

Pentaho and Hadoop Framework Fundamentals

Course Code DI2000

Version 7.1

08.24.2017

Student Guide

Table of Contents

Exercise 1: Using the Virtual Exercise Environment.....	4
Exercise 1: Overview.....	4
Exercise 2: Reading and Writing Data with PDI and HDFS.....	10
Exercise 2: Overview.....	10
Exercise 3: Using Pentaho MapReduce	15
Exercise 3: Overview.....	15
Exercise 4: Working with Hive and Impala	28
Exercise 4: Overview.....	28
Exercise 5: Working with HBase	35
Exercise 5: Overview.....	35

DI2000: Pentaho Big Data Fundamentals

Course Description

Duration 2 Days

**Course
Overview**

With growing volumes and varieties of data flowing at increasing speed, organizations need a fast and easy way to harness and gain insight from their big data sources. Pentaho accelerates the realization of value from big data with the most complete solution for big data analytics.

Pentaho provides the right set of tools to each user, all within a tightly coupled data integration and analytics platform that supports the entire big data lifecycle. For IT and developers, Pentaho provides a complete, visual design environment to simplify and accelerate data preparation and modeling. For business users, Pentaho provides visualization and exploration of data. And for data analysts and scientists, Pentaho provides full data discovery, exploration and predictive analytics.

Using a combination of instructor-led presentations and hands-on exercises, this course provides an overview of big data technologies and an overview of the Pentaho tools for both working with big data and for visualizing it. This course helps prepare you for the Pentaho Data Integration Certification Exam.

Audience

This course is intended for technical users who integrate disparate data sources (including big data sources), build/maintain data models for analysis, and manage BI data/metadata, including: Database Developers, Power Users, Technical Business Analysts, BI Solution Architects, Systems Integrators, and Data Scientists.

Prerequisites

Before taking this class, students should complete course DI1000: Pentaho Data Integration or have equivalent field experience with Pentaho Data Integration. Big data knowledge is helpful but not required. Some basic knowledge of the Linux operating system (CentOS/Ubuntu) is required.

Continued on next page

Course Description, Continued

Course Objectives	<p>After completing this course, students are able to:</p> <ul style="list-style-type: none">• Identify the purpose and value of various big data technologies: Hadoop, HDFS, Hive, MapReduce, and NoSQL databases.• Read and write data to/from HDFS using PDI• Use PDI to transform data from one format to another• Write a MapReduce application using PDI transformations.• Use PDI to query data with Hive.• Use PDI to write data to and query data from HBase.• Orchestrate big data jobs in Pentaho Data Integration
--------------------------	---

Exercise 1: Using the Virtual Exercise Environment

Exercise 1: Overview

Introduction All the exercises for this course are completed on a virtual machine image provided by Pentaho. Student should have received instructions for downloading the VM image and a ReadMe document that gives details about the image. Prior to class, students are required to install the VM on their computers and make sure it successfully boots using VMware software described in the ReadMe document.

Allotted Time Exercise 1 Overview
The timing of this exercise is as follows:

Component	Time
Overview	1 minute
Exercise	29 minutes
Total:	30 minutes

Objectives After completing this exercise, you will be able to:

- Login to your virtual machine used in this course
- Describe the virtual machine details and provide an overview of the exercise scenario

VM Image Information The VM image provided by Pentaho is a self-contained training environment which has all of the following components installed:

- Linux (Ubuntu)
- Hadoop (CDH, Hive, HBase, Spark, etc...)
- Pentaho BA Server (this includes PDI/Spoon)

The VM will be hosted online for students to access using an HTML 5 compliant web browser (Latest Chrome Web Browser is recommended).

Exercise 1 Overview

Part I: Connect to your VM

In part I of this exercise you will login into your assigned hosted virtual image used in this course. After logging into the image, you may need to replace the Pentaho license files.

Accessing the Virtual Image

Instructor will provide instructions on how to access the hosted VM. The VM will be accessed using the Chrome web browser. Students will be given a unique login to access their individual VM.


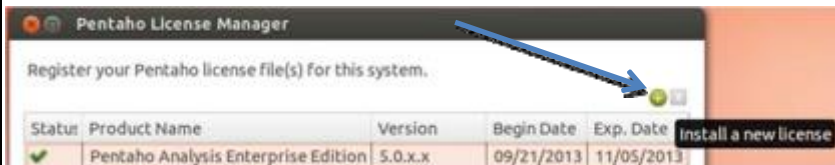
Continued on next page

Exercise 1 Overview, Continued

Configuring Licenses

After logging into the virtual image, you may need to update the license files for the installed trial version of Pentaho. Students taking this course online do not need to complete these steps and may skip to the next section. Students taking the course in a public classroom or on-site may need to complete this section. Your instructor will inform you whether a license update is required.


To update the licenses:

Step	Action
1	Your instructor will inform you of the location of the license files. If they have not been copied to the virtual image, copy them before proceeding.
2	<p>Start PDI from the Launcher by clicking on the PDI icon:</p>  <p>Alternatively, you can start PDI from the command line by starting Terminal from the Launcher, and then type the following command at the command prompt:</p> <pre>cd ~/pentaho/design-tools/data-integration/ ./spoon.sh</pre> <p>(Note: The spoon start script must be run from the directory it is in)</p>
3	<p>If licenses have already expired, then you will be prompted to enter a new license. If license is still active, then from Spoon's menu, select Help --> Pentaho License Manager dialog, click the Install a new license button (green +).</p> 

Continued on next page

Exercise 1 Overview, Continued

Configuring Licenses (continued)

Step	Action
4	Browse to the location of the license files you copied to your system, choose the Pentaho PDI Enterprise Edition.lic license file, and click Open .
5	When the license appears in the license manager, verify the End Date occurs after your training class ends. 
6	Repeat for the remaining license files. Then click Close .
7	Click OK when you receive the message: Licenses have been updated...
8	If you are asked for the Repository Connection window, click Cancel .
9	If you see the Spoon tips dialog, <i>deselect</i> Show tips at startup and click Close .
10	Close Pentaho Data Integration.

Part II: About the Image

Part II of this exercise provides you with an overview of the virtual exercise environment used in this course. Refer to this section as necessary.

Directories

All training material is located in a folder on the Desktop:
/home/zeus/Desktop/di2000/

Pentaho Enterprise Edition is installed in:
/home/zeus/pentaho/

Continued on next page

Exercise 1 Overview, Continued

Using the Launcher

The desktop toolbar provides icons for launching various applications, including PDI.

Note: Location of the folders and icons maybe different.



Exercises

Throughout this course, we will build a word count application through a series of exercises:

- Upload data files to HDFS
- Perform word count on the data files using Pentaho MapReduce
- Store results of word count from HDFS to MongoDB.

The completed lab exercises are stored in subdirectories under the following base folder (they are on the Desktop, the following are the fully qualified path to the folders):

/home/zeus/Desktop/di2000/

Students will store their work in the following folder:

/home/zeus/Desktop/di2000/student

Continued on next page

Exercise 1 Overview, Continued

Spoon Configuration

Each time Spoon (a.k.a. Kettle) is started, it will read the following configuration file: `~/.kettle/kettle.properties`

This file contains all variables that the completed training exercises use for setting various hostnames and directories. Although the exercises that the students perform will not use these variables, the completed solution will use the variables in this properties file. You can view the file by opening it in the text editor.

