Enterprise Edition Only

Big Data Integration Workshop

Contents

VM Login and Setup 3
Linux OS Login 3
Hue Login (web browser) 3
Pentaho User Console Login (web browser) 3
Use Case 1: Fill the Data Lake 4
What is it? 4
Why do it? 4
Value of Pentaho 4
What You Will Accomplish 4
Fill the Data Lake Exercise 1: Use Pentaho to ingest 3 sources to Hadoop (CDR, Geography, HCP storage logs) 4
Fill the Data Lake Exercise 2: Explore how metadata injection can be used to automate data onboarding 7
Fill the Data Lake Exercise 3: Use Kafka to ingest IoT data (smartphone geo location tracking) 13
Use Case 2: Create a Data Refinery 18
What is it? 18
Why do it? 18
Value of Pentaho 18
What You Will Accomplish 18
Create a Data Refinery Exercise 1: Transform CDR data and blend with geography data 19
Create a Data Refinery Exercise 2: Process CDR blended data in Hadoop 30
Create a Data Refinery Exercise 3: Extend the PDI job to execute Visual MapReduce 34

Create a Data Refinery Exercise 4: Use Pentaho with Impala to blend CDR and IoT Data 42
Create a Data Refinery Exercise 5: Extend the PDI job to blend data and load to multiple locations 45
Create a Data Refinery Exercise 6: Explore data in PostgreSQL with Pentaho Analyzer 51
Use Case 3: Self Service Data Preparation 56
What is it? 56
Why do it? 56
Value of Pentaho 56
What You Will Accomplish 56
Self Service Data Preparation Exercise 1: Use Pentaho and Impala to prepare data for analytics 57
Use Case 4: Self Service Analytics 62
What is it? 62
Why do it? 62
Value of Pentaho 62
What You Will Accomplish 62
Self Service Analytics Exercise 1: Use Pentaho to visualize Impala data 62
Self Service Analytics Exercise 2: Use Pentaho to report on Hbase data 69

VM Login and Setup

Linux OS Login

workshop_user / bigdata

Hue Login (web browser)

- Username: cloudera
- Password: cloudera

Pentaho User Console Login (web browser)

- workshop_user/bigdata

Enterprise Edition Only

Use Case 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 complete **3 exercises** to ingest the following data sources to Hadoop: CDR, Cell Tower Logs, Geography Master Data, and HCP Storage Logs. You will use *3 different ingestion methods* along the way: 1) file copy 2) metadata injection and 3) Kafka Stream Processing.

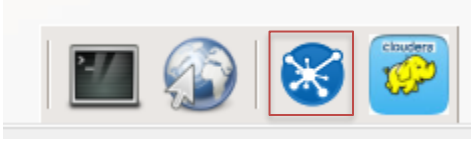
Fill the Data Lake Exercise 1: Use Pentaho to ingest 3 sources to Hadoop (CDR, Geography, HCP storage logs)

Create a PDI Job to load CDR and HCP log data to Hadoop

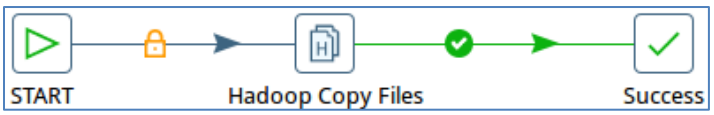
This first exercise steps you through creating a job that uses a "copy files" method of ingestion. You will ingest three sources of data into Hadoop: call detail records (CDR), Area Geography

Master Records, and Hitachi Content Platform (HCP) logs. This first job step loads the three sources of raw data to HDFS so that it can be processed in Hadoop in a later exercise.

1. Open Pentaho Data Integration from the command bar at the bottom of the screen.



2. From the main menu choose **File | New | Job**
3. From the **Design** tab on the left, expand the **General** folder and drag **START** and **Success** job steps onto the canvas.
4. Expand the **Big Data** folder and drag **Hadoop Copy Files** onto the canvas between the **START** and **Success** steps.
5. Create a hop between the **Start** step and the **Hadoop Copy Files** step.
6. 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.

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

```
file:///pentaho/shared_content/WorkshopTraining/01_Fill_the_data
_lake/data/callrecords_all.csv
```

12. Click the **OK** button to return to the **Copy Files** dialog box
13. Select the cell beneath the “Destination Environment” header on the file tab and select “CDH” from the drop down.
14. Select the cell beneath the “Destination File/Folder” step and enter `/BDO/callrecords/input`

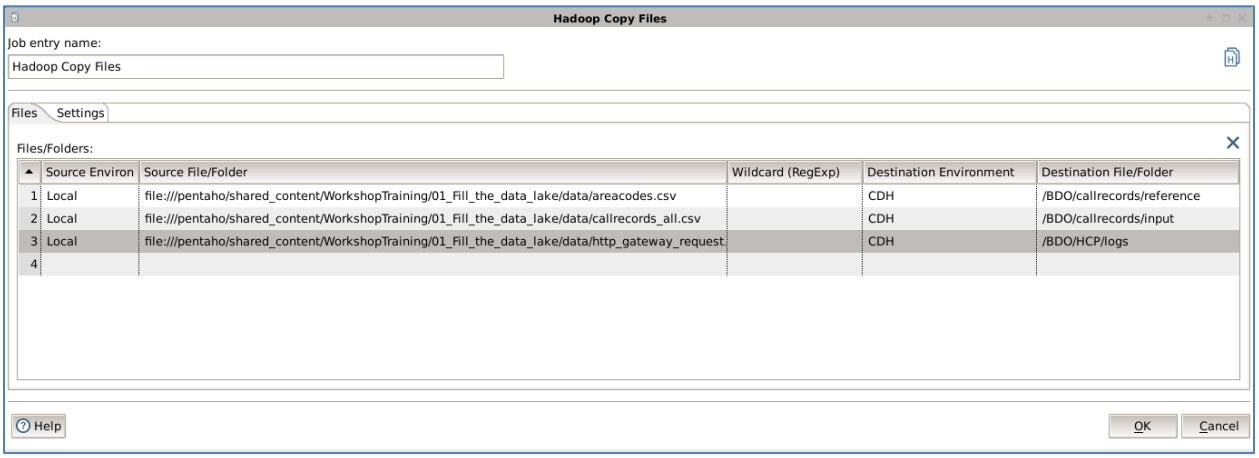
Hit the Enter key after typing in the desired path to ensure that it is persisted.

15. Repeat steps **8-13** to copy the “Source File/Folder” `file:///pentaho/shared_content/WorkshopTraining/01_Fill_the_data_lake/data/areacodes.csv` to the “Destination File/Folder” `/BDO/callrecords/reference`

    and

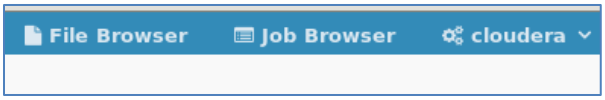
    `file:///pentaho/shared_content/WorkshopTraining/01_Fill_the_data_lake/data/http_gateway_request.log.0` to the “Destination File/Folder” `/BDO/HCP/logs`

16. Your **Hadoop Copy Files** dialog box should match the following image:

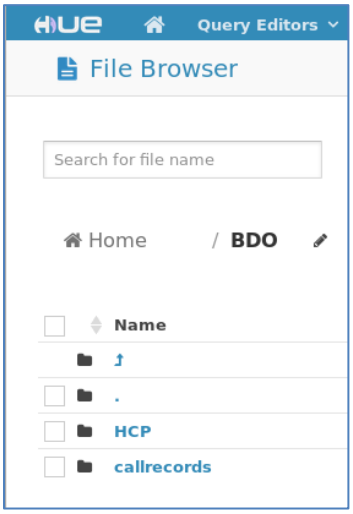


17. Click OK to return to the canvas.
18. From the **File** menu, choose **Save**.
19. In the Name field, specify `Fill Data Lake Exercise 1.kjb` and save to the following location: `/pentaho/shared_content/WorkshopTraining/student_files/01_Fill_the_data_lake`
20. From the **Action** menu, choose **Run**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

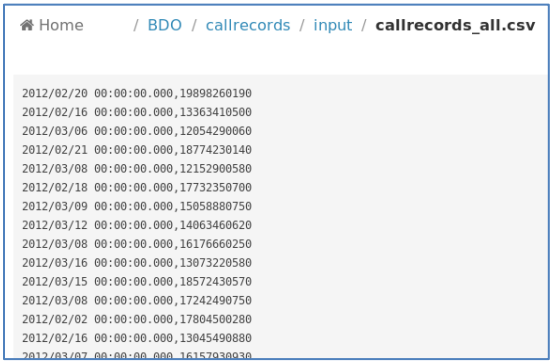
22. From the Firefox bookmarks bar click **Hue**
23. On the sign in page, click **Sign In** (no need to change user id or password)
24. In the far right corner, click **File Browser** to locate the files



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:



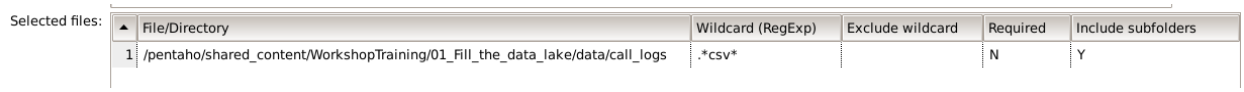
Fill the Data Lake Exercise 2: Explore how metadata injection can be used to automate data onboarding

This second exercise steps you through creating a transformation 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. The metadata injection feature uses automated metadata extraction and injects that metadata into a transformation template. Templates allow for automated data ingestion with minimal configuration even when source files are changing.

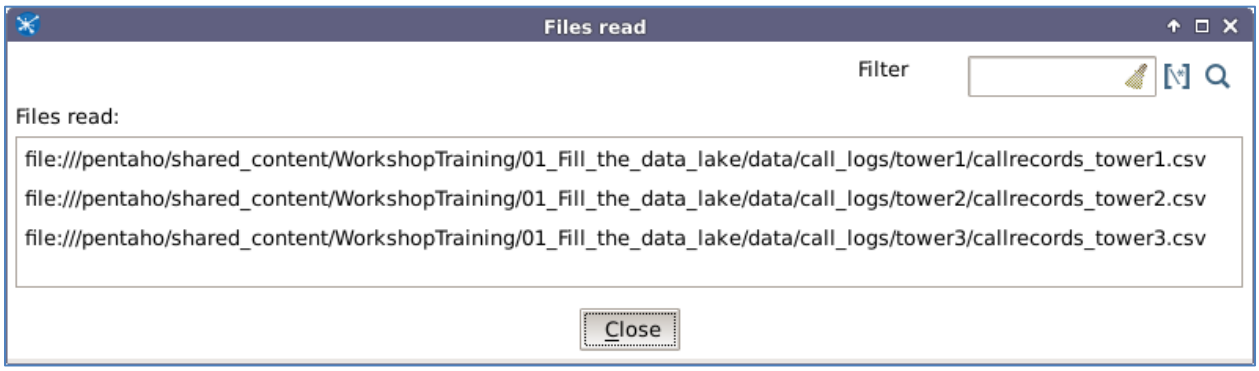
We will be creating two transformations in this exercise named: `t_process_tower_logs` and `t_process_single_log_via_metadata_injection`

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:

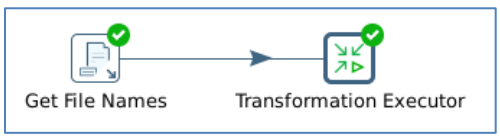
   /pentaho/shared_content/WorkshopTraining/01_Fill_the_data_lake/data/call_logs
5. In the **Selected Files** window, provide **Wildcard (RegExp)** = `.*csv`
6. Click **Add** to make sure it adds the directory under **Selected Files**.
7. Ensure **Include Subfolders** is set to `Y`



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



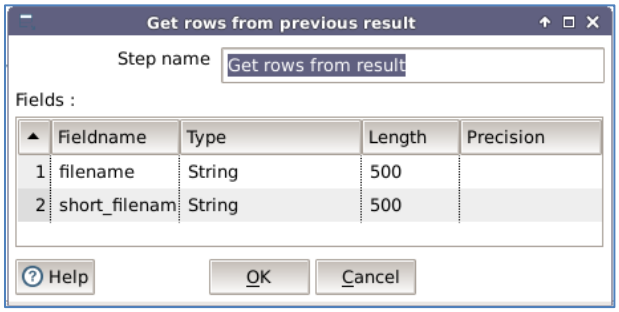
9. Click **Close** and **Ok** to get back to the design canvas.
10. Next, we are going to add a **Transformation Executor** step in order to call another transformation we will be building. From the **Flow** folder in **Design** view, select and drag the **Transformation Executor** step onto the canvas.
11. Build a hop between the **Get File Names** and **Transformation Executor** steps.
12. 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 the File Metadata step in order to discover metadata for each of the incoming files and inject it into a pre-built template

13. Start a new transformation.

14. 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.



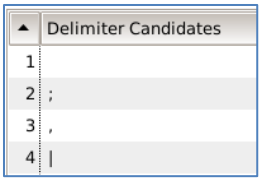
15. From the **Transform** folder, select and drag **File Metadata** step onto the canvas. Double-click the step to modify its **properties**.

    Note: no need to create any hops here yet.

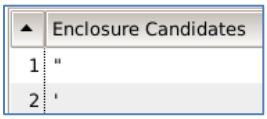
    a. **Filename**: `${filename}`

       Note: we are using a variable here to pick up the file names.

    b. **Delimiter candidates**:

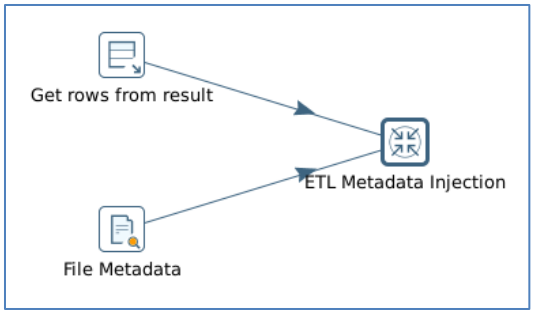


    c. **Enclosure candidates**:



    d. Click **OK**

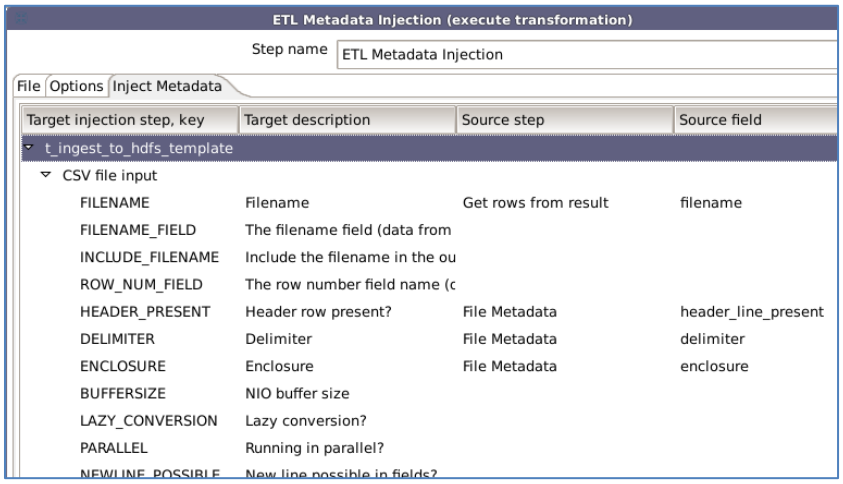
16. From the **Flow** folder, select and drag the **ETL Metadata Injection** step onto the canvas. Double-click the step to modify its properties. Note: you may get an error on this step. Please click OK if you get an error. Once you are back to properties, Click **Validate and Refresh**
17. Create a **Hop** between **Get Rows from result** and **ETL Metadata Injection**.
18. Create a **Hop** between **File Metadata** and **ETL Metadata Injection** your transformation should look like the following



19. Double-click the **ETL Metadata Injection** step to modify its properties.
    Note: you may get an error on this step. Please click OK if you get an error.
20. 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:///pentaho/shared_content/WorkshopTraining/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. Click Ok to return to the design canvas.
21. Double-click the **ETL Metadata Injection** step and then select the **Inject Metadata** tab inside the properties window. Complete the configuration of this step so that it matches the image below:

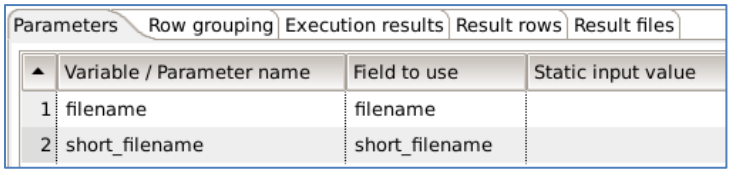


22. Click **OK** to close the **properties** window.
23. Save this transform to `file:///pentaho/shared_content/WorkshopTraining/student_files/01_Fill_the_data_lake` as `t_process_single_log_via_metadata_injection`

24. 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.
25. 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 you just created.

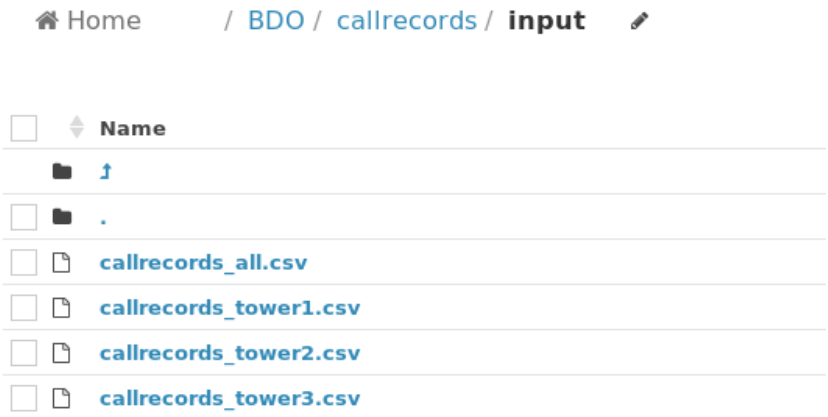
26. In the **t_process_tower_logs** transformation double-click on **Transformation Executor** step to update the properties.
    a. **File Name:**
    `///pentaho/shared_content/WorkshopTraining/student_files/01_Fill_the_data_lake/t_process_single_log_via_metdata_injection.ktr`
    b. **Parameters**:



    c. Make sure that **Inherit all variables from the transformation** is checked

27. Save and Execute this transformation

28. Check the results of your transformation by logging back into Hue and looking for the files you just copied over. Navigate to http://quickstart.cloudera:8888/filebrowser/#/BDO/callrecords/input and you should see the following



29. 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.
30. Open the **Fill Data Lake Exercise 1** job from your student files. (Or simply switch to that job if you still have it open.)
31. From the **General** folder select and drag the **Transformation** step onto the canvas.

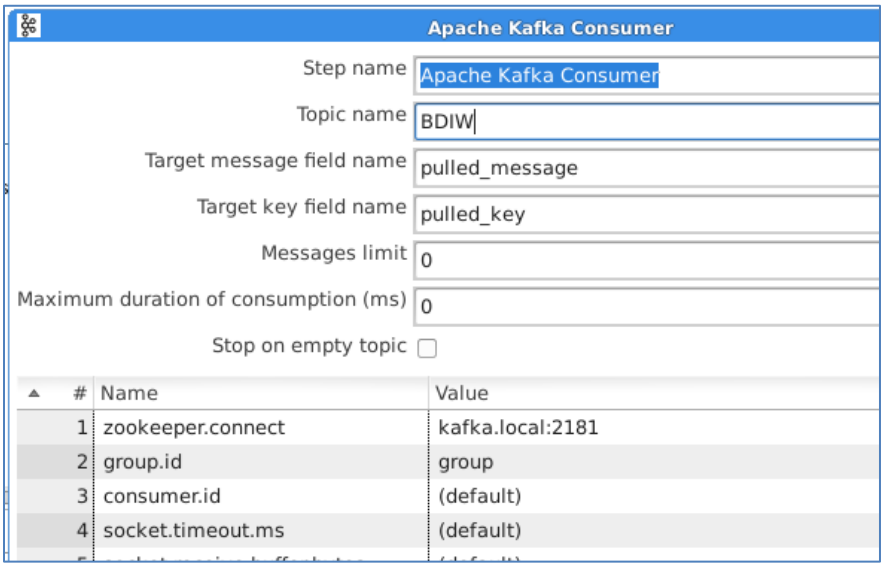
32. Set that step between **Start** and **Hadoop Copy files** steps. There are several ways to insert a step, you can drop the step on the canvas and then drag it on to the hop and wait for the hop to go bold and then drop it, you can split the hop or you can delete the hop and create two new ones. Place the **Transformation** step between **Start** and **Hadoop Copy Files**.
33. Double-Click the **Transformation** Step. Rename it to `Ingest Tower Log Records to Hadoop`.
34. Transformation filename should be `/pentaho/shared_content/WorkshopTraining/student_files t_process_tower_logs.ktr` Note: The path in the dialog may also look like the following `${Internal.Job.Filename.Directory}/t_process_tower_logs.ktr`
35. Your job should look like this now:

36. Save this job as **Fill Data Lake Exercise 1 – MI_Updated.**
37. Execute the job

Fill the Data Lake Exercise 3: Use Kafka to ingest IoT data (smartphone geo location tracking)

This third exercise steps you through creating a transformation using “stream processing with Kafka” method of ingestion. You will be building a transformation to pull smartphone geolocation data in XML format from a Kafka queue, parse and process the XML, and then load the parsed XML data into HDFS for use in later exercises.

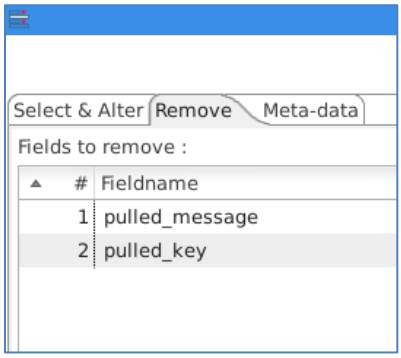
1. Click **New Transformation** in PDI.
2. From the **Design** tab on the left, expand the **Input** folder and drag the **Apache Kafka Consumer** step onto the canvas.
3. Double-click on **Apache Kafka Consumer** step to open its properties
4. Enter `BDIW` in the **Topic name** field.
5. Enter `pulled_message` in the **Target message field name**
6. Enter `pulled_key` in the **Target key field name**.
7. Accept the default values for the rest of the settings. When complete your dialog should look like the following.



