Connecting the
Data-Driven Enterprise

Talend Real-time Big Data Platform Getting Started Guide

6.4.1

Contents

Copyright

Introduction to Talend Real-time Big Data Platform

Talend Real-time Big Data Platform combines Talend products into a common set of powerful, easy-to-use solutions.

Talend's data integration solution helps companies deal with growing system complexities by addressing both ETL for analytics and ETL for operational integration needs and offering industrialization features and extended monitoring capabilities.

Built on top of Talend's data integration solution, the big data solution is a powerful tool that enables users to access, transform, move and synchronize big data by leveraging the Apache Hadoop Big Data Platform and makes the Hadoop platform ever so easy to use.

Talend's ESB solution provides a versatile and flexible, enterprise service bus (ESB) that allows organizations to address any integration challenge - from simple departmental projects to complex, heterogeneous IT environments.

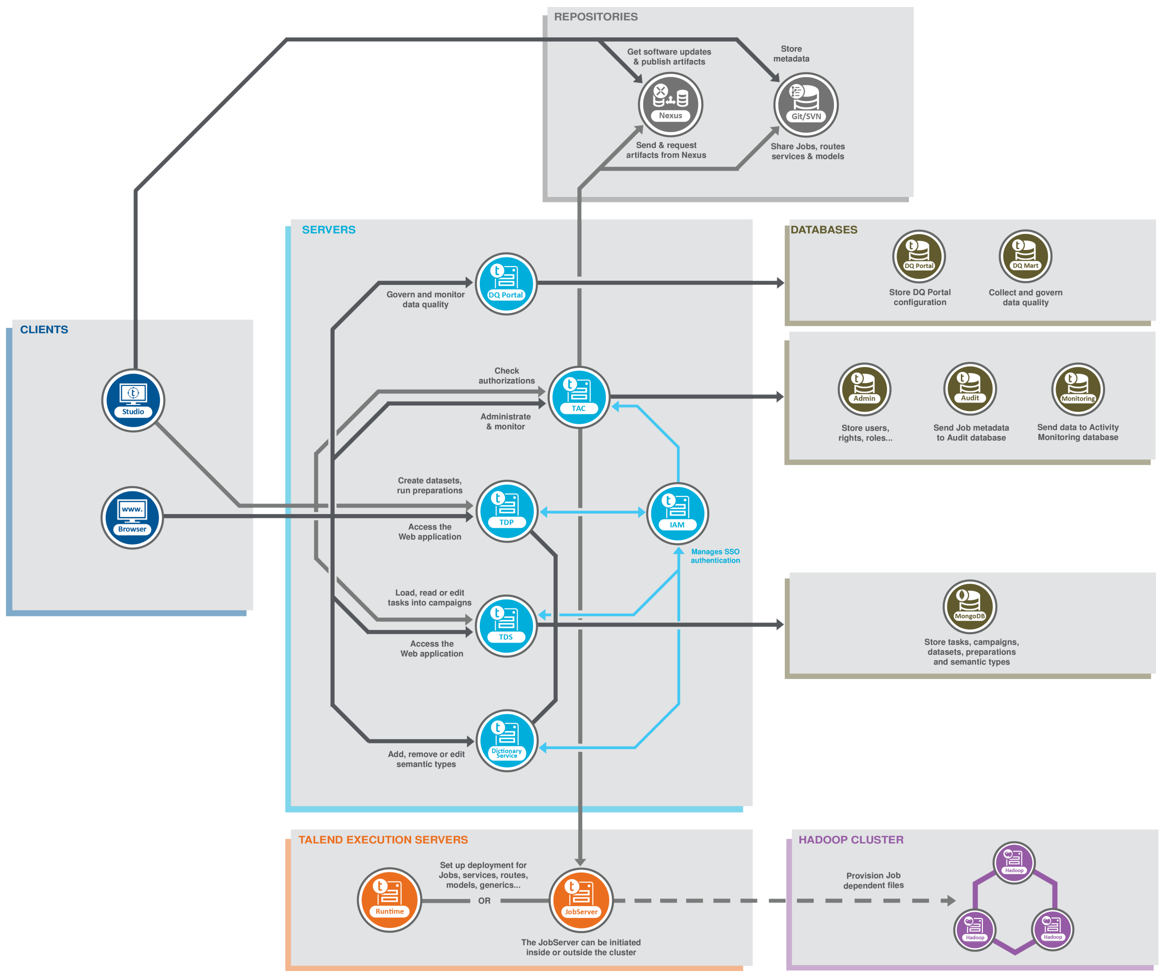
In the field of data quality, users can profile and cleanse the data available in different data sources, browse the analyzed data and generate reports on analysis results from Talend DQ Portal.

This Getting Started Guide focuses on the Big Data and Data Quality features of Talend Real-time Big Data Platform.

Talend Real-time Big Data Platform functional architecture

The Talend Real-time Big Data Platform functional architecture is an architectural model that identifies Talend Real-time Big Data Platform functions, interactions and corresponding IT needs. The overall architecture has been described by isolating specific functionalities in functional blocks.

The following chart illustrates the main architectural functional blocks.


REPOSITORIES
Get software updates & publish artifacts
Store metadata
Nexus
Git/SVN
Send & request artifacts from Nexus
Share Jobs, routes services & models
SERVERS
DATABASES
DQ Portal
DQ Mart
Store DQ Portal configuration
Collect and govern data quality
Govern and monitor data quality
CLIENTS
Check authorizations
Administrate & monitor
TAC
Admin
Audit
Monitoring
Store users, rights, roles...
Send Job metadata to Audit database
Send data to Activity Monitoring database
Studio
Create datasets, run preparations
Access the Web application
TDP
IAM
Browser
Manages SSO authentication
Load, read or edit tasks into campaigns
Access the Web application
TDS
MongoDB
Store tasks, campaigns, datasets, preparations and semantic types
Add, remove or edit semantic types
Dictionary Service
TALEND EXECUTION SERVERS
HADOOP CLUSTER
Set up deployment for Jobs, services, routes, models, generics...
Runtime
OR
JobServer
The JobServer can be initiated inside or outside the cluster
Provision Job dependent files
Hadoop

The different types of functional blocks are:

- From Talend Studio, you design and launch Big Data Jobs that leverage a Hadoop cluster to handle large data sets. Once launched, these Jobs are sent to, deployed on and executed on this Hadoop cluster.

  From the Studio, you can also use predefined patterns and indicators to analyze data stored in different data sources, browse and query analysis results and remove corrupt, incomplete or inaccurate data.
- A Hadoop cluster independent of the Talend system to handle large data sets.
- A Talend JobServer or Runtime installed inside or outside the Hadoop cluster to deploy and execute Jobs.

  For a Hortonworks cluster, it is recommended to install the JobServer or Runtime in the EDGE node machine in order to avoid potential firewall and access issues.

  For an Amazon EMR cluster, it is also recommended to install the JobServer or Runtime in the cluster.
- From the Talend DQ Portal, you can generate reports on analysis results and share them with other business users.

Prerequisites to using Talend Real-time Big Data Platform

This chapter provides basic software and hardware information required and recommended to get started with your Talend Real-time Big Data Platform.

- Memory requirements on page 6

- Software requirements on page 6

It also guides you to install and configure required and recommended third-party tools:

- Installing Java on page 7
- Setting up the Java environment variable on Windows on page 8 or Setting up the Java environment variable on Linux on page 8
- Installing and configuring a versioning and revision control system on Windows on page 8
- Installing 7-Zip (Windows) on page 9

Memory requirements

To make the most out of your Talendproduct, please consider the following memory and disk space usage:

Memory usage	4GB minimum, 8GB recommended
Disk space	30GB

Software requirements

To make the most out of your Talend product, please consider the following system and software requirements:

Required software

- Operating System for Talend Studio:

Support type	**Operating System**	**Version**	**Processor**
Recommended	Microsoft Windows Professional	7	64-bit
Recommended	Linux Ubuntu	14.04	64-bit
Supported	Apple OS X	El Capitan/10.11 Yosemite/10.10 Mavericks/10.9	64-bit 64-bit 64-bit

- Operating System for Talend Server modules:

Support type	**Operating System**	**Version**	**Processor**
Recommanded	Microsoft Windows Server	2012 R2	64-bit

Support type	Operating System	Version	Processor
Recommanded	Red Hat Enterprise Linux Server	7.2	64-bit
Supported	Solaris (SunOs)	11 11 10 10	x86/64-bit Sparc/64-bit x86/64-bit Sparc/64-bit

- Java 8 JRE Oracle. See Installing Java on page 7.
- For Windows, VisualSVN Server. See Installing and configuring a versioning and revision control system on Windows on page 8.
- A properly installed and configured Hadoop cluster.

  Ensure that the client machine on which the Talend Studio is installed can recognize the host names of the nodes of the Hadoop cluster to be used. For this purpose, add the IP address/hostname mapping entries for the services of that Hadoop cluster in the `hosts` file of the client machine.

  For example, if the host name of the Hadoop Namenode server is talend-cdh550.weave.local, and its IP address is 192.168.x.x, the mapping entry reads `192.168.x.x talend-cdh550.weave.local`.
- A properly installed and configured MySQL database, with a database named `gettingstarted`.

Optional software

- 7-Zip. See Installing 7-Zip (Windows) on page 9.

Installing Java

To use your Talend product, you need Oracle Java Runtime Environment installed on your computer.

1. From the Java SE Downloads page, under **Java Platform, Standard Edition**, click the **JRE Download**.
2. From the **Java SE Runtime Environment 8 Downloads** page, click the radio button to **Accept License Agreement**.
3. Select the appropriate download for your Operating System.
4. Follow the Oracle installation steps to install Java.

When Java is installed on your computer, you need to set up the `JAVA_HOME` environment variable.

For more information, see:

- Setting up the Java environment variable on Windows on page 8.
- Setting up the Java environment variable on Linux on page 8.

Setting up the Java environment variable on Windows

Prior to installing your Talend product, you need to set the JAVA_HOME and Path environment variables.

1. Go to the **Start Menu** of your computer, right-click on **Computer** and select **Properties**.
2. In the **Control Panel Home** window, click **Advanced system settings**.
3. In the **System Properties** window, click **Environment Variables...**.
4. Under **System Variables**, click **New...** to create a variable. Name the variable `JAVA_HOME`, enter the path to the Java 8 JRE, and click **OK**.

   Example of default JRE path: `C:\Program Files\Java\jre1.8.0_77`.
5. Under **System Variables**, select the **Path** variable and click **Edit...** to add the previously defined `JAVA_HOME` variable at the end of the `Path` environment variable, separated with semi colon.

   Example: `<PathVariable>;%JAVA_HOME%\bin`.

Setting up the Java environment variable on Linux

Prior to installing your Talend product, you have to set the JAVA_HOME and Path environment variables.

1. Find the JRE installation home directory.

   Example: `/usr/lib/jvm/jre1.8.0_65`
2. Export it in the `JAVA_HOME` environment variable.

   Example:

   ```
   export JAVA_HOME=/usr/lib/jvm/jre1.8.0_65
   export PATH=$JAVA_HOME/bin:$PATH
   ```
3. Add these lines at the end of the user profiles in the `~/.profile` file or, as a superuser, at the end of the global profiles in the `/etc/profile` file.
4. Log on again.

Installing and configuring a versioning and revision control system on Windows

Talend Real-time Big Data Platform supports Git and Subversion as the software versioning and revision control systems. Talend uses them through Talend Administration Center as a repository for Talend projects to store Jobs, connections, schema definitions, custom jars, third-party libraries, and properties files.

You can install both or either of these systems on your computer. This section describes how to install and configure Subversion on Windows. For Unix based Operating Systems, Talend recommends that you to let the Talend Installer install it on your machine. For how to install and configure Git, see the Talend Installation Guide.

1. From the VISUALSVN SERVER // Download page, click **64-bit** to download VisualSVN Server.
2. Follow VisualSVN steps to install the VisualSVN Server.
3. Start VisualSVN Server to create a new repository and a user.
4. In **VisualSVN Server**, right-click on **Repositories** and click **Create new repository...**.
5. In the **Create new Repository** wizard, select the default options until the end, in the **Repository Name** field, name the new repository `gettingstarted_repo` for example, and click **Create** and **Finish**.
6. Back in the main window, right-click the newly created **gettingstarted_repo** and select **Properties...**.

7. In the **Properties** wizard, click **Add...** and click **Create user...**.
8. In the **Create New User** wizard, define a user name and password: `admin/admin` for example, and click **OK** to all the wizards.

   The admin user has now Read / Write permissions to access the repository gettingstarted_repo.

Your SVN server is now installed and the repository is ready to store all Talend projects. The URL to this repository and the user's credentials will be needed later on to configure the Talend Administration Center Web application used to manage the projects.

Installing 7-Zip (Windows)

Talend recommends to install 7-Zip and to use it to extract the installation files: http://www.7-zip.org/download.html.

1. Download the 7-Zip installer corresponding to your Operating System.
2. Navigate to your local folder, locate and double-click the 7z exe file to install it.

The download will start automatically.

Downloading and installing Talend Real-time Big Data Platform

This chapter provides basic information about downloading and installing Talend Real-time Big Data Platform.

The installation described here is a local installation using the Talend Installer and the default settings, only for testing purposes. If you want to do any customization or a real life installation, we recommend that you follow the Talend Real-time Big Data Platform Installation Guide, contact the IT service of your company, or contact Talend Professional Services.

Downloading Talend Real-time Big Data Platform

To install Talend Real-time Big Data Platform, you need to:

1. Save and keep in a safe place the license file contained in the email you received in response to your subscription to Talend Real-time Big Data Platform.

   This file contains your license key, which is required for you to install Talend Real-time Big Data Platform and access each module of it.
2. Download the Talend Installer, which is the recommended way to install Talend Real-time Big Data Platform, by following the corresponding link provided in the email:
   - `Talend-Installer-Starter-YYYYMMDD_HHmm-VA.B.C-installer.zip`
   - `dist`, which is mandatory to run the Talend Installer

   The Talend Installer is a wizard-based application that guides you step by step through the installation and configuration of the Talend Real-time Big Data Platform modules.

   Warning:

   Mac users can use only the Studio zip file, and not the Talend Installer, to install Talend Real-time Big Data Platform.

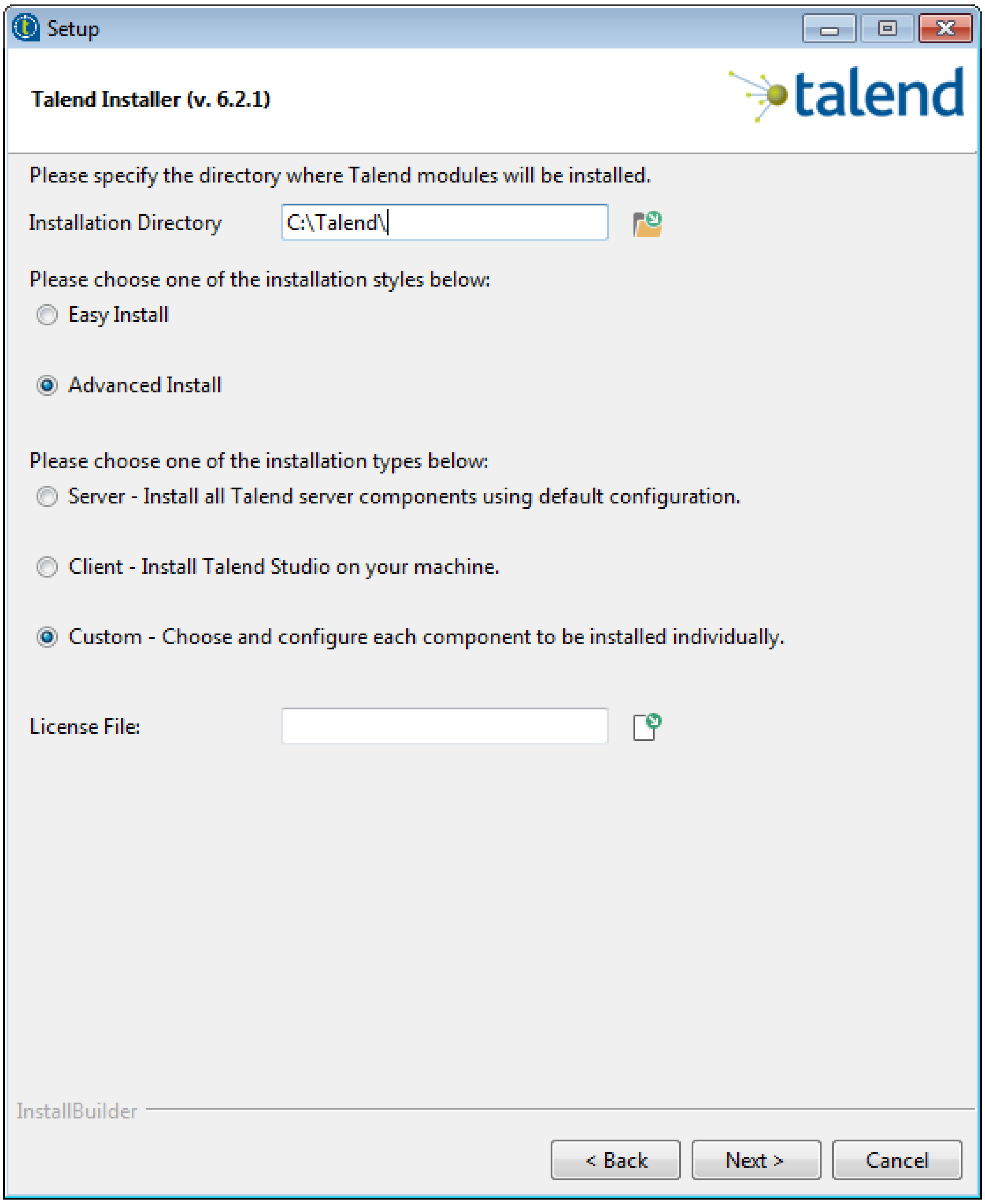
Installing Talend Real-time Big Data Platform

Follow the procedure below to install Talend Real-time Big Data Platform via the Talend Installer, using a number of default settings. If you need any information on the default ports used by the Talend Installer, see the section on port information of the Talend Real-time Big Data Platform Installation Guide.

1. Unzip the Talend Installer package to a local folder, and make sure that the `dist` file is in the same directory as the Installer executable.
2. Double-click the executable file of the Talend Installer corresponding to your Operating System to open the Talend Installation wizard.
3. Click **Next** to view the License Agreement.
4. Select **I accept the agreement** and click **Next**.

   The wizard lets you:

   - define the installation directory.
   - select an installation style:
     - **Easy Install**: lets you install Talend Real-time Big Data Platform with default configuration (default local host, default ports and so on) without asking you for any information.
     - **Advanced Install**: lets you customize your installation.
   - select an installation type:
     - **Server**: to install all Talend server modules.
     - **Client**: to install only the Talend Studio.
     - **Custom**: to install the Talend Studio and selected server modules on this machine.
   - specify your license file that will authorize your installation.



Talend Installer (v. 6.2.1)

Please specify the directory where Talend modules will be installed.

Installation Directory: C:\Talend\

Please choose one of the installation styles below:

Easy Install [✓ Advanced Install]

Please choose one of the installation types below:

Server - Install all Talend server components using default configuration.

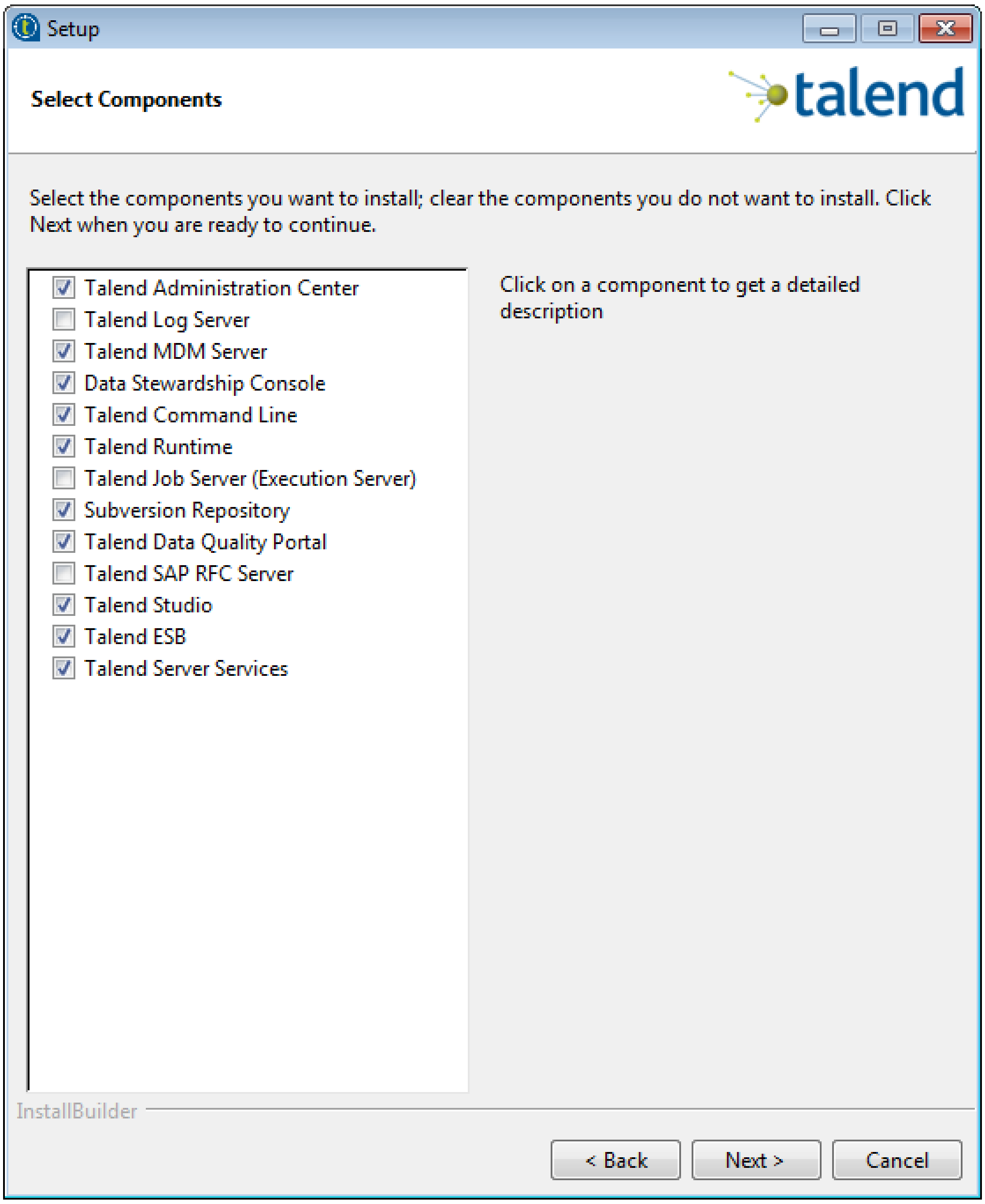
Client - Install Talend Studio on your machine.

[✓ Custom - Choose and configure each component to be installed individually.]

License File:

5. Specify the installation directory, and select the **Advanced Install** and **Custom** installation options, browse to the license file you received from Talend and click **Next**.

   The wizard lets you select the Talend Real-time Big Data Platform modules you want to install. Note that the modules available vary depending on your license.


Setup

Select Components

Select the components you want to install; clear the components you do not want to install. Click Next when you are ready to continue.

- [x] Talend Administration Center
- [ ] Talend Log Server
- [x] Talend MDM Server
- [x] Data Stewardship Console
- [x] Talend Command Line
- [x] Talend Runtime
- [ ] Talend Job Server (Execution Server)
- [x] Subversion Repository
- [x] Talend Data Quality Portal
- [ ] Talend SAP RFC Server
- [x] Talend Studio
- [x] Talend ESB
- [x] Talend Server Services

Click on a component to get a detailed description

6. Select the modules to install, and select the **Talend Server Services** option so that the Talend server modules are installed as services and will be automatically launched at system boot. When ready, click **Next**.

In this procedure, make sure that the following modules are selected:

- **Talend Administration Center**
- **Talend Command Line**
- **Talend Runtime**
- **Subversion Repository** (Linux only)

  For Windows, you need to install VisualSVN Server manually. For more information, see Installing and configuring a versioning and revision control system on Windows on page 8.
- **Talend Data Quality Portal**
- **Talend Studio**
- **Talend ESB**

7. In the rest of the steps, just click **Next** in succession to install the selected modules with the default options and settings. When done, click **Finish** to close the wizard.

Now the selected modules have been installed to your computer, and you can start setting up your

Talend Real-time Big Data Platform.

Configuring and setting up your Talend product

This chapter provides basic information required to configure and set up your Talend Real-time Big Data Platform.

Starting and configuring the Talend Administration Center

If you are using one of Talend's subscription products, you would need to start the Talend Administration Center at this point.

From the Talend Administration Center, you will be able to:

- create a new remote project, create a new user account for yourself and give access right for yourself to this newly created project.

  The advantage of such remote projects is that several users can work collaboratively on the same centralized project, which is the least for company-scale projects.

  For more information, see Setting up your first user and project on page 15.
- manage the publishing and deployment of data integration Jobs developed via the Talend Studio into the Talend Runtime directly from the Talend Administration Center.

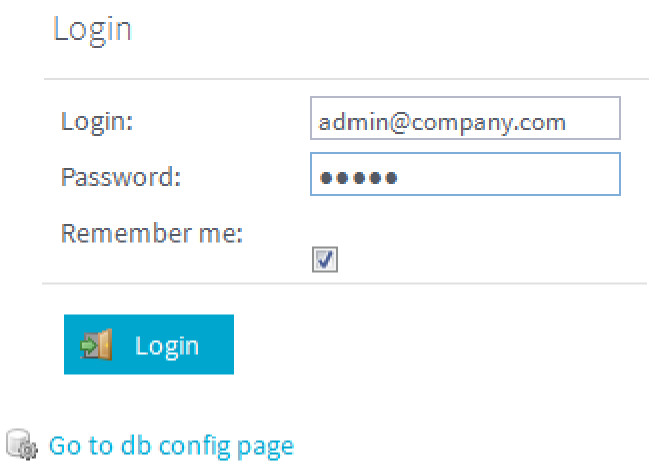
  To do so, you have to connect the Talend Administration Center to the Talend Runtime.

  For more information, see Connecting Talend Runtime Container to Talend Administration Center on page 17.

Running the Talend Administration Center

1. As you installed the product via the Talend Installer, go to the directory named tac.
2. Double-click the executable file: `start_tac.bat`.

   This will launch the servlet container (by default, Apache Tomcat) holding the Talend Administration Center.
3. Once started, log in to Talend Administration Center with the default account: `admin@company.com`/`admin`.


Login

Login: admin@company.com

Password: ●●●●●

Remember me: ✓

Configuring the Talend Administration Center

1. Click the **Configuration** link in the left-hand menu to finish configuring the Talend Administration Center.

   Most of the warnings displaying are due to the fact that you did not start all the modules of the product yet. For example:

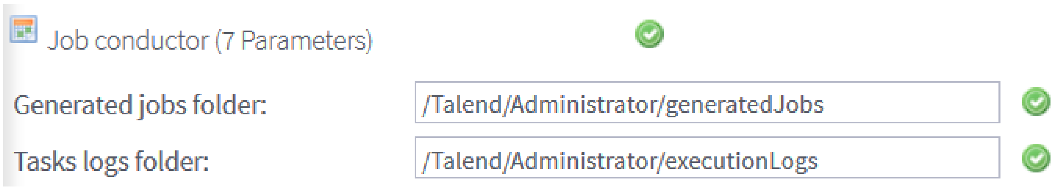
   - **Commandline/secondary** will turn green only if you decide to use and start a secondary commandline, which will not be the case in the following demo use cases.
   - **ESB Service Locator and SAM** should be green if you started the Talend Runtime and its Infrastructure Services.
   - **ESB Identity and Access Management** will turn green only if you install and start the Talend Identity Management Service, which will not be the case in the following demo use cases.
   - **ESB Service Registry** should be green if you started the Talend Runtime and its Infrastructure Services.
   - **Software Update** will turn green if you start the dedicated Talend Artifact Repository, which will not be the case in the following demo use cases.
   - The **Commandline/primary** section should be green if you installed Talend Administration Center using the Talend Installer.

     The Commandline must be started to use the **ESB Publisher**.
2. For the **Artifact Repository** section of the Talend Administration Center **Configuration** page, simply start the Talend Artifact Repository.

   The Talend Artifact Repository is provided with the Talend Administration Center. So, if you installed the product via the Talend Installer, you will find it in the tac folder of your installation, in an `Artifact-Repository-Nexus-VA.B.C.D.E` subfolder. If you installed the product manually, the Talend Artifact Repository will be available in the Talend Administration Center package as well (`Talend-AdministrationCenter-rXXXXXX-VA.B.C`).
3. From the `Artifact-Repository-Nexus-VA.B.C.D.E` directory, run:

   - `./bin/nexus console` (Linux)
   - `.\bin\nexus.bat console` (Windows).

   It will now be running on http://localhost:8081/nexus/index.html. Replace `localhost` and `8081` with the IP address and the port of the server on which you installed the Talend Artifact Repository. You can log in with the default login and password: `admin`/`Talend123`
4. To configure the **Job conductor** parameters, create two folders in the machine in which Talend Administration Center is installed by replicating the directories given by default.



Job conductor (7 Parameters)

Generated jobs folder:	/Talend/Administrator/generatedJobs
Tasks logs folder:	/Talend/Administrator/executionLogs

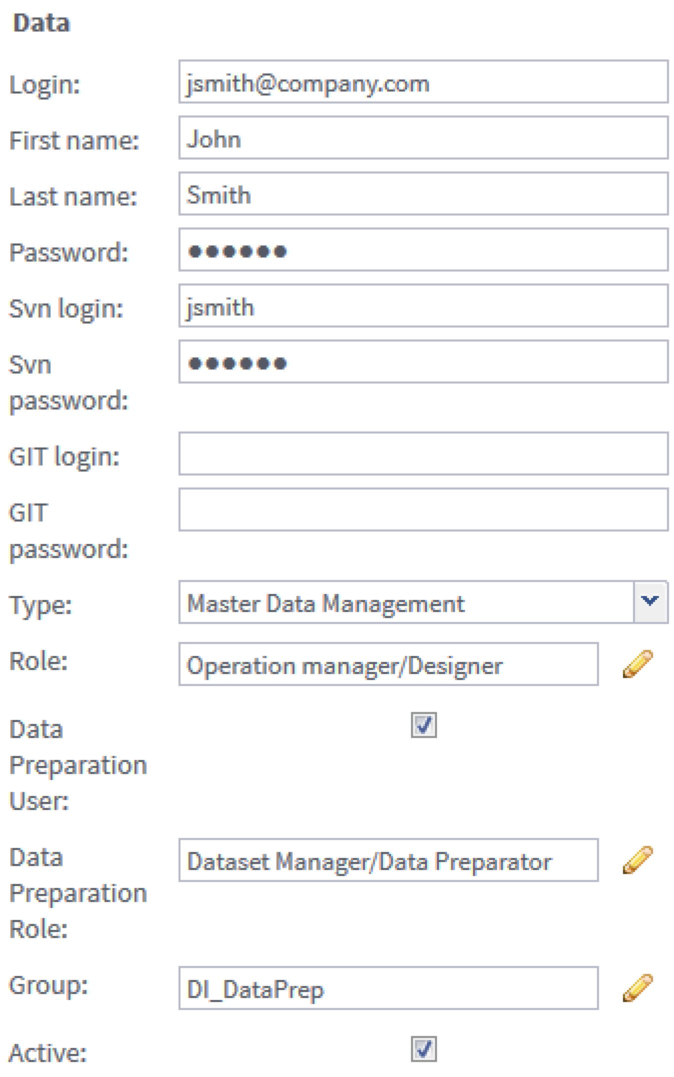
5. Configure the **Logging** parameters by filling in the path to technical and business log files.

   The **Logstash host and port** parameter will get green when you will start the Talend Log Server, by executing `start_logserver.bat` (Windows) or `start_logserver.sh` (Linux), which will not be the case in the following demo use cases.
6. Configure the **Svn** or **Git** parameters by filling in the connection information to your SVN or Git.

Setting up your first user and project

Now that the Talend Administration Center is started and configured, you can use its administration functionalities to set up your first user and project.

1. Click the **Users** link in the left-hand menu, and in the **Users** page, click the **Add** button to create a new user.
2. Fill in the **Data** panel to the right with your Talend Administration Center and Svn credentials.

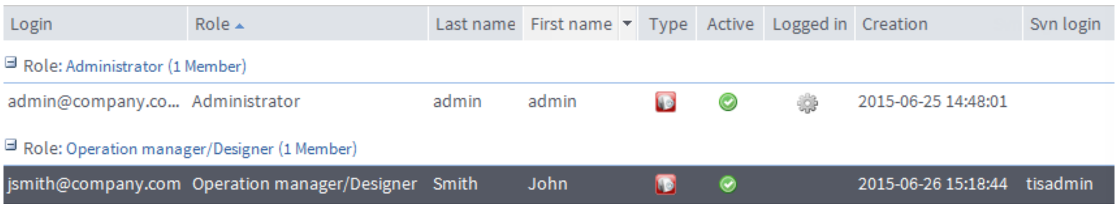


Data

Login:	jsmith@company.com
First name:	John
Last name:	Smith
Password:	●●●●●●
Svn login:	jsmith
Svn password:	●●●●●●
GIT login:	
GIT password:	
Type:	Master Data Management
Role:	Operation manager/Designer
Data Preparation User:	✓
Data Preparation Role:	Dataset Manager/Data Preparator
Group:	DI_DataPrep
Active:	✓

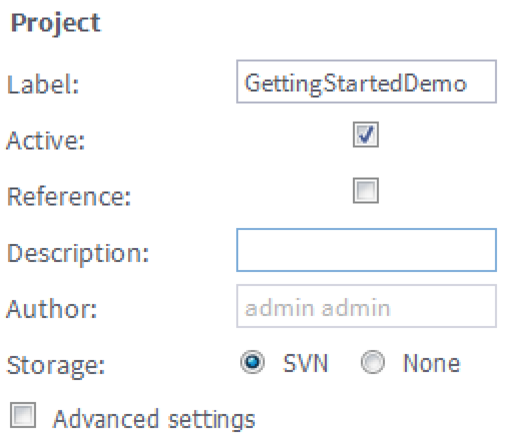
- From the **Type** list select a type according to the nature of tasks the user is going to perform.
- Make sure to assign both **Designer** and **Operation manager** roles to the account, to be able to use the Studio and manage the deployment of artifacts from the Talend Administration Center respectively.