Starting Processes

In order to conserve memory the Pentaho background processes are not automatically started. Since the majority of the exercises use only Pentaho Data Integration, the background processes are started only when needed. Each exercise will include instructions for starting the Pentaho processes when required.

If needed, use the `ctlscript.sh` command to start all Pentaho processes. This command is located in `/home/zeus/pentaho/`. For information on configuring the Pentaho processes to start on system boot, see the Pentaho Support Site:
<https://support.pentaho.com/hc/en-us/articles/205793589-Start-Pentaho-at-Boot>

Achievements

Having completed this exercise, you are able to:

- Login to your assigned virtual machine image used in this course
- Describe the virtual machine details and provide an overview of the exercise scenario

Your results are verified by your ability to complete the exercise in the allotted time. When you have completed the exercise, your instructor will facilitate a discussion of your results.

Exercise 2: Reading and Writing Data with PDI and HDFS

Exercise 2: Overview

Introduction In this exercise, you will be introduced to Cloudera Manager. You will also learn how upload files to and download files from HDFS from both command line and PDI-Spoon.

Allotted time The timing of this exercise is as follows:

Component	Time
Overview	1 minute
Exercise	29 minutes
Total:	30 minutes

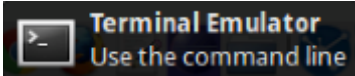
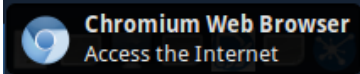
Prerequisites You must complete the prior exercise on configuring the virtual image in order to complete this exercise. You must also have access to support files required by the exercise (if any).

Objectives After completing this exercise, you will be able to:

- Browse HDFS via Hue & NameNode UI
- Navigate HDFS using the command line
- Upload/download files from the local file system to HDFS using the command line.
- Use PDI to create a directory in HDFS
- Upload a file from the local file system to HDFS with PDI

Exercise 2: Reading and Writing Data with PDI and HDFS

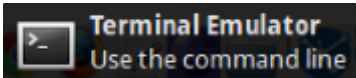
PDI and HDFS Create a HDFS and PDI job to copy data from local disk to HDFS:

Step	Action
1	<p>Note: Cloudera Manager is not installed within this VM. Memory requirements for the VM prevent installing and running Cloudera Manager.</p> <p>Verify the following services are running. Start terminal from the Launcher by clicking on the Terminal icon:</p>  <p>Execute the following command to view the status of services: Hadoop, Hive, YARN, HBase, Spark, ZooKeeper, etc.</p> <pre>sudo service --status-all</pre>
2	<p>Start the Chrome web browser from the Launcher by clicking on the Chromium Web Browser icon:</p> 
3	<p>Navigate to the Hue web interface by typing the following web address in the address bar http://localhost:8888/</p> <p>A link to it is also Bookmarked in the toolbar under Hadoop/Hue.</p> <p>User: zeus Password: zeus</p>

Continued on next page

Exercise 2 Overview, Continued



PDI and HDFS (continued)

Step	Action
4	In Hue on the far right of the menu bar, click on the File Browser . This will take you to a web based HDFS file browser.
5	<p>Start terminal from the Launcher by clicking on the Terminal icon:</p>  <p>Type the following to get a list of all the commands that can be used to operate on HDFS:</p> <pre>hdfs dfs</pre> <p>(Detailed info about HDFS command line can be found here: http://hadoop.apache.org/docs/current/hadoop-project-dist/hadoop-common/FileSystemShell.html)</p>
6	<p>Get a file/directory listing:</p> <pre>hdfs dfs -ls hdfs://pentahovm.localdomain:8020/</pre> <p>Note: Since all the Hadoop framework is running on the provided VM image, you do not have to enter the fully qualified host/port. If you leave off the HDFS NameNode host/port then it is assumed to be <i>localhost</i> and the default port of 8020 (this applies to all the hdfs commands in this section). For example, the following two commands will give the same result:</p> <pre>hdfs dfs -ls hdfs://pentahovm.localdomain:8020/ hdfs dfs -ls /</pre>
7	<p>Create directory in HDFS:</p> <pre>hdfs dfs -mkdir /training/</pre>
8	<p>Upload file to HDFS (one line):</p> <pre>hdfs dfs -copyFromLocal /home/zeus/Desktop/di2000/data/course.txt /training/</pre>

Continued on next page

Exercise 2 Overview, Continued

PDI and HDFS (continued)

Step	Action
9	View the data using the Hue Web UI by going to <i>http://localhost:8888/</i> and click on File Browser.
10	Delete file from HDFS (one line): <code>hdfs dfs -rm /training/course.txt</code>
11	Start PDI from the Launcher by clicking on the PDI icon: 
12	From the menu options, choose File > New > Job .
13	Drag the following job entries onto the canvas and connect them: <ul style="list-style-type: none"> • General > Start • File management > Create a folder • Big Data > Hadoop Copy Files 
14	Edit the Create a Folder step and set the following: <ul style="list-style-type: none"> • Folder name: <code>hdfs://pentahovm.localdomain:8020/training/input</code> • Uncheck Fail if folder exists

Continued on next page

Exercise 2 Overview, Continued

PDI and HDFS (continued)

Step	Action
15	<p>Edit the Copy Hadoop Files step and set the following:</p> <ul style="list-style-type: none"> • Add the following to Files tab: <ul style="list-style-type: none"> – Source Environment: Local – Source File/Folder: <code>file:///home/zeus/Desktop/di_2000/data/</code> – Wildcard (RegExp): <code>sample.*</code> – Destination Environment: CDH – Destination File/Folder: <code>/training/input</code> • Modify the following to Setting tab: <ul style="list-style-type: none"> – Check ‘Copy empty folders’ – Check ‘Replace existing files’ <p>Note: When choosing files, click the icon to browse the file</p>
16	<p>Run the job. When prompted, save the file to: <code>/home/zeus/Desktop/di2000/student/wordcount.kjb</code></p> <p>View the uploaded files in the Hue Web UI by going to http://localhost:8888/ and click on the icon with mouse over of Manage HDFS.</p>

Achievements

Having completed this exercise, you are able to:

- Browse HDFS via Hue & NameNode UI
- Navigate HDFS using the command line
- Upload/download files from the local file system to HDFS using the command line
- Use PDI to create a directory in HDFS
- Upload a file from the local file system to HDFS with PDI

Your results are verified by your ability to complete the exercise in the allotted time. When you have completed the exercise, your instructor will facilitate a discussion of your results.

Exercise 3: Using Pentaho MapReduce

Exercise 3: Overview

Introduction In this lab you will learn how to develop MapReduce applications using Pentaho MapReduce and how to use Hadoop's web interface to monitor Hadoop Jobs.

Allotted time The timing of this exercise is as follows:

Component	Time
Overview	1 minute
Exercise	59 minutes
Total:	1 hour

Prerequisites You must have access to the Pentaho/Cloudera virtual image used for the course exercises. You must also have access to support files required by the exercise (if any).