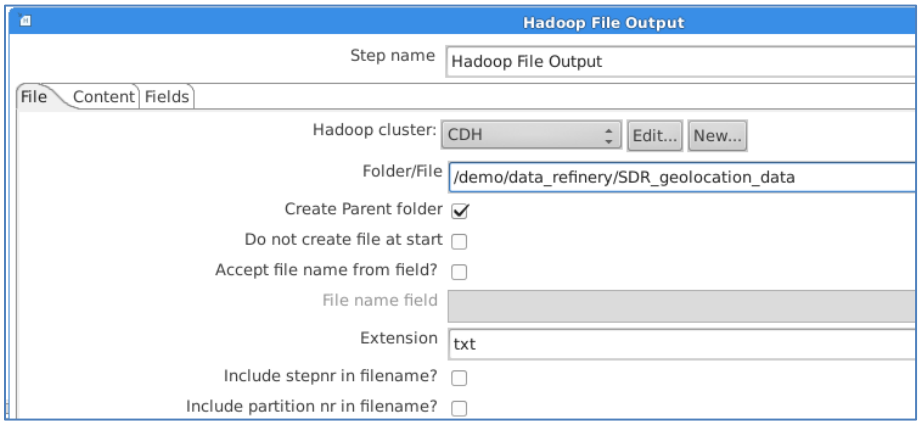
The data in the Kafka queue is coming in as XML, we need to parse out the XML to get at the rows and columns of data before we put it in our data lake. To save time we have fully configured this step for you. We will now copy that into our transformation.

8. Open **File Manager** from the Applications Menu at the top left of your VM. From there navigate to the **Solutions** folder (`///pentaho/shared_content/WorkshopTraining/01_Fill_the_data_lake/Solutions`), right-click and open the “`configured get data from xml task.txt`” file in **gedit**.

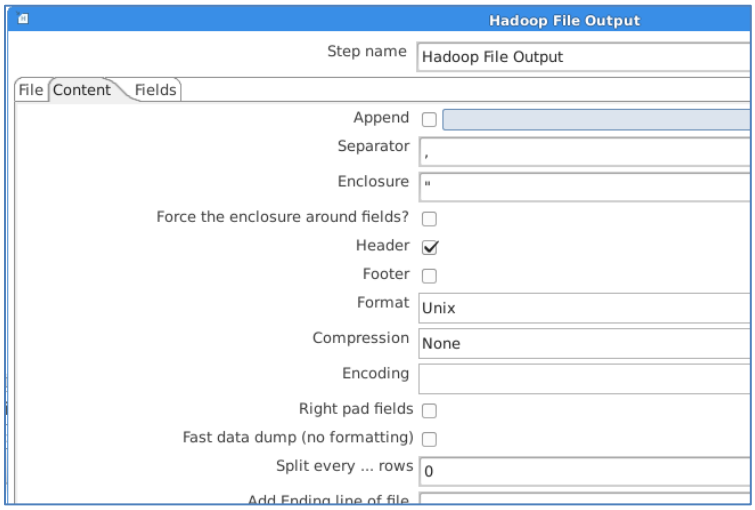
9. Select all the text in the file (`ctrl+A`) and copy it to the clipboard (`ctrl+C`)
10. Back in spoon, right-click in the canvas and select **Paste**. This will paste in a fully configured get data from xml step.
11. Connect the **Apache Kafka Consumer** to the **Get data from XML** step.
12. From the **Design** tab **Transform Folder**, drag the **Select values** step onto your canvas.
13. Connect to the **Get data from XML** step to the **Select values** step.
14. Double-click the **Select values** step and remove the `pulled_message` and the `pulled_key` fields.



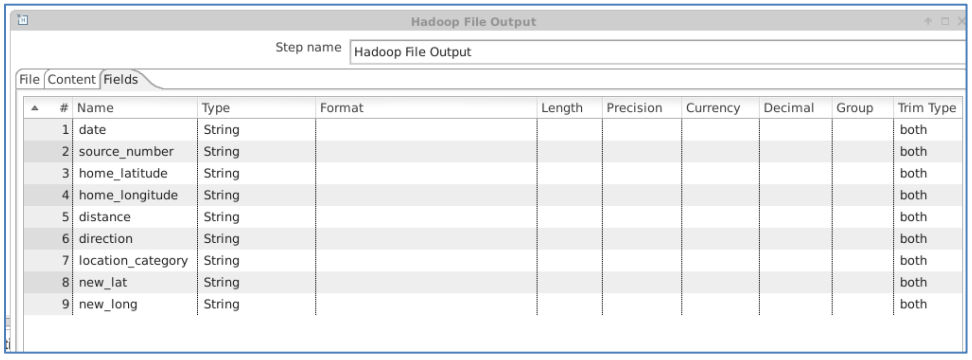
15. From the **Design** tab, drag the **Hadoop File Output** step onto your canvas.
16. Connect the **Select values** step to the **Hadoop File Output** step.
17. From the **Big Data Folder** Double-click the **Hadoop File Output** step and on the **File** tab set the **Folder/File** field to `/demo/data_refinery/SDR_geolocation_data`.
18. Set the **Extension** field to `txt`. Select `CDH` for the **Hadoop Cluster**. The completed tab should look like the following image.



19. On the **Contents** tab, ensure the **Separator** is set to , **Enclosure** is set to ", **Header** is checked and the **Format** is set to `Unix`.



20. On the **Fields** tab, click the **Get Fields** and **Minimum Width** buttons. This will populate the rest of this tab. When complete it should look like the following.



#	Name	Type	Format	Length	Precision	Currency	Decimal	Group	Trim Type
1	date	String							both
2	source_number	String							both
3	home_latitude	String							both
4	home_longitude	String							both
5	distance	String							both
6	direction	String							both
7	location_category	String							both
8	new_lat	String							both
9	new_long	String							both

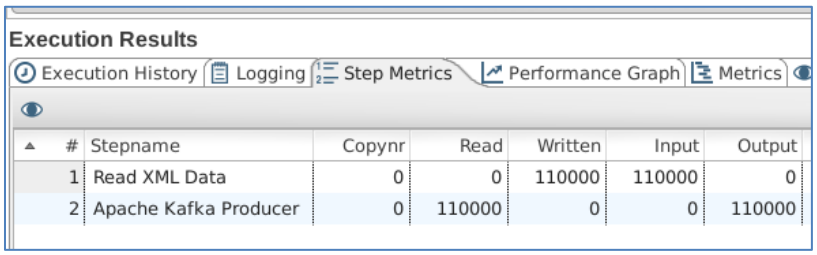
21. **Save** your transformation to the **Student Files** folder (`/pentaho/shared_content/WorkshopTraining/student_files/01_Fill_the_data_lake/`) as `t_kafka_to_hdfs`. Your transformation should look like the following.

We need to start Kafka before we run this transformation. On the desktop of the VM there is an icon named Kafka start, double-click that to ensure kafka is up and running.

22. **Run** your transformation

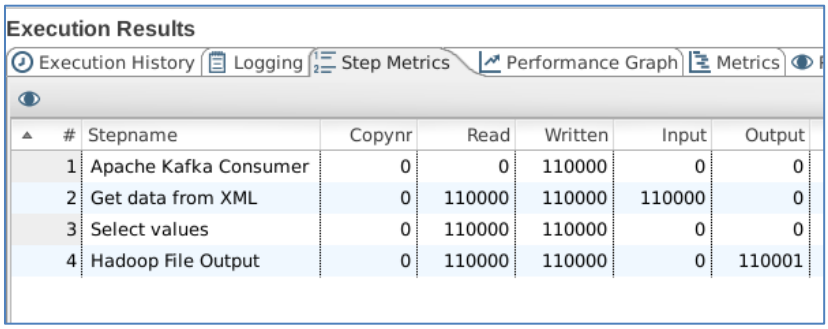
This transformation is now waiting for data to be placed into the BDIW Queue in Kafka. Once data is in the queue, your transformation will pick up and process that data. We will now load data into that queue.

23. From spoon, open the `trigger geolocation.ktr` transformation from the **Solutions** folder (`/pentaho/shared_content/WorkshopTraining/01_Fill_the_data_lake/Solutions/`) and **run** it.
24. When the triggering has completed you will see that 110,000 messages have been placed into the Kafka queue.



#	Stepname	Copynr	Read	Written	Input	Output
1	Read XML Data	0	0	110000	110000	0
2	Apache Kafka Producer	0	110000	0	0	110000

25. Switch back to the transformation you created and watch it process the data. When completed, your transformation results should look like the following.



#	Stepname	Copynr	Read	Written	Input	Output
1	Apache Kafka Consumer	0	0	110000	0	0
2	Get data from XML	0	110000	110000	110000	0
3	Select values	0	110000	110000	0	0
4	Hadoop File Output	0	110000	110000	0	110001

26. **Stop** your transformation
27. In the browser, navigate to the File Browser in HUE (http://hadoop.local:8888/filebrowser/view=/user/cloudera)
28. Navigate up to the `/demo/data_refinery` folder. In there you will find the `SDR_geolocation_data.txt` file that you have just created.

You have reached the end of the "**Use Kafka to ingest IoT data**" exercise, congratulations! You now have all required data in Hadoop and are ready to start refining that data in the next use case section called "**Creating a Data Refinery**".

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 **6 exercises** to create a data refinery that processes and blends a combination of data sources from previous exercises.

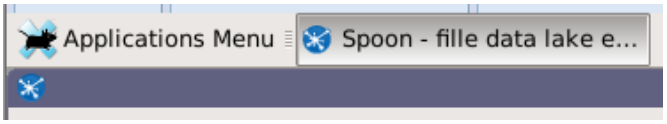
Create a Data Refinery Exercise 1: Transform CDR data and blend with geography data

This exercise steps you through creating an advanced transformation to process and blend CDR data in a *non-Hadoop* environment. You will process and blend CSV data and load it into PostGreSQL Database. The transform will include working with the **Stream Lookup** step to blend master geographic data. You will also add additional steps to enrich and filter the data as needed to complete the transformation. Then, in later exercises, you will modify this same transformation to run in Hadoop.

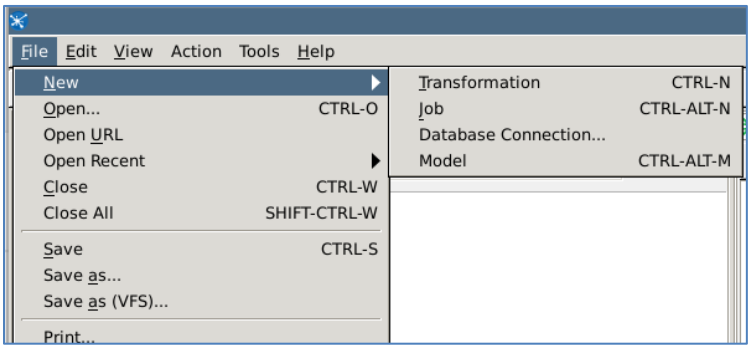
1. Launch the PDI client-based authoring tool (Spoon) from the launch menu icon at the bottom of your screen. Click this icon just *once* to launch Spoon.

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 | Transformation**



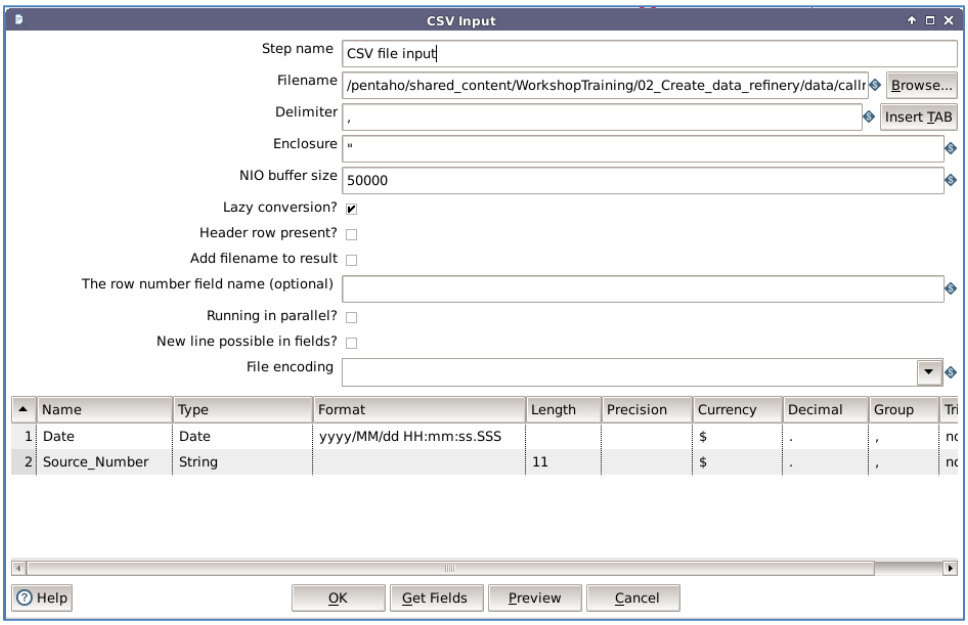
4. From the **Design** tab on the left, expand the **Input** folder; then, select and drag **CSV File Input** onto the **canvas**
5. Double-click on **CSV file Input** step to update its properties.
   a. **Filename**: `/pentaho/shared_content/WorkshopTraining/02_Create_data_refinery/data/callrecords_10years.csv`
   b. Our source file does not include a header row, so make sure to leave the **Header Row Present** property unchecked
   c. At the bottom of the properties window, 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.

d. In the first row change the **Name** to `Date`, **Type** to `Date` and the **Format** to `yyyy/MM/dd HH:mm:ss:SSS`

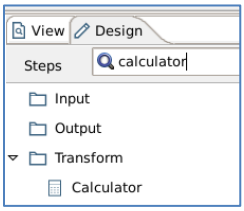
e. In the second row, change the **Name** to `Source_Number`, **Type** to `String` and **Length** to `11`

f. The properties window should look like the screenshot below:



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

7. 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

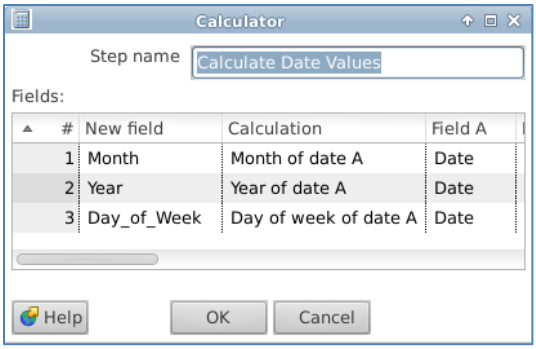


8. Double-click on the **Calculator** step to open its properties.

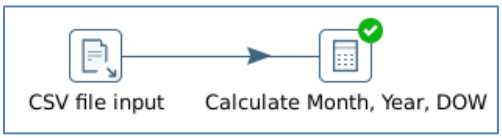
9. Create a hop from the **CSV file Input** step to the **Calculate Month, Year, DOW** step.

10. Change the **Step name** to `Calculate Month, Year, DOW`

11. In the **Fields** section add `Month`, `Year` and `Day_of_Week` as **New Fields**. In **Field A**, select `Date` field from the drop down. In the **Calculation** field, select the appropriate pre-defined calculation. The properties for these fields should match the following screenshot:

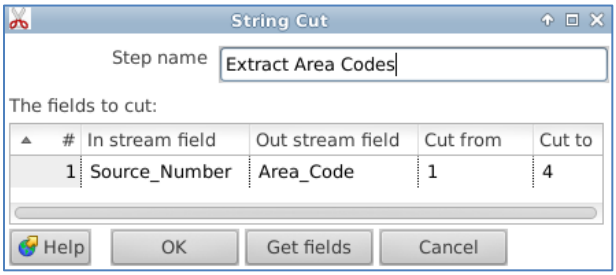


12. Click **OK** to return to the canvas.
13. The first part of your transformation should now match the following image:

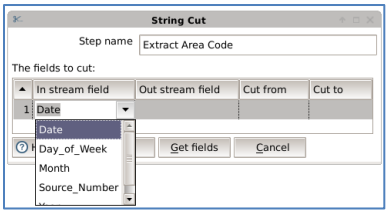


Next, we'll need the location information. To derive location information from the data, we must know the Area Code within the phone number.

14. Select and drag the **Strings Cut** step onto the canvas.
15. Create a hop between the **Calculate Month, Year, DOW** step and the **Strings Cut** step.
16. Double-click on the **Strings Cut** step to open its properties.
17. 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:



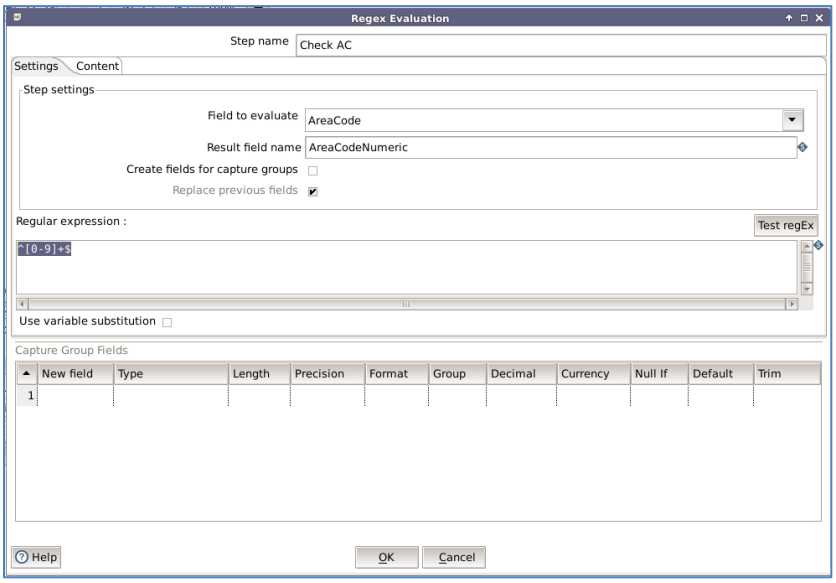
Note: To select the **In Stream field**, you can click on the drop down and select it from the list of fields available. If no fields are available, click **Get Fields** to get the litst.



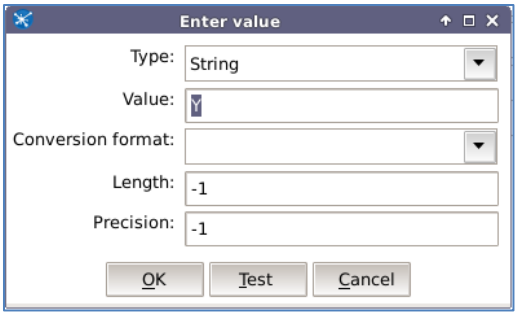
18. Click **OK** to return to the canvas.

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.

