

Big Data Integration Workshop

Contents

[VM Login and Setup 3](#_Toc448237690)

[Linux OS Login 3](#_Toc448237691)

[Cloudera Control 3](#_Toc448237692)

[Hue Login (web browser) 3](#_Toc448237693)

[Pentaho User Console Login (web browser) 3](#_Toc448237694)

[Start Pentaho and PostGreSQL (terminal window) 3](#_Toc448237695)

[DW Optimization Use Case 4](#_Toc448237696)

[What is it? 4](#_Toc448237697)

[Why do it? 4](#_Toc448237698)

[Value of Pentaho 4](#_Toc448237699)

[Pentaho components used in this workshop 5](#_Toc448237700)

[What you will accomplish in this workshop 5](#_Toc448237701)

[Part 1: Building PDI Transformations 6](#_Toc448237702)

[Part 2: Pentaho Visual Map Reduce (VMR) 16](#_Toc448237703)

[Streamlined Data Refinery Use Case 30](#_Toc448237704)

[What is it? 30](#_Toc448237705)

[Why do it? 30](#_Toc448237706)

[Value of Pentaho 30](#_Toc448237707)

[Pentaho components used in this workshop 31](#_Toc448237708)

[What you will accomplish in this workshop 31](#_Toc448237709)

[Part 1: Use PDI and Impala to process data in Hadoop 32](#_Toc448237710)

[Part 2: Explore data in PostgreSQL with Pentaho Analyzer 41](#_Toc448237711)

[HBase Customer 360 Use Case 46](#_Toc448237712)

[What is it? 46](#_Toc448237713)

[Why do it? 46](#_Toc448237714)

[Value of Pentaho 46](#_Toc448237715)

[Pentaho components used in this workshop 47](#_Toc448237716)

[What you will accomplish in this workshop 47](#_Toc448237717)

[Part 1: Use PDI to create a single view in HBase 48](#_Toc448237718)

[Create the HBase tables 48](#_Toc448237719)

[Part 2: Visualize Data with Analyzer 75](#_Toc448237720)

[MongoDB Customer 360 Use Case 80](#_Toc448237721)

[What is it? 80](#_Toc448237722)

[Why do it? 80](#_Toc448237723)

[Value of Pentaho 80](#_Toc448237724)

[Pentaho components used in this workshop 81](#_Toc448237725)

[What you will accomplish in this workshop 81](#_Toc448237726)

[Part 1: Use PDI to create a single view in MongoDB 82](#_Toc448237727)

[Use PDI to create a MongoDB customer data store 82](#_Toc448237728)

[Part 2: Visualize Data with Analyzer for MongoDB 94](#_Toc448237729)

[Part 3: Pentaho Report Designer (PRD) 98](#_Toc448237730)

VM Login and Setup

Linux OS Login

* See document on Desktop in Docs folder:
  + hds - BDI Workshop Credentials.pdf

-or-

* + bdiw - BDI Workshop Credentials.pdf

Cloudera Control

* Click icon in launcher on bottom screen 

Hue Login (web browser)

* Username: cloudera
* Password: cloudera

Pentaho User Console Login (web browser)

* See document on Desktop in Docs folder:
  + hds - BDI Workshop Credentials.pdf

-or-

* + bdiw - BDI Workshop Credentials.pdf

Start Pentaho and PostGreSQL (terminal window)

* cd /pentaho/current\_version
* ./ctlscript.sh start postgresql
* ./ctlscript.sh start baserver

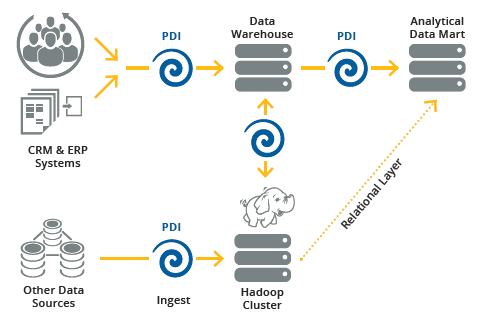
DW Optimization Use Case

What is it?

* Existing Data Warehouse implementations are struggling with rising data volumes, causing performance concerns and impacting a business’s ability to meet SLA’s. And at the same time, the costs involved in adding additional infrastructure has increased rapidly.
* The optimization solution is intended to relieve pressure on existing infrastructure by offloading less frequently used data, and improve performance by moving data transformation workloads to Hadoop.

Why do it?

* Reduce data warehouse costs by reducing the need for expensive DW infrastructure, and save management costs through the use of low latency data store.
* Meet SLAs to deliver data on time, & empower business users to meet goals on time.
* Satisfy compliance requirements by providing ready access to historical data (such as long-archivedfinancial transaction records).



Value of Pentaho

* Staff savings & productivity: Pentaho Data Integration for big data, along with Pentaho MapReduce means existing data warehouse developers can more easily move and process data between the data warehouse and Hadoop without coding.
* Time to value: Organizations can speed development time with Pentaho MapReduce by up to 15 times versus hand-coding and scripting.
* Faster job execution: Pentaho MapReduce runs in parallel in cluster, delivering faster performance when compared to other scripting tools.

Pentaho components used in this workshop

Pentaho Data Integration (PDI)

Pentaho Analyzer

What you will accomplish in this workshop

**Part 1** – Building PDI Transformations (3 exercises)

**Part 2** – Pentaho Visual MapReduce (3 exercises)

Part 1: Building PDI Transformations

Pentaho Data Integration (PDI) gives users a graphical user interface to build transformations and jobs that solve complex data integration challenges. PDI provides broad connectivity to a variety of sources including relational databases, text files, web services, XML, JSON and other big data technologies. The user interface reduces data integration complexity by eliminating the need to code data extractions, data transformations and data loads.

Part One of the Data Warehouse Optimization use case includes three exercises. The first exercise gives you experience building your first transformation to extract, transform and blend call detail records (CDR) with geographic data based on area code. The second exercise adds data filtering and enrichment steps and a final step to load the transformed CDR data to PostgreSQL. The third and final exercise is the fun part, as you use PDI visualize the CDR data resulting from your hard work in the first two exercises.

PDI Exercise 1: Create a transformation to blend CDR data and load to PostgreSQL

This first exercise steps you through the process of creating a transformation to transform and blend two CSV files containing call detail records and area code geographies. The first call detail records file, callrecords\_10years.csv, contains two fields, a call timestamp and the source phone number. The second file, areacodes.csv, contains a mapping of area codes to geographies in the United States. In this exercise you blend the CDR records with the geography information.

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.



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

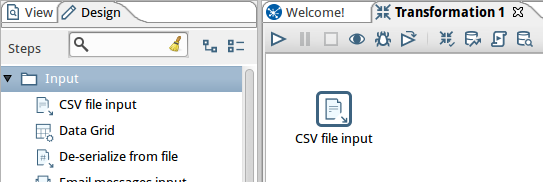
All open applications appear in the top left section of your screen. Once open, you will see Spoon as shown in the following screenshot.



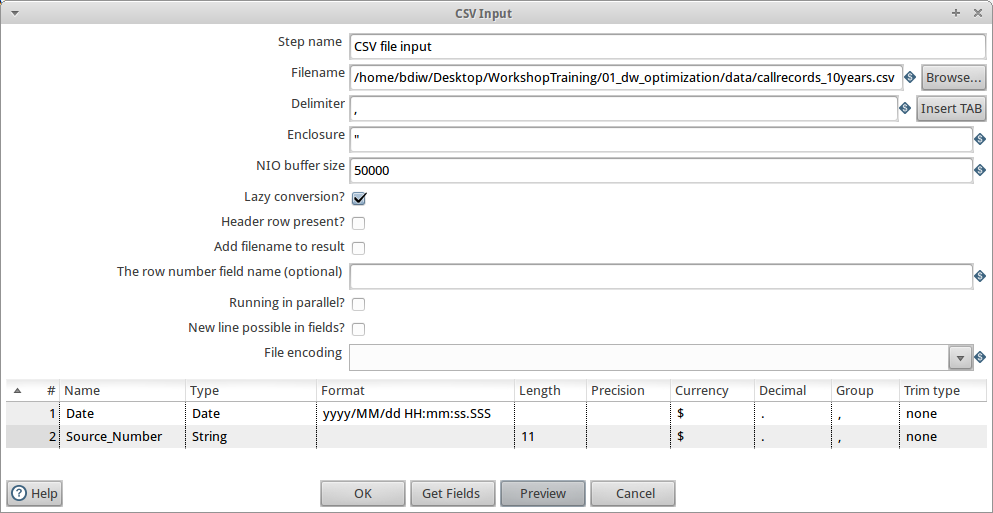
1. From the main menu choose File | New | Transformation

We need to access Call Data Records from within a flat file. The file format is a csv file, so you will access the data by configuring a CSV file input step.

1. From the Design tab on the left, expand the Input folder; then, select and drag CSV file input onto the **canvas**



1. Double-click on CSV file input to open its properties
2. Browse to the directory /pentaho/shared\_content/WorkshopTraining/ 01\_dw\_optimization/data/ and select callrecords\_10years.csv.
3. Uncheck the Header Row present? option
4. Click the Get Fields button at the bottom and enter 5,000 for the Sample Size.
5. Rename the first field, Field\_000, to Date
6. Rename the second field, Field\_001, to Source\_Number
7. Click the Preview button to preview the data and then click Close. Your CSV File Input dialog box should match the following image:



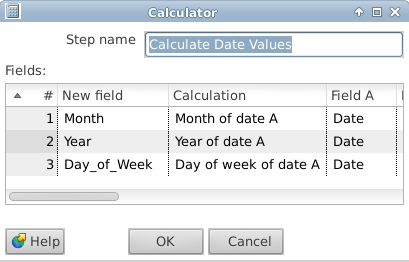
1. Click **OK** to return to the canvas.

Now that we have data, we will calculate the Year, Month, and Day of Week. This will allow the developer to filter the call records in a subsequent step to include only weekend calls.

1. Select and drag the Calculator step onto the canvas.
2. To draw a hop between two steps, shift-click the CSV file input step and while holding down your mouse key, drag a **hop** over to the Calculator step. When prompted, select Main output of step.

Going forward the workbook will simply instruct you to create a hop between two steps without a detailed description as in the previous step 14.

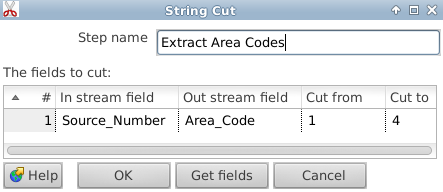
1. Double-click on the Calculator step to open its properties.
2. Change the Step name to Calculate Date Values
3. In the **Fields** section add Month**,** Year and Day\_of\_Week. The properties for these fields should match the following screenshot:



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

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

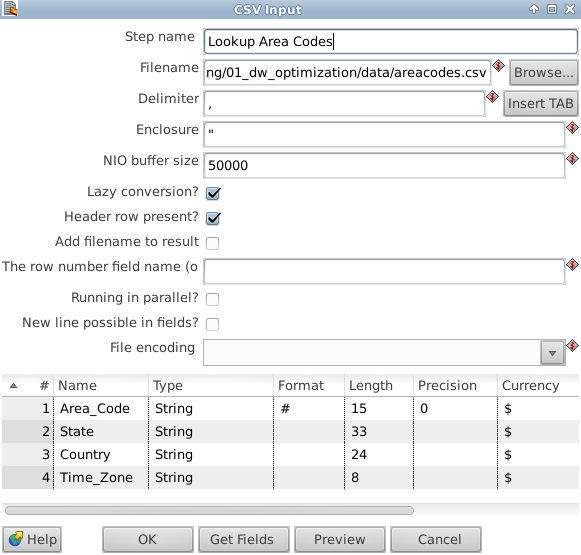
1. Select and drag the Strings Cut step onto the canvas.
2. Create a hop between the **Calculate Date Values** step and the Strings Cut step.
3. Double-click on the Strings Cut step to open its properties.
4. In the **Step name** field, type Extract Area Codes. In the **Fields to cut** section select 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:



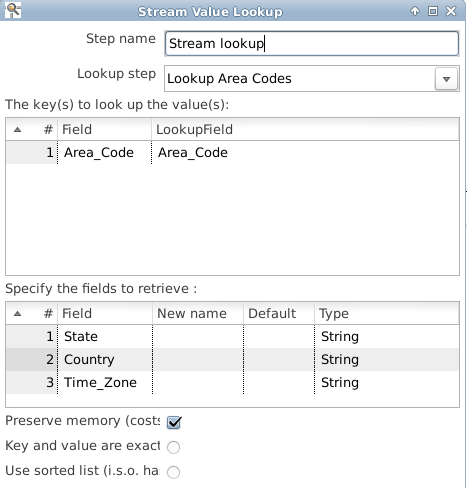
1. Click **OK** to return to the canvas.

Now that we know the Area Code, we will use a lookup file to map the Area Code to State, Country, and Time Zone.

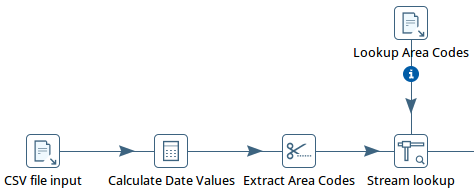
1. From the Design tab on the left, expand the Input folder; then, select and drag CSV file input onto the **canvas**
2. Double-click on CSV file input to open its properties
3. Change the Step name to Lookup Area Codes
4. Browse to the directory /pentaho/shared\_content/WorkshopTraining/ 01\_dw\_optimization/data/ and select areacodes.csv.
5. Click the Get Fields button at the bottom and enter 5,000 for the Sample Size.
6. For the Area\_Code field, change the Type from Integer to String.
7. Click the Preview button to preview the data and then click Close to return to the CSV File Input step configuration. Your CSV File Input dialog box should match the following image:



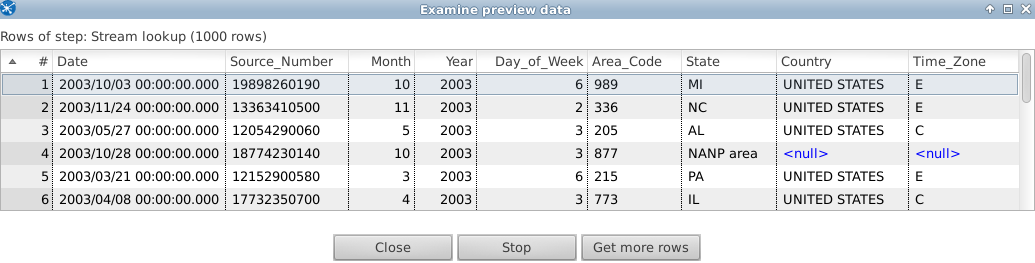
1. Click **OK** to return to the canvas.
2. Expand the Lookup folder; then, select and drag Stream Lookup onto the canvas.
3. Create a hop from the **Extract Area Codes** step to the Stream Lookup step. When prompted choose, Main Output of Step, to complete the hop connection.
4. Create a hop from the **Lookup Area Codes** step to the Stream Lookup step and choose Main Output of Step, to complete the hop connection.
5. Double-click on the Stream Lookup step to open its properties.
6. In the **Lookup** step select Lookup Areas Codes.
7. In the **Key(s) to Lookup Value(s)** section select Area\_Code in the **Field** column and select Area\_Code as the **LookupField** column.
8. Click the **Get Lookup Fields** button to populate the fields to retrieve section at the bottom.
9. Highlight and delete Area\_Code from the fields to retrieve section. Your Stream Lookup dialog box should now look like this:



1. Click **OK** to return to the canvas. Your transformation should match the following screenshot:



1. From the **File** menu, choose **Save**.
2. In the Name field, specify CDR–RDBMS, and save to: /pentaho/shared\_content/WorkshopTraining/student\_files/01\_dw\_optimization.
3. To test your transformation, right-click on the Stream lookup step and choose Preview from the list.
4. Click the Quick Launch button to open the Examine preview data dialog. You should see records similar to the image below:



1. Click the Stop button to close the preview data box.

PDI Exercise 2: Enhance the PDI transformation to filter and add call count

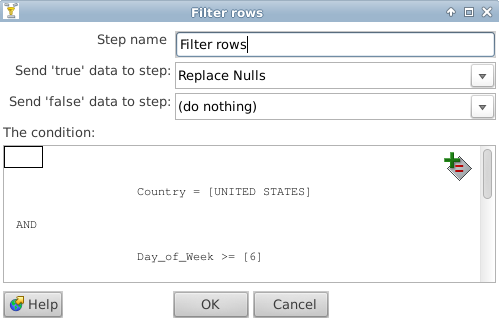
This second exercise steps you through adding additional steps to your transformation to filter for U.S. only calls, replace null values, and to add a call count measure field. The final step of this exercise loads the transformed data to PostgreSQL.

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.

1. Expand the Flow folder; then, select and drag Filter Rows onto the canvas.
2. Create a hop between the **Stream Lookup** step and the Filter Rows step.
3. Double-click on the Filter Rows step to open its properties.
4. Rename this step to U.S.-only calls.
5. Click **OK** to return to the canvas.
6. From the Flow folder, select and drag Dummy (do nothing) onto the canvas above the U.S.-only calls step.
7. Create a hop from the **U.S.-only calls** step to the Dummy (do nothing) step. Select Result is False.
8. Double click the **U.S.-only calls** step to open its properties.
9. Under **The condition** section select the <field> on the left. From the pop-up window select Country and then click OK.
10. Click in the middle box and select “=” as the function and then click OK.
11. Click the bottom right <value> box and type UNITED STATES as the Value.

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

1. Click the Add condition icon on the right side to add a filter for Friday and Saturday calls.
2. Click the null = [ ] area to add the second condition.
3. Select the <field> on the left. From the pop-up window select Day\_of\_Week and then click OK.
4. Click in the middle box and select >= as the function.
5. Click the bottom right <value> box and type 6 as the Value.



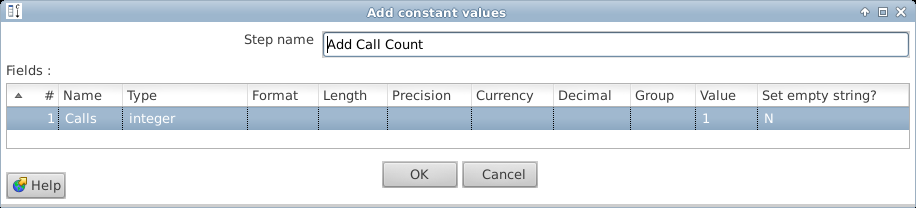
1. Click **OK** to return to the canvas.

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

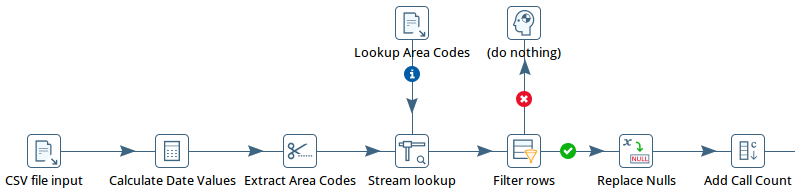
1. Expand the Utility folder and drag Null if… step onto the canvas.
2. Create a hop between the **U.S.-only calls** step and the Null if… step. Select Result is true to complete the hop.
3. Double click the Null if… step to open its properties.
4. Rename this step to Replace Nulls
5. Under the **Fields** 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.

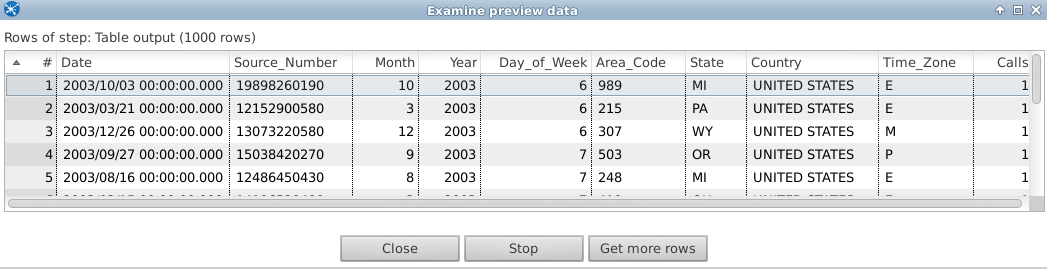
1. Expand the Transform folder; then, select and drag the Add Constants step onto the canvas.
2. Create a hop between the **Replace Nulls** step and the Add Constants step.
3. Double click the Add Constants step to open its properties.
4. In the **Step Name** field, type Add Call Count.
5. Add a Field named Calls as a Type Integer with a Value of 1 as illustrated here:



1. Click **OK** to return to the canvas. Your transformation should match the following image:



1. From the **File** menu, choose **Save** to save your work.
2. To test your transformation, right-click on the Add Call Count step and choose Preview from the list.
3. Click the Quick Launch button to open the Examine preview data dialog. You should see records similar to the image below:

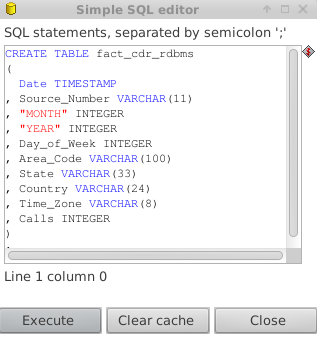


1. Click the Stop button to close the preview data box.

Our data is ready to be placed into a database table for reporting. We need to configure a target table within a PostgreSQL database for the enriched call record data.

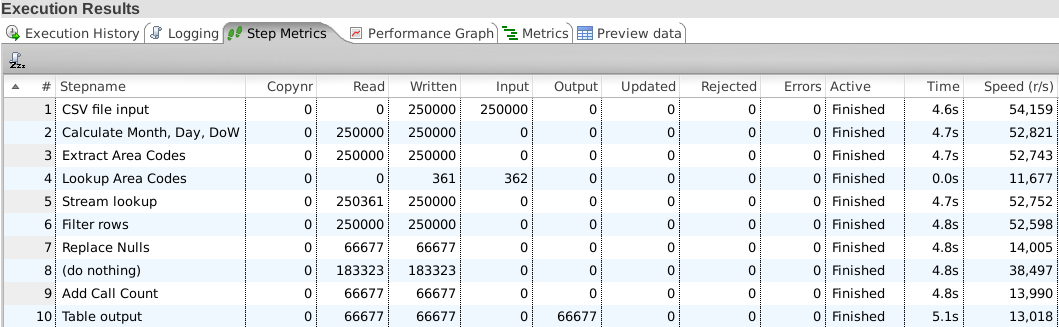
1. Expand the Output folder; then, select and drag Table output onto the canvas.
2. Create a hop between the **Add Call Count** step and the Table output step.
3. Double-click the Table output step to open its properties.
4. For Connection select workshop\_postgres
5. For Target table enter fact\_cdr\_rdbms
6. Check Truncate table and then click the SQL button at the bottom to generate a create table SQL statement matching the following image:

If the fact\_cdr\_rdbms table already exist then your SQL will not match the SQL shown in the following image, and you can close this dialog box without executing any code.



1. Click the Execute button to create the fact\_cdr\_rdbms table in PostgreSQL.
2. The Results of the SQL statements dialog will appear. Click OK to return to the Table output configuration dialog box.
3. From the **File** menu, choose **Save**.
4. To load PostgreSQL, execute the transformation by choosing Action 🡪 Run from the main menu or by clicking the run icon .
5. The Execute a transformation dialog will appear. Click the Launch button at the bottom.

A green check will appear on the Table output step when the transformation finishes without errors. The Step Metrics tab shows the 250,000 records input from the CSV file input step and 66,677 records output to PostgreSQL.



PDI Exercise 3: Create a data table and area chart of the CDR data.

The Visualize PDI perspective allows developers to instantly analyze and visualize data from the output of a data transformation. In this exercise, you use the visualize perspective to visualize the newly transformed CDR data by creating a conditionally-formatted table with a custom calculation for CDR annual growth % by year. Next you create and an area chart showing call volumes by year and a drop in call volumes during the 2008 recession.

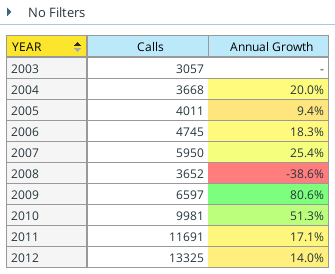
1. Right-click on the **Table output** step and select Visualize 🡪 Analyzer to launch into the Visualize perspective.
2. Under the Measure drop down in the Available fields section on the left, drag Calls to the **Measures** drop zone in the Layout section.