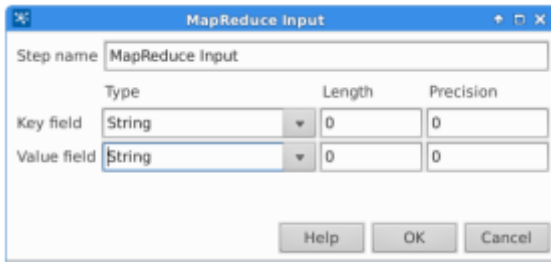
Objectives After completing this exercise, you will be able to:

- Implement Word Count application using Pentaho MapReduce.
- Use Hue web interface to monitor Job information.

Exercise 3: Using Pentaho Map Reduce

Using Pentaho MapReduce

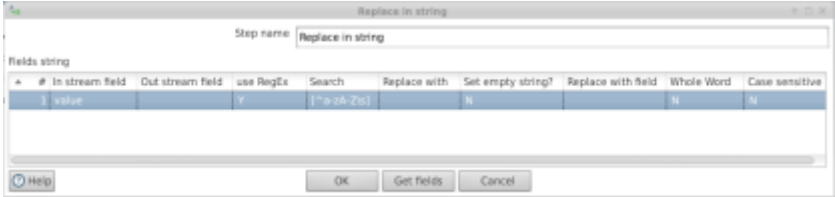

To configure the MapReduce Input and Output steps for a transformation:

Step	Action
1	If necessary, start PDI from the Launcher.
2	Create a new transformation that will be the mapper. From the menu options, choose File > New > Transformation .
3	<p>We will create the mapper transformation. Drag the following transformation steps onto the canvas and connect them:</p> <ul style="list-style-type: none"> • Big Data > MapReduce Input • Transform > Replace in string • Transform > Split field to rows • Transform > Add constants • Big Data > MapReduce Output 
4	<p>Edit the <i>MapReduce Input</i> step:</p> <ul style="list-style-type: none"> • Step name: Hadoop Input • Key Field: String • Value Field: String 

Continued on next page

Exercise 3: Using Pentaho Map Reduce, Continued

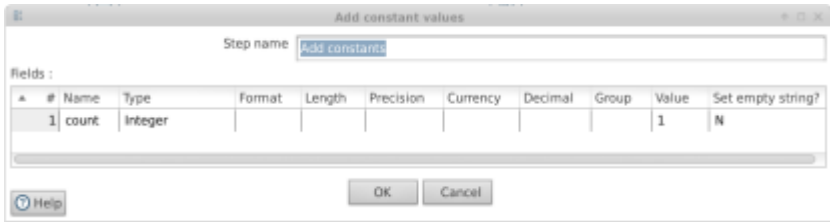
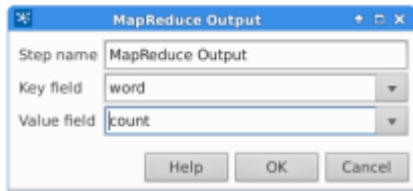

Using Pentaho MapReduce (continued)

Step	Action
5	<p>Edit the <i>Replace in string</i> step to rows step:</p> <ul style="list-style-type: none"> • Field to split: value • Use Regex: Y • Search: [^a-zA-Z/s] • Set Empty String?: N • Whole Word: N • Case sensitive: N 
6	<p>Edit the <i>Split field to rows</i> step:</p> <ul style="list-style-type: none"> • Field to split: value • Delimiter: replace the default semi-colon with a space, not blank/empty • New field name: word 

Continued on next page

Exercise 3: Using Pentaho Map Reduce, Continued

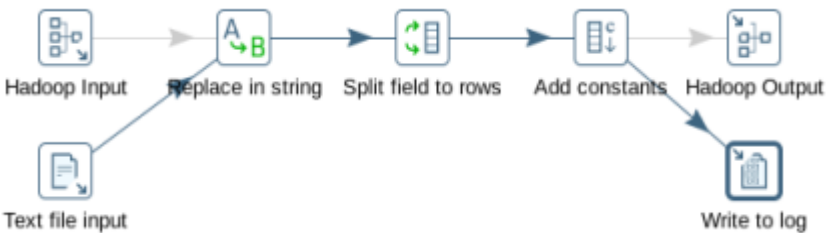
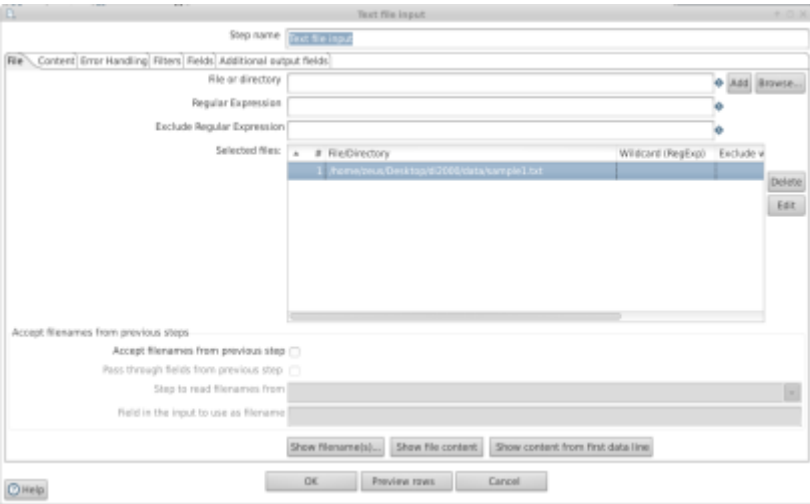
Using Pentaho MapReduce (continued)

Step	Action
7	<p>Edit the <i>Add constants</i> step. For row 1:</p> <ul style="list-style-type: none"> • Name: count • Type: Integer • Value: 1 
8	<p>Edit the <i>MapReduce Output</i> step:</p> <ul style="list-style-type: none"> • Step name: Hadoop Output • Key Field: word • Value Field: count 
9	<p>The mapper transformation should look like this:</p> 
10	<p>Save the transformation to: /home/zeus/Desktop/di2000/student/wc_mapper.ktr</p>

Continued on next page

Exercise 3: Using Pentaho Map Reduce, Continued

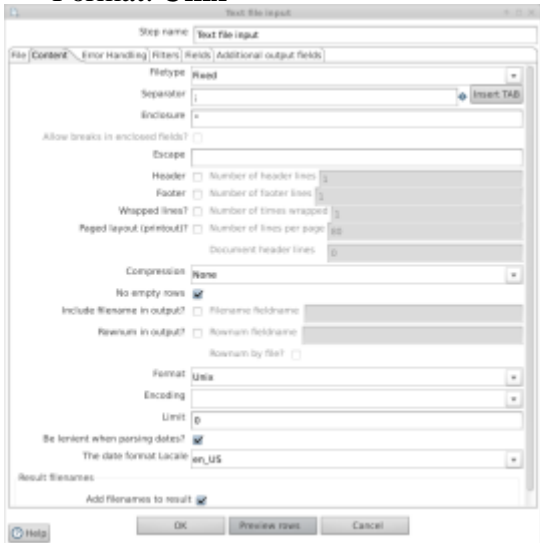

Using Pentaho MapReduce (continued)