19. From the **Scripting** Folder, select and drag **Regex Evaluation** step onto the canvas
20. Create a **hop** from **Extract Area Code** step to the **Regex Evaluation** step.
21. 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]+$
22. The properties of the
23. **Check AC** step should look like this:

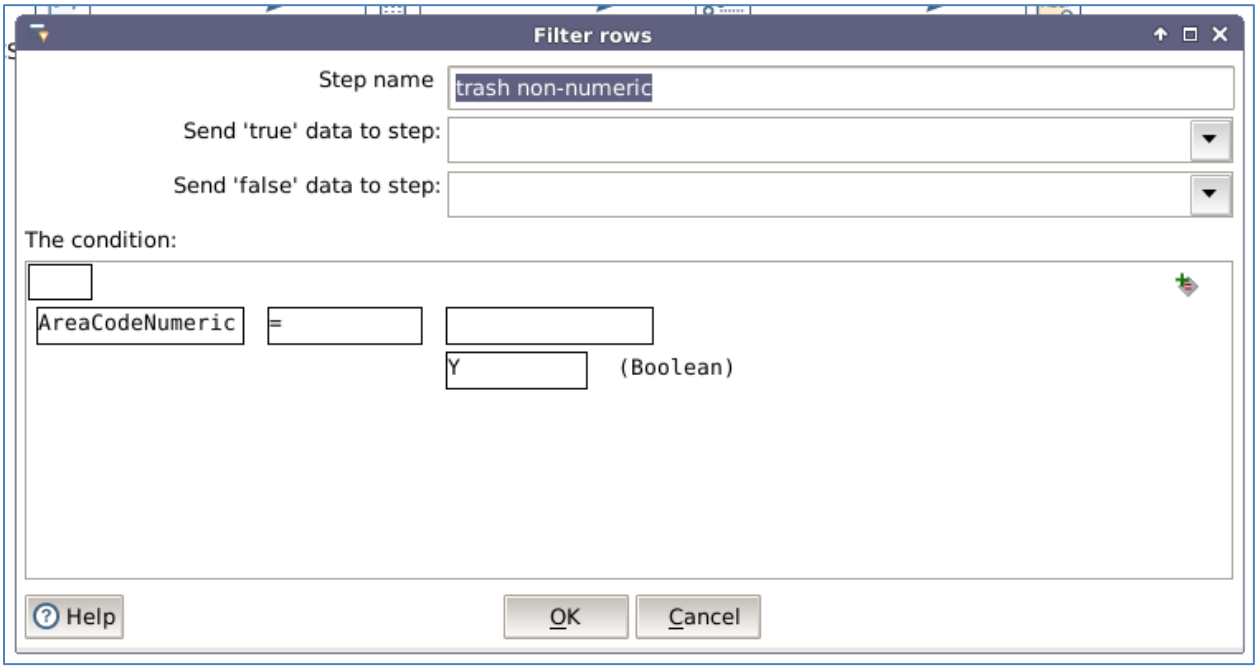


24. From the **Flow** folder, select and drag the **Filter Rows** step onto the canvas
25. Create a hop between **Check AC** step and **Filter rows** step.
26. Double-click the **Filter rows** step to update its properties.
27. Rename this step to `Trash non-numeric.`
    a. In **The Condition** window, 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:



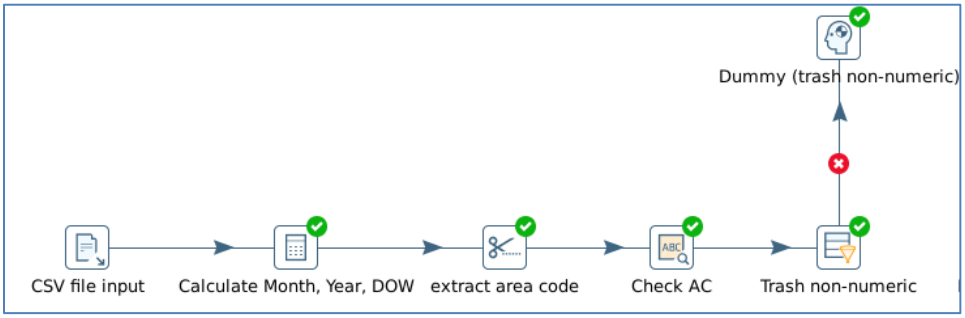
c. Click **OK**

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

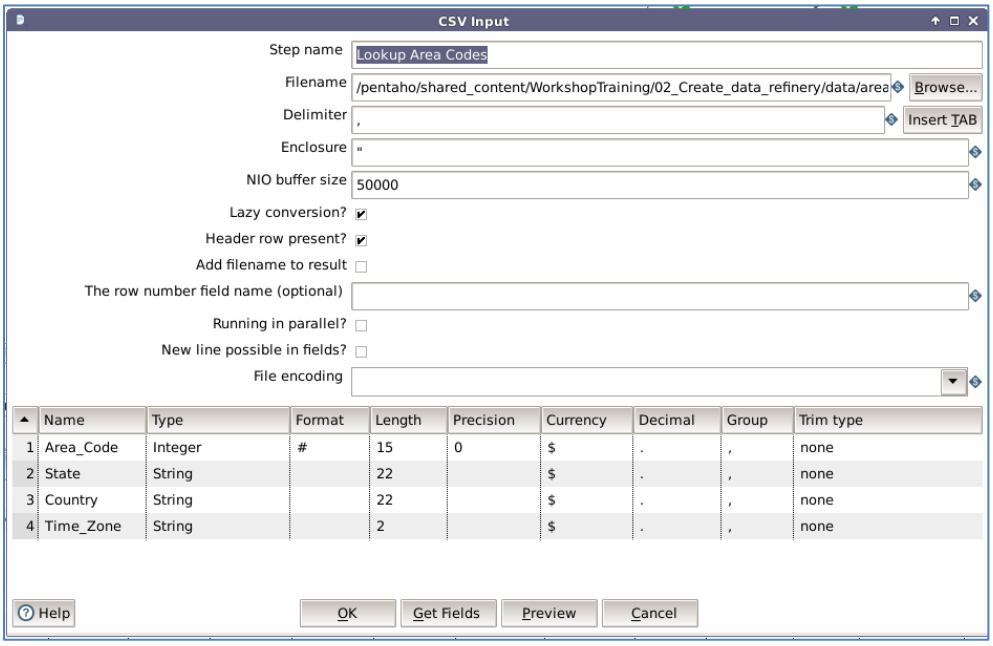


28. Click **OK**
29. Expand the **Flow** folder; then, select and drag **Dummy(do nothing)** onto the canvas, directly above the **Trash non-numeric** step. Double-click to open the properties of this step. Rename the step to **Dummy(Trash non-numeric)**
30. Create a hop between the **Trash non-numeric**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

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

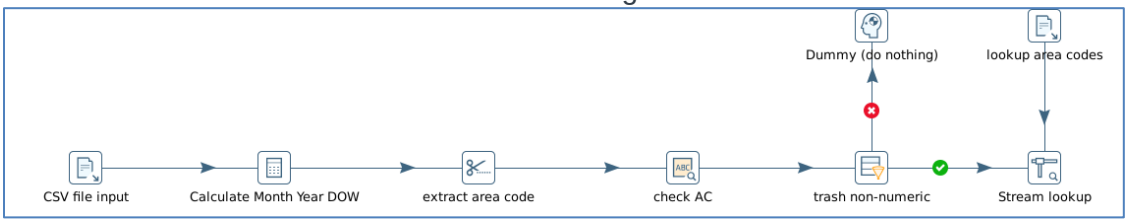


32. Expand the **Lookup** folder; then, select and drag **Stream Lookup** onto the canvas.
33. Expand the **Input** folder and select and drag the **CSV File Input** step onto the canvas directly above the Stream Lookup step.
34. Double-click on the **CSV File Input** step to open its properties. Update the properties as follows:
    a. **Step Name**: `Lookup Area Codes`
    b. **Filename**: `/pentaho/shared_content/WorkshopTraining/02_Create_data_refinery/data/areacodes.csv`
35. This file does contain the header row, so make sure to check the box next to **Header Rows Present** property.
36. Click on **Get Fields** to get the field names and data types. In the sample size, type 0.
37. The Properties for this lookup file should resemble the image below:

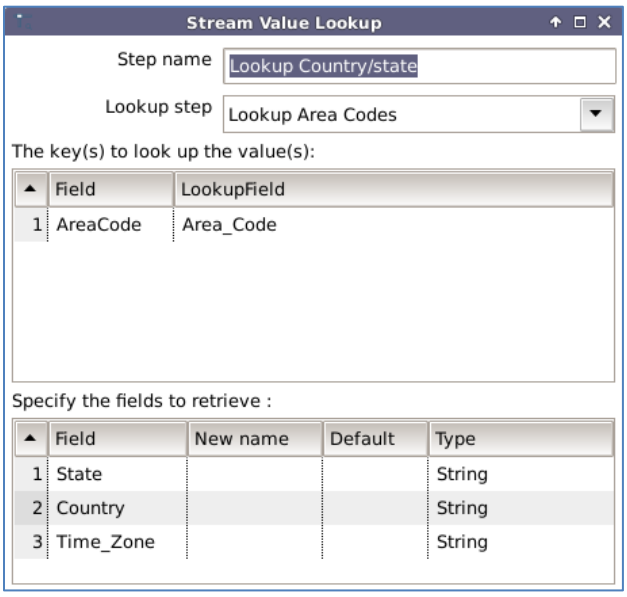


38. Click **OK** to return to the canvas.
39. Create a hop from the **Trash non-numeric** step to the **Stream Lookup** step. When prompted choose, **Main Output of Step**, to complete the hop connection.

40. 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.
41. Your transformation should look like the following



42. Double-click on the **Stream Lookup** step to open its properties.
43. In the **Lookup step** select `Lookup Areas Codes`.
44. In the **Key(s) to Lookup Value(s)** section select `AreaCode` in the **Field** column and select `Area_Code` as the **LookupField** column.
45. Click the **Get Lookup Fields** button (bottom right) to populate the **Fields to retrieve** section at the bottom.
46. 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:



47. 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.

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

51. 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:

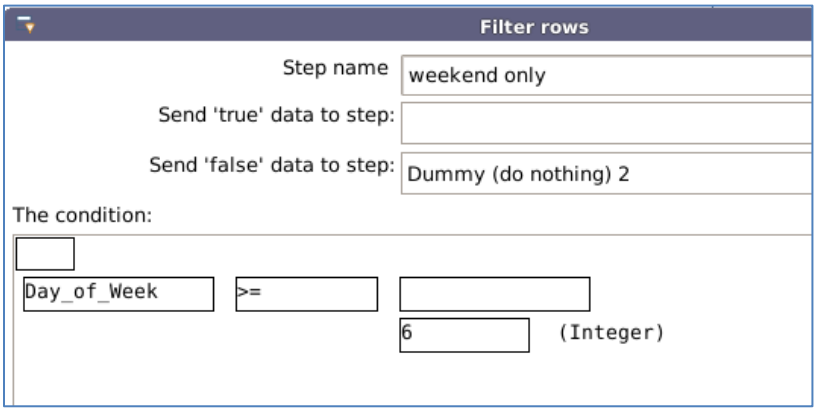
	Source value	Target value
1	1	Sunday
2	2	Monday
3	3	Tuesday
4	4	Wednesday
5	5	Thursday
6	6	Friday
7	7	Saturday

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

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.

53. Expand the **Flow** folder; then, select and drag **Filter Rows** onto the canvas.
54. Create a hop between the **Day of Week** step and the **Filter Rows** step.
55. Double-click on the **Filter Rows** step to open its properties.
56. Rename this step to **U.S.-only calls**.
57. Click **OK** to return to the canvas.
58. From the **Flow** folder, select and drag **Dummy (do nothing)** onto the canvas above the **U.S.-only calls** step.
59. Create a hop from the **U.S.-only calls** step to the **Dummy (do nothing)** step. Select `Result is False`.
60. Double click the **U.S.-only calls** step to open its properties.
61. Under **The condition** section select the **<field>** on the left. From the pop-up window select `Country` and then click **OK**.
62. Under the **value** select string and enter `UNITED STATES` and click **OK**.
63. Add another **Filter Rows** Step to the canvas. Name it **Weekend Only**. Create a Hop between **US Calls only** and **Weekend Only**, set that to “result is true”. Create a Hop between **Weekend Only** and **Dummy (do nothing)** step used above, set it to “result is false”. Double click the **Weekend Only** step to open its properties.
64. Click in the middle box and select “=” as the function and then click **OK**.
65.

66. Select the **<field>** on the left. From the pop-up window select `Day_of_Week` and then click **OK**.
67. Click in the middle box and select `>=` as the function.
68. Click the bottom right **<value>** box and type `6` as the **Value.**



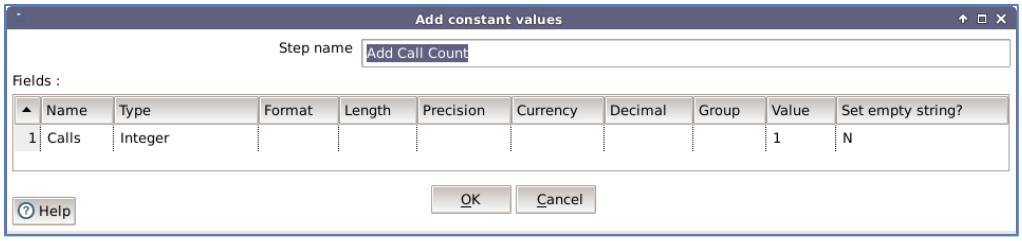
69. Click **OK** to return to the canvas.

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

70. Expand the **Utility** folder and drag **Null if…** step onto the canvas.
71. Create a hop between the **U.S.-only calls** step and the **Null if…** step. Select `Result is true` to complete the hop.
72. Double click the **Null if…** step to open its properties.
73. Rename this step to `Replace Nulls`
74. Under the **Fields** section select `Area_Code` and type `000` in the **Value to turn to NULL** field, then click **OK**.

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.

75. Expand the **Transform** folder; then, select and drag the **Add Constants** step onto the canvas.
76. Create a hop between the **Null If** step and the **Add Constants** step.
77. Double click the **Add Constants** step to open its properties.
78. In the **Step Name** field, type `Add Call Count`.
79. Add a **Field** named `Calls` as a **Type** `Integer` with a **Value** of `1` as illustrated here:



80. From the **output** folder, select and drag **Table Output** step onto the canvas.

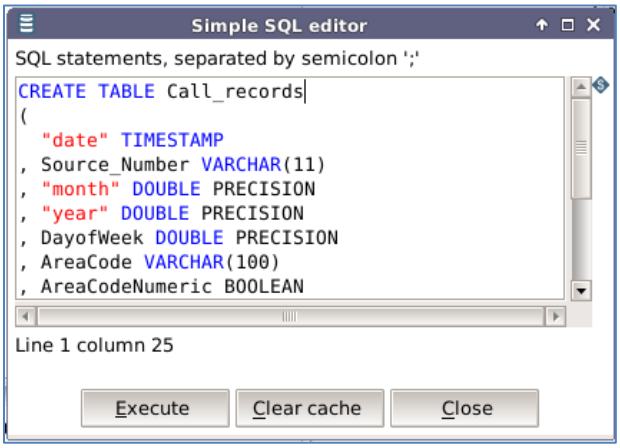
    We will be loading this data into our `postgres` database.

81. Double-click the **Table Output** step to update its properties

    a. From the **connection** drop down, select `workshop_postgres` connection

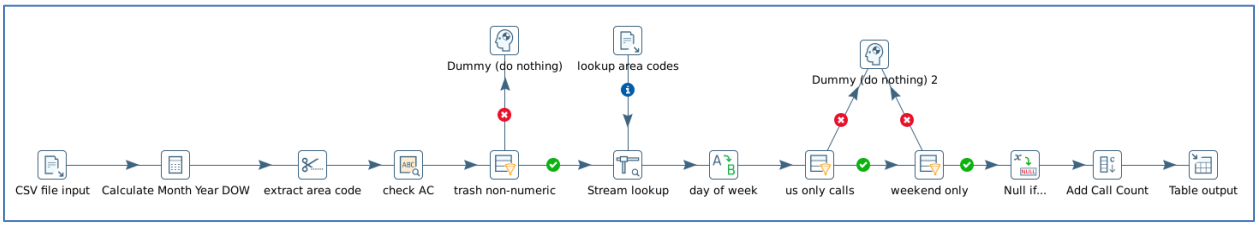
    b. Target table: `Call_records`

    c. At the bottom of the window, Click **SQL**. This will bring up the create table command. Go ahead and **Execute** this command.



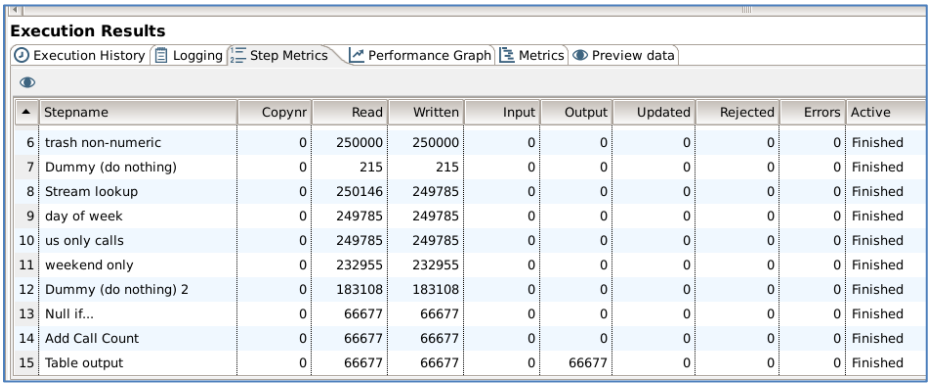
82. Create a hop between the **Add Call Count** step and the **Table Output** step.

83. Your completed transformation should match the following image:



84. From the **File** menu, choose **Save As**. Save this transformation as `t_call_Vol_analysis_RDBMS` in the following directory: `/pentaho/shared_content/WokshopTraining/student_files/02_create_data_refinery`

85. Run this transformation. Your results should look like the following.



	Stepname	Copynr	Read	Written	Input	Output	Updated	Rejected	Errors	Active
6	trash non-numeric	0	250000	250000	0	0	0	0	0	Finished
7	Dummy (do nothing)	0	215	215	0	0	0	0	0	Finished
8	Stream lookup	0	250146	249785	0	0	0	0	0	Finished
9	day of week	0	249785	249785	0	0	0	0	0	Finished
10	us only calls	0	249785	249785	0	0	0	0	0	Finished
11	weekend only	0	232955	232955	0	0	0	0	0	Finished
12	Dummy (do nothing) 2	0	183108	183108	0	0	0	0	0	Finished
13	Null if...	0	66677	66677	0	0	0	0	0	Finished
14	Add Call Count	0	66677	66677	0	0	0	0	0	Finished
15	Table output	0	66677	66677	0	66677	0	0	0	Finished

Create a Data Refinery Exercise 2: Process CDR blended data in Hadoop

In this exercise, we will update the transformation we built in the previous *Exercise 1* of this module to process this data inside Hadoop with a visual map reduce transformation and job. We will replace all source CSV files with files in the Hadoop File System (HDFS).

1. In Spoon, open the `t_call_Vol_analysis_RDBMS` transformation from `/pentaho/shared_content/WokshopTraining/02_create_data_refinery/solutions` directory.

   In this transformation, we replace our CSV file sources with Map Reduce Input.
2. Save this transformation as: `t_Call_vol_analysis_mapper` in the following directory: `/pentaho/shared_content/WokshopTraining/student_files/02_create_data_refinery/`
3. Right-click on the **CSV File Input** step and delete it.
4. From the **Big Data** folder in the Design tab, select and drag the **MapReduce Input** onto the canvas
5. Double-click on **MapReduce Input** step to open its properties.

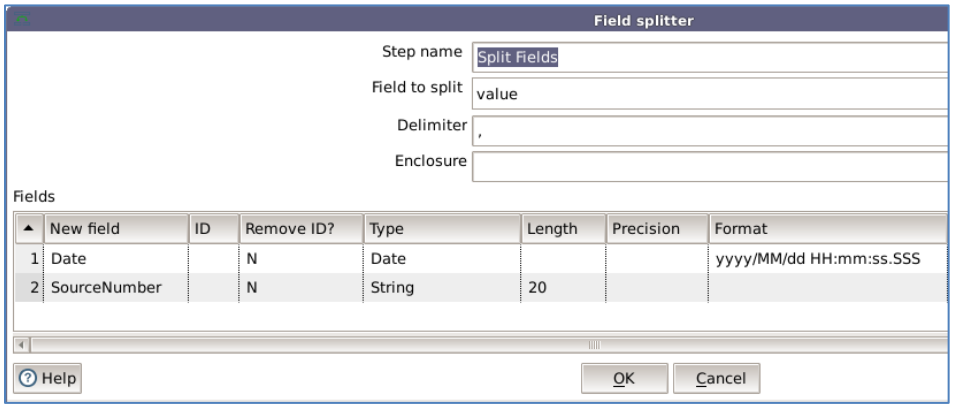
The MapReduce Input step is introducing two new fields into the stream, **key** and **value**. At this point in the transformation, the **key** field is null, and the **value** field is the full CDR record. In a subsequent step, we will split out the **value** field to Date and Source_Number. The **key** field will be assigned later in the mapper transformation as part of the MapReduce Output configuration, so the Hadoop reducer can get the data in the order we need.

6. For both the **Key field** and the **Value field** select `String` for **Type**.
7. Click **OK** to return to the canvas.

Our input **value** of the Key/Value pair is a comma-delimited record of Date and Source Phone Number. We need to split this data into two defined fields of Date and Source Number.

8. Expand the **Transform** folder and drag **Split Fields** onto the canvas.
9. Create a hop from the **MapReduce Input** step to the **Split Fields** step.
10. To draw a hop between two steps, shift-click the **MapReduce input** step and while holding down your mouse key, drag a **hop** over to the **Split Fields** step. When prompted, select **Main output of step**.
11. Double-click on the **Split Fields** step select `value` for **Field to split** and leave "**,**" as the **Delimiter**.

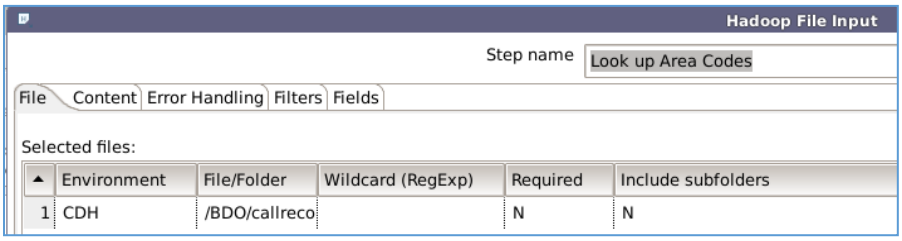
12. In the **Fields** section add `Date` and `Source_Number`.
13. Complete the **Type**, **Length** and **Format** columns for these fields to match the following screenshot:



14. Create a hop between the Split Fields step and the Calculate Month, Year, DOW step.

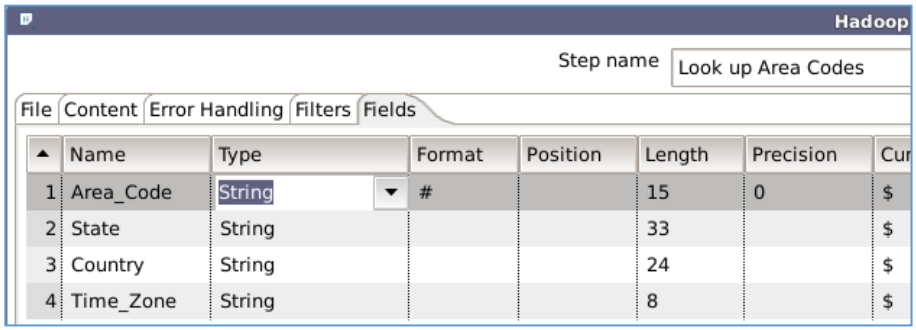
Now, we will also replace the CSV file used as a reference for our State/Country lookup with a file stored in HDFS.

15. Remove the **Lookup Area Codes** Step from the canvas
16. Expand the **Big Data** folder and select and drag the **Hadoop File Input** step onto the canvas directly above the **Stream Lookup** step.
17. Double-click on the **Hadoop File Input** step to open its properties. Rename this step to `Look Up Area Codes`
18. Select the cell beneath the **Environment** header on the file tab and select `CDH` from the drop down.
19. Select the cell beneath the **File/Folder** step and select the ... button.
20. Browse to the following HDFS directory, `/BDO/callrecords/reference` and select the file `areacodes.csv` and then click the **OK** button.
21. Your **Hadoop File Input** dialog box should match the following image:

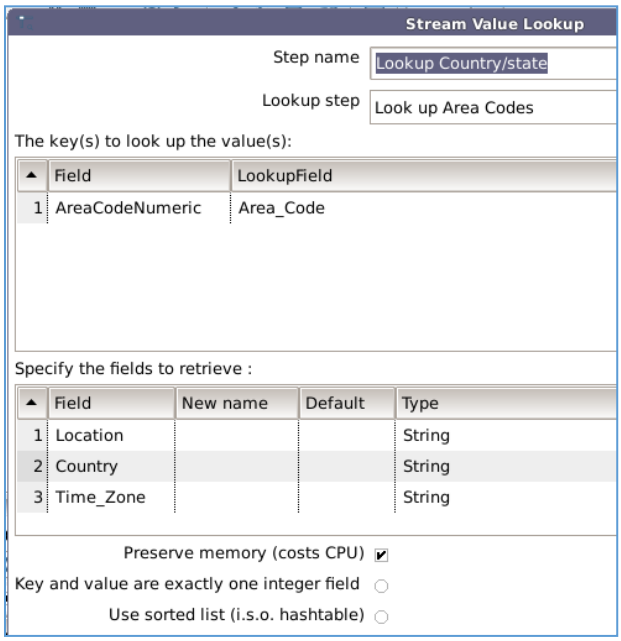


22. Click on the **Content** tab.
23. Change the **Separator** field to a `comma ,`.
24. Click on the **Fields** tab.
25. Click the Get Fields button and sample `5000` records.