3. Click **Save**. The new account you created displays on the list.



Login	Role	Last name	First name	Type	Active	Logged in	Creation	Svn login
Role: Administrator (1 Member)								
admin@company.co...	Administrator	admin	admin		✓		2015-06-25 14:48:01	
Role: Operation manager/Designer (1 Member)								
jsmith@company.com	Operation manager/Designer	Smith	John		✓		2015-06-26 15:18:44	tisadmin

4. Click the **Projects** link in the left-hand menu, and in the **Projects** page, click the **Add** button to create a new project.
5. In the **Project** panel to the right, put the name of your project in the **Label** field.

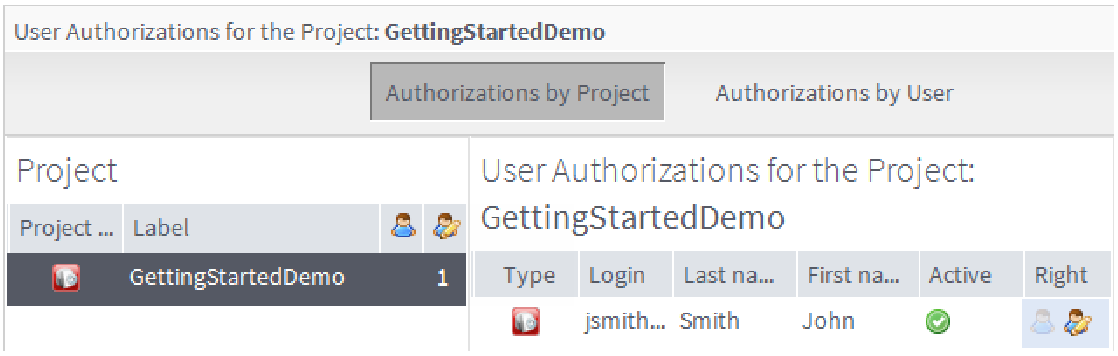


Project

Label:	GettingStartedDemo
Active:	☑
Reference:	☐
Description:	
Author:	admin admin
Storage:	[✓ SVN] None

☐ Advanced settings

6. Click **Save**. The new project you created displays on the list.
7. Click the **Project authorizations** link in the left-hand menu, and in the **Project authorizations** page, give access rights to the project to the user account you just created.
8. In the **Project** panel to the left, click the project to select it, and in the **User Authorizations** panel to the right, click the **Read write** icon of the user: .



User Authorizations for the Project: **GettingStartedDemo**

Project

Project ...	Label		
	GettingStartedDemo		1

User Authorizations for the Project: GettingStartedDemo

Type	Login	Last na...	First na...	Active	Right
	jsmith...	Smith	John	✓	

Now, you will be able to connect to this project.

Connecting Talend Runtime Container to Talend Administration Center

To do so, you have to be logged in as a user with an **Operation manager** role. So, if you are logged in as **Administrator** with the `admin@company.com` account, log out and log in again with the user you created in Setting up your first user and project on page 15.

1. Click the **Servers** page in the menu to the left to display the list of servers.

   If you installed the product via the Talend Installer, depending on your license, one server, the Jobserver, might already be installed and will display by default: **serv1**.
2. Create a new server by clicking **Add** > **Add server**. The **Execution server** form is displayed on the right.
3. Fill in the form as follows and click **Save**:
   - **Label**: `Talend Runtime`, for example.
   - **Host**: `127.0.0.1`
   - Select the **Talend Runtime** check box.

Launching the Studio for the first time

The Studio installation directory contains binaries for several platforms including Mac OS X and Linux/Unix.

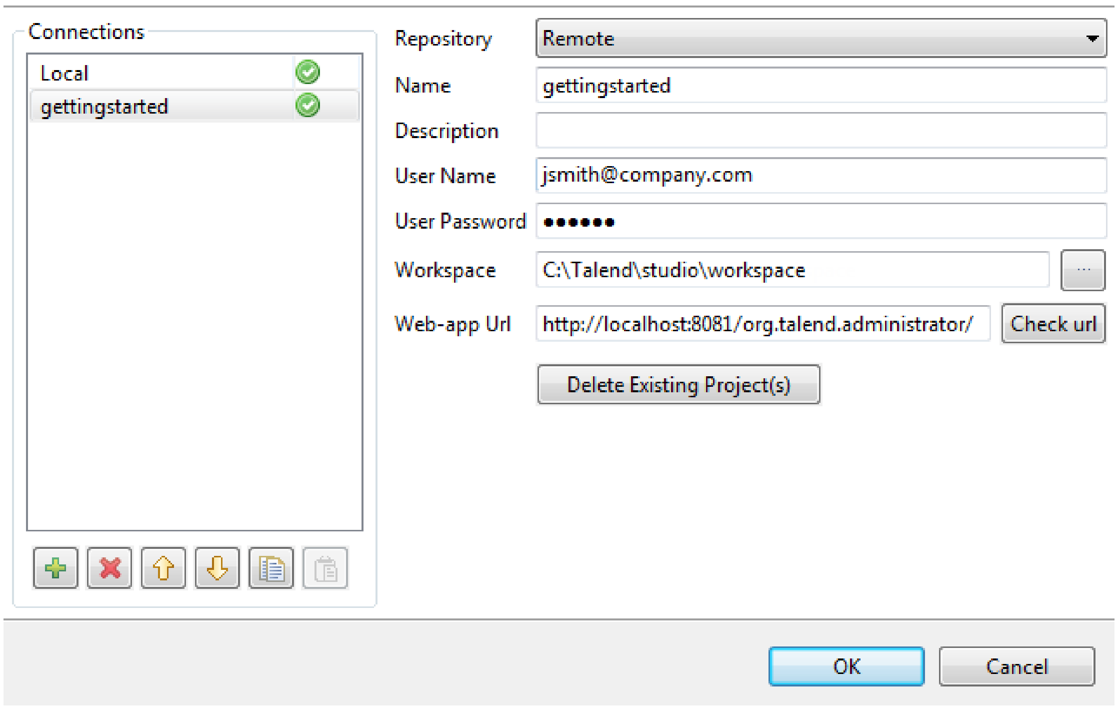
To open the Talend Studio for the first time, do the following:

1. Double-click the executable file corresponding to your operating system, for example:
   - `Talend-Studio-win-x86_64.exe`, for Windows.
   - `Talend-Studio-linux-gtk-x86_64`, for Linux.
   - `Talend-Studio-macosx-cocoa.app`, for Mac.
2. In the **User License Agreement** dialog box that opens, read and accept the terms of the end user license agreement to proceed.

Logging in to the Studio

To log in to the Talend Studio for the first time, do the following:

1. In the Talend Studio login window, click the **Manage Connections** button. The connection setup wizard opens.

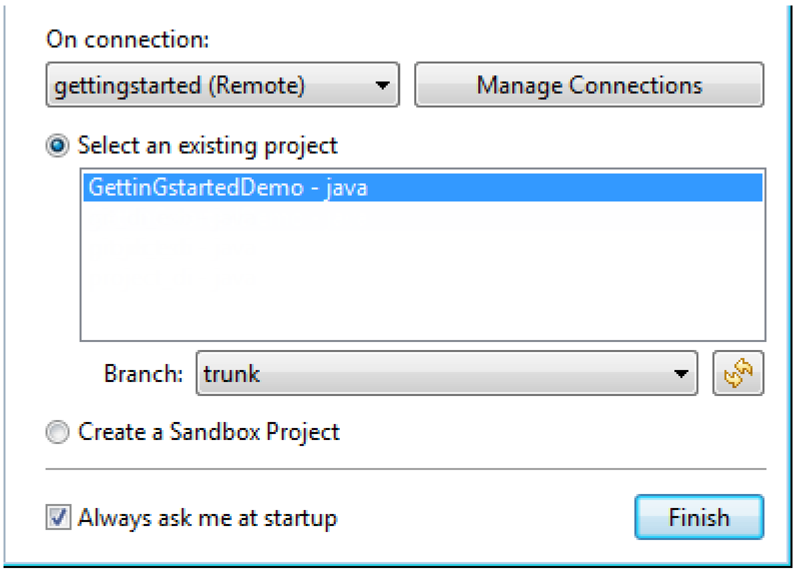


Connections

Local	✓
gettingstarted	✓

Repository: Remote
Name: gettingstarted
Description:
User Name: jsmith@company.com
User Password: ●●●●●●
Workspace: C:\Talend\studio\workspace
Web-app Url: http://localhost:8081/org.talend.administrator/

2. Click the **[+]** button under the **Connections** area to create a new connection, select **Remote** from the **Repository** drop-down list, and enter a name for the connection in the **Name** field.
3. Provide the following Talend Administration Center logon information:
   a) In the **User Name** and **User Password** fields, enter a user name defined and authorized in the Talend Administration Center and the corresponding password. Note that the logon credential information is case sensitive.
   b) In the **Web-app Url** field, enter the **URL** of the Talend Administration Center.
4. Click the **Check url** button to verify the Talend Administration Center connectivity.
5. If the Talend Administration Center connectivity is successful, click **OK** to create the remote connection. Otherwise, check the user name, password, and/or the Talend Administration Center URL according to the prompt message, and try again.
6. In the Talend Studio login window, select the remote connection you just created, select the project you want to open from the **Select an existing project** list, select the project branch you want to work on from the **Branch** list, and then click **Finish** to open the project.



On connection: gettingstarted (Remote)

[✓ Select an existing project]

GettinGstartedDemo - java

Branch: trunk

Create a Sandbox Project

[✓] Always ask me at startup

7. Depending on the license you are using, either of the following opens:
   - the Quick Tour. Play it to get more information on the User Interface of the Studio, and click **Stop** to end it.
   - the Welcome page. Follow the links to get more information about the Studio, and click **Start Now!** to close the page and continue opening the Studio.

   Tip:

   After your Studio successfully launches, you can also click the **Videos** link on the top of the Studio main window to watch a couple of short videos that help you get started with your Talend Studio. For some operating systems, you may need to install an MP4 decoder/player to play the videos.

Now you have successfully logged in to the Talend Studio. Next you need to install additional packages required for the Talend Studio to work properly.

Installing additional packages

Talend recommends that you install additional packages, including third-party libraries and database drivers, as soon as you log in to your Talend Studio to allow you to fully benefit from the functionalities of the Studio.

1. When the **Additional Talend Packages** wizard opens, install additional packages by selecting the **Required** and **Optional third-party libraries** check boxes and clicking **Finish**.

   This wizard opens each time you launch the studio if any additional package is available for installation unless you select the **Do not show this again** check box. You can also display this wizard by selecting **Help** > **Install Additional Packages** from the menu bar.

   For more information, see the section about installing additional packages in the Talend Real-time Big Data Platform Installation Guide
2. In the **Download external modules** window, click the **Accept all** button at the bottom of the wizard to accept all the licenses of the external modules used in the studio.

   Depending on the libraries you selected, you may need to accept their license more than once.

   Wait until all the libraries are installed before starting to use the studio.
3. If required, restart your Talend Studio for certain additional packages to take effect.

Setting up Hadoop connection manually

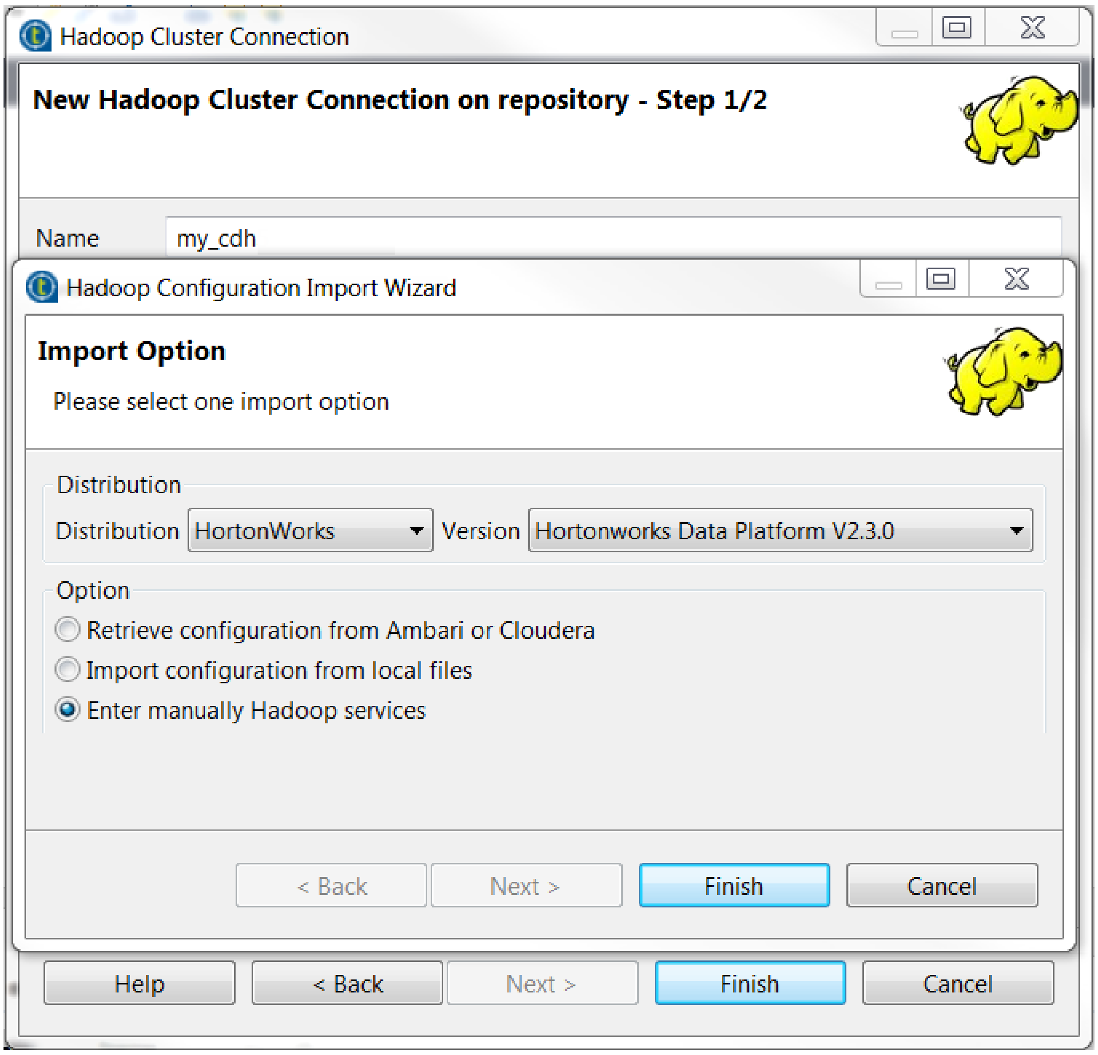
Setting up the connection to a given Hadoop distribution in the **Repository** allows you to avoid configuring that connection each time when you need to use the same Hadoop distribution.

- Ensure that the client machine on which the Talend Studio is installed can recognize the host names of the nodes of the Hadoop cluster to be used. For this purpose, add the IP address/hostname mapping entries for the services of that Hadoop cluster in the `hosts` file of the client machine.

  For example, if the host name of the Hadoop Namenode server is talend-cdh550.weave.local, and its IP address is 192.168.x.x, the mapping entry reads `192.168.x.x talend-cdh550.weave.local`.
- The Hadoop cluster to be used has been properly configured and is running.
- The **Integration** perspective is active.

The Cloudera Hadoop cluster to be used in this example is of the CDH V5.5 in the Yarn mode and applies the default configuration of the distribution without enabling the Kerberos security. For further information about the default configuration of the CDH V5.5 distribution, see Deploy CDH 5 on a cluster and Default ports used in CDH5.

1. In the **Repository** tree view of your studio, expand **Metadata** and then right-click **Hadoop cluster**.
2. Select **Create Hadoop cluster** from the contextual menu to open the **Hadoop cluster connection** wizard.
3. Fill in generic information about this connection, such as **Name** and **Description** and click **Next** to open the **Hadoop configuration import wizard** that helps you import the ready-for-use configuration if any.
4. Select the **Enter manually Hadoop services** check box to manually enter the configuration information for the Hadoop connection being created.


Hadoop Cluster Connection

New Hadoop Cluster Connection on repository - Step 1/2

Name: my_cdh

Hadoop Configuration Import Wizard

Import Option

Please select one import option

Distribution

Distribution: HortonWorks — Version: Hortonworks Data Platform V2.3.0

Option

- ○ Retrieve configuration from Ambari or Cloudera
- ○ Import configuration from local files
- ● Enter manually Hadoop services

5. Click **Finish** to close this import wizard.
6. From the **Distribution** list, select **Cloudera** and then from the **Version** list, select **Cloudera CDH5.5 (YARN mode)**.
7. In the **Namenode URI** field, enter the URI pointing to the machine used as the NameNode service of the Cloudera Hadoop cluster to be used.

The NameNode is the master node of a Hadoop system. For example, assume that you have chosen a machinecalled machine1 as the NameNode, then the location to be entered is `hdfs://machine1:portnumber`.

On the cluster side, the related property is specified in the configuration file called `core-site.xml`. If you do not know what URI is to be entered, check the `fs.defaultFS` property in the `core-site.xml` file of your cluster.

8. In the **Resource manager** field and the **Resource manager scheduler** field, enter the URIs pointing to these two services, respectively.

   On the cluster side, these two services share the same host machine but use different default portnumbers. For example, if the machine hosting them is resourcemanager.company.com, the location of the Resource manager is `resourcemanager.company.com:8032` and the location of the Resource manager scheduler is `resourcemanager.company.com:8030`.

   If you do not know the name of the hosting machine of these services, check the `yarn.resourcemanager.hostname` property in the configuration file called `yarn-site.xml` of your cluster.
9. In the **Job history** field, enter the location of the JobHistory service. This service allows the metrics information of the current Job to be stored in the JobHistory server.

   The related property is specified in the configuration file called `mapred-site.xml` of your cluster. For the value you need to put in this field, check the `mapreduce.jobhistory.address` property in this `mapred-site.xml` file.
10. In the **Staging directory** field, enter this directory defined in your Hadoop cluster for temporary files created by running programs.

    The related property is specified in the `mapred-site.xml` file of your cluster. For further information, check the `yarn.app.mapreduce.am.staging-dir` property in this `mapred-site.xml` file.

11. Select the **Use datanode hostname** check box to allow the Studio to access each Datanode of your cluster via their host names.

    This actually sets the `dfs.client.use.datanode.hostname` property of your cluster to `true`.
12. In the **User name** field, enter the user authentication name you want the Studio to use to connect to your Hadoop cluster.
13. Since the Hadoop cluster to be connected to is using the default configuration, leave the other fields or check boxes in this wizard as they are because they are used to define any custom Hadoop configuration.
14. Click the **Check services** button to verify that the Studio can connect to the NameNode and the ResourceManager services you have specified.

    A dialog box pops up to indicate the checking process and the connection status.

    If the connection fails, you can click **Error log** at the end of each progress bar to diagnose the connection issues.
15. Once this check indicates that the connection is successful, click **Finish** to validate your changes and close the wizard.

The new connection, called my_cdh in this example, is displayed under the **Hadoop cluster** folder in the **Repository** tree view.

You can then continue to create the child connections to different Hadoop elements such as HDFS or Hive based on this connection.

Setting up connection to HDFS

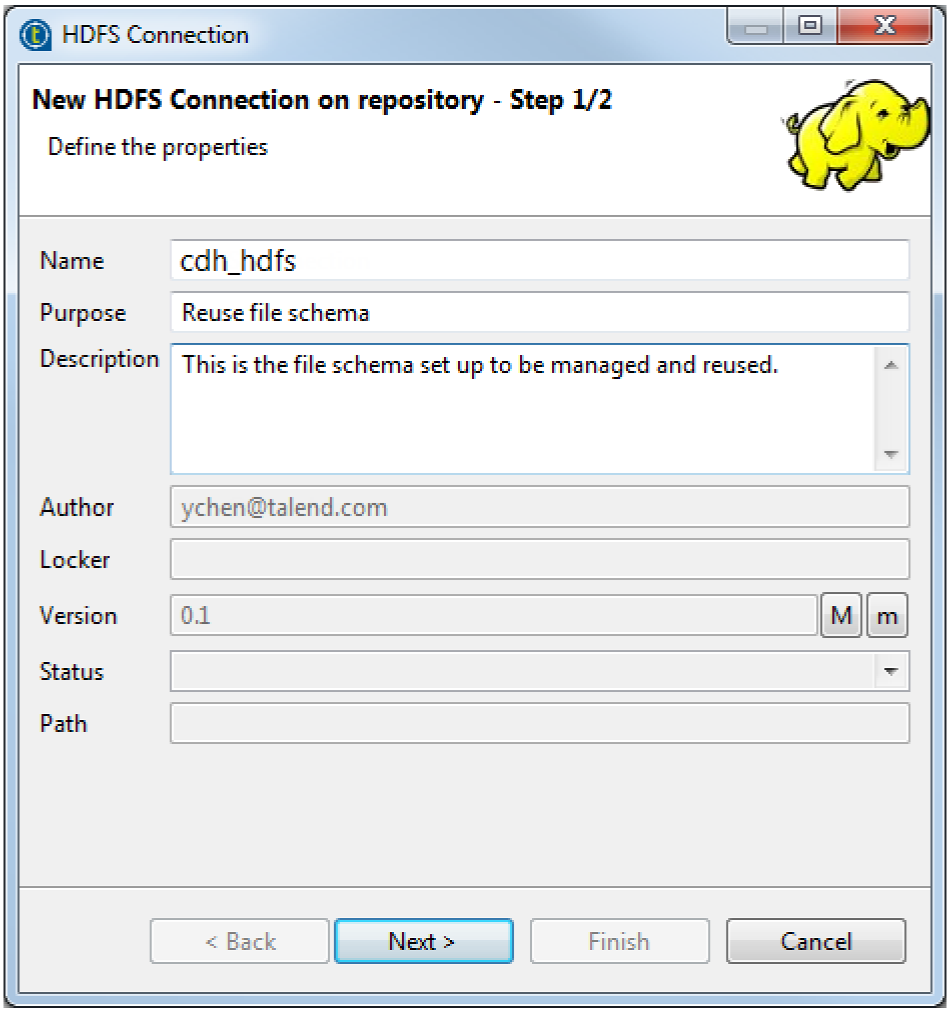
A connection to HDFS in the **Repository** allows you to reuse this connection in related Jobs.

- The connection to the Hadoop cluster hosting the HDFS system to be used has been set up from the **Hadoop cluster** node in the **Repository**.

  For further information about how to create this connection, see Setting up Hadoop connection manually on page 19.
- The Hadoop cluster to be used has been properly configured and is running and you have the proper access permission to that distribution and its HDFS.
- Ensure that the client machine on which the Talend Studio is installed can recognize the host names of the nodes of the Hadoop cluster to be used. For this purpose, add the IP address/hostname mapping entries for the services of that Hadoop cluster in the `hosts` file of the client machine.

  For example, if the host name of the Hadoop Namenode server is talend-cdh550.weave.local, and its IP address is 192.168.x.x, the mapping entry reads `192.168.x.x talend-cdh550.weave.local`.

1. Expand the **Hadoop cluster** node under **Metadata** in the **Repository** tree view, right click the Hadoop connection to be used and select **Create HDFS** from the contextual menu.
2. In the connection wizard that opens up, fill in the generic properties of the connection you need create, such as **Name**, **Purpose** and **Description**.



New HDFS Connection on repository - Step 1/2

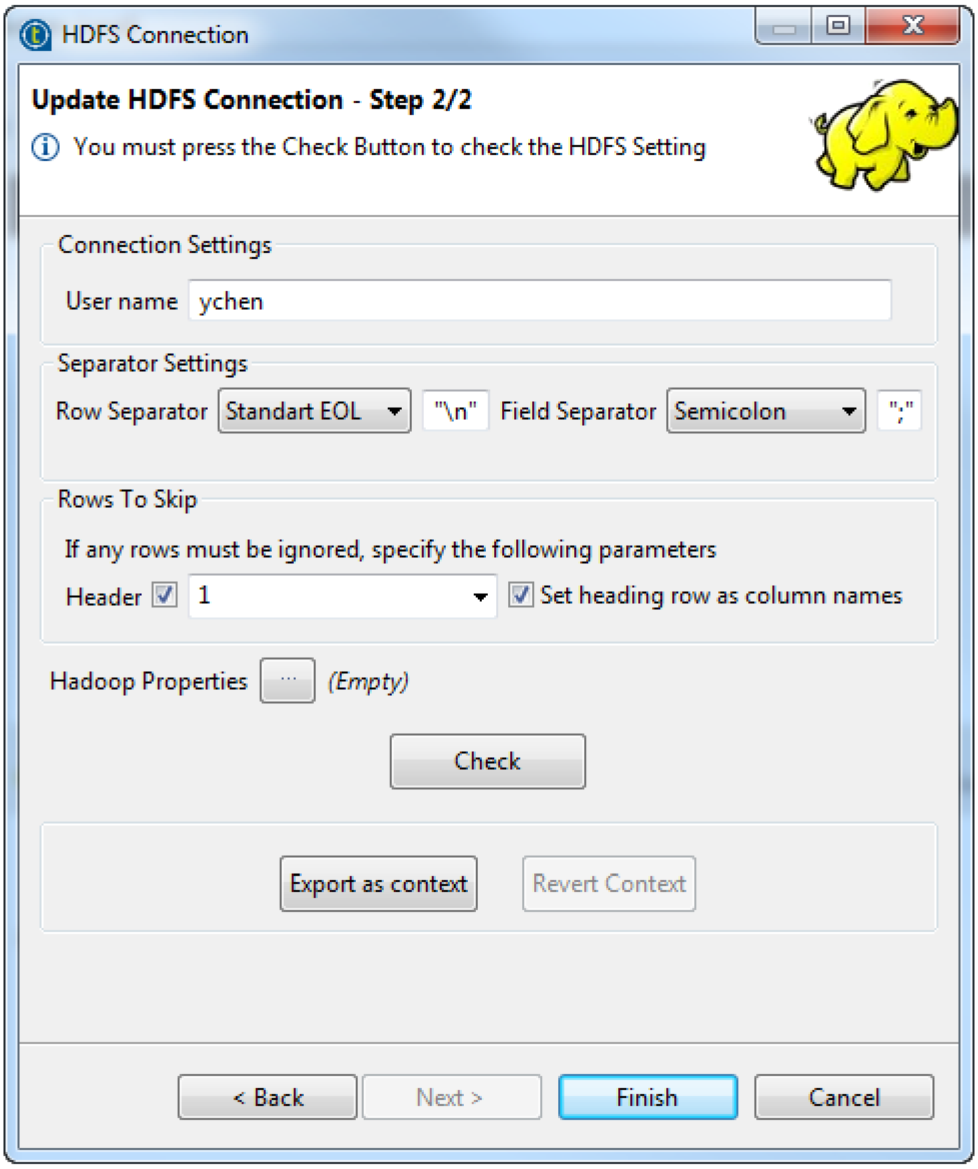
Define the properties

Name	cdh_hdfs
Purpose	Reuse file schema
Description	This is the file schema set up to be managed and reused.
Author	ychen@talend.com
Locker	
Version	0.1
Status	
Path	

3. Click **Next** when completed. The second step requires you to fill in the HDFS connection data.

   The **User name** property is automatically pre-filled with the value inherited from the Hadoop connection you selected in the previous steps.

   The **Row separator** and the **Field separator** properties are using the default values.


HDFS Connection

Update HDFS Connection - Step 2/2

You must press the Check Button to check the HDFS Setting

Connection Settings

User name: ychen

Separator Settings

Row Separator: Standart EOL "\n" Field Separator: Semicolon ";"

Rows To Skip

If any rows must be ignored, specify the following parameters

Header ☑ 1 ☑ Set heading row as column names

Hadoop Properties ... (Empty)

4. Select the **Set heading row as column names** check box to use the data in the heading rows of the HDFS file to be used to define the column names of this file.

   The **Header** check box is then automatically selected and the **Header** field is filled with `1`. This means that the first row of the file will be ignored as data body but used as column names of the file.
5. Click **Check** to verify your connection.

   A message pops up to indicate whether the connection is successful.
6. Click **Finish** to validate these changes.

The new HDFS connection is now available under the **Hadoop cluster** node in the **Repository** tree view. You can then use it to define and centralize the schemas of the files stored in the connected HDFS system in order to reuse these schemas in a Talend Job.

Uploading files to HDFS

Uploading a file to HDFS allows the Big Data Jobs to read and process it.

In this procedure, you will create a Job that writes data in the HDFS system of the Cloudera Hadoop cluster to which the connection has been set up in the **Repository** as explained in Setting up Hadoop connection manually on page 19. This data is needed for the use case described in Performing data integration tasks for Big Data. For the files needed for the use case, download `tprtbd_gettingstarted_source_files.zip` from the **Downloads** tabof the online version of this page at https://help.talend.com.

- The connection to the Hadoop cluster to be used and the connection to the HDFS system of this cluster have been set up from the **Hadoop cluster** node in the **Repository**.

  If you have not done so, see Setting up Hadoop connection manually on page 19 and then Setting up connection to HDFS on page 22 to create these connections.
- The Hadoop cluster to be used has been properly configured and is running and you have the proper access permission to that distribution and the HDFS folder to be used.
- Ensure that the client machine on which the Talend Jobs are executed can recognize the host names of the nodes of the Hadoop cluster to be used. For this purpose, add the IP address/hostname mapping entries for the services of that Hadoop cluster in the `hosts` file of the client machine.

  For example, if the host name of the Hadoop Namenode server is talend-cdh550.weave.local, and its IP address is 192.168.x.x, the mapping entry reads `192.168.x.x talend-cdh550.weave.local`.
- You have launched your Talend Studio and opened the **Integration** perspective.

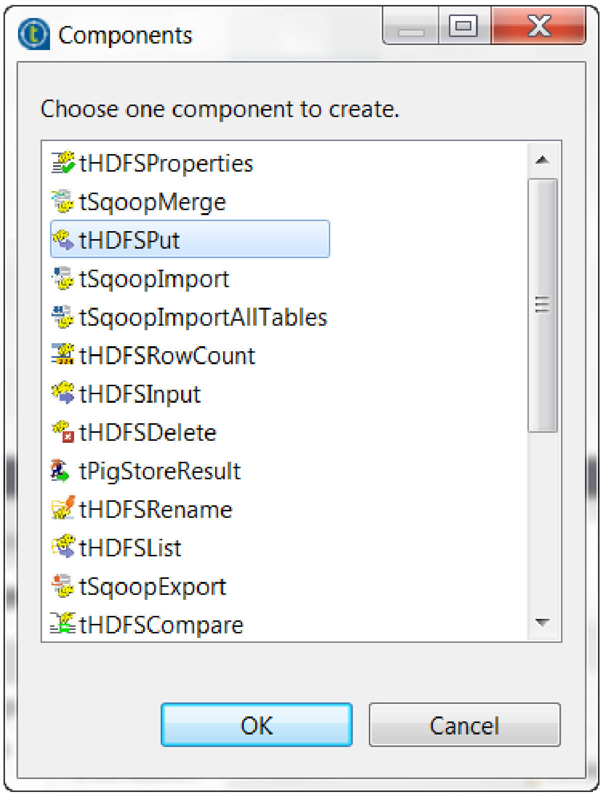
1. In the **Repository** tree view, expand the **Job Designs** node, right-click the **Standard** node, and select **Create folder** from the contextual menu.
2. In the **New Folder** wizard, name your Job folder `getting_started` and click **Finish** to create your folder.
3. Right-click the **getting_started** folder and select **Create Standard Job** from the contextual menu.
4. In the **New Job** wizard, give a name to the Job you are going to create and provide other useful information if needed.

   For example, enter `write_to_hdfs` in the **Name** field.

   In this step of the wizard, **Name** is the only mandatory field. The information you provide in the **Description** field will appear as hover text when you move your mouse pointer over the Job in the **Repository** tree view.
5. Click **Finish** to create your Job.

   An empty Job is opened in the Studio.
6. Expand the **Hadoop cluster** node under **Metadata** in the **Repository** tree view.
7. Expand the Hadoop connection you have created and then the **HDFS** folder under it. In this example, it is the **my_cdh** Hadoop connection.
8. Drop the HDFS connection from the **HDFS** folder into the workspace of the Job you are creating. This connection is **cdh_hdfs** in this example.