Step	Action
11	<p>Test mapper without Hadoop:</p> <ul style="list-style-type: none"> • Disable the hop from <i>Hadoop Input</i> to <i>Split field to rows</i>. • Disable the hop from <i>Add constants</i> to <i>Hadoop output</i>. • Add <i>Text file input</i> step (Input > Text file input) and connect it to <i>Split field to rows</i>. • Add <i>Write to log</i> step (Utility > Write to log) and connect <i>Add constants</i> step to the <i>Write log</i> step. 
12	<p>Edit <i>Text file input</i> step File tab:</p> <p>File or directory: Browse and select the following file: /home/zeus/Desktop/di2000/data/sample1.txt</p> <p>Add the file (click on Add button and it will be added to the Selected files table).</p> 

Continued on next page

Exercise 3: Using Pentaho Map Reduce, Continued


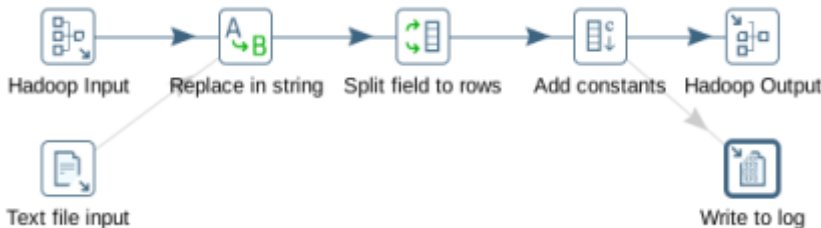

Using Pentaho MapReduce (continued)

Step	Action
13	<p>Edit <i>Text file input</i> step Content tab:</p> <ul style="list-style-type: none"> Filetype: Fixed Header: Unchecked Format: Unix 
14	<p>Edit <i>Text file input</i> step Fields tab:</p> <p>Field #1</p> <ul style="list-style-type: none"> Name: key Type: String Position: 0 Length: 0 <p>Field #2</p> <ul style="list-style-type: none"> Name: value Type: String Position: 0 Length: 100 

Continued on next page

Exercise 3: Using Pentaho Map Reduce, Continued

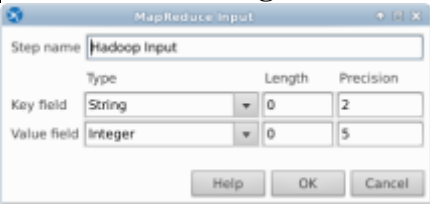

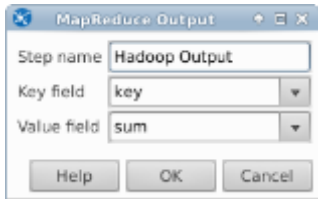
Using Pentaho MapReduce (continued)

Step	Action
15	<p>Edit <i>Write to log</i> step to add the following fields:</p> <ul style="list-style-type: none"> • word • count 
16	Save the transformation and run it. In the Execution Results Logging tab, you should see log output for all words and count.
17	<p>Disable the testing hops and re-enable the Hadoop input/output:</p> <ul style="list-style-type: none"> • Disable hop going from <i>Text file input</i> to <i>Replace in string</i> • Disable hop going from <i>Add constant</i> to <i>Write to log</i> • Enable the hop going from <i>Hadoop Input</i> to <i>Replace in string</i> • Enable hop going from <i>Add constant</i> to <i>Hadoop Output</i>. 
18	Create a new transformation that will be the reducer. From the menu options, choose File > New > Transformation .
19	<p>Drag the following transformation steps onto the canvas and connect them:</p> <ul style="list-style-type: none"> • Big Data > MapReduce Input • Statistics > Group by • Big Data > MapReduce Output 

Continued on next page

Exercise 3: Using Pentaho Map Reduce, Continued



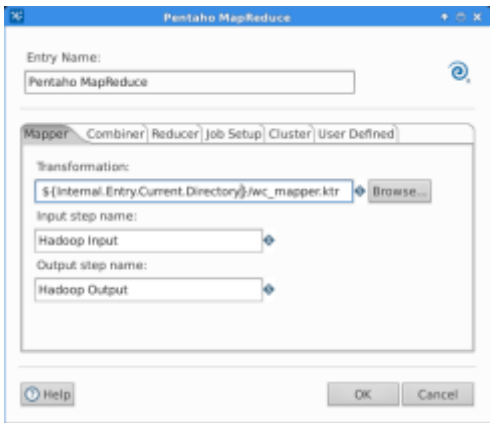
Using Pentaho MapReduce (continued)

Step	Action
20	<p>Edit the MapReduce Input step:</p> <ul style="list-style-type: none"> • Step name: Hadoop Input • Key Field: String • Value Field: Integer 
21	<p>Edit the <i>Group By</i> step:</p> <ul style="list-style-type: none"> • Add group field: key • Add an aggregate: <ul style="list-style-type: none"> – Name: sum – Subject: value – Type: Sum 
22	<p>Edit the MapReduce Output step:</p> <ul style="list-style-type: none"> • Step name: Hadoop Output • Key Field: key • Value Field: sum 

Continued on next page

Exercise 3: Using Pentaho Map Reduce, Continued


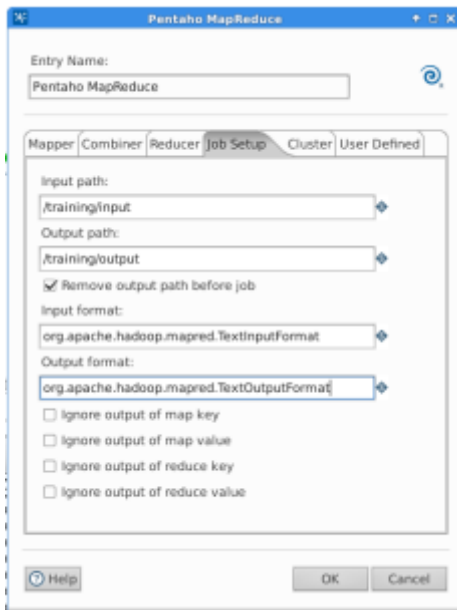
Using Pentaho MapReduce (continued)

Step	Action
23	<p>The reducer transformation should look like this:</p>  <p>Hadoop Input Group by Hadoop Output</p>
24	<p>Save the reducer transformation to: /home/zeus/Desktop/di2000/student/wc_reducer.ktr</p>
25	<p>Create the Pentaho job that executes the Pentaho MapReduce.</p> <ul style="list-style-type: none"> • Open the wordcount.kjb application created in the previous exercise (/home/zeus/Desktop/di2000/student/wordcount.kjb). • Add Pentaho MapReduce step to the job (Big Data > Pentaho MapReduce) and connect the Hadoop Copy Files instance to the Pentaho MapReduce step. The job should look like:  <p>START Create a folder Hadoop Copy Files Pentaho MapReduce</p>
26	<p>Edit the <i>Pentaho MapReduce</i> step:</p> <ul style="list-style-type: none"> • Mapper Transformation: \${Internal.Entry.Current.Directory}/wc_mapper.ktr • Mapper Input Step Name: Hadoop Input • Mapper Output Step Name: Hadoop Output 

