or during travel. This helps to determine which calling product has the most potential for this market. Later you will plot the calls on a map based on the customers' source area code to determine the highest volume geographical markets to target for a new VOIP service.

### Part 1: Analyze data for a new VOIP pilot service

In this exercise you create a table and column-line combo chart to aggregate call volume and average distance from home by location categories such as: at home, in the neighborhood, in town, during travel, etc.

1. Login to Pentaho User Console, using “demouser” for both the user name and password.
2. Click on the **Create New | Analysis Report** buttons.
3. Select the SDR GEO CUBE data source from the **Data Sources** list.

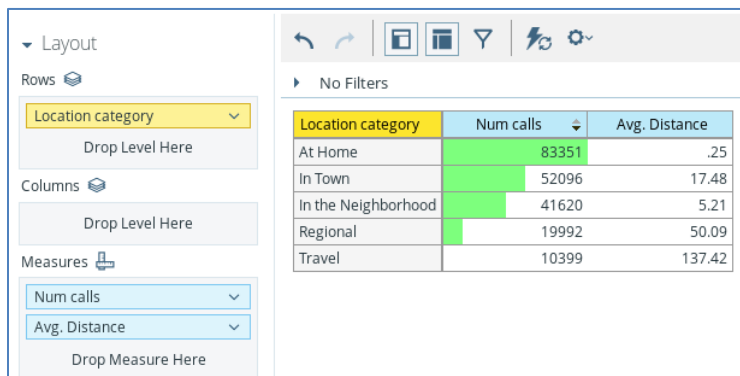


4. This will launch you into a new **Analysis Report** view.

- From the **Available fields** section on the left, double-click or select and drag **Location Category**, **Num calls**, and **Distance** to the canvas.


Notice how the measures automatically aggregate the records in Analyzer.

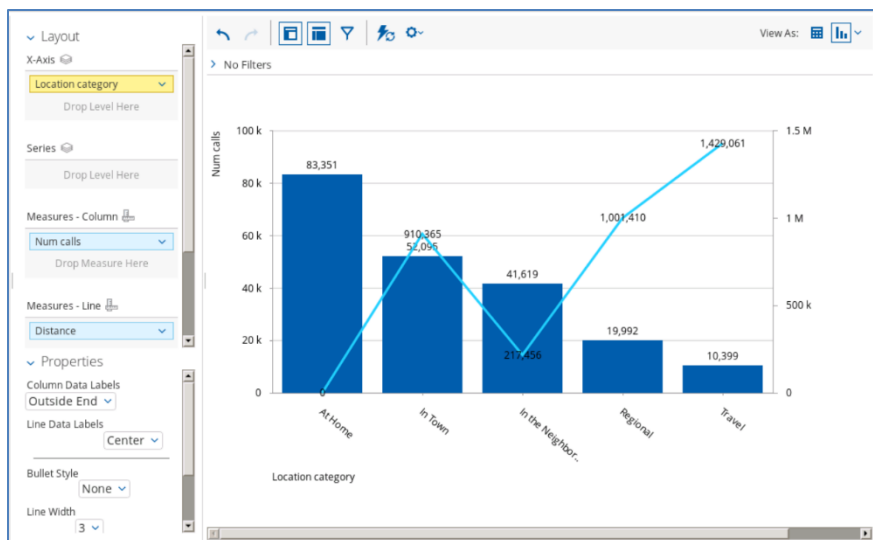
- Sort the **Num calls** field in descending order by right clicking the **Num Calls** column header and selecting **Sort Values High->Low**.
- Add conditional formatting to the **Num Calls** column by right clicking the **Num Calls** column header and selecting **Conditional Formatting | Data Bar: Green**.



Location category	Num calls	Avg. Distance
At Home	83351	.25
In Town	52096	17.48
In the Neighborhood	41620	5.21
Regional	19992	50.09
Travel	10399	137.42

Note that most calls are made **At Home** or **In Town**, and that there is a drop in calls made **In the Neighborhood**. Given the high number of calls made at home, we've verified that a new VOIP calling service for homes may make sense.

- To visualize the data, click the chart drop down in the top right of your screen  and choose **Column-Line Combo**.
- In the **Layout** section, drag **Distance** from the **Measures – Column** drop zone down to the **Measures – Line** drop zone.
- In the **Properties** section, change **Column Data Labels** to **Outside End**, **Line Data Labels** to **Center**, **Bullet Style** to **None**, and **Line Width** to **3**. The resulting chart should match the following image:



- Click the Save icon to save the view as SDR Analyzer Exercise 1 in the default /home/demouser directory.

## Part 2: Analyze and map calling data by geography

Now that you have verified that a VOIP calling plan makes sense, let's determine the geographic region where this service would make the most sense. In this exercise, you filter on calls made from home and plot call volume by geography in an interactive map.

- Click on the **Create New | Analysis Report** buttons.
- Select the SDR Geo Cube data source at the bottom of the **Data Sources** list.
- In the Available Fields section, right-click on Location category and choose **Filter** to filter the view where Location Category equals At Home.

Filter on Location category

☒ Select from a list (Includes, Excludes)  
☐ Match a specific string (Contains, Doesn't Contain)

Choose values from list: Find Currently Included

Available Values	Selected Values
At Home	At Home
In Town	
In the Neighborhood	
Regional	
Travel	