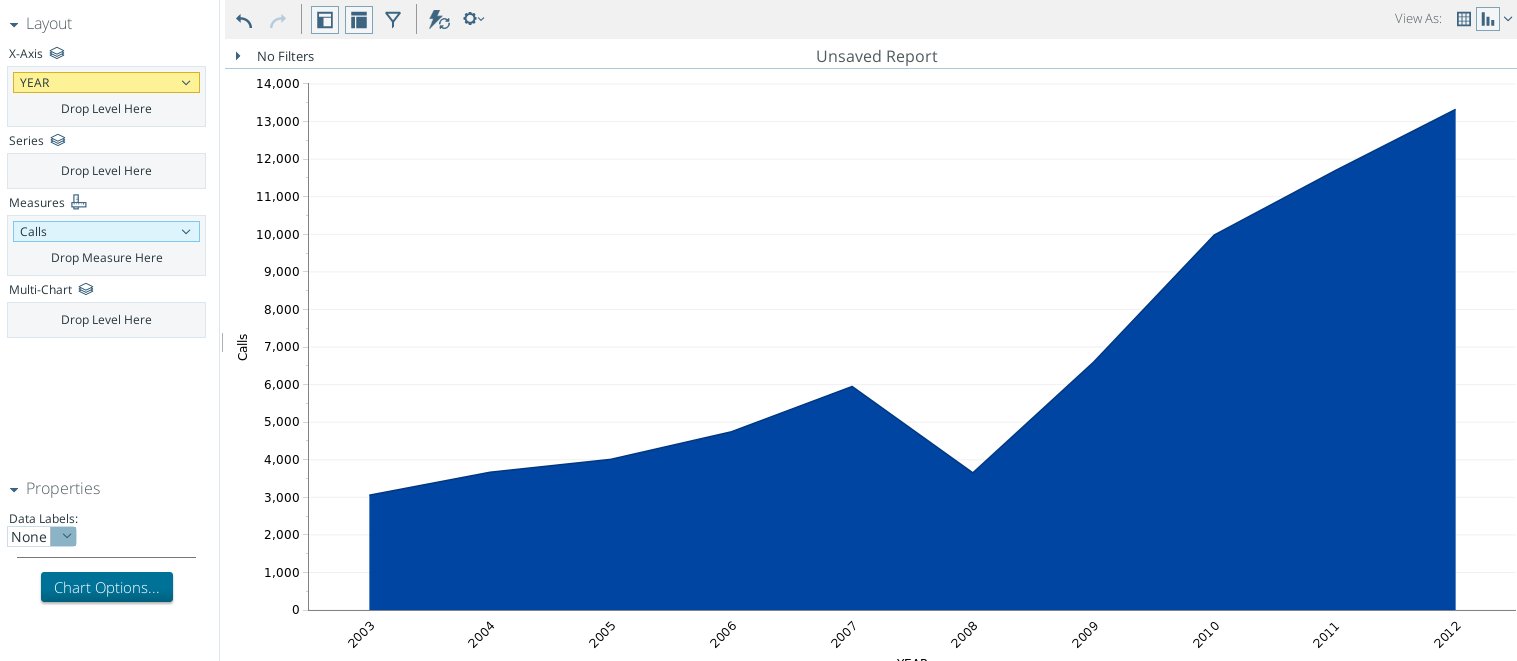
1. Under the Level drop down in the Available fields section on the left, drag Year to the **Rows** drop zone in the Layout section.



1. Right-click on the column header for Calls and select **User Defined Measure 🡪 Trend Measure…**
2. In the **Name** field, type Annual Growth.
3. In the **Period Type** field, select Year.
4. In the **Number of periods** field, type 1 (should be the default).
5. In the **Show trend as** field, select% of change from previous period.
6. In the **Decimal Places** field, select 1.
7. Click **OK** to return to the report.
8. Right-click on the column header for Annual Growth and select **Conditional Formatting | Color Scale: Green-Yellow-Red**.
9. Your analysis should now match the following image:



1. Click on the **Switch to Chart** button on the upper right hand side of the canvas () and select **Area**.
2. Click the Annual Growth measure in Layout section and choose **Hide from Chart**. The resulting area chart should match the following screenshot:



1. Click the Data Integration perspective in the top right area of PDI to return to your transformation.



1. From the **File** menu, choose **Save**.
2. From the **File** menu, choose **Save As** to create a copy of this transformation to be used in the next exercise.
3. In the Name field, specify CDR–Mapper.ktr, and save to: /pentaho/shared\_content/WorkshopTraining/student\_files/01\_dw\_optimization.

Congratulations on completing a PDI transformation to blend, enrich and load CDR data to PostgreSQL for analysis! The next section introduces the concept of Visual MapReduce for running transformations in a Hadoop cluster to leverage the powerful parallel processing capabilities of Hadoop.

Part 2: Pentaho Visual Map Reduce (VMR)

Part Two of the Data Warehouse Optimization use case has you modify your first transformation to create a Visual MapReduce transformation that can be run inside the Hadoop cluster; leveraging the processing power of Hadoop. You convert this transformation to a mapper transformation and use Pentaho to run this as a native mapper job inside of Hadoop. This is powerful because it takes the transformation logic to where the data resides at each data node within the Hadoop cluster. The Pentaho engine and mapper transformation are automatically copied to the data nodes and run as a native MapReduce application.

Part Two includes three exercises. In the first exercise you create a job to load the CDR data to HDFS. secondThe third and final exercise extends the job to add a Visual MapReduce step to execute the mapper transformation created in exercise one against the CDR data loaded to HDFS in exercise two.

MapReduce Exercise 1: Create a PDI Job to load CDR data to Hadoop

This first exercise steps you through creating a job that executes a single job step. You need the CDR data in HDFS to be able to process it with your mapper transformation. This job step loads the raw CDR data to HDFS so that it can be processed in a later exercise.

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

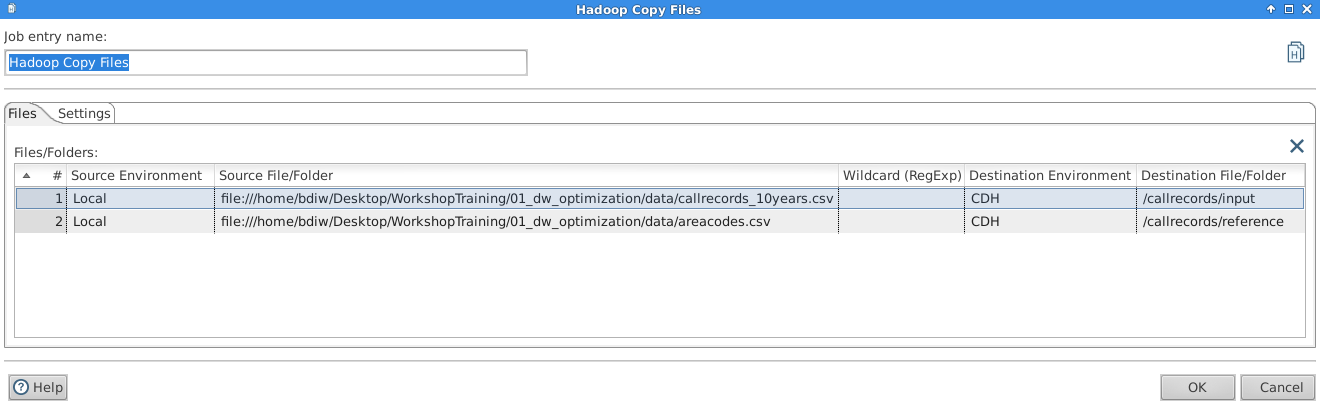


To process data with our mapper transformation inside Hadoop, we need to define the data file that will be copied, and where it will be placed inside HDFS.

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

file:///pentaho/shared\_content/WorkshopTraining/01\_dw\_optimization/data/callrecords\_10years.csv

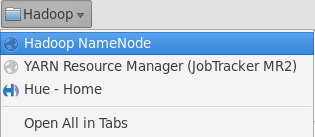
1. Click the OK button to return to the Copy Files dialog box
2. Select the cell beneath the “Destination Environment” header on the file tab and select “CDH” from the drop down.
3. Select the cell beneath the “Destination File/Folder” step and enter /callrecords/input
4. Repeat steps **8-13** to copy the “Source File/Folder” file:///pentaho/shared\_content/WorkshopTraining/01\_dw\_optimization/data/areacodes.csv to the “Destination File/Folder” /callrecords/reference
5. Your Hadoop Copy Files dialog box should match the following image:



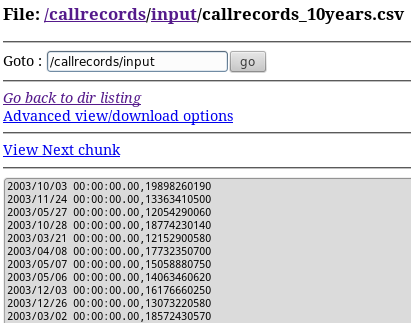
1. Click OK to return to the canvas.
2. From the **File** menu, choose **Save**.
3. In the Name field, specify CDR-CopyFiles.kjb and save to the following location: /pentaho/shared\_content/WorkshopTraining/student\_files/01\_dw\_optimization.
4. From the **Action** menu, choose **Run** and then click Launch. On the Job metrics tab located in the bottom section of Spoon, you should see the job progress.
5. To view the newly copied CDR data in Hadoop, launch Firefox by single-clicking the Firefox icon at the bottom of your screen.



1. From the Firefox bookmarks bar click Hadoop NameNode.



1. Click the following links Browse the file system 🡪 callrecords 🡪 input 🡪 callrecords\_10years.csv
2. Click the following links Browse the file system 🡪 callrecords 🡪 reference 🡪 areacodes.csv
3. If your job executes successfully you will see CDR records as shown in following screenshot:



The file is now ready to be processed within Hadoop and by the mapper transformation previously created in MapReduce Exercise 1.

MapReduce Exercise 2: Create a mapper transformation to process CDR data

This second exercise steps you through modifying your existing transformation to add Map Reduce Input and Map Reduce Output steps and change the Stream Lookup step to query geographic data from HDFS instead of a local CSV file. You will also add additional steps needed to complete the mapper transformation.

1. Make sure that you are working with the newly created CDR-Mapper.ktr transformation created in the previous exercise.
2. Delete the following *three* steps: CSV file input step, Lookup Area Codes, and Table Output.

To convert this transformation to a mapper transformation and use Pentaho to run this as a native mapper job inside of Hadoop, we need to define a MapReduce Input and MapReduce Output. This will push the transformation logic to where the data resides at each data node within the Hadoop cluster.

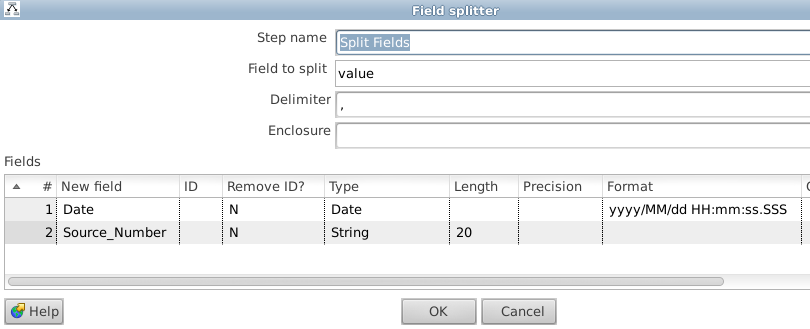
1. Expand the Big Data folder and drag the MapReduce Input and MapReduce Output steps onto the canvas. Place the MapReduce Input step at the beginning of your transformation and the MapReduce Output step at the end of your transformation.
2. Double-click on MapReduce Input step to open its properties.

 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.

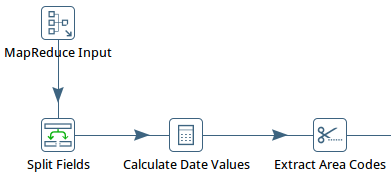
1. For both the Key field and the **Value field** select String for Type.
2. Click **OK** to return to the canvas.

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.

1. Expand the Transform folder and drag Split Fields onto the canvas.
2. Create a hop from the **MapReduce Input** step to the **Split Fields** step.
3. Create a hop from the **Split Fields** step to the **Calculate Date Values** step.
4. Double-click on the Split Fields step select value for Field to split and leave “**,**” as the **Delimiter**.
5. In the **Fields** section add Date and Source\_Number.
6. Complete the Type, Length and Format columns for these fields to match the following screenshot:

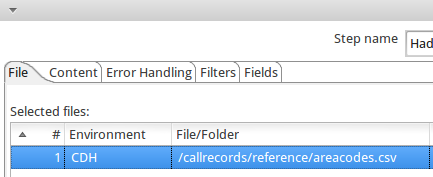


1. Click **OK** to return to the canvas. The first part of your transformation should now match the following image:

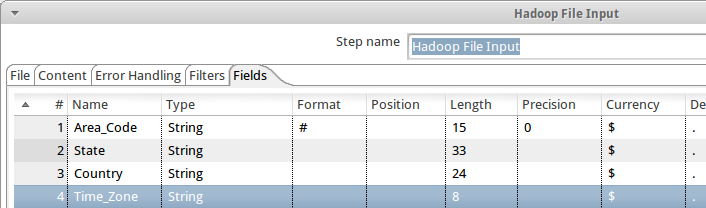


Previously in Part 1, Exercise 1, we used a lookup file on the Pentaho server to map the Area Code to State, Country, and Time Zone. Now we will use a lookup file that is same in data and structure, but is placed on the Hadoop file system HDFS.

1. Expand the Big Data folder and select and drag the Hadoop File Input step onto the canvas directly above the Stream Lookup step.
2. Create a hop from the **Hadoop File Input** step to the Stream Lookup step.
3. Double-click on the Hadoop File Input step to open its properties.
4. Select the cell beneath the “Environment” header on the file tab and select “CDH” from the drop down.
5. Select the cell beneath the “File/Folder” step and select the  button.
6. Browse to the following HDFS directory, /callrecords/reference and select the file areacodes.csv and then click the OK button.
7. Your **Hadoop File Input** dialog box should match the following image:



1. Click on the Content tab.
2. Change the Separator field to a comma ,.
3. Click on the Fields tab.
4. Click the Get Fields button and sample 5000 records.
5. Change the Area\_Code Type from Integer to String.
6. The dialog box should look like this:



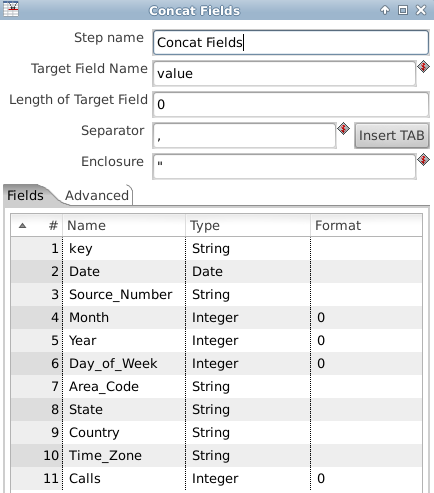
1. Click **OK** to return to the canvas.

Previously in Part 1, Exercise 1, the **Stream Lookup** step used CSV input from a step named ‘Lookup Area Codes’. This should be changed to reference the new **Hadoop File Input** step we left named as ‘Hadoop File Input’.

1. Double click the Stream Lookup step and for the Lookup step drop-down, select Hadoop File Input.
2. Click **OK** to return to the canvas.

This mapper transformation has generated several new fields. The fields need to be combined into a new target field and delimited. The target field, which we will creatively name ‘value’, will ultimately be used as the **Value** in the Key/Value pair for the MapReduce output.

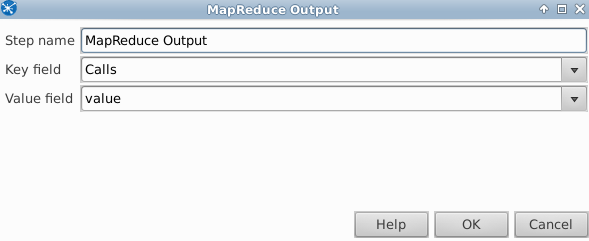
1. Expand the Transform folder and select and drag the Concat Fields step onto the canvas at the end of your transformation.
2. Create a hop from the **Add Call Count** step to the Concat Fields step.
3. Double click the Concat Fields step to open its properties.
4. In the **Target Field Name** field, type value and change the **Separator** field to a comma ,.
5. Click on the **Get Fields** button.
6. Click on the Minimal width button to remove excessive field lengths.
7. Confirm the **Fields** properties match the following and click OK to return to the canvas.



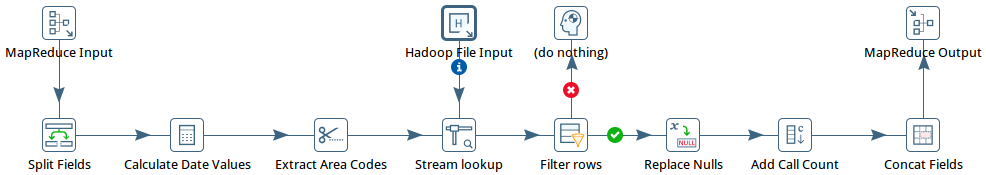
1. Create a hop between the Concat Fields step and the **MapReduce Output** step.

The output of MapReduce will be a key/value pair. We can define the output key for any field for which we want to aggregate with a Reducer. The output value will be the contents of the target field we defined in the **Concat Fields** step (Date, Source Number, Area Code, Month, Year, Calls, etc).

1. Double click the MapReduce Output step to open its properties.
2. Select Calls for the Key field and select value for the **Value field**. Your **MapReduce Output** step should match the following image:



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



1. From the **File** menu, choose **Save**.
2. Do not run this transformation. This transformation will be executed by a job you will created in the last exercise.

MapReduce 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 CDR\_CopyFiles.kjb Job you created in the previous exercise.
2. From the **File** menu, choose **Save As**.
3. In the Name field, specify “CDR-MapReduce-BuildJob” and save to the following location: /pentaho/shared\_content/WorkshopTraining/student\_files/01\_dw\_optimization.

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 Exercise 1, and define parameters such as data inputs and Hadoop cluster connection information.

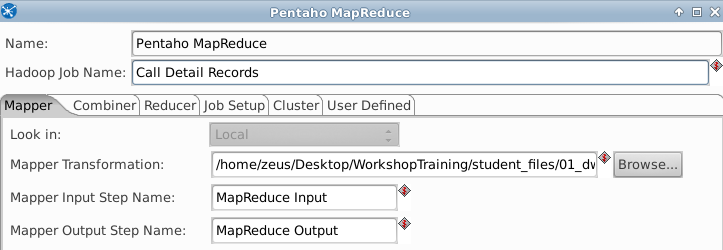
1. From the **Design** tab, expand the **Big Data** folder, then select and drag the **Pentaho MapReduce** step onto the canvas.
2. Select the **Pentaho MapReduce** step and drag it above the connecting arrow between the **Hadoop Copy Files** step and the **Success** step until the arrow becomes highlighted. Once highlighted, drop the **Pentaho MapReduce** step (release the mouse button).
3. 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:



1. Open the **Pentaho MapReduce** step.
2. In the **Hadoop Job Name** field, specify a name, such as Call Detail Records.
3. 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/01\_dw\_optimization CDR–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.

1. In the **Mapper Input Step Name** field, type MapReduce Input.
2. In the **Mapper Output Step Name** field, type MapReduce Output.
3. Your Pentaho MapReduce Step should match the following image:



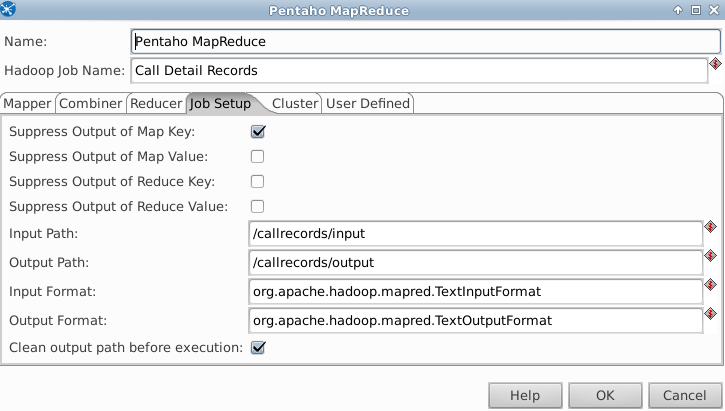
1. Click on the **Job Setup** tab.

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.

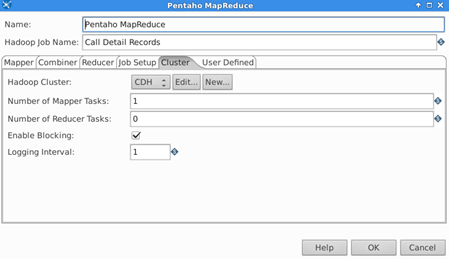
1. Click the check boxes for **Suppress Output of Map Key** and **Clean output path before execution**.

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

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



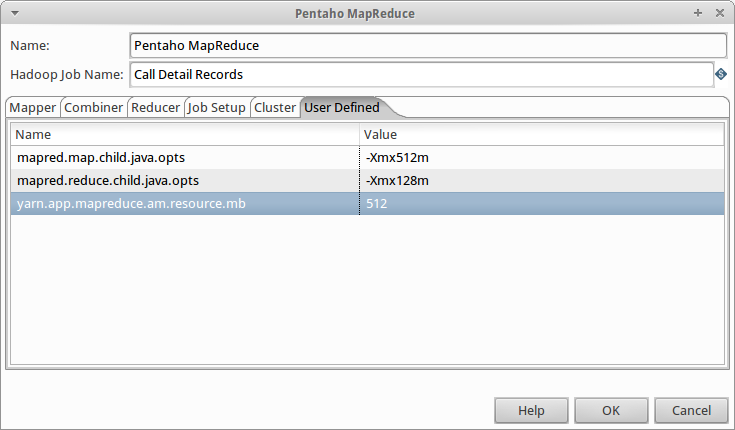
1. Next, click the **Cluster** tab to configure your connection to the Hadoop cluster.
2. In the Hadoop Cluster drop down select “CDH”
3. In the **Number of Mapper Tasks** field, specify 1.
4. In the Number of Reducer Tasks field, specify 0.
5. Check the box to **Enable Blocking**.
6. Specify a **Logging Interval** of 1 seconds.
7. The **Cluster** tab should match the following image:



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

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

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



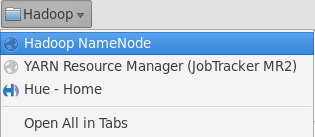
1. Click **OK** to return to the canvas.
2. From the **File** menu, choose **Save**.
3. From the **Action** menu, choose **Run** and then click Launch.
4. Click the Logging tab located in the bottom section of Spoon to see the Mapper job progress.

**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.

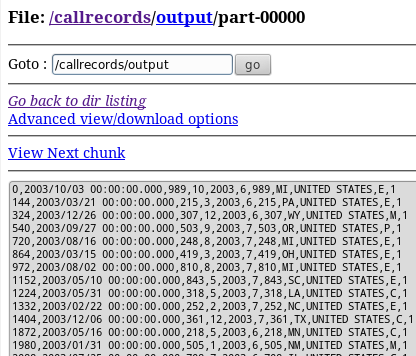
1. When the job successfully completes, you will see a green checkmark on the Success step.



1. To view the newly processed CDR data in Hadoop, expand Firefox from the top left open applications section if Firefox is already open; otherwise launch it from the bottom launch menu icon.
2. From the Firefox bookmarks bar click Hadoop NameNode.

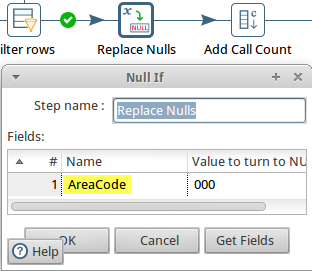


1. Click the following links Browse the filesystem 🡪 callrecords 🡪 ouput 🡪 part-0000
2. If your job executes successfully you will see processed and blended CDR records as shown in following screenshot:

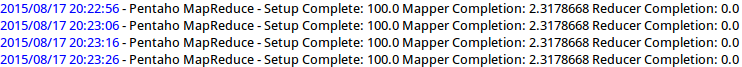


(OPTIONAL) MapReduce Exercise 4: Debug a MapReduce job error

If your MapReduce job failed in the previous exercise, you can debug the map task by using the YARN Resource Tracker utility bookmarked in your Firefox browser. This exercise shows you how to debug a spelling issue in the mapper transformation. If you rename the Area\_Code field in the Replace Nulls step to AreaCode with no underscore, the mapper transformation will cause your job to fail. You can try it or simply read thru these exercise steps to learn.



This spelling error will cause your job to fail because the AreaCode field does exist in the transformation stream. During execution the Execution Results log record starts repeating the % complete value as illustrated in the following screenshot.

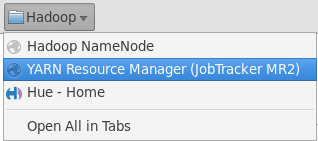


In the previous screenshot you can see the Mapper Completion value of 2.3178668 is repeating. This indicates a problem that should be investigated in the YARN Resource Manager utility. The following steps show how the Cloudera YARN Resource Manager is used to debug the issue.