26. Change the `Area_Code` Type from `Integer` to `String`.
27. The dialog box should look like this:

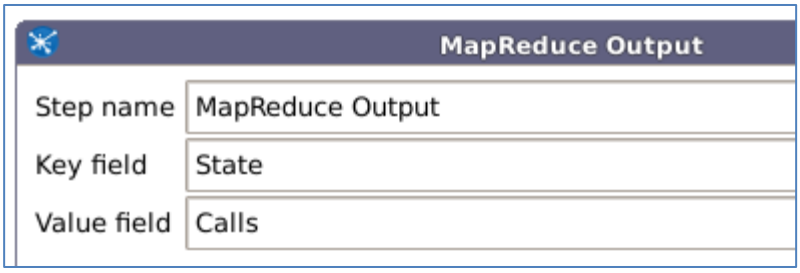


28. Click **OK** to return to the canvas.
29. Create a hop from the **Look Up Area Codes** step to the **Stream Lookup** step and choose **Main Output of Step**, to complete the hop connection.
30. Double Click the **Lookup Country/State** step to verify its properties.
31. The properties in the **Lookup Country/State** step should resemble the image below:

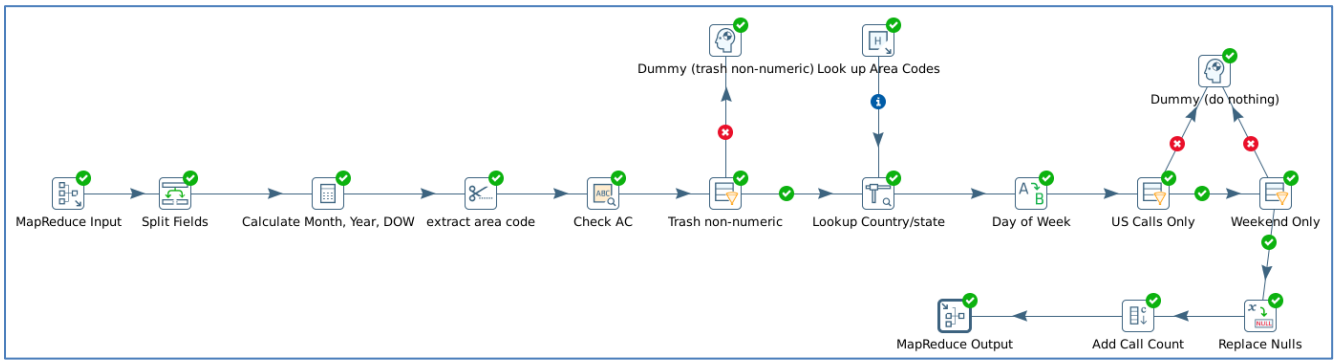


Note: we are checking this just to make sure nothing needs to be updated after you have changed the source of the reference file.

32. Now we need to replace the destination of the processed data. Delete the **Table Output** step at the end of the transformation.
33. Create a hop from **Add Call Count** to **MapReduce Output**
34. From the **Big Data** folder, select and drag **MapReduce Output** step onto the canvas
35. Double click the **MapReduce Output** step to open its properties.
36. Select `Calls` for the **Value field** and select `state` for the **Key field**. Your **MapReduce Output** step should match the following image:



37. Click **OK** to return to the canvas.
38. Your completed mapper transformation should match the following image:



39. From the **File** menu, choose **Save**.
40. Do not run this transformation. This transformation will be executed by a job you will create in the next exercise.

Create a Data Refinery Exercise 3: Extend the PDI job to execute Visual MapReduce

This final VMR exercise has you extend your existing PDI job with a second job step to execute the mapper transformation you created in the first exercise.

1. Open the `Fill Data Lake Exercise 1_Updated.kjb` job from the following directory: `/pentaho/shared_content/WorkshopTraining/01_Fill_data_lake/Solutions`
2. From the **File** menu, choose **Save As**.
3. In the Name field, specify `Create_Data_Refinery_VMR` and save to the following location: `/pentaho/shared_content/WorkshopTraining/student_files/02_create_data_refinery`

To trigger a Pentaho transformation-based MapReduce job within Hadoop, we need to define a **Pentaho MapReduce** step. We will use this step to call our mapper transformation created in the previous exercise, and define parameters such as data inputs and Hadoop cluster connection information.

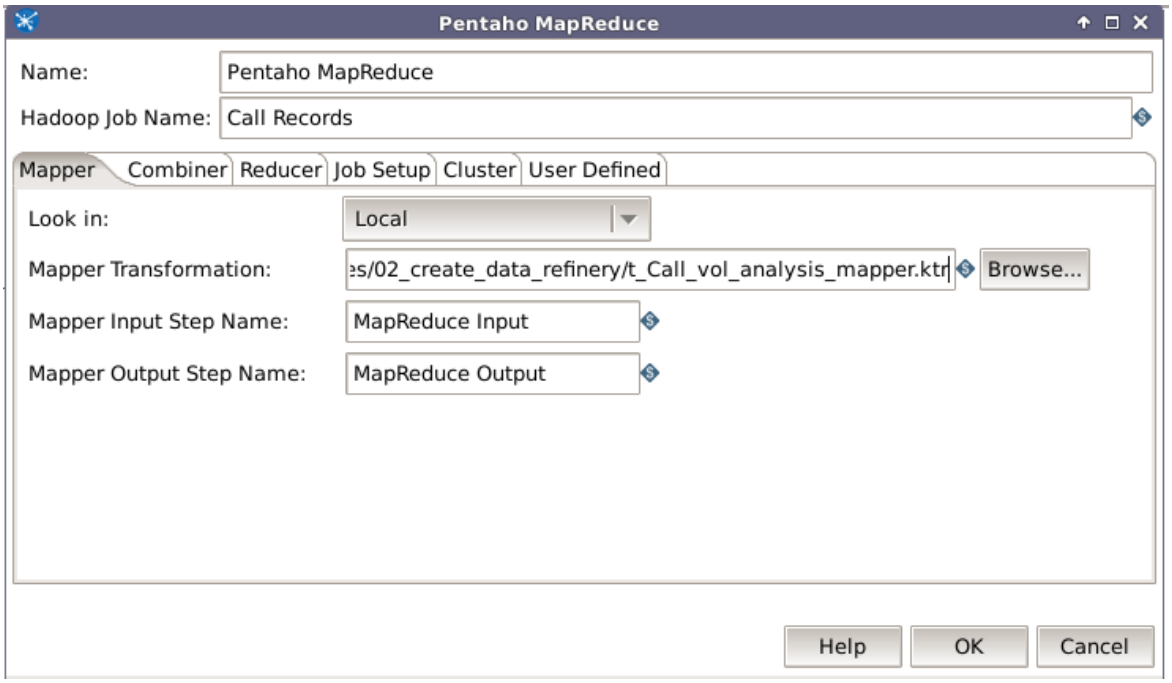
4. From the **Design** tab, expand the **Big Data** folder, then select and drag the **Pentaho MapReduce** step onto the canvas.
5. Select the **Pentaho MapReduce** step and drag it above the connecting arrow between the **Hadoop Copy Files** step and the **Success** step until the arrow becomes highlighted. Once highlighted, drop the **Pentaho MapReduce** step (release the mouse button).
6. Right click on the hop between **Pentaho MapReduce** step and the **Success** step. Select **Evaluation | Follow when result is true**. The job should match the following image:

7. Open the **Pentaho MapReduce** step. Rename the step to Pentaho MapReduce CallRecords.
8. In the **Hadoop Job Name** field, specify a name, such as `Call Detail Records`.
9. In the **Mapper** tab, next to the **Mapper Transformation** field, choose the **Browse** button. Navigate to and select this file: `pentaho/shared_content/WorkshopTraining/student_files/02_create_data_refinery/t_Call_vol_analysis_mapper.ktr`.

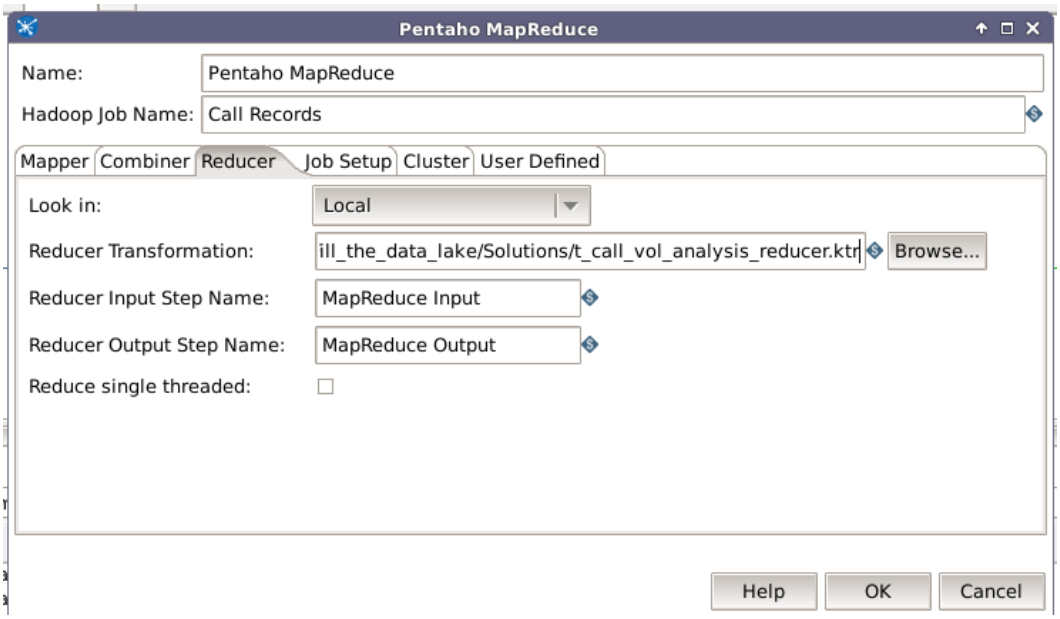
   Click **OK**.

We need to tell Visual MapReduce the names of the MapReduce Input and MapReduce Output steps defined within the mapper transformation created in MapReduce Exercise 1.

10. In the **Mapper Input Step Name** field, type `MapReduce Input`.
11. In the **Mapper Output Step Name** field, type `MapReduce Output`.
12. Your **Pentaho MapReduce** Step should match the following image:

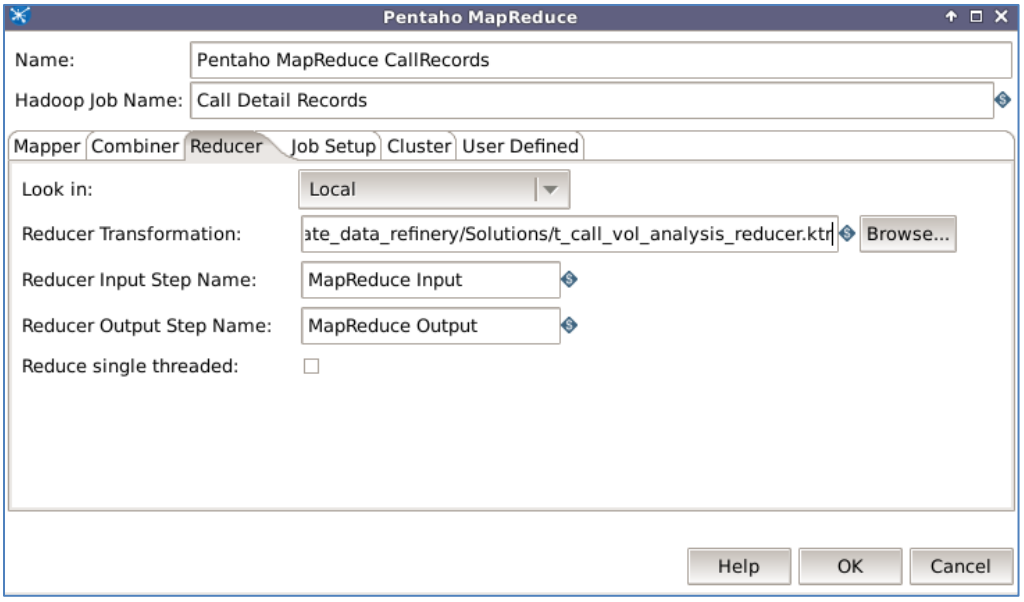


13. On the reducer tab, specify the following:



In the **Reducer Input Step Name** field type `MapReduce Input`

14. In the **Reducer Output Step Name** field type `MapReduce Output`
15. Your **Pentaho MapReduce** (Reducer tab) step should match the following image:



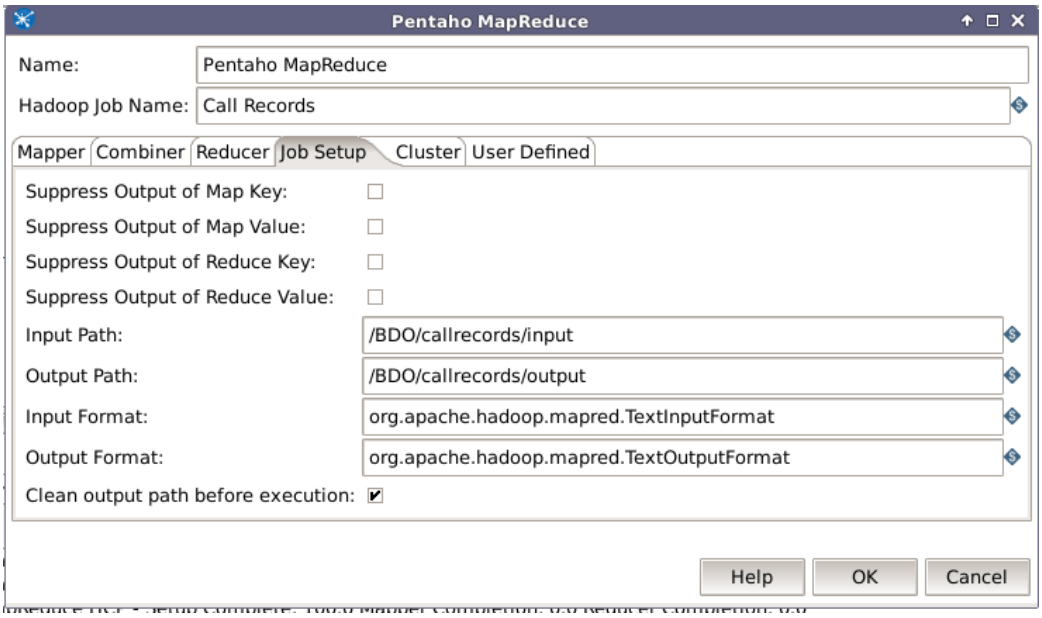
16. Click on the **Job Setup** tab.

We do not want to see the Hadoop-generated mapper key within the Hadoop output data files. And we will remove any files from our output directory before writing new data files to the directory.

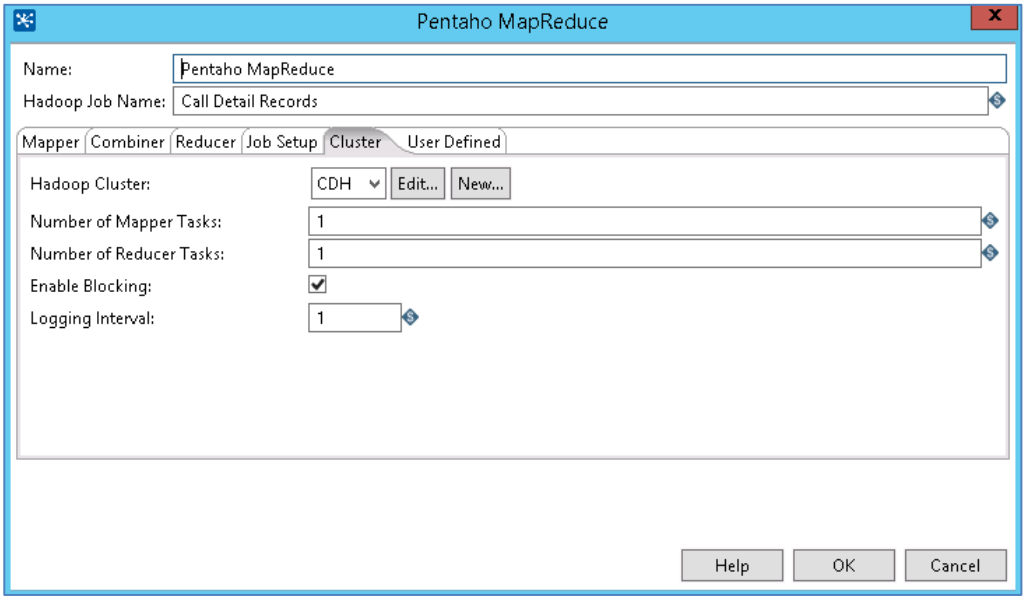
17. Click the check box for **Clean output path before execution**.

Define the **Input Path** as the directory to which we previously copied the Call Data file to as part of MapReduce Exercise 2.

18. In the **Input Path** field, specify `/BDO/callrecords/input`. This is the location in HDFS from which the MapReduce job will retrieve its input data.
19. In the **Output Path** field, specify `/BDO/callrecords/output`. This is the location in HDFS where the resulting data processed by the Mappers will be written.
20. In the **Input Format** field, specify `org.apache.hadoop.mapred.TextInputFormat`
21. In the **Output Format** field, specify `org.apache.hadoop.mapred.TextOutputFormat`
22. Your **Job Setup** tab should match the following image:



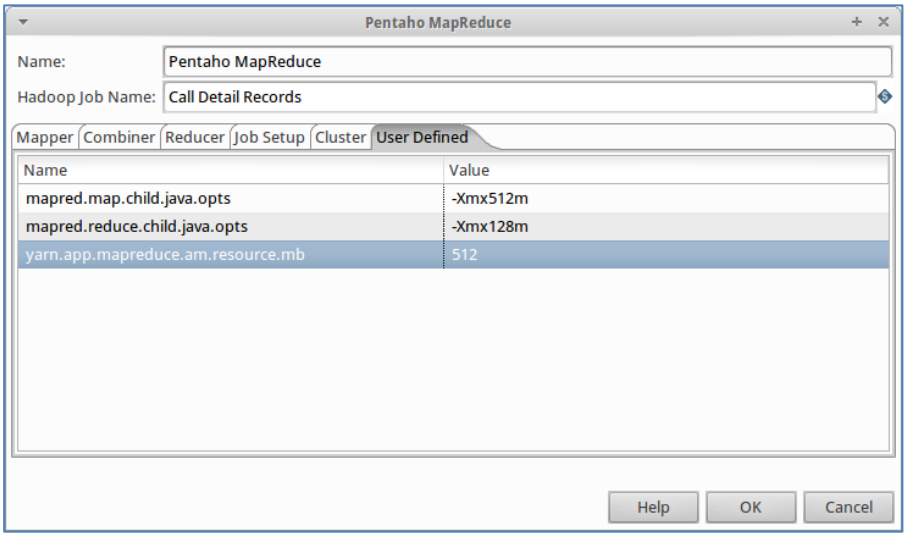
23. Next, click the **Cluster** tab to configure your connection to the Hadoop cluster.
24. In the Hadoop Cluster drop down select “CDH”
25. In the **Number of Mapper Tasks** field, specify `1`.
26. In the **Number of Reducer Tasks** field, specify `1`.
27. Check the box to **Enable Blocking**.
28. Specify a **Logging Interval** of `1` seconds.
29. The **Cluster** tab should match the following image:



30. Click on the **User Defined** tab.

It is important to tune the Java Virtual Machine (JVM) heap size to optimize performance.

31. Click on the first row under the **Name** label. Enter `mapred.map.child.java.opts`.
32. Click in the **Value** field on the same row. Enter `-Xmx512m`.
33. Add another row with **Name** `mapred.reduce.child.java.opts` and **Value** `-Xmx128m`.
34. Add one more row with **Name** `yarn.app.mapreduce.am.resource.mb` and **Value** `512`
35. The **User Defined** tab should look like this:



36. Click **OK** to return to the canvas.
37. From the **File** menu, choose **Save**.
38. Add another **Pentaho Map Reduce** step. Set that step between **Hadoop Copy files** and **Pentaho MapReduce CallRecords** steps.