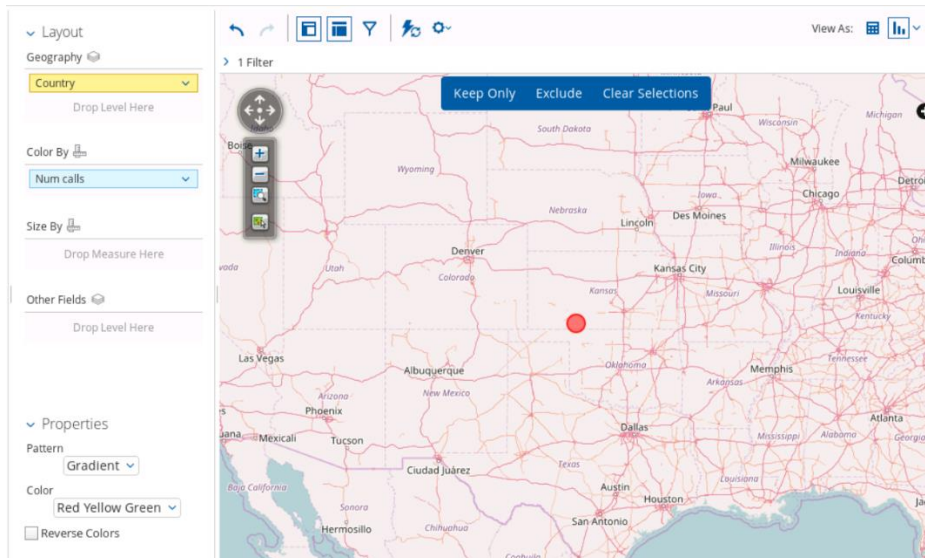
Showing all 5 values 1 value selected

☐ Parameter Name

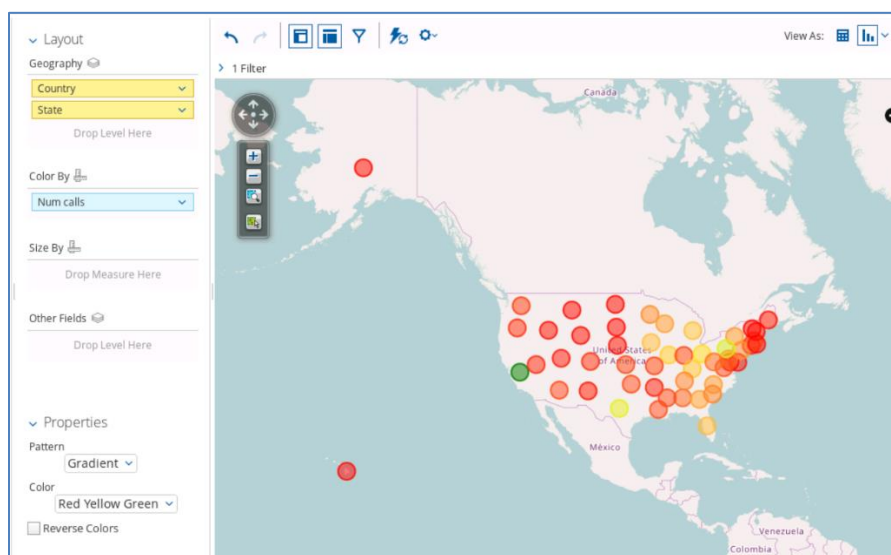
OK Cancel

15. Add Country, and the measure Num calls to the canvas.

16. Click the chart dropdown and select to **Geo Map**.



17. Double on the country circle and you will drill down to state



Note California has the most calls at home as denoted by the darkest green circle.

Now see that California is the state to focus on. Congratulations, you have now completed the SDR use case!



## Use Case 3: Self Service Data Preparation

### What is it?

Analytics users spend a significant amount of time preparing data for analysis or waiting for other people to prepare data for them. Self-service data preparation is a process for exploring, combining, cleaning and transforming raw data into curated datasets for business intelligence and analytics.

### Why do it?

Existing data integration approaches are time-consuming and complex for data analyst to keep up with demand from the business. The growing volumes and variety of data are increasing the demand for easy-to-use data preparation tools. Companies have increased pressure to be responsive to the data needs of the business, and the need to quickly enrich and blend more data sources has become increasingly important. These challenges have made data preparation one of the primary blockers to delivering on the promise of analytics.

### Value of Pentaho

- Data agnostic – access to any data source
- Data exploration and profiling – a visual environment to explore and profile data
- Data transformation, blending and modeling – blending, cleansing, filtering, modeling
- Collaboration – iterative, agile development environment with ability to publish and share models
- Data curation and governance – data encryption, security and data lineage
- Integrated Analytics and Machine learning – use of analytics to improve data preparation

### What You Will Accomplish

A Marketing Company offers drivers a service to wrap their cars with an advertisement to make extra money. An analyst for the Marketing Company needs to start targeting certain markets to attract new drivers and needs to determine which state and cities to focus on first. To figure this out, we need to combine call data records (CDR) that they purchase and blend it IoT Data produced by cell towers.

By using these two data sources you can determine someone's cell phone usage and measure the distance they are from home. The further the distance, the more "exposure" that person provides for the advertisement and hence would be the best target market.

You will complete **1 exercise** to create a transformation that merges data from two files and then categorizes the drivers into three buckets: Low Exposure, Medium Exposure, and Large Exposure. You'll correct the data along the way and create a geographical hierarchy to view the data on an interactive map.

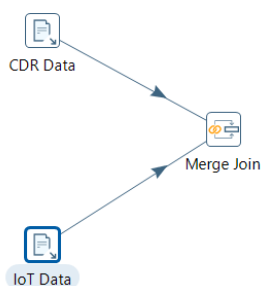
## Self Service Data Preparation Exercise 1: Use Pentaho to prepare data for analytics

1. In Spoon, create a new **Transformation**
2. From the **Input** folder, select and drag the **CSV Input** step.  
Note: you'll need two of those, so drag it to the canvas twice.
3. Double click on the first **CSV Input** step to edit the step properties. Set the properties as follows:
  - a. Step Name: CDR Data
  - b. Filename: Browse to the following file to select it, and then click the **Open** button  
[file:///pentaho/evaluation/03\\_Self\\_service\\_data\\_prep/data/call\\_detail\\_records.csv](file:///pentaho/evaluation/03_Self_service_data_prep/data/call_detail_records.csv)
  - c. Click **Get Fields** and enter "0" for the sample size to get the data types by scanning all rows of the file. Note that this may take a few seconds.
  - d. Click **Preview** to preview the data. In the sample size, type 500. You should see data coming back.
  - e. Click **OK**
4. Now, let's update the properties of the second **CSV Input** step. Double click it to update the properties.
  - a. Rename the step to IoT Data
  - b. Filename: Browse to the following file to select it, and then click the **Open** button  
[file:///pentaho/evaluation/03\\_Self\\_service\\_data\\_prep/data/call\\_detail\\_geolocation\\_output.csv](file:///pentaho/evaluation/03_Self_service_data_prep/data/call_detail_geolocation_output.csv)
  - c. Click **Get Fields** and enter "0" for the sample size to get the data types by scanning all rows of the file. Note that this may take a few seconds.
  - d. Click **Preview** to ensure that you can see the data
  - e. Click **OK**
5. Next, we will need to blend the data from these two sources. From the **Joins** Folder, select and drag **Merge Join** step onto the canvas.