Continued on next page

Exercise 3: Using Pentaho Map Reduce, Continued

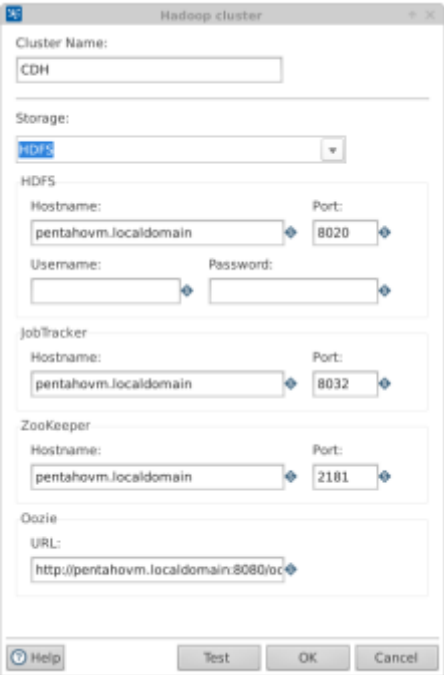
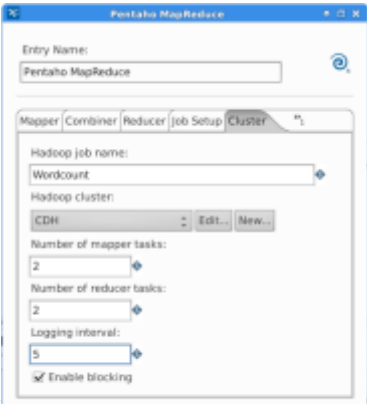
Using Pentaho MapReduce (continued)

Step	Action
27	<p>Edit the <i>Pentaho MapReduce</i> step, Reducer tab:</p> <ul style="list-style-type: none"> Reducer Transformation: <code>\${ Internal.Entry.Current.Directory }/wc_reducer.ktr</code> Reducer Input Step Name: Hadoop Input Reducer Output Step Name: Hadoop Output Reducer single threaded: unchecked 
28	<p>Edit the <i>Pentaho MapReduce</i> step, Job Setup tab:</p> <ul style="list-style-type: none"> Input Path: <code>/training/input</code> Output Path: <code>/training/output</code> Input Format: org.apache.hadoop.mapred.TextInputFormat Output Format: org.apache.hadoop.mapred.TextOutputFormat Remove output path before job: checked 

Continued on next page

Exercise 3: Using Pentaho Map Reduce, Continued

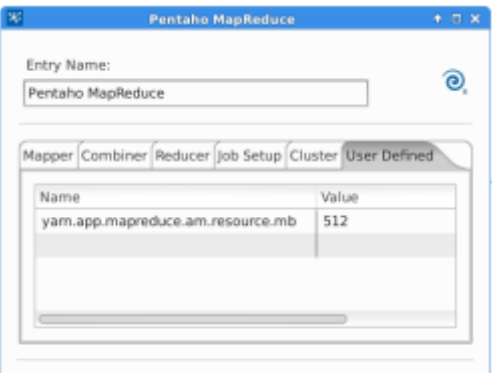
Using Pentaho MapReduce (continued)

Step	Action
29	<p>Edit the <i>Pentaho MapReduce</i> step, Cluster tab. Click New to create a new cluster.</p> 
30	<p>Continue editing the <i>Pentaho MapReduce</i> step, Cluster tab:</p> <ul style="list-style-type: none"> • Hadoop Job Name: Wordcount • Hadoop Cluster: CDH • Number of Mapper Tasks: 2 • Number of Reducer Tasks: 2 • Enable Blocking: checked • Logging interval: 5 

Continued on next page

Exercise 3: Using Pentaho Map Reduce, Continued

Using Pentaho MapReduce (continued)

Step	Action
31	<p>Edit the <i>Pentaho MapReduce</i> step, User Defined tab. Add the following parameter:</p> <ul style="list-style-type: none"> • Name: <code>yarn.app.mapreduce.am.resource.mb</code> • Value: <code>512</code> <p>This parameter/value is needed to allow the app to run in the limited resources of the VM environment.</p> 
32	<p>Run the word count job. You can monitor the progress in PDI-Spoon under the Execution Results, Logging tab.</p> <p>In a Web Browser, navigate to Hue web UI. On the far right of the menu bar, click on the icon with mouse over of Manage jobs. From here, you can monitor the progress of your word count job.</p>
33	<p>View the results of the Word Count in the HDFS output directory:</p> <ul style="list-style-type: none"> • In the Hue Web UI (http://localhost:8888/) and click on the icon with mouse over of Manage HDFS. • Navigate to: <code>/training/output</code> • Each reducer will create an output file. Since we configured Pentaho MapReduce to have 2 reducers you should see the following 2 output files: <code>part-00000</code> and <code>part-00001</code> • Click on <code>part-00000</code> file and view its content

Achievements

Having completed this exercise, you are able to:

- Implement Word Count application using Pentaho MapReduce.
- Use the Hue / YARN Resource Manager web interface to monitor Job information.

Your results are verified by your ability to complete the exercise in the allotted time. When you have completed the exercise, your instructor will facilitate a discussion of your results.

Exercise 4: Working with Hive and Impala

Exercise 4: Overview

Introduction In this exercise you learn how to connect to Hive and Impala to run queries using the native command line interfaces and PDI.

Allotted time The timing of this exercise is as follows:

Component	Time
Overview	1 minute
Exercise	59 minutes
Total:	1 hour

Prerequisites You must have access to the Pentaho/Cloudera virtual image used for the course exercises. You must also have access to support files required by the exercise (if any). You must have successfully completed all the previous exercises.

Objectives After completing this exercise, you will be able to:

- Create and query Hive tables using HQL with Beeline command line
- Create and query Impala tables using HQL with impala-shell command line
- Load Data into Impala using PDI
- Query Data in Impala using PDI

Exercise 4 Working with Hive and Impala

Create a Hive Data Source

To create a Hive data source using a PDI job:

Step	Action
1	<p>From a terminal, issue the following command:</p> <pre>sudo service --status-all</pre> <p>Verify the following services are already started:</p> <ul style="list-style-type: none"> • hive-metastore • hive-server2 • hive-webhcat-server • impala-catalog • impala-server • impala-state-store <p>Note: PDI and beeline clients require HiveServer2 (a.k.a. Hive Thrift server) to be running. By default, this VM is configured to automatically start HiveServer2 on port 9988.</p>
2	<p>Connect to Hive using Beeline command line interface.</p> <ul style="list-style-type: none"> • Start a new Terminal • Start Beeline at the command prompt: beeline • Connect to Hive (via HiveServer2) by issuing the following command at the Beeline prompt: !connect jdbc:hive2://localhost:9988 • When asked for username and password, just press Enter (No username/password).
3	<p>Create a table that will use the data from the wordcount output. At the beeline prompt, enter the following command: create external table wordcount(word STRING, counter INT) row format delimited fields terminated by '\t' stored as textfile location '/training/output';</p>
4	<p>Check the table. At the beeline prompt:</p> <ul style="list-style-type: none"> • Get a list of tables: show tables; • Get a description of the wordcount table: describe wordcount;