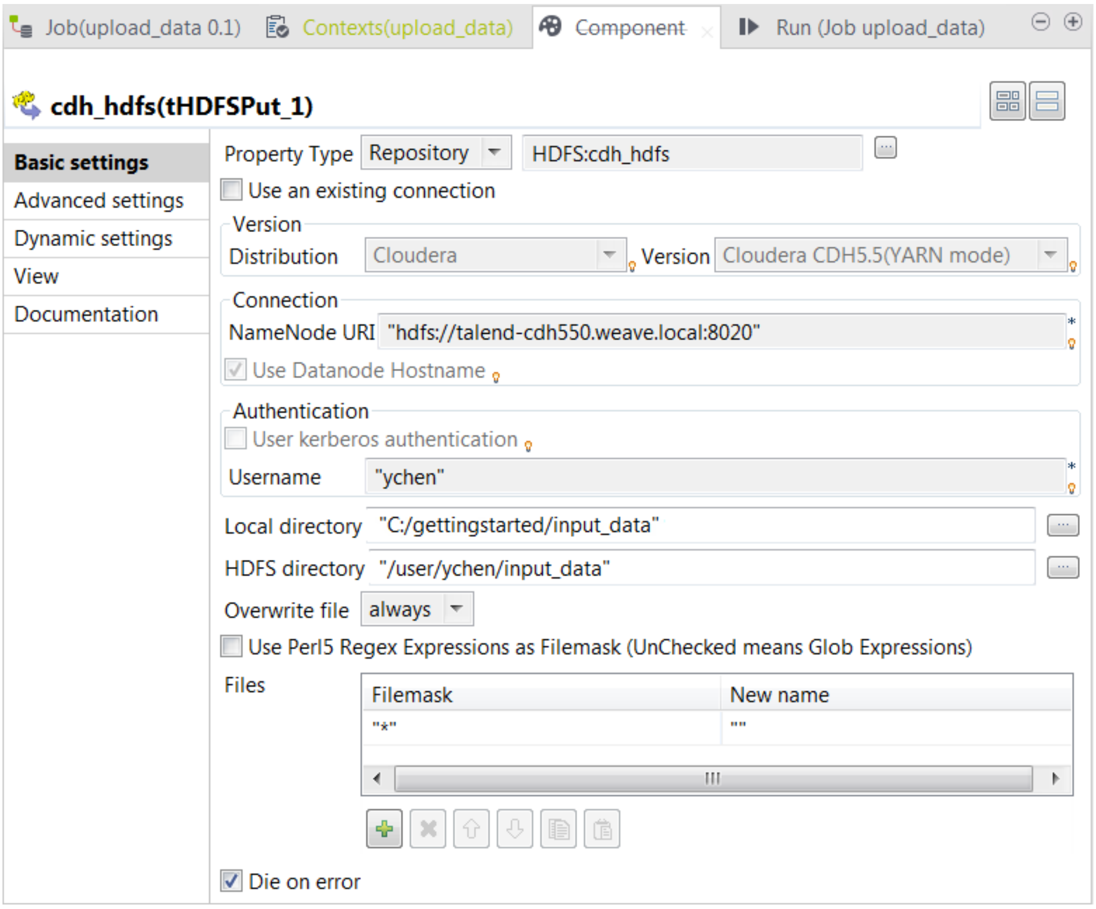
   The **Components** window is displayed to show all the components that can directly reuse this HDFS connection in a Job.



9. Select tHDFSPut and click **OK** to validate your choice.

This **Components** window is closed and a tHDFSPut component is automatically placed in the workspace of the current Job, with this component having been labelled using the name of the HDFS connection mentioned in the previous step.

10. Double-click tHDFSPut to open its **Component** view.



cdh_hdfs(tHDFSPut_1)

Property Type: Repository | HDFS:cdh_hdfs

☐ Use an existing connection

Version
Distribution: Cloudera | Version: Cloudera CDH5.5(YARN mode)

Connection
NameNode URI: "hdfs://talend-cdh550.weave.local:8020"
✓ Use Datanode Hostname

Authentication
☐ User kerberos authentication
Username: "ychen"

Local directory: "C:/gettingstarted/input_data"
HDFS directory: "/user/ychen/input_data"
Overwrite file: always
☐ Use Perl5 Regex Expressions as Filemask (UnChecked means Glob Expressions)

Files

Filemask	New name
"*"	""

✓ Die on error

The connection to the HDFS system to be used has been automatically configured by using the configuration of the HDFS connection you have set up and stored in the **Repository**. The related parameters in this tab therefore becomes read-only. These parameters are: **Distribution**, **Version**, **NameNode URI**, **Use Datanode Hostname**, **User kerberos authentication** and **Username**.

11. In the **Local directory** field, enter the path, or browse to the folder in which the files to be copied to HDFS are stored.

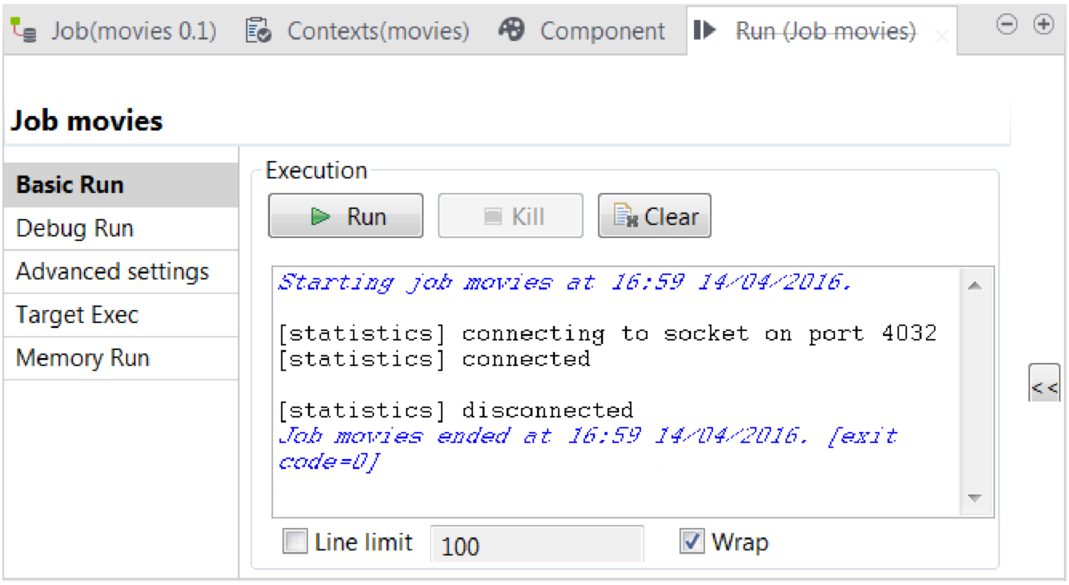
    The files about movies and their directors are stored in this directory.

12. In the **HDFS directory** field, enter the path, or browse to the target directory in HDFS to store the files.

    This directory is created on the fly if it does not exist.

13. From the **Overwrite file** drop-down list, select **always** to overwrite the files if they already exist in the target directory in HDFS.
14. In the **Files** table, add one row by clicking the **[+]** button in order to define the criteria to select the files to be copied.
15. In the **Filemask** column, enter an asterisk (*) within the double quotation marks to make tHDFSPut select all the files stored in the folder you specified in the **Local directory** field.
16. Leave the **New name** column empty, that is to say, keep the default double quotation marks as is, so as to make the name of the files unchanged after being uploaded.
17. Press **F6** to run the Job.

    The **Run** view is opened automatically. It shows the progress of this Job.

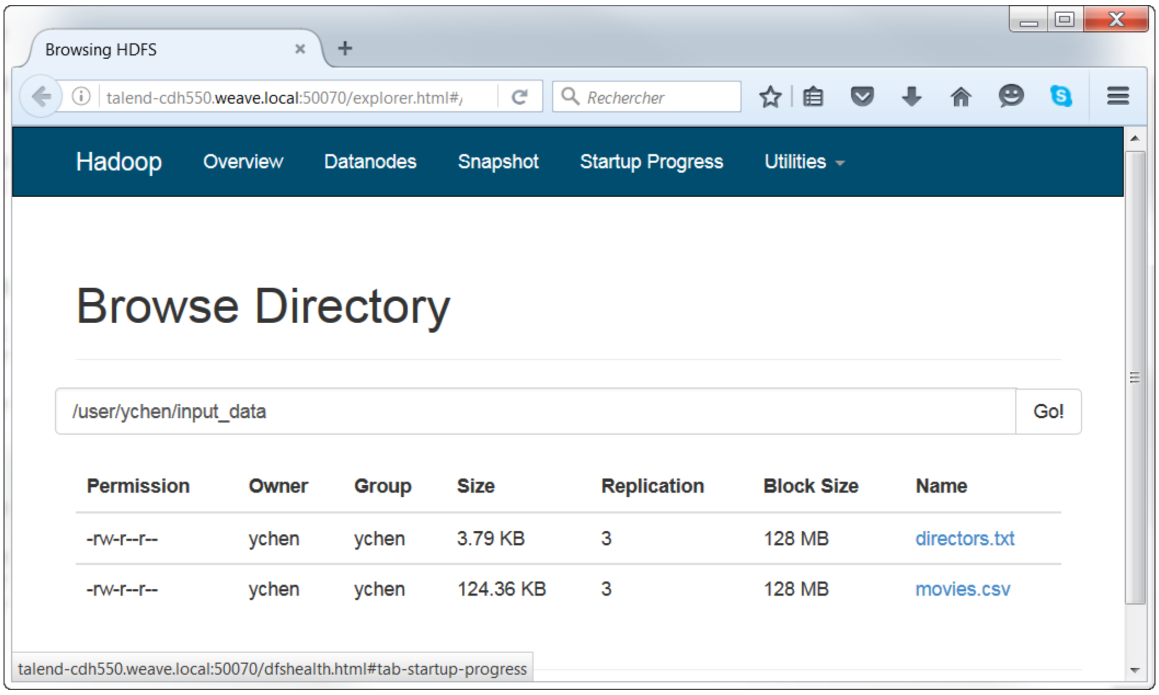


```
Starting job movies at 16:59 14/04/2016.

[statistics] connecting to socket on port 4032
[statistics] connected

[statistics] disconnected
Job movies ended at 16:59 14/04/2016. [exit
code=0]
```

When the Job is done, the files you uploaded can be found in HDFS in the directory you have specified.



Permission	Owner	Group	Size	Replication	Block Size	Name
-rw-r--r--	ychen	ychen	3.79 KB	3	128 MB	directors.txt
-rw-r--r--	ychen	ychen	124.36 KB	3	128 MB	movies.csv

Preparing file metadata

In the **Repository**, setting up the metadata of a file stored in HDFS allows you to directly reuse its schema in a related Big Data component without having to define each related parameter manually.

Since the `movies.csv` file you need to process has been stored in the HDFS system being used, you can retrieve its schema to set up its metadata in the **Repository**.

The schema of the `directors.txt` file can also be retrieved, but is intentionally ignored in the retrieval procedure explained below, because in this scenario, this `directors.txt` file is used to demonstrate how to manually define a schema in a Job.

- You have launched your Talend Studio and opened the **Integration** perspective.

- The source files `movies.csv` and `directors.txt` have been uploaded into HDFS as explained in Uploading files to HDFS on page 24.
- The connection to the Hadoop cluster to be used and the connection to the HDFS system of this cluster have been set up from the **Hadoop cluster** node in the **Repository**.

  If you have not done so, see Setting up Hadoop connection manually on page 19 and then Setting up connection to HDFS on page 22 to create these connections.
- The Hadoop cluster to be used has been properly configured and is running and you have the proper access permission to that distribution and the HDFS folder to be used.
- Ensure that the client machine on which the Talend Studio is installed can recognize the host names of the nodes of the Hadoop cluster to be used. For this purpose, add the IP address/hostname mapping entries for the services of that Hadoop cluster in the `hosts` file of the client machine.

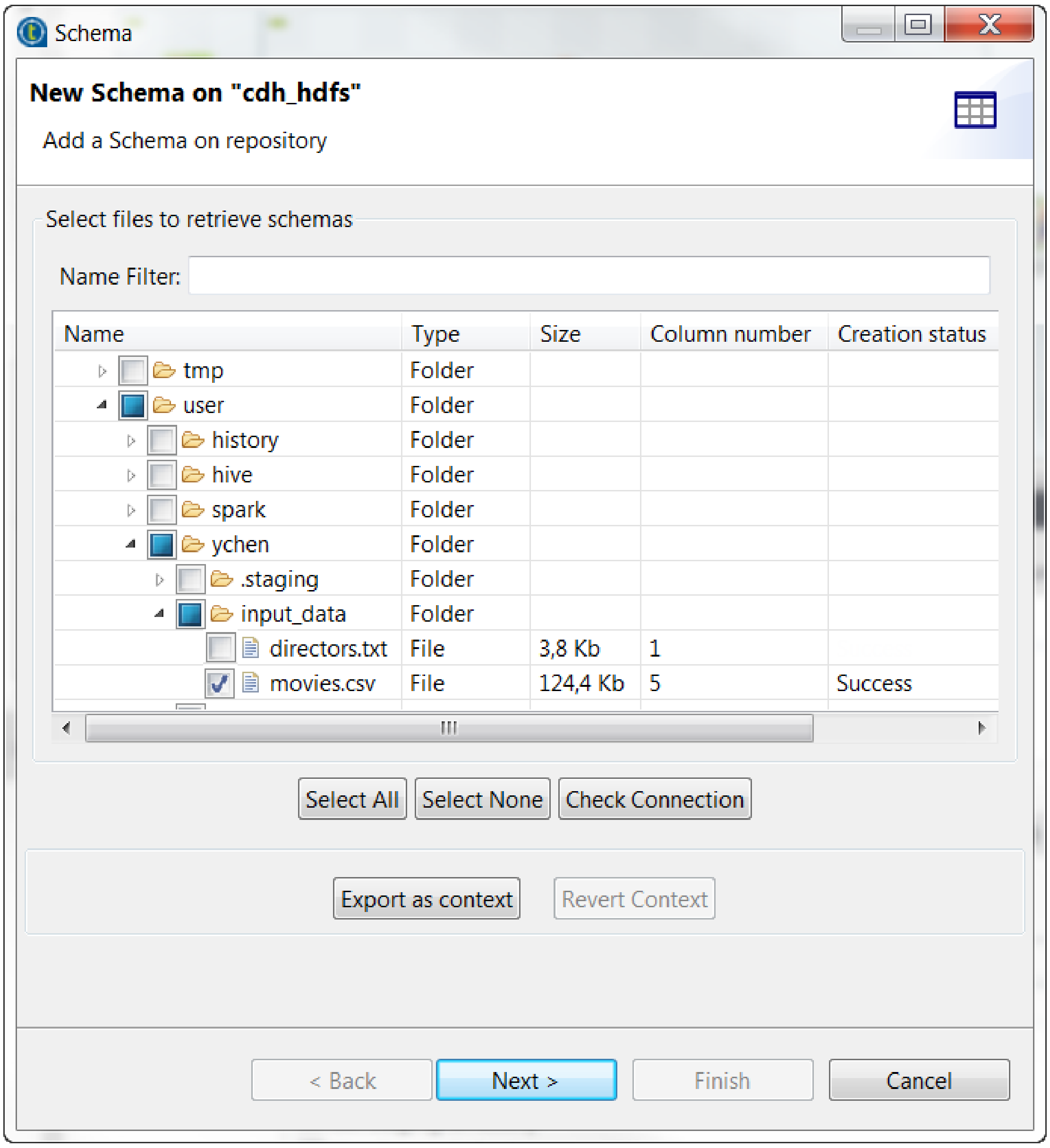
  For example, if the host name of the Hadoop Namenode server is talend-cdh550.weave.local, and its IP address is 192.168.x.x, the mapping entry reads `192.168.x.x talend-cdh550.weave.local`.

1. Expand the **Hadoop cluster** node under **Metadata** in the **Repository** tree view.
2. Expand the Hadoop connection you have created and then the **HDFS** folder under it.

   In this example, it is the **my_cdh** Hadoop connection.
3. Right click the HDFS connection in this **HDFS** folder and from the contextual menu, select **Retrieve schema**.

   In this scenario, this HDFS connection is named **cdh_hdfs**.

   A **Schema** wizard is displayed, allowing you to browse to files in HDFS.



Schema

New Schema on "cdh_hdfs"

Add a Schema on repository

Select files to retrieve schemas

Name Filter:

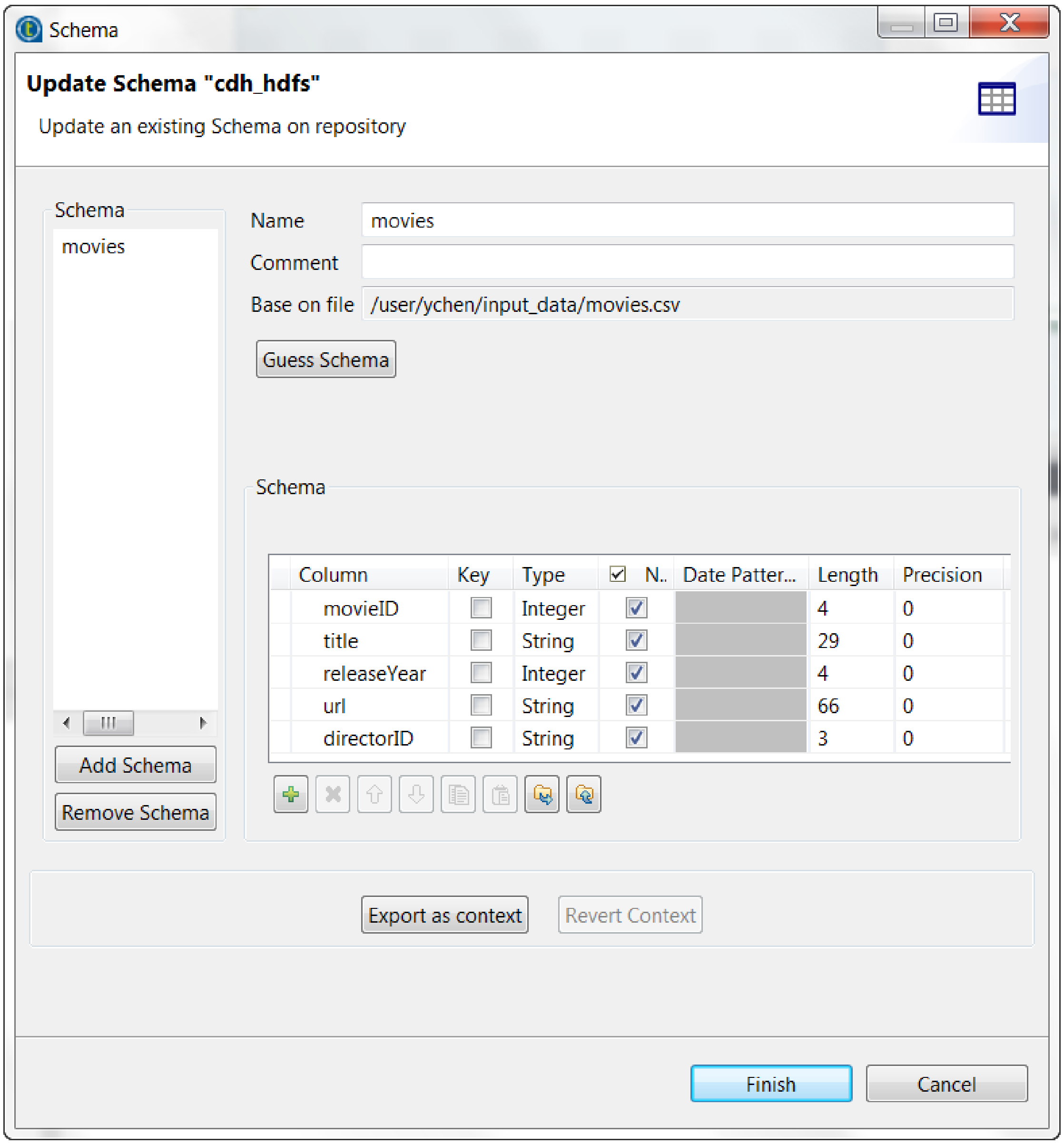
Name	Type	Size	Column number	Creation status
tmp	Folder			
user	Folder			
history	Folder			
hive	Folder			
spark	Folder			
ychen	Folder			
.staging	Folder			
input_data	Folder			
directors.txt	File	3,8 Kb	1	
✓ movies.csv	File	124,4 Kb	5	Success

4. Expand the file tree to show the `movies.csv` file, from which you need to retrieve the schema, and select it.

In this scenario, the `movies.csv` file is stored in the following directory: `/user/ychen/input_data`.

5. Click **Next** to display the retrieved schema in the wizard.

The schema of the movie data is displayed in the wizard and the first row of the data is automatically used as the column names.



Schema: movies

Name: movies

Comment:

Base on file: /user/ychen/input_data/movies.csv

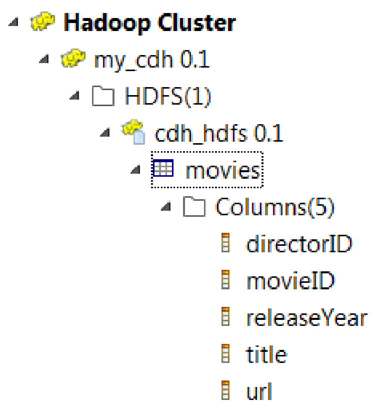
Schema

Column	Key	Type	☑ N..	Date Patter...	Length	Precision
movieID	☐	Integer	✓		4	0
title	☐	String	✓		29	0
releaseYear	☐	Integer	✓		4	0
url	☐	String	✓		66	0
directorID	☐	String	✓		3	0

If the first row of the data you are using is not used this way, you need to review how you set the **Header** configuration when you were creating the HDFS connection as explained in Setting up connection to HDFS on page 22.

6. Click **Finish** to validate these changes.

You can now see the file metadata under the HDFS connection you are using in the **Repository** tree view.



Talend Real-time Big Data Platform in use

This chapter takes the example of a company that provides movie rental and streaming video services, and shows how such a company could make use of Talend Real-time Big Data Platform.

You will work with data about movies and directors and data about your customers as you learn how to:

- validate email addresses for customers and standardize phone numbers before sending them to the Customer Support System
- upload data stored in a local file system to the HDFS file system of the company's Hadoop cluster
- join the director data to the movie data to produce a new dataset and store this dataset in the HDFS system too

Profiling and cleansing data

This example profiles US customer email addresses and phone numbers. It shows how to identify anomalies in address columns, how to use Talend Jobs to recuperate non-matching data and finally how to generate periodicevolution reports to monitor data evolution and share statistics with business users

Setting up input data

The example in this document assumes that the customer data you want to profile is stored in a MySQL database.

If you want to replicate the example and use the exact input data, you can download the `gettingstarted.sql` file of the customer data and then import it in a MySQL database.

- You have an access to a MySQL database.
- You have downloaded tprtbd_gettingstarted_source_files.zip from the **Downloads** tab of the online version of this page at https://help.talend.com, and stored the source file `gettingstarted.sql` locally.

1. Open the MySQL Workbench to launch an instance of the database.
2. From the menu bar, select **Server** > **Data Import** to open the import wizard wizard.
3. Select the **Import from Self-Contained File** option and browse to where you have stored the `gettingstarted.sql` file.
4. Select the schema to which you want to import the data, or click **New...** to define a new schema.
5. Click **Start Import** in the lower right corner.

The gettingstarted database is imported in the MySQL database.

Identifying anomalies in data

The use case explains how to use the **Profiling** perspective of the studio to analyze customer email addresses and phone numbers. It uses out-of-box indicators and patterns on the columns and shows the matching and non-matching address data.
Profiling Jobs are then generated on the analysis results to clean customer data and monitor its evolution.

You can then use the **Data Explorer** perspective to browse the non-matching data.

The sequence of profiling and cleansing customer data involves the following steps:

1. Create a column analysis on customer email addresses and phone numbers. For further information, see Defining a column analysis on page 33.
2. Connect to the database which holds the customer data from the analysis editor. For further information, see Creating the database connection on page 34.
3. Add indicators to provide simple statistics on data such as row , blank and duplicate counts. For further information, see Setting system indicators on page 36.
4. Add standard patterns against which to match email addresses and phone numbers. For further information, see Setting patterns on page 38.
5. Execute the analysis to show results in tables and charts. For further information, see Showing analysis results on page 39.
6. Access a view of the analyzed data to see invalid records. For further information, see Browsing non-match data on page 41.
7. Generate out-of-box Jobs from analysis results to remove duplicate values from the Email and Phone columns. For further information, see Removing duplicate values on page 42.
8. Generate out-of-box Jobs from analysis results to remove values which do not respect the standard email format or phone number format from the Email and Phone columns respectively. For further information, see Removing non-matching values on page 43.

Defining a column analysis

You want to create a column analysis from the **Profiling** perspective of the Studio to examine the Email and Phone columns in a MySQL databases and collect statistics on them. The analysis runs on several columns but each column is analyzed separately and independently.

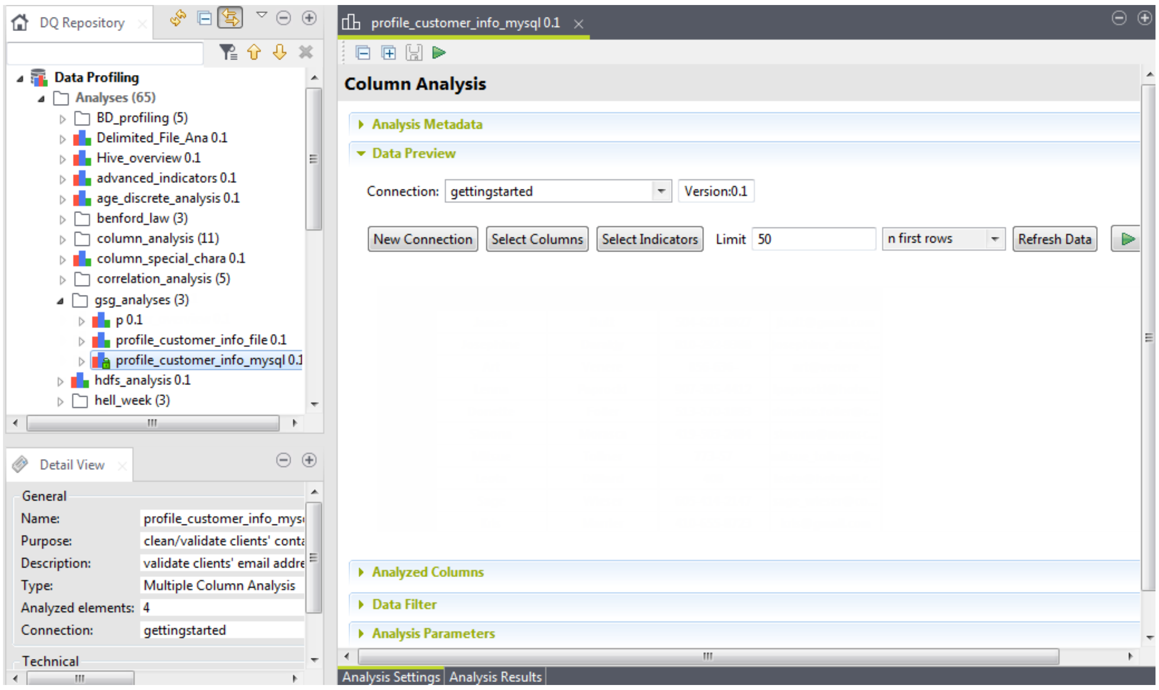
- You have opened the **Profiling** perspective in the Studio.

1. In the **DQ Repository** tree view, right-click **Analyses** and select **New Analysis**.

   The **[Create New Analysis]** wizard opens.
2. Start typing `Basic column analysis` in the search field, select **Basic Column Analysis** from the list and click **Next**.
3. In the **Name** field, enter a name for the analysis.

   The **Name** field is mandatory. Do not use spaces or special characters in the analysis name.
4. Set a purpose and a description for the analysis, and click **Finish** to open the analysis editor.

   The **Purpose** and **Description** fields are not mandatory, but you are advised to fill in this information which is displayed in **Detail View** when you select the analysis.



General

Name:	profile_customer_info_mys
Purpose:	clean/validate clients' conta
Description:	validate clients' email addre
Type:	Multiple Column Analysis
Analyzed elements:	4
Connection:	gettingstarted

Connection: gettingstarted Version:0.1

Limit 50 n first rows

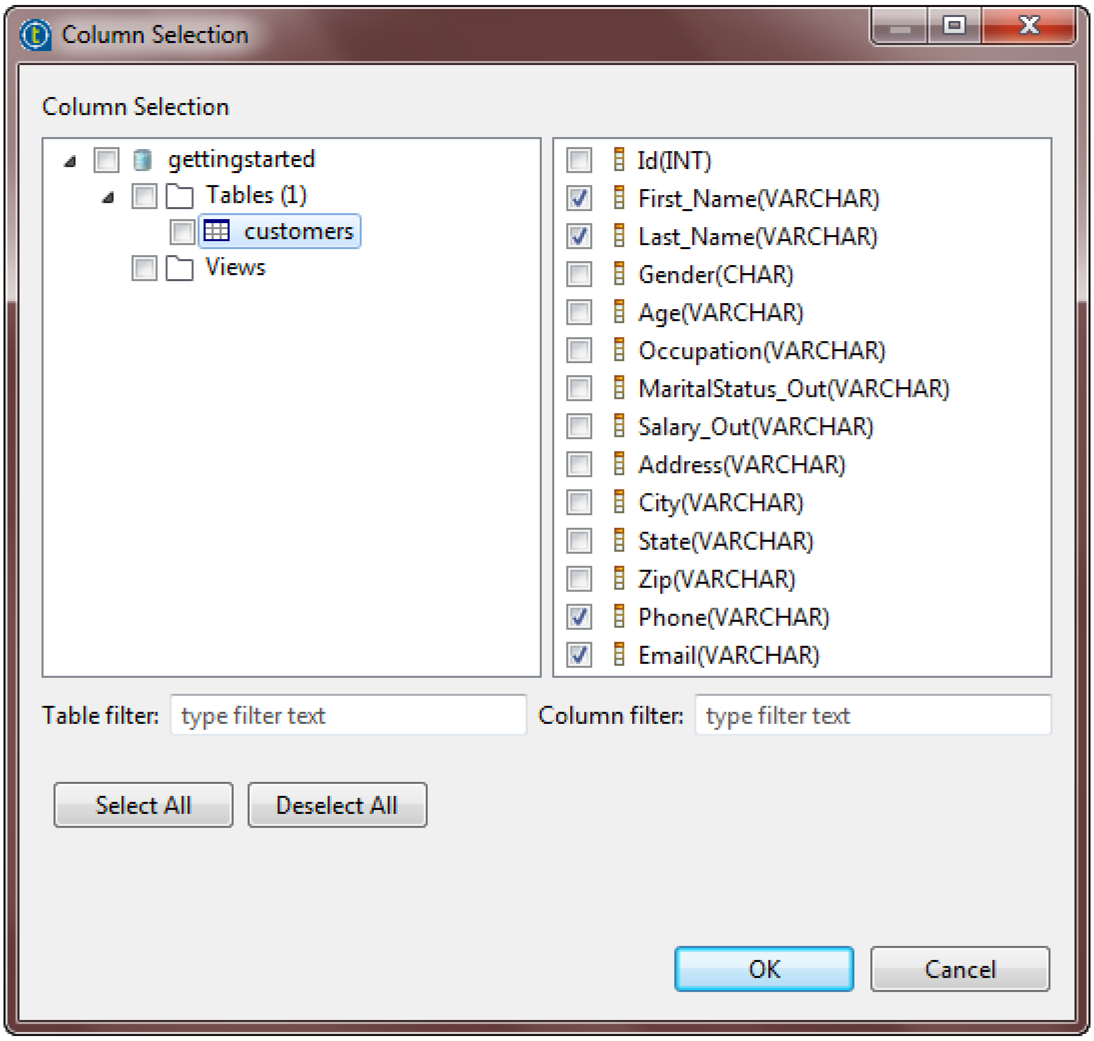
The new analysis is listed under the **Analysis** folder in the **DQ Repository** tree view.

Creating the database connection

Before you proceed to analyze customer data, stored in the MySQL database in this example, you must first set up the connection to the database.

- You have opened the **Profiling** perspective in the Studio.
- You have created a column analysis and opened the analysis editor. For further information, see Defining a column analysis on page 33.
- You have imported the `gettingstarted.sql` file which holds the customer data into a MySQL database to which you have access. For further information, see Setting up input data on page 32.

1. In the analysis editor, click the **New Connection** tab to open the **[Create New Connection]** wizard.
2. From the **Connection Type** list, select **DB connections** and click **Next**.
3. Click **Finish** to create the database connection, list it under the **Metadata** node and open a new step in the wizard.
4. Expand the database connection, click on the table name and select the check boxes of the columns on which you want to create the analysis.



Column Selection

Column Selection

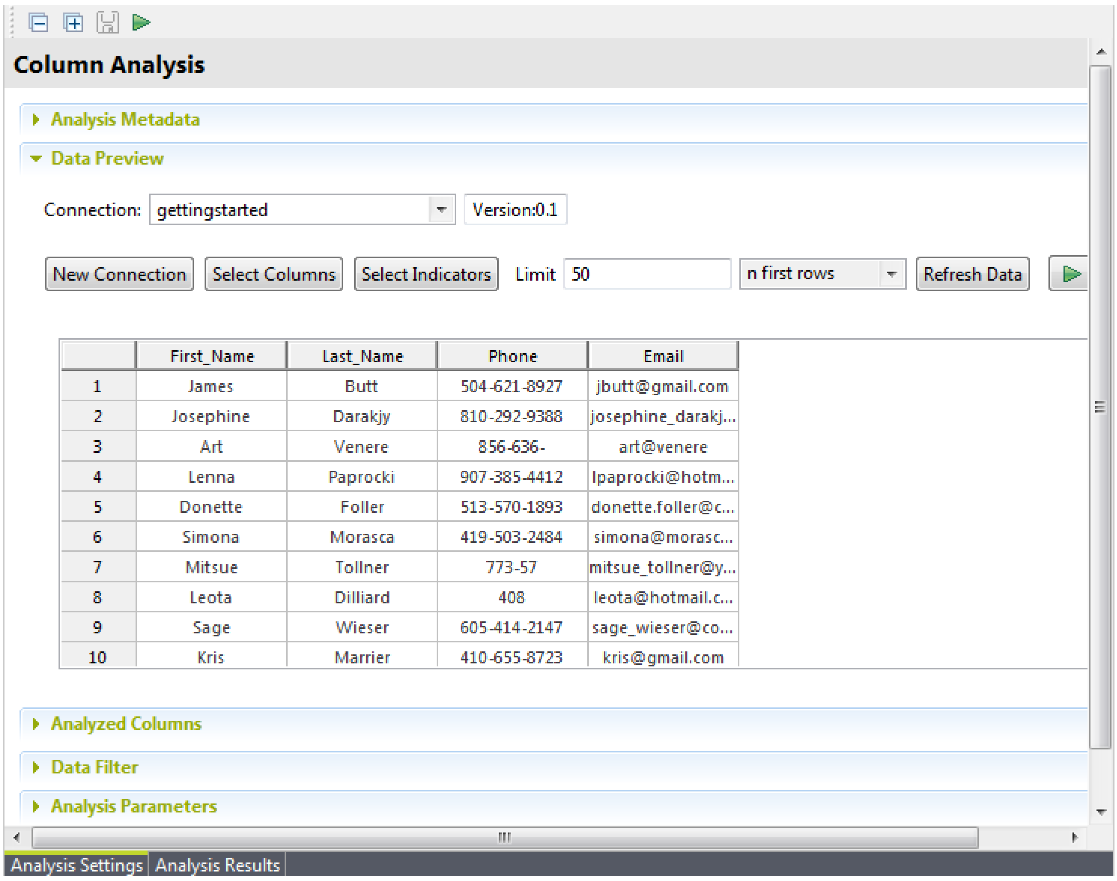
- gettingstarted
  - Tables (1)
    - customers
  - Views

- [ ] Id(INT)
- [✓] First_Name(VARCHAR)
- [✓] Last_Name(VARCHAR)
- [ ] Gender(CHAR)
- [ ] Age(VARCHAR)
- [ ] Occupation(VARCHAR)
- [ ] MaritalStatus_Out(VARCHAR)
- [ ] Salary_Out(VARCHAR)
- [ ] Address(VARCHAR)
- [ ] City(VARCHAR)
- [ ] State(VARCHAR)
- [ ] Zip(VARCHAR)
- [✓] Phone(VARCHAR)
- [✓] Email(VARCHAR)

Table filter: type filter text Column filter: type filter text

5. Click **OK** to close the wizard and list the columns in the analysis editor.

You can click **Refresh Data** to display the actual data in the analysis editor.



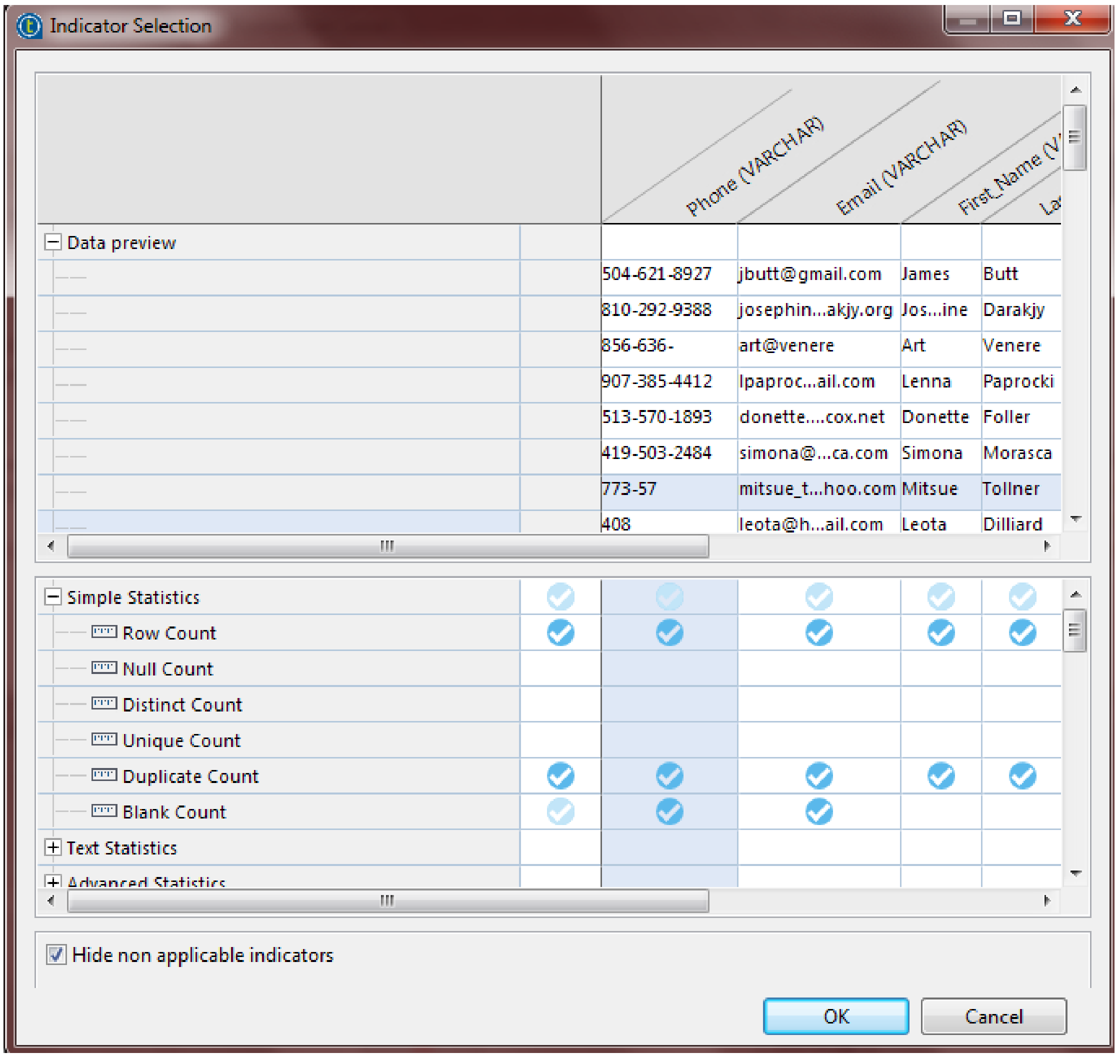
	First_Name	Last_Name	Phone	Email
1	James	Butt	504-621-8927	jbutt@gmail.com
2	Josephine	Darakjy	810-292-9388	josephine_darakj...
3	Art	Venere	856-636-	art@venere
4	Lenna	Paprocki	907-385-4412	lpaprocki@hotm...
5	Donette	Foller	513-570-1893	donette.foller@c...
6	Simona	Morasca	419-503-2484	simona@morasc...
7	Mitsue	Tollner	773-57	mitsue_tollner@y...
8	Leota	Dilliard	408	leota@hotmail.c...
9	Sage	Wieser	605-414-2147	sage_wieser@co...
10	Kris	Marrier	410-655-8723	kris@gmail.com

Setting system indicators

This column analysis uses out-of-box indicators to provide simple statistics such as row, blank and duplicate counts on the Email and Phone columns.

- You have opened the **Profiling** perspective in the Studio.
- You have created a column analysis and defined the connection to the database. For further information, see Defining a column analysis on page 33 and Creating the database connection on page 34 respectively.

1. In the **Data Preview** section in the analysis editor, click **Select indicators** to open the **[Indicator Selection]** dialog box.



		Phone (VARCHAR)	Email (VARCHAR)	First_Name (V	La
Data preview					
		504-621-8927	jbutt@gmail.com	James	Butt
		810-292-9388	josephin...akjy.org	Jos...ine	Darakjy
		856-636-	art@venere	Art	Venere
		907-385-4412	lpaproc...ail.com	Lenna	Paprocki
		513-570-1893	donette....cox.net	Donette	Foller
		419-503-2484	simona@...ca.com	Simona	Morasca
		773-57	mitsue_t...hoo.com	Mitsue	Tollner
		408	leota@h...ail.com	Leota	Dilliard

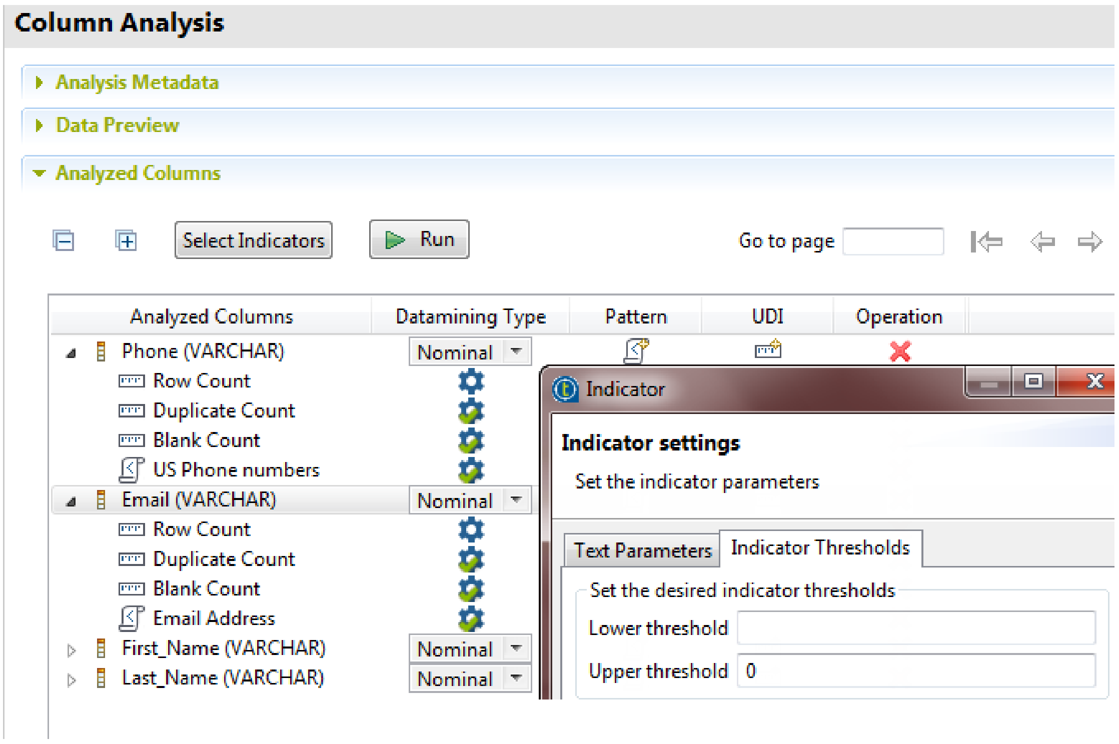
Simple Statistics	✓	✓	✓	✓	✓
Row Count	✓	✓	✓	✓	✓
Null Count					
Distinct Count					
Unique Count					
Duplicate Count	✓	✓	✓	✓	✓
Blank Count	✓	✓	✓		
Text Statistics					
Advanced Statistics					

[✓] Hide non applicable indicators

2. Expand **Simple Statistics** and select **Row Count**, **Blank Count** and **Duplicate Count**. Click **OK** to close the wizard.

   You want to see the row, blank and duplicate counts in the `Email` and `Phone` columns to see how consistent the data is.

   Indicators are added accordingly to the columns in the **Analyzed Columns** section.



Analyzed Columns	Datamining Type	Pattern	UDI	Operation
Phone (VARCHAR)	Nominal			
Row Count				
Duplicate Count				
Blank Count				
US Phone numbers				
Email (VARCHAR)	Nominal			
Row Count				
Duplicate Count				
Blank Count				
Email Address				
First_Name (VARCHAR)	Nominal			
Last_Name (VARCHAR)	Nominal			

Indicator settings

Set the indicator parameters

Set the desired indicator thresholds

Lower threshold

Upper threshold 0

3. Click the icon next to the **Duplicate Count** and **Blank Count** indicator and set `0` in the **Upper threshold** field.

   Defining thresholds on the `Email` and `Phone` columns is very helpful as it will write in red the count of the duplicate and blank values in the analysis results.

Setting patterns

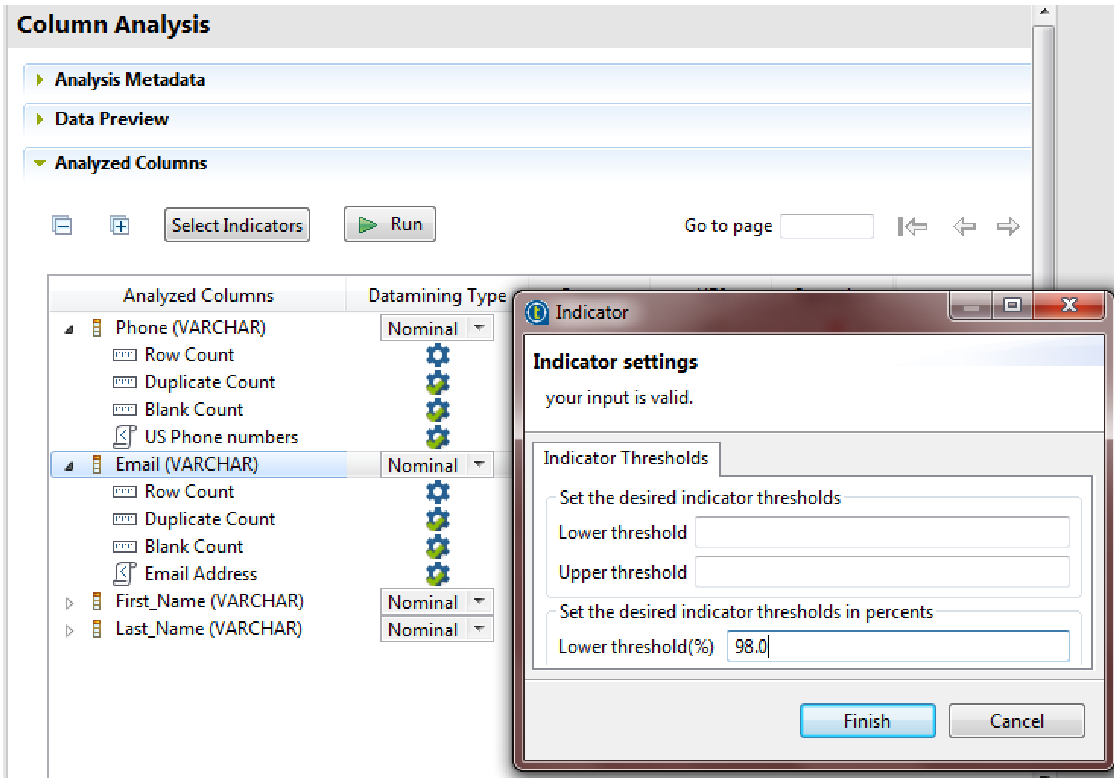
This column analysis uses predefined patterns to match the content of the Email and Phone columns against standard email and US phone patterns respectively. This defines the content, structure and quality of emails and phone numbers and give a percentage of the data that match the standard formats and the data that does not match.

- You have opened the **Profiling** perspective in the Studio.
- You have created a column analysis and defined the connection to the database. For further information, see Defining a column analysis on page 33 and Creating the database connection on page 34 respectively.

1. In the **Data Preview** section in the analysis editor, click the icon next to the `Email` column to open the **[Pattern Selector]** dialog box.
2. Expand **Regex** > **internet**, select the **Email Address** check box and click **OK** to close the dialog box.

   The pattern is added to the column in the **Analyzed Columns** section.
3. Click the icon next to the `Phone` column to open the **[Pattern Selector]** dialog box.
4. Expand **Regex** > **phone**, select the **US phone numbers** check box and click **OK** to close the dialog box.

   The pattern is added to the column in the **Analyzed Columns** section.
5. Click the icon next to the **Email Address** and **US phone numbers** patterns and set `98.0` in the **Lower threshold (%)** fields.


Column Analysis

Analysis Metadata

Data Preview

Analyzed Columns

Analyzed Columns	Datamining Type
Phone (VARCHAR)	Nominal
Row Count	
Duplicate Count	
Blank Count	
US Phone numbers	
Email (VARCHAR)	Nominal
Row Count	
Duplicate Count	
Blank Count	
Email Address	
First_Name (VARCHAR)	Nominal
Last_Name (VARCHAR)	Nominal

Indicator

Indicator settings

your input is valid.

Indicator Thresholds

Set the desired indicator thresholds

Lower threshold

Upper threshold

Set the desired indicator thresholds in percents

Lower threshold(%) 98.0

If the number of the records that match the patterns is fewer than 98%, it will be written in red in the analysis results.

Showing analysis results

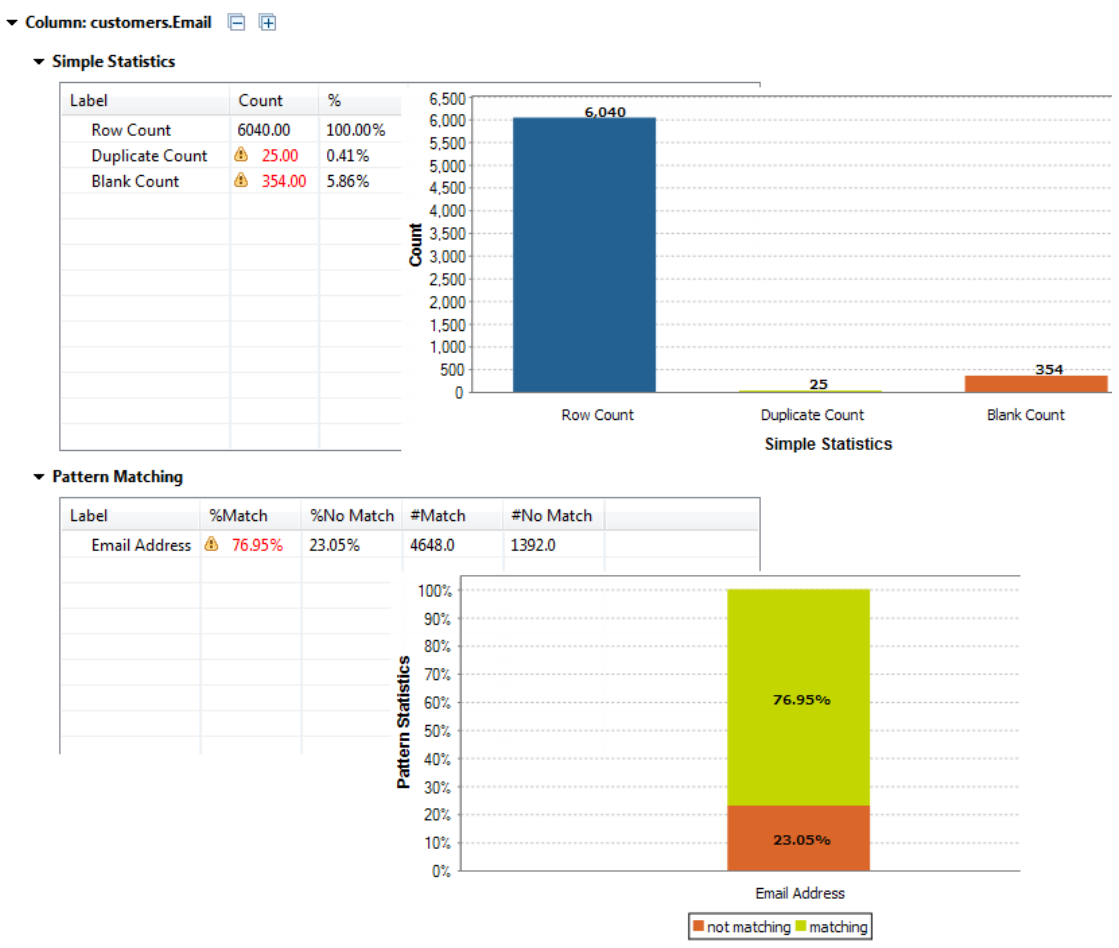
Once you finalize creating the column analysis and setting the indicators and patterns, you can execute it and display analysis results in tables and charts.

- You have opened the **Profiling** perspective in the Studio.
- You have created a column analysis. For further information, see Identifying anomalies in data on page 33.

1. In the **Analysis Parameters**, select **java** from the **Execution engine** list to run the analysis with the Java engine.
2. In the analysis editor, press **F6** to execute the analysis or click the **Run** button.

   The editor switches to the **Analysis Results** view. The analysis results show the generated charts for the analyzed columns accompanied with tables that detail the statistic and pattern matching results.

   The results for the Email column look as the following:



Column: customers.Email

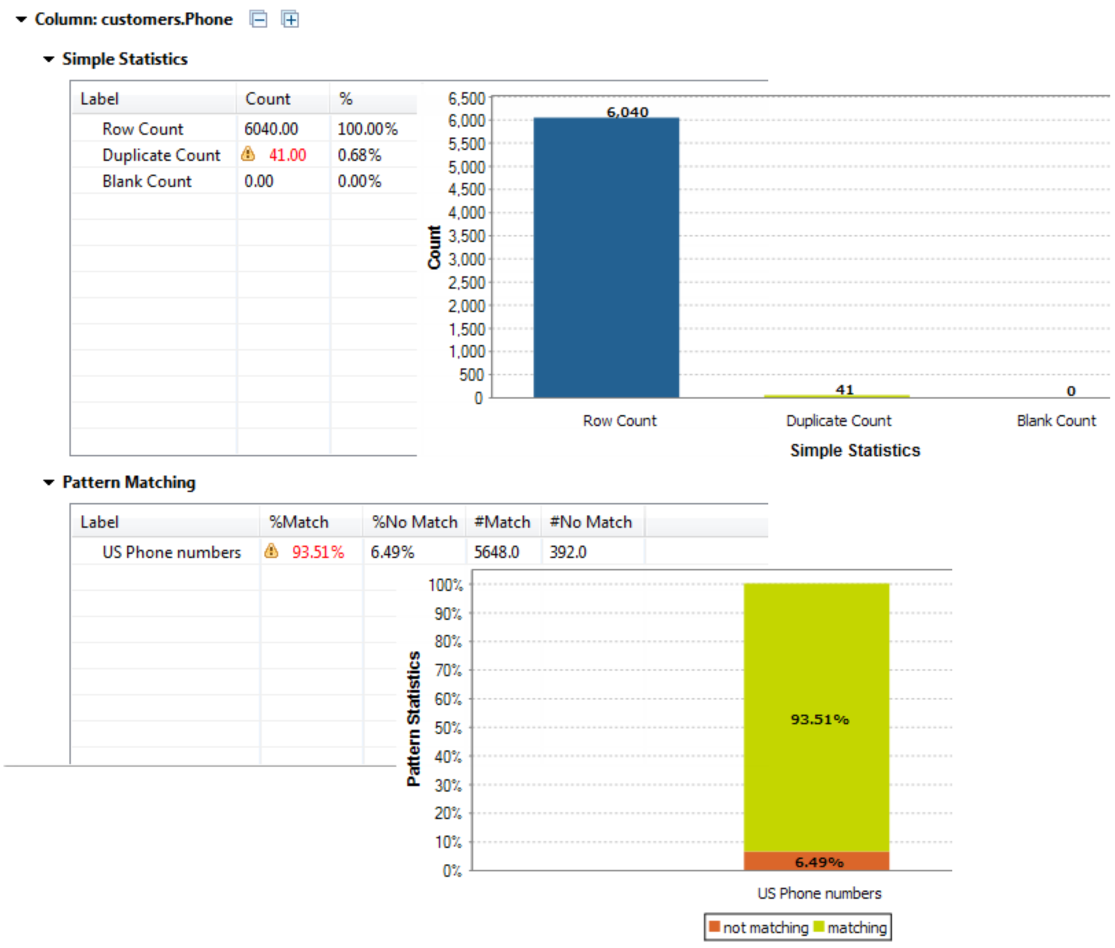
Simple Statistics

Label	Count	%
Row Count	6040.00	100.00%
Duplicate Count	⚠ 25.00	0.41%
Blank Count	⚠ 354.00	5.86%

Pattern Matching

Label	%Match	%No Match	#Match	#No Match
Email Address	⚠ 76.95%	23.05%	4648.0	1392.0

The results for the Phone column look as the following:



Column: customers.Phone

Simple Statistics

Label	Count	%
Row Count	6040.00	100.00%
Duplicate Count	41.00	0.68%
Blank Count	0.00	0.00%

Pattern Matching

Label	%Match	%No Match	#Match	#No Match
US Phone numbers	93.51%	6.49%	5648.0	392.0

The result sets for the Email and Phone columns give the count of the records that match and those that do not match the standard email pattern and the standard US phone numbers respectively. The results also give the blank and duplicate counts. This shows that the data is not very consistent and that it needs to be corrected.

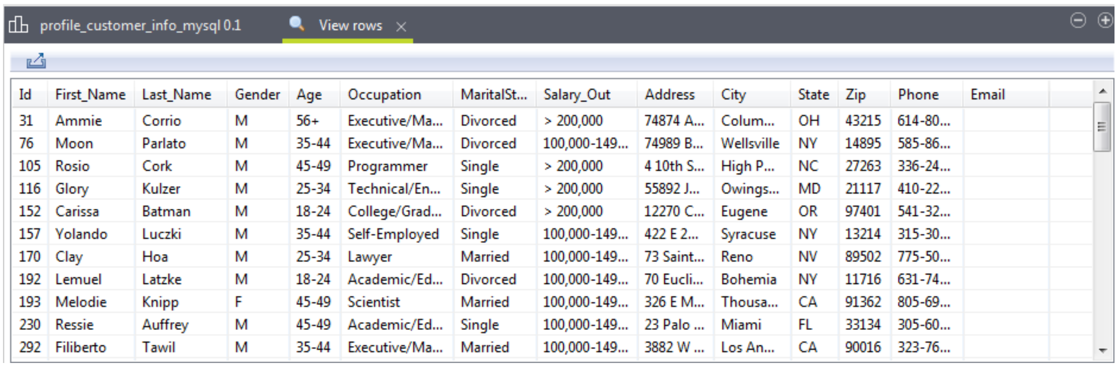
Browsing non-match data

After running the column analysis, you can access a view of the matching and non-matching data. This could be very helpful to see invalid rows for example and start analyzing what needs to be done to validate and cleanse such data.

- You have opened the **Profiling** perspective in the Studio.
- You have created and executed a column analysis. For further information, see Identifying anomalies in data on page 33.

1. In the **Analysis Results** view, right-click the **Blank Count** in the statistic results of the Email column and select **View rows** for example.

   A view opens listing all the blank rows in the Email column.

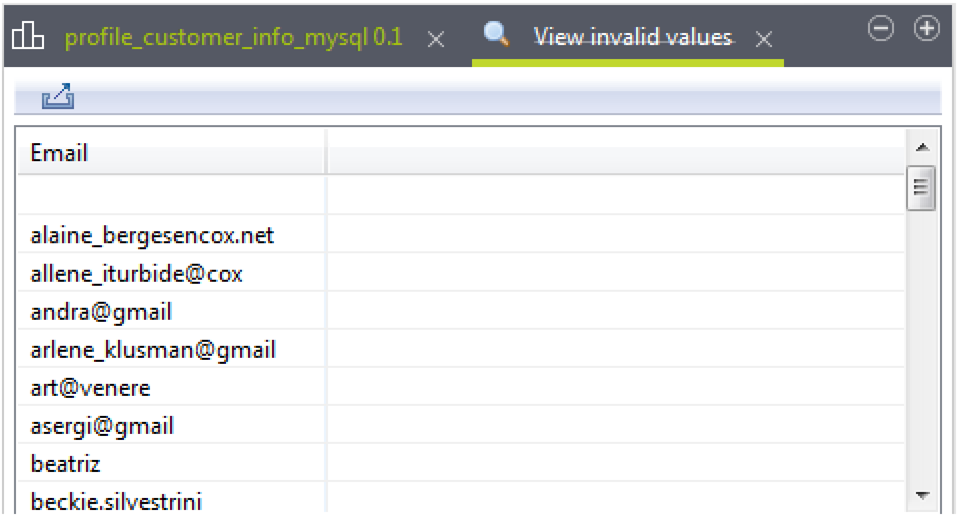


profile_customer_info_mysql 0.1 | View rows

Id	First_Name	Last_Name	Gender	Age	Occupation	MaritalSt...	Salary_Out	Address	City	State	Zip	Phone	Email
31	Ammie	Corrio	M	56+	Executive/Ma...	Divorced	> 200,000	74874 A...	Colum...	OH	43215	614-80...	
76	Moon	Parlato	M	35-44	Executive/Ma...	Divorced	100,000-149...	74989 B...	Wellsville	NY	14895	585-86...	
105	Rosio	Cork	M	45-49	Programmer	Single	> 200,000	4 10th S...	High P...	NC	27263	336-24...	
116	Glory	Kulzer	M	25-34	Technical/En...	Single	> 200,000	55892 J...	Owings...	MD	21117	410-22...	
152	Carissa	Batman	M	18-24	College/Grad...	Divorced	> 200,000	12270 C...	Eugene	OR	97401	541-32...	
157	Yolando	Luczki	M	35-44	Self-Employed	Single	100,000-149...	422 E 2...	Syracuse	NY	13214	315-30...	
170	Clay	Hoa	M	25-34	Lawyer	Married	100,000-149...	73 Saint...	Reno	NV	89502	775-50...	
192	Lemuel	Latzke	M	18-24	Academic/Ed...	Divorced	100,000-149...	70 Eucli...	Bohemia	NY	11716	631-74...	
193	Melodie	Knipp	F	45-49	Scientist	Married	100,000-149...	326 E M...	Thousa...	CA	91362	805-69...	
230	Ressie	Auffrey	M	45-49	Academic/Ed...	Single	100,000-149...	23 Palo ...	Miami	FL	33134	305-60...	
292	Filiberto	Tawil	M	35-44	Executive/Ma...	Married	100,000-149...	3882 W ...	Los An...	CA	90016	323-76...	

2. In the **Analysis Results** view, right-click the result in the **Pattern Matching** of the Email column and select **View invalid values** for example.

A view opens listing all the invalid email addresses.



profile_customer_info_mysql 0.1 | View invalid values

Email
alaine_bergesencox.net
allene_iturbide@cox
andra@gmail
arlene_klusman@gmail
art@venere
asergi@gmail
beatriz
beckie.silvestrini

Cleansing customer contact information

After profiling customer data and identifying its problems, some actions should be taken on data to cleanse it. You may start by generating out-of-the-box **Talend** Jobs. These Jobs remove duplicates from the columns you analyze. The Jobs also remove the values that do not match the patterns used in the analyses.

This helps you identify the issues you have in the address data and what you need to resolve.

Removing duplicate values

The profiling results of the column analysis show that there are some duplicate records in the email and phone columns. Check Showing analysis results on page 39 for detail.

From the analysis results, you can generate out-of-box Jobs that separate unique from duplicate records in the selected columns. Such Jobs output all the duplicates in a reject delimited file by default, and writes the unique values in the database used in the analysis.

You can follow the same procedure to remove duplicates from the Email or Phone columns.

- You have opened the **Profiling** perspective in the Studio.
- You have created and executed the column analysis. For further information, see Identifying anomalies in data on page 33.

1. Open the column analysis in the **Profiling** perspective and click **Analysis Results** at the bottom of the editor.
2. In the **Simple Statistics** results of the Email or Phone column, right-click **Duplicate Count** and select **Identify duplicates**.

   This example uses the outcome of the simple statistics used on the Email column.

   The **Integration** perspective opens showing the generated Job, and the Job is listed in the **Repository** tree view.

   The **tMysqlInput**, **tUniqueRow** and **tMysqlOutputBulkExec** components are automatically configured according to your connection and the columns you are analyzing. **tMysqlOutputBulkExec** writes unique records to a new table in MySQL and **tFileOutputDelimited** writes duplicate records in an output delimited file.
3. Press **F6** to execute the Job.

Duplicate values are written to the output file and unique records are written to a new table in the gettingstarted database in MySQL.

Removing non-matching values

The results of the patterns used on the Email and Phone columns show that some records do not respect the standard email and phone formats. Check Showing analysis results on page 39 for detail.

From the analysis results, you can generate out-of-box Jobs to recuperate the non-matching rows from the columns.

You can follow the same procedure to remove non-matching values from the Email or Phone columns.

- You have opened the **Profiling** perspective in the Studio.
- You have created and executed the column analysis. For further information, see Identifying anomalies in data on page 33.

1. Open the column analysis in the **Profiling** perspective and click **Analysis Results** at the bottom of the editor.
2. In the **Pattern Matching** tables of the `Email` or `Phone` column, right-click the results and select **Generate Job**.

   This example uses the results of the **US Phone numbers** pattern used on the Phone column.
3. In the wizard that opens, click **Finish** to confirm the creation of the Job.

   The **Integration** perspective opens showing the generated Job, and the Job is listed in the **Repository** tree view.

   This Job uses the Extract Transform Load process to write in two separate output files the Phone rows that match and do not match the pattern.

   The **tMysqlInput** is automatically configured according to your connection and **tPatternCheck** is automatically configured according the column you analyze.
4. Double-click each of the output component and change the default name or path of the output files, if needed.
5. Press **F6** to execute the Job.

   Matching and non-matching phone numbers are written to two separate output files.
6. Right-click each of the **tFileOutputDelimited** components and select **Data Viewer** to open a view on the data which matches and that which does not match the phone pattern.

You can then design a Job, for example, to standardize the phone numbers which match the pattern and give them the correct international format by using the **tStandardizePhoneNumber** component.

What's next

You have learned how Talend Studio helps you profile your data and collect statistics and information about it in order to assess the quality level of the data according to defined set goals.

You have seen:

- How to use the **Profiling** perspective of the studio to analyze customer email addresses and phone numbers by using out-of-box indicators and patterns.
- How the analysis results show the matching and non-matching address records and how it is possible to browse such data.
- How to generate **Talend** Jobs on analysis results and recuperate the non-matching data.
- How to use a **Talend** Job to standardize phone numbers.

Once you succeed with the simple procedures outlined in Identifying anomalies in data on page 33, you can start digging deeper to see in detail all the profiling and cleansing capabilities of Talend Studio.

Now you can learn to use the Talend Big Data solution to integrate the information about the movies your company provides for rental and streaming and their directors.

Performing data integration tasks for Big Data

A Big Data Job is used to read, transform and write data about movies and movie directors in a Hadoop environment.

Joining movie and director information using a MapReduce Job

This scenario demonstrates:

1. How to create a Talend MapReduce Job. See Creating the MapReduce Job on page 44 for details.
2. How to drop and link the components to be used in a MapReduce Job. See Dropping and linking MapReduce components on page 45 for details.
3. How to configure the input components using the related metadata from the **Repository**. See Configuring the input data on page 46 for details.
4. How to configure the transformation to join the input data. See Configuring the data transformation on page 48 for details.
5. How to write the transformed data to HDFS. See Writing the output to HDFS on page 49 for details.

Creating the MapReduce Job

A Talend MapReduce Job allows you to access and use the Talend MapReduce components to visually design MapReduce programs to read, transform or write data.

- You have launched your Talend Studio and opened the **Integration** perspective.

1. In the **Repository** tree view, expand the **Job Designs** node, right-click the **Big Data Batch** node and select **Create folder** from the contextual menu.
2. In the **New Folder** wizard, name your Job folder `getting_started` and click **Finish** to create your folder.
3. Right-click the `getting_started` folder and select **Create folder** again.

4. In the **New Folder** wizard, name the new folder to `mapreduce` and click **Finish** to create the folder.
5. Right-click the `mapreduce` folder and select **Create Big Data Batch Job**.
6. In the **New Big Data Batch Job** wizard, select **MapReduce** from the **Framework** drop-down list.
7. Enter a name for this MapReduce Job and other useful information.

   For example, enter `aggregate_movie_director_mr` in the **Name** field.

The MapReduce component **Palette** is now available in the Studio. You can start to design the Job by leveraging this **Palette** and the **Metadata** node in the **Repository**.

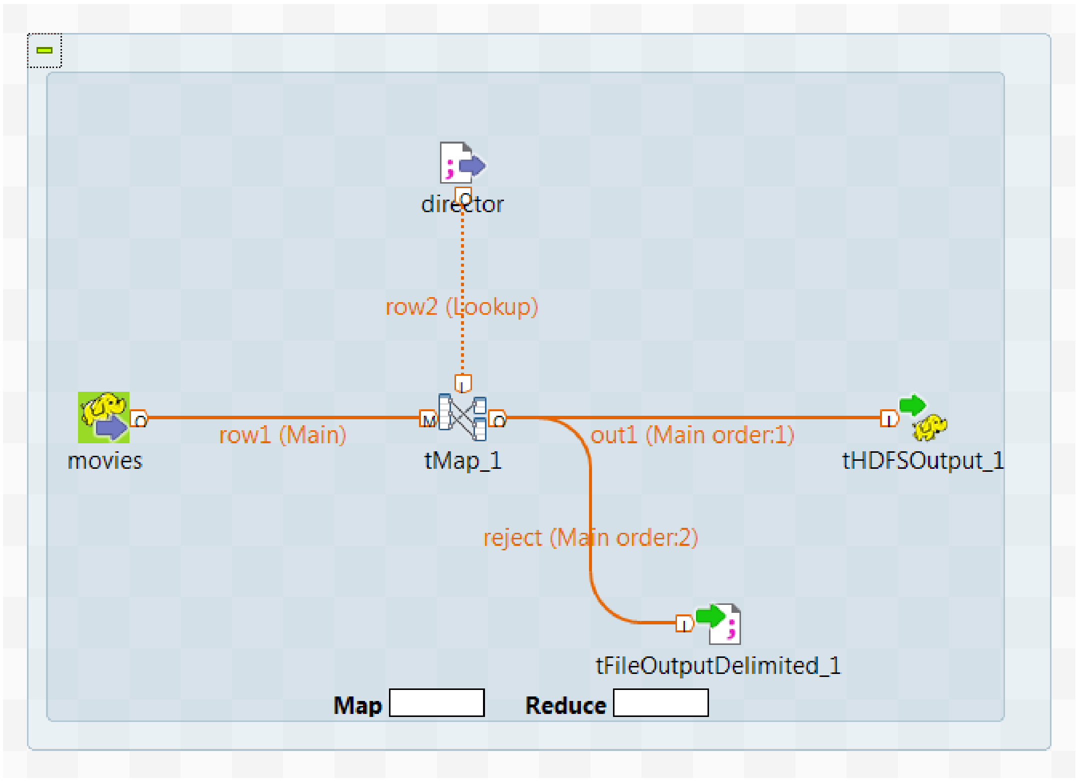
Dropping and linking MapReduce components

You orchestrate the MapReduce components in the Job workspace in order to design a data transformation process that runs in the MapReduce framework.

- You have launched your Talend Studio and opened the **Integration** perspective.
- An empty Job has been created as described in Creating the MapReduce Job on page 44 and is open in the workspace.

1. In the Job, enter the name of the component to be used and select this component from the list that appears. In this scenario, the components are two **tHDFSInput** components, a **tFileInputDelimited** component, a **tMap** component, a **tHDFSOutput** component and a **tFileOutputDelimited** component.
   - The **tHDFSInput** and the **tFileInputDelimited** components are used to load the movie data and the director data, respectively, from HDFS into the data flow of the current Job.
   - The **tMap** component is used to transform the input data.
   - The **tHDFSOuput** and the **tFileOutputDelimited** components write the results into given directories in HDFS.
2. Double-click the **tHDFSInput** component to make this label editable and then enter `movie` to change the label of this component.
3. Do the same to label **tFileInputDelimited** to `director`.
4. Right click the **tHDFSInput** component that is labelled `movie`, then from the contextual menu, select **Row** > **Main** and click **tMap** to connect it to **tMap**. This is the main link through which the movie data is sent to **tMap**.
5. Do the same to connect the `director` **tFileInputDelimited** component to **tMap** using the **Row** > **Main** link. This is the **Lookup** link through which the director data is sent to **tMap** as lookup data.
6. Do the same to connect the **tMap** component to **tHDFSOutput** using the **Row** > **Main** link, then in the pop-up wizard, name this link to `out1` and click **OK** to validate this change.
7. Repeat these operations to connect the **tMap** component to **tFileOutputDelimited** component using the **Row** > **Main** link and name it to `reject`.

In the workspace, the whole Job looks like this:



Configuring the input data

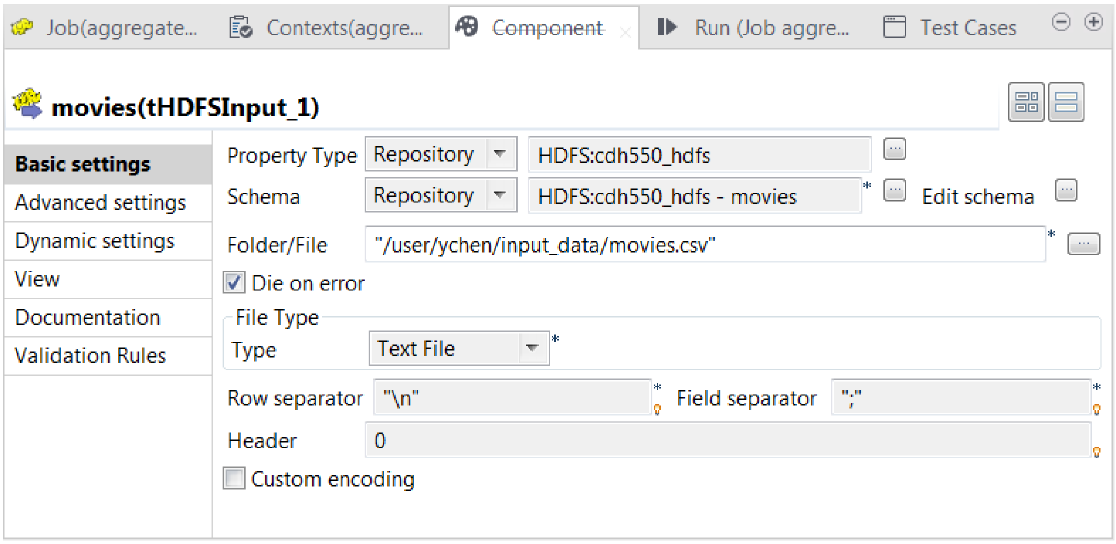
The **tHDFSInput** component and the **tFileInputDelimited** components are configured to load data from HDFS into the Job.

- The source files, `movies.csv` and `directors.txt` have been uploaded into HDFS as explained in Uploading files to HDFS on page 24.
- The metadata of the `movie.csv` file has been set up in the HDFS folder under the **Hadoop cluster** node in the **Repository**.

  If you have not done so, see Preparing file metadata on page 28 to create the metadata.

1. Expand the **Hadoop cluster** node under the **Metadata** node in the **Repository** and then the `my_cdh` Hadoop connection node and its child node to display the `movies` schema metadata node you have set up under the **HDFS** folder as explained in Preparing file metadata on page 28.
2. Drop this schema metadata node onto the `movie` **tHDFSInput** component in the workspace of the Job.
3. Double-click the `movie` **tHDFSInput** component to open its **Component** view.

   This **tHDFSInput** has automatically reused the HDFS configuration and the movie metadata from the **Repository** to define the related parameters in its **Basic settings** view.

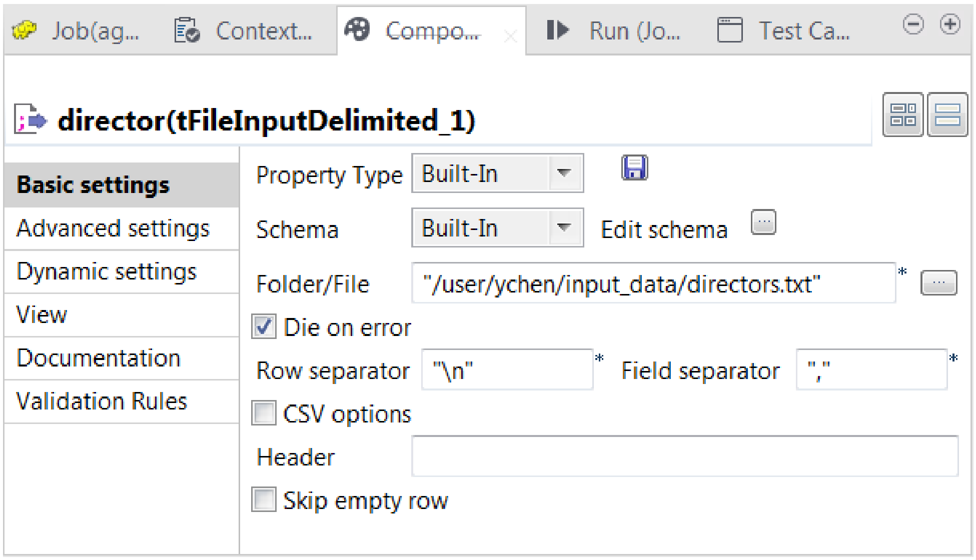


movies(tHDFSInput_1)

Basic settings | Advanced settings | Dynamic settings | View | Documentation | Validation Rules

Property Type: Repository — HDFS:cdh550_hdfs
Schema: Repository — HDFS:cdh550_hdfs - movies — Edit schema
Folder/File: "/user/ychen/input_data/movies.csv"
[✓] Die on error
File Type
Type: Text File
Row separator: "\n" Field separator: ";"
Header: 0
[ ] Custom encoding

4. Double-click the `director` **tFileInputDelimited** component to open its **Component** view.

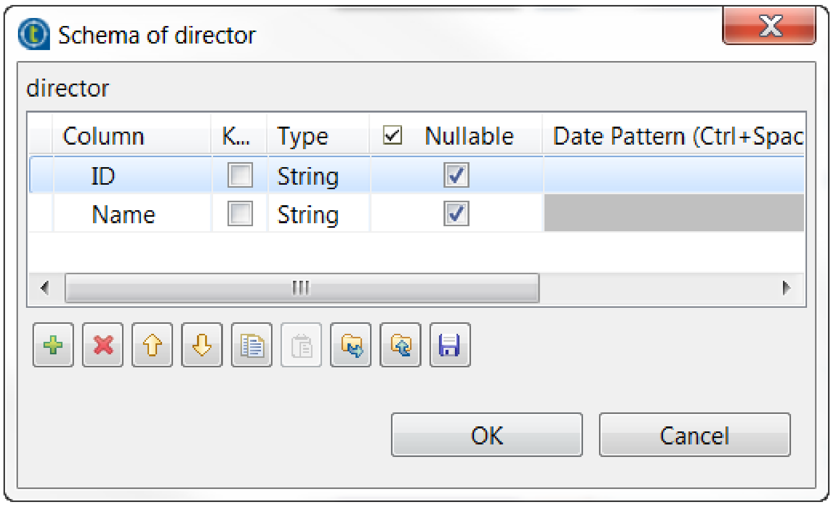


director(tFileInputDelimited_1)

Basic settings | Advanced settings | Dynamic settings | View | Documentation | Validation Rules

Property Type: Built-In
Schema: Built-In — Edit schema
Folder/File: "/user/ychen/input_data/directors.txt"
[✓] Die on error
Row separator: "\n" Field separator: ","
[ ] CSV options
Header:
[ ] Skip empty row

5. Click the **[...]** button next to **Edit schema** to open the schema editor.
6. Click the **[+]** button twice to add two rows and in the **Column** column, rename them to `ID` and `Name`, respectively.



director

Column	K...	Type	✓ Nullable	Date Pattern (Ctrl+Spac
ID	☐	String	✓	
Name	☐	String	✓	

7. Click **OK** to validate these changes and accept the propagation prompted by the pop-up dialog box.

8. In the **Folder/File** field, enter or browse to the directory where the director data is stored. As is explained in Uploading files to HDFS on page 24, this data has been written in `/user/ychen/input_data/directors.txt`.
9. In **Field separator** field, enter a comma (,) as this is the separator used by the director data.

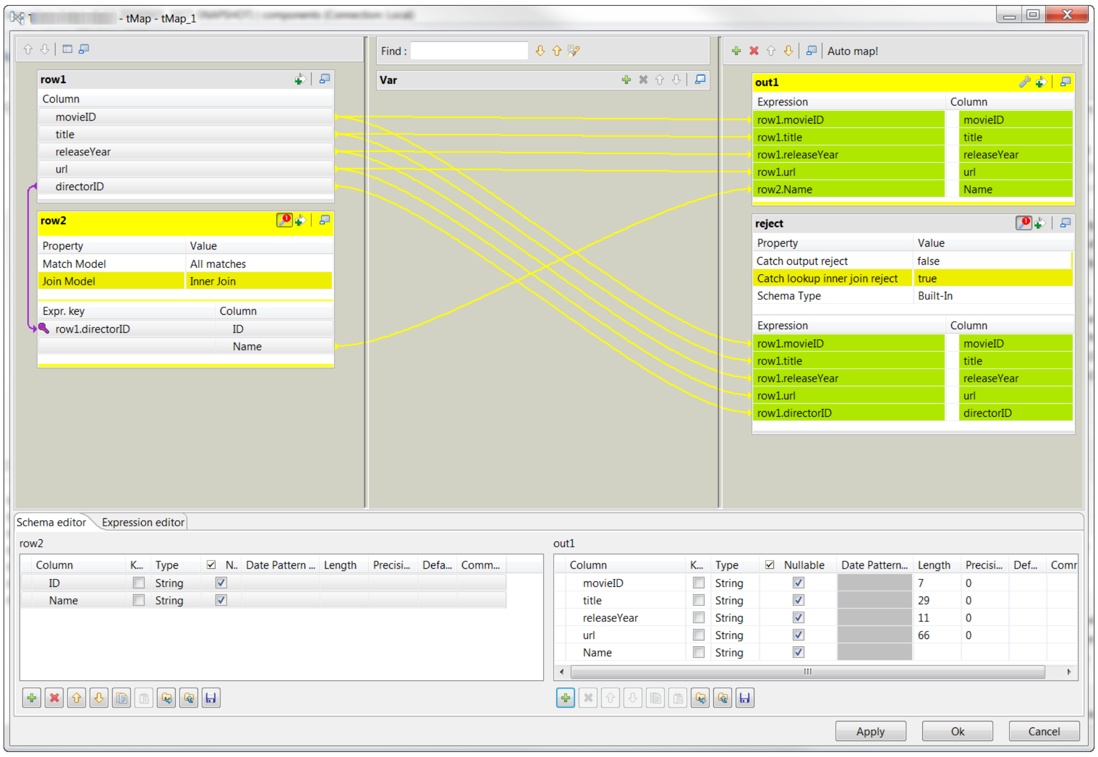
The input components are now configured to load the movie data and the director data to the Job.

Configuring the data transformation

The **tMap** component is configured to join the movie data and the director data.

Once the movie data and the director data are loaded into the Job, you need to configure the **tMap** component to join them to produce the output you expect.

1. Double-click **tMap** to open its **Map Editor** view.



row1

Column
movieID
title
releaseYear
url
directorID

row2

Property	Value
Match Model	All matches
Join Model	Inner Join

Expr. key	Column
row1.directorID	ID
	Name

out1

Expression	Column
row1.movieID	movieID
row1.title	title
row1.releaseYear	releaseYear
row1.url	url
row2.Name	Name

reject

Property	Value
Catch output reject	false
Catch lookup inner join reject	true
Schema Type	Built-In

Expression	Column
row1.movieID	movieID
row1.title	title
row1.releaseYear	releaseYear
row1.url	url
row1.directorID	directorID

Schema editor — row2

Column	K...	Type	N..	Date Pattern ...	Length	Precisi...	Defa...	Comm...
ID	☐	String	✓					
Name	☐	String	✓					

Schema editor — out1

Column	K...	Type	Nullable	Date Pattern...	Length	Precisi...	Def...	Comr
movieID	☐	String	✓		7	0		
title	☐	String	✓		29	0		
releaseYear	☐	String	✓		11	0		
url	☐	String	✓		66	0		
Name	☐	String	✓					

2. Drop the `movieID` column, the `title` column, the `releaseYear` column and the `url` column from the left side onto each of the output flow table.

   On the input side (left side) of the **Map Editor**, each of the two tables represents one of the input flow, the upper one for the main flow and the lower one for the lookup flow.

   On the output side (right side), the two tables represent the output flows that you named to `out1` and `reject` when you linked **tMap** to **tHDFSOutput** and **tFileOutputDelimited** in Dropping and linking MapReduce components on page 45.
3. On the input side, drop the `directorID` column from the main flow table to the **Expr.key** column of the `ID` row in the lookup flow table. This way, the join key between the main flow and the lookup flow is defined.
4. Drop the `directorID` column from the main flow table to the `reject` table on the output side and drop the `Name` column from the lookup flow table to the `out1` table.

   From the **Schema editor** view in the lower part of the editor, you can see the schemas on both sides have been automatically completed.
5. On the lookup flow table, click the button to display the settings panel for the join operation.

6. In the **Join model** row, click the **Value** column and click the **[...]** button that is displayed.

   The **Options** window is displayed.
7. Select **Inner join** in order to output only the records that contain join keys that exist in both the main flow and lookup flow.
8. On the `reject` output flow table, click the 🔧 button to open the setting panel.
9. In the **Catch Lookup inner join reject** row, select **true** to output the records that are rejected by the inner join performed on the input side.
10. Click **Apply**, then click **OK** to validate these changes and accept the propagation prompted by the pop-up dialog box.

The transformation is now configured to complete the movie data with the names of their directors and write the movie records that do not contain any director data into a separate data flow.

Writing the output to HDFS

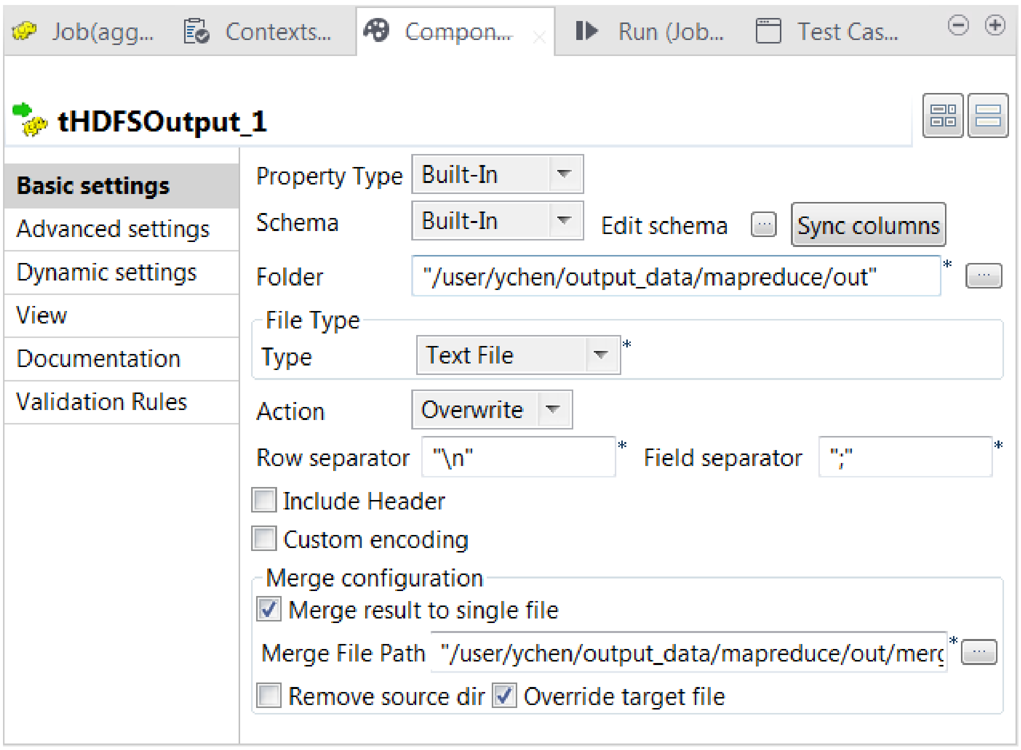
Two output components are configured to write the expected movie data and the rejected movie data to different directories in HDFS.

- Ensure that the client machine on which the Talend Jobs are executed can recognize the host names of the nodes of the Hadoop cluster to be used. For this purpose, add the IP address/hostname mapping entries for the services of that Hadoop cluster in the `hosts` file of the client machine.

  For example, if the host name of the Hadoop Namenode server is talend-cdh550.weave.local, and its IP address is 192.168.x.x, the mapping entry reads `192.168.x.x talend-cdh550.weave.local`.
- The Hadoop cluster to be used has been properly configured and is running.
- The administrator of the cluster has given read/write rights and permissions to the username to be used for the access to the related data and directories in HDFS.

1. Double-click the **tHDFSOutput** component, which receives the `out1` link.

   Its **Basic settings** view is opened in the lower part of the Studio.



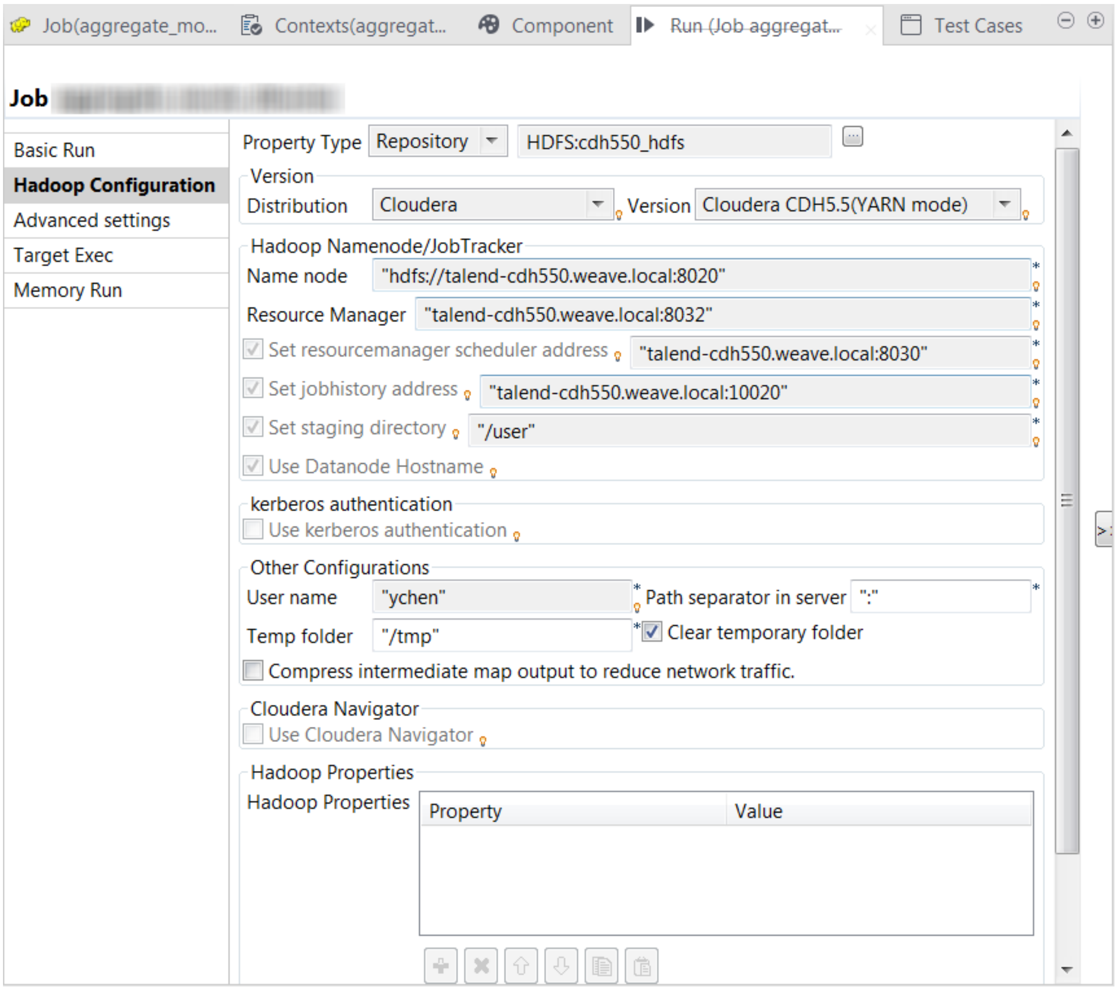
tHDFSOutput_1

Basic settings
Advanced settings
Dynamic settings
View
Documentation
Validation Rules

Property Type: Built-In
Schema: Built-In
Edit schema
Sync columns
Folder: "/user/ychen/output_data/mapreduce/out"
File Type
Type: Text File
Action: Overwrite
Row separator: "\n"
Field separator: ";"
☐ Include Header
☐ Custom encoding
Merge configuration
☑ Merge result to single file
Merge File Path: "/user/ychen/output_data/mapreduce/out/merg
☐ Remove source dir ☑ Override target file

2. In the **Folder** field, enter or browse to the directory you need to write the result in. In this scenario, it is `/user/ychen/output_data/mapreduce/out`, which receives the records that contain the names of the movie directors.
3. Select **Overwrite** from the **Action** drop-down list. This way, the target directory is overwritten if it exists.
4. Select the **Merge result to single file** check box in order to merge the `part-` files typically generated by MapReduce into one single file. The **Merge file path** field is displayed.
5. In the **Merge file path** field, enter or browse to the file into which you want the `part-` files to merge.

   In this scenario, this file is `/user/ychen/output_data/mapreduce/out/merged`.
6. Repeat the same operations to configure the **tFileOutputDelimited** component, which receives the `reject` link, but set the directory, in the **Folder** field, to `/user/ychen/output_data/mapreduce/reject` and leave the **Merge result to single file** check box clear.
7. In the **Run** view, click the **Hadoop configuration** tab to verify that the Hadoop connection metadata has been properly imported from the **Repository**.



Property Type: Repository — HDFS:cdh550_hdfs

Version
Distribution: Cloudera — Version: Cloudera CDH5.5(YARN mode)

Hadoop Namenode/JobTracker
Name node: "hdfs://talend-cdh550.weave.local:8020"
Resource Manager: "talend-cdh550.weave.local:8032"
[✓] Set resourcemanager scheduler address: "talend-cdh550.weave.local:8030"
[✓] Set jobhistory address: "talend-cdh550.weave.local:10020"
[✓] Set staging directory: "/user"
[✓] Use Datanode Hostname

kerberos authentication
[ ] Use kerberos authentication

Other Configurations
User name: "ychen" — Path separator in server: ":"
Temp folder: "/tmp" — [✓] Clear temporary folder
[ ] Compress intermediate map output to reduce network traffic.

Cloudera Navigator
[ ] Use Cloudera Navigator

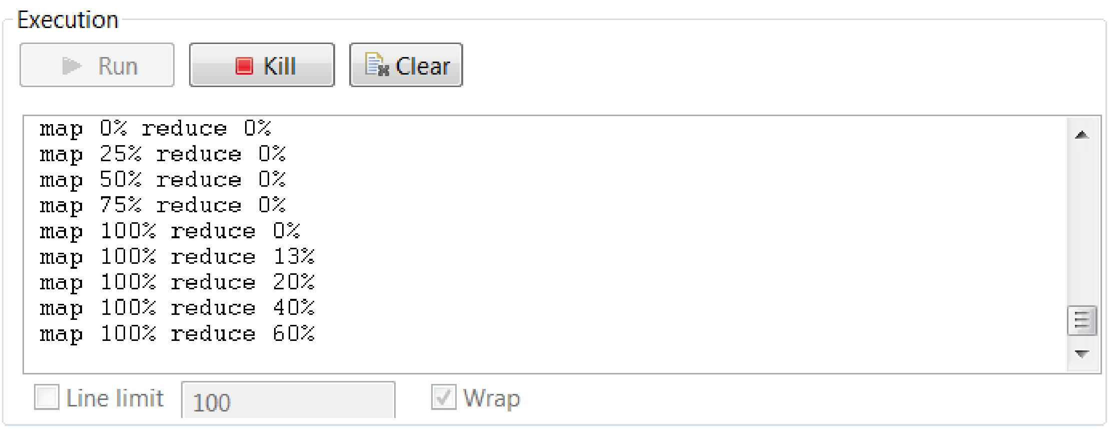
Hadoop Properties

Property	Value

You always need to use this **Hadoop Configuration** tab to define the connection to a given Hadoop distribution for the whole MapReduce Job and this connection is effective on a per-Job basis.

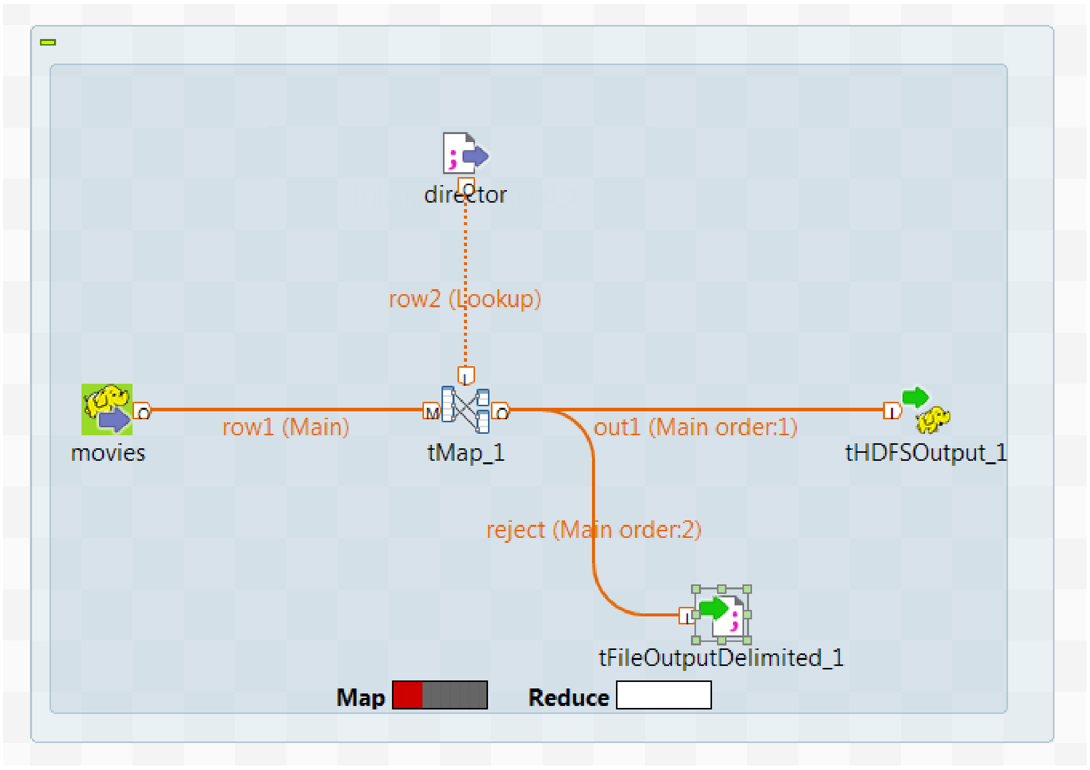
8. Press **F6** to run the Job.

The **Run** view is automatically opened in the lower part of the Studio and shows the execution progress of this Job.

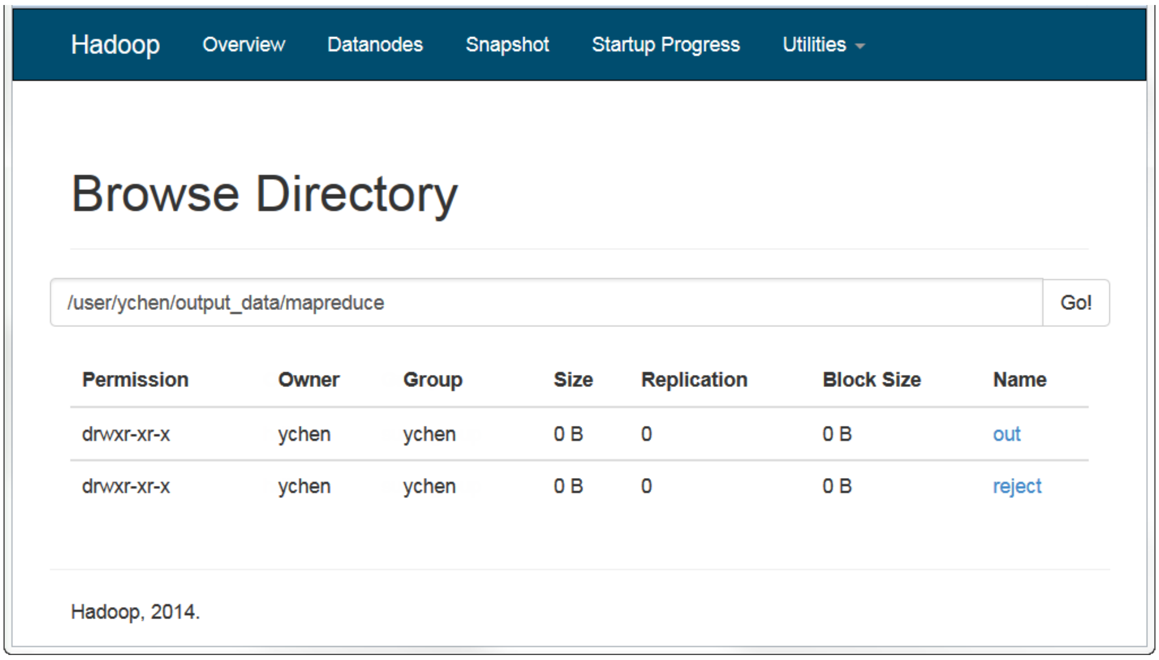


```
map 0% reduce 0%
map 25% reduce 0%
map 50% reduce 0%
map 75% reduce 0%
map 100% reduce 0%
map 100% reduce 13%
map 100% reduce 20%
map 100% reduce 40%
map 100% reduce 60%
```

The Job itself also shows the progress graphically.



Once done, you can check, for example in the web console of your HDFS system, that the output has been written in HDFS.



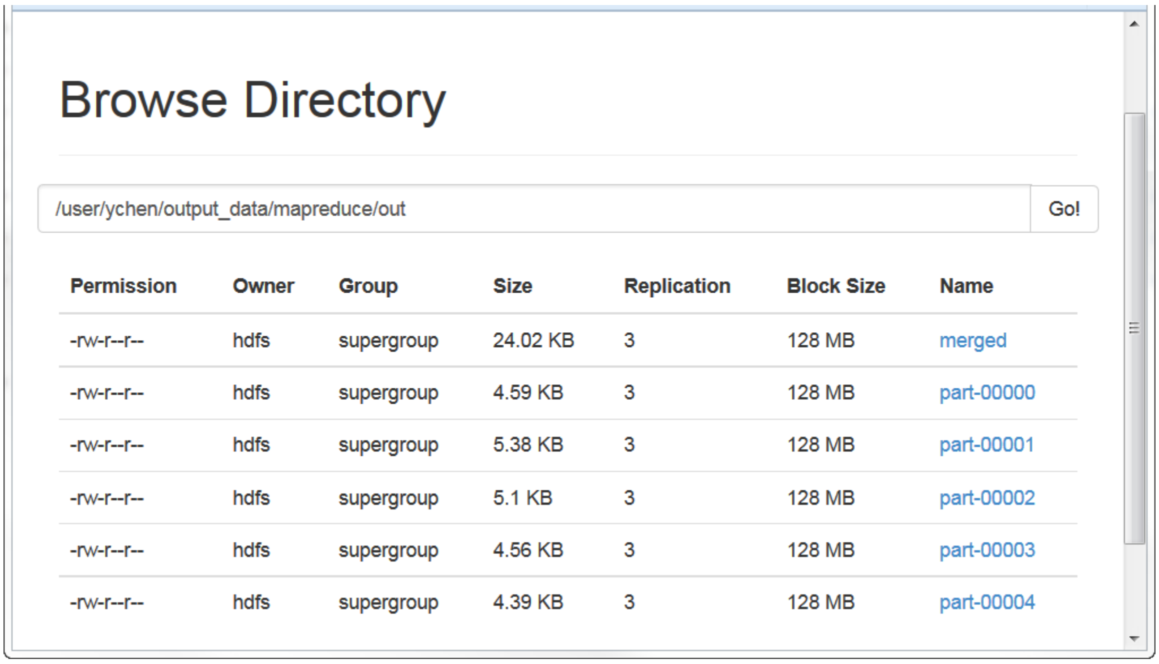
Browse Directory

/user/ychen/output_data/mapreduce

Permission	Owner	Group	Size	Replication	Block Size	Name
drwxr-xr-x	ychen	ychen	0 B	0	0 B	out
drwxr-xr-x	ychen	ychen	0 B	0	0 B	reject

Hadoop, 2014.

A merged file has also been created.



Browse Directory

/user/ychen/output_data/mapreduce/out

Permission	Owner	Group	Size	Replication	Block Size	Name
-rw-r--r--	hdfs	supergroup	24.02 KB	3	128 MB	merged
-rw-r--r--	hdfs	supergroup	4.59 KB	3	128 MB	part-00000
-rw-r--r--	hdfs	supergroup	5.38 KB	3	128 MB	part-00001
-rw-r--r--	hdfs	supergroup	5.1 KB	3	128 MB	part-00002
-rw-r--r--	hdfs	supergroup	4.56 KB	3	128 MB	part-00003
-rw-r--r--	hdfs	supergroup	4.39 KB	3	128 MB	part-00004

Converting a MapReduce Job to a Spark Batch Job

This scenario shows how to easily create Spark Batch Job by converting the existing MapReduce Job in order to process data about movies and movie directors in the Spark environment.

This scenario demonstrates:

1. How to convert a Job. See Converting the Job on page 53 for details.
2. How to make the edits needed after the Job conversion. See Editing the converted Job on page 54 for details.

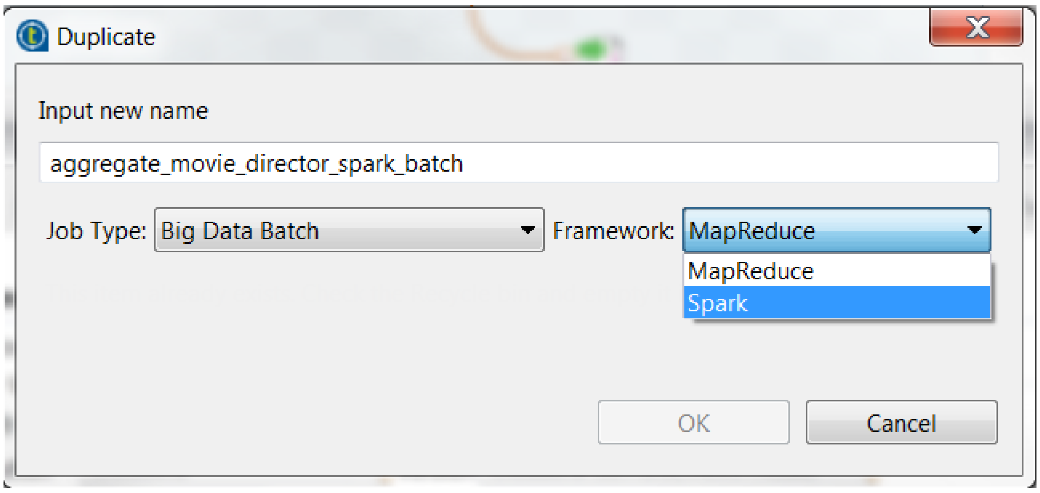
Converting the Job

Converting the existing MapReduce Job to a Spark Batch Job allows you to make full use of existing assets to easily create Spark Jobs.

- You have launched your Talend Studio and opened the Integration perspective.
- You have created the `aggregate_movie_director_mr` MapReduce Job described in Joining movie and director information using a MapReduce Job on page 44 and run it successfully.

1. In the **Repository** tree view, expand the **Job Designs** node, the **Big Data Batch** node and then the `getting_started` folder and the `mapreduce` folder.
2. Right-click the `aggregate_movie_director_mr` Job and from the contextual menu, select **Duplicate**.

   The **Duplicate** window is opened.



3. In the **Input new name** field, name this duplicate to `aggregate_movie_director_spark_batch`.
4. From the **Framework** list, select **Spark** and click **OK** to validate the changes.

   The `aggregate_movie_director_spark_batch` Job is displayed in the `mapreduce` folder in the **Repository**.
5. Right-click the **getting_started** folder and select **Create folder** from the contextual menu.
6. In the **New Folder** wizard, name the new folder to `spark_batch` and click **Finish** to create the folder.
7. Drop the `aggregate_movie_director_spark_batch` Job into this `spark_batch` folder.

This new Spark Batch Job is now ready for further editing.

Editing the converted Job

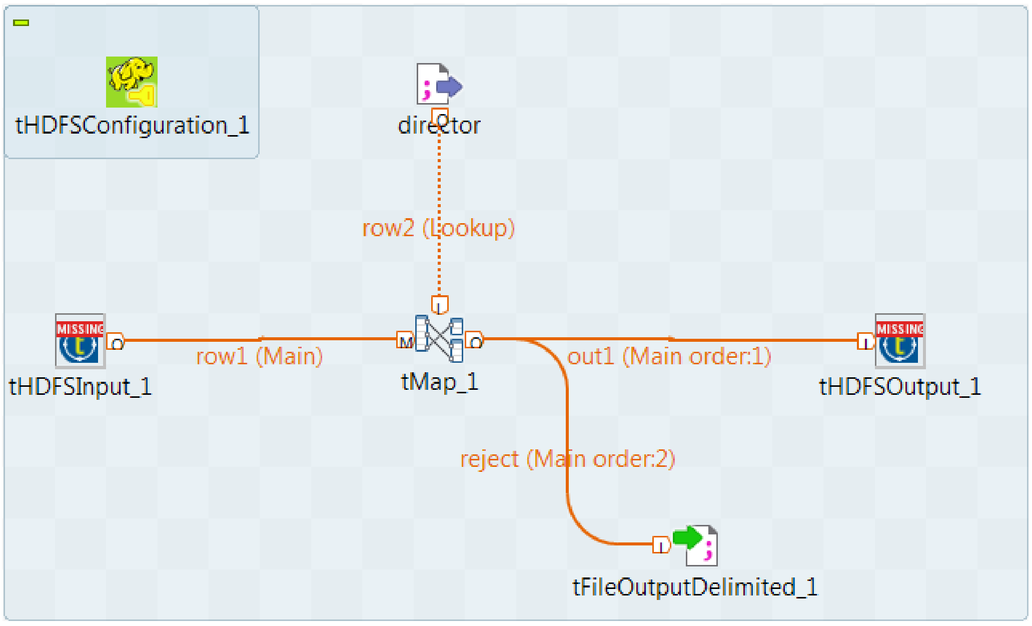
You update the components, when necessary, to finalize a data transformation process that runs in the Spark framework.

- Ensure that the client machine on which the Talend Jobs are executed can recognize the host names of the nodes of the Hadoop cluster to be used. For this purpose, add the IP address/hostname mapping entries for the services of that Hadoop cluster in the `hosts` file of the client machine.

  For example, if the host name of the Hadoop Namenode server is talend-cdh550.weave.local, and its IP address is 192.168.x.x, the mapping entry reads `192.168.x.x talend-cdh550.weave.local`.
- The Hadoop cluster to be used has been properly configured and is running.

  The Cloudera CDH V5.5 cluster used in this use case integrates Spark by default.
- The administrator of the cluster has given read/write rights and permissions to the username to be used for the access to the related data and directories in HDFS.

1. In the **Repository**, double-click the `aggregate_movie_director_spark_batch` Job to open it in the workspace.

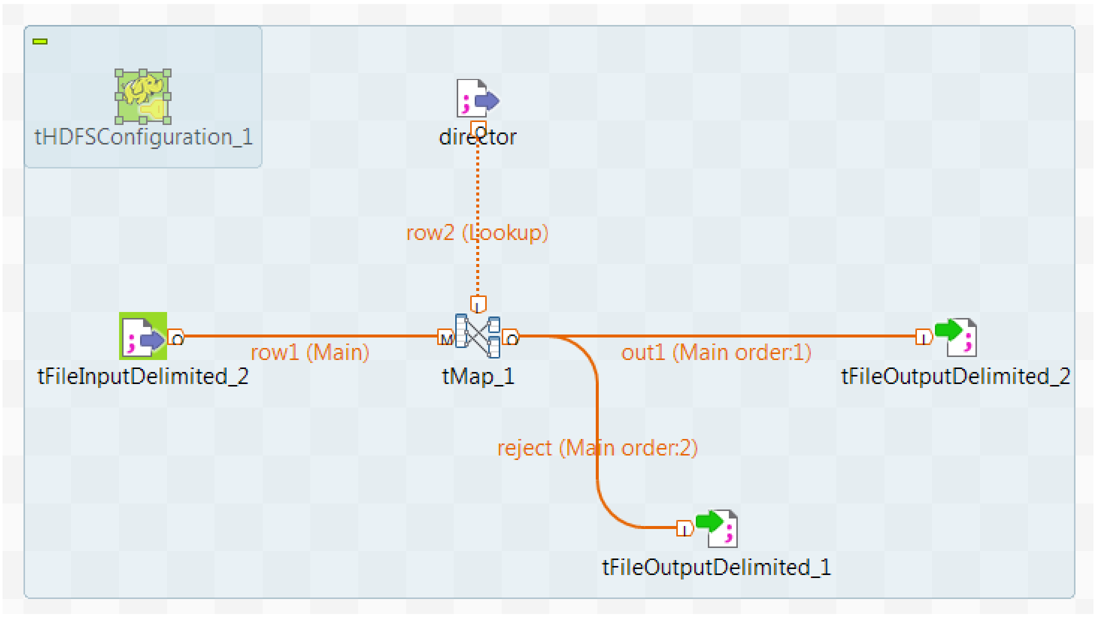


A tHDFSConfiguration component has been added automatically and inherits the configuration for the connection to HDFS from the original MapReduce Job.

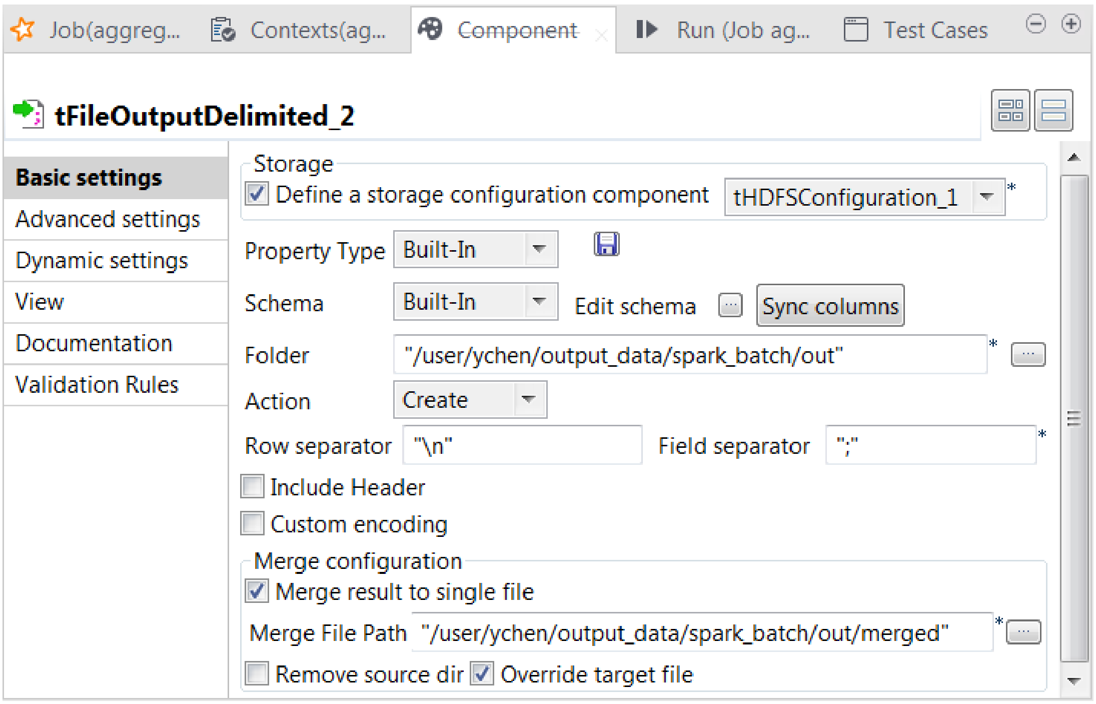
The MISSING icons indicate that the components that are used in the original Job do not exist in the current Job framework, Spark Batch. They are tHDFSInput and tHDFSOutput in this example.

2. Click **tHDFSInput** to select it and then in the popup **Warning** window, click **OK** to close this window.
3. Press **Delete** on your keyboard to remove **tHDFSInput**.
4. In the Job workspace, enter **tFileInputDelimited** and select this component from the list that appears.

   tFileInputDelimited is added to the workspace.
5. Do the same to replace **tHDFSOutput** with **tFileOutputDelimited**.
6. Expand the **Hadoop cluster** node under the **Metadata** node in the **Repository** and then the `my_cdh` connection node and its child node to display the `movies` schema metadata node you have set up under the **HDFS** folder as explained in Preparing file metadata on page 28.
7. Drop this schema metadata node onto the new **tFileInputDelimited** component in the workspace of the Job.
8. Right-click this **tFileInputDelimited** component, then from the contextual menu, select **Row** > **Main** and click **tMap** to connect it to **tMap**.
9. Right-click **tMap**, then from the context menu, select **Row** > **out1** and click the new **tFileOutputDelimited** to connect **tMap** to this component.



10. Double-click the new **tFileOutputDelimited** component to open its **Component** view.

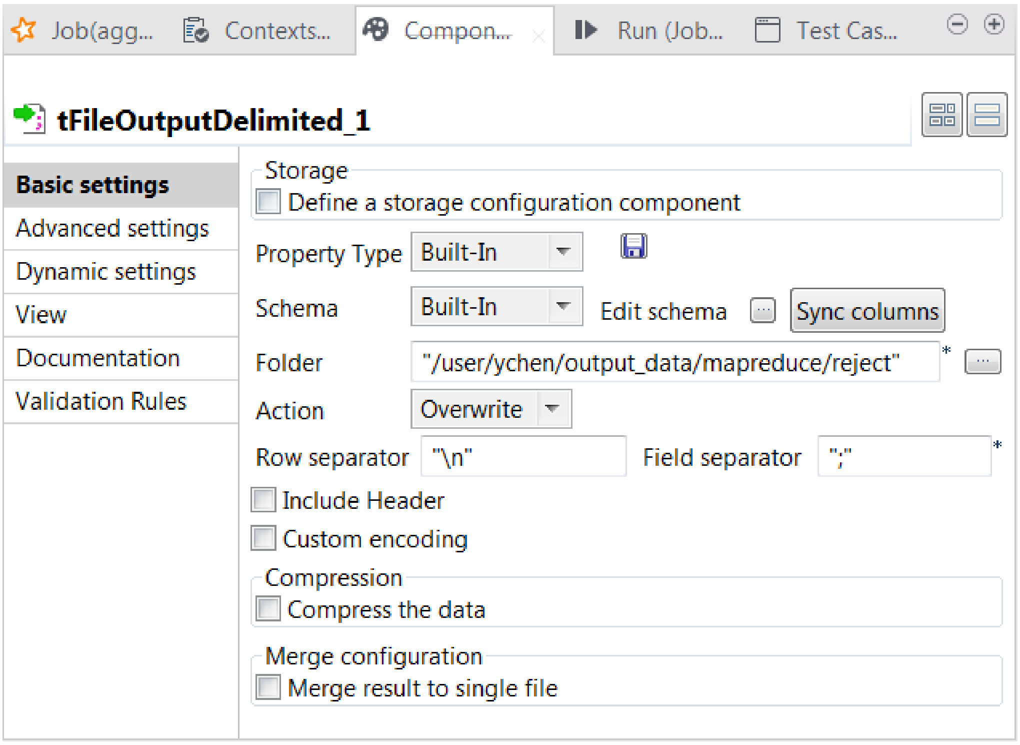


Storage
☑ Define a storage configuration component: tHDFSConfiguration_1
Property Type: Built-In
Schema: Built-In
Folder: "/user/ychen/output_data/spark_batch/out"
Action: Create
Row separator: "\n"
Field separator: ";"
☐ Include Header
☐ Custom encoding
Merge configuration
☑ Merge result to single file
Merge File Path: "/user/ychen/output_data/spark_batch/out/merged"
☐ Remove source dir ☑ Override target file

11. In the **Folder** field, enter or browse to the directory you need to write the result in. In this scenario, it is `/user/ychen/output_data/spark_batch/out`, which receives the records that contain the names of the movie directors.
12. Select the **Merge result to single file** check box in order to merge the `part-` files typically generated by Spark into one single file.

    The **Merge file path** field is displayed.
13. In the **Merge file path** field, enter or browse to the file into which you want the `part-`part- files to merge.

    In this scenario, this file is `/user/ychen/output_data/spark_batch/out/merged`.
14. Double-click the other **tFileOutputDelimited** component which receives the **reject** link from **tMap** to open its **Component** view.



tFileOutputDelimited_1

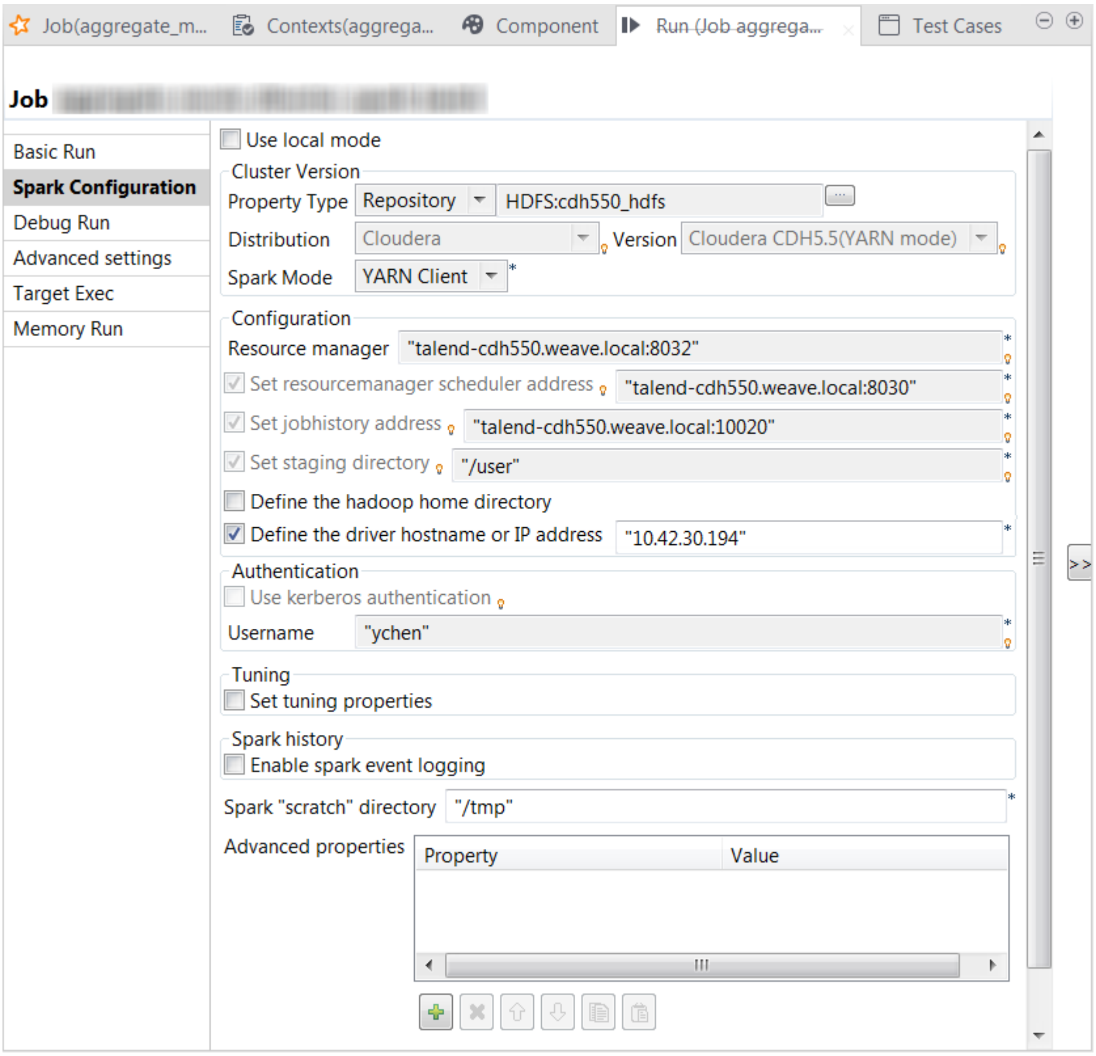
Storage
- [ ] Define a storage configuration component

Property Type: Built-In
Schema: Built-In
Edit schema
Sync columns
Folder: "/user/ychen/output_data/mapreduce/reject"
Action: Overwrite
Row separator: "\n"
Field separator: ";"
- [ ] Include Header
- [ ] Custom encoding

Compression
- [ ] Compress the data

Merge configuration
- [ ] Merge result to single file

15. In the **Folder** field, set the directory to `/user/ychen/output_data/spark_batch/reject`.
16. In the **Run** view, click the **Spark configuration** tab to verify that the Hadoop/Spark connection metadata has been properly inherited from the original Job.



Job [redacted]

- [ ] Use local mode

Cluster Version

Property Type	Repository; HDFS:cdh550_hdfs
Distribution	Cloudera
Version	Cloudera CDH5.5(YARN mode)
Spark Mode	YARN Client

Configuration

Resource manager	"talend-cdh550.weave.local:8032"
✓ Set resourcemanager scheduler address	"talend-cdh550.weave.local:8030"
✓ Set jobhistory address	"talend-cdh550.weave.local:10020"
✓ Set staging directory	"/user"
☐ Define the hadoop home directory	
✓ Define the driver hostname or IP address	"10.42.30.194"

Authentication

- [ ] Use kerberos authentication
- Username: "ychen"

Tuning

- [ ] Set tuning properties

Spark history

- [ ] Enable spark event logging

Spark "scratch" directory: "/tmp"

Advanced properties

Property	Value

You always need to use this **Spark Configuration** tab to define the connection to a given Hadoop/Spark distribution for the whole Spark Batch Job and this connection is effective on a per-Job basis.

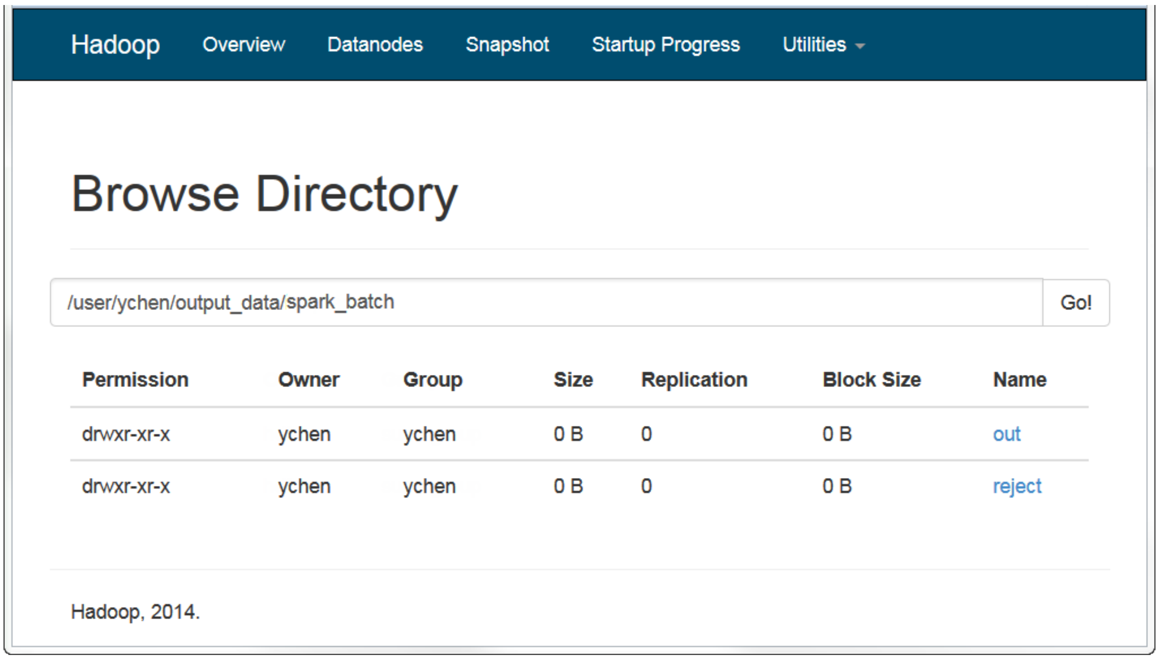
17. If you are not sure that the Spark cluster is able to resolve the hostname of the machine where the Job is executed, select the **Define the driver hostname or IP address** check box and in the field that is displayed, enter the IP address of this machine.

If you leave this check box clear, the Spark cluster looks at the machine located at 127.0.0.1, that is to say, the machine within the cluster itself for the Spark driver.

18. Press **F6** to run the Job.

The **Run** view is automatically opened in the lower part of the Studio and shows the execution progress of this Job.

Once done, you can check, for example in the web console of your HDFS system, that the output has been written in HDFS.



Browse Directory

/user/ychen/output_data/spark_batch Go!

Permission	Owner	Group	Size	Replication	Block Size	Name
drwxr-xr-x	ychen	ychen	0 B	0	0 B	out
drwxr-xr-x	ychen	ychen	0 B	0	0 B	reject

Hadoop, 2014.

Running a Job in Talend Administration Center

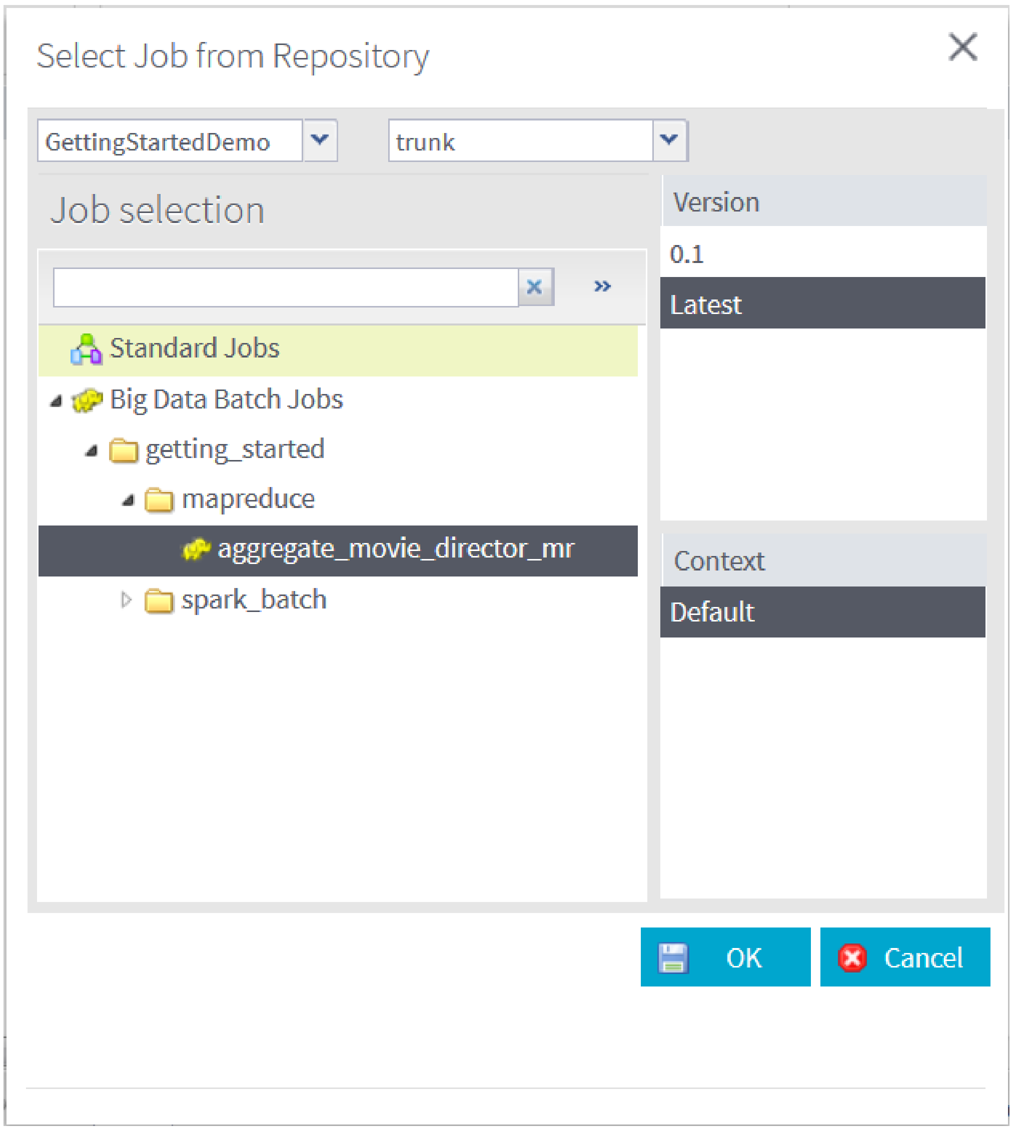
In the **Job Conductor** page of Talend Administration Center, you define an execution task to gather the script generation, deployment and execution phases of your MapReduce and Spark Batch Jobs.

- Ensure that the client machine on which the Talend Jobs are executed can recognize the host names of the nodes of the Hadoop cluster to be used. For this purpose, add the IP address/hostname mapping entries for the services of that Hadoop cluster in the `hosts` file of the client machine.

  In this use case, this machine is the one on which Talend Runtime is installed.
- The Hadoop cluster to be used has been properly configured and is running.
- The administrator of the cluster has given read/write rights and permissions to the username to be used for the access to the related data and directories in HDFS.
- You have created the use case Jobs described in the previous sections and run them successfully from the Studio.

1. Log in to Talend Administration Center with the account you have created in Setting up your first user and project on page 15.
2. In the **Menu** tree view of your Talend Administration Center, click **Job Conductor** to display the **Job conductor** page.
3. From the toolbar on the **Job Conductor** page, click **Add** > **Normal Task** to clear the **Execution task** configuration panel.
4. In the **Label** field, enter the name you want to give to the task to be triggered. For example, `getting_started`.
5. Click the icon to open a Job filter to search for the Job to be run from **Job conductor** and select it from the filter using its **Latest** version.

   For example, it can be the MapReduce Job described in Joining movie and director information using a MapReduce Job on page 44.



Once you have selected the Job, the **Project**, the **Branch**, the **Name**, the **Version** and the **Context** fields are all automatically filled with the related information of the selected Job.

6. Select the **Regenerate Job on change** check box to regenerate the selected Job before task deployment and execution every time a modification is made to the Job itself.

   Note that if you selected **Latest version**, in case a new version of the Job is created in Studio, the Job will be regenerated even if you did not select the **Regenerate Job on change** check box.

7. Select the server on which the task should be deployed.

   In this scenario, the server is the Talend Runtime service you have configured in Connecting Talend Runtime Container to Talend Administration Center on page 17.

8. Click **Save** to validate the configuration.

   This new task is added to the task list.

9. In the **Job conductor** page, click the **getting_started** task to select it and on the toolbar, click **Generate** to allow the task to fetch the relevant Job script in the relevant project from the Talend Studio **Repository** and generates the code.

   Once done, the status of the task changes to **Ready to deploy**, meaning that the code generated is now ready to be deployed on the execution server.

10. Click **Deploy** to deploy the Job on the execution server.

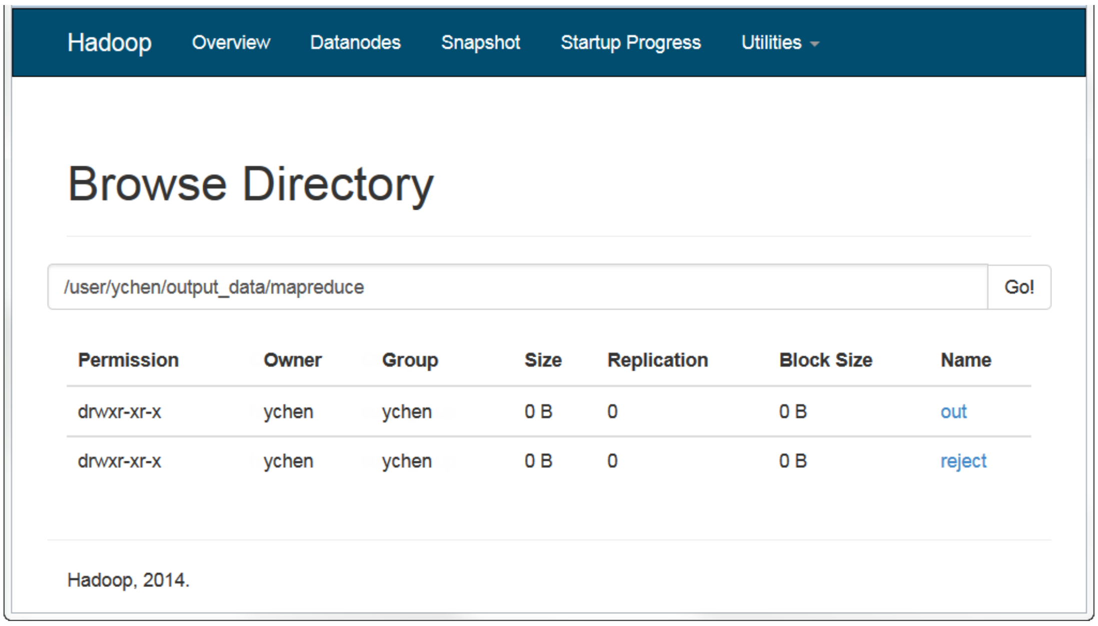
Once done, the status changes to **Ready to run**. This means that the server has received the Job and is now ready to execute it.

11. Click **Run** to execute the Job.

    Once done, the status switches back to **Ready to run**, which means that the Job can be run again if needed.

    In case the task did not complete properly, check the **Error Status** column as well as the task log for the Job completion information.

Once done, you can check, for example in the web console of your HDFS system, that the output has been written in HDFS.



Browse Directory

/user/ychen/output_data/mapreduce

Permission	Owner	Group	Size	Replication	Block Size	Name
drwxr-xr-x	ychen	ychen	0 B	0	0 B	out
drwxr-xr-x	ychen	ychen	0 B	0	0 B	reject

Hadoop, 2014.

Getting started with a Spark Streaming Job

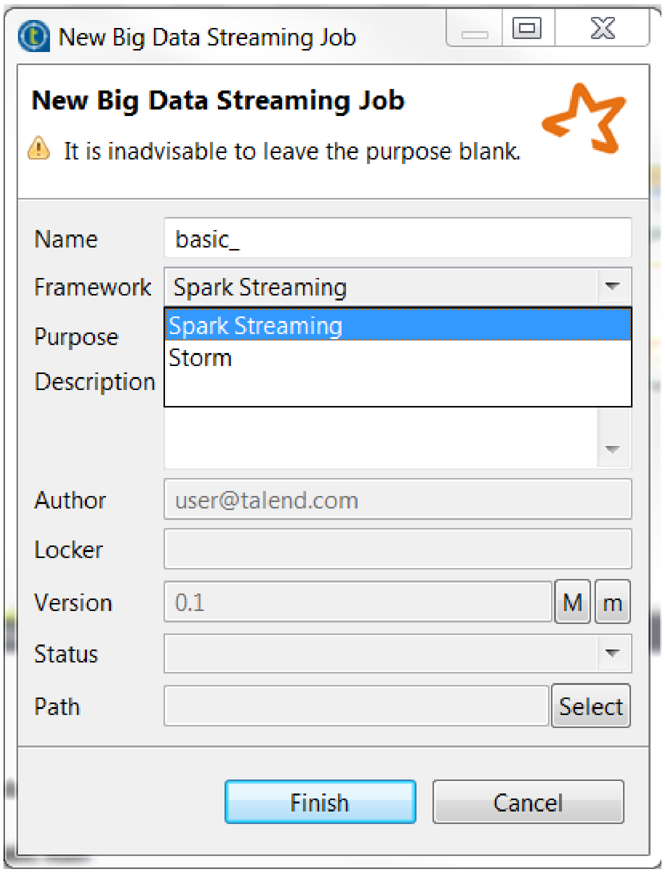
This section shows how to create a simple Spark Batch Job using the components provided in the Spark Streaming specific **Palette**.

For further information about the architecture on top of which aTalend Spark Streaming Job runs and as well about other related advanced features, see Talend Studio User Guide.

Creating a Spark Streaming Job

1. In the **Repository** tree view of the **Integration** perspective, right-click the **Job Designs** node and select **Create Big Data Streaming Job** from the contextual menu.

   The **New Big Data Streaming Job** wizard opens to help you define the main properties of the new Job.



New Big Data Streaming Job

New Big Data Streaming Job

It is inadvisable to leave the purpose blank.

Name: basic_

Framework: Spark Streaming

Purpose

Description

Author: user@talend.com

Locker

Version: 0.1

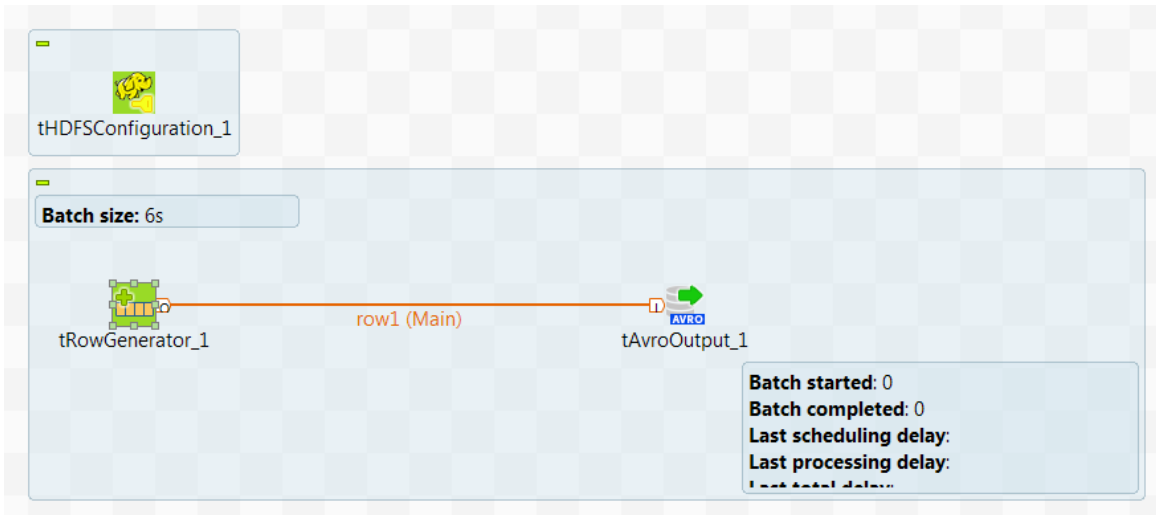
Status

Path

2. Once you click **Finish**, an empty design workspace opens up showing the name of the Spark Streaming Job as a tab label.

Scenario: Writing Avro data stream into HDFS

In this scenario, you created a very simple Spark Streaming Job. This Job will generate sample data stream by itself and write this stream in Avro format onto a given HDFS system.



Before replicating this scenario, ensure that you have appropriate rights and permissions to access the Hadoop distribution to be used.

Linking components

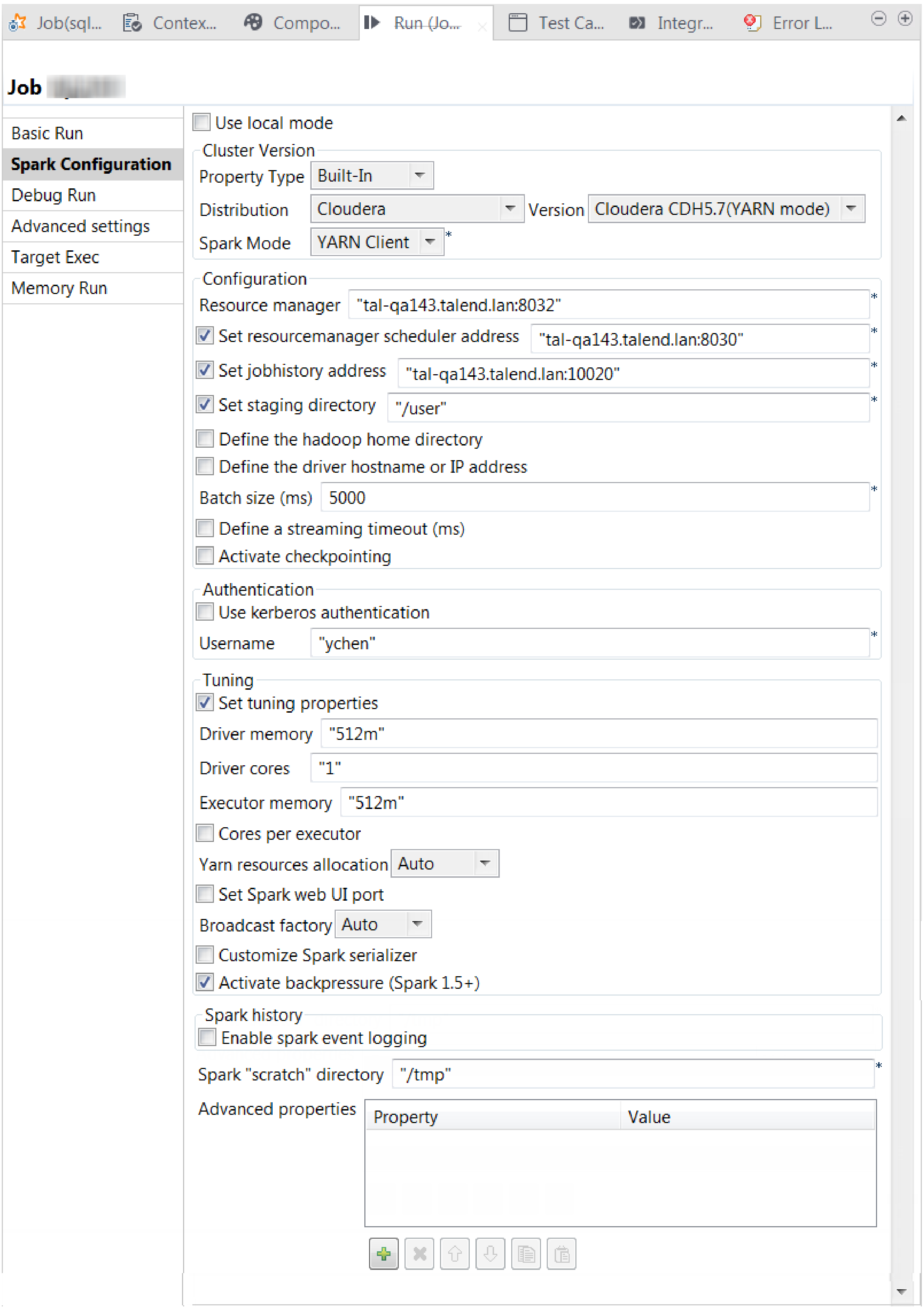
1. In the workspace of the empty Job you have created following the procedure detailed in Creating a Spark Streaming Job on page 61, enter the name of the component to be used and select this component from the list that appears. The components to be used are tHDFSConfiguration, tRowGenerator and tAvroOutput.

   The tHDFSConfiguration component provides the connection information to the HDFS system that the Job and the other components use to store temporary data, such as the dependant jar files.
2. Connect **tRowGenerator** to **tAvroOutput** using the **Row > Main** link.

Setting up Spark Streaming connection

1. Click **Run** to open its view and then click the **Spark Configuration** tab to display its view for configuring the Spark connection.

   This view looks like the image below:



Job(sql... | Contex... | Compo... | Run (Jo... | Test Ca... | Integr... | Error L...

Job [redacted]

Basic Run
Spark Configuration
Debug Run
Advanced settings
Target Exec
Memory Run

☐ Use local mode

Cluster Version
Property Type: Built-In
Distribution: Cloudera — Version: Cloudera CDH5.7(YARN mode)
Spark Mode: YARN Client *

Configuration
Resource manager: "tal-qa143.talend.lan:8032" *
☑ Set resourcemanager scheduler address: "tal-qa143.talend.lan:8030" *
☑ Set jobhistory address: "tal-qa143.talend.lan:10020" *
☑ Set staging directory: "/user" *
☐ Define the hadoop home directory
☐ Define the driver hostname or IP address
Batch size (ms): 5000 *
☐ Define a streaming timeout (ms)
☐ Activate checkpointing

Authentication
☐ Use kerberos authentication
Username: "ychen" *

Tuning
☑ Set tuning properties
Driver memory: "512m"
Driver cores: "1"
Executor memory: "512m"
☐ Cores per executor
Yarn resources allocation: Auto
☐ Set Spark web UI port
Broadcast factory: Auto
☐ Customize Spark serializer
☑ Activate backpressure (Spark 1.5+)

Spark history
☐ Enable spark event logging

Spark "scratch" directory: "/tmp" *

Advanced properties

Property	Value

2. Select the type of the Spark cluster you need to connect to.

- **Local**: the Studio builds the Spark environment in itself at runtime to run the Job locally within the Studio. With this mode, each processor of the local machine is used as a Spark worker to perform the computations. This mode requires minimum parameters to be set in this configuration view.

  Note this local machine is the machine in which the Job is actually run. The **Local** mode is the default mode and you need to clear its check box to display the drop-down list for you to select the other modes.
- **Standalone**: the Studio connects to a Spark-enabled cluster to run the Job from this cluster.
- **Yarn client**: the Studio runs the Spark driver to orchestrate how the Job should be performed and then send the orchestration to the Yarn service of a given Hadoop cluster so that the Resource Manager of this Yarn service requests execution resources accordingly.

3. With the **Yarn client** mode, the **Property type** list is displayed to allow you to select an established Hadoop connection from the **Repository**, on the condition that you have created this connection in the **Repository**. Then the Studio will reuse that set of connection information for this Job.

   For further information about how to create an Hadoop connection in **Repository**, see the chapter describing the **Hadoop cluster** node of the *Talend Studio User Guide*.
4. Select the version of the Hadoop distribution you are using along with Spark.

   - If you select **Microsoft HD Insight 3.4**, you need to configure the connections to the Livy service, the HD Insight service and the Windows Azure Storage service of that cluster in the areas that are displayed. A demonstration video about how to configure a connection to Microsoft HD Insight cluster is available in the following link: https://www.youtube.com/watch?v=A3QTT6VsNoM.

     The configuration of Livy is not presented in this video. The **Hostname** of Livy uses the following syntax: *your_spark_cluster_name.azurehdinsight.net*. For further information about the Livy service used by HD Insight, see Submit Spark jobs using Livy.
   - If you select **Amazon EMR**, you can find more details about how to configure an Amazon EMR cluster in Talend Help Center (https://help.talend.com). It is recommended to install your *Talend* Jobserver in the EMR cluster. For further information about this Jobserver, see *Talend Installation and Upgrade Guide* .

   1. Select **Import from existing version** to import an officially supported distribution as base and then add other required jar files which the base distribution does not provide.
   2. Select **Import from zip** to import the configuration zip for the custom distribution to be used. This zip file should contain the libraries of the different Hadoop/Spark elements and the index file of these libraries.

      In **Talend Exchange**, members of **Talend** community have shared some ready-for-use configuration zip files which you can download from this Hadoop configuration list and directly use them in your connection accordingly. However, because of the ongoing evolution of the different Hadoop-related projects, you might not be able to find the configuration zip corresponding to your distribution from this list; then it is recommended to use the **Import from existing version** option to take an existing distribution as base to add the jars required by your distribution.

      Note that custom versions are not officially supported by **Talend** . **Talend** and its community provide you with the opportunity to connect to custom versions from the Studio but cannot guarantee that the configuration of whichever version you choose will be easy. As such, you should only attempt to set up such a connection if you have sufficient Hadoop and Spark experience to handle any issues on your own.
5. Configure the connection information to the principal services of the cluster to be used.

If you are using the **Yarn client** mode, you need to enter the addresses of the following different services in their corresponding fields (if you leave the check box of a service clear, then at runtime, the configuration about this parameter in the Hadoop cluster to be used will be ignored):

- In the **Resource manager** field, enter the address of the ResourceManager service of the Hadoop cluster to be used.
- Select the **Set resourcemanager scheduler address** check box and enter the Scheduler address in the field that appears.
- Select the **Set jobhistory address** check box and enter the location of the JobHistory server of the Hadoop cluster to be used. This allows the metrics information of the current Job to be stored in that JobHistory server.
- Select the **Set staging directory** check box and enter this directory defined in your Hadoop cluster for temporary files created by running programs. Typically, this directory can be found under the *yarn.app.mapreduce.am.staging-dir* property in the configuration files such as *yarn-site.xml* or *mapred-site.xml* of your distribution.
- If you are accessing the Hadoop cluster running with Kerberos security, select this check box, then, enter the Kerberos principal names for the ResourceManager service and the JobHistory service in the displayed fields. This enables you to use your user name to authenticate against the credentials stored in Kerberos. These principals can be found in the configuration files of your distribution, such as in *yarn-site.xml* and in *mapred-site.xml*.

  If you need to use a Kerberos keytab file to log in, select **Use a keytab to authenticate**. A keytab file contains pairs of Kerberos principals and encrypted keys. You need to enter the principal to be used in the **Principal** field and the access path to the keytab file itself in the **Keytab** field.

  Note that the user that executes a keytab-enabled Job is not necessarily the one a principal designates but must have the right to read the keytab file being used. For example, the user name you are using to execute a Job is *user1* and the principal to be used is *guest*; in this situation, ensure that *user1* has the right to read the keytab file to be used.
- The **User name** field is available when you are not using Kerberos to authenticate. In the **User name** field, enter the login user name for your distribution. If you leave it empty, the user name of the machine hosting the Studio will be used.

  Since the Job needs to upload jar files to HDFS of the cluster to be used, you must ensure that this user name is the same as the one you have put in **tHDFSConfiguration**, the component used to provides HDFS connection information to Spark.

If you are using the **Standalone** mode, you need to set the following parameters:

- In the **Spark host** field, enter the URI of the Spark Master of the Hadoop cluster to be used.
- In the **Spark home** field, enter the location of the Spark executable installed in the Hadoop cluster to be used.

6. If you need to launch from Windows, it is recommended to specify where the *winutils.exe* program to be used is stored.

- If you know where to find your *winutils.exe* file and you want to use it, select the **Define the Hadoop home directory** check box and enter the directory where your *winutils.exe* is stored.
- Otherwise, leave this check box clear, the Studio generates one by itself and automatically uses it for this Job.

7. If the Spark cluster cannot recognize the machine in which the Job is launched, select this **Define the driver hostname or IP address** check box and enter the host name or the IP address of this machine. This allows the Spark master and its workers to recognize this machine to find the Job and thus its driver.

Note that in this situation, you also need to add the name and the IP address of this machine to its *host* file.

8. In the **Batch size** field, enter the time interval at the end of which the Job reviews the source data to identify changes and processes the new micro batches.
9. If needs be, select the **Define a streaming timeout** check box and in the field that is displayed, enter the time frame at the end of which the streaming Job automatically stops running.
10. If you need the Job to be resilient to failure, select the **Activate checkpointing** check box to enable the Spark checkpointing operation. In the field that is displayed, enter the directory in which Spark stores, in the file system of the cluster, the context data of the computations such as the metadata and the generated RDDs of this computation.

    For further information about the Spark checkpointing operation, see http://spark.apache.org/docs/latest/streaming-programming-guide.html#checkpointing .
11. Select the **Set Tuning properties** check box to optimize the allocation of the resources to be used to run this Job. These properties are not mandatory for the Job to run successfully, but they are useful when Spark is bottlenecked by any resource issue in the cluster such as CPU, bandwidth or memory.
12. In the **Spark "scratch" directory** field, enter the directory in which the Studio stores in the local system the temporary files such as the jar files to be transferred. If you launch the Job on Windows, the default disk is *C:*. So if you leave */tmp* in this field, this directory is *C:/tmp*.
13. In the **Yarn client** mode, you can enable the Spark application logs of this Job to be persistent in the file system. To do this, select the **Enable Spark event logging** check box.

    The parameters relevant to Spark logs are displayed:

    - **Spark event logs directory**: enter the directory in which Spark events are logged. This is actually the *spark.eventLog.dir* property.
    - **Spark history server address**: enter the location of the history server. This is actually the *spark.yarn.historyServer.address* property.
    - **Compress Spark event logs**: if needs be, select this check box to compress the logs. This is actually the *spark.eventLog.compress* property.

    Since the administrator of your cluster could have defined these properties in the cluster configuration files, it is recommended to contact the administrator for the exact values.
14. In the **Advanced properties** table, add any Spark properties you need to use to override their default counterparts used by the Studio.

Hortonworks Data Platform V2.4	• `spark.yarn.am.extraJavaOptions: -Dhdp.version=2.4.0.0-169` • `spark.driver.extraJavaOptions: -Dhdp.version=2.4.0.0-169` In addition, you need to add `-Dhdp.version=2.4.0.0-169` to the **JVM settings** area either in the **Advanced settings** tab of the **Run** view or in the Talend > Run/Debug view of the **[Preferences]** window. Setting this argument in the **[Preferences]** window applies it on all the Jobs that are designed in the same Studio.
MapR V5.1 and V5.2	When the cluster is used with the HBase or the MapRDB components: *spark.hadoop.yarn.application.classpath*: enter the value of this parameter specific to your cluster and add, if missing, the classpath to

	HBase to ensure that the Job to be used can find the required classes and packages in the cluster. For example, if the HBase version installed in the cluster is *1.1.1*, copy and paste all the paths defined with the *spark.hadoop.yarn.application.classpath* parameter from your cluster and then add *opt/mapr/hbase/hbase-1.1.1/lib/** and */opt/mapr/lib/** to these paths, separating each path with a comma(,). The added paths is where HBase is usually installed in a MapR cluster. If your HBase is installed elsewhere, contact the administrator of your cluster for details and adapt these paths accordingly. For a step-by-step explanation about how to add this parameter, see You can find more details about how to run HBase/MapR-DB on Spark with a MapR distribution in Talend Help Center (https://help.talend.com).

For further information about the valid Spark properties, see Spark documentation at https://spark.apache.org/docs/latest/configuration.

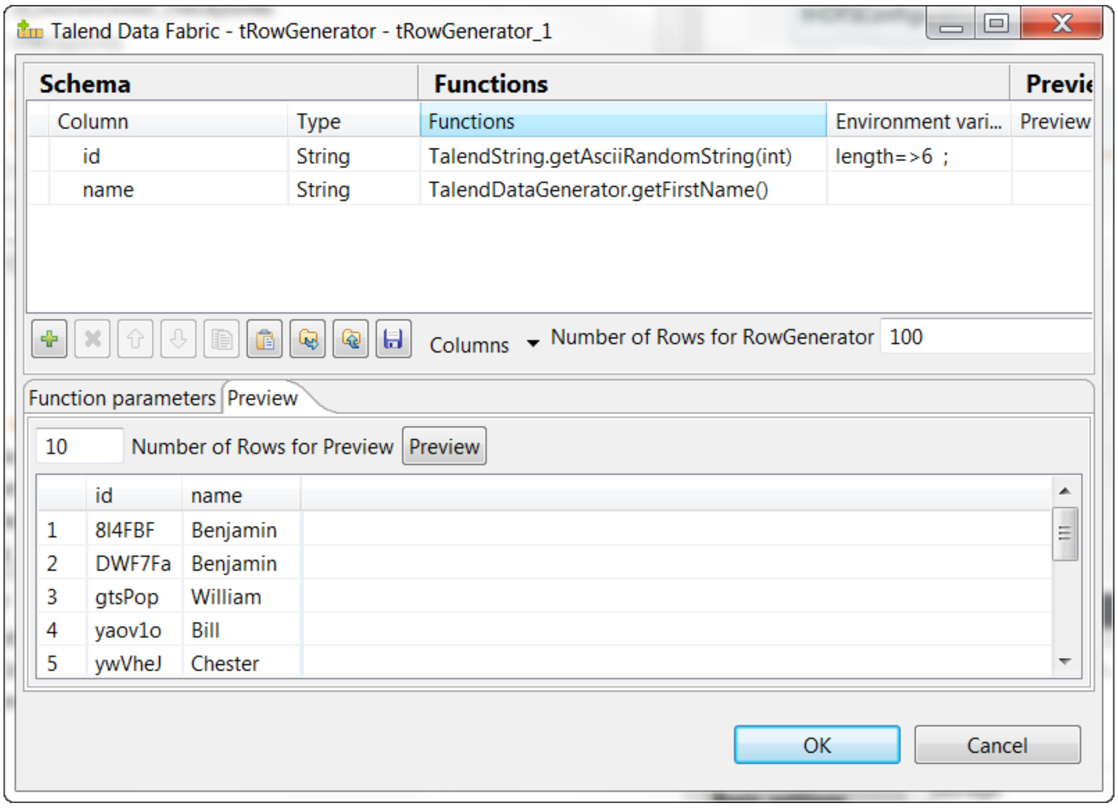
Configuring the connection to the file system to be used by Spark

1. Double-click **tHDFSConfiguration** to open its **Component** view. Note that **tHDFSConfiguration** is used because the Spark **Yarn client** mode is used to run Spark Jobs in this scenario.

   Spark uses this component to connect to the HDFS system to which the jar files dependent on the Job are transferred.
2. In the **Version** area, select the Hadoop distribution you need to connect to and its version.
3. In the **NameNode URI** field, enter the location of the machine hosting the NameNode service of the cluster.
4. In the **Username** field, enter the authentication information used to connect to the HDFS system to be used. Note that the user name must be the same as you have put in the **Spark configuration** tab.

Generating sample data

1. Double-click **tRowGenerator** to open its **RowGenerator Editor** view.



Column	Type	Functions	Environment vari...	Preview
id	String	TalendString.getAsciiRandomString(int)	length=>6 ;	
name	String	TalendDataGenerator.getFirstName()		

Number of Rows for RowGenerator 100

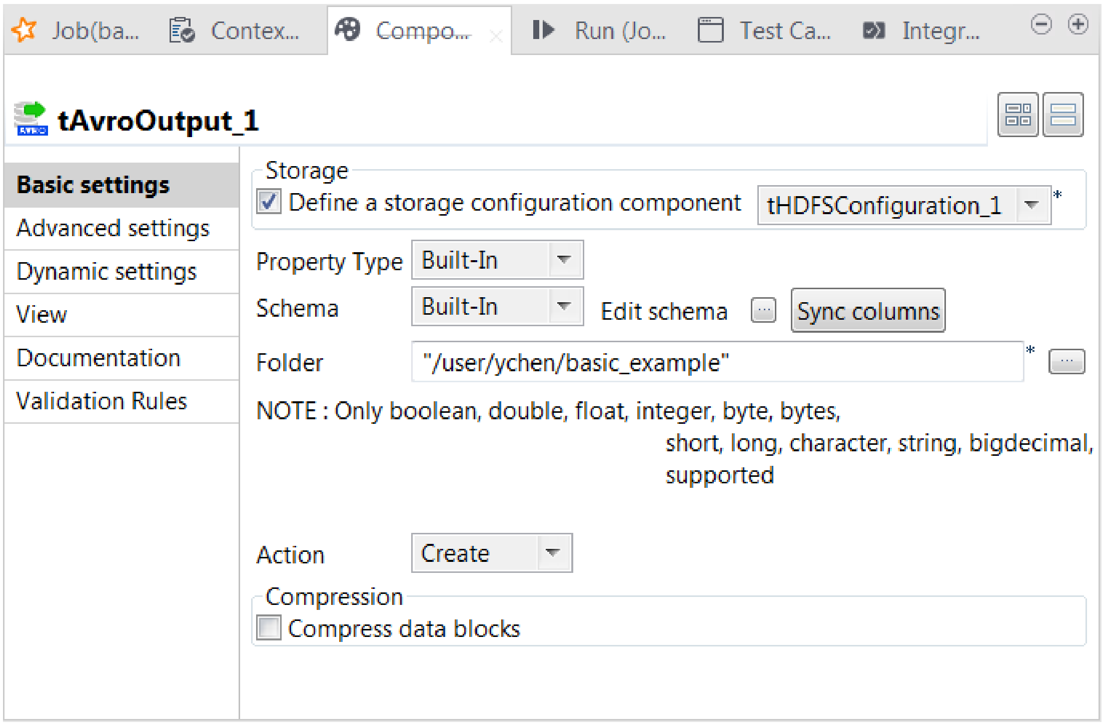
10 Number of Rows for Preview

	id	name
1	8I4FBF	Benjamin
2	DWF7Fa	Benjamin
3	gtsPop	William
4	yaov1o	Bill
5	ywVheJ	Chester

2. Click the **[+]** button twice to add two columns and name them to `id` and `name`, respectively.
3. In the **Functions** column, select the `TalendString.getAsciiRandomString(int)` function for `id` and `TalendDataGenerator.getFirstName()` for `name`.
4. In the **Number of rows for RowGenerator** field, enter the number of rows to be created in each generation. In this example, enter `100`.
5. Click **OK** to validate these changes and accept the propagation prompted by the pop-up dialog box.
6. In the **Input repetition interval** field in the **Basic settings** view, enter the time interval (in milliseconds) every time at the end of which **tRowGenerator** generates a set of data, the set of `100` rows. In this example, it is `3000` milliseconds.

Writing data to HDFS

1. Double-click **tAvroOutput** to open its **Component** view.



tAvroOutput_1

Basic settings
Advanced settings
Dynamic settings
View
Documentation
Validation Rules

Storage
☑ Define a storage configuration component: tHDFSConfiguration_1

Property Type: Built-In
Schema: Built-In
Edit schema
Sync columns
Folder: "/user/ychen/basic_example"

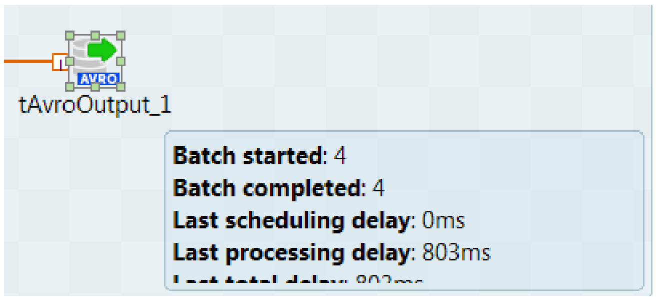
NOTE : Only boolean, double, float, integer, byte, bytes, short, long, character, string, bigdecimal, supported

Action: Create

Compression
☐ Compress data blocks

2. Ensure that the **Define a storage configuration component** check box is selected.
3. In the **Folder** field, enter the path, or browse to the folder in which you want to write the data.
4. From the **Action** list, select the operation you need to perform on the folder in question. If the folder already exists, select `Overwrite`; otherwise, select `Create`.
5. Press **F6** to run this Job.

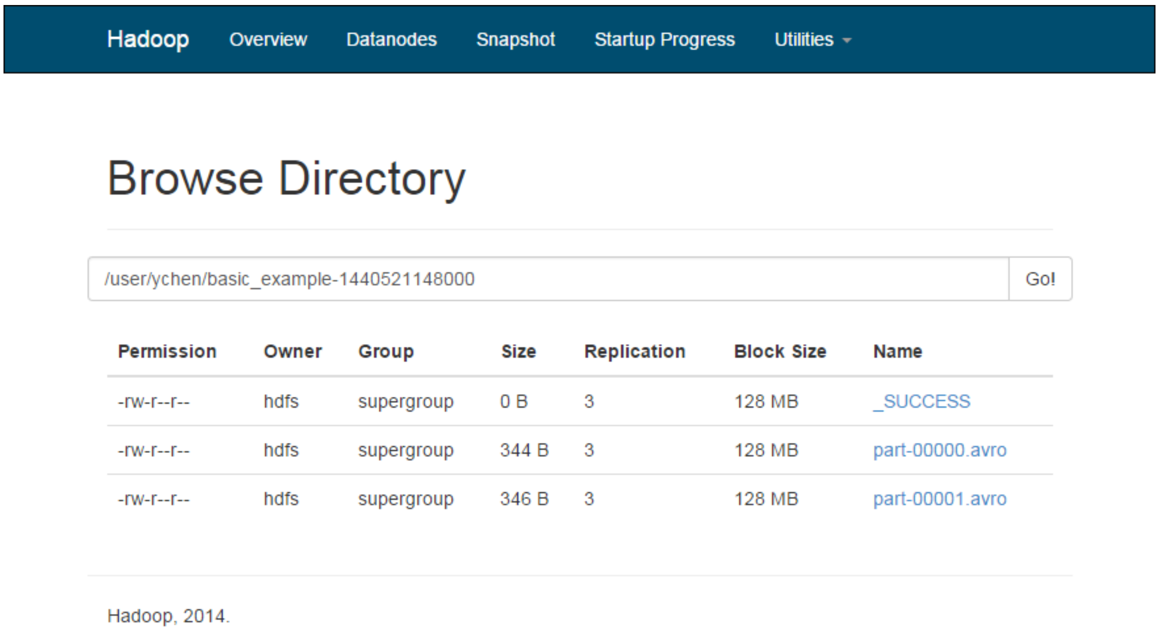
The statistics area in the workspace shows the progress of the processing of the generated datasets, segmented and counted as batches.



Batch started: 4
Batch completed: 4
Last scheduling delay: 0ms
Last processing delay: 803ms

Once done, view the result in the web console of the HDFS system being used.

You can read that the `basic_example-` folder is continuously created and click to open one of them, you can see avro files written by this Job.



Browse Directory

/user/ychen/basic_example-1440521148000

Permission	Owner	Group	Size	Replication	Block Size	Name
-rw-r--r--	hdfs	supergroup	0 B	3	128 MB	_SUCCESS
-rw-r--r--	hdfs	supergroup	344 B	3	128 MB	part-00000.avro
-rw-r--r--	hdfs	supergroup	346 B	3	128 MB	part-00001.avro

Hadoop, 2014.

Note that not all the `basic_example-` folders contain actual data.

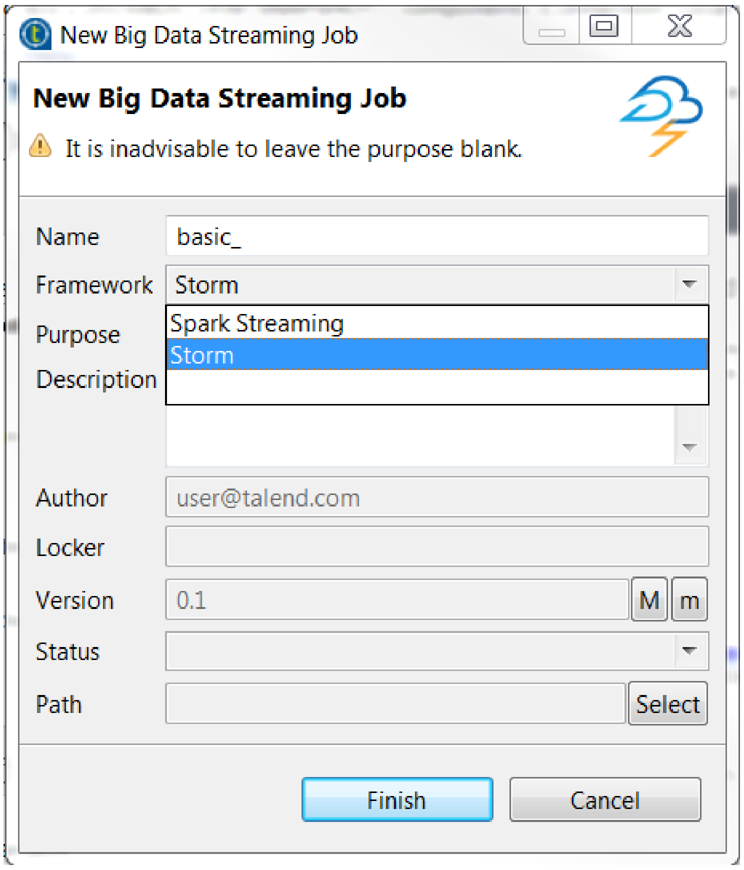
Getting started with a Storm Job

This section presents more details of this template by explaining a series of actions to be taken to create a Storm Job, or in other words, a Storm topology.

For further information about the architecture on top of which a Talend Storm Job runs and as well about other related advanced features, see Talend Studio User Guide.

1. Once the Studio is launched, in the **Repository** of the **Integration** perspective, right-click the **Job designs** node, and then from the contextual menu, select **Create Big Data Streaming Job**.

   The **New Big Data Streaming Job** wizard opens to help you define the main properties of the new Job.


New Big Data Streaming Job

New Big Data Streaming Job

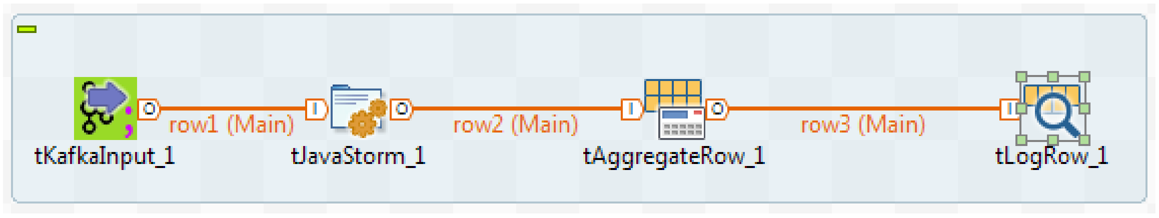
It is inadvisable to leave the purpose blank.

Name	basic_
Framework	Storm
Purpose	
Description	
Author	user@talend.com
Locker	
Version	0.1
Status	
Path	

2. Once you click **Finish**, an empty design workspace opens up showing the name of the Spark Streaming Job as a tab label.

Scenario: analyzing people's activities using a Storm topology

In this scenario, a four-component Storm Job (a topology) is created to transmit messages about the activities of some given people to the topology you are designing in order to analyze the popularities of those activities.



This Job subscribes to the related topic created by the Kafka topic producer, which means you need to install the Kafka cluster in your messaging system to maintain the feeds of messages. For further information about the Kafka messaging service, see Apache's documentation about Kafka.

On the other hand, since this Job runs on top of Storm, you need to ensure that your Storm system is ready for use. For further information about Storm, see Apache's documentation about Storm.

Note that when you use the Storm system installed in the Hortonwork Data Platform 2.1 (HDP2.1), ensure that the Storm DRPC (distributed remote procedure call) servers' names have been properly defined in the Custom storm.yaml section of the Config tab of Storm in Ambari's web console. For example, you need to use two Storm DRPC servers which are "Server1" and "Server2", then you must define them in the Custom storm.yaml secton as follows: `[Server1,Server2]`.

Producing the sample messages

In the real-world practice, the system that produces messages to Kafka is completely decoupled. While in this scenario, Kafka itself is used to produce the sample messages. You need to perform the following operations to produce these messages:

1. Create the Kafka topic to be used to categorize the messages. The following command is used for demonstration purposes only. If you need further information about the creation of a Kafka topic, see Apache's documentation for Kafka.
   ```
   /usr/lib/kafka/bin/kafka-topics.sh --zookeeper localhost:2181 --
   create --topic activities --partitions 1 --replication-factor 1
   ```
   This command creates a topic named `activities`, using the Kafka brokers managed by the Zookeeper service on the `localhost` machine.
2. Publish the message you want to analyze to the `activities` topic you have just created. In this scenario, Kafka is used to perform this publication using, for example, the following command:

   ```
   echo 'Ryan|M|Pool
   Remy|M|Drink
   Remy|M|Hiking
   Irene|F|Drink
   Pierre|M|Movie
   Irene|F|Pool
   Thomas|M|Drink
   Ryan|M|Cooking
   Wang|F|Cooking
   Chen|M|Drink | /usr/lib/kafka/bin/kafka-console-producer.sh --broker-list  localhost:9092 --
   topic  activities
   ```

   This command publishes 10 simple messages

   ```
   Ryan|M|Pool
   Remy|M|Drink
   Remy|M|Hiking
   Irene|F|Drink
   Pierre|M|Movie
   Irene|F|Pool
   Thomas|M|Drink
   Ryan|M|Cooking
   Wang|F|Cooking
   Chen|M|Drink
   ```

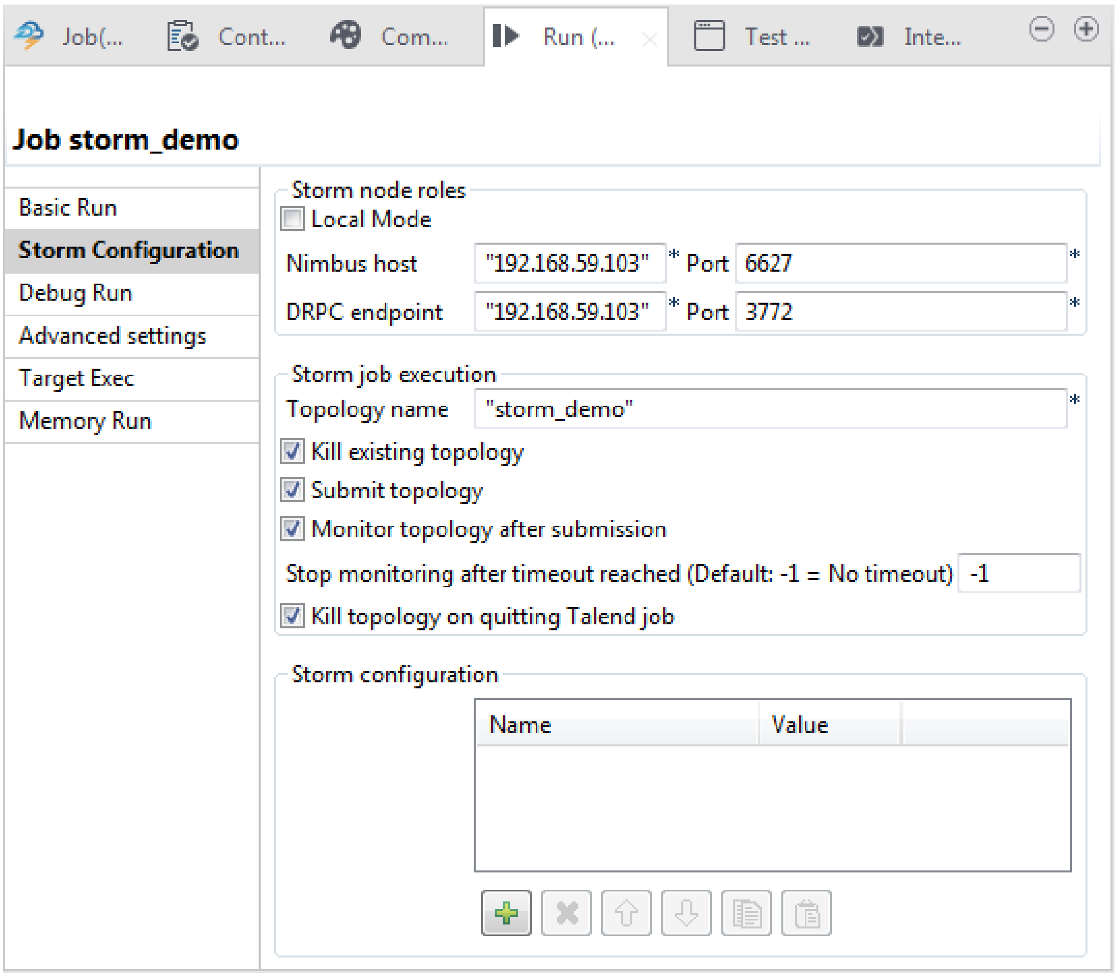
As explained previously, you can use your actual message producer system instead to perform the publication.

Linking the components

1. In the **Integration** perspective of the Studio, create an empty Storm Job from the **Job Designs** node in the **Repository** tree view.
2. In the workspace, enter the name of the component to be used and select this component from the list that appears. In this scenario, the components are tKafkaInput, tJavaStorm, tAggregateRow, and tLogRow.
3. Connect them using **Row** > **Main** links.

Configuring the connection

1. Click **Run** to open its view and then click **Storm configuration** to set up the connection to the Storm system to be used.



Job storm_demo

Basic Run
Storm Configuration
Debug Run
Advanced settings
Target Exec
Memory Run

Storm node roles
- [ ] Local Mode

Nimbus host "192.168.59.103" * Port 6627 *
DRPC endpoint "192.168.59.103" * Port 3772 *

Storm job execution
Topology name "storm_demo" *
- [x] Kill existing topology
- [x] Submit topology
- [x] Monitor topology after submission

Stop monitoring after timeout reached (Default: -1 = No timeout) -1
- [x] Kill topology on quitting Talend job

Storm configuration

Name	Value

2. In the **Storm node roles** area, indicate the locations of the Nimbus server and the DRPC endpoint of the Storm cluster to be used.

Option	Description
Local mode	Select this check box to build the Storm environment within the Studio. In this situation, the Storm Job you are designing is run locally within the Studio and you do not need to define any specific Nimbus and DRPC endpoint addresses.
Nimbus host and **Port**	In these two fields, enter the location of the Storm Nimbus server and its port number, respectively.

Option	Description
DRPC endpoint and **Port**	In these two fields, enter the location of the Storm DRPC endpoint and its port number, respectively.

3. In the **Topology name** field, enter the name you want the Storm system to use for the Storm Job, or topology in terms of the Storm, you are designing. It is recommended to use the same name as you have named this Storm Job so that you can easily recognize this Job even within the Storm system.
4. Select the **Kill existing topology** check box to make the Storm system stop any topology that has the same name as the Job you are designing and want to run.

   If you clear this check box and that Job of the same name is already running in the Storm system, the current Job will fail when you send it to Storm to run.
5. Select the **Submit topology** check box to submit the current Job to the Storm system. This feature is used when you need to send a new topology to Storm or used together with the **Kill existing topology** feature when you need to update a running topology in the Storm system.

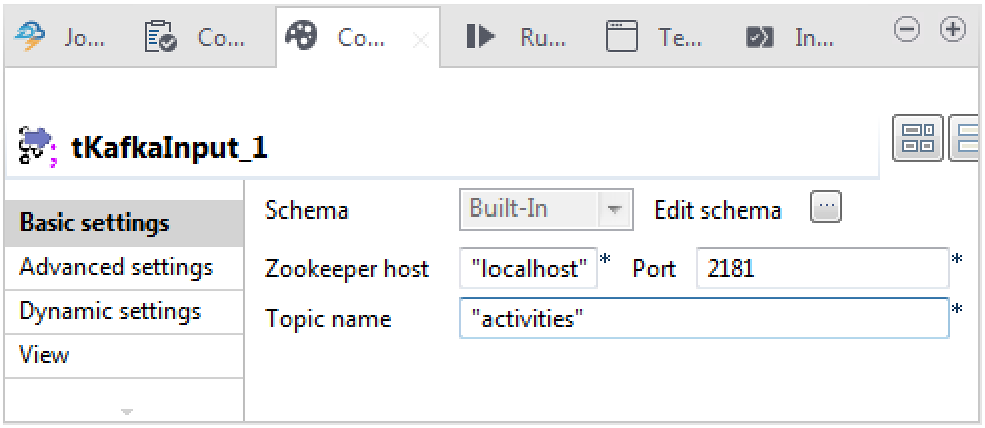
   If you clear this check box, when you run the current Job, the submission of this Job is ignored while the other configuration information is still taken into account, for example, killing an existing topology.
6. Select the **Monitor topology after submission** check box to monitor the current Storm Job in the console of the **Run** view.

   If you clear this check box, you cannot read the monitoring information in the console.
7. In the **Stop monitoring after timeout reached** field, enter, without the double quotation marks, the numeric value to indicate whether to stop the monitoring when a running topology reaches its timeout. The default value is `-1`, which means no timeout is applied.
8. Select the **Kill topology on quitting Talend Job** check box to allow the Studio to kill, if it is still running, the current Job from the Storm system when you stop this Job from the Studio.

   If you clear this check box, the topology for this Job continues to run in the Storm system even though you kill it within the Studio or eventually shut down the Studio.
9. If you need to use any other Storm properties specific to your situation, add them to the **Storm configuration** table.

Receiving the message from the Kafka channel

1. Double-click **tKafkaInput** to open its **Component** view.



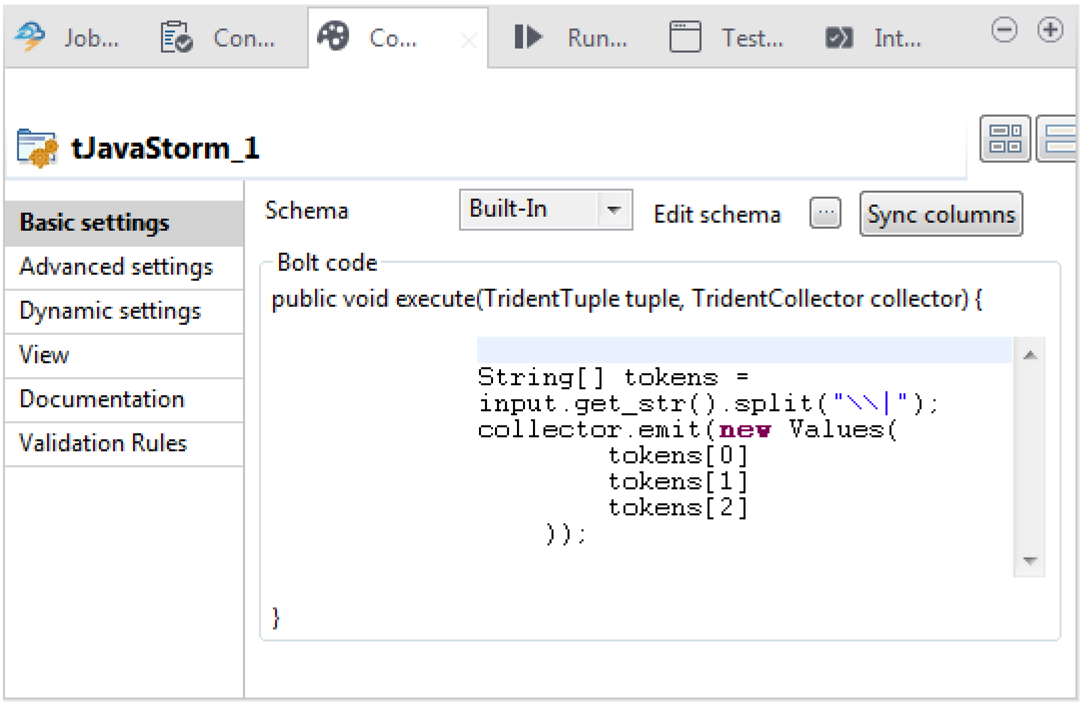
Schema: Built-In
Zookeeper host: "localhost" * Port: 2181 *
Topic name: "activities" *

2. In the **Zookeeper host** field, enter the location of the Zookeeper service used to coordinate the Kafka cluster to be used.
3. In the **Port** field, enter the port number of this Zookeeper service.

4. In the **Topic name** field, enter the name of the topic in which you need to receive messages. In this scenario, the topic is `activities`.

Extracting information from the message

1. Double-click **tJavaStorm** to open its **Component** view.

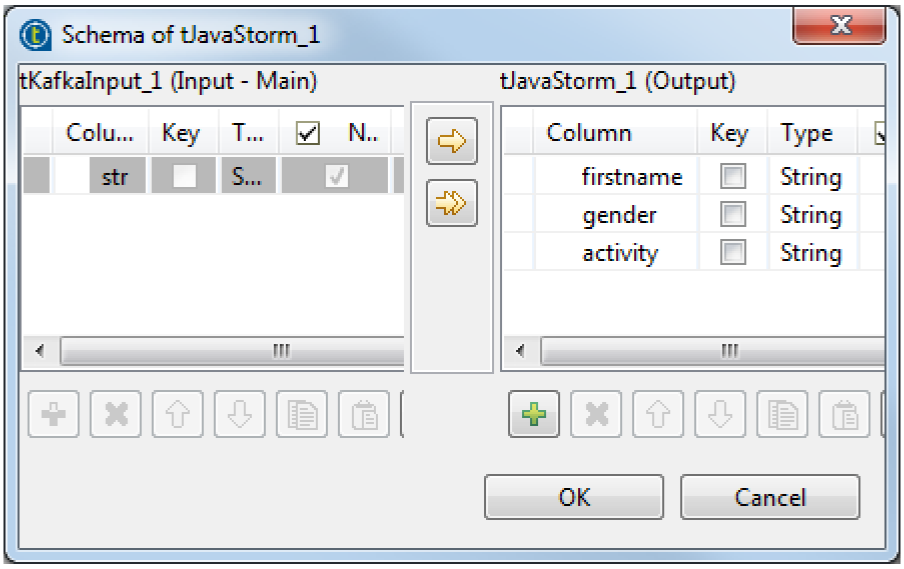


```
public void execute(TridentTuple tuple, TridentCollector collector) {

		String[] tokens =
		input.get_str().split("\\|");
		collector.emit(new Values(
			tokens[0]
			tokens[1]
			tokens[2]
		));

}
```

2. Click the **[...]** button next to **Edit schema** to open the schema editor.
3. On the output side (right), click the **[+]** button three times to add three rows and in the **Column** column, rename them to `firstname`, `gender` and `activity`, respectively. These columns correspond to the information you can extract from the sample message.



Colu...	Key	T...	✓	N...
str		S...	✓	

Column	Key	Type
firstname		String
gender		String
activity		String

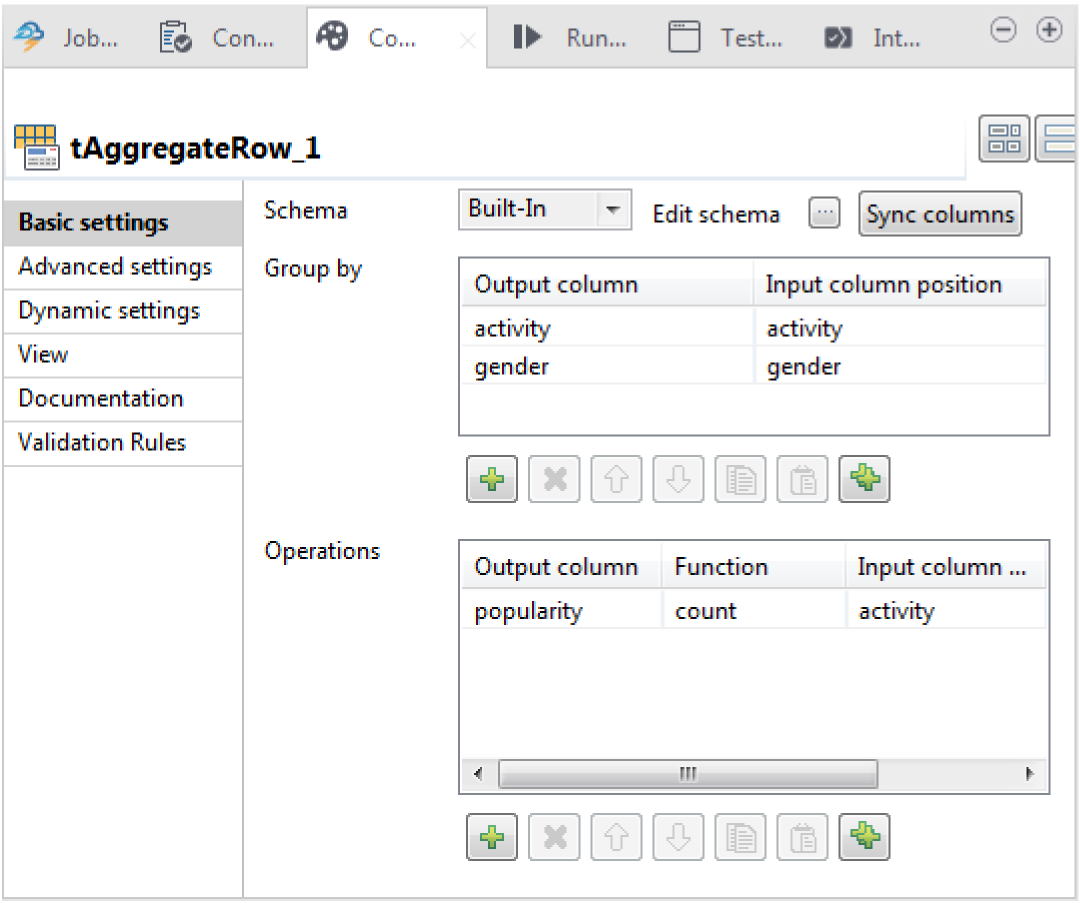
4. Click **OK** to validate these changes and accept the propagation prompted by the pop-up dialog box.
5. In the **Bolt code** area, enter the main method of the bolt to be executed.

   In this scenario, the code is as follows:

```
String[] tokens = input.get_str().split("\\|");
collector.emit(new Values( tokens[0], tokens[1], tokens[2] ));
```

Aggregating the extracted information

1. Double-click **tAggregateRow** to open its **Component** view. This component allows you to find out the most popular activity recorded in the received messages.



Schema: Built-In

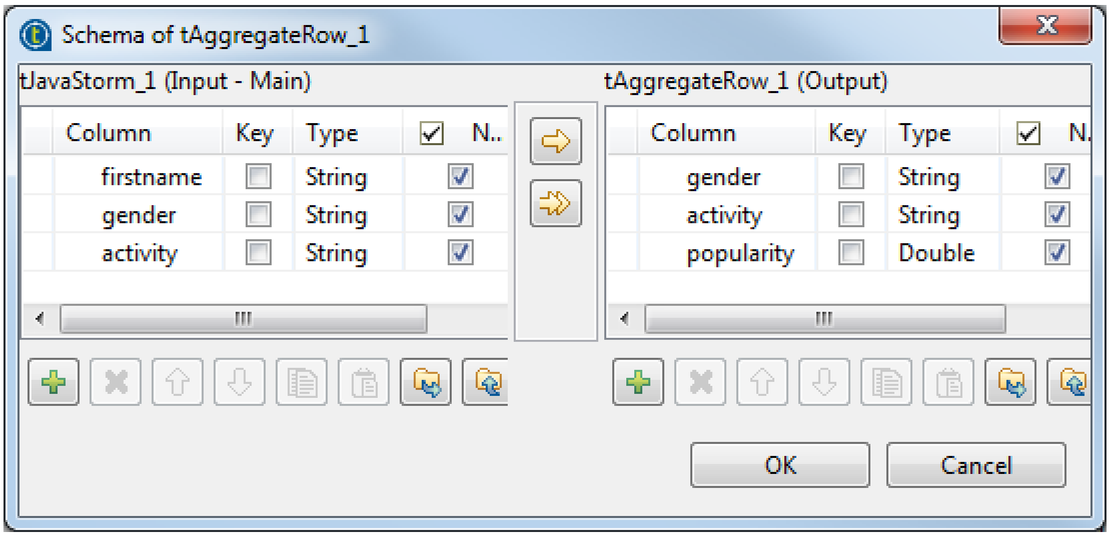
Group by

Output column	Input column position
activity	activity
gender	gender

Operations

Output column	Function	Input column ...
popularity	count	activity

2. Click the **[...]** button next to **Edit schema** to open the schema editor.
3. On the output side (right), click the **[+]** button three times to add three rows and in the **Column** column, rename them to `activity`, `gender` and `popularity`, respectively.



tJavaStorm_1 (Input - Main)

Column	Key	Type	✓ N..
firstname	☐	String	✓
gender	☐	String	✓
activity	☐	String	✓

tAggregateRow_1 (Output)

Column	Key	Type	✓ N.
gender	☐	String	✓
activity	☐	String	✓
popularity	☐	Double	✓

4. In the **Type** column of the `popularity`popularity row of the output side, select `Double`.
5. Click **OK** to validate these changes and accept the propagation prompted by the pop-up dialog box.
6. In the **Group by** table, add two rows by clicking the **[+]** button twice and configure these two rows as follows to group the outputted data:

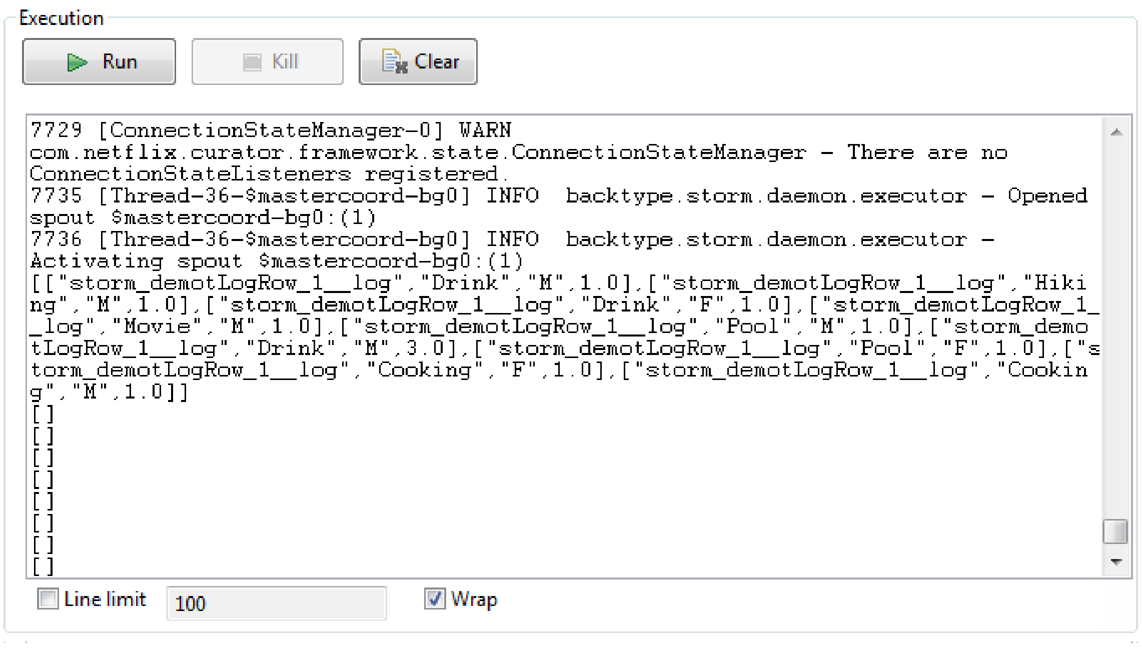
- **Output column**: select the columns from the output schema to be used as the conditions to group the outputted data. In this example, they are `activity` and `gender`.
- **Input column position**: select the columns from the input schema to send data to the output columns you have selected in the **Output column** column. In this scenario, they are `activity` and `gender`.

7. In the **Operations** table, add one row by clicking the **[+]** button once and configure this row as follows to calculate the popularity of each activity:

   - **Output column**: select the column from the output schema to carry the calculated results. In this scenario, it is `popularity`.
   - **Function**: select the function to be used to process the incoming data. In this scenario, select `count`. It counts the frequency of each activity in the received messages.
   - **Input column position**: select the column from the input schema to provide the data to be processed. In this scenario, it is `activity`.

8. Press **F6** to run this Job

Once done, the **Run** view is opened automatically, where you can check the execution result.



```
7729 [ConnectionStateManager-0] WARN
com.netflix.curator.framework.state.ConnectionStateManager - There are no
ConnectionStateListeners registered.
7735 [Thread-36-$mastercoord-bg0] INFO  backtype.storm.daemon.executor - Opened
spout $mastercoord-bg0:(1)
7736 [Thread-36-$mastercoord-bg0] INFO  backtype.storm.daemon.executor -
Activating spout $mastercoord-bg0:(1)
[["storm_demotLogRow_1__log","Drink","M",1.0],["storm_demotLogRow_1__log","Hiki
ng","M",1.0],["storm_demotLogRow_1__log","Drink","F",1.0],["storm_demotLogRow_1_
_log","Movie","M",1.0],["storm_demotLogRow_1__log","Pool","M",1.0],["storm_demo
tLogRow_1__log","Drink","M",3.0],["storm_demotLogRow_1__log","Pool","F",1.0],["s
torm_demotLogRow_1__log","Cooking","F",1.0],["storm_demotLogRow_1__log","Cookin
g","M",1.0]]
[]
[]
[]
[]
[]
[]
[]
[]
```

You can read that the activity `Drink` is the most popular with 3 occurrences for the gender `M` (Male) and 1 occurrence for the gender `F` (Female) in the messages.

The Storm topology continues to run, waiting for messages to appear on the Kafka message broker until you kill the Job. In this scenario, because the **Kill topology on quitting Talend Job** check box is selected, the Storm topology will be stopped and removed from the cluster when this Job is stopped.

What's next?

You have seen how Talend Studio , as well as Talend Administration Center, helps you manage your big data using Talend Jobs. You have learned how to access and move your data to a given Hadoop cluster via Talend Jobs, filter and transform your data, and store the filtered and transformed data in the HDFS system of the Hadoop cluster. Along the way, you have learned how to centralize frequently used Hadoop connections in the **Repository** and easily reuse these connections in your Jobs.

To learn more about Talend Studio, see:

- Talend Studio User Guide
- Talend components documentation

To learn more about Talend Administration Center, see Talend Administration Center User Guide.

To ensure that your data is clean, you can try Talend Open Studio for Data Quality and Talend Data Preparation Free Desktop.

To learn more about Talend products and solutions, visit www.talend.com.