The **Merge Join** step allows you to blend data from disparate data sources based on one common object. The type of join will drive the result set. We only want data that appears in both data sets so we will choose Inner join for our transform.

6. Create a **Hop** between **CDR Data** and **Merge Join** steps and select **Main output of step**
7. Create a **Hop** between **IoT Data** and **Merge Join** steps and select **Main output of step**
8. The result should look like the following:



9. Double-Click the **Merge Join** step to update its properties.
10. Set the properties as follows:
  - a. **Step Name:** Merge Join
  - b. **First Step:** CDR Data
  - c. **Second Step:** IoT Data
  - d. **Join Type:** Inner
11. Click on **Get key fields** for **1<sup>st</sup> Step** and for **2<sup>nd</sup> Step**.
12. We will blend data using `source_number` as the **key field**. Remove all other fields. Your **Merge Join** Properties should resemble the image below:

Merge Join	
Step name	Merge Join
First Step:	CDR Data
Second Step:	IoT Data
Join Type:	INNER
Keys for 1st step:	Keys for 2nd step:
<div>Key field</div> <div>1 source_number</div>	<div>Key field</div> <div>1 source_number</div>
Get key fields	Get key fields
Help	OK Cancel

13. Click **OK** and then click **I Understand** in the subsequent prompt.



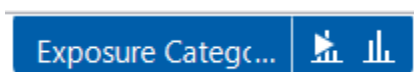
Sometimes, you need to add calculated fields to your data to provide additional analytics. We will create a new field called Exposure, to categorize our data for analysis. The number range step in PDI, allows you to categorize data based on thresholds.

14. From the **Transform** Folder, select and drag the **Number Range** step onto the canvas. Double Click the step to update its properties.
15. Rename the step to **Exposure Category**. Update the properties as follows
  - a. **Input Field**: distance
  - b. **Output field**: Exposure
  - c. **Default Value**: Unknown
16. In the **Ranges** window, specify the following ranges:

▲	Lower Bound	Upper Bound	Value
1	0.0	10.0	Low Exposure
2	10.0	25.0	Medium Exposure
3	25.0		Large Exposure

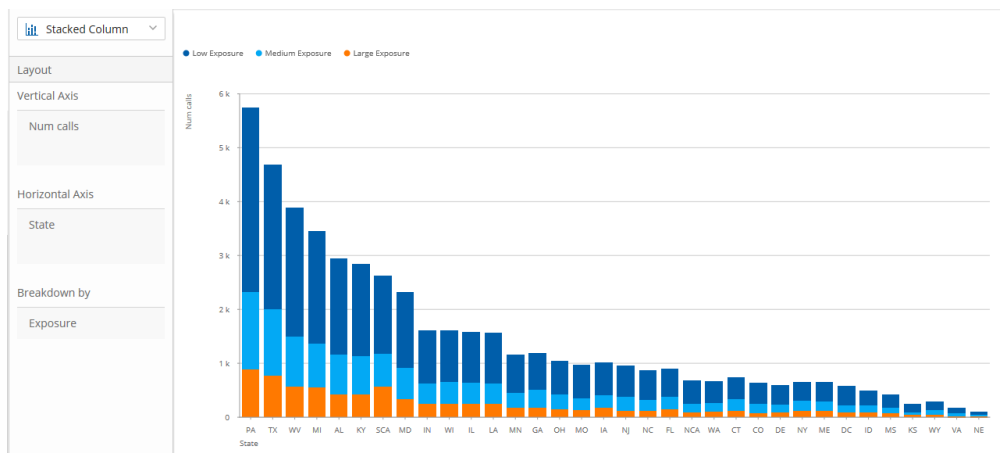
Click **OK**

17. Validate the data up to this point by clicking the Data Explorer icon that runs the transformation (the play icon)



18. From the dropdown selector click on the **Stacked Column** visualization.
19. Add **State** to the **Horizontal axis** and **Num calls** to the **vertical axis**.
20. Add **Exposure** to Breakdown By.

The graph should look like the following:



21. Note that California (CA) has been divided into Southern California (SCA) and Northern California (NCA) in the source data. We need to combine these to understand the exposure for California.

22. Return to the PDI canvas.

- From the **Transform** Folder, select and drag the **Value Mapper** step onto the canvas.
- Insert the step between the **Merge Join** and the **Exposure Category** steps.
- Double Click the **Value Mapper** step to update its properties
- Set the Step Name: `Combine Norther and Southern CA`
- Set **Fieldname to use** to `State`
- Add a **Source Value** of `SCA` with a **Target Value** set to `CA`
- Add another **Source Value** of `NCA` with a **Target Value** set to `CA`
- The result should look like the following:

Value Mapper

Step name :

Combine Northern and Southern California

Fieldname to use :

state

Target field name (empty=overwrite) :

Default upon non-matching :

Field values:

#	Source value	Target value
1	NCA	CA
2	SCA	CA

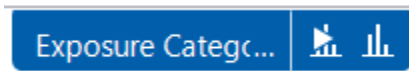
Help

OK

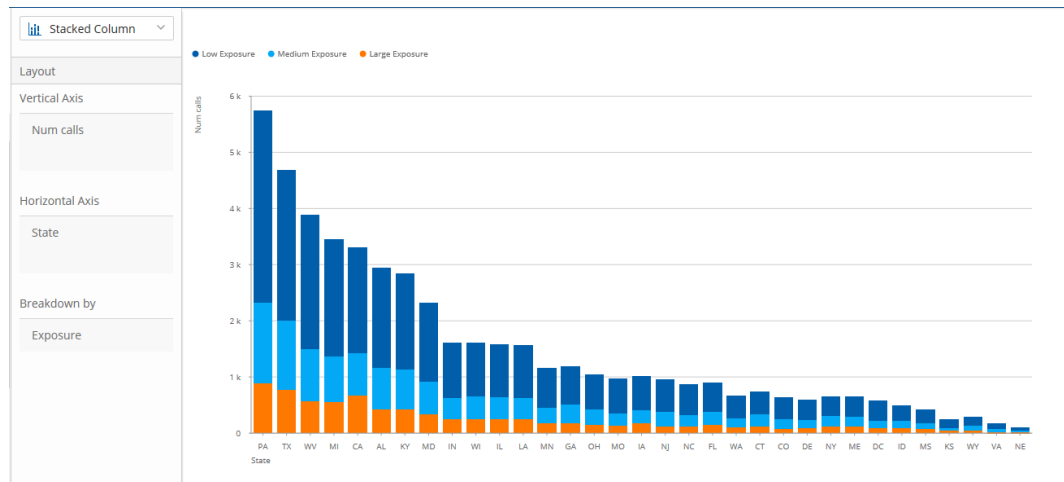
Cancel

Click **OK**

23. Click the play icon again and click **Yes** to save the transformation.



24. You should now see only CA rather than separate values for the state.



25. Now we can filter out Low and Medium exposure and only look at the states with the most Large distances traveled giving the more exposure.

26. From the **Flow** folder, select and drag the **Filter Rows** step onto the canvas.

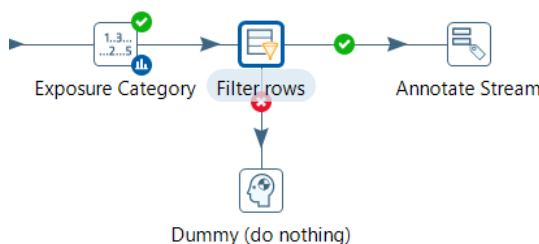
27. From the **Flow** folder, select and drag the **Annotate Stream** step onto the canvas.

28. From the **Flow** folder, select and drag the **Dummy (Do Nothing)** step onto the canvas.

29. Create a **Hop** from **Exposure Category** to **Filter Rows**.

30. Create a **Hop** from **Filter Rows** to **Annotate Stream** and select **Result is True**.

31. Create a **Hop** from **Filter Rows** to **Dummy (Do Nothing)** and select **Result is False**.  
The result should look like the following:



32. Double Click Filter Rows and enter the following:

- Set the condition where the <field> is `Exposure`, function is `Contains`, and the value is `Large`

b. The result should look like the following:

The screenshot shows a 'Filter rows' dialog box. The 'Step name' field contains 'Filter rows'. Below it, 'Send 'true' data to step:' is set to 'Annotate Stream' and 'Send 'false' data to step:' is set to 'Dummy (do nothing)'. The 'The condition:' section shows a logical expression: 'Exposure' followed by 'CONTAINS' and then 'Large (String)'. At the bottom, there are three buttons: 'Help', 'OK', and 'Cancel'.

33. Now we want create a geographical hierarchy to drill down to primary cities of each state.

34. Double click the **Annotate Steam** step and change the **Step Name** to Add Geo Hierarchy

- a. Click **Select Fields** and select `country` from the **available fields** and move it to the **selected fields**. Click **OK**
  - i. Double Click on `country` and select **Create Attribute** from the **Actions** dropdown list.
  - ii. For **Geo Type** select **Country**
  - iii. **Dimension** enter `Geo`
  - iv. **Hierarchy** enter `Geo`
  - v. Click **OK**
  - vi. The Annotate Stream step should now look like the following

Annotate Stream

Step Name:  
Add Geo Hierarchy

☒ Local  
☐ Shared

Description:

Annotations:

Field	Model Action	Summary
country	Create Attribute	country is top level in hierarchy Geo

Add Calculated Measure... Select Fields...

Help Apply OK Cancel

- b. Click **Select Fields** and select `state` from the **available fields** and move it to the **selected fields**. Click **OK**
  - i. Double Click on `state` and select **Create Attribute** from the **Actions** dropdown list.
  - ii. For **Geo Type** select **State**
  - iii. **Parent Attribute** select **Country**
  - iv. **Dimension** select **Geo**
  - v. **Hierarchy** select **Geo**
  - vi. Click **OK**
- c. Click **Select Fields** and select `primary_city` from the **available fields** and move it to the **selected fields**. Click **OK**
  - i. Double Click on `primary_city` and select **Create Attribute** from the **Actions** dropdown list.
  - ii. For **Geo Type** select **City**
  - iii. **Parent Attribute** select **State**
  - iv. **Dimension** select **Geo**
  - v. **Hierarchy** select **Geo**
  - vi. Click **OK**



- d. Click **Select Fields** and select `source_number` from the **available fields** and move it to the **selected fields**. Click **OK**
  - i. Double Click on `source_number` and select **Create Attribute** from the **Actions** dropdown list.
  - ii. For **Geo Type** select **Location**
  - iii. **Latitude** select `home_latitude`
  - iv. **Longitude** select `home_longitude`
  - v. **Parent Attribute** select `primary_city`
  - vi. **Dimension** select **Geo**
  - vii. **Hierarchy** select **Geo**
  - viii. Click **OK**
  - ix. The **Annotate Stream** step should now look like the following:

Annotate Stream

Step Name: Add Geo Hierarchy

☒ Local  
☐ Shared

Description:

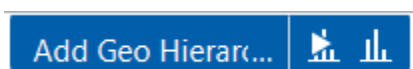
Annotations:

Field	Model Action	Summary
country	Create Attribute	country is top level in hierarchy Geo
state	Create Attribute	state participates in hierarchy Geo with parent country
primary_city	Create Attribute	primary_city participates in hierarchy Geo with parent state
source_number	Create Attribute	source_number participates in hierarchy Geo with parent primary_city

Add Calculated Measure... Select Fields...