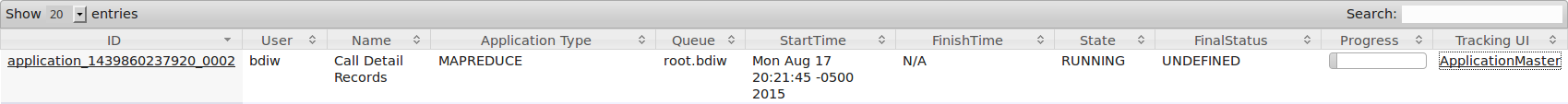
1. Launch Firefox by single-clicking the Firefox icon at the bottom of your screen.



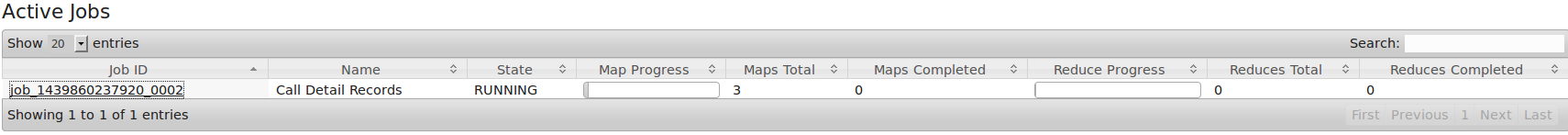
1. From the Firefox bookmarks bar click YARN Resource Manager.



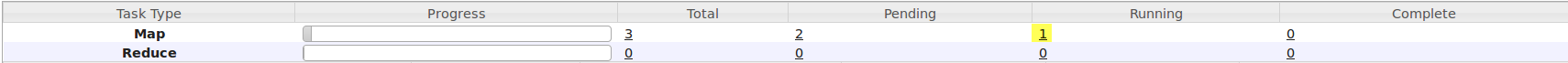
1. The YARN Resource Manager home page shows your current running application, Call Detail Records, is stuck at 2.3 % complete.



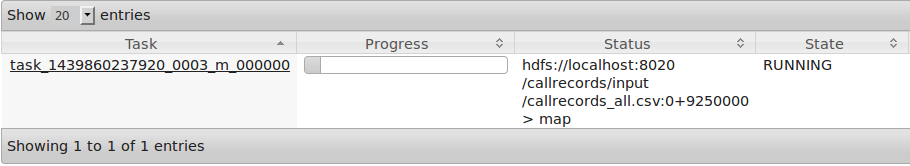
1. Click the ApplicationMaster hyperlink to drill into the application. The image below shows that the job is stuck in the map phase. Click the job id link to show map and reduce tasks.



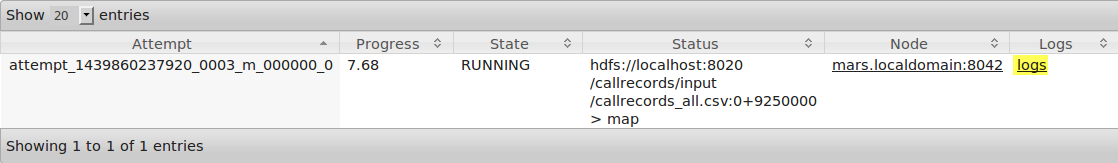
1. Click the map hyperlink to drill into the individual map task details.



1. Click the Task hyperlink to drill into the tasks attempts. As the process continues to run, you might see more than one attempt to run the task.



1. The far right column in the resulting screen provides links for the Logs. Click on the log link for the current task attempt.



1. Click the stderr line in the resulting screen.



1. The area code spelling issue is highlighted in the stderr logs screenshot below.



1. To fix the issue return to Pentaho and stop the CDR-MapReduce-BuildJob job by clicking the green stop icon.



1. Return to your CDR–Mapper.ktr transformation and fix the area code spelling in the Replace Nulls step.

Congratulations, you have just completed the Data Warehouse Optimization Use Case! This use case is designed to show a development pattern for processing historical CDR data in Hadoop. The next use case, Streamlined Data Refinery, extends the use of Hadoop as a central processing hub for new sources of data that do not necessarily reside in an existing data warehouse.

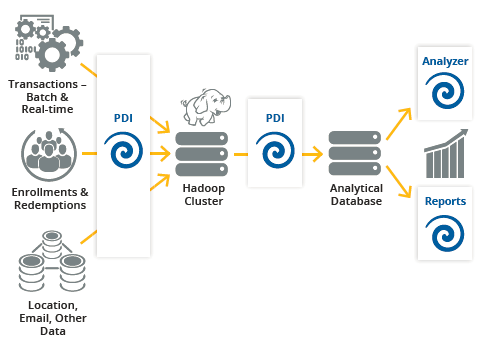
Streamlined Data Refinery Use Case

What is it?

* Streamlined Data Refinery (SDR) is an agile data integration process that includes transformation steps and tools to enable data processing on a low latency data store, such as Hadoop.
* Traditional extract, transform and load (ETL) processes can no longer cope with the variety and volume of data that need to be incorporated in analytics because they are designed for top-down development processes.
* SDR includes self-service analytics along with an analytic database for high speed data querying and visualization.

Why do it?

* Provide business users insight into all data in a timely fashion, including diverse sources at high volume.
* Engineer new data sets for predictive analytics more quickly thanks to rapid ingestion and powerful processing.
* Scale ETL and data management cost savings by utilizing the right technology for the most appropriate purpose.
* Logical next step from Data Warehouse Optimization use case.



Value of Pentaho

* **Data Collaboration**: Using Pentaho Data Integration (PDI), a data developer and business analyst can collaborate in real-time on the creation of new data sets using one product for both data engineering and data visualization.
* **Staff savings & productivity**: Pentaho provides a simple mechanism for creating visual map reduce jobs along with Big Data integration which means data developers can move and process data between Hadoop and a variety of other sources and systems easily and efficiently.
* **Complete Solution**: Broad data integration to accommodate existing architectures plus a powerful array of self-service analytics and visualizations for all end users, whether they be business users, analysts, and/or data scientists!
* **Time to value**: Significantly reduce development time for MapReduce by 15x over typical hand coding and scripting processes.

Pentaho components used in this workshop

Pentaho Data Integration (PDI)

Pentaho Analyzer

What you will accomplish in this workshop

* Part 1 – Use PDI and Impala to process data in Hadoop (2 exercises)
* Part 2 – Explore data in PostgreSQL with Pentaho Analyzer (2 exercises)

Part 1: Use PDI and Impala to process data in Hadoop

Now that you have moved the CDR data to HDFS, the next step is to look for other sources of data to take advantage of Hadoop’s processing power. Geo-location data feeds from smart phones have traditionally been expensive and difficult to process, but with Hadoop this data can be processed using a streamlined data refinery architecture. Devices register their location periodically on the wireless network. Now instead of estimating the caller's location by an area code lookup, we can more accurately track their location by their geo-location at the time a phone call was made.

Part 1 of the SDR Use Case includes exercises showcasing how to use Impala to join data from multiple tables and load the blended data to both Hadoop and PostgreSQL. The first exercise steps you through the process of creating a PDI job to execute a transformation for loading geo-location data to HDFS and creating an Impala table. The second exercise takes advantage of Impala queries to join two tables into a combined data set. This combined set is saved into Impala first and then loaded to a PostgreSQL database to enable high performance OLAP analysis on the combined data. The combined data is stored in two places to illustrate the flexibility of PDI to orchestrate data into multiple target locations.

PDI Exercise 1: Create a job to load data to HDFS and Impala

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\_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.

1. Expand the File management folder and drag the **Create a folder** step onto the canvas.
2. Create a hop between the **START** and **Create a folder** steps.
3. Double click the **Create a folder** step to edit its properties. In the **Folder name** field type hdfs://quickstart.cloudera:8020/demo/data\_refinery.
4. Uncheck the **Fail if folder exists** box.
5. Click **OK** to return to the canvas.

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

1. Expand the **Scripting** folder and drag the **Shell** step onto the canvas to the right of **Create a folder** and double click to open.
2. In **Job entry name** enter **Set HDFS Directory Permissions.**
3. Check the box next to **Insert script.**
4. In **Working directory**, enter /tmp
5. Switch to the **Script** tab and copy/paste the following 2 lines:

#!/bin/sh

docker exec $(docker ps -q) sudo -u hdfs hadoop fs -chmod -R 777 /demo/data\_refinery

1. Click **OK** to return to the canvas.
2. Create a hop between the **Create a folder** and **Set HDFS Directory Permissions** steps
3. 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.

1. Double Click the **Transformation** step to edit its properties.
2. In the Name of job entry box, type Load Geolocation Data.
3. 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\_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.

1. 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’.

1. Expand the Conditions folder and drag the **Table Exists** step onto the canvas.
2. Create a hop between the **Load Geolocation Data** and **Table Exists** steps.
3. Double Click the **Tables Exists** step to edit its properties.
4. In the **Job entry name** field type Does Impala table exist?
5. In the **Connection** field select Impala.

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

1. In the **Table Name** field type call\_detail\_geo.
2. 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.

1. Expand the **Scripting** folder and drag a **SQL** step to the right of the Does Impala table exist? step.
2. 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.

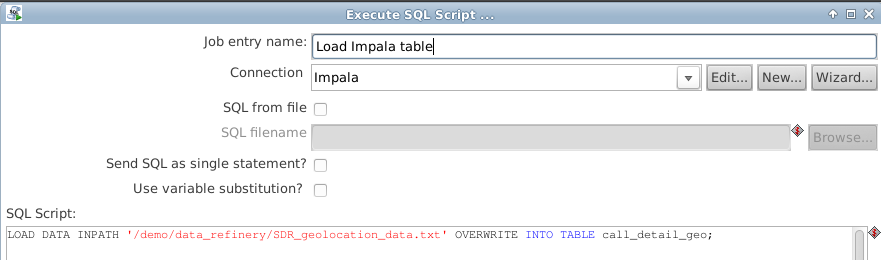
1. Drag a second SQL step from the Scripting folder and place it below the Does Impala table exist? step.
2. Rename this SQL step to Create Impala Table
3. Create a hop from the Does Impala table exist? step to the **Load Impala table** step to the right. This hop should be green in color indicating this path will be taken if the previous step returns a true evaluation.

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

1. 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.
2. Create a hop from the Create Impala table step to the **Load Impala table** step. The hops should match the following image:



1. Double click the **Load Impala table** step to edit its properties.
2. In the **Connection** field select Impala.
3. In the **SQL Script** section type the following: LOAD DATA INPATH '/demo/data\_refinery/SDR\_geolocation\_data.txt' OVERWRITE INTO TABLE call\_detail\_geo**;**
4. Your SQL script step should now look like this:



1. Click **OK** to return to the canvas.
2. Double click the **Create Impala table** step to edit its properties.
3. In the **Connection** field select Impala.
4. In the **SQL Script** section type or copy/paste the following code which can be found in the bdi\_workshop\_code.txt file on your machine:

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

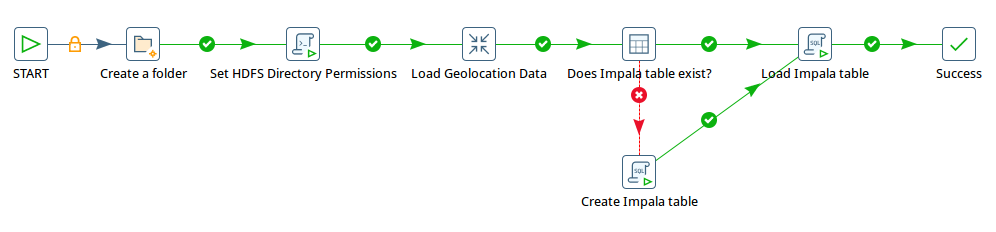
)

row format delimited

fields terminated by '|'

STORED AS TEXTFILE;

1. Click **OK** to return to the canvas.
2. Create a hop from the Load Impala table step to the Success step.
3. Your PDI job should now look like this:



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

PDI Exercise 2: 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.

1. 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.
2. Create a hop between the Load Impala Table step and the **SQL** step.
3. Double click the **SQL** step to edit its properties. In the **Step name** field type Join CDR to Geo Location Data.
4. In the **Connection** field select Impala.
5. In the SQL field type or copy/paste in the following code which can be found in the bdi\_workshop\_code.txt file on your machine:

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

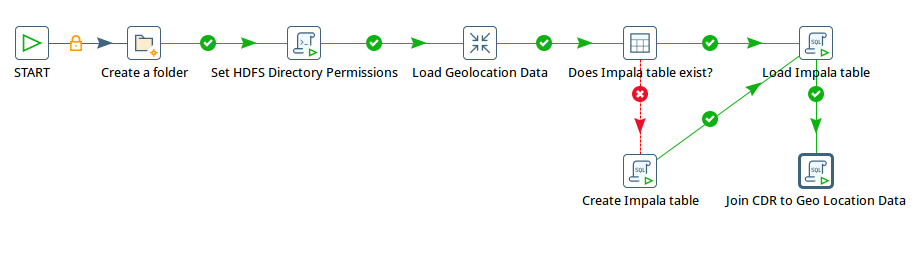
, cdg.new\_lat

, cdg.new\_long

FROM

call\_detail\_records cdr JOIN call\_detail\_geo cdg ON (cdr.source\_number = cdg.source\_number);

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



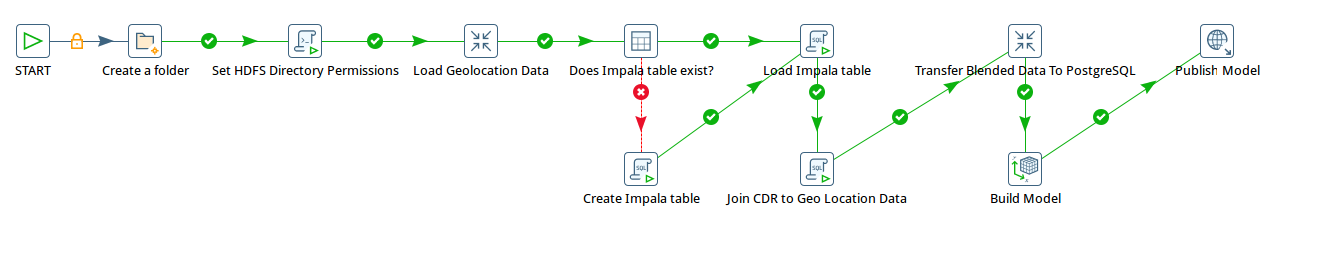
To enable high-performance OLAP analysis and reporting, we would also like to load the combined data set to a PostgreSQL database.

1. 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.
2. Create a hop between the Join CDR to Geo Location Data step and the **Transformation** step.
3. Double click the **Transformation** step to edit its properties. In the **Name of job entry** field type Transfer Blended Data to PostgreSQL.
4. 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\_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.

1. 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.
2. Also from the Modeling folder drag the Publish Model step onto the canvas to the right of the Transfer Blended Data to PostgreSQL step.
3. Create a hop between the Transfer Blended Data to PostgreSQL step and the **Build Model** step.
4. 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.

1. Double click the **Build Model** step to edit its properties. In the **Model Name** field type SDR Geo Cube and for the Output Step field, confirm it is set Postgres Output.
2. Double click the **Publish Model** step to edit its properties.
3. Check Replace Existing Published Model.
4. For the URL enter: [http://localhost:${CURRENT\_PENTAHO\_BA\_PORT}/pentaho/](http://localhost:$%7bCURRENT_PENTAHO_BA_PORT%7d/pentaho/)



The ${CURRENT\_PENTAHO\_BA\_PORT} value referenced above is a variable which is storing the current port the Pentaho Business Analytics server is running on. This port may change. This variable value is set in

/home/[OS Uername]/.kettle/kettle.properties

1. Login to the Pentaho User Console

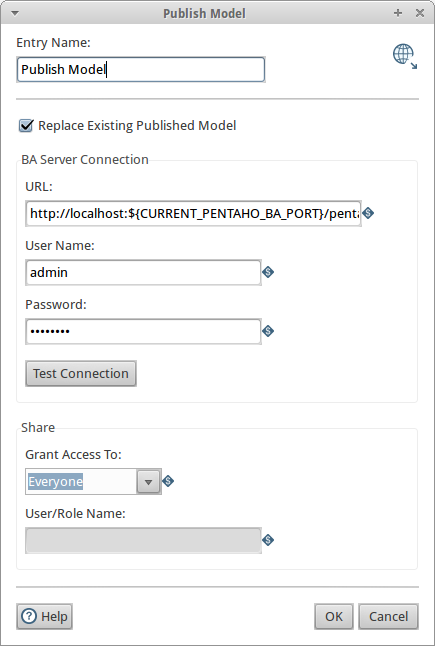
For Pentaho User Console credentials see the document on the Desktop in the Docs folder:

* 1. hds - BDI Workshop Credentials.pdf

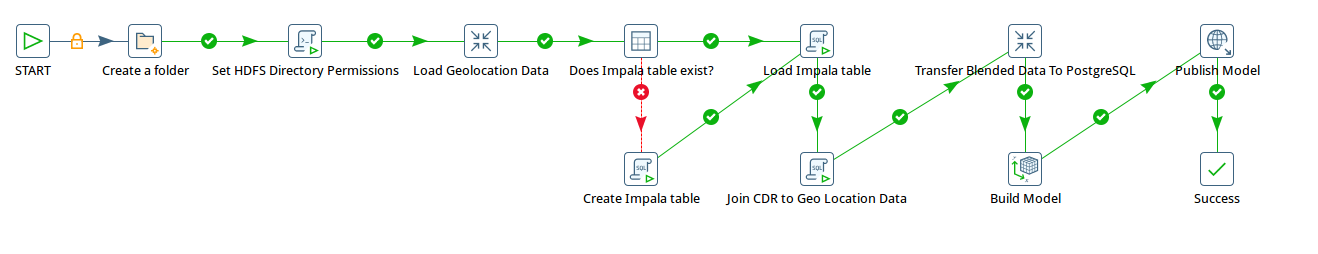
-or-

* 1. bdiw - BDI Workshop Credentials.pdf

1. The remaining fields can be left as the default
2. Click Test Connection.



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



1. From the **File** menu, choose **Save**.
2. Execute the job by choosing Action 🡪 Run from the main menu or by clicking the run icon .
3. The Execute a job dialog will appear. Click the Launch button at the bottom.
4. 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.

Part 2: Explore data in PostgreSQL with Pentaho Analyzer

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. Part Two of the SDR Use Case includes two exercises showcasing how to Pentaho Analyzer for OLAP analysis against PostgreSQL. Your company is considering introducing a new VOIP service. You need to analyze the geo-location and calling data to determine the calling patterns of your customers. You leverage this information to determine the best markets to launch a VOIP pilot service.

The first exercise has you 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. The second exercise plots 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.

Analyzer - Exercise 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. With your Firefox browser navigate to [http://localhost:8081/pentaho/Login](http://localhost:8080/pentaho/Login) to launch the Pentaho User Console (PUC).
2. Login to PUC

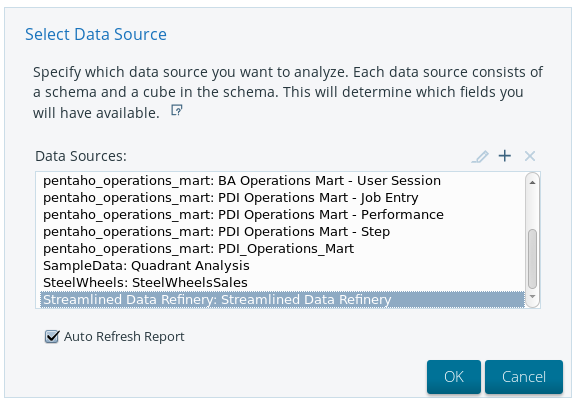
For login credentials, see the document on the Desktop in the Docs folder:

* 1. hds - BDI Workshop Credentials.pdf

-or-

* 1. bdiw - BDI Workshop Credentials.pdf

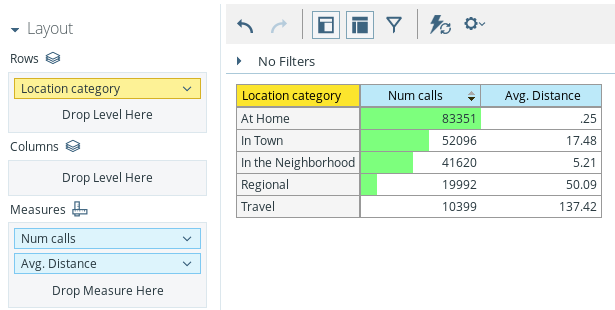
1. Click on the **Create New | Analysis Report** buttons.
2. Select the Streamlined Data Refinery data source at the bottom of the Data Sources list.



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.

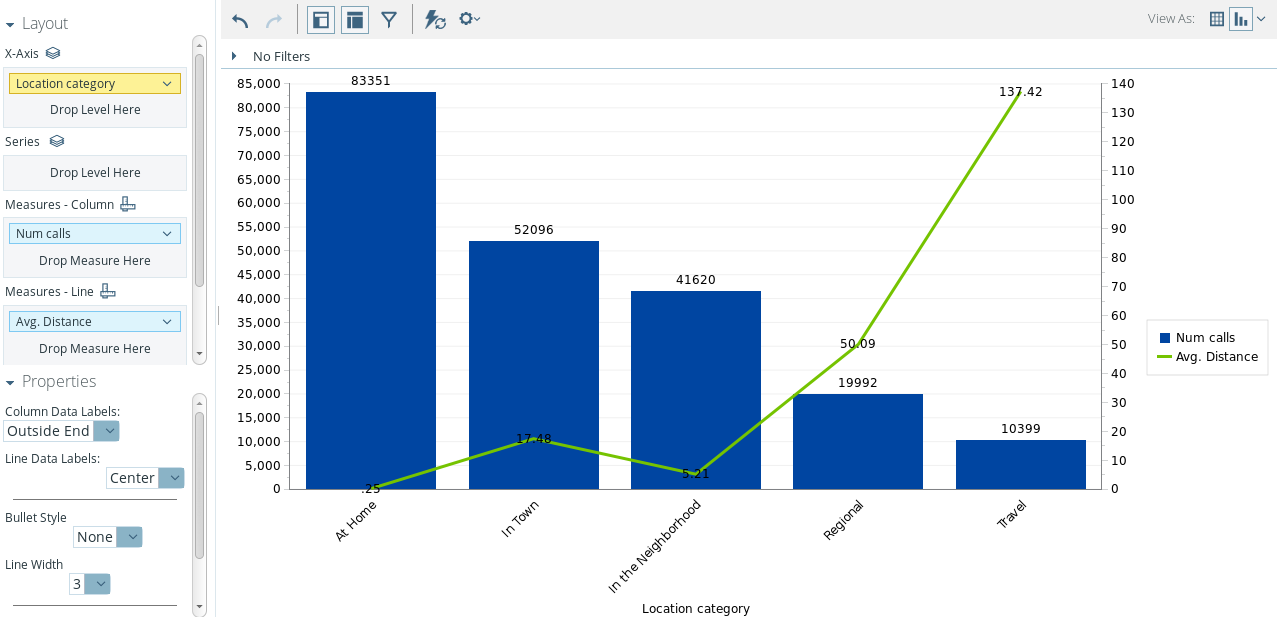
Notice how the measures automatically aggregate the records in Analyzer.

1. Sort the Num calls field in descending order by right clicking the Num Calls column header and selecting **Sort Values High->Low**.
2. 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.

1. To visualize the data, click the chart drop down in the top right of your screen  and choose Column-Line Combo.
2. In the Layout section, drag Avg. Distance from the Measures – Column drop zone down to the Measures – Line drop zone.
3. 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:

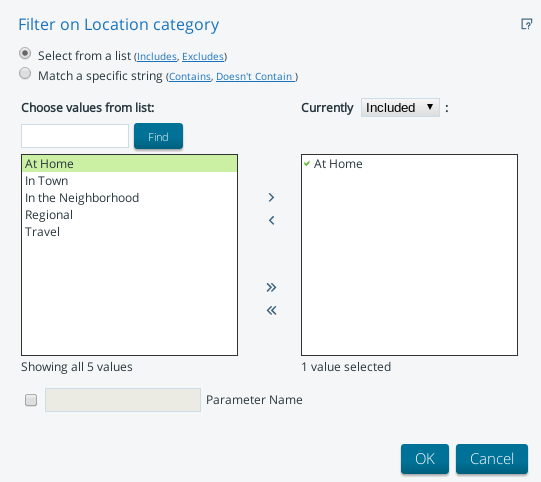


1. Click the Save icon to save the view as SDR Analyzer Exercise 1 in the default /home/[usename] directory.

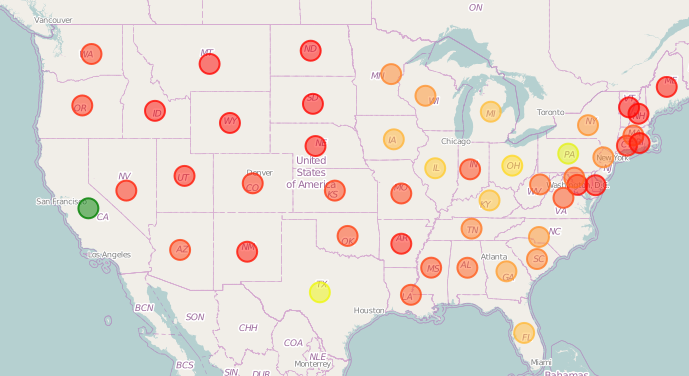
Analyzer - Exercise 2: Analyze and map calling data by geography

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.