    The properties of that step should be as follows:

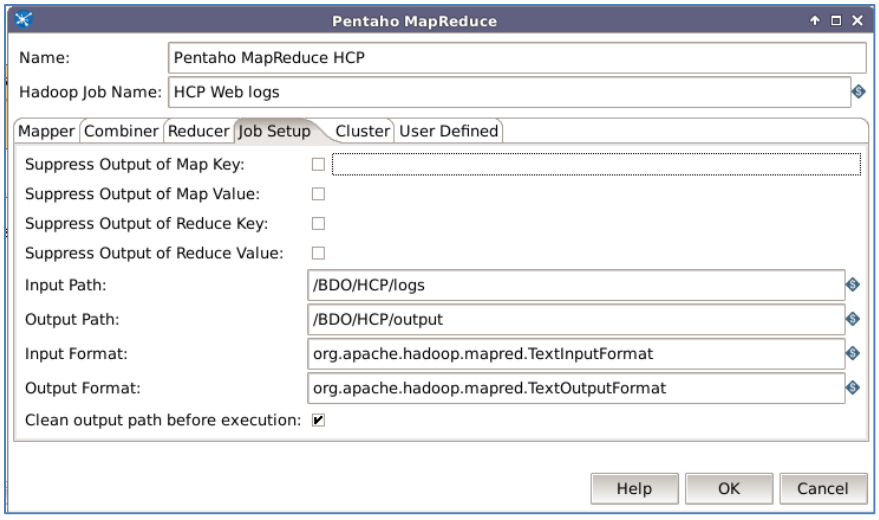
    a. **Mapper Transformation:** `/pentaho/shared_content/WorkshopTraining/01_Fill_the_data_lake/Solutions/HCP_log_parsing.ktr`

    b. **Mapper Input Step Name**: `MapReduce Input`

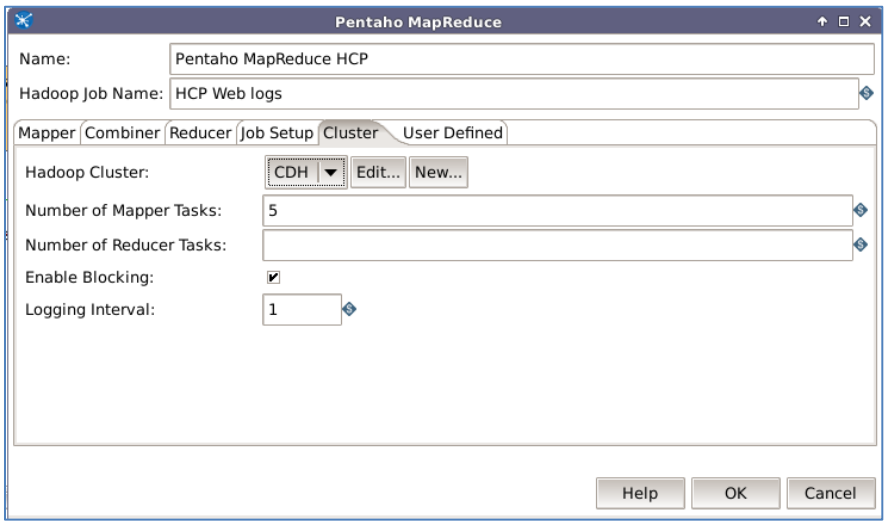
    c. **Mapper Output Step Name**: `MapReduce Output`

    Note: We will be building the HCP Log parsing transformation in an upcoming exercise

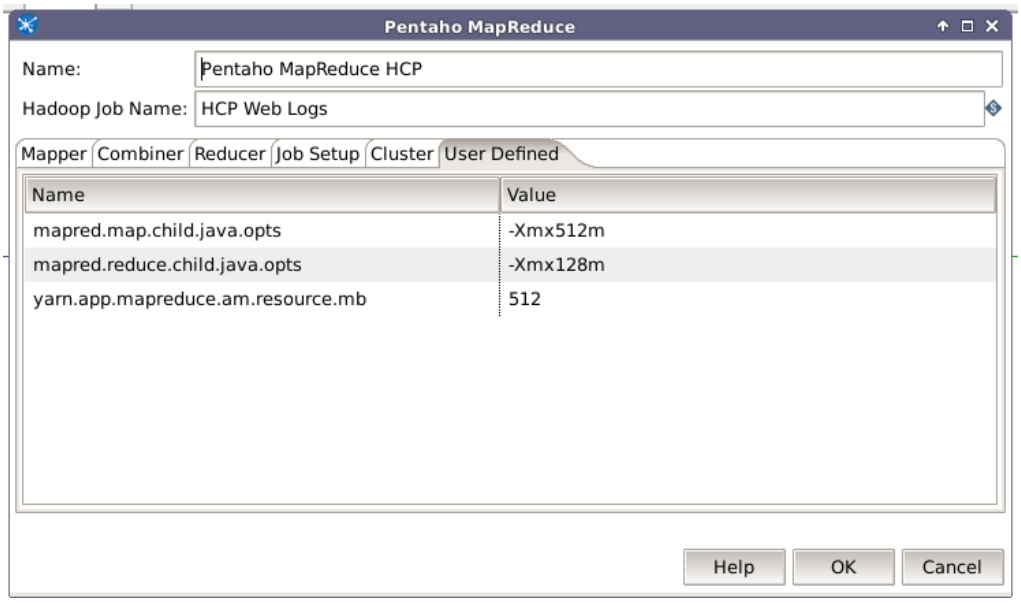
39. The Job Setup tab should look like this:



40. And the cluster tab should look like this:



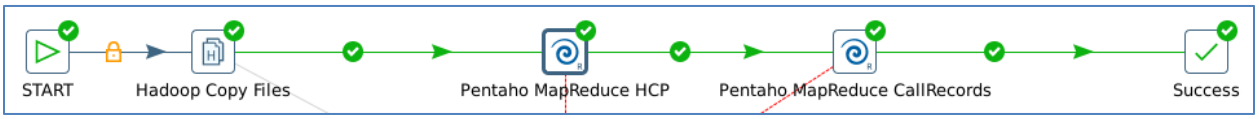
41. And the User Defined tab should look like this:



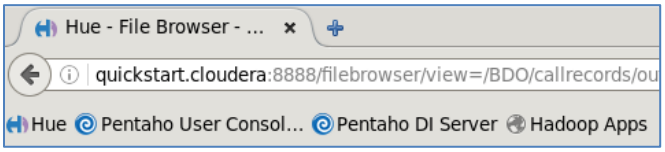
42. From the **Action** menu, choose **Run** and then click **Launch**.
43. Click the **Logging** tab located in the bottom section of Spoon to see the Mapper job progress.

> **Important note**: If the **Logging** tab shows that the mapper is stuck at 4% complete, for example, then your mapper has failed for some reason. The reason for the failure is usually a typing error or missed exercise step. However, the **Logging** tab will not display this reason. To debug this mapper job, you have to go into the mapper task logs via the YARN Resource Manager utility, bookmarked in your Firefox browser. For details on how to debug your mapper job proceed to the optional Exercise 4 on the next page.

44. When the job successfully completes, you will see a green checkmark on the **Success** step.

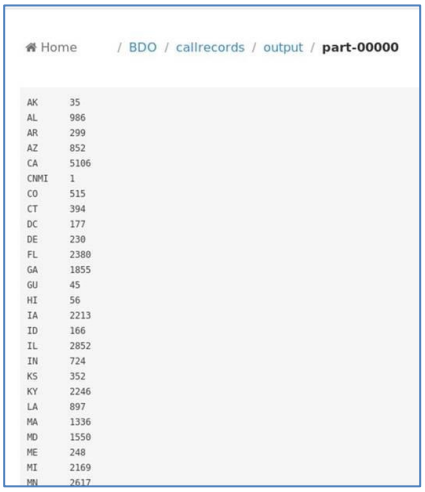


45. To view the newly processed CDR data in Hadoop, launch Firefox from the bottom launch menu icon.
46. From the Firefox bookmarks bar click **Hue**.



47. In the upper right corner of the browser, click on File Browser. Navigate to the following directory **BDO → callrecords → ouput → part-0000**

48. If your job executes successfully you will see processed and blended CDR records as shown in following screenshot:



Create a Data Refinery Exercise 4: 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: `/pentaho/shared_content/WorkshopTraining/student_files/02_create_data_refinery`.
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://quickstart.cloudera: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 2 lines:

```
#!/bin/sh

docker exec $(docker ps -a | grep -i cloudera | awk '{print $1}') sudo -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/shared_content/WorkshopTraining/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.

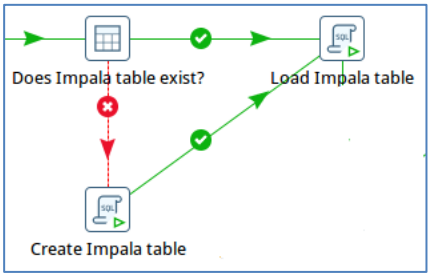
29. Expand the **Scripting** folder and drag a **SQL** step to the right of the **Does Impala table exist?** step.
30. Rename this **SQL** step to `Load Impala Table`

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

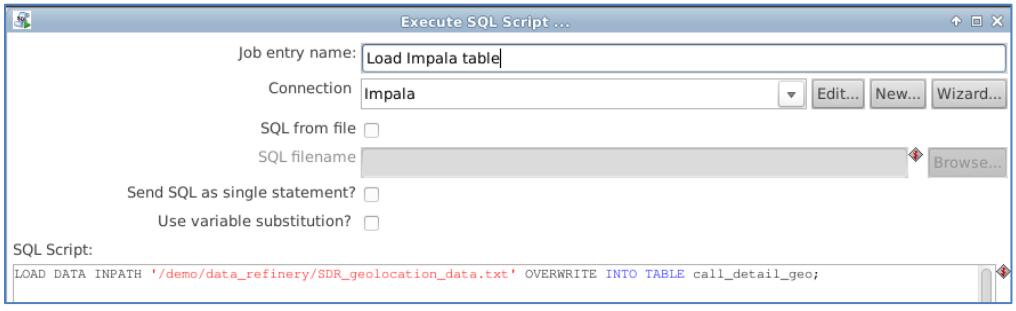
31. Drag a second **SQL** step from the **Scripting** folder and place it below the **Does Impala table exist?** step.
32. Rename this **SQL** step to `Create Impala Table`
33. Create a hop from the **Does Impala table exist?** step to the **Load Impala table** step to the right. This hop should be green 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:

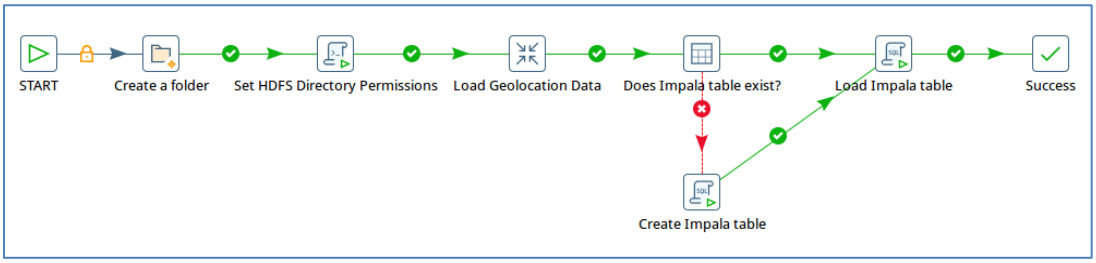


40. Click **OK** to return to the canvas.
41. Double click the **Create Impala table** step to edit its properties.
42. In the **Connection** field select `Impala`.

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

```
CREATE TABLE call_detail_geo
(
  calldate STRING,
  source_number STRING,
  home_latitude DOUBLE,
  home_longitude DOUBLE,
  distance DOUBLE,
  direction STRING,
  location_category STRING,
  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 5: 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. Now that we have both the geolocation data and the call detail

records data in Impala, 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. 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.

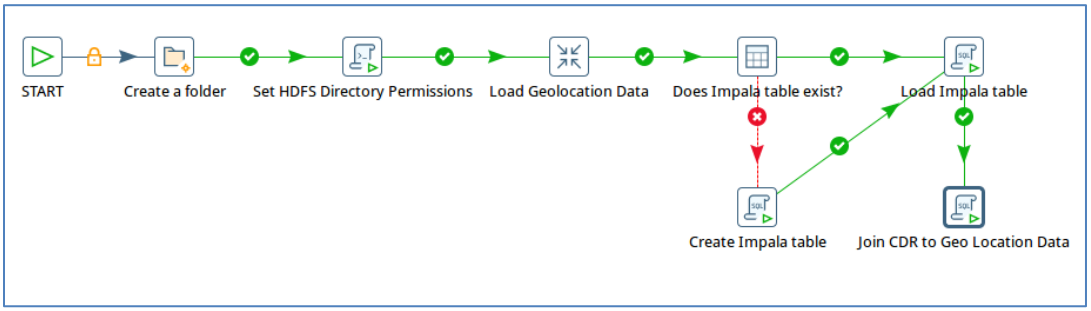
3. From the **Design** tab on the left, expand the **Scripting** folder and drag the **SQL** step onto the canvas below the **Load Impala Table** step.
4. Create a hop between the **Load Impala Table** step and the **SQL** step.
5. Double click the **SQL** step to edit its properties. In the **Step name** field type `Join CDR to Geo Location Data`.
6. In the **Connection** field select `Impala`.
7. 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 STRING
  , source_number STRING
  , call_date STRING
  , call_month INT
  , call_year INT
  , day_of_week INT
  , area_code STRING
  , state STRING
  , country STRING
  , time_zone STRING
  , weekday STRING
  , num_calls INT
  , home_latitude DOUBLE
  , home_longitude DOUBLE
  , distance DOUBLE
  , direction STRING
  , location_category STRING
  , 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);
```

8. 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.

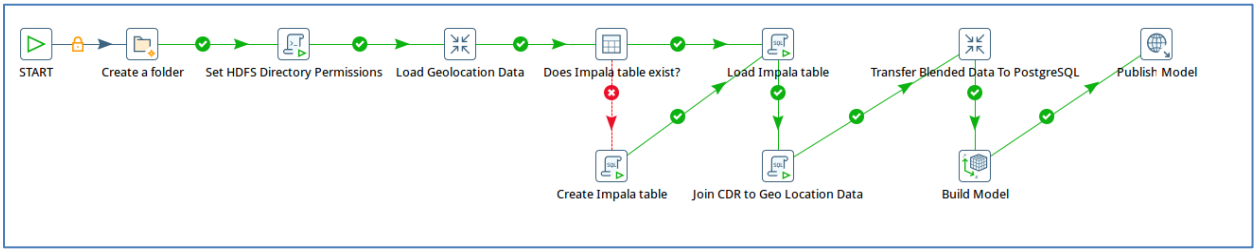
9. From the **Design** tab on the left, expand the **General** folder and drag the **Transformation** step onto the canvas to the right of the **Load Impala Table** step.
10. Create a hop between the **Join CDR to Geo Location Data** step and the **Transformation** step.
11. Double click the **Transformation** step to edit its properties. In the **Name of job entry** field type `Transfer Blended Data to PostgreSQL`.

12. On **Transformation Specification** tab click the browse icon for the **Transformation filename** field and browse to select the following file: `/pentaho/shared_content/WorkshopTraining/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.

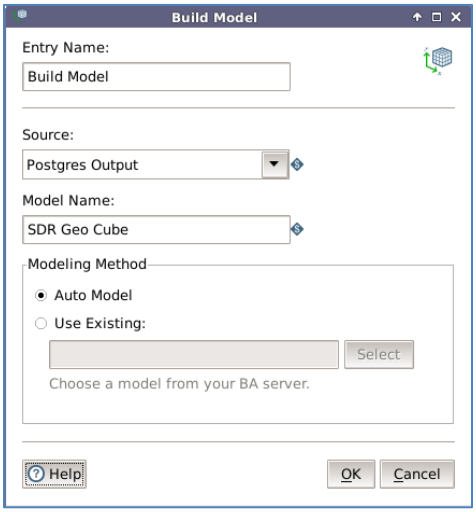
Now that our data is blended, we need to create and publish a model to the Pentaho Analytics server for web-based dimensional analysis and reporting.

13. 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.
14. Also from the **Modeling** folder drag the **Publish Model** step onto the canvas to the right of the **Transfer Blended Data to PostgreSQL** step.
15. Create a hop between the **Transfer Blended Data to PostgreSQL** step and the **Build Model** step.
16. Create a hop between the **Build Model** step and the **Publish Model** step to match the following screenshot.

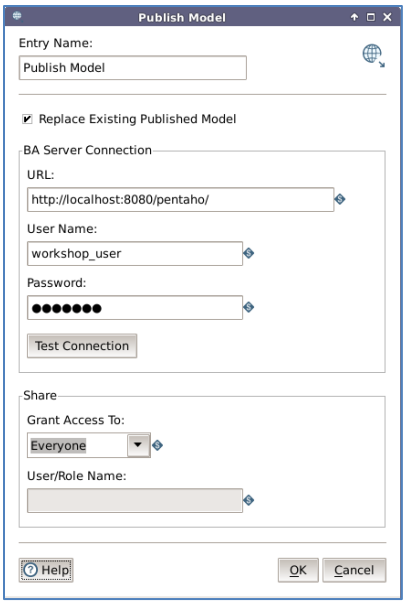


> Note: the build model and publish model steps will automatically create and publish a metadata model to the analytics server for web-based dimensional analysis on the blended data created by this job.

17. 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`.

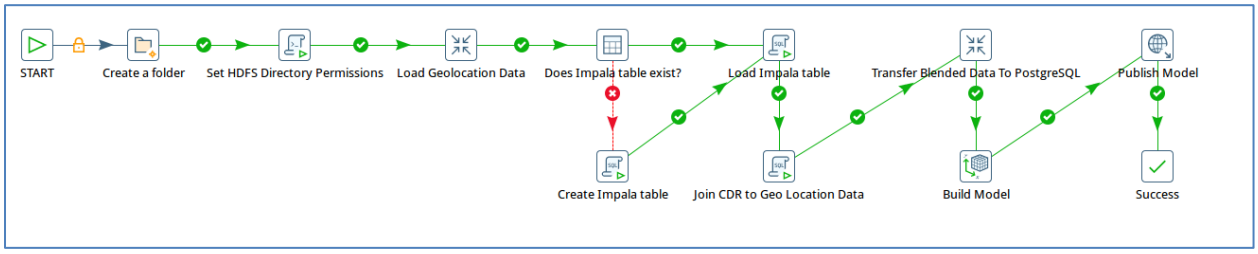


18. Double click the **Publish Model** step to edit its properties.
19. Check **Replace Existing Published Model**.
20. For the **URL** enter: http://localhost:8080/pentaho/
21. Login to the Pentaho User Console
    - User id: workshop_user
    - Password: bigdata
22. The remaining fields can be left as the default
23. Click **Test Connection**.



24. From the **Design** tab on the left, expand the **General** folder and drag the **Success** step onto the canvas below the **Publish Model** step.

25. Create a hop between the **Publish Model** step and the **Success** step to match the following screenshot.



26. From the **File** menu, choose **Save**.
27. Execute the job by choosing **Action** → **Run** from the main menu or by clicking the run icon ▷.
28. The **Execute a job** dialog will appear. Click the **Launch** button at the bottom.
29. 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! The next section contains exercises for analyzing the data you loaded to PostgreSQL.

Create a Data Refinery Exercise 6: 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.

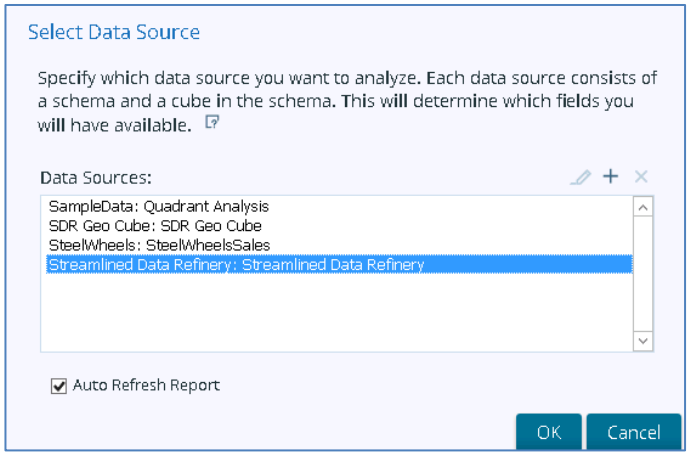
In Part One you will analyze data to determine where calls originate: from the house, the neighborhood, in town or during travel. This helps to determine which calling product has the most potential for this market. In Part Two you will plot the calls on a map based on the customers' source area code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

> Note: In case the Pentaho BA server is not started, you can start it from command line:
> `/pentaho/current_version/ctlscript.sh start`

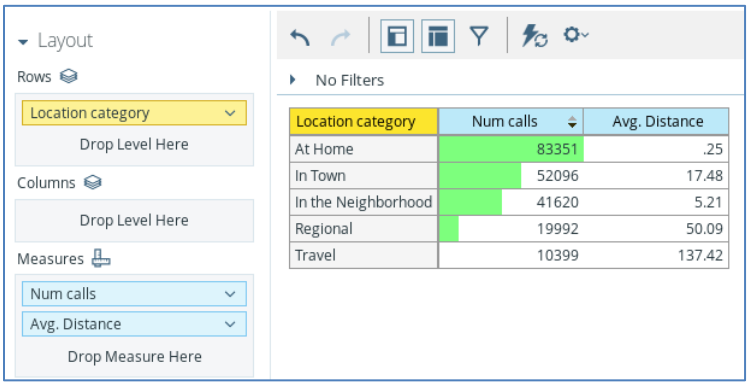
1. Login to Pentaho User Console (open the Pentaho User Console bookmark from the browser).
   - Userid: workshop_user
   - Password: bigdata
2. Click on the **Create New | Analysis Report** buttons.
3. Select the `Streamlined Data Refinery` data source from the **Data Sources** list.



1. This will launch you into a new **Analysis Report** view.
2. From the **Available fields** section on the left, double-click or select and drag `Location Category`, `Num calls`, and `Avg. Distance` to the canvas.

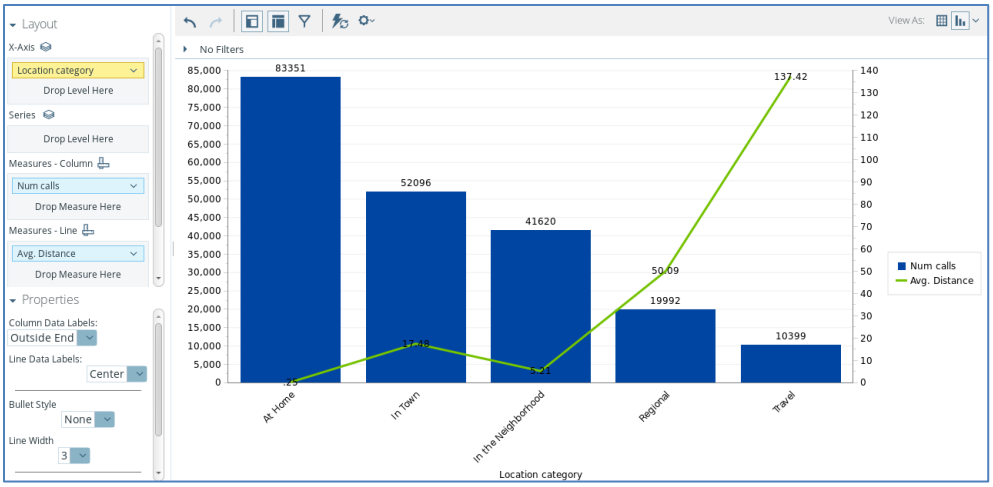
   Notice how the measures automatically aggregate the records in Analyzer.

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



   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.

5. To visualize the data, click the chart drop down in the top right of your screen and choose **Column-Line Combo**.
6. In the **Layout** section, drag `Avg. Distance` from the **Measures – Column** drop zone down to the **Measures – Line** drop zone.
7. 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:

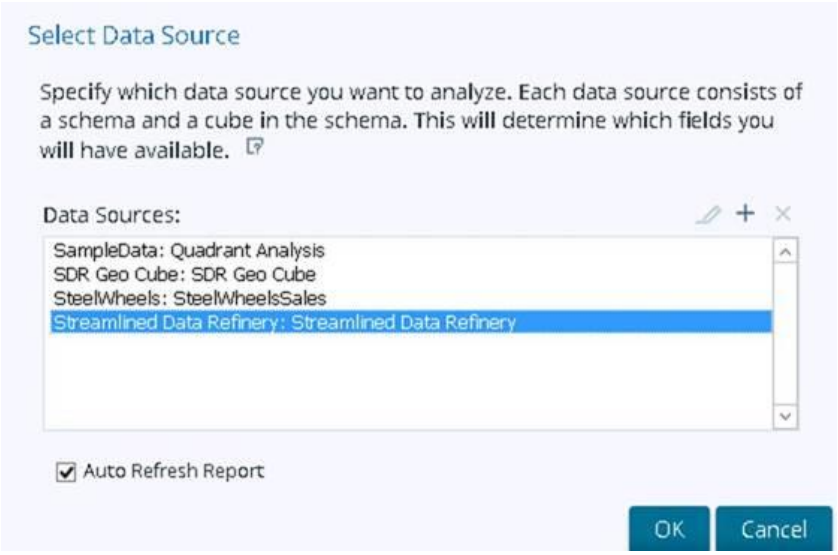


8. Click the Save icon to save the view as `SDR Analyzer Exercise 1` in the default `/Public/BDIW` directory.

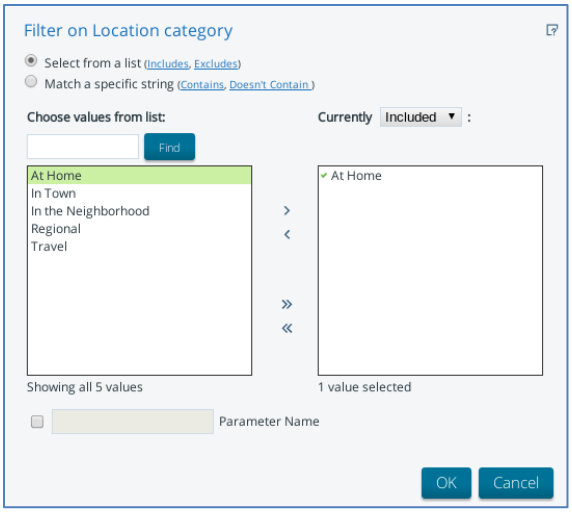
Part 2: Analyze and map calling data by geography

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.

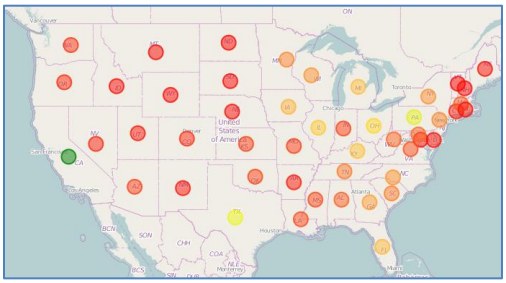
9. Click on the **Create New | Analysis Report** buttons.
10. Select the `Streamlined Data Refinery` data source at the bottom of the **Data Sources** list.



11. In the Available Fields section, right-click on `Location category` and choose **Filter** to filter the view where `Location Category` equals `At Home`.

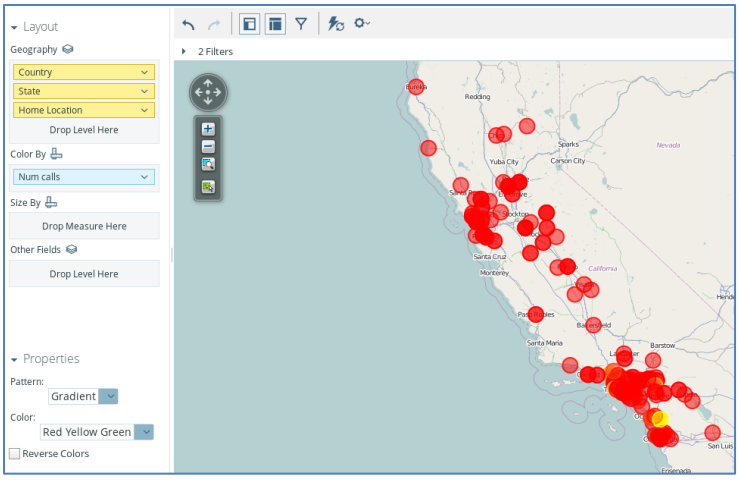


12. Add `Country, State, Num Calls` to the canvas.
13. Click the chart dropdown and select to **Geo Map**.



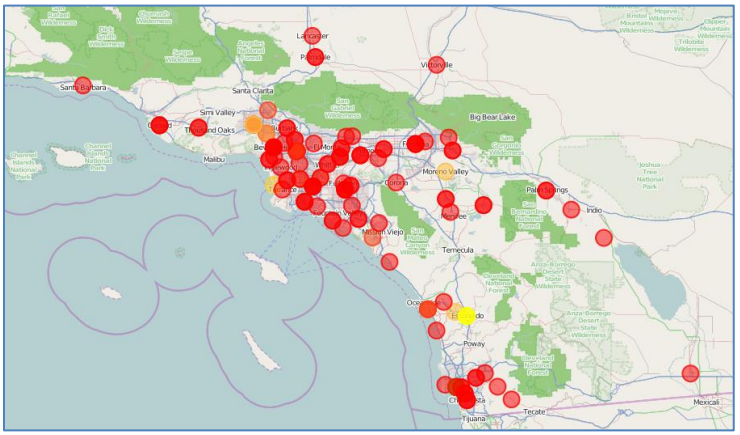
Note California has the most calls at home as denoted by the darkest green circle.

14. Drill into `California` by double clicking on the state's green circle.
15. If presented with the alert below, click OK to display the more detailed map.



If the zoom level is too low, click the center of the navigation controls  to reset the map.

16. Click the lasso filter  and then using your mouse to click and drag a lasso filter around the circles representing Southern California, once selected, click the **Keep Only** button.



We can conclude that Southern California is a good place to pilot our VOIP service based on the number of callers who make calls close to home on their cell phones.

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 that 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 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 queries Impala, joins two tables, and then categorizes the drivers into three buckets: Low Exposure, Medium Exposure, and Large Exposure. You will then load this data into a database and perform analytics directly in PDI.

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

1. In Spoon, create a new **Transformation**
2. From the **Input** folder, select and drag the **Table Input** step.

   Note: you'll need two of those, so drag it to the canvas twice.
3. Double click on the first **Table Input** step to edit the step properties. Set the properties as follows:
   a. Step Name: `CDR Data`
   b. Connection: `Impala` (select from the drop down)
   c. SQL: Type the following SQL Command in the SQL window