Help Apply OK Cancel

35. Click **OK**
36. Click to run the transformation and click Yes to save the transformation.
37. Click the play icon while the **Annotate Stream** step is selected on the canvas.

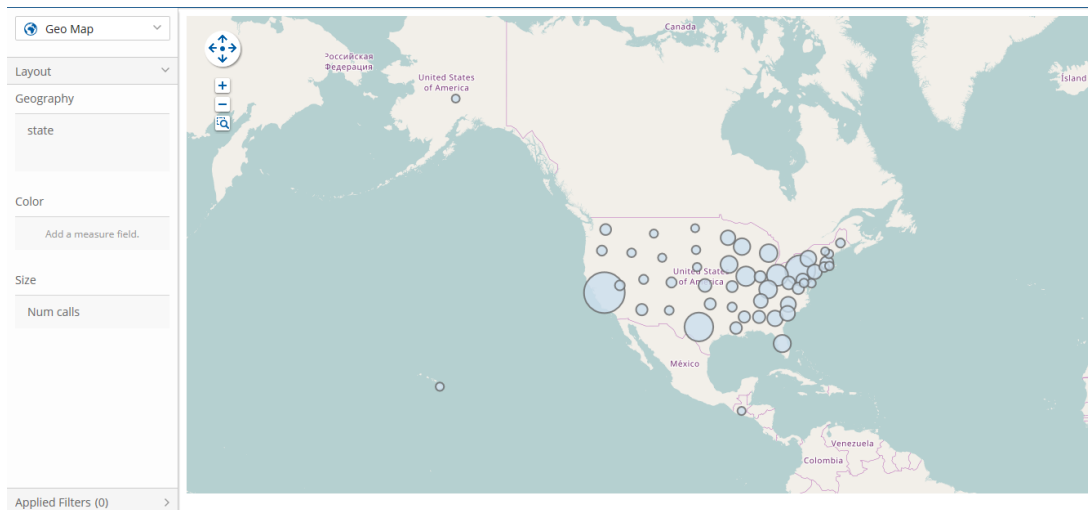


38. From within Data Explorer Select **Geo Map**

39. Scroll down the list of attributes to find the **Geo** hierarchy. Select `state` and drag it under **Geography**.

40. Scroll up to the **Measures** and select `Num calls` and drag it under **Size**

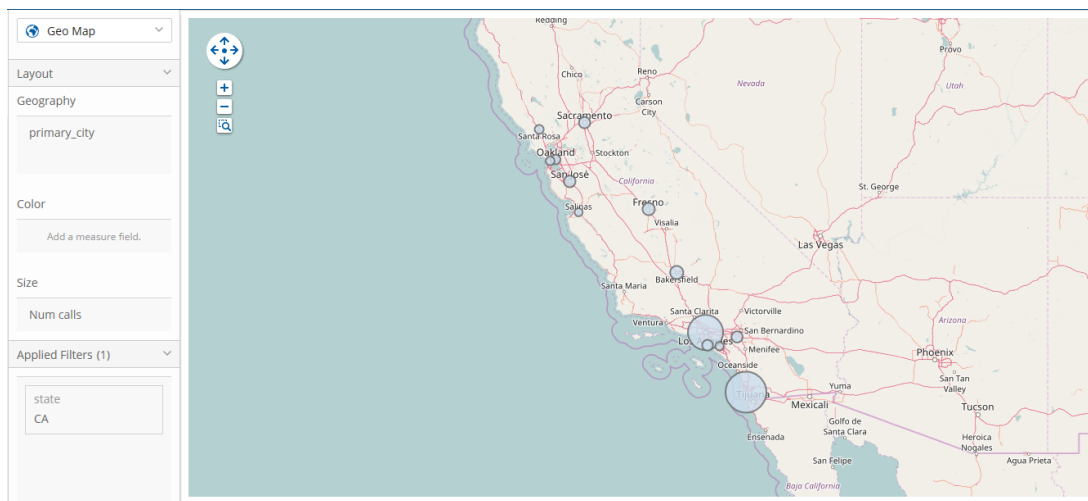
41. Your map should look like the following:



42. Click the **+** to zoom in.

43. You can see that California has the largest circle of the states. Double click on the circle over California to drill down to the cities.

44. Your map now looks like the following:



45. You can see the primary cities in California where you'll get the most exposure for your campaign. Hover over those circles and you can see the number of calls for each.

## Use Case 4: Self Service Analytics

### What is it?

Self-service analytics enables users to visualize business metrics quickly and easily to improve decision making based on accurate, up-to-date and governed data. Users need a reliable system for delivering consistent business metrics at the right time and in the right format. Self-service analytics includes reporting, ad hoc analysis, dashboards and advanced visualization.