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

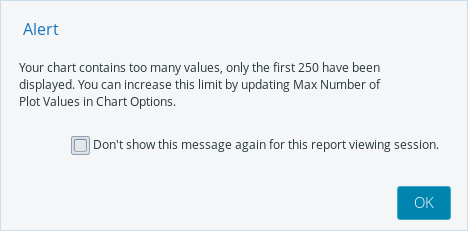


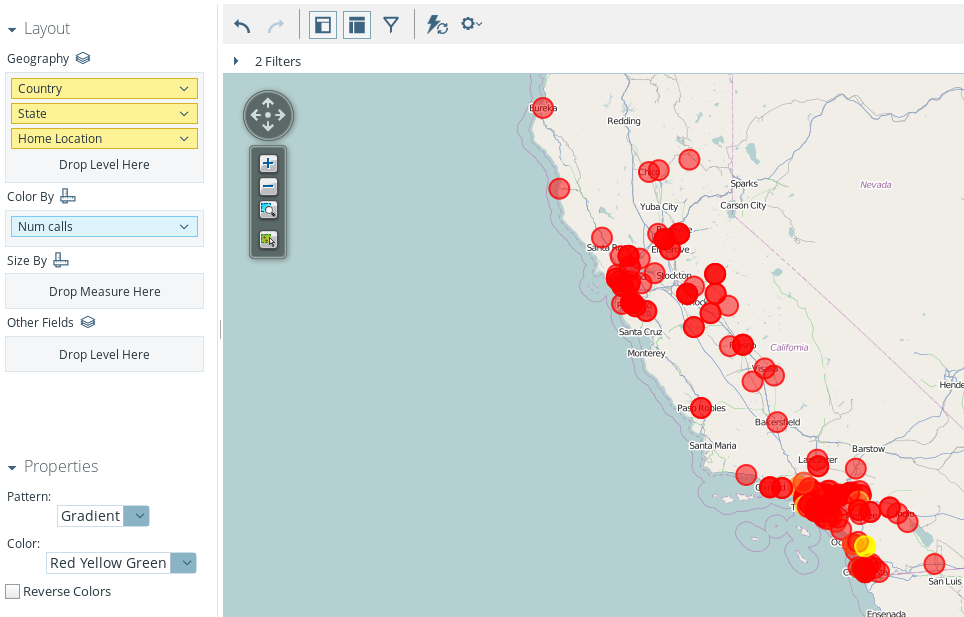
1. Add Country, State, Num Calls to the canvas.
2. Click the chart dropdown and select to **Geo Map**.



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

1. Drill into California by double clicking on the state’s green circle.
2. 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.

1. 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!

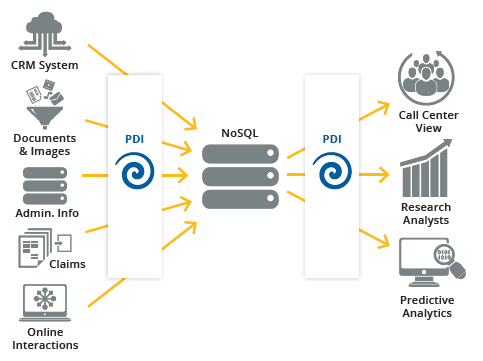
HBase Customer 360 Use Case

What is it?

* Blending a variety of operational & transactional data sources to create an on-demand analytical view across customer touch points and surface the customer experience.
* Providing customer-facing employees and partners with information made available inside everyday line-of-business applications
* Leverages NoSQL database technologies like HBase that provide flexible storage options for single views of data.

Why do it?

* Provide single customer view to sales & services teams, empowering them to grow upsell/cross-sell revenue
* Understand customer perception of your products & services, while decreasing customer churn
* Avoid point-to-point integrations with single repository, and fast queries to improve access to metrics



Value of Pentaho

* Staff savings & productivity: Rapid time to value through drag/drop visual development for big data integration – make big data accessible to all data developers
* Operational Intelligence: Ability to embed analytics into actionable line-of-business applications for each relevant customer-facing role
* Broad & robust data connectivity: Ability to blend traditional sources & big data
* Comprehensive analytics: Visualizations, reports, dashboards, ad hoc analysis provide for all roles

Pentaho components used in this workshop

Pentaho Data Integration (PDI)

Analyzer

What you will accomplish in this workshop

* Part 1 – Use PDI to create a single view in HBase (4 exercises)
* Part 2 – Visualize data with Analyzer (3 exercises)

Part 1: Use PDI to create a single view in HBase

PDI gives users a graphical user interface to build transformations and jobs that solve complex data integration challenges with HBase. PDI provides a scalable solution for migrating to HBase with connectors for all types of disparate data for building out a single customer view. PDI leverages the power of the HBase in-memory database for blending and processing data.

Create the HBase tables

HBase tables must be created via the command line prior to continuing to the exercises. Each table has a corresponding column family. Complete the following steps to create the tables to store the HBase data.

1. Launch the Cloudera Control program by clicking the launch menu icon



1. Enter 4 to select and launch the Cloudera shell (follow prompts)



1. In new terminal, type “hbase shell” and enter the command
2. In hbase shell, create the tables and column families by executing the three commands below
   1. 
   2. 
   3. 

PDI Exercise 1: Create a transformation to load customer records to HBase

This first exercise steps you through the process of creating a transformation to load customer master records from a CSV file, calculate the customer age, and then load the results into the HBase customers table.

1. If PDI is not already open, launch it from the launch menu icon  at the bottom of your screen. Click this icon just *once* to launch Spoon, the client-based authoring GUI for PDI.



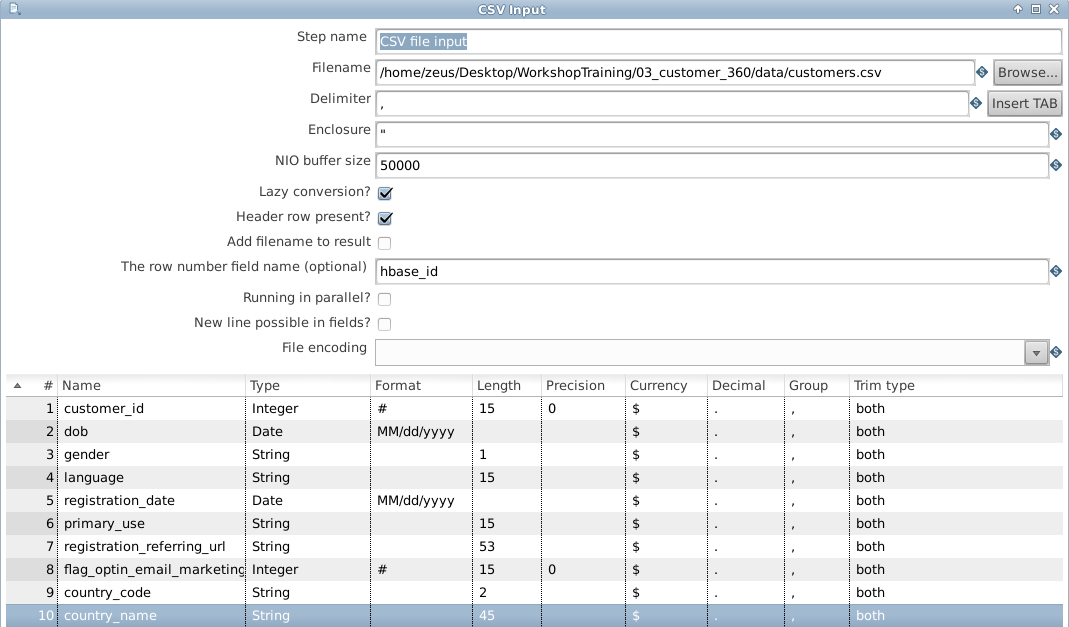
1. You will see the PDI splash screen appear while PDI loads. All open applications appear in the top left section of your screen. You can see Spoon is open in the following screenshot.



1. From the main menu choose File | New | Transformation
2. File | Save to /pentaho/shared\_content/WorkshopTraining/student\_files/03\_customer\_360\_hbase/**t\_load\_customers.ktr**

We need to access Customer data from within a flat file. The file format is a csv file, so you will access the data by configuring a CSV file input step.

1. From the Design tab on the left, expand the Input folder and drag CSV file input onto the canvas
2. Double-click on CSV file input to open its properties
3. Browse to the directory /pentaho/shared\_content/WorkshopTraining/ 03\_customer\_360\_hbase/data/ and select customers.csv.
4. In The row number field name (optional) type hbase\_id
5. Click the Get Fields button at the bottom and enter 50,000 for the Sample Size.
6. Once the scan is finished, click Close to close the scan results.
7. Trim any possible source data character spaces by setting Trim Type to both for every field.

Click the Preview button to preview the data and then click Close. Your CSV Input dialog box should match the following image: 

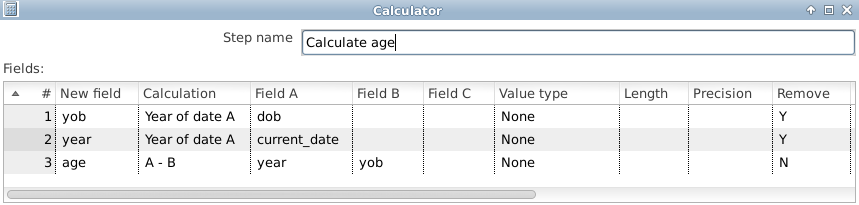
1. Click OK to return to the canvas.

To calculate the age of the customer, we need to know the current year and the year in which the customer was born. But to calculate the current year, we need to know the current date. We will use the **Get System Info** step to learn the current date.

1. From the Design tab on the left, expand the Input folder and drag Get System Info onto the canvas.
2. Double-click on Get System Info step to open its properties.
3. In the Name field type current\_date
4. In the Type field, select from the drop-down menu, system date (variable)
5. Click OK to return to the canvas.
6. To draw a hop between two steps, shift-click the CSV file input step and while holding down your mouse key, drag a **hop** over to the Get System Info step. When prompted, select Main output of step.

Now that we have the current date, we need to derive the current year from the date. And we need to know the year in which the customer was born, which is based on the customer date of birth. With both years in hand, we can calculate the age of the customer. All of these calculations can be performed with the **Calculator** step.

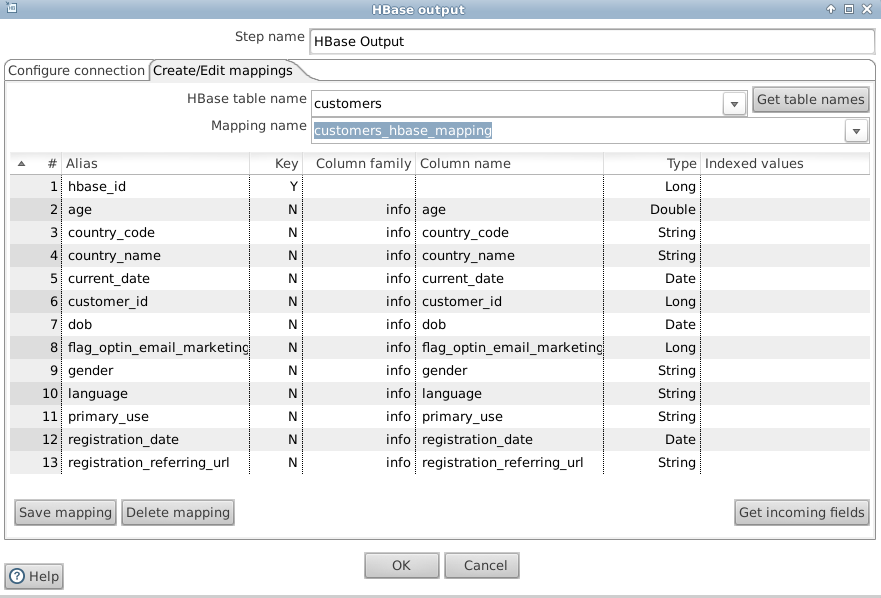
1. From the Design tab on the left, expand the Transform folder and drag the Calculator step onto the canvas.
2. Draw a hop between the Get System Info and Calculator steps.
3. Double-click on Calculator step to open its properties.
4. In the Step name box type Calculate age
5. Follow these steps to create the first year of birth calculation: In the New Field column type yob, in the Calculation column select Year of date A, in the Field A column select the dob field, and finally in the Remove column select Y.
6. Follow these steps to create the second year calculation: In the New Field column type year, in the Calculation column select Year of date A, in the Field A column select the current\_date field, and finally in the Remove column select Y.
7. Follow these steps to create the third customer age calculation: In the New Field column type age, in the Calculation column select A - B, in the Field A column select the year field, in the Field B column select the yob field, and finally in the Remove column, make sure it is set to N.
8. Once your Calculate age step matches the following image, click OK to return to the canvas.



Customer data has been enriched with age information, and we are ready to write the data to HBase.

1. From the Design tab on the left, expand the Big Data folder and drag HBase Output onto the canvas.
2. Draw a hop between the Calculate Age and HBase Output steps. The resulting transformation should match the following image:



1. Double-click on HBase Output step to open its properties.
2. In the Configure connection tab, from the Hadoop cluster drop-down select CDH.
3. Select the Create/Edit mappings tab
4. Click the Get table names button and select customers from the list.
5. Click the Get incoming fields button and select hbase\_id as the Key, with the final mapping matching the below image: 
6. In Mapping name enter customers\_hbase\_mapping and then click the Save Mapping button. Click OK to exit.
7. Once again, double-click on HBase Output
8. In the Configure connection tab, click the button Get table names and from the HBase table name drop-down select customers.
9. In the Configure connection tab, click the Get mappings for the specified table and from the Mapping name drop-down select customers\_hbase\_mapping. Click OK to exit.
10. Click the Launch button at the bottom of the Execute a transformation dialog box. *NOTE: If you get an error about not being able to connect to HBase, or no records are loading to HBase, restart Cloudera using the Cloudera Control program (option 3).*
11. This transformation will load 19,673 customers into the HBase table customers. You will see green checkmarks on each box when the transformation has succeeded: 

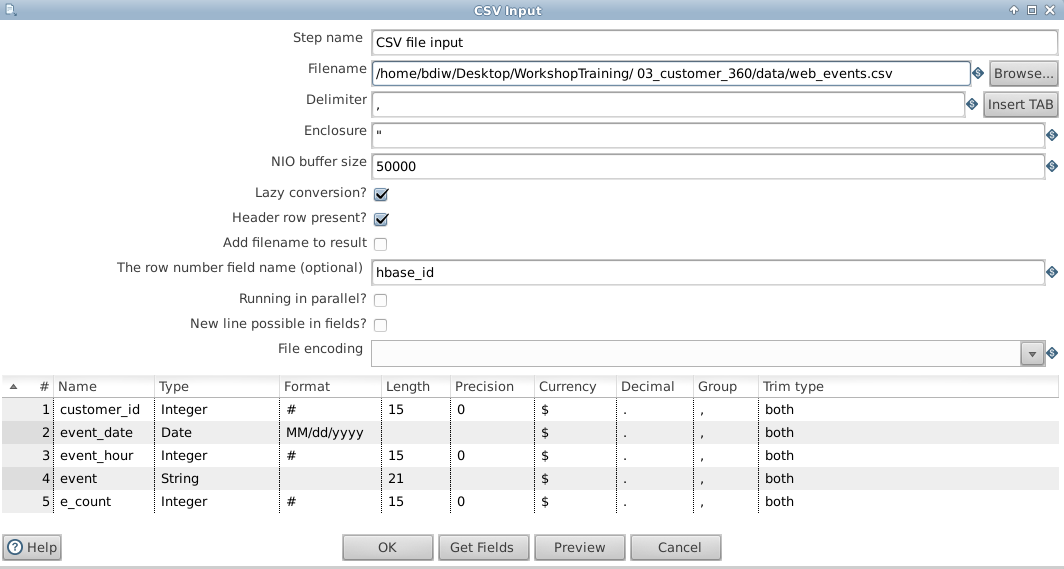
PDI Exercise 2: Create a transformation to load web events to HBase

This second exercise steps you through the process of creating a transformation to load the second source of data, customer web clickstream (web event) records, from a CSV file into the webevents HBase table created prior to exercise 1.

1. From the main menu choose File | New | Transformation
2. File | Save to /pentaho/shared\_content/WorkshopTraining/student\_files/03\_customer\_360\_hbase/t\_load\_web\_events.ktr

To load customer web events data from a CSV file into the HBase table, first we need to configure a CSV file input step to access the web events.

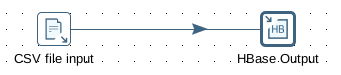
1. From the Design tab on the left, expand the Input folder and drag CSV file input onto the canvas**.**
2. Double-click on CSV file input to open its properties
3. Browse to the directory /pentaho/shared\_content/WorkshopTraining/ 03\_customer\_360\_hbase/data/ and select web\_events.csv.
4. In the field “The row number field name (optional)”, enter hbase\_id
5. Click the Get Fields button at the bottom and enter 50,000 for the Sample Size.
6. Click Close to close the scan results.
7. Click the Preview button to preview the data and then click Close. Your CSV Input dialog box match the following image:

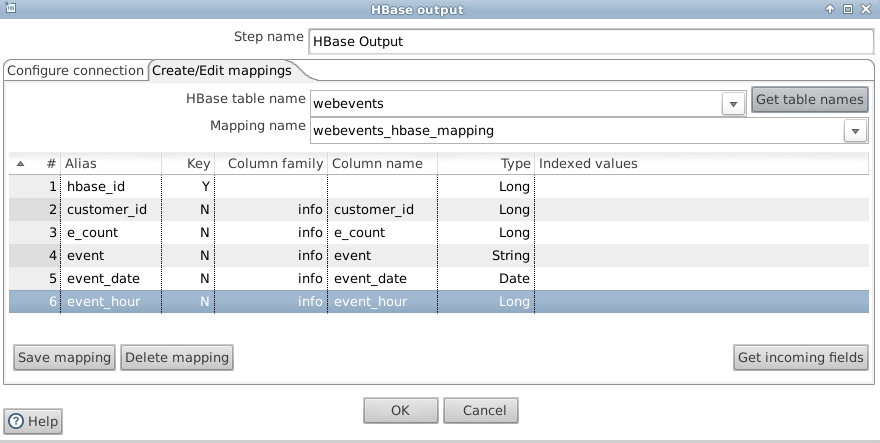
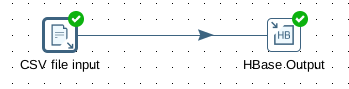


1. Click OK to return to the canvas.

The HBase Output step will truncate the target table prior to loading it.

1. From the Design tab on the left, collapse the Input folder and expand the Big Data folder; then, select and drag HBase Output onto the canvas.
2. Shift-click the CSV file input step and while holding down your mouse key, drag a **hop** over to the HBase Output step. When prompted, select Main output of step.



1. Double-click on HBase Output step to open its properties.
2. In the Configure connection tab, from the Hadoop cluster drop-down select CDH 5.3.
3. Select the Create/Edit mappings tab
4. Click on Get table names button and select webevents from the list.
5. Click the Get incoming fields button and enter hbase\_id as the Key, with the final mapping matching the below image: 
6. In Mapping name enter webevents\_hbase\_mapping and then click the Save Mapping button. Click OK to exit.
7. Double-click on HBase Output.
8. In the Configure connection tab, click the Get table names button and from the HBase table name drop-down select webevents.
9. In the Configure connection tab, click the Get mappings for the specified table and from the Mapping name drop-down select webevents\_hbase\_mapping. Click OK to exit.
10. Click the Launch button at the bottom of the Execute a transformation dialog box.
11. This transformation will load 51,419 web event records into the HBase table webevents. You will see green checkmarks on each box when the transformation has succeeded: 

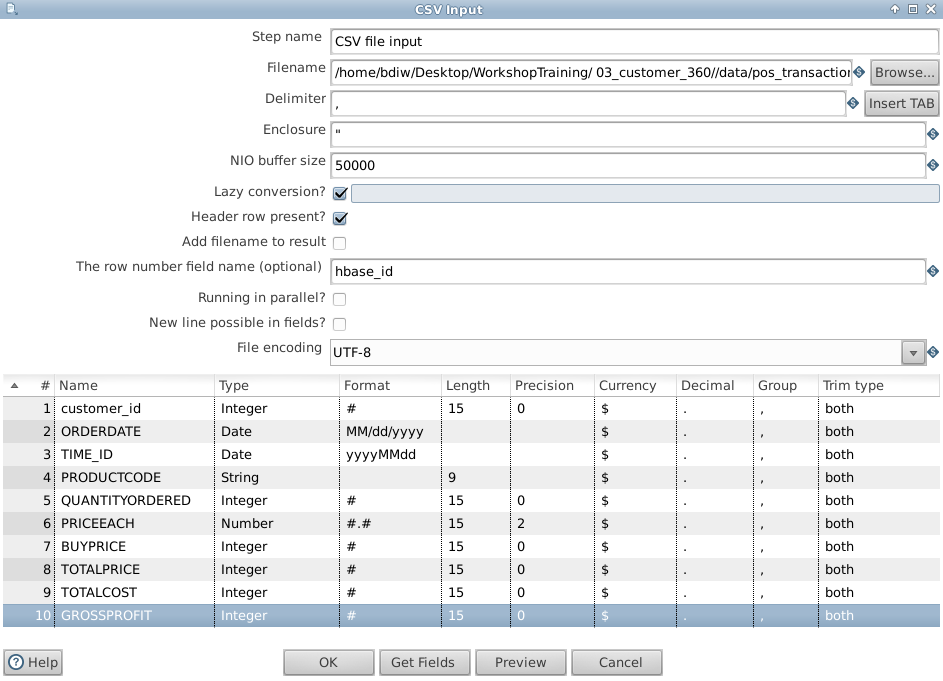
PDI Exercise 3: Create a transformation to load POS transactions to HBase

This third exercise steps you through the process of creating a transformation to load the third and final data source, customer point-of-sale (POS) sale transaction records, from a CSV file into the pos HBase table created prior to exercise 1. You use the number range step to bin sales orders into buckets based on the size of the order.

1. From the main menu choose File | New | Transformation
2. File | Save to /pentaho/shared\_content/WorkshopTraining/student\_files/03\_customer\_360\_hbase/t\_load\_pos.ktr

To blend point-of-sale transactions from a CSV file into the HBase data collection, we need to configure a CSV file input step to access the transactions.

1. From the Design tab on the left, expand the Input folder and drag CSV file input onto the canvas.
2. Double-click on CSV file input to open its properties
3. Browse to the directory /pentaho/shared\_content/WorkshopTraining/ 03\_customer\_360\_hbase/data/ and select pos\_transactions.csv.
4. In the field The row number field name (optional), enter hbase\_id
5. Click the Get Fields button at the bottom and enter 5,000 for the Sample Size.
6. Click the Preview button to preview the data and then click Close. Your CSV Input dialog box should match the following image:

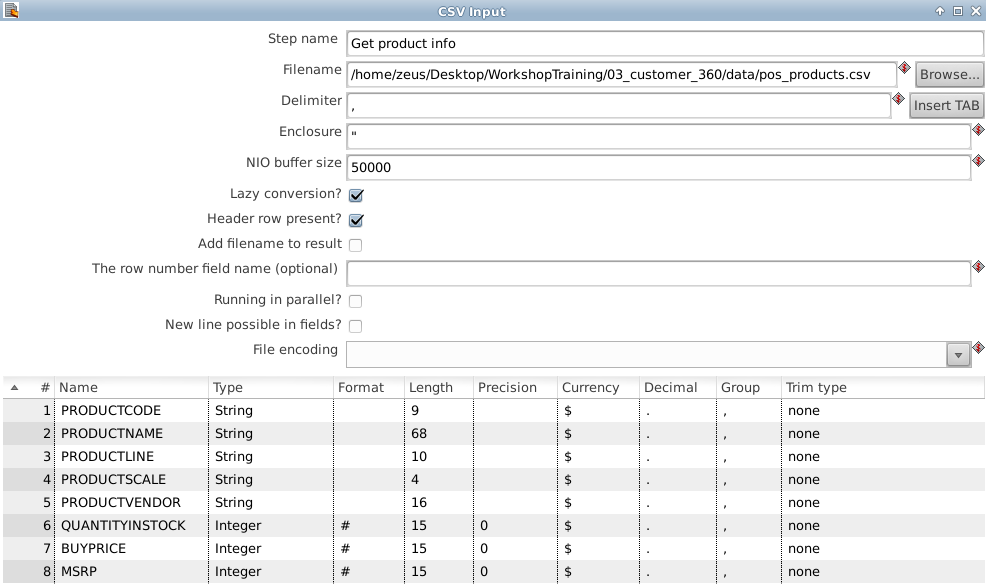


1. Click OK to return to the canvas.



The POS file only has product code on the transaction. We need to enhance the data with Product Name, Product Line, and Product Vendor. To do so, we will look up the product information in a flat file to match the transaction product code with its relevant name, line, and vendor.