```
SELECT
  key
, call_date
, source_number
, call_month
, call_year
, day_of_week
, area_code
, state
, country
, time_zone
, weekday
, num_calls
FROM call_detail_records
ORDER BY source_number
```

4. Click Preview to preview the data. In the sample size, `type 500`. You should see data coming back.
5. Click `Close` and then `Ok`
6. Now, let's update the properties of the second **Table Input Step**. Double click it to update the properties.
7. Rename the step to `IoT Data`
8. For the connection, select Impala from the drop down list.
9. In the SQL Window, type the following Select query:

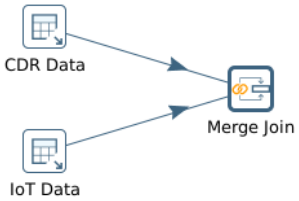
```
SELECT
  source_number
, home_latitude
, home_longitude
, distance
, direction
, location_category
, new_lat
, new_long
FROM call_detail_geo
```

```
Order By source_number
```

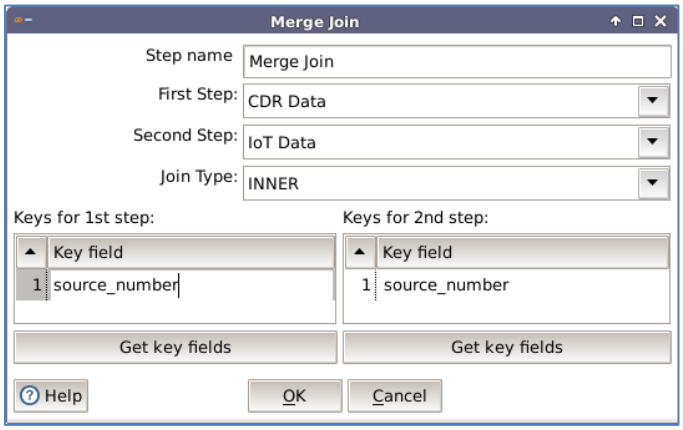
10. Click Preview to ensure that you can see data.
11. Click `Close` and then `Ok`
12. Next, we will need to blend the data from these two sources. From the **Joins** Folder, select and drag **Merge Join** step onto the canvas.

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.

13. Create a hop from each of the **Table Input** steps to the **Merge Join** step. Your transformation should look like the following:



14. Double-Click the **Merge Join** step to update its properties.
15. Set the properties as follows:
    a. Step Name: `Merge Join`
    b. First Step: `CDR Data`
    c. Second Step: `IoT Data`
    d. Join Type: `Inner`
16. Click on **Get key fields** for 1st Step and for 2nd Step, this will get all the fields from each respective input step. Remove all but `source_number` from each list of fields.
17. Your **Merge Join** Properties should resemble the image below:

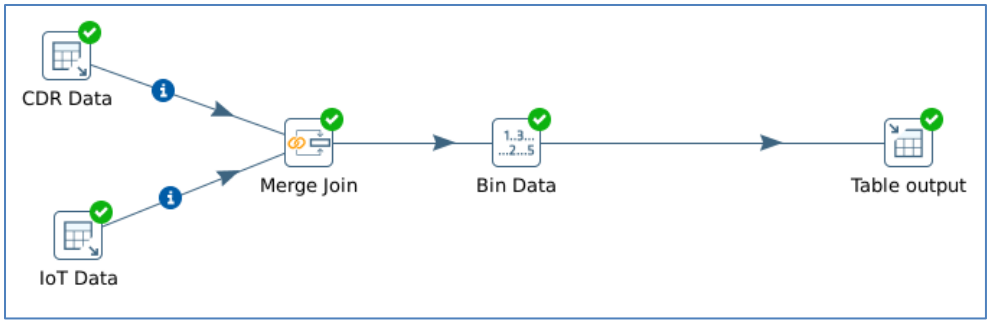


Sometimes, you need to add calculated fields to your data to provide additional analytics. We will create a new field called Exposure, to categorize our data for analysis. The number range step in PDI, allows you to categorize data based on thresholds.

18. From the **Transform** Folder, select and drag the **Number Range** step onto the canvas.
19. Create a hop from the **Merge Join** step to the **Number Range** step
20. Double Click the step to update its properties.
21. Rename the step to `Bin Data`. Update the properties as follows
    a. **Input Field**: `distance`
    b. **Output field**: `Exposure`
    c. **Default Value**: `Unknown`
22. In the **Ranges** window, specify the following ranges:

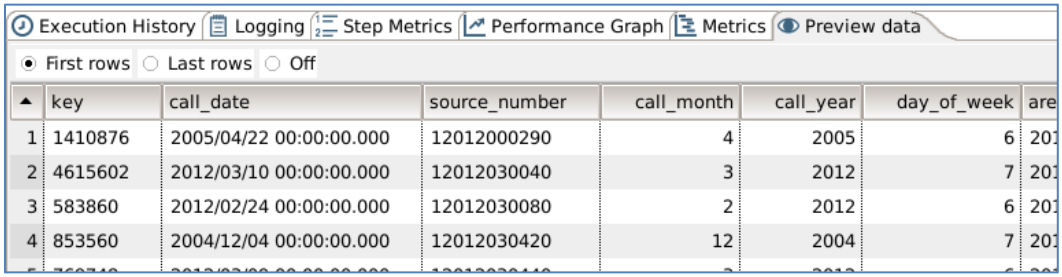
	Lower Bound	Upper Bound	Value
1	0.0	10.0	Low Exposure
2	10.0	25.0	Medium Exposure
3	25.0		Large Exposure

    Click **OK**
23. From the Output folder, select and drag the **Table Output** Step onto the canvas and create a hop from your Bin Data step to the **Table Output** Step.
24. Double-Click the **Table Output** Step to update its properties. We will be loading the blended data into postgres.
    a. **Connection**: `workshop_postgres`
    b. **Target table**: `Blend_CDR_IoT`
    c. **Commit size:** `1000`
    d. Make sure to `Check` the **Truncate Table** box
    e. At the bottom of the window, Click on **SQL**. This will bring up a window with the create table command. Click **Execute** to make sure the command completes successfully. Click Close once it completes.
    f. Leave the rest of the properties with the default values. Click **OK**
25. Your transform should look like the image below:



26. Save the transformation here: `/pentaho/shared_content/WorkshopTraining/student_files/03_Self_service_data_prep/`

27. Run the transformation. When the transform completes successfully, you'll be able to preview the data by selecting the **Preview** tab at the bottom

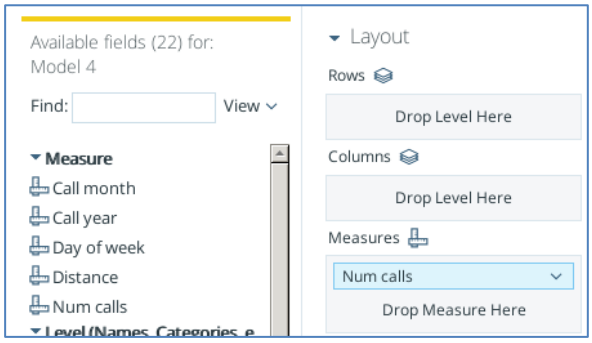


	key	call_date	source_number	call_month	call_year	day_of_week	are
1	1410876	2005/04/22 00:00:00.000	12012000290	4	2005	6	201
2	4615602	2012/03/10 00:00:00.000	12012030040	3	2012	7	201
3	583860	2012/02/24 00:00:00.000	12012030080	2	2012	6	201
4	853560	2004/12/04 00:00:00.000	12012030420	12	2004	7	201

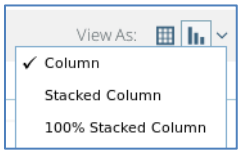
In addition to viewing the data in a tabular view, we can visualize the data an analyze it directly within PDI. We will try that next.