### Why do it?

- Make better decisions based on a comprehensive view of the business across the organization
- Empower business users with the information they need to make the best and most timely decisions
- To replace an existing BI solution that no longer serves customer needs or to make the information more reliable and consistent

### Value of Pentaho

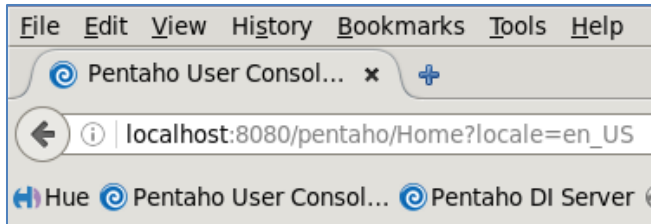
- Better Together: Improve analytics and lower costs with business intelligence and data integration
- Simplified Analytics: Drag and drop interfaces for building analytic applications with the right data, in the right format at the right time for better decision making across your organization
- Time to Value and Low TCO: Pentaho provides a complete analytics offering that is easy to deploy to the broadest set of users (typically deploys in less than 4 weeks)

### What You Will Accomplish

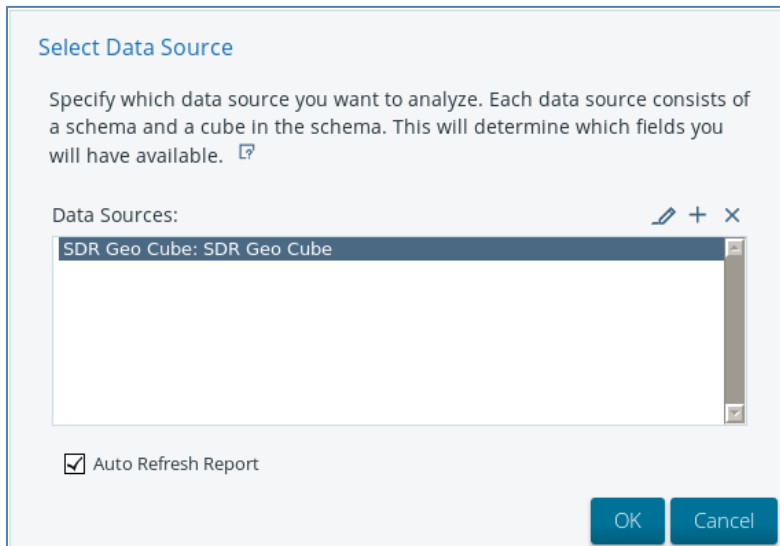
- You will complete **1 exercise** to perform interactive query and analysis on data from previous exercises.

### Self Service Analytics Exercise 1: Use Pentaho to visualize data

1. From a web Browser, connect to Pentaho User Console by clicking the bookmark.



2. From the Pentaho User Console Select **Create New**. Select Analysis Report
3. Select the **SDR Geo Cube: SDR Geo Cube** Data Source



4. Create the following visualization by dragging **Country**, **State** and the measure **Num calls** into the report

The screenshot shows the Pentaho Analysis Report interface. On the left, the 'Available fields (20) for: SDR Geo Cube' list includes categories like Call year, Day of week, Direction, Distance, Geography (Country, State), Key, Location category, and Measures (Call month, Call year, Day of week, Distance, Num calls). The 'Layout' section on the right shows 'Country' and 'State' in the Rows section and 'Num calls' in the Measures section. The main area displays a table with the following data:

Country	State	Num calls
	AK	85
	AL	3665
	AR	916
	AZ	2467
	CA	29267
	CNMI	5
	CO	1695
	CT	1296
	DC	578
	DE	590
	FL	7132
	GA	6105
	GU	143
	HI	190
	IA	6941
	ID	492
	IL	8721
	IN	2524
	KS	2385
	KY	7875
	LA	2892

5. Then select the **Switch to Chart Format** pull down menu then select **Geo Map**

The screenshot shows the 'View As' dropdown menu in the Pentaho interface. The menu lists various visualization types: Column, Stacked Column, 100% Stacked Column, Column-Line Combo, Bar, Stacked Bar, 100% Stacked Bar, Line, Area, Pie, Sunburst, Scatter, Heat Grid, and Geo Map. The 'Geo Map' option is selected, indicated by a checkmark.

- Save the Visualization under the **/home/demouser** folder. Name this analysis GEO\_MAP.

### Save As

Filename:

Location:

Name	Type	Date Modified
SDR Analyzer Exercise 1	File	Jun 15, 2017 12:46:39 PM

- Create a new visualization using the same Data Source by clicking the Icon with a **Plus Sign** at the top, then select **New Analysis Report** and select the data source **SDR Geo Cube: SDR Geo Cube**
- We'll look at calls by area code. Add `State`, `Area code` and the measure `Num calls` to the report as depicted below:

GEO\_MAP x Analyzer Report x

Available fields (22) for: SDR Geo Cube

Find:  View v

Location category

Measures

Num calls

Source number

Time zone

Weekday

Layout

Rows

State

Area code

Drop Level Here

Columns

Drop Level Here

Measures

Num calls

Drop Measure Here

Properties

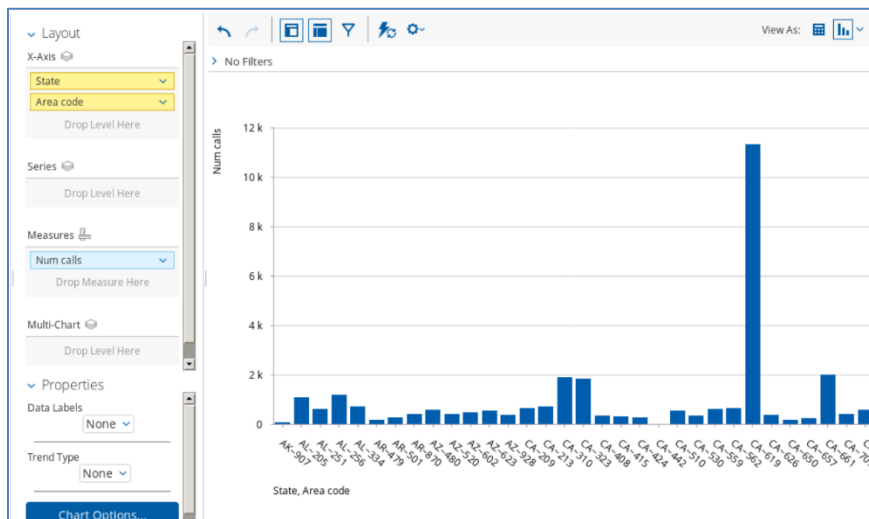
Report Options...