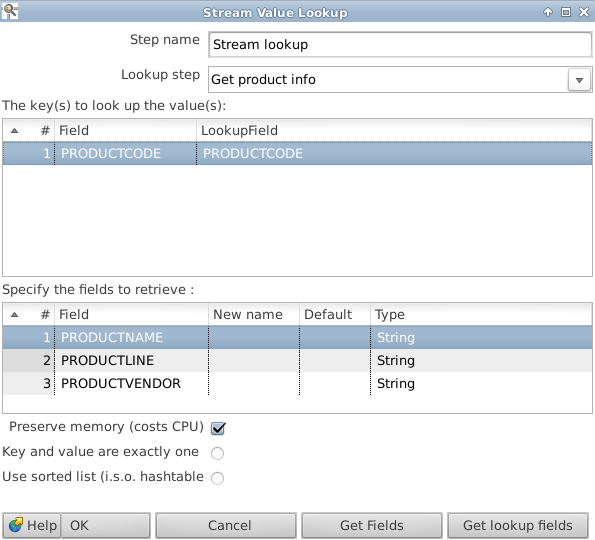
1. From the Design tab on the left, expand the Lookup folder and drag Stream lookup onto the canvas.
2. Create a hop between the CSV file input and Stream lookup steps.
3. From the Design tab on the left, expand the Input folder and drag a second CSV file input onto the canvas and place it above the Stream lookup step.
4. Double-click on CSV file input to open its properties
5. Change the Step name to Get product info
6. Browse to the directory /pentaho/shared\_content/WorkshopTraining/ 03\_customer\_360\_hbase/data/ and select pos\_products.csv.
7. Click the Get Fields button at the bottom and enter 5,000 for the Sample Size.
8. Click the Preview button to preview the data and then click Close. Your CSV Input dialog box should look like the following image:



1. Click OK to return to the canvas
2. Create a hop from the second Get product info step to the Stream lookup step.

We need to configure the Stream Lookup to retrieve Product Name, Product Line, and Product Vendor from the lookup file for every product code we find in the POS records.

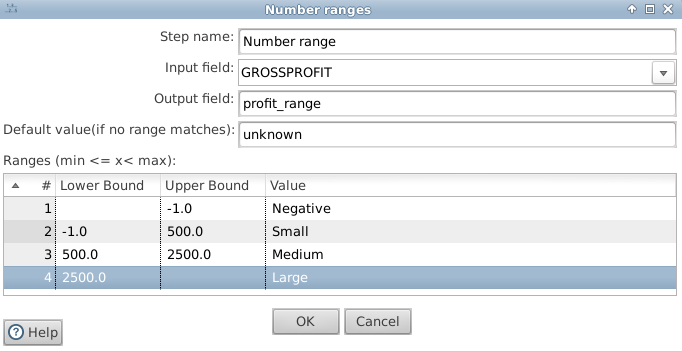
1. Double-click on Stream lookup step to open its properties
2. In the **Lookup** step select Get product info.
3. In the **Key(s) to Lookup Value(s)** section select PRODUCTCODE in the **Field** column and select PRODUCTCODE as the **LookupField** column.
4. Click the **Get Lookup Fields** button to populate the fields to retrieve section at the bottom.
5. Highlight and delete the following fields: PRODUCTCODE, PRODUCTSCALE, QUANTITYINSTOCK, BUYPRICE, MSRP from the fields to retrieve section. Your Stream Lookup dialog box should now match the following image:



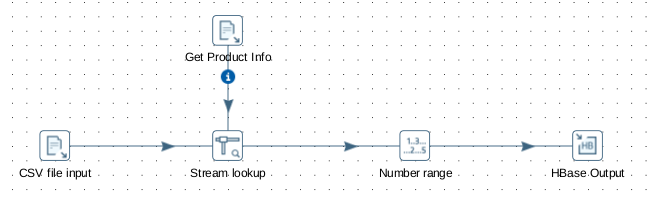
1. Click OK to return to the canvas.
2. From the Design tab on the left, expand the Transform folder and drag Select Values onto the canvas.
3. Click OK to return to the main canvas.

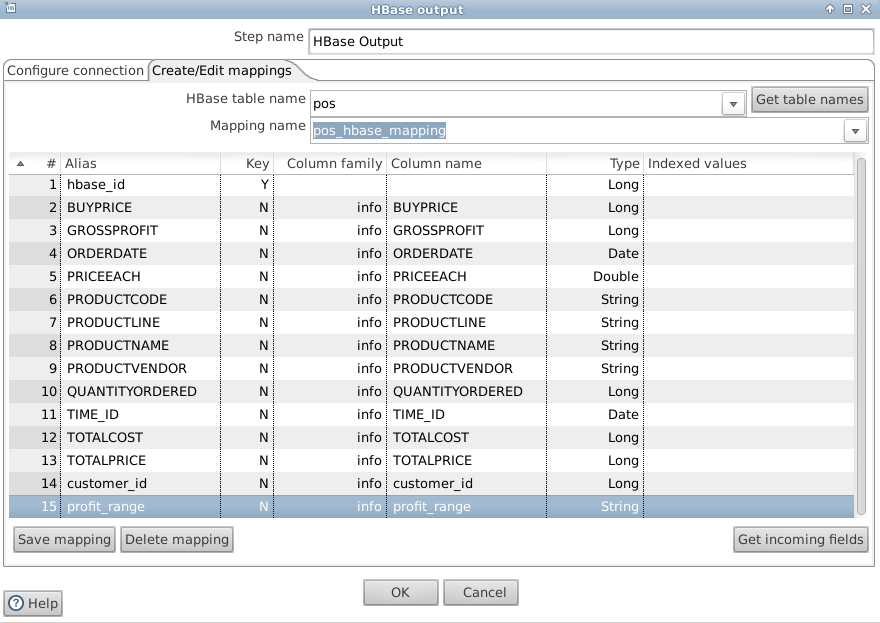
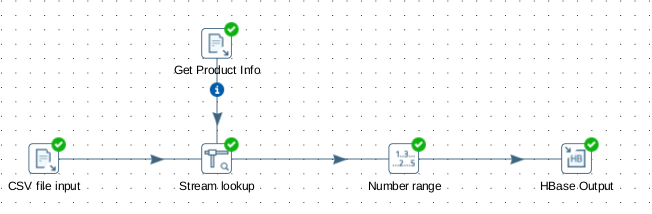
 It may be useful to categorize the sale orders into buckets for further analysis, based on order size. We can create a categorization of small, medium, and large.

1. From within the Transform folder, select and drag Number Range onto the canvas.
2. Create a hop between the Stream lookup and Number Range steps.
3. Double-click on Number Range step to open its properties.
4. For Input field select GROSSPROFIT from the list.
5. For Output field type profit\_range.
6. Enter the following values for Lower Bound, Upper Bound and Value columns.



1. Click OK to return to the canvas.
2. From the Design tab on the left, expand the Big Data folder and drag HBase Output onto the canvas.
3. Create a hop between the Number Range and HBase Output step. The resulting transformation should match the following image:

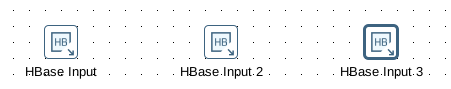


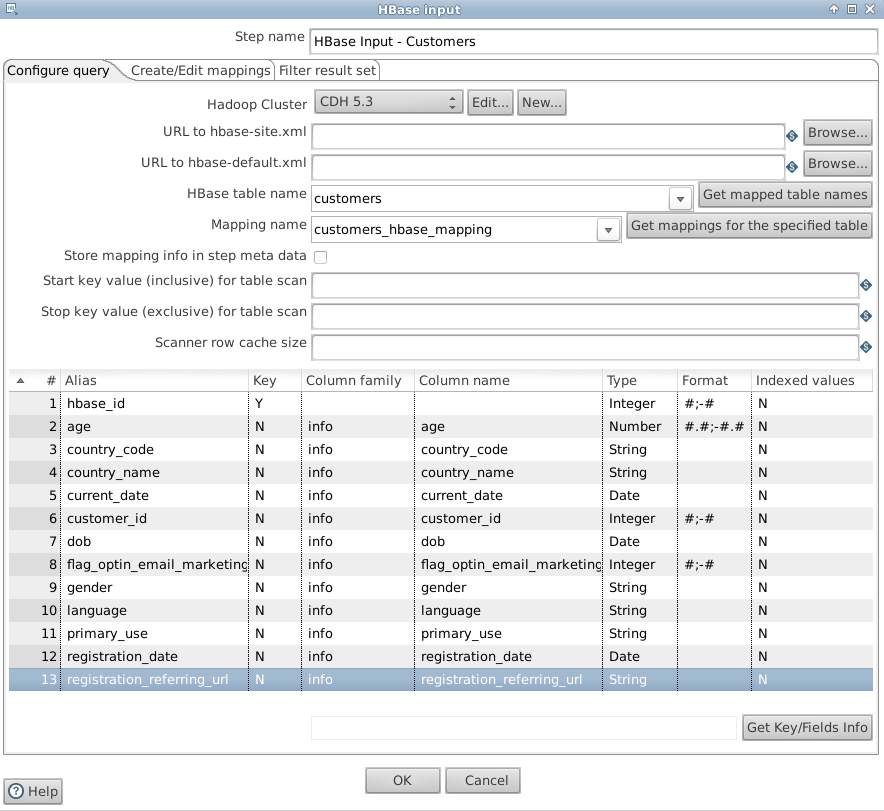
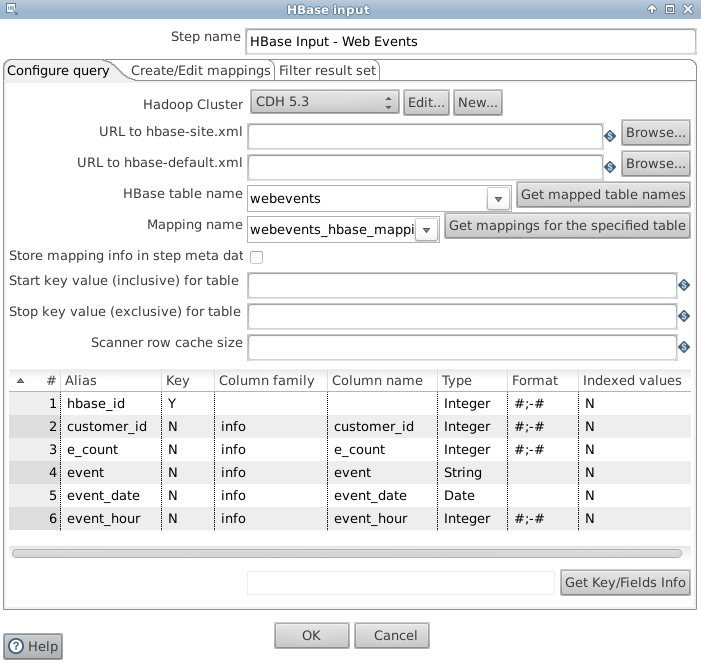
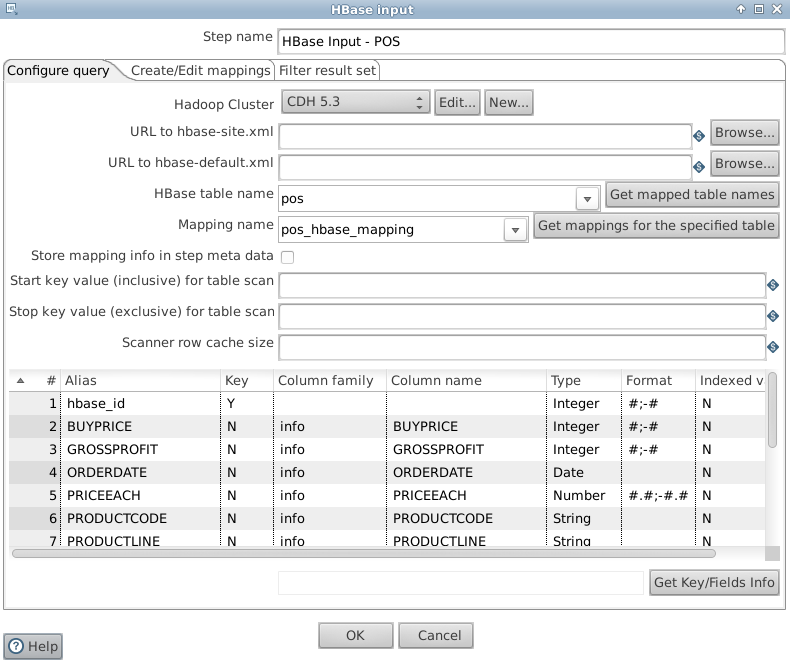
1. Double-click on HBase Output step to open its properties. In the Configure connection tab, from the Hadoop cluster drop-down select CDH 5.3.
2. Select the Create/Edit mappings tab
3. Click on Get table names button and select pos from the list.
4. Click the Get incoming fields button and define hbase\_id as the Key, with the final mapping matching the below image: 
5. In Mapping name enter pos\_hbase\_mapping and then click the Save Mapping button. Click OK to exit.
6. Double click HBase Output
7. In the Configure connection tab, click the Get table names button and then from the HBase table name drop-down select pos
8. In the Configure connection tab, click the Get mappings for the specified table button and from the Mapping name drop-down select pos\_hbase\_mapping. Click OK to exit.
9. Click the Launch button at the bottom of the Execute a transformation dialog box.
10. This transformation will load 991 POS records into the HBase table pos. You will see green checkmarks on each box when the transformation has succeeded: 

PDI Exercise 4: Create a transformation to merge and annotate the three HBase tables

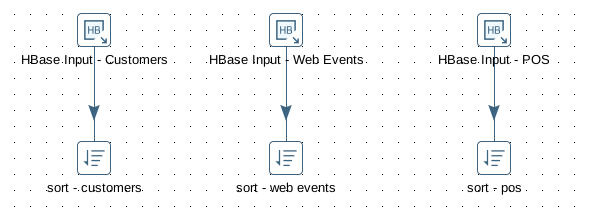
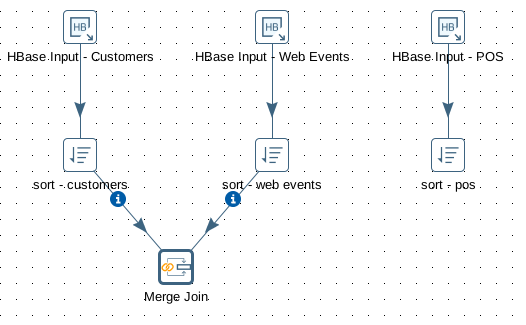
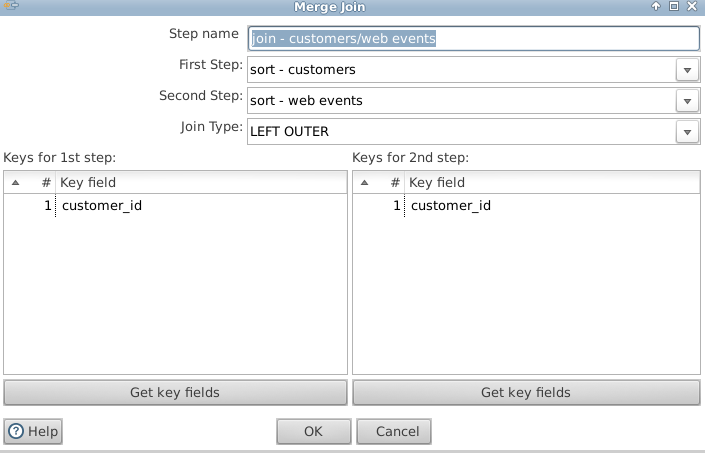
In the fourth exercise, you merge the three HBase tables and prepare them for Analyzer Reports. The final output is to a PostgreSQL database where the data is persisted for reporting. Note that sections of the final transformation are shown in the following steps with the whole image appearing at the end of this exercise. You will be using a blank, pre-existing transformation to utilize an existing database connection.

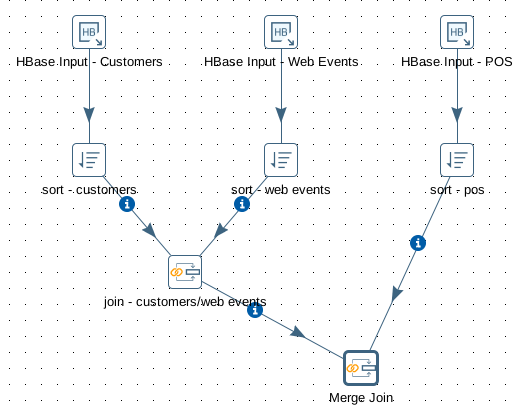
1. File | Open… the transformation /pentaho/shared\_content/WorkshopTraining/03\_customer\_360\_mongodb/pdi\_transformations/t\_load\_postgres.ktr
2. From the Design tab on the left, expand the Big Data folder and drag HBase Input onto the canvas.
3. Repeat step one for a total of 3 HBase Input steps:



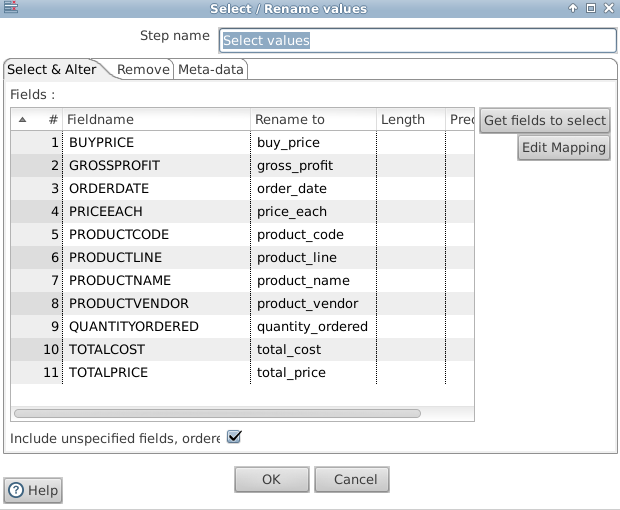
1. Double-click the HBase Input step to open its properties.
2. Change Step name to HBase Input – Customers.
3. From the Hadoop Cluster drop-down select CDH 5.3.
4. Click the Get mapped table names button and select customers from the drop-down.
5. Click Get mappings for the specified table button and select customers\_hbase\_mapping from the drop-down.
6. Click the Get Key/Fields info button. The step should now match the image below: 
7. Open the HBase Input 2 step and change Step name to HBase Input – Web Events.
8. From the Hadoop Cluster drop-down select CDH 5.3.
9. Click the Get mapped table names button and select webevents from the drop-down.
10. Click Get mappings for the specified table button and select webevents\_hbase\_mapping from the drop-down.
11. Click the Get Key/Fields info button. The step should now match the image below: 
12. Open the HBase Input 3 step and change Step name to HBase Input – POS.
13. From the Hadoop Cluster drop-down select CDH 5.3.
14. Click the Get mapped table names button and select pos from the drop-down.
15. Click Get mappings for the specified table button and select pos\_hbase\_mapping from the drop-down.
16. Click the Get Key/Fields info button. The step should now match the image below: 

All three data sources will be merged together on the field customer\_id. The “Merge join” requires data to be sorted by the merge key.

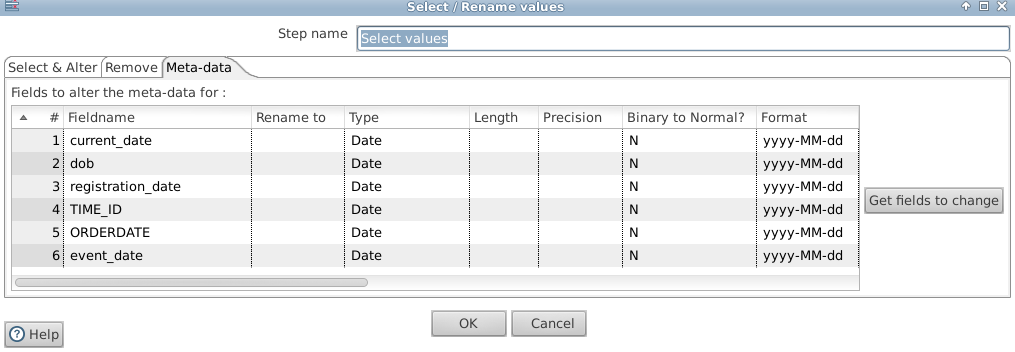
1. From the Design tab on the left, expand the Transform folder and select and drag Sort rows onto the canvas.
2. Connect a hop from one of the Sort rows to a corresponding HBase Input:
3. Open each Sort rows, change the Step name per above image and from Fieldname select customer\_id. The data is now sorted and ready to be merged.
4. From the Design tab on the left, expand the Joins folder and drag Merge join onto the canvas.
5. Create a hop from sort – customers to Merge join
6. Create another hop from sort – web events to Merge join: 
7. Double click Merge join to open its properties.
8. Change Step name to join - customers/web events.
9. From First step select sort – customers; from Second step select sort – web events.
10. From Join Type select LEFT OUTER
11. For Keys for 1st step enter customer\_id; for Keys for 2nd step enter customer\_id. Your step should now match the image below: 
12. Click OK, and then click I understand in the pop-up warning to exit.
13. From the Design tab on the left, expand the Joins folder and drag Merge join onto the canvas.
14. Create a hop from join - customers/web events to Merge join.
15. Create another hop from sort – pos to step Merge join:

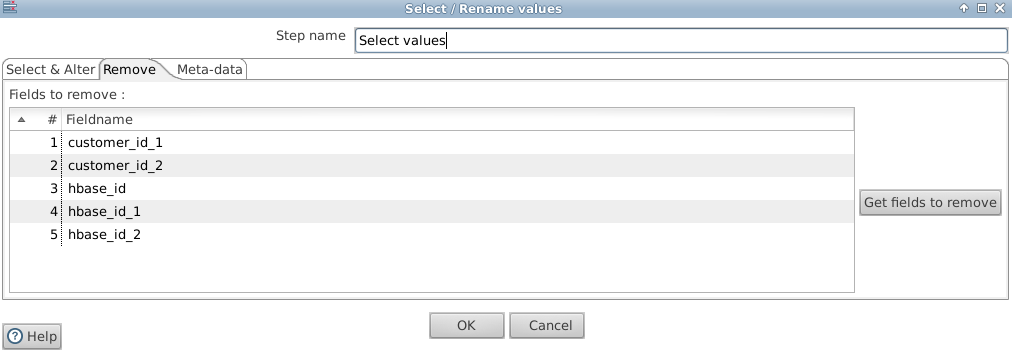


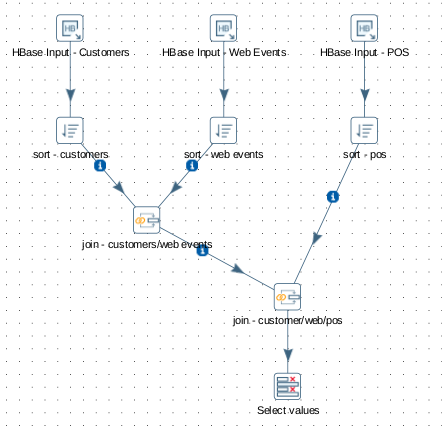
1. Double click Merge join to open its properties.
2. Change Step name to join - customer/web/pos.
3. From First step select join - customers/web events; from Second step select *sort – pos*.
4. From Join Type select “LEFT OUTER”.
5. For Keys for 1st step enter *customer\_id*; for Keys for 2nd step enter *customer\_id.*
6. Click OK, and then click I understand in the pop-up warning to exit.
7. From the Design tab on the left, expand the Transform folder; then, select and drag Select values onto the canvas
8. Connect a hop from join - customer/web/pos to Select Values.
9. Double-click Select values to open its properties.
10. In the Select & Alter tab we will rename fields so that they are more readable in the reports created in the next section. Ensure that the box is checked next to Include unspecified fields, ordered by name and then select the Fieldname(s) and make Rename to entries per the image below:



1. In the Meta-data tab select the Fieldname and enter the Format for fields per below image:

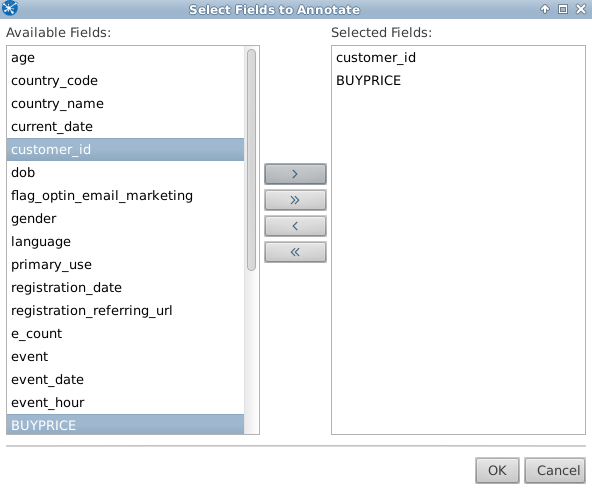


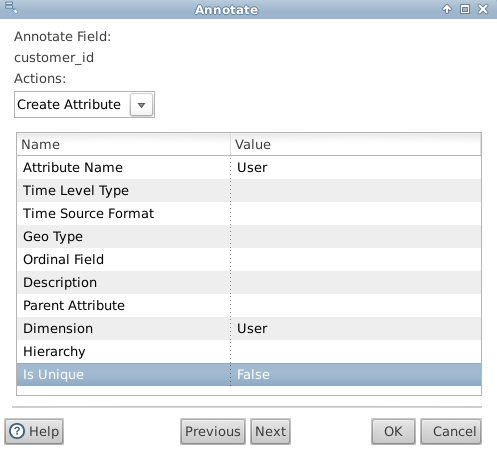
1. Next, go to the Remove tab and select the following fields in Fieldname: 
2. Click OK to exit. Your transformation should now match the below image:



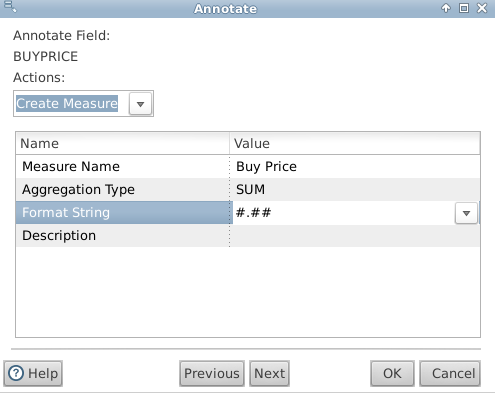
In the next steps, you will partially build the “Annotate Stream” step. This step assists in building the Analyzer reports in the next section by creating aggregations and time measures.

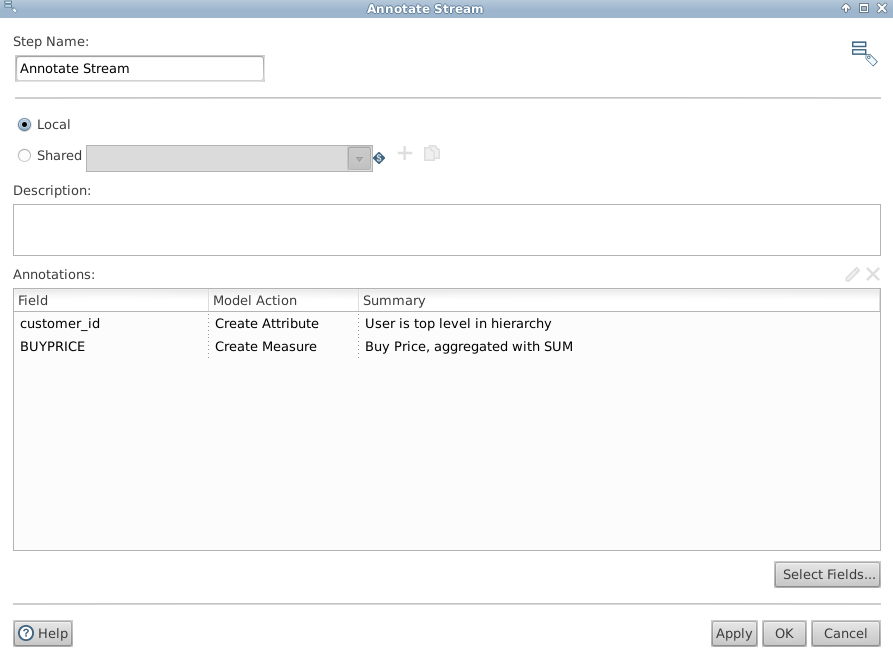
As this is a lengthy and repetitive step to develop, the step source code is provided in a text file. You will be instructed to use the pre-developed step later.

1. From the Design tab on the left, expand the Flow folder; then select and drag Annotate Stream onto the canvas.
2. Connect a hop from Select values to Annotate Stream.
3. Double-click Annotate Stream to open its properties.
4. Click the Select Fields… button at the bottom right.
5. In the new window, select customer\_id from the left pane; hold Ctrl and also select BUYPRICE.
6. Click the single right arrow button . The window should match the image below: 
7. Click OK to return to the main Annotate Stream window.
8. Double-click customer\_id in the bottom grid.
9. From Actions select Create Attribute and make entries per image below:



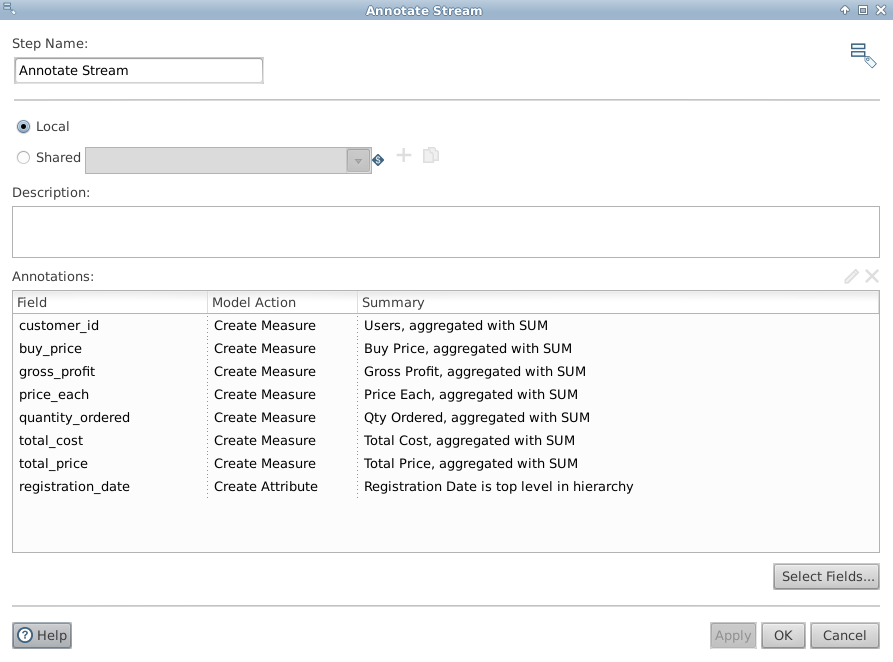
1. Click OK to exit back to the main Annotate Stream window.
2. Next, double click BUYPRICE in the grid.
3. From Actions select Create Measure and make to match the image below:



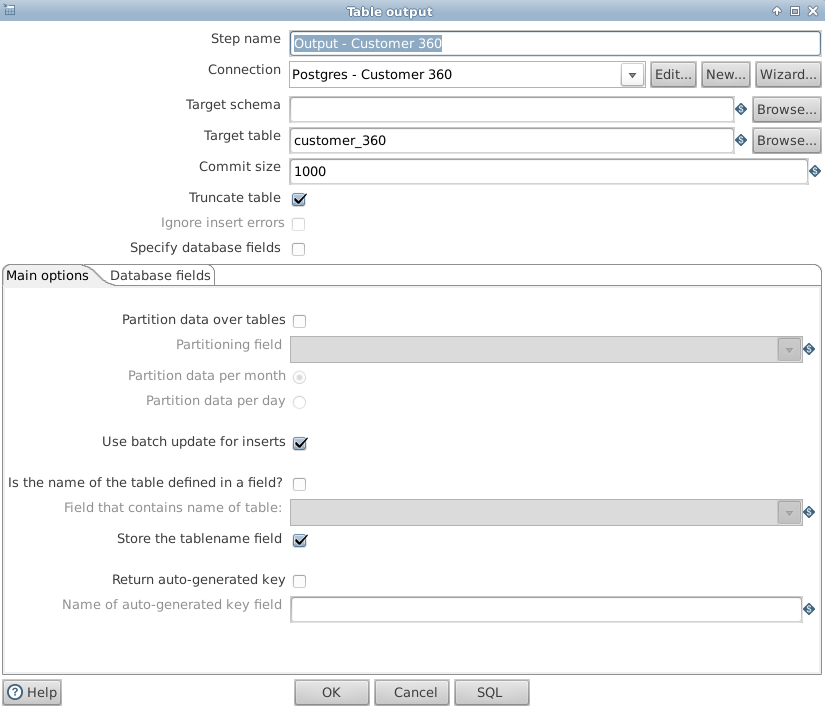
1. Click OK to exit back to the main Annotate Stream window
2. Click Apply and then click OK to exit.

Next, you will paste in the full, pre-developed “Annotate Stream” step. PDI accepts copying/pasting of raw PDI step XML to the canvas to create a step.

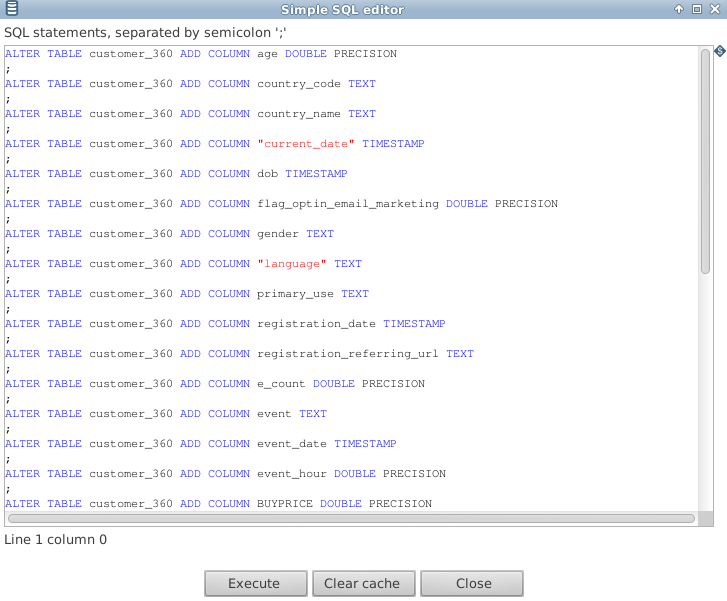
1. Delete the Annotate Stream step you’ve just created by right-click | Delete Step. Select Yes in warning prompt.
2. Open the file /pentaho/shared\_content/WorkshopTraining/03\_customer\_360\_hbase/customer\_360\_annotate\_stream\_step.txt
3. Select all of the file contents and copy to clipboard (place cursor in document, Ctrl+a then Ctrl+c)
4. Return to Spoon canvas, right click | Paste from clipboard. You will now see the step Annotate Stream on the canvas.
5. Connect a hop from Select values to Annotate Stream
6. Double click Annotate Stream to see the additional development:



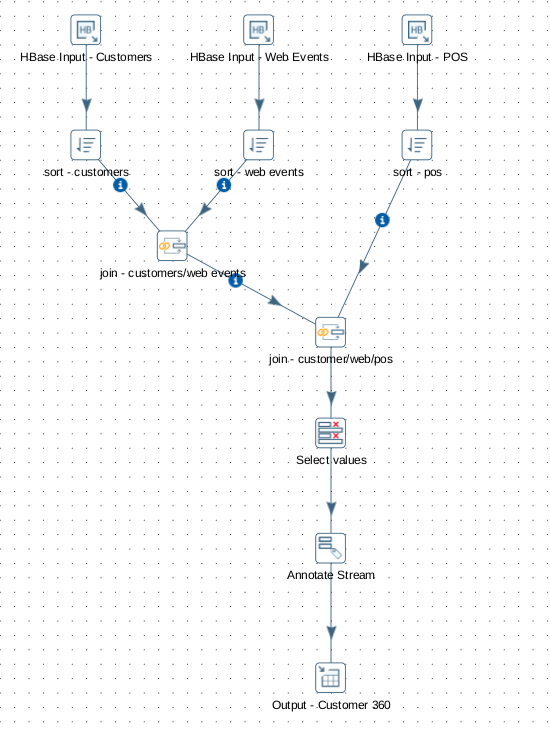
1. Click Cancel to close.
2. From the Design tab on the left, expand the Output folder and drag Table output onto the canvas.
3. Connect a hop from Annotate Stream to Table Output.
4. Double-click Table Output to open its properties.
5. Change Step name to Output – Customer 360
6. Under Connection select Postgres – Customer 360
7. Change Target table to customer\_360
8. Check Truncate table
9. Your step should match the image below:



1. While in Table output, click the SQL button at the bottom right, and a new window will open:



1. Click **Execute.**
2. Click **OK** in new window.
3. Click **Close**.
4. Click **OK** to exit **Table output**. Your final transformation should match the image below:

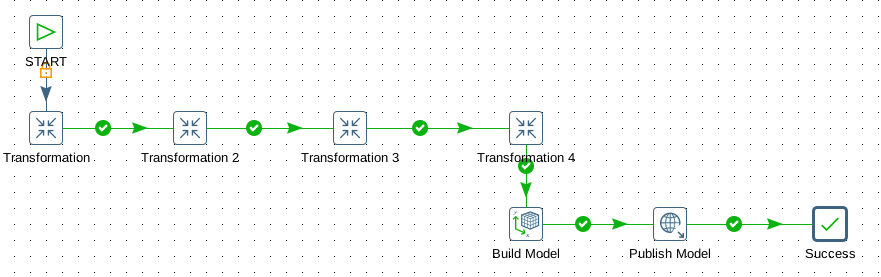


1. Click the Launch button at the bottom of the Execute a transformation dialog box. 52,772 records should be output.

PDI Exercise 5: Create a job to sequence and automate transformations and publish a model

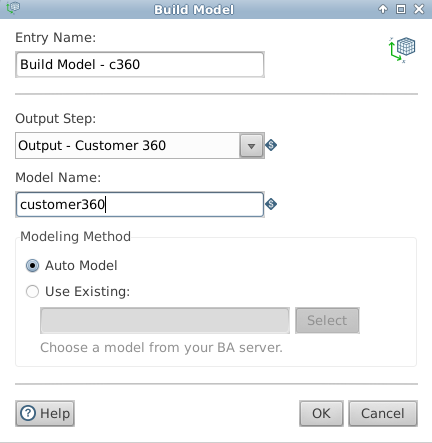
In this exercise you build a PDI job used to execute all three transformations created in the previous three exercises. PDI jobs can be scheduled for automated processing on a recurring schedule.

1. From the main menu choose File | New | Job.
2. From the Design tab on the left, expand the General folder; then, select and drag Start onto the canvas.
3. While in the General folder, add three Transformation steps and a Success step to the canvas.
4. Connect each job step with a hop in the following order: START | Transformation | Transformation 2 | Transformation 3 | Transformation 4 | Build Model | Publish Model | Success.



1. Double-click the first Transformation job step to edit the job entry details.
2. Change the Name of job entry to Load HBase - Customers.
3. For the Transformation filename section specify the transformation to run by browsing to /pentaho/shared\_content/WorkshopTraining/student\_files/03\_customer\_360\_hbase and choosing t\_load\_customers.ktr.
4. Click OK to return to the job canvas.
5. Double-click the Transformation 2 job step to edit the job entry details.
6. Change the Name of job entry to Load HBase - Web Events.
7. For the **Transformation filename** section specify the transformation to run by browsing to /pentaho/shared\_content/WorkshopTraining/student\_files/03\_customer\_360\_hbase and choosing t\_load\_web\_events.ktr.
8. Click OK to return to the job canvas.
9. Double-click the Transformation 3 job step to edit the job entry details.
10. Change the Name of job entry to Load HBase - POS.
11. For the **Transformation filename** section specify the transformation to run by browsing to /pentaho/shared\_content/WorkshopTraining/student\_files/03\_customer\_360\_hbase and choosing t\_load\_pos.ktr.
12. Double-click the Transformation 4 job step to edit the job entry details
13. Change the Name of job entry to Load Postgres.
14. For the **Transformation filename** section specify the transformation to run by browsing to /pentaho/shared\_content/WorkshopTraining/student\_files/03\_customer\_360\_hbase and choosing t\_load\_postgres.ktr.
15. Double-click the **Build Model** job step to edit the job entry details.

Here, you create the customer360 model that will be published and used for analysis in Part 2.

1. Change the **Name of job entry** to Build Model – c360.
2. From **Output Step** select Output – Customer 360.
3. In **Model Name** enter customer360. Your step should match the image below: 
4. Click **OK** to exit.
5. Double-click the **Publish Model** job step to edit the job entry details.
6. Change the **Entry name** to Publish – c360
7. In **URL**, enter <http://localhost:${CURRENT_PENTAHO_BA_PORT}/pentaho/>
8. For **User Name** and **Password**

See Pentaho User Console credentials section in the document on the Desktop in the Docs folder:

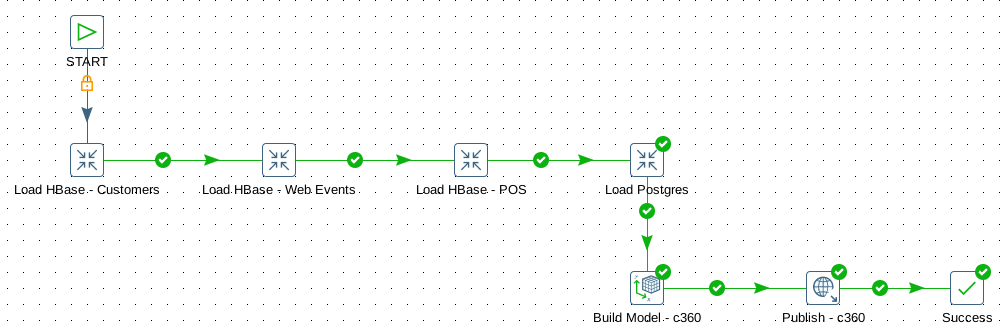
* 1. hds - BDI Workshop Credentials.pdf

-or-

* 1. bdiw - BDI Workshop Credentials.pdf

1. Your step should match the image below:



1. Click OK to return to the job canvas. Your job should match the following screenshot: 
2. Use your mouse to lasso-select all five job steps and then right-click any step.
3. From the right-click menu select Align / Distribute and then Snap to grid.
4. Click the save icon in the toolbar to save your new job as j\_load\_customer360.kjb in the following directory: /pentaho/shared\_content/WorkshopTraining/student\_files/03\_customer\_360\_hbase.
5. Run your job by either clicking  or choosing Action | Run from the main menu. When the job completes without errors, green check marks will appear on each job step.

Part 2: Visualize Data with Analyzer

You now have a customer360 collection containing customer master data, point-of-sale transactions, and website clickstream event data. Pentaho Analyzer provides analysis using the blended HBase data. The following exercises use Analyzer for HBase to analyze and visualize the data in the customer360 collection.

Analyzer Exercise 1: Create a scatterplot visualization

This exercise has you create a scatterplot displaying individual products across two measures and then color each point in the scatterplot by the newly calculated field, Profit\_range.

Note: The BA server should already be started from the previous exercises, but if you need to start it, execute the following script: /pentaho/current\_version/ctlscript.sh start.

1. With your Firefox browser navigate to [http://localhost:8081/pentaho/Login](http://localhost:8080/pentaho/Login) to launch the Pentaho User Console (PUC).
2. Login to PUC

See Pentaho User Console credentials section in the document on the Desktop in the Docs folder:

* 1. hds - BDI Workshop Credentials.pdf

-or-

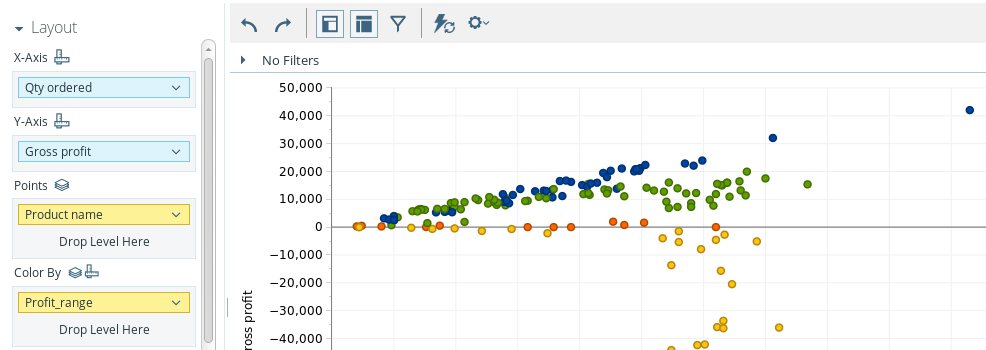
* 1. bdiw - BDI Workshop Credentials.pdf

1. Click on the **Create New | Analysis Report** buttons.
2. Select the customer360: customer360 data source.
3. In the View As section in the top right, click the chart icon drop-down and select Scatter.



1. Drag Qty Ordered to X-Axis.
2. Drag Gross Profit to Y-Axis.
3. Drag Product name to Points. Analyzer returns a scatterplot view of the products.
4. Drag Profit range to Color By.

Profit range was calculated by PDI upon loading the POS data into HBase. This calculation can now be used in visualizations and reports to improve insight.

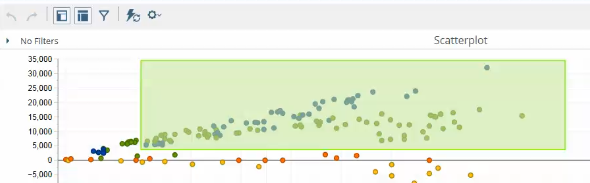


1. Click the Save as button () and save as Scatterplot in the following PUC directory: /home/[PUC Username].

**Analyzer Exercise 2: Create a high value product list report**

This exercise has you lasso-select the high-value products and then create a top 50 product list based on products with the highest gross profit.

1. Open the Scatterplot analysis (if not already open).
2. Lasso-select the points greater than 80 Qty Ordered and 5000 Gross Profit.



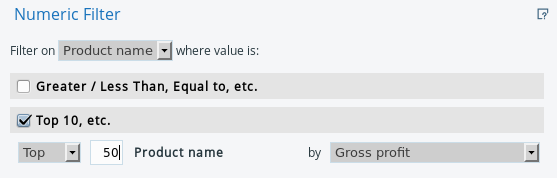
1. Select Keep Only from the pop-up menu.



1. From the View As menu, click the grid icon to return to a table view



1. Right-click Product name and select Top 10, etc.
2. In the Numeric Filter dialog box, change Top 10 to 50 and choose Gross profit in the by field.

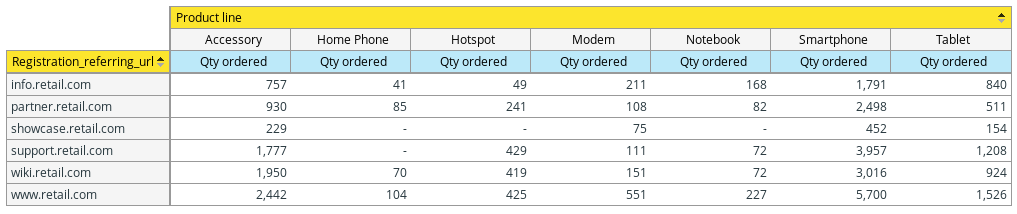


1. Click OK.
2. From the toolbar, click the more actions icon  and select Export **🡪** To CSV.
3. When prompted, accept the defaults and click the Export button.
4. Choose OK to open the file in Gnumeric, and then save the file to your local disk.
5. Return to PUC and click the Save as button in PUC () and save as Top 50 Products in the following PUC directory: /home/[PUC Username].

**Analyzer Exercise 3: Create a linear trend line chart for gross profit**

In this exercise you create a linear trend line to trend Gross profit across Order date. After trending Gross profit, you will then create a multi-chart for each Language.

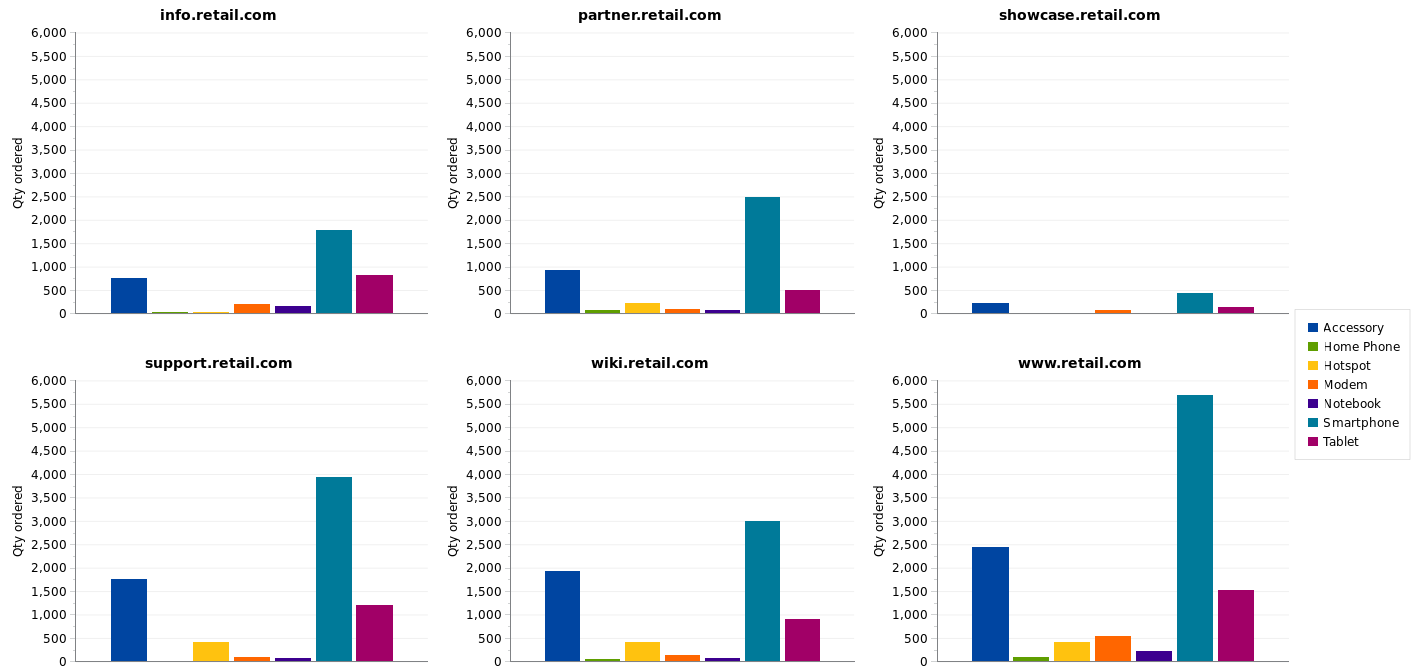
1. Click on the **Create New | Analysis Report** buttons.
2. Select the customer360 data source.
3. Drag Registration referring url to Rows.
4. Drag Qty Ordered to Measures.
5. Drag Product line to Columns. Analyzer returns a crosstab view of the Qty Ordered as shown in the following image:



1. In the View As section in the top right, click the chart icon drop-down and select Column.



1. In the Layout section, drag Registration referring url from X-axis to Multi-Chart. The result is a multi-chart by product line for each referring site as shown in the following screenshot.



1. Click the Save as button () and save as Multi-Chart by Line in the following directory: /home/[PUC Username].

You have reached the end of the Customer 360 Use Case Workshop and completed the entire Pentaho Big Data Integration Workshop! From all of us at Pentaho, we congratulate you on job well done!

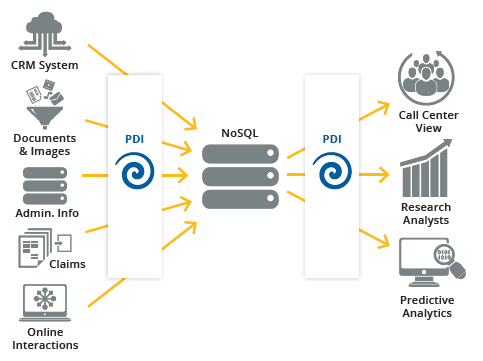
MongoDB Customer 360 Use Case

What is it?

* Blending a variety of operational & transactional data sources to create an on-demand analytical view across customer touch points and surface the customer experience.
* Providing customer-facing employees and partners with information made available inside everyday line-of-business applications
* Leverages NoSQL database technologies like MongoDB that provide flexible storage options for single views of data.

Why do it?

* Provide single customer view to sales & services teams, empowering them to grow upsell/cross-sell revenue
* Understand customer perception of your products & services, while decreasing customer churn
* Avoid point-to-point integrations with single repository, and fast queries to improve access to metrics



Value of Pentaho

* Staff savings & productivity: Rapid time to value through drag/drop visual development for big data integration – make big data accessible to all data developers
* Operational Intelligence: Ability to embed analytics into actionable line-of-business applications for each relevant customer-facing role
* Broad & robust data connectivity: Ability to blend traditional sources & big data
* Comprehensive analytics: Visualizations, reports, dashboards, ad hoc analysis provide for all roles

Pentaho components used in this workshop

Pentaho Data Integration (PDI)

Analyzer For Mongodb

Pentaho Report Designer (PRD)

What you will accomplish in this workshop

* Part 1 – Use PDI to create a single view in MongoDB (4 exercises)
* Part 2 – Visualize data with Analyzer for MongoDB (3 exercises)
* Part 3 – Pentaho Report Designer (4 exercises)

Part 1: Use PDI to create a single view in MongoDB

PDI gives users a graphical user interface to build transformations and jobs that solve complex data integration challenges with MongoDB. PDI provides a scalable solution for migrating to MongoDB with connectors for all types of disparate data for building out a single customer view. PDI leverages the power of the MongoDB aggregation framework query language for aggregating and processing MongoDB data.

Use PDI to create a MongoDB customer data store

A new MongoDB collection, customer360, will store customer master data, point-of-sale (POS) transactions, and website clickstream event data. The customer master records are stored as documents, while the POS transactions and event transactions are stored as data arrays within each customer document.

PDI Exercise 1: Create a transformation to load customer records to MongoDB

This first exercise steps you through the process of creating a transformation to load customer master records from a CSV file, calculate the customer age, and then load the results into a MongoDB collection.

1. If PDI is not already open, launch it from the launch menu icon  at the bottom of your screen. Click this icon just *once* to launch Spoon, the client-based authoring GUI for PDI.



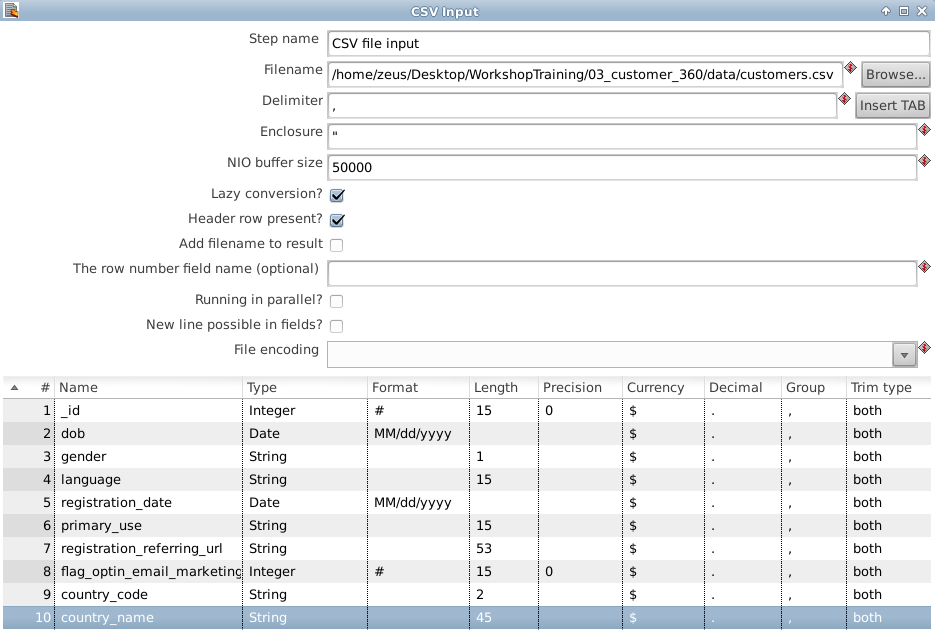
1. You will see the PDI splash screen appear while PDI loads. All open applications appear in the top left section of your screen. You can see Spoon is open in the following screenshot.



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

We need to access Customer data from within a flat file. The file format is a csv file, so you will access the data by configuring a CSV file input step.

1. From the Design tab on the left, expand the Input folder and drag CSV file input onto the canvas
2. Double-click on CSV file input to open its properties
3. Browse to the directory /pentaho/shared\_content/WorkshopTraining/ 03\_customer\_360\_mongodb/data/ and select customers.csv.
4. Click the Get Fields button at the bottom and enter 50,000 for the Sample Size.
5. Once the scan is finished, click Close to close the scan results.
6. Trim any possible source data character spaces by setting Trim Type to both for every field.
7. Click the Preview button to preview the data and then click Close. Your CSV Input dialog box should match the following image:



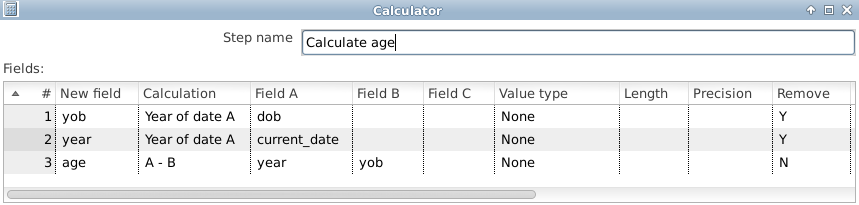
1. Click OK to return to the canvas.

To calculate the age of the customer, we need to know the current year and the year in which the customer was born. But to calculate the current year, we need to know the current date. We will use the **Get System Info** step to learn the current date.

1. From the Design tab on the left, expand the Input folder and drag Get System Info onto the canvas.
2. Double-click on Get System Info step to open its properties.
3. In the Name field type current\_date
4. In the Type field, select from the drop-down menu, system date (variable)
5. Click OK to return to the canvas.
6. To draw a hop between two steps, shift-click the CSV file input step and while holding down your mouse key, drag a **hop** over to the Get System Info step. When prompted, select Main output of step.

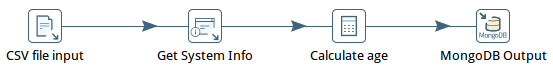
Now that we have the current date, we need to derive the current year from the date. And we need to know the year in which the customer was born, which is based on the customer date of birth. With both years in hand, we can calculate the age of the customer. All of these calculations can be performed with the **Calculator** step.

1. From the Design tab on the left, expand the Transform folder and drag the Calculator step onto the canvas.
2. Draw a hop between the Get System Info and Calculator steps.
3. Double-click on Calculator step to open its properties.
4. In the Step name box type Calculate age
5. Follow these steps to create the first year of birth calculation: In the New Field column type yob, in the Calculation column select Year of date A, in the Field A column select the dob field, and finally in the Remove column select Y.
6. Follow these steps to create the second year calculation: In the New Field column type year, in the Calculation column select Year of date A, in the Field A column select the current\_date field, and finally in the Remove column select Y.
7. Follow these steps to create the third customer age calculation: In the New Field column type age, in the Calculation column select A - B, in the Field A column select the year field, in the Field B column select the yob field, and finally in the Remove column, make sure it is set to N.
8. Once your Calculate age step matches the following image, click OK to return to the canvas.

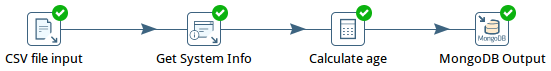


Customer data has been enriched with age information, and we are ready to write the data to a MongoDB collection.

1. From the Design tab on the left, expand the Big Data folder and drag MongoDB Output onto the canvas.
2. Draw a hop between the Calculate age and MongoDB Output steps. The resulting transformation should match the following image:



1. Double-click on MongoDB Output step to open its properties.
2. On the Output Options tab, type customer360 in the Database and Collection entry boxes.
3. Check Truncate collection.
4. On the Mongo document fields tab click the Get fields button at the bottom to populate the field names.
5. Click OK to return to the main canvas.
6. Click the save icon in the toolbar to save your new transformation as t\_load\_customers in the following directory: /pentaho/shared\_content/WorkshopTraining/student\_files/03\_customer\_360
7. Click OK to return to the main canvas.
8. Run your transformation by either clicking  or choosing Action| Run from the main menu.
9. Click the Launch button at the bottom of the Execute a transformation dialog box. Your transformation has run successfully when you see the green checkmark boxes on each step as shown in the following image. This transformation will load 19,673 customers into the newly created MongoDB collection.



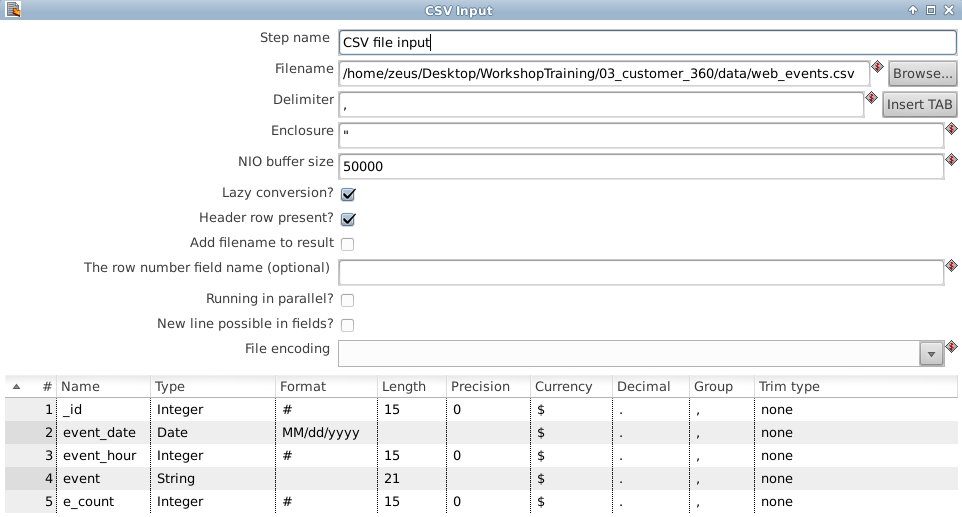
PDI Exercise 2: Create a transformation to load web events to MongoDB

This second exercise steps you through the process of creating a transformation to load the second source of data, customer web clickstream (web event) records, from a CSV file into a MongoDB collection. In this MongoDB output step, you declare the customer “\_id” as the lookup key and load the clickstream records to data arrays for each customer.

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

To blend customer web events data from a CSV file into the MongoDB data collection, first we need to configure a CSV file input step to access the web events.

1. From the Design tab on the left, expand the Input folder and drag CSV file input onto the canvas**.**
2. Double-click on CSV file input to open its properties
3. Browse to the directory /pentaho/shared\_content/WorkshopTraining/ 03\_customer\_360\_mongodb/data/ and select web\_events.csv.
4. Click the Get Fields button at the bottom and enter 50,000 for the Sample Size.
5. Click Close to close the scan results.
6. Click the Preview button to preview the data and then click Close. Your CSV Input dialog box match the following image:



1. Click OK to return to the canvas.

The customer web events records should be appended to the appropriate customer records already loaded the MongoDB collection. We will use customer “\_id” as the lookup key to join the web events records to data arrays we will name “event\_data []” for each customer.

1. From the Design tab on the left, collapse the Input folder and expand the Big Data folder; then, select and drag MongoDB Output onto the canvas.
2. Shift-click the CSV file input step and while holding down your mouse key, drag a **hop** over to the MongoDB Output step. When prompted, select Main output of step.



1. Double-click on MongoDB Output step to open its properties.
2. On the Output Options tab, click the Get DBs button and select customer360 from the list. Click the Get collections button and select customer360 from the list.
3. IMPORTANT - Check Update, Upsert and Modifier Update.
4. On the Mongo document fields tab click the Get fields button at the bottom to populate the field names and then enter the following values in the Mongo document path column.

Row #1 - \_id

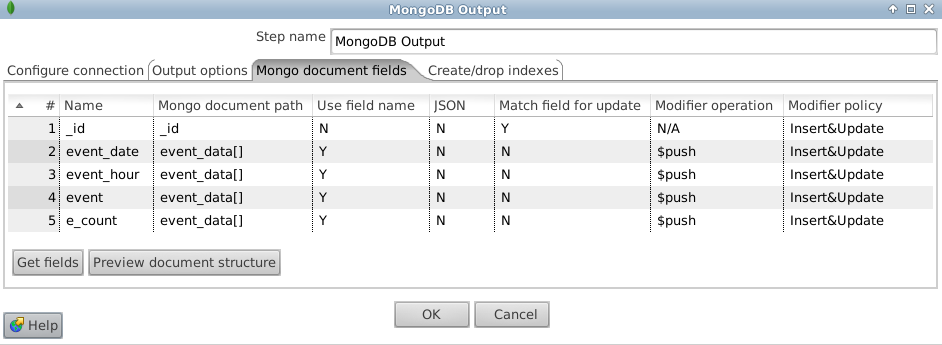
Row #2 – event\_data[]

Row #3 – event\_data[]

Row #4 – event\_data[]

Row #5 – event\_data[]

1. On the first row change Use field name to N and Match field for update to Y.
2. On rows 2 thru 5, change Modifier operation to $push.
3. For every row change Modifier policy to Insert&Update.
4. Your Mongo document fields should match the following screenshot.



1. Click Preview document structure to validate the settings.
2. Click OK to return to the main canvas.
3. Click the save icon in the toolbar to save your new transformation as t\_load\_web\_events in the following directory: /pentaho/shared\_content/WorkshopTraining/student\_files/03\_customer\_360
4. Run your transformation by either clicking  or choosing Action | Run from the main menu. In the Step Metrics tab you should see the execution results from loading 51,419 clickstream records into MongoDB.

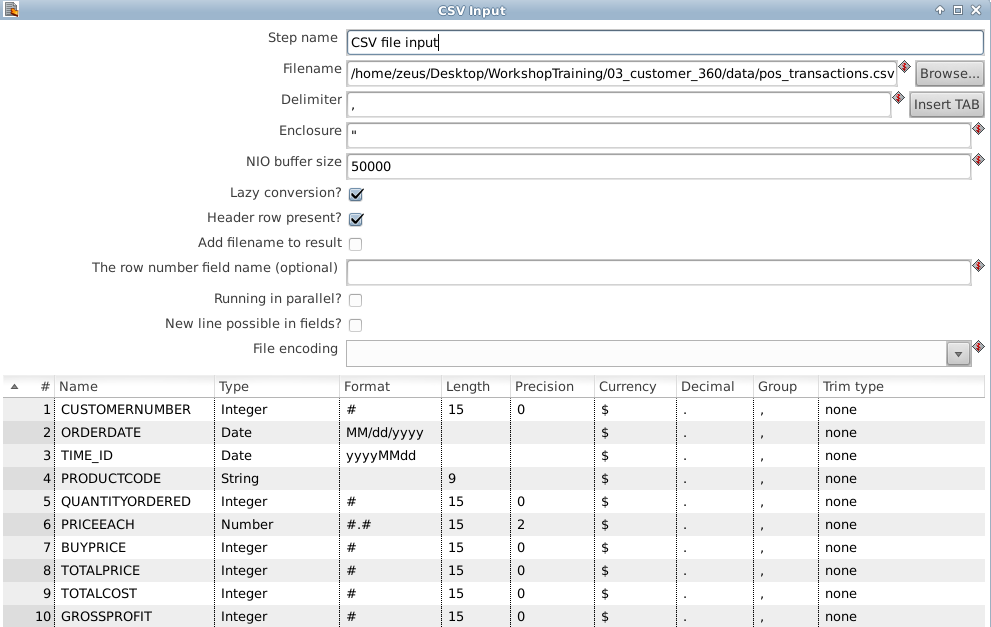
PDI Exercise 3: Create a transformation to load POS transactions to MongoDB

This third exercise steps you through the process of creating a transformation to load the third and final data source, customer point-of-sale (POS) sale transaction records, from a CSV file into a MongoDB collection. You use the number range step to bin sales orders into buckets based on the size of the order. In this MongoDB output step, you declare the customer “\_id” as the lookup key and load the POS records to POS data arrays for each customer.

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

To blend point-of-sale transactions from a CSV file into the MongoDB data collection, we need to configure a CSV file input step to access the transactions.

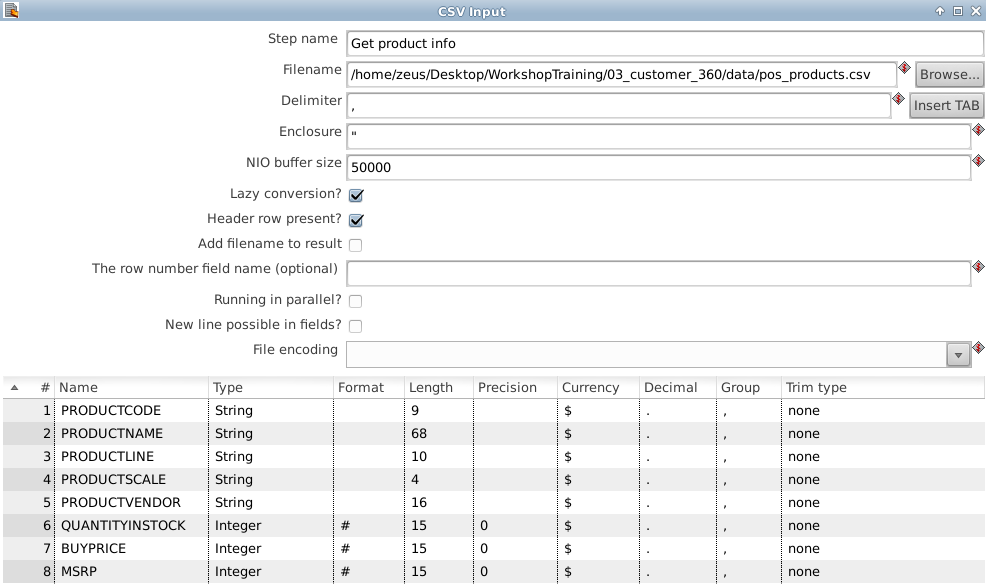
1. From the Design tab on the left, expand the Input folder and drag CSV file input onto the canvas.
2. Double-click on CSV file input to open its properties
3. Browse to the directory /pentaho/shared\_content/WorkshopTraining/ 03\_customer\_360\_mongodb/data/ and select pos\_transactions.csv.
4. Click the Get Fields button at the bottom and enter 5,000 for the Sample Size.
5. Click the Preview button to preview the data and then click Close. Your CSV Input dialog box should look like the following image:



1. Click OK to return to the canvas.

The POS file only has product code on the transaction. We need to enhance the data with Product Name, Product Line, and Product Vendor. To do so, we will look up the product information in a flat file to match the transaction product code with its relevant name, line, and vendor.

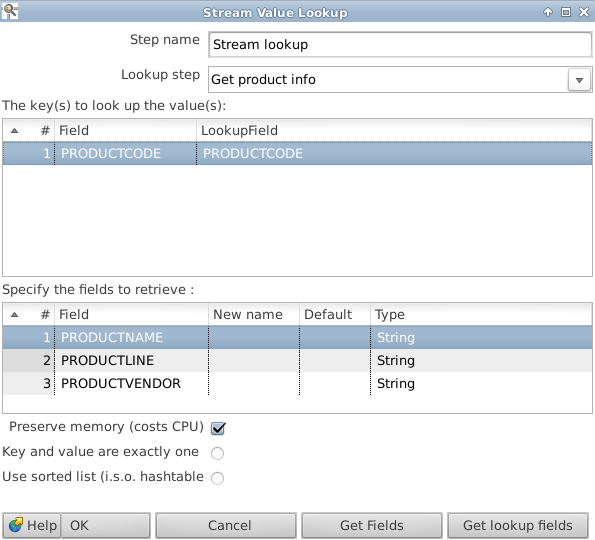
1. From the Design tab on the left, expand the Lookup folder and drag Stream lookup onto the canvas.
2. Create a hop between the CSV file input and Stream lookup steps.
3. From the Design tab on the left, expand the Input folder and drag a second CSV file input onto the canvas and place it above the Stream lookup step.
4. Double-click on CSV file input to open its properties
5. Change the Step name to Get product info
6. Browse to the directory /pentaho/shared\_content/WorkshopTraining/ 03\_customer\_360\_mongodb/data/ and select pos\_products.csv.
7. Click the Get Fields button at the bottom and enter 5,000 for the Sample Size.
8. Click the Preview button to preview the data and then click Close. Your CSV Input dialog box should look like the following image:



1. Click OK to return to the canvas
2. Create a hop from the second Get product info step to the Stream lookup step.

We need to configure the Stream Lookup to retrieve Product Name, Product Line, and Product Vendor from the lookup file for every product code we find in the POS records.

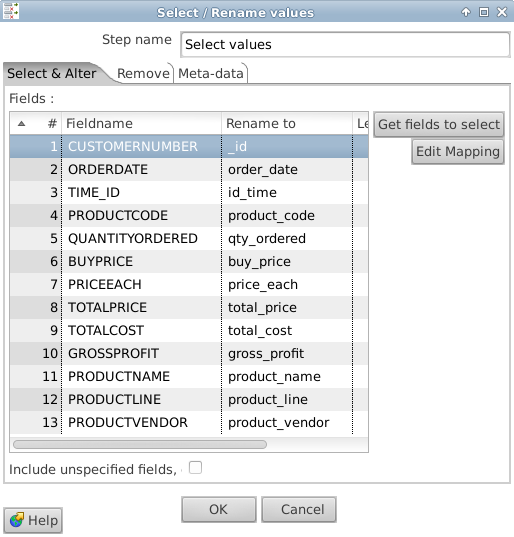
1. Double-click on Stream lookup step to open its properties
2. In the **Lookup** step select Get product info.
3. In the **Key(s) to Lookup Value(s)** section select PRODUCTCODE in the **Field** column and select PRODUCTCODE as the **LookupField** column.
4. Click the **Get Lookup Fields** button to populate the fields to retrieve section at the bottom.
5. Highlight and delete the following fields: PRODUCTCODE, PRODUCTSCALE, QUANTITYINSTOCK, BUYPRICE, MSRP from the fields to retrieve section. Your Stream Lookup dialog box should now match the following image:



1. Click OK to return to the canvas.

CUSTOMERNUMBER is the same as customer “\_id” within the MongoDB collection. We prefer to rename this field and all the other fields to match and standardize on a lowercase naming convention used in the MongoDB collection target.

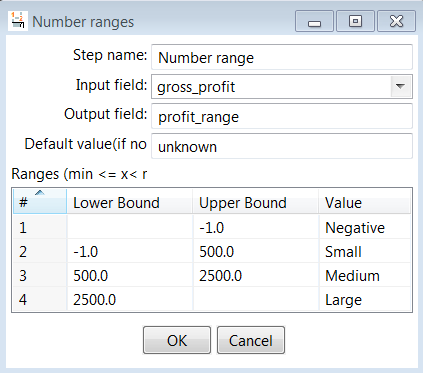
1. From the Design tab on the left, expand the Transform folder and drag Select Values onto the canvas.
2. Create a hop between the Stream lookup and Select values steps.
3. Double-click the Select Values step. On the Select & Alter tab, click Get fields to select to add all the available stream fields.
4. To rename CUSTOMERNUMBER, click inside the Rename to column and enter \_id.
5. Rename all the remaining fields to standardize on a lowercase naming convention and match the other data sets. The result should match the following image:



1. Click OK to return to the main canvas.

 It may be useful to categorize the sale orders into buckets for further analysis, based on order size. We can create a categorization of small, medium, and large.

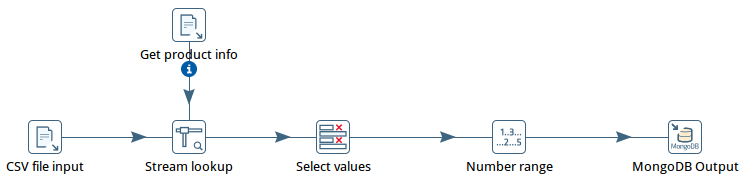
1. From within the Transform folder, select and drag Number Range onto the canvas.
2. Create a hop between the Select values and Number Range steps.
3. Double-click on Number Range step to open its properties.
4. For Input field select gross\_profit from the list.
5. For Output field type profit\_range.
6. Enter the following values for Lower Bound, Upper Bound and Value columns.



1. Click OK to return to the canvas.

The point-of-sale records should be appended to the appropriate customer records already loaded to the MongoDB collection. We will use customer “\_id” as the lookup key to join the POS records to data arrays we will name “pos\_data []” for each customer.

1. From the Design tab on the left, expand the Big Data folder and drag MongoDB Output onto the canvas.
2. Create a hop between the Number Range and MongoDB Output step. The resulting transformation should match the following image:



1. Double-click on MongoDB Output step to open its properties.
2. On the Output Options tab, click the Get DBs button and select customer360 from the list. Click the Get collections button and select customer360 from the list.
3. IMPORTANT - Check Update, Upsert and Modifier Update.
4. On the Mongo document fields tab click the Get fields button at the bottom to populate the field names.
5. Click OK to return to the main canvas.

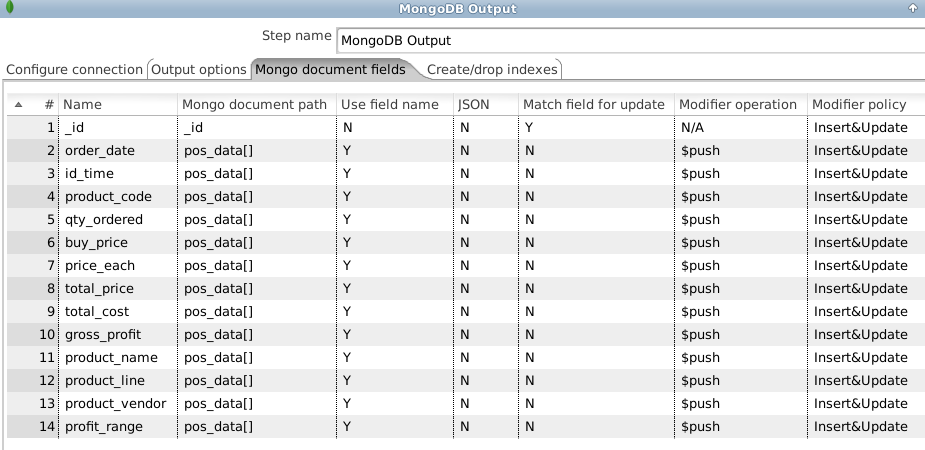
By clicking OK the dialog will register your changes and set default field values to be used in subsequent steps.

1. Double-click the MongoDB Output step again, and on the Mongo document fields tab, enter the following values in the Mongo document path column.

Row #1 - \_id

Row #2-#14 – pos\_data[]

1. On the first row change Use field name to N and Match field for update to Y.
2. On rows 2 thru 14, change Modifier operation to $push. You can use the right-click option on the mouse button to copy/paste values.
3. For every row change Modifier policy to Insert&Update.
4. Your Mongo document fields should match the following screenshot.



1. Click Preview document structure to validate the settings.
2. Click OK to return to the main canvas.
3. Click the save icon in the toolbar to save your new transformation as t\_load\_pos in the following directory:

/pentaho/shared\_content/WorkshopTraining/student\_files/03\_customer\_360

1. Run your transformation by either clicking  or choosing Action | Run from the main menu. In the Step Metrics tab you should see the execution results from loading 991 POS records into MongoDB.

PDI Exercise 4: Create a job to sequence and automate 3 transformations

In this exercise you build a PDI job used to execute all three transformations created in the previous three exercises. PDI jobs can be scheduled for automated processing on a recurring schedule.

1. From the main menu choose File | New | Job.
2. From the Design tab on the left, expand the General folder; then, select and drag Start onto the canvas.
3. While in the General folder, add three Transformation steps and a Success step to the canvas.
4. Connect each job step with a hop in the following order: START | Transformation | Transformation 2 | Transformation 3 | Success.



1. Double-click the first Transformation job step to edit the job entry details.
2. Change the Name of job entry to Load Customers.
3. For the Transformation filename section specify the transformation to run by browsing to /pentaho/shared\_content/WorkshopTraining/student\_files/03\_customer\_360 and choosing t\_load\_customers.ktr.
4. Click OK to return to the job canvas.
5. Double-click the Transformation 2 job step to edit the job entry details.
6. Change the Name of job entry to Load Web Events.
7. For the **Transformation filename** section specify the transformation to run by browsing to /pentaho/shared\_content/WorkshopTraining/student\_files/03\_customer\_360 and choosing t\_load\_web\_events.ktr.
8. Click OK to return to the job canvas.
9. Double-click the Transformation 3 job step to edit the job entry details.
10. Change the Name of job entry to Load POS.
11. For the **Transformation filename** section specify the transformation to run by browsing to /pentaho/shared\_content/WorkshopTraining/student\_files/03\_customer\_360 and choosing t\_load\_pos.ktr.
12. Click OK to return to the job canvas. Your job should match the following screenshot.



1. Use your mouse to lasso-select all five job steps and then right-click any step.
2. From the right-click menu select Align / Distribute and then Snap to grid.
3. Click the save icon in the toolbar to save your new job as j\_load\_mongo360 in the following directory: /pentaho/shared\_content/WorkshopTraining/student\_files/03\_customer\_360.
4. Run your job by either clicking  or choosing Action | Run from the main menu. When the job completes without errors, green check marks will appear on each job step.

Part 2: Visualize Data with Analyzer for MongoDB

You now have a customer360 collection containing customer master data, point-of-sale transactions, and website clickstream event data. Pentaho Analyzer provides multi-dimensional analysis a single MongoDB collection. The following exercises use Analyzer for MongoDB to analyze and visualize the data in the customer360 collection.

Analyzer Exercise 1: Create a scatterplot visualization

This exercise has you create a scatterplot displaying individual products across two measures and then color each point in the scatterplot by the newly calculated field, Profit\_range.

Note: The BA server should already be started from the previous exercises, but if you need to start it, execute the following script: /pentaho/current\_version/ctlscript.sh start.

1. With your Firefox browser navigate to <http://localhost:8081/pentaho/Login> to launch the Pentaho User Console (PUC).
2. Login to PUC

See Pentaho User Console credentials section in the document on the Desktop in the Docs folder:

* 1. hds - BDI Workshop Credentials.pdf

-or-

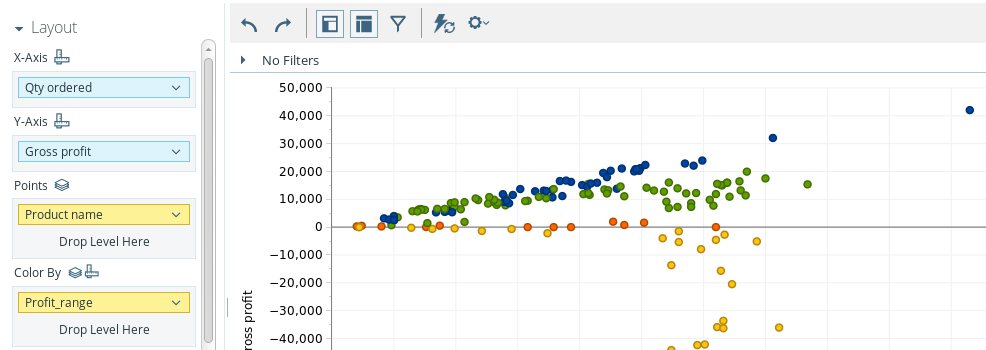
* 1. bdiw - BDI Workshop Credentials.pdf

1. Click on the **Create New | Analysis Report** buttons.
2. Select the Customer360Mongo: Customer360PoS data source.
3. In the View As section in the top right, click the chart icon drop-down and select Scatter.



1. Drag Qty ordered to X-Axis.
2. Drag Gross profit to Y-Axis.
3. Drag Product name to Points. Analyzer returns a scatterplot view of the products.
4. Drag Profit\_range to Color By.

Profit range was calculated by PDI upon loading the POS data into MongoDB. This calculation can now be used in visualizations and reports to improve insight.

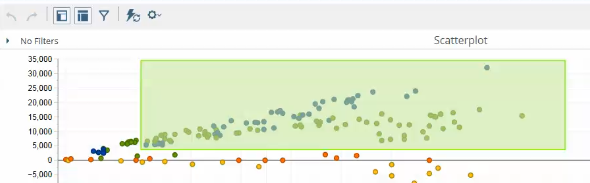


1. Click the Save as button () and save as Scatterplot in the following PUC directory: /home/[PUC Username].

**Analyzer Exercise 2: Create a high value product list report**

This exercise has you lasso-select the high-value products and then create a top 50 product list based on products with the highest gross profit.

1. Open the Scatterplot analysis (if not already open).
2. Lasso-select the points greater than 80 Qty ordered and 5000 Gross profit.



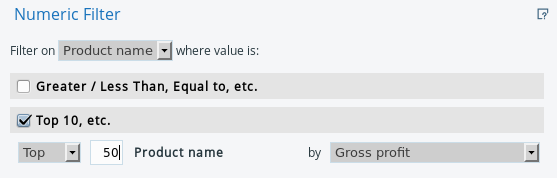
1. Select Keep Only from the pop-up menu.



1. From the View As menu, click the grid icon to return to a table view



1. Right-click Product name and select Top 10, etc.
2. In the Numeric Filter dialog box, change Top 10 to 50 and choose Gross profit in the by field.

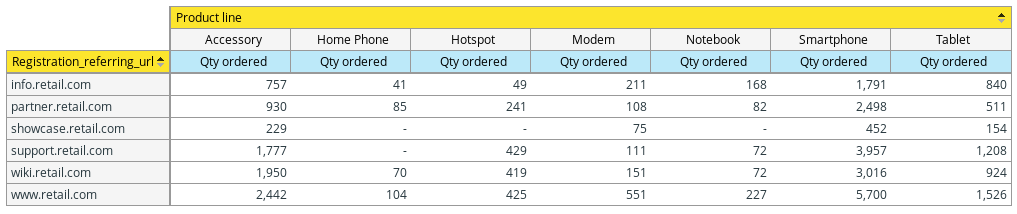


1. Click OK.
2. From the toolbar, click the more actions icon  and select Export **🡪** To CSV.
3. When prompted, accept the defaults and click the Export button.
4. Choose OK to open the file in Gnumeric, and then save the file to your local disk.
5. Return to PUC and click the Save as button in PUC () and save as Top 50 Products in the following PUC directory: /home/[PUC Username].

**Analyzer Exercise 3: Create a linear trend line chart for gross profit**

In this exercise you create a linear trend line to trend Gross profit across Order date. After trending Gross profit, you will then create a multi-chart for each Language.

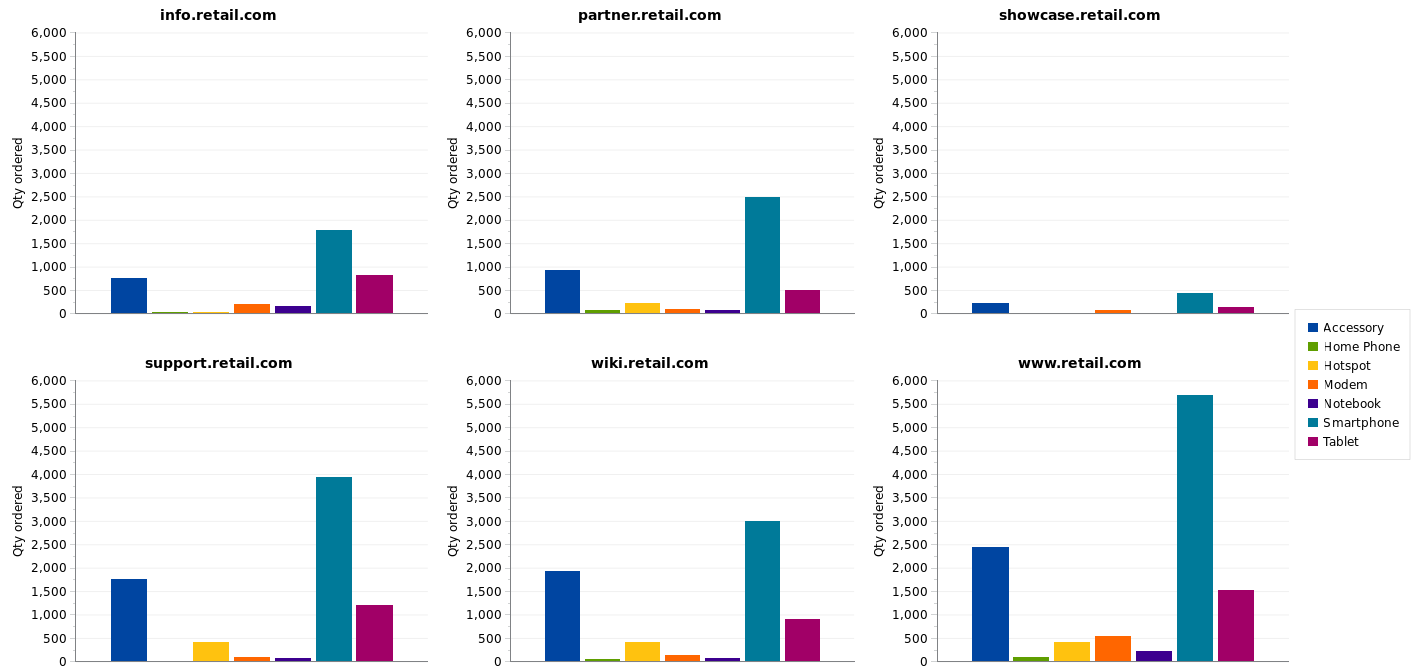
1. Click on the **Create New | Analysis Report** buttons.
2. Select the Customer360PoS data source.
3. Drag Registration\_referring\_url to Rows.
4. Drag Qty ordered to Measures.
5. Drag Product line to Columns. Analyzer returns a crosstab view of the Qty ordered as shown in the following image:



1. In the View As section in the top right, click the chart icon drop-down and select Column.



1. In the Layout section, drag Registration\_referring\_url from X-axis to Multi-Chart. The result is a multi-chart by product line for each referring site as shown in the following screenshot.



1. Click the Save as button () and save as Multi-Chart by Line in the following directory: /home/[PUC Username].

Part 3: Pentaho Report Designer (PRD)

Pentaho Report Designer (PRD) gives users a graphical user interface to build highly formatted, multi-page production reports. PDI provides native connectivity to MongoDB and allows users to construct, parameterize, and execute multiple MongoDB queries in a single report.

**PRD Exercise 1: Add a MongoDB web metrics query**

In this exercise you create a query to unwind the event\_data array and count the number of website Visits, Purchases and Responses. This query will be used to populate the summary metrics at the top of the report page. A PRD report template is included with the data files to give you a head start developing a PRD report. You will need to launch PRD from the Pentaho applications menu and open this report template in PRD to begin.

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



1. You will see the PRD splash screen appear while PRD loads. All open applications appear in the top left section of your screen. You can see PRD is open in the following screenshot.



1. Once PRD opens you are presented with a Welcome startup box. Close the Welcome box.
2. Click open icon on the menu toolbar or choose File🡪Open from the main menu, browse to /pentaho/shared\_content/WorkshopTraining/03\_customer\_360\_mongodb/reports and open report\_template.prpt.
3. Click the Data tab in the top right of your screen, right click Data Sets and select MongoDB from the list.
4. While in the MonogDB Data Source dialog box, click the green circle with a plus symbol to add a new query.
5. Rename Query 1 to web\_metrics\_summary.
6. Complete the four data source tabs starting with the Configure connection tab. Type localhost for Host name and 27017 for Port.
7. On the Input Options tab, click Get DBs and select customer360 from the list of databases.
8. Click Get collections and select customer360 from the list of collections.
9. On the Query tab, type or paste the following query into the Query expressioninput box. This query is located in the code files provided (/pentaho/shared\_content/WorkshopTraining/03\_customer\_360\_mongodb/reports/code.txt) so that you can copy and paste the code.

{ $unwind : "$event\_data" },

{ $group : {

\_id: 1,

Visits : {$sum: { $cond: [ { $eq: [ "$event\_data.event", "Visit" ]},1,0]}},

Purchases : {$sum: { $cond: [ { $eq: [ "$event\_data.event", "Purchase" ]},1,0]}},

Responses : {$sum: { $cond: [ { $eq: [ "$event\_data.event", "Responded to Offer" ]},1,0]}},

}}}

1. *Be sure* and check Query is aggregation pipeline.
2. On the Fields tab, click Get fields. Pentaho will scan the customer360 collection to detect the collection schema and parse the available document fields.
3. Click the Previewbutton to run the query and view the results.

**PRD Exercise 2: Add a MongoDB POS query**

In this exercise you create a query to unwind the pos\_data array and count the number of orders within each profit\_range. This query will be used to populate the summary metrics at the top of the report page.

1. While in the MonogDB Data Source dialog box, click the green circle with a plus symbol to add a new query.



1. Rename Query 2 to pos\_pie\_chart.
2. Complete the four data source tabs starting with the Configure connection tab. Type localhost for Host name and 27017 for Port.
3. On the Input Options tab, click Get DBs and select customer360 from the list of databases.
4. Click Get collections and select customer360 from the list of collections.
5. On the Query tab, type or paste the following query into the Query expressioninput box. This query is located in the code files provided so that you can copy and paste the code.

{ $unwind : "$pos\_data" },

{ $group : { \_id : "$pos\_data.profit\_range", count : { $sum : 1 } } },

{$sort:{count: -1}}

1. *Be sure* and check Query is aggregation pipeline.
2. On the Fields tab, click Get fields. Pentaho will scan the customer360 collection to detect the collection schema and parse the available document fields.
3. In the Name column, rename \_id field to profit\_range.
4. Click the Previewbutton to run the query and view the results.
5. Click OK to return to the workspace.

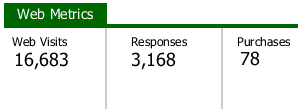
**PRD Exercise 3: Add web metrics fields as report elements**

In this exercise you add the newly created query elements to the report by dragging them from the Data tab to the report canvas and configuring the alignment and font.

1. On the Data tab, right-click the web\_metrics\_summary query and choose Select Query to display the query fields.
2. Click the Visits query element from web\_metrics\_summary query and drag it underneath the Web Visits label on the Details band.
3. Click the Responses query element from the web\_metrics\_summary query and drag it underneath the Responses label on the Details band.
4. Click the Purchases query element from the metrics\_summary\_query and drag it underneath the Purchases label on the Details band.
5. Shift-click all three new number fields in the canvas area to select them together for editing.
6. With all three number-fields highlighted, right-click one of the highlighted fields and select Alignment 🡪 Top.
7. With all three number-fields still highlighted, click the Structure tab, on the right side of the screen, and locate the font-size property in the Style tab below.
8. Change the font to SansSerif and font-size to 18.
9. Click the eye icon on the tab toolbar to preview your report.



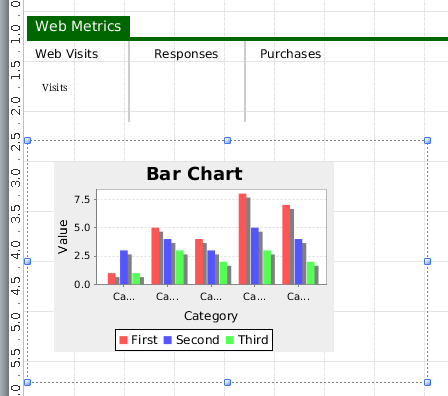
1. Your report should match the following screenshot.



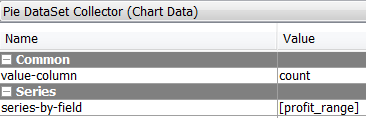
**PRD Exercise 4: Create a pie chart from POS data**

In this exercise you create a new Pie Chart by adding the new POS query elements to a pie chart element and configuring the Pie DataSet Collector properties.

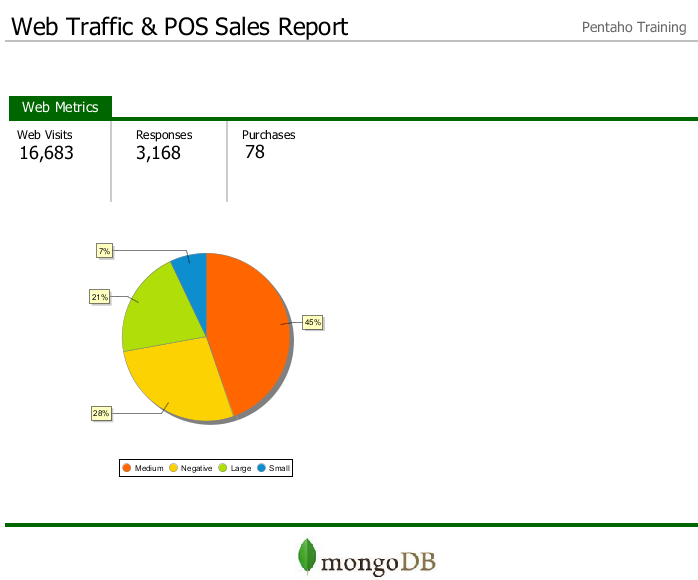
1. Click the sub-report icon  from the Paletteon the left and drag it directly below the Web Visits metric on the Details band.
2. Select Inline when prompted to choose a sub-report type.
3. Select pos\_pie\_chart when prompted by the Select Data Source dialog box.
4. After choosing a query, a new tab, <Untitled Subreport>, will open. Click the chart icon  from the Paletteon the left and drag it onto the new sub-report’s Report Header band.
5. Click the report\_template tab to return to your main report.
6. Resize your sub-report area by clicking and dragging a side of the dotted line box area. You want to resize enough to accommodate the bar chart as shown in the image below:



1. Click the <Untitled Subreport> tab to return to the sub-report canvas.
2. Double-click the Bar Chart element you just added, and Edit Chart dialog box will appear.
3. Select Pie Chart from the chart type menu.
4. The Primary DataSource tab contains two Pie DataSet Collector properties, value-column and series-by-field, to assign to query fields.
5. Select the Value field for value-column**,** and choose the count field from the drop-down menu.
6. Select the Value field for series-by-field and then click the  button to open the Edit Array dialog box.
7. Select profit\_range from the Available Items section and click the add items arrow to add it to the Selected Items section.
8. Click OK and your Pie DataSet Collector properties should match the screenshot below.



1. Click OK again to close the Edit Chart dialog and return to your report.
2. Click the eye icon on the tab toolbar to preview your report with the new pie chart.



1. Click the save icon in the toolbar to save your new report as 360\_operational\_report in the following directory:

/pentaho/shared\_content/WorkshopTraining/student\_files/03\_customer\_360\_mongodb

You have reached the end of the Customer 360 Use Case Workshop and completed the entire Pentaho Big Data Integration Workshop! Sit back and take in the fact that you just completed three big data use cases in one day. From all of us at Pentaho, we congratulate you on job well done!