28. Right Click on the **Table Output** step and select **Visualize**. (Visualize will appear all the way at the bottom of the menu). In the sub-menu, choose **Analyzer**. This will bring up the **Visualize** perspective of PDI. Here you will be able to drag the objects into the **Layout** pane to see a graphical representation of the data.

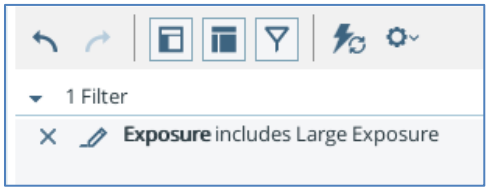
29. From the **Measures** on the left-hand side, select **Num Calls** and drag it across to the **measures placeholder** (where it says Drop Measure Here) under Layout.



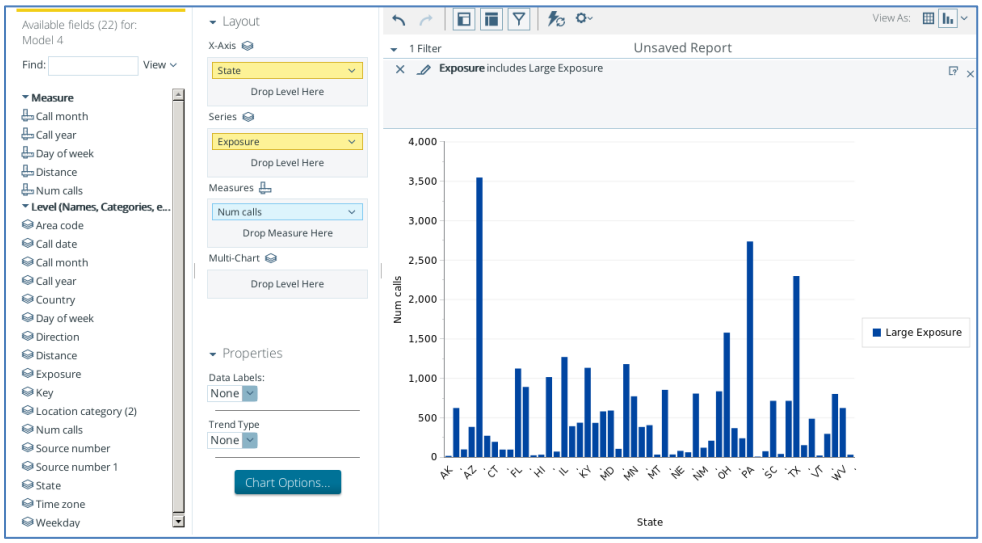
30. Drag **State** to rows and **Exposure** to Columns
31. Select **Column** chart to represent the data in the **View As** options.



32. You may want to filter out some data to focus your attention on a subset. To add a filter, open the **Filter Pane** by clicking on the filter Icon.
33. Drag the fields you would like to use in the filter and set the condition.



34. Create a filter to only include Large Exposure bins. Your resulting visualization should look like the following:



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

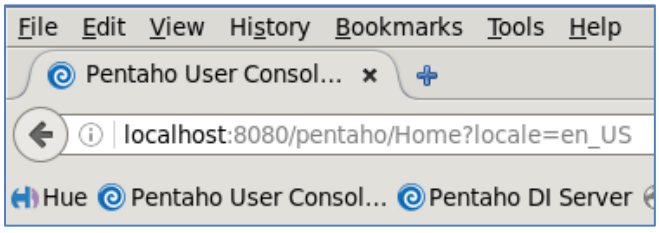
- Better Together: Improve analytics and lower costs with business intelligence and data integration together in a single platform
- Simplified Analytics: Drag and drop interfaces for building analytic applications with the right data, in the right format at the right time for better decision making 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 **2 exercises** to perform interactive query and analysis on data from Impala and HBase.

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

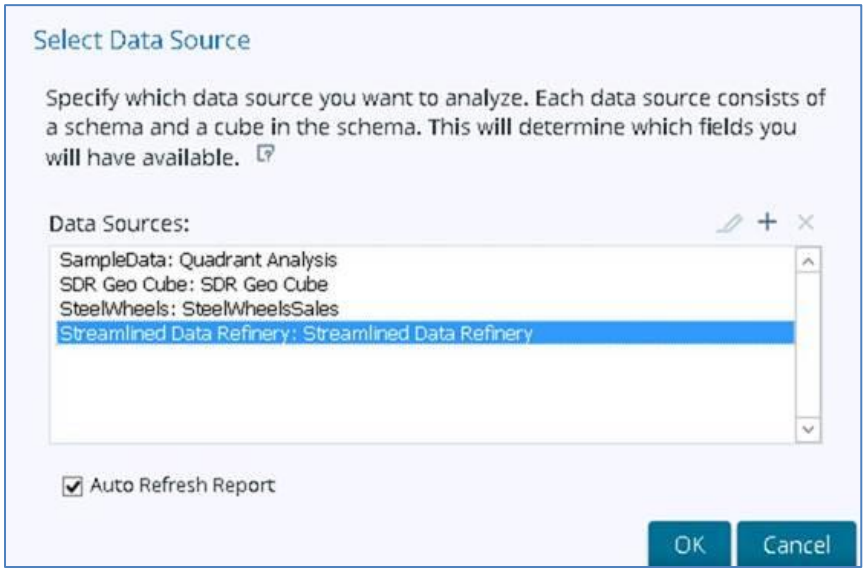
1. From a web Browser, connect to Pentaho User Console



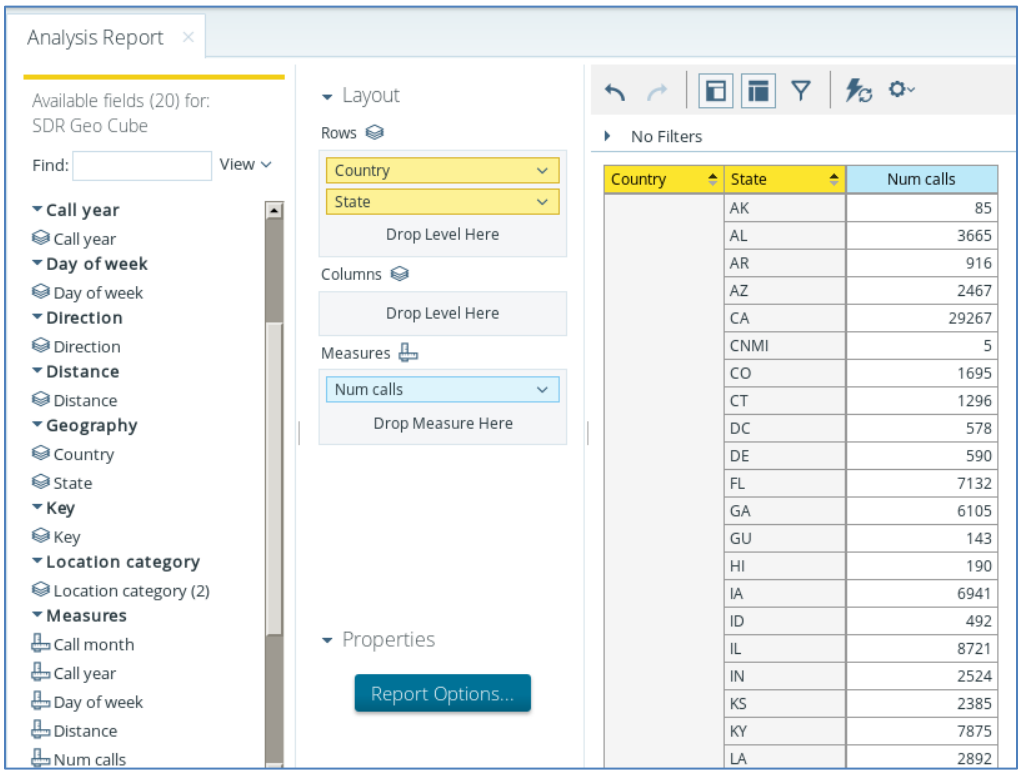
using the following credentials:

- Username: workshop_user
- Password: bigdata

2. From the Pentaho User Console Select **Create New**. Select Analysis Report
3. Select the Streamlined Data Refinery Data Source

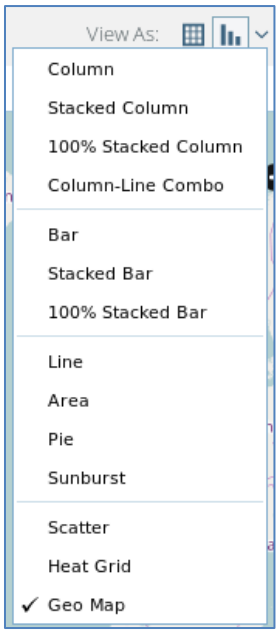


4. Create the following visualization by dragging Country and State to Rows and dragging Num Calls to Measures.

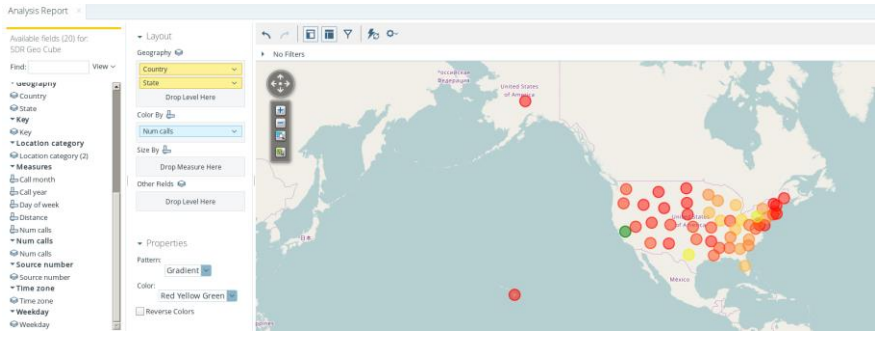


Sometimes, a field may show up as a dimension and as a measure in a data model. Be sure you pull the measure Num Calls into your report rather than the dimension.

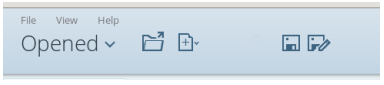
5. Then select the **Switch to Chart Format** pull down menu  then select **Geo Map**



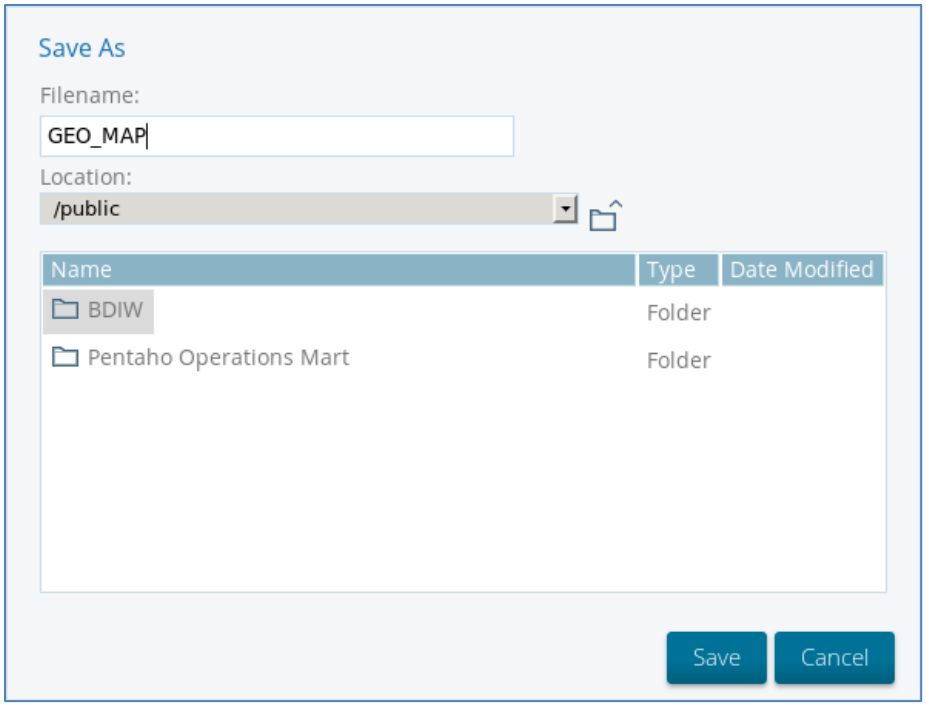
Your resulting visualization should look like the following:



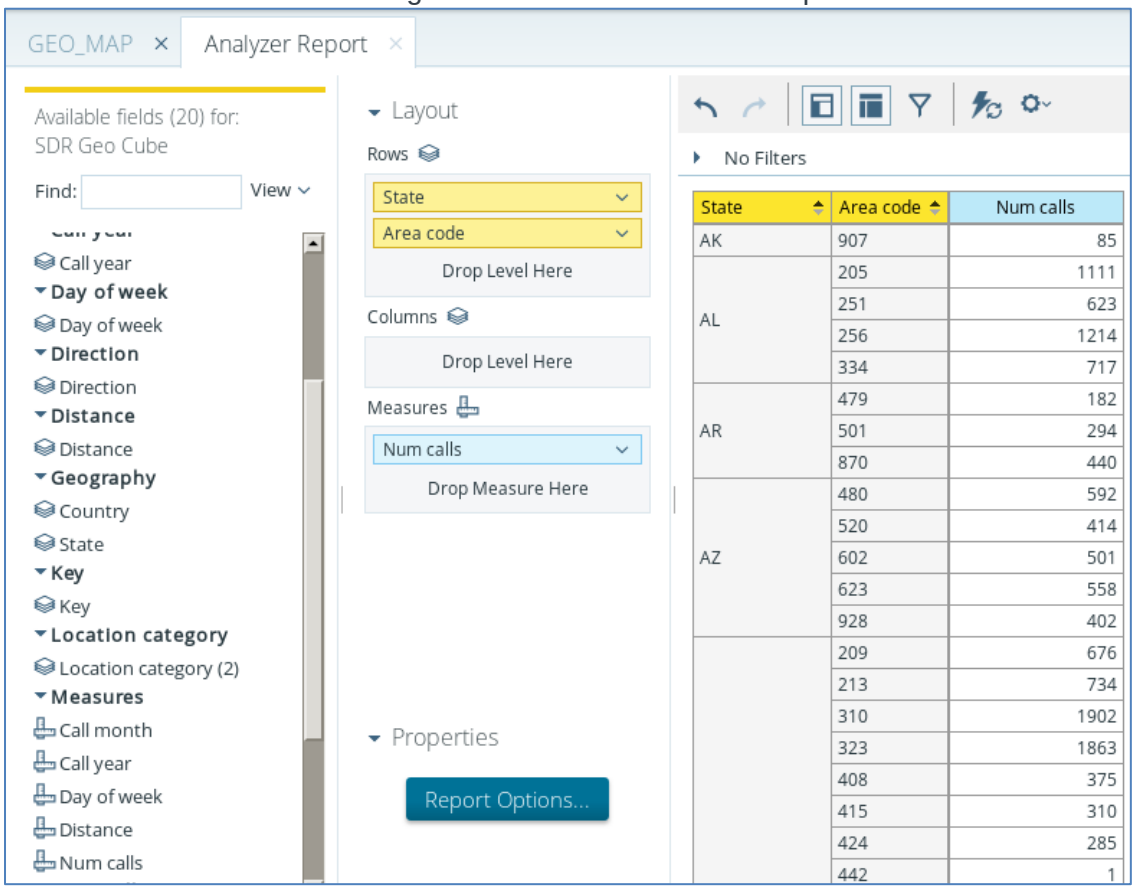
6. Save the Visualization under the **/public/BDIW** folder by clicking on the `save` icon.



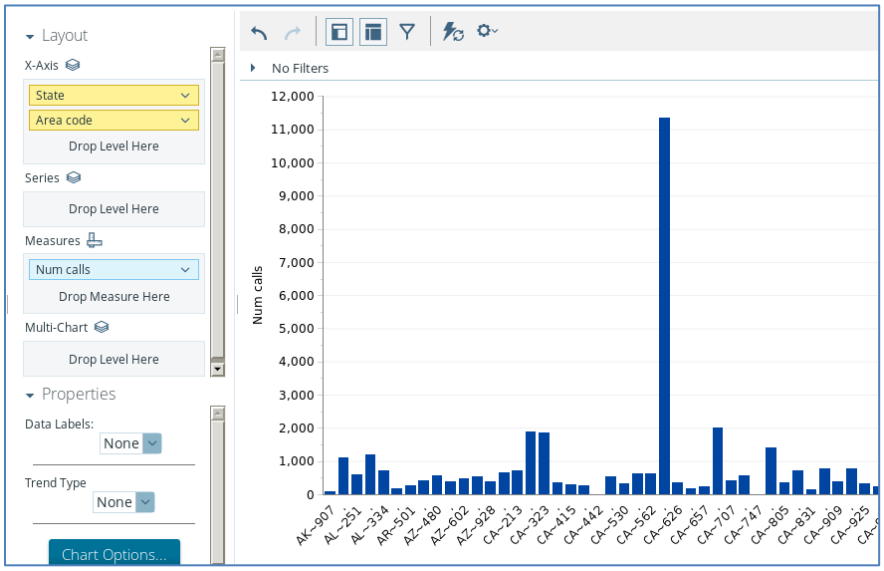
Name this analysis `GEO_MAP`.



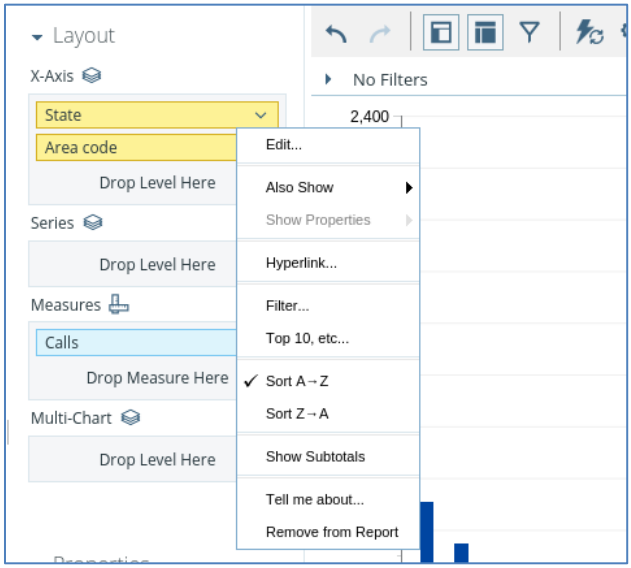
7. Create a new visualization using the same Data Source as depicted below:



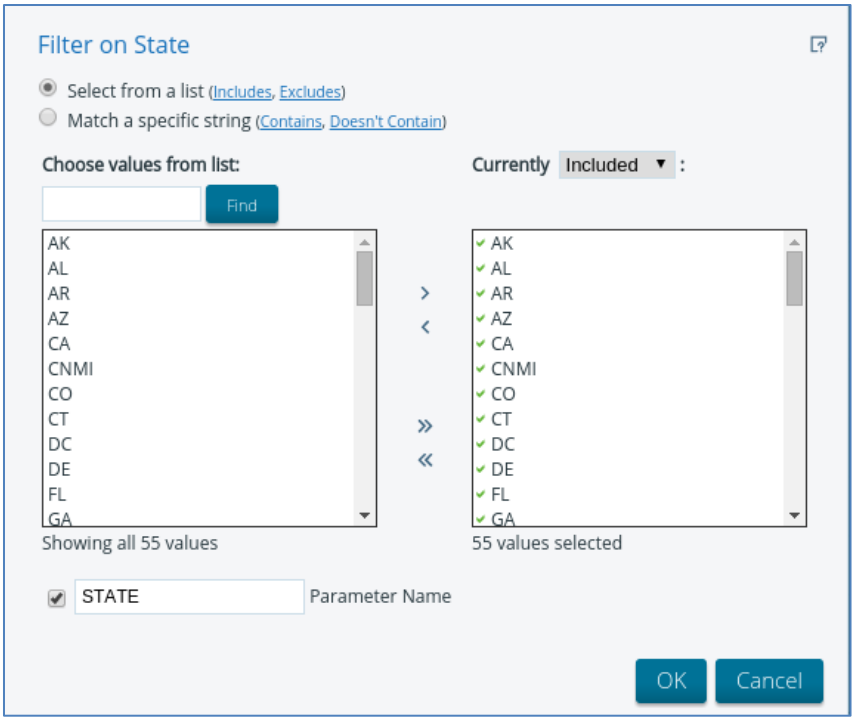
8. Change the **View As** to **Column** – Click **OK** if you see a warning about too many records.
9. Your visualization should now look like this:



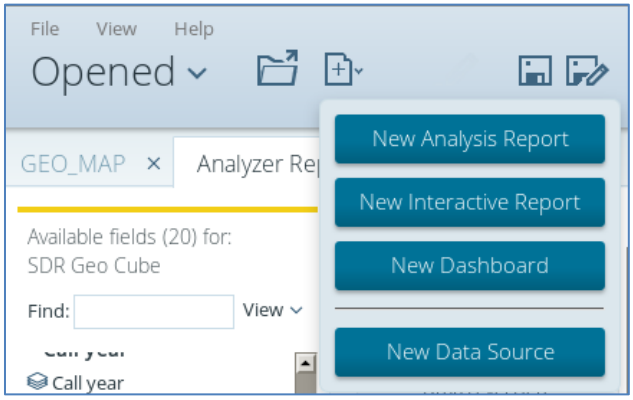
10. Right Click on **state** and add a filter on that field:



11. On the **filter options** select:
    - Select from a list.
    - Include all states by clicking the >> icon
    - Tick the box next to Parameter Name and set the name to `STATE`



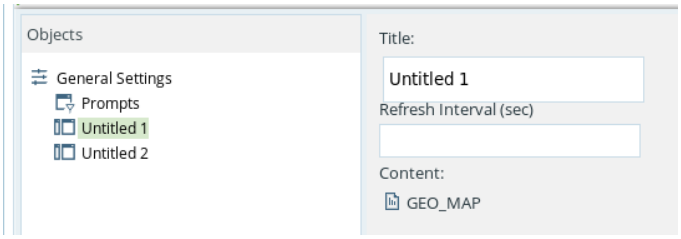
12. Click `Ok`
13. Save this Analysis. Name this analysis `CALLS_BY_AREA_CODE`.
14. From the top tab, click on the Icon with a **Plus Sign**. Select **New Dashboard**.

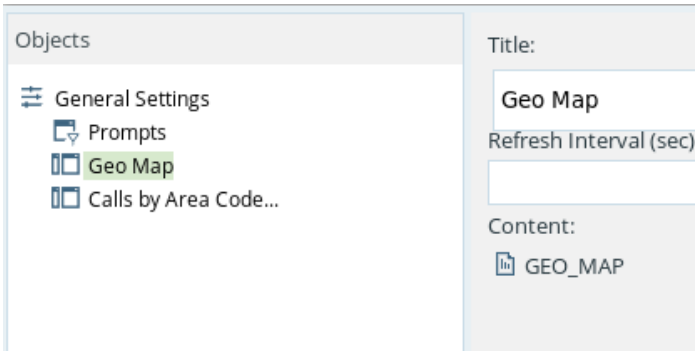


15. Select the 2 column template:

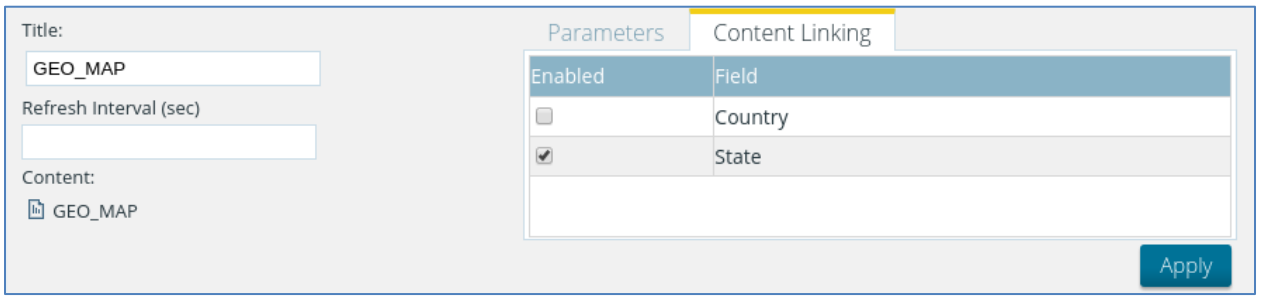
16. In the left pane, navigate to where you saved your reports. Drag the `GEO_MAP` to the first dashboard column and then drag the `CALLS_BY_AREA_CODE` to the second dashboard column

17. In the objects section of the dashboard designer, replace untitled 1 and untitled 2 with the names of your two reports.

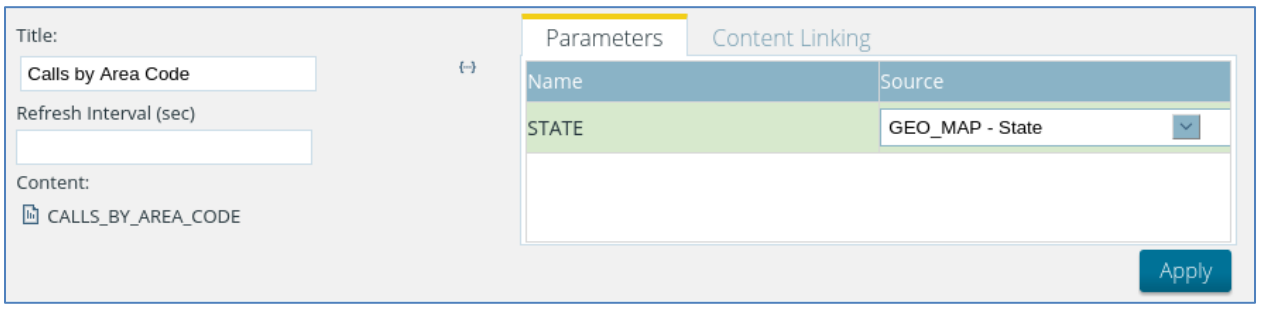