No Filters

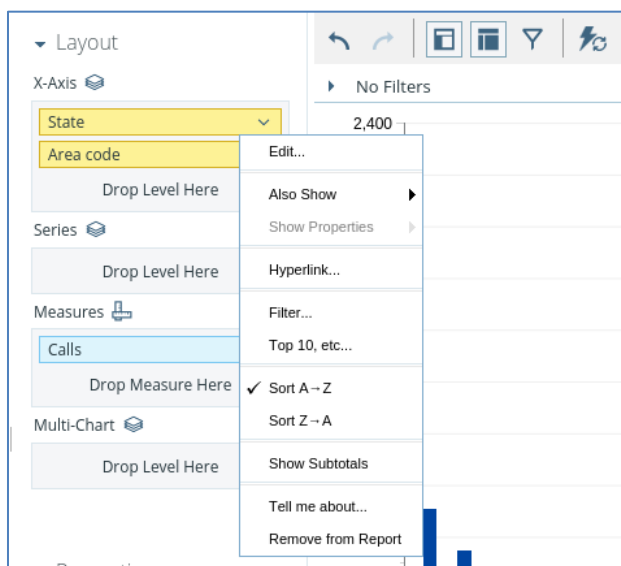
State	Area code	Num calls
AK	907	85
	205	1,111
AL	251	623
	256	1,214
	334	717
AR	479	182
	501	294
	870	440
AZ	480	592
	520	414
	602	501
	623	558
	928	402
	209	676
	213	734
	310	1,902
	323	1,863
	408	375
	415	310
	424	285
	442	1
	510	549
	530	349

9. Change the **View As** to **Column** – Click **OK** if you see a warning about too many records.

10. Your visualization should now look like this:



11. Right Click on **state** and add a filter on that field:

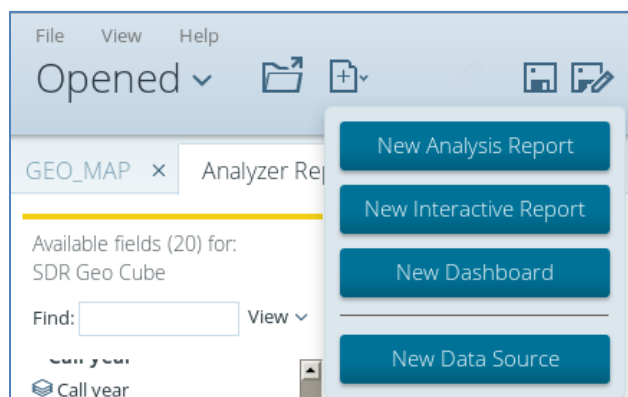


12. On the **filter options** select:

- Select from a list
- Include all states
- Add a parameter called "STATE"

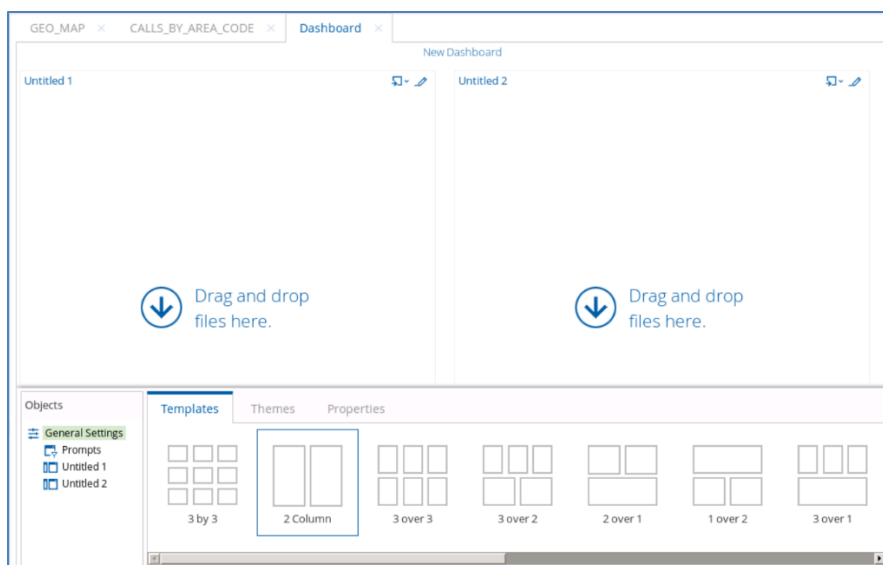
13. Save this Analysis in **/home/demouser** and Name this analysis **CALLS\_BY\_AREA\_CODE**.

14. From the top tab, click on the Icon with a **Plus Sign**. Select **New Dashboard**.

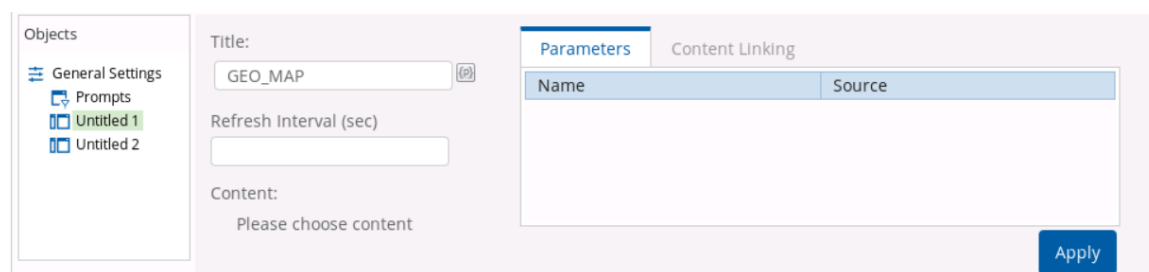


15. Select the 2 column template:





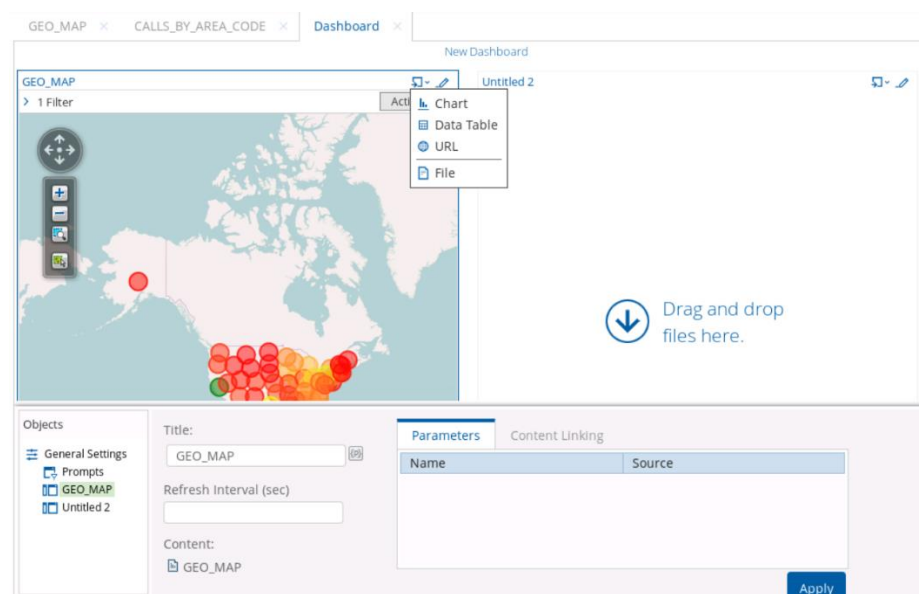
16. Add a title to the dashboard components. You can use the corresponding report names as titles.



17. Click **Apply**

18. From the left column of the dashboard select the icon to **Insert Content** and select **File**

19. Navigate to **/home/demouser** and select **GEO\_MAP** and click **Select**



20. Select the **Content Linking Tab** and enable select **State**:

The screenshot shows the 'Content Linking' tab in the configuration interface. The 'Title' field is set to 'GEO\_MAP'. The 'Refresh Interval (sec)' field is empty. The 'Content' dropdown is set to 'GEO\_MAP'. The 'Parameters' table is as follows:

Enabled	Field
<input type="checkbox"/>	Country
<input checked="" type="checkbox"/>	State

An 'Apply' button is located at the bottom right.

21. Click **Apply**

22. From the right column of the dashboard select the icon to **Insert Content** and select **File**

23. Navigate to **/home/demouser** and select `CALLS_BY_AREA_CODE` and click **Select**

24. Select the `CALLS_BY_AREA_CODE` component and then on the State Parameter select **GEO\_MAP - State**

The screenshot shows the 'Content Linking' tab in the configuration interface. The 'Title' field is set to 'CALLS\_BY\_AREA\_CODE'. The 'Refresh Interval (sec)' field is empty. The 'Content' dropdown is set to 'CALLS\_BY\_AREA\_CODE'. The 'Parameters' table is as follows:

Name	Source
STATE	GEO_MAP - State


An 'Apply' button is located at the bottom right.

25. On the Content Linking tab select State:

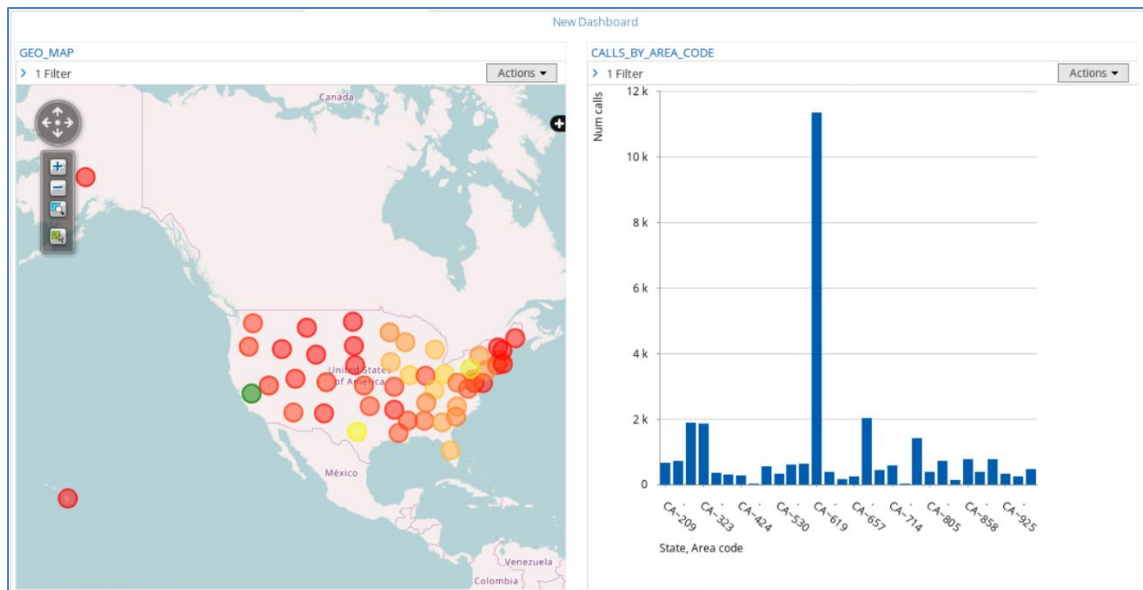
The screenshot shows the 'Content Linking' tab in the configuration interface. The 'Title' field is set to 'CALLS\_BY\_AREA\_CODE'. The 'Refresh Interval (sec)' field is empty. The 'Content' dropdown is set to 'CALLS\_BY\_AREA\_CODE'. The 'Parameters' table is as follows:

Enabled	Field
<input checked="" type="checkbox"/>	State
<input type="checkbox"/>	Area code

An 'Apply' button is located at the bottom right.

26. Save the Dashboard and get out of the edit mode by hitting the pencil icon (  )

27. Now, if you select `California` (double click on the green circle in California)... you should now see the Time Zone component of the Dashboard reflect only California Area Codes:



Congratulations! You have now completed all use cases for the Big Data Sandbox VM.