Continued on next page

Exercise 4 Working with Hive, Continued


Create a Hive
Data Source
(continued)

Step	Action
5	Get a list of the words that have at least 5000 instances in alphabetical order (hive requires limit with order by): select * from wordcount where counter>5000 order by word ASC limit 1000;
6	Get a list of the top 10 most frequently used words by entering the following statement in the beeline prompt: select * from wordcount order by counter DESC limit 10;
7	Delete the table by entering the following statement in the beeline prompt: drop table wordcount; This only removes the table definition from Hive. The raw data files are not removed from HDFS (in /training/output). Exit beeline using '!q' or '!quit' (without quotes)
8	Take a look at all the MapReduce jobs that Hive ran from the Hue's JobBrowser web interface. In a Web Browser, navigate to Hue web UI. On the far right of the menu bar, click on the icon with mouse over of Manage jobs . Remove the Username search and you will see all jobs that have been run.
9	We will do the same in Impala with its' command line interface: <ul style="list-style-type: none"> Start a new Terminal Execute the following command at the terminal prompt: impala-shell You should see a prompt similar to: [pentaho-di2000:21000] >
10	Create a table that will use the data from the wordcount output. At the impala-shell prompt, enter the following command: create external table wordcount(word STRING, counter INT) row format delimited fields terminated by '\t' stored as textfile location '/training/output';
11	Check the table. From the impala-shell prompt, execute the following: <ul style="list-style-type: none"> Get a list of tables: show tables; Get a description of the wordcount table: describe wordcount;

Continued on next page

Exercise 4 Working with Hive, Continued


Create a Hive Data Source (continued)

Step	Action
12	<p>Execute the following commands from impala-shell prompt. You will notice Impala executes them much faster than Hive:</p> <ul style="list-style-type: none"> • Get a list of the words that have at least 5000 instances in alphabetical order (impala requires limit with order by: select * from wordcount where counter>5000 order by word ASC limit 1000; • Get a list of the top 10 most frequently used words: select * from wordcount order by counter DESC limit 10; • Delete the table (This only removes the table definition from Impala. The raw data files are not removed from HDFS): drop table wordcount;
13	Next we will create and query a Hive table using PDI. Start PDI from the Launcher by clicking on the PDI icon.
14	Create a new Job. From the menu options, choose File > New > Job .
15	<p>Drag the following job entries onto the canvas and connect them:</p> <ul style="list-style-type: none"> • General > Start • Scripting > SQL • General > Transformation  <p>START SQL Transformation</p>
16	<p>Edit the <i>SQL</i> step:</p> <ul style="list-style-type: none"> • In the <i>Executing SQL Script</i> window, create a new connection by clicking on the New button next to the Connection field. Set the following fields for the new connection: <ul style="list-style-type: none"> – Connection Name: My Jobs Impala – Connection Type: Impala – Host Name: localhost – Database Name: default – Port Number: 21050 – Leave Username/Password empty – You can test to make sure that the connection works by clicking on the Test button. <p>Click on OK to save the connection and return back to the Executing SQL Script window.</p>

Continued on next page

Exercise 4 Working with Hive, Continued


Create a Hive Data Source (continued)

Step	Action
17	In the SQL Script field enter the following: drop table if exists wordcount; create external table wordcount(word STRING, counter INT) row format delimited fields terminated by '\t' stored as textfile location '/training/output';
18	Edit the Transformation step. In the Transformation specification tab, select the Transformation filename option and set the value to: \${Internal.Entry.Current.Directory}/get_top_ten.ktr
19	Save the job to: /home/zeus/Desktop/di2000/student/impala.kjb
20	Create transformation to query Impala data. From the menu options, choose File > New > Transformation .
21	Drag the following transformation steps onto the canvas and connect them: <ul style="list-style-type: none"> • Input > Table Input • Utility > Write to log  <p>The diagram shows two transformation steps on a canvas. On the left is a 'Table input' step, represented by a blue icon of a table with a cursor. On the right is a 'Write to log' step, represented by a blue icon of a document with a checkmark. A blue arrow points from the 'Table input' step to the 'Write to log' step, indicating a data flow between them.</p>
22	Edit the <i>Table input</i> step: <ul style="list-style-type: none"> • Create a new connection by clicking on the New button next to the Connection field. Set the following fields for the new connection: <ul style="list-style-type: none"> – Connection Name: My Xform Impala – Connection Type: Impala – Host Name: localhost – Database Name: default – Port Number: 21050 – Leave Username/Password empty – You can test to make sure that the connection works by clicking on the Test button. – Click on OK to save the connection and return back to the Table input window. • In the SQL field enter the following: select * from wordcount order by counter DESC limit 10

Continued on next page

Exercise 4 Working with Hive, Continued

Create a Hive Data Source (continued)

Step	Action
23	<p>Add the following fields to the Write to log step:</p> <ul style="list-style-type: none"> • word • counter 
24	<p>Save the transformation to:</p> <p>/home/zeus/Desktop/di2000/student/get_top_ten.ktr</p>
25	<p>Execute the job you saved above. View the output shown in Execution results, Logging tab. You should get a list of the top 10 most frequently used words.</p>

Achievements

Having completed this exercise, you are able to:

- Create and query Hive tables using HQL with beeline command line
- Load Data into Hive using PDI

Your results are verified by your ability to complete the exercise in the allotted time. When you have completed the exercise, your instructor will facilitate a discussion of your results.

Exercise 5: Working with HBase

Exercise 5: Overview

Introduction In this exercise you store and retrieve data using HBase.

Allotted time The timing of this exercise is as follows:

Component	Time
Overview	1 minute
Exercise	44 minutes
Total:	45 minutes

Prerequisites You must have access to the Pentaho/Cloudera virtual image used for the course exercises. You must also have access to support files required by the exercise (if any). You must have successfully completed all the previous exercises.

Objectives After completing this exercise, you will be able to:

- Use the HBase shell to create a table, add rows/columns, and query rows/columns
- Use PDI to load data from HDFS to HBase
- Use PDI to read data from HBase

Exercise 5: Working with HBase

Using HBase

To use the HBase shell to create a table:

Step	Action
1	From a terminal, issue the following command: <code>sudo service --status-all</code> Verify the following HBase services are running. <ul style="list-style-type: none"> • hbase-master • hbase-regionserver
2	Check if the following services are running (HBase depends on ZooKeeper): <ul style="list-style-type: none"> • zookeeper
3	Connect to HBase using HBase shell command line interface: <ul style="list-style-type: none"> • Start a new Terminal • Start HBase shell at the command prompt: hbase shell
4	Create an HBase table named <i>cars</i> with a single column family named <i>cf1</i> using the following command in the HBase shell: create 'cars', 'cf1'
5	List all tables in HBase: list
6	Insert a row with three columns that are part of the column family <i>cf1</i> : put 'cars', 'row_key1', 'cf1:make', 'ford' put 'cars', 'row_key1', 'cf1:model', 'taurus' put 'cars', 'row_key1', 'cf1:year', '1995'
7	Insert another row with three columns that are part of the column family <i>cf1</i> : put 'cars', 'row_key2', 'cf1:make', 'audi' put 'cars', 'row_key2', 'cf1:model', 'A6' put 'cars', 'row_key2', 'cf1:year', '2012'
8	List the all the values in the table : scan 'cars'
9	List all rows that have a given value using filters: scan 'cars', FILTER => "ValueFilter(=, 'binary:2012')"

Continued on next page

Exercise 5: Working with HBase, Continued

Using HBase (continued)

Step	Action
10	List all column values for a given column using filters: scan 'cars', {COLUMNS => ['cf1:make']}
11	Get a particular set of columns for a given: get 'cars', 'row_key1', {COLUMN => ['cf1:model', 'cf1:year']}
12	Delete a column: delete 'cars', 'row_key1', 'cf1:year' See the changes by listing the all the values in the table: scan 'cars'
13	Delete the table: disable 'cars' drop 'cars'

Load Data from HDFS

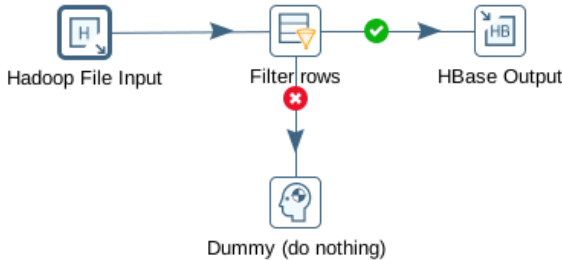
Use PDI to read data from HDFS and write data to HBase. We will copy the wordcount MapReduce results in HDFS and copy them to HBase:

Step	Action
1	First we must create the table we want to use. Execute the following in the HBase shell: create 'wordcount', 'cf1'
2	Start PDI from the Launcher by clicking on the PDI icon.
3	Create a new PDI transformation to get data from HDFS and store it in HBase: From the menu options, choose File > New > Transformation .

Continued on next page

Exercise 5: Working with HBase, Continued

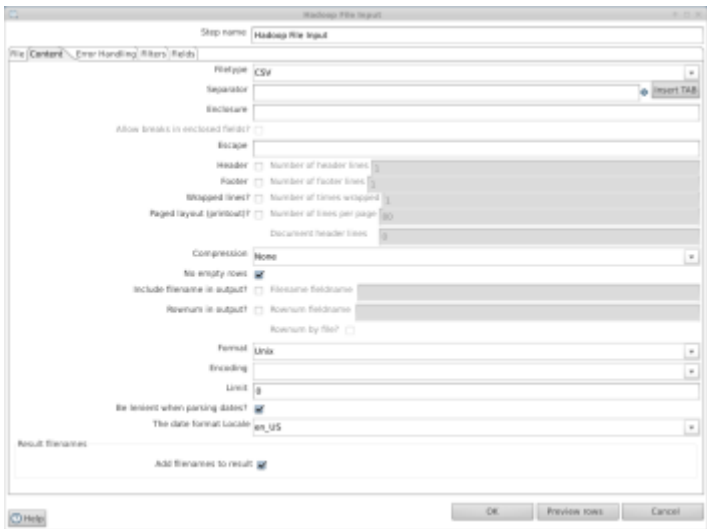
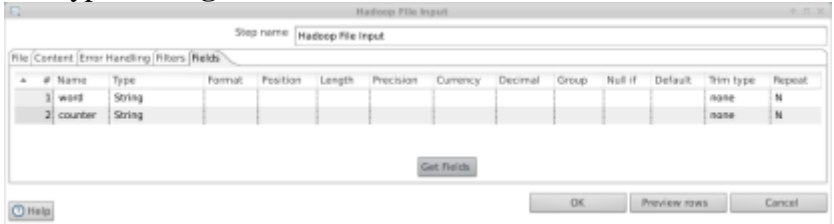
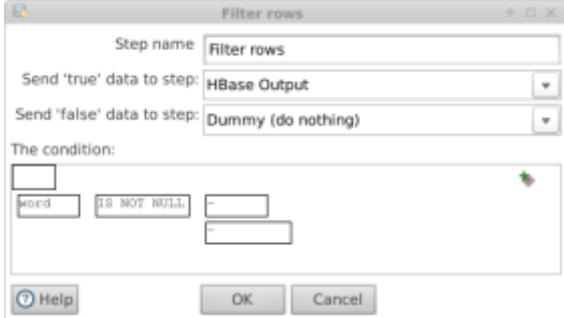
Load Data from HDFS (continued)

Step	Action
4	<p>Drag the following transformation steps onto the canvas and connect them:</p> <ul style="list-style-type: none"> • Big Data > Hadoop File Input • Flow > Filter rows • Flow > Dummy • Big Data > HBase Output <p>Note: Filter row's True output should go to HBase Output step and the False output should connect to the Dummy step.</p>  <pre> graph LR A[Hadoop File Input] --> B[Filter rows] B --> C[HBase Output] B --> D[Dummy do nothing] </pre>
5	<p>Edit <i>Hadoop File Input</i> step's File Tab. Add the following row to the Selected file:</p> <ul style="list-style-type: none"> • Environment: CDH • File/Directory: /training/output • Wildcard (RegExp): part.*
6	<p>Edit Hadoop File Input step's Content Tab:</p> <ul style="list-style-type: none"> • Filetype: CSV • Separator: (Remove the existing semi-column and add tab by clicking on the Insert Tab button) • Enclosure: (none, remove the existing quote) • Header: (uncheck) • Format: Unix

Continued on next page

Exercise 5: Working with HBase, Continued

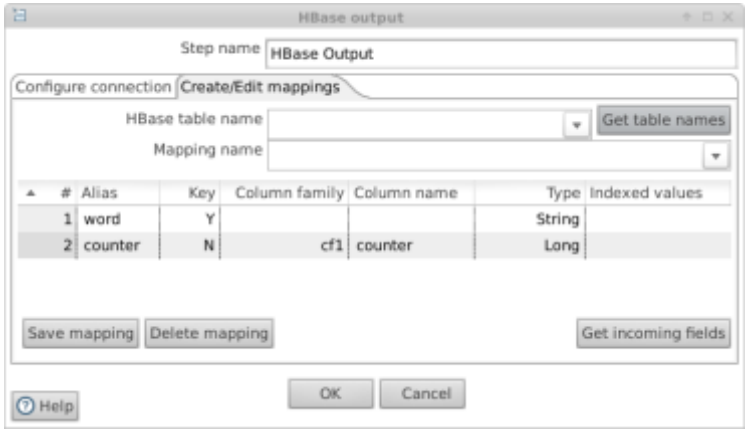
Load Data from HDFS (continued)

Step	Action
6 (cont)	
7	<p>Edit Hadoop File Input step's Fields Tab. Add two rows:</p> <ul style="list-style-type: none"> • Row #1: <ul style="list-style-type: none"> – Name: word – Type: String • Row #2: <ul style="list-style-type: none"> – Name: counter – Type: String 
8	<p>Edit Filter rows: Add condition that word IS NOT NULL.</p> 

Continued on next page

Exercise 5: Working with HBase, Continued

Load Data from HDFS (continued)

Step	Action
9	<p>Edit the <i>HBase Output</i> step's Configure connection tab:</p> <ul style="list-style-type: none"> • Hadoop Cluster: CDH • HBase table name: wordcount • Store mapping info in step meta data: checked • Size of write buffer (bytes): 2097152
10	<p>Edit the <i>HBase Output</i> step's Create/Edit mappings tab:</p> <ul style="list-style-type: none"> • HBase table name: wordcount • Click the Get incoming fields button to auto populate the mapping table with two rows. <p>Edit the table to have the following values:</p> <ul style="list-style-type: none"> • Row #1: <ul style="list-style-type: none"> – Alias: word – Key: Y – Type: String • Row #2: <ul style="list-style-type: none"> – Alias: counter – Key: N – Column family: cf1 – Column name: counter – Type: Long 

Continued on next page


Exercise 5: Working with HBase, Continued

Load Data from HDFS (continued)

Step	Action
11	Save the transformation to: <code>/home/zeus/Desktop/di2000/student/store_hbase.ktr</code>
12	Run the transformation.
13	Validate data was added to HBase by using HBase shell: scan 'wordcount', LIMIT => 10

Read Data from HBase

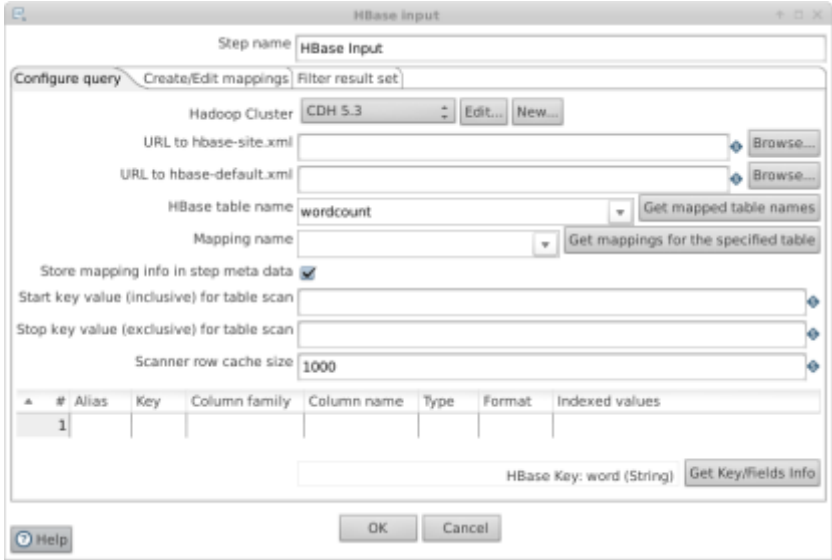
To create transformation to get words that appear at least 1000 times by reading and filtering the wordcount results that are stored in HBase:

Step	Action
1	Create transformation to get words that appear at least 1000 times by reading and filtering the wordcount results that are stored in HBase. From the menu options, choose File > New > Transformation .
2	Drag the following transformation steps onto the canvas and connect them: <ul style="list-style-type: none"> • Big Data > HBase Input • Utility > Write to log  <pre> graph LR A[HB] --> B[Log] subgraph Labels direction LR L1[HBase Input] L2[Write to log] end </pre>

Continued on next page

Exercise 5: Working with HBase, Continued

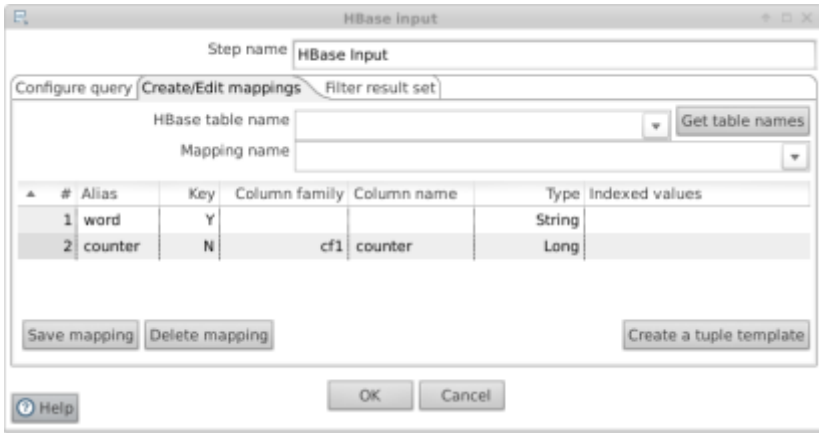
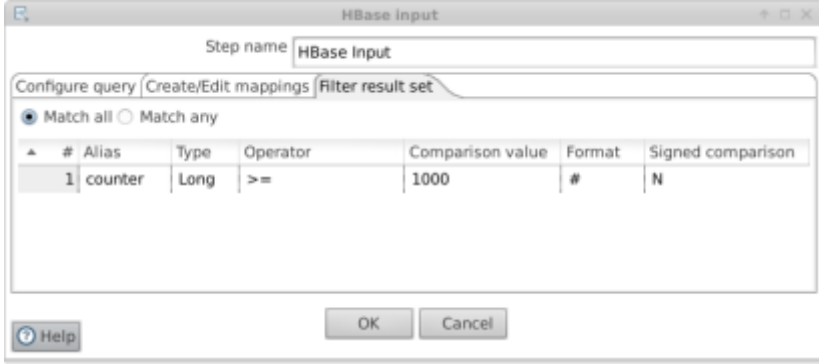
Read Data from HBase (continued)

Step	Action
3	<p>Edit the <i>HBase Input</i> step's Configure query tab:</p> <ul style="list-style-type: none"> • Hadoop Cluster: CDH57 • HBase table name: wordcount • Store mapping info in step meta data: checked 
4	<p>Edit the <i>HBase Input</i> step's Create/Edit mappings tab. Add the following two rows:</p> <ul style="list-style-type: none"> • Row #1: <ul style="list-style-type: none"> – Alias: word – Key: Y – Type: String • Row #2: <ul style="list-style-type: none"> – Alias: counter – Key: N – Column family: cf1 – Column name: counter – Type: Long

Continued on next page

Exercise 5: Working with HBase, Continued

**Read Data
from HBase
(continued)**

Step	Action
4 (cont)	
5	<p>Edit the <i>HBase Input</i> step's Filter result set tab. Add the following row:</p> <ul style="list-style-type: none">• Alias: counter• Type: Long• Operator: >=• Comparison value: 1000• Format: #• Signed comparison: N 
6	<p>Add the following fields to the Write to log step:</p> <ul style="list-style-type: none">• word• counter

Continued on next page

Exercise 5: Working with HBase, Continued

Read Data from HBase (continued)

Step	Action
7	Save the transformation to: <code>/home/zeus/Desktop/di2000/student/read_hbase.ktr</code>
8	Execute the transformation. View the output shown in Execution results, Logging tab. You should get a list of the top all words that appear more than 1000 times (there should be 34 words).

Achievements

Having completed this exercise, you are able to:

- Use the HBase shell to create a table, add rows/columns, and query rows/columns
- Use PDI to load data from HDFS to HBase
- Use PDI to read data from HBase

Your results are verified by your ability to complete the exercise in the allotted time. When you have completed the exercise, your instructor will facilitate a discussion of your results.