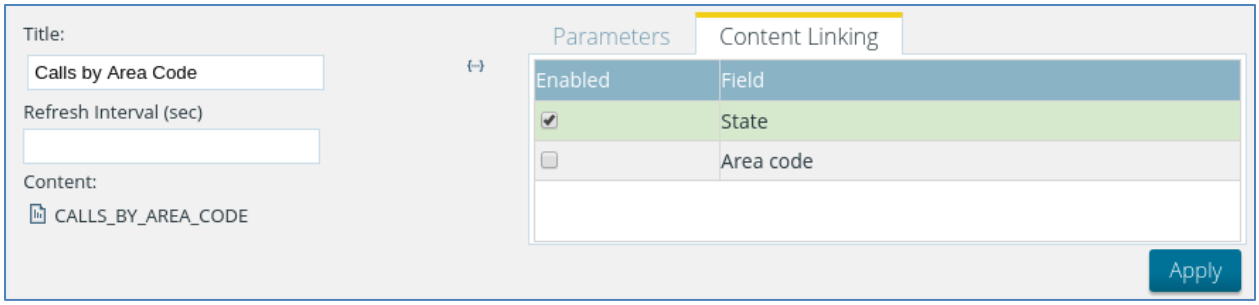
18. Select (focus on) the map component and then select the Content Linking Tab and enable State:



19. Select (focus on) the Call by Area Code component and then on the State Parameter select “GEO_MAP – State”

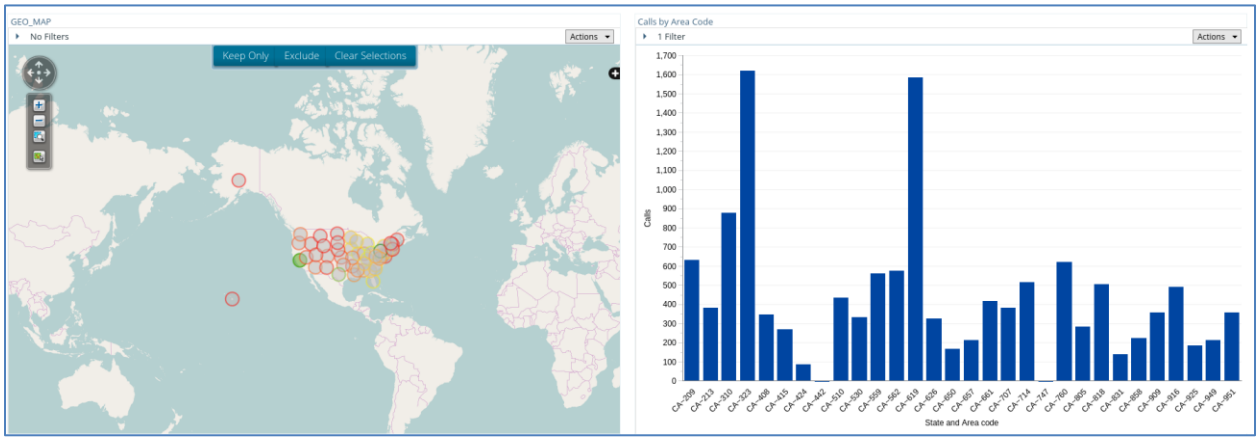


20. On the Content Linking tab select State:

21. 

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

23. 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:

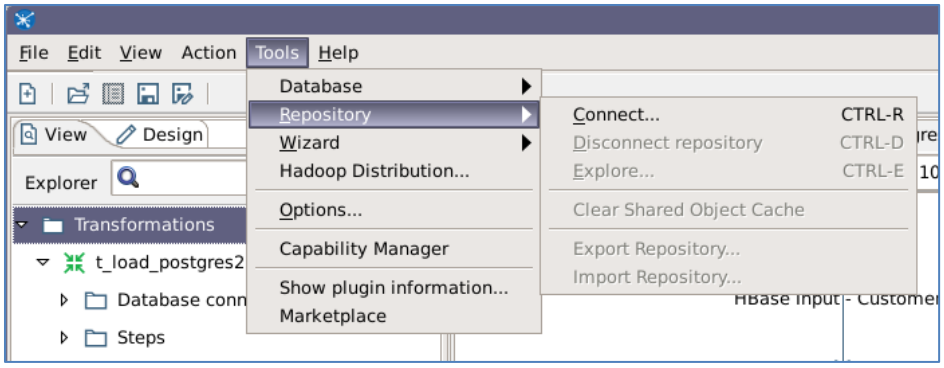


Self Service Analytics Exercise 2: Use Pentaho to report on Hbase data

1. Open the following transformation from Spoon:

`/pentaho/shared_content/WorkshopTraining/04_Self_Service_Analytics/Solutions/t_load_postgres2.ktr`

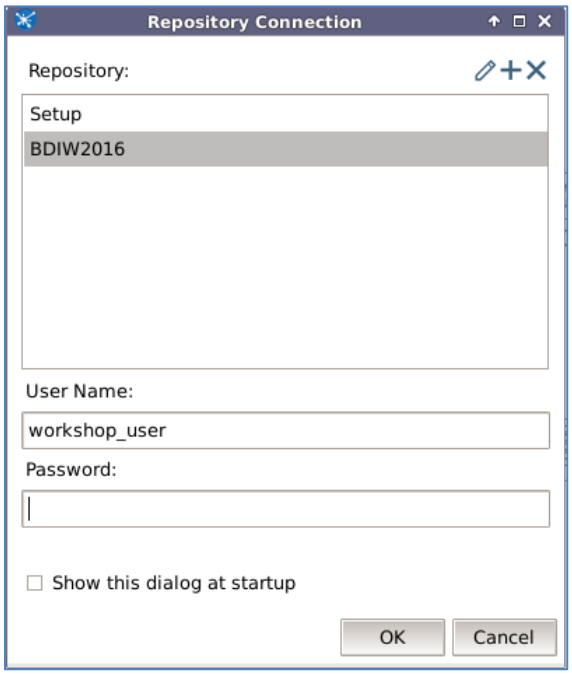
2. Connect to the Pentaho Data Integration Repository:
   a. Click on **Tools → Repository→Connect**



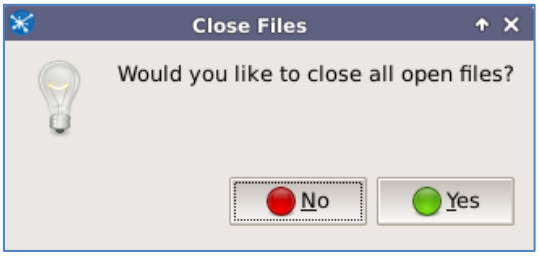
3. Select the BDIW2016 repository using the following credentials:

   User Name: `workshop_user`

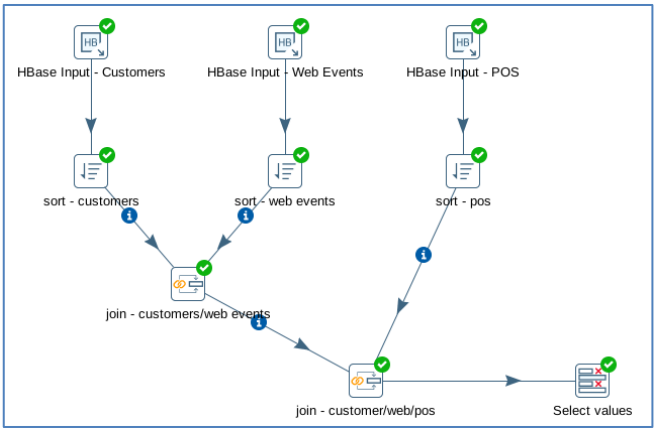
   Password: `bigdata`



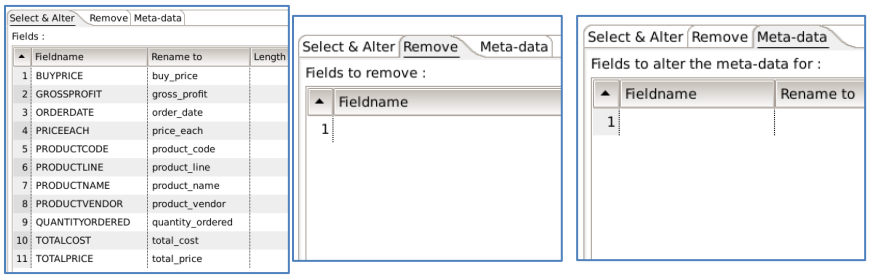
4. When prompted to close all open files, click `NO`:



5. Your transformation should look like this:

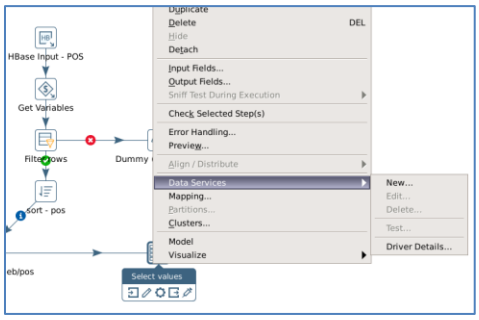


Edit the **Select Values** step. If you see any fields listed under **Remove** or **Meta-data** tabs, remove them. Do not change the fields in the **Select & Alter** tab. The tabs should look like this.

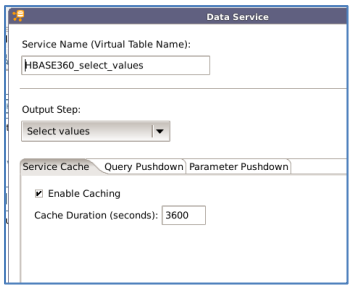


Hit **OK** to save this step.

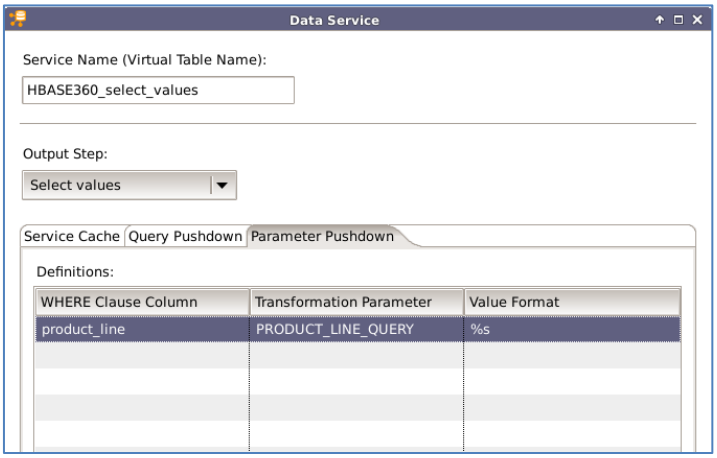
6. Right Click on the **Select values** step and create a new **Data Service**:



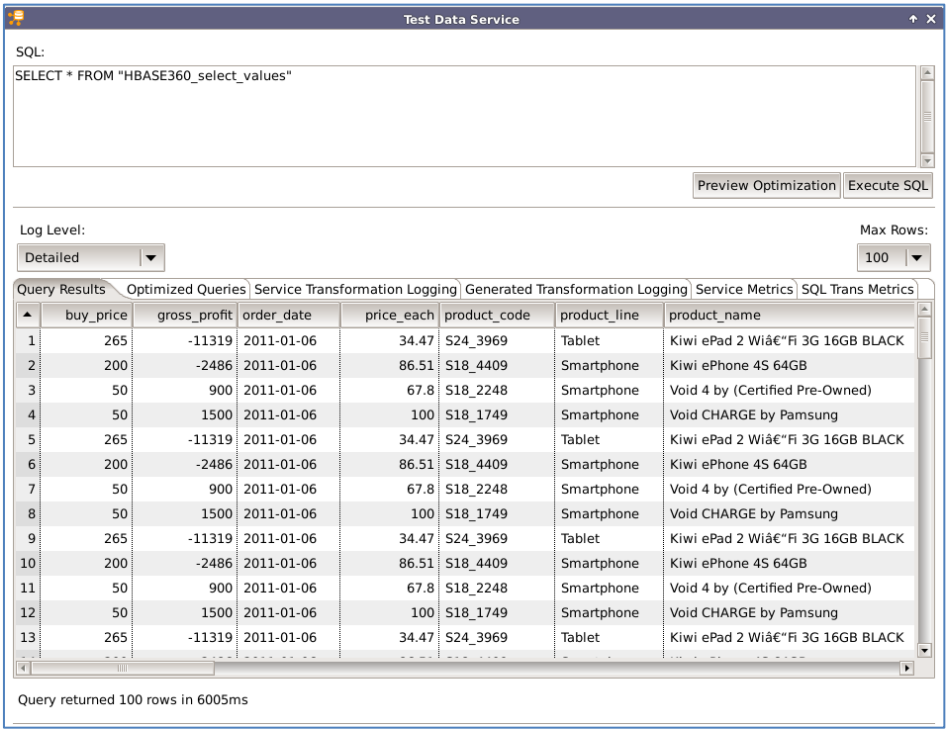
7. Call the new **Data Service** `HBASE360_select_values`



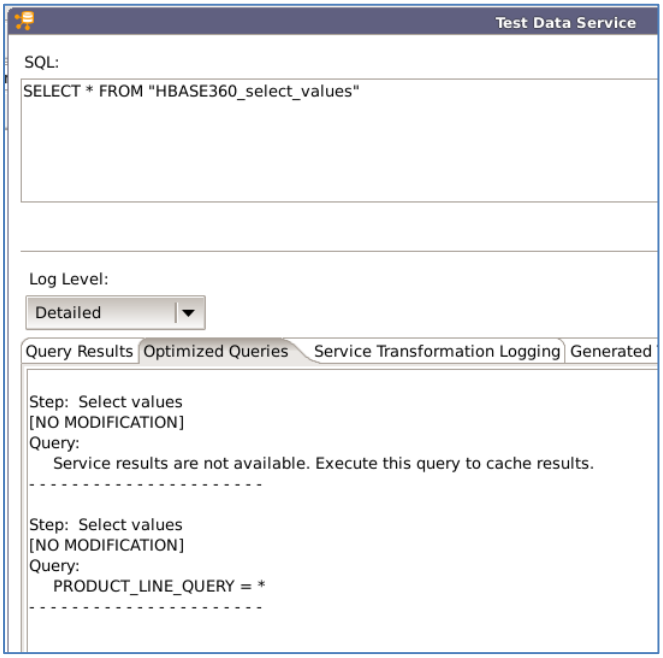
8. Select the **Parameter Pushdown** tab and select `product_line` from the **WHERE clause column.** Your data Service should look like this:



9. Click the **Test Data Service** button and then **Execute SQL** on the pop-up window. You should see results like this:



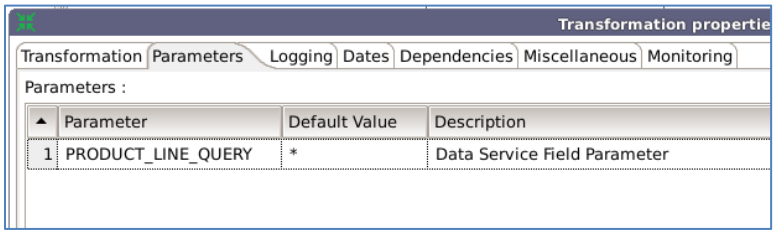
10. The **Optimized Queries** tab should look like this:



a. Click **Close**

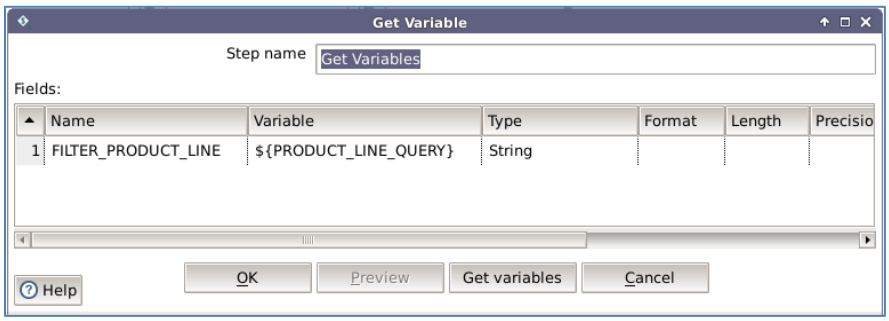
b. Click **OK**

11. Save the transformation. Double Click on the **Canvas** (white space) then select the **Parameters** tab and enter `*` `(asterisk)` under the Default Value column:



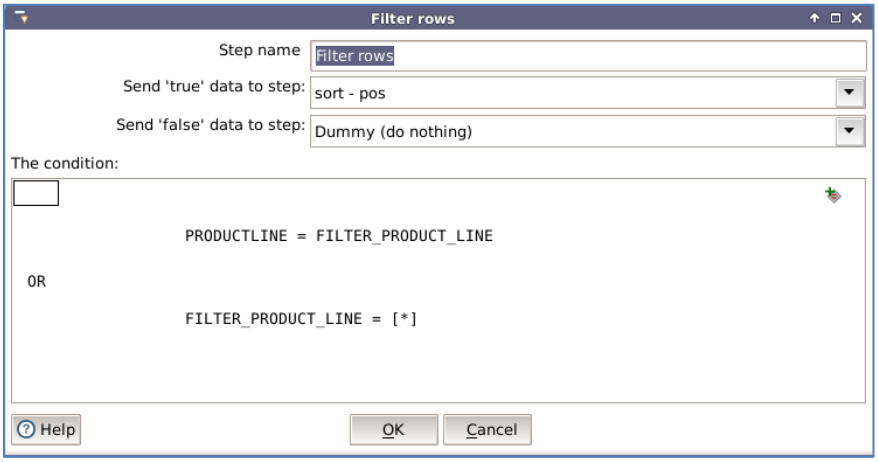
Click **OK**

12. Edit the **Get Variables** step right after the **HBase Input – POS** step to make sure the following configuration is in place:

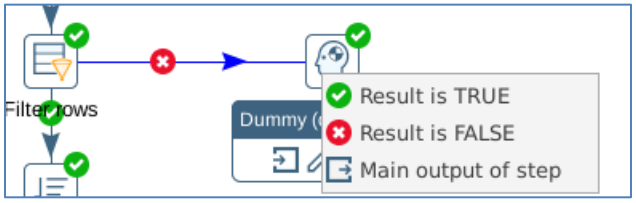


13. Let's take a look at the **Filter Rows** step right after the **Get Variables** step you just edited.

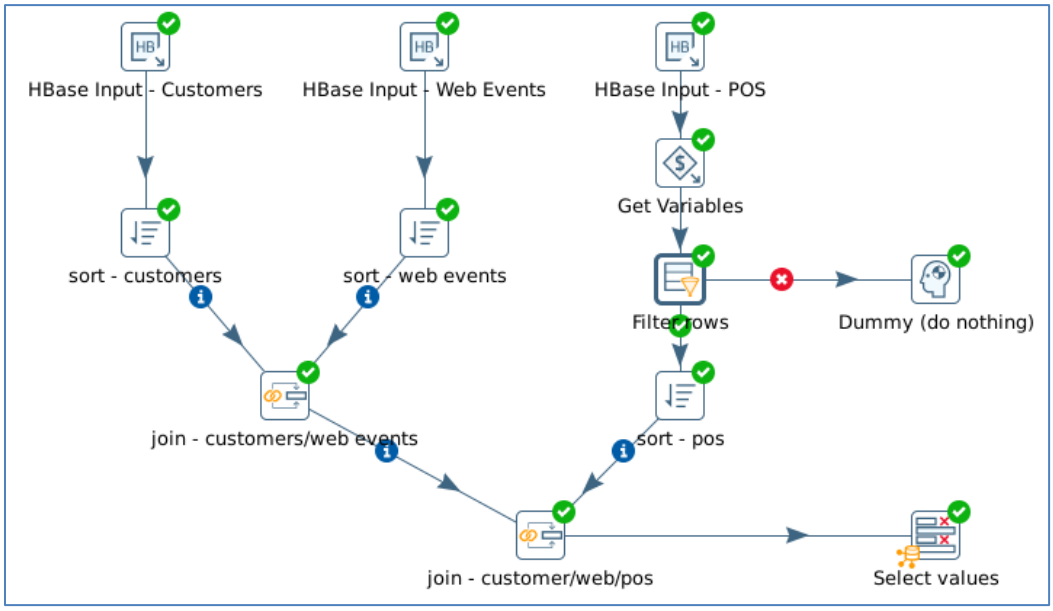
14. The properties should match the screenshot below:



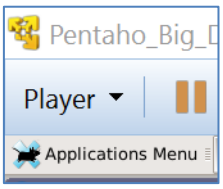
15. Verify that a **Dummy** step exists after the **Filter Rows** and the **hop** is set to `Result is False`.



16. Run the transformation. After running the transformation, it should look like this:

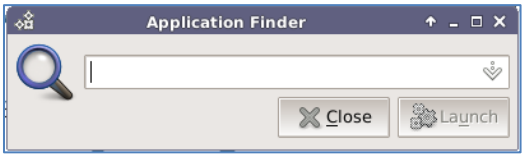


17. Now, using the Ubuntu Application Menu on the upper left side of the VM frame

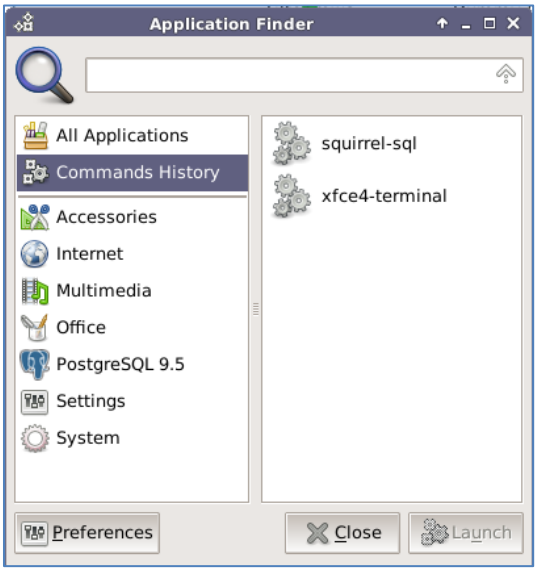


launch squirrel-sql (3rd party app):

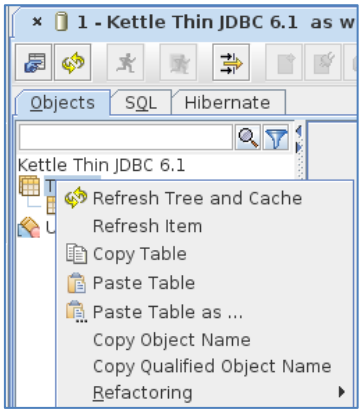
- Click on **Run Program**
- Click on the **Down Arrow** in the **Application Finder** pop-up



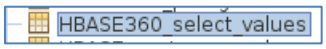
- Click on **Commands History** and select **squirrel-sql** on the right:



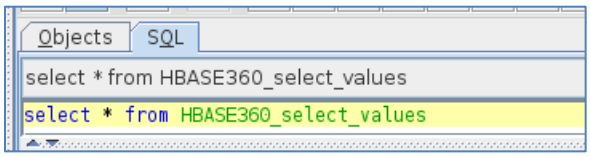
18. In the **Objects** tab, right-click on **Table** and select Refresh Tree and Cache:



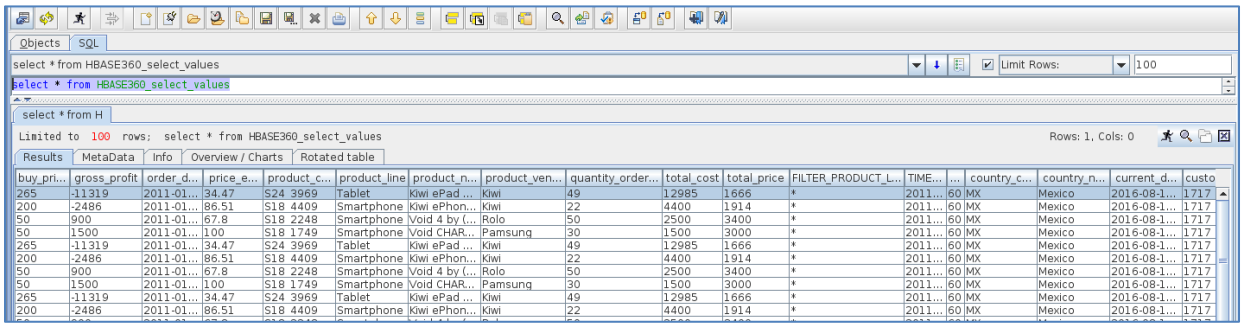
19. Now you should be able to see the **HBASE360_select_values** data service as a table:



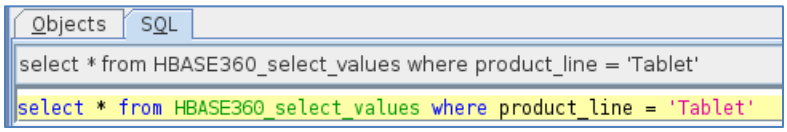
20. Now from the **SQL** tab execute ( ) the following query: `select * from HBASE360_select_values`:



21. You should see the following results:



22. Finally, execute a query with a WHERE clause as follows: `select * from HBASE360_select_values where product_line = 'Tablet'`:



23. You should see the following results: