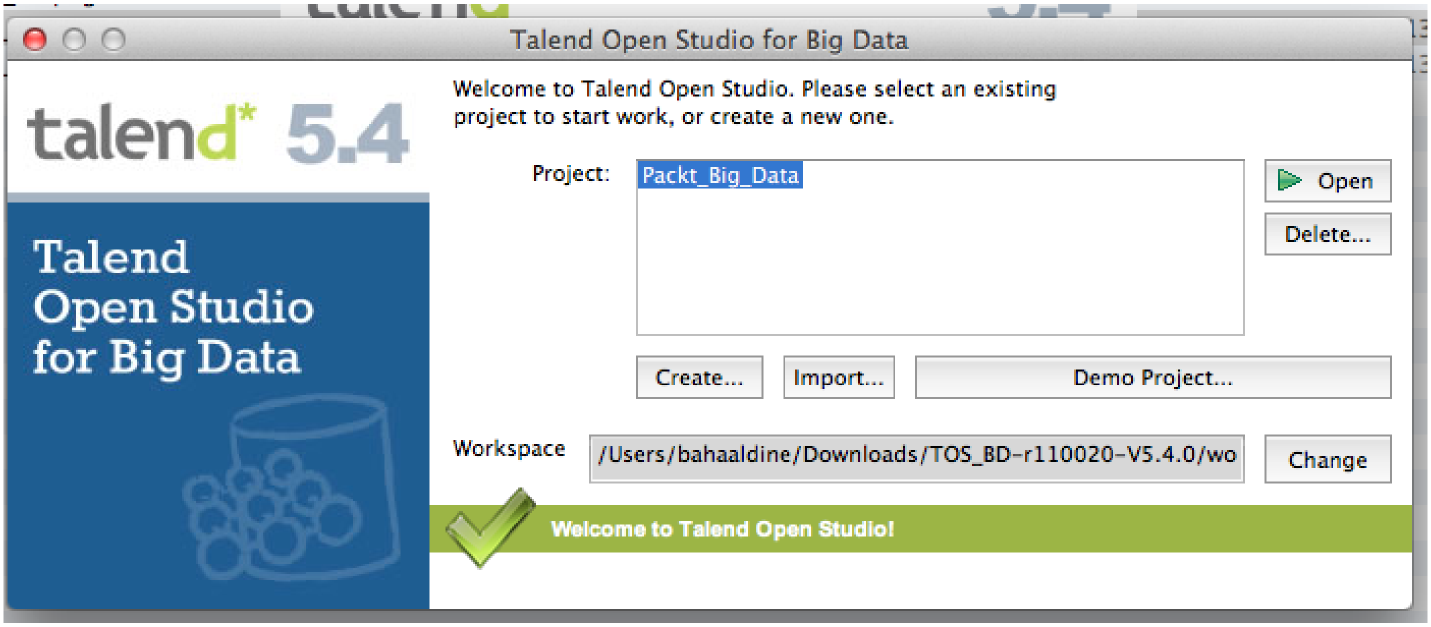
# 1. Getting Started with Talend for Big Data

and versatile big data integration and quality...



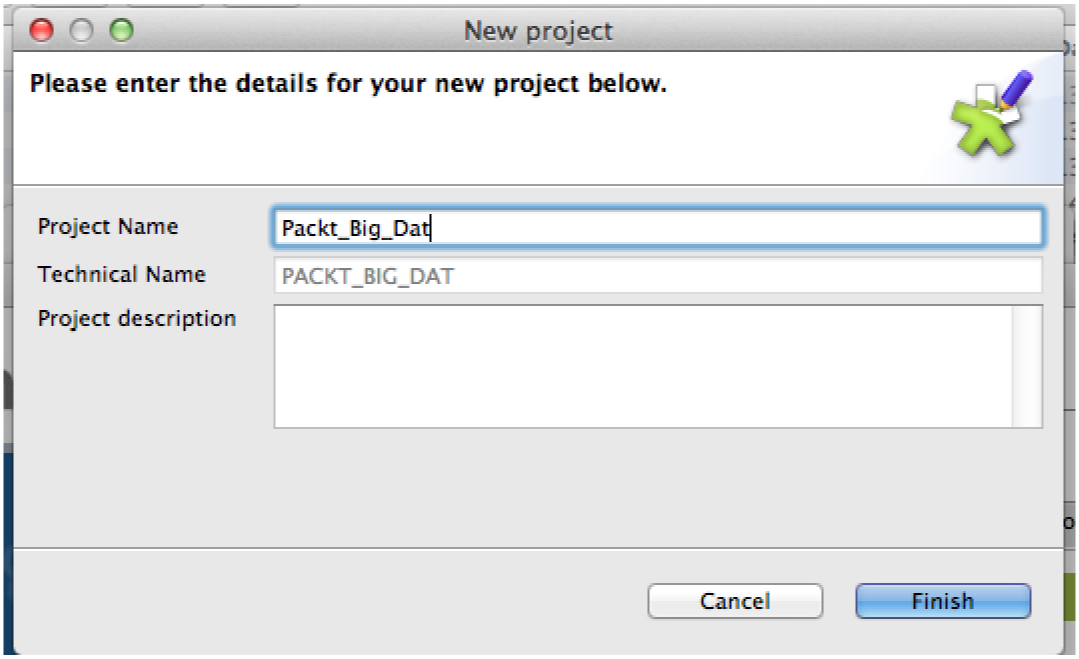
Talend Open Studio for Big Data

Welcome to Talend Open Studio. Please select an existing project to start work, or create a new one.

Project: Packt_Big_Data

Workspace /Users/bahaaldine/Downloads/TOS_BD-r110020-V5.4.0/wo

Welcome to Talend Open Studio!

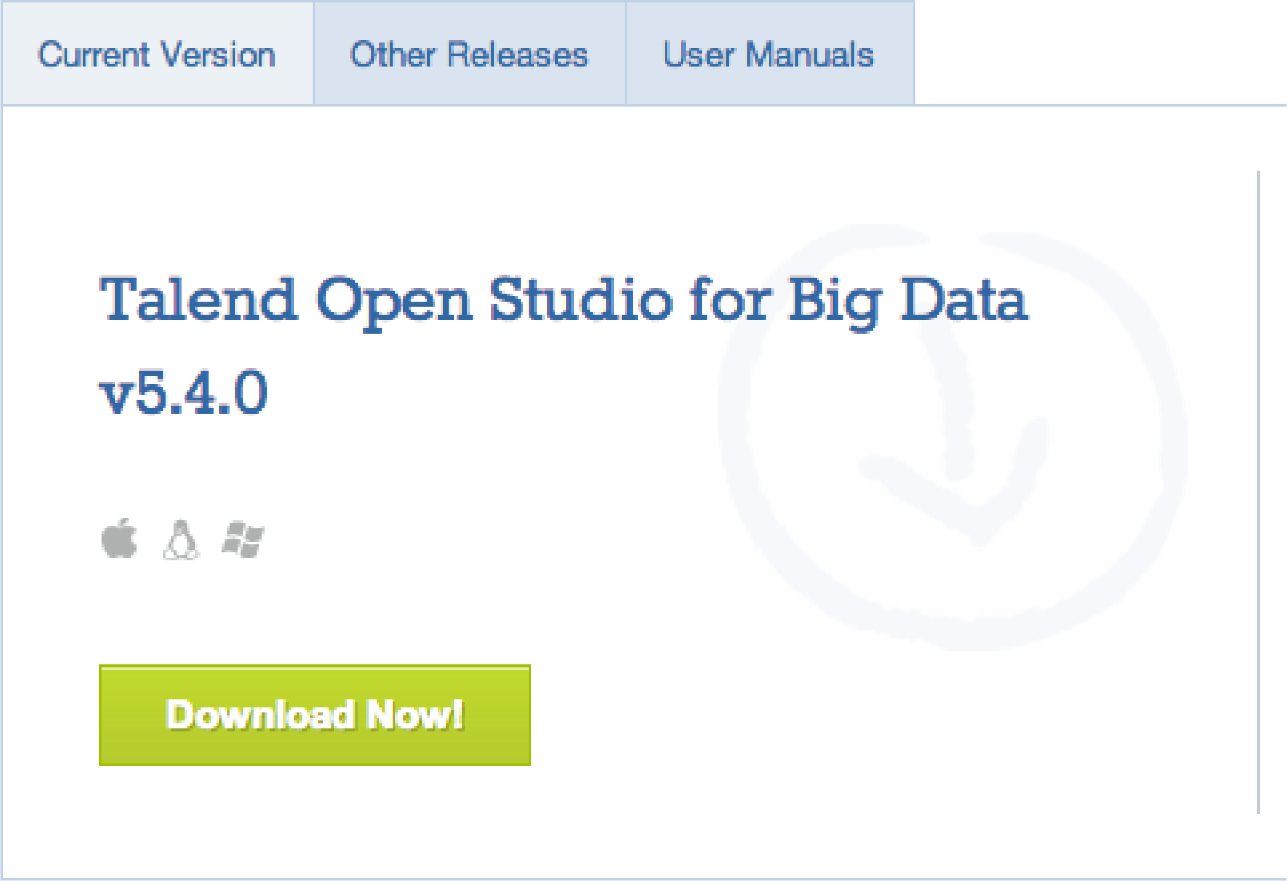


Please enter the details for your new project below.

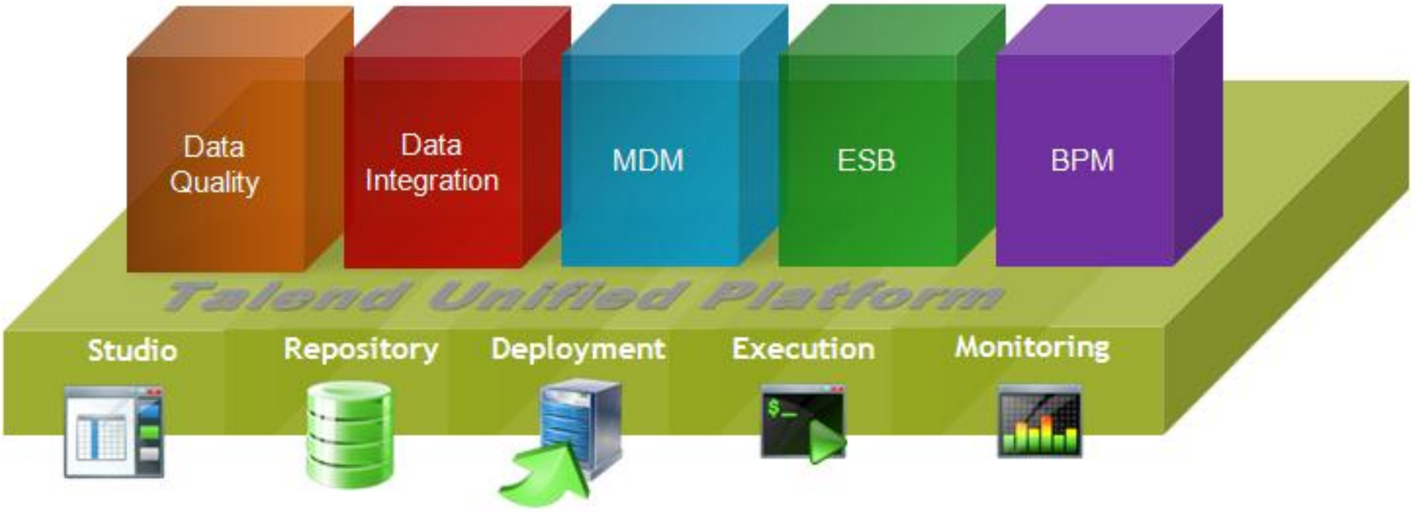
Project Name: Packt_Big_Dat

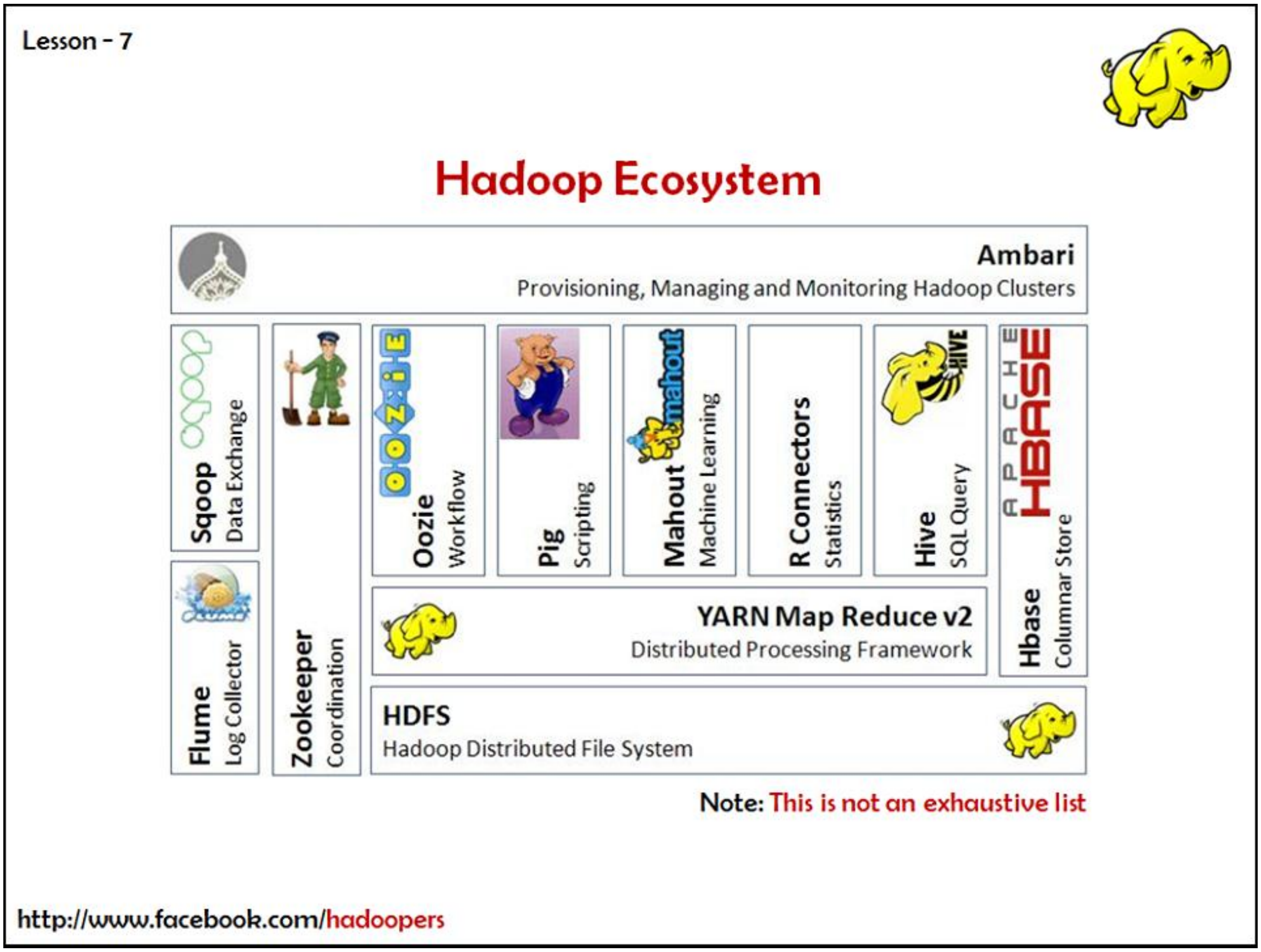
Technical Name: PACKT_BIG_DAT

Project description:



Talend Open Studio for Big Data v5.4.0





Lesson - 7

## Hadoop Ecosystem

**Ambari**
Provisioning, Managing and Monitoring Hadoop Clusters

**Sqoop** Data Exchange

**Flume** Log Collector

**Zookeeper** Coordination

**Oozie** Workflow

**Pig** Scripting

**Mahout** Machine Learning

**R Connectors** Statistics

**Hive** SQL Query

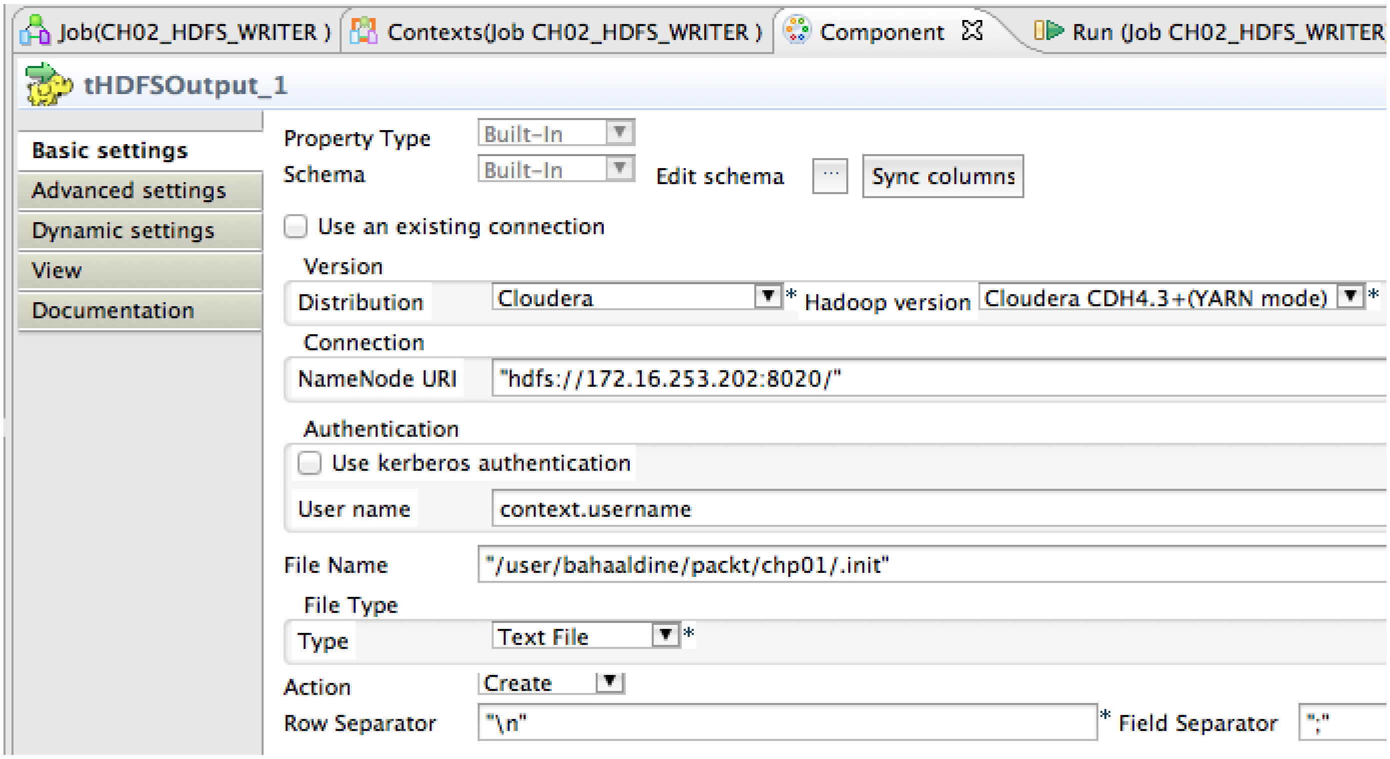
**Hbase** Columnar Store

**YARN Map Reduce v2**
Distributed Processing Framework

**HDFS**
Hadoop Distributed File System

**Note:** This is not an exhaustive list

http://www.facebook.com/hadoopers



tHDFSOutput_1

Property Type: Built-In

Schema: Built-In

□ Use an existing connection

Version

Distribution: Cloudera * Hadoop version: Cloudera CDH4.3+(YARN mode) *

Connection

NameNode URI: "hdfs://172.16.253.202:8020/"

Authentication

□ Use kerberos authentication

User name: context.username

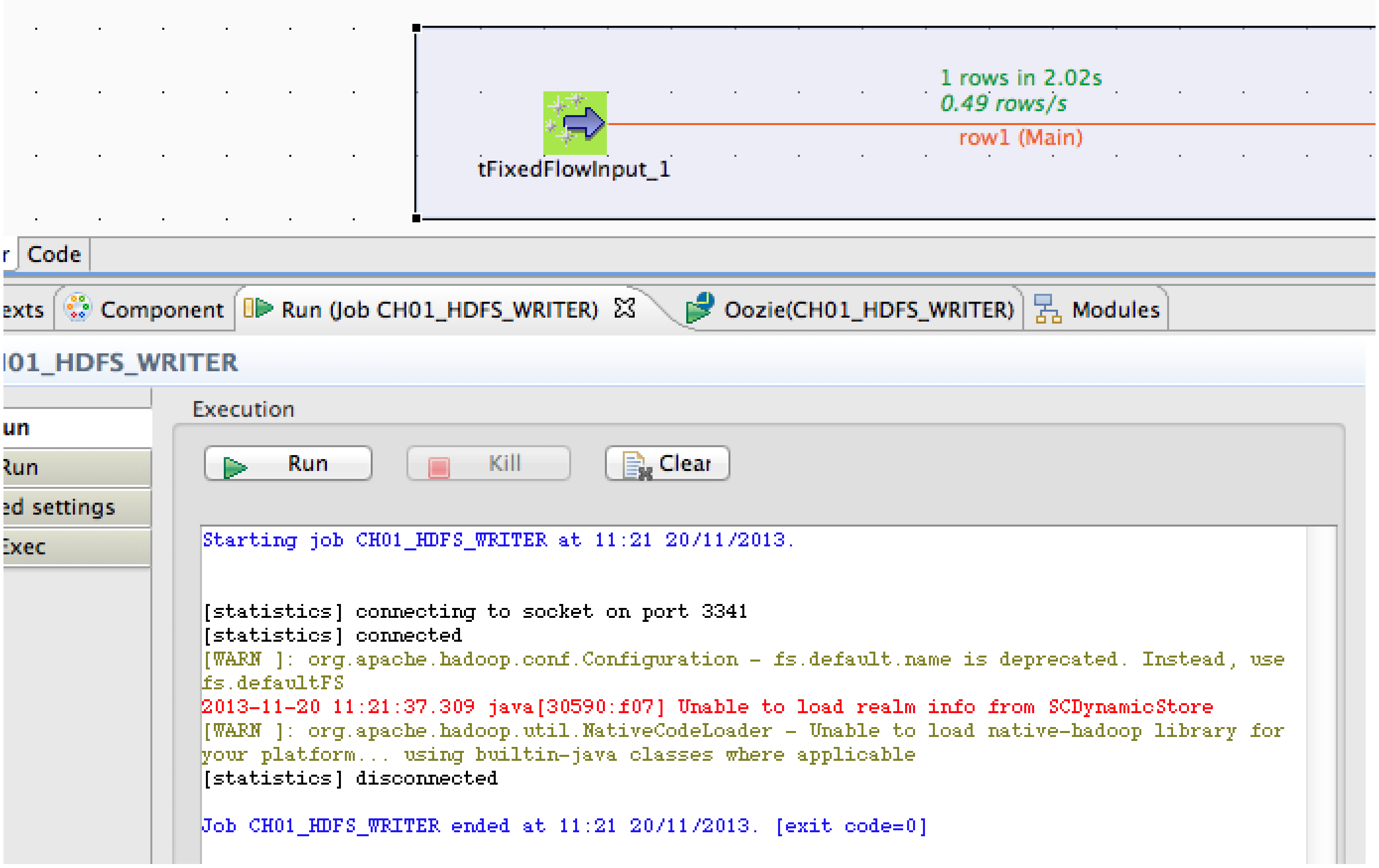
File Name: "/user/bahaaldine/packt/chp01/.init"

File Type

Type: Text File *

Action: Create

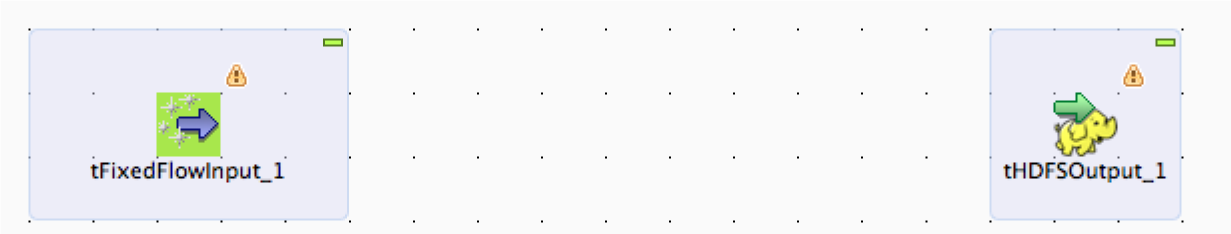
Row Separator: "\n" * Field Separator: ";"

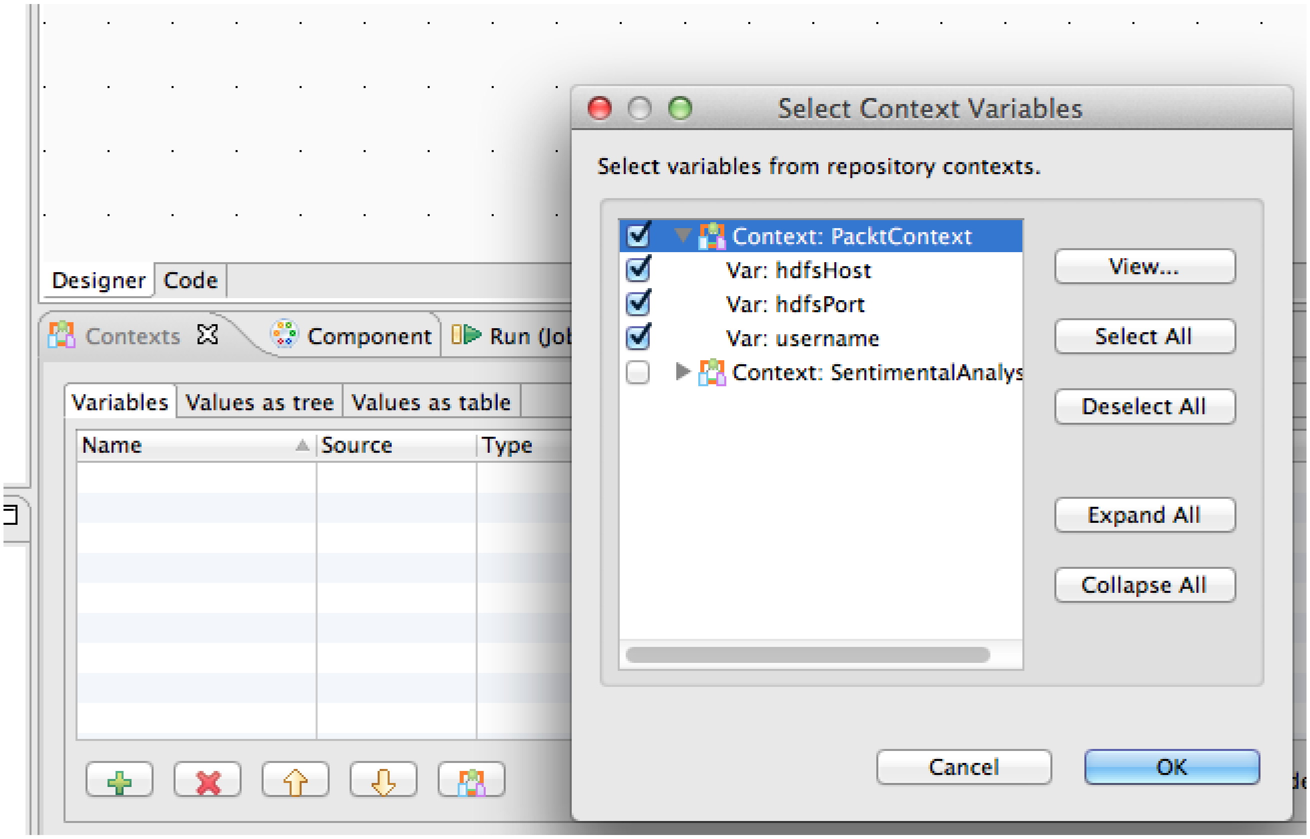


```
Starting job CH01_HDFS_WRITER at 11:21 20/11/2013.


[statistics] connecting to socket on port 3341
[statistics] connected
[WARN ]: org.apache.hadoop.conf.Configuration - fs.default.name is deprecated. Instead, use
fs.defaultFS
2013-11-20 11:21:37.309 java[30590:f07] Unable to load realm info from SCDynamicStore
[WARN ]: org.apache.hadoop.util.NativeCodeLoader - Unable to load native-hadoop library for
your platform... using builtin-java classes where applicable
[statistics] disconnected

Job CH01_HDFS_WRITER ended at 11:21 20/11/2013. [exit code=0]
```

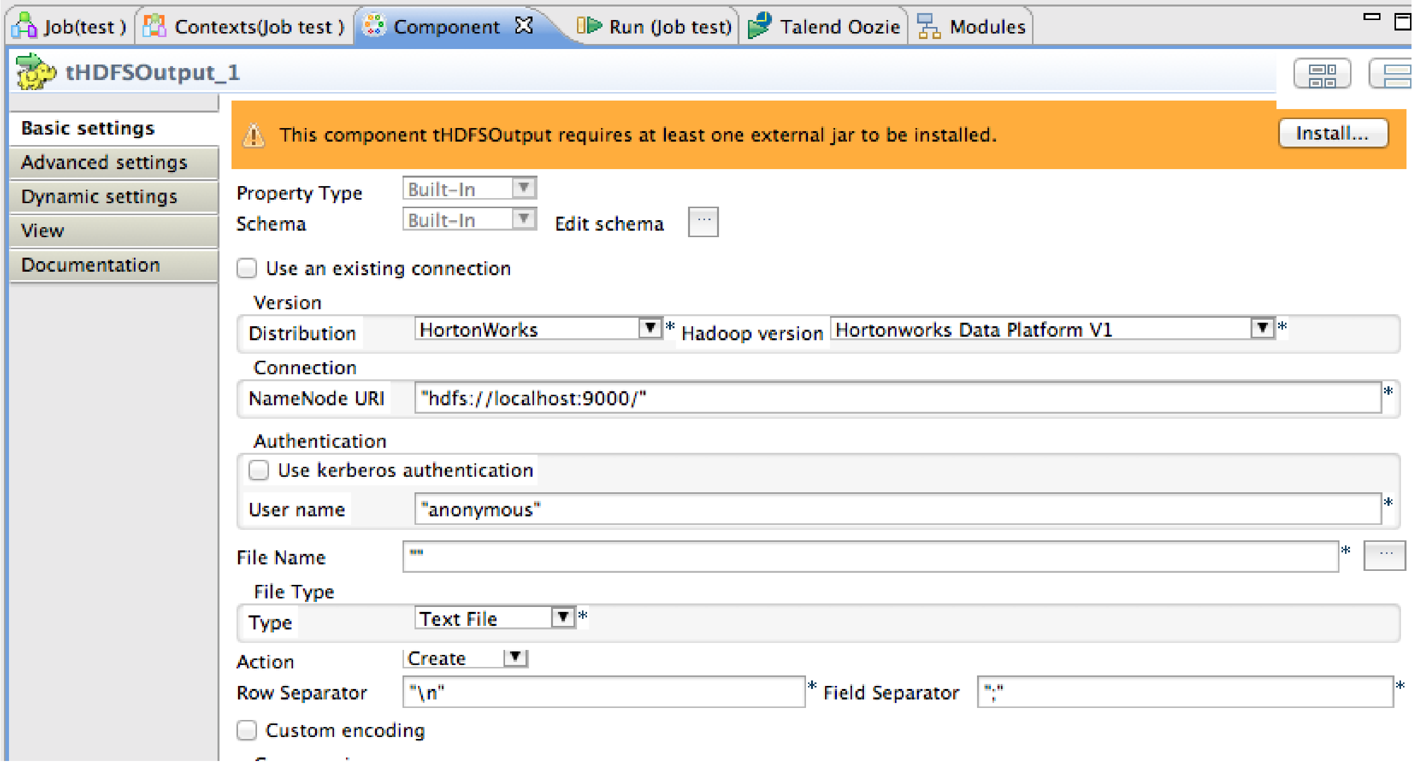




Select Context Variables

Select variables from repository contexts.

- ☑ Context: PacktContext
  - ☑ Var: hdfsHost
  - ☑ Var: hdfsPort
  - ☑ Var: username
- □ Context: SentimentalAnalys



tHDFSOutput_1

This component tHDFSOutput requires at least one external jar to be installed.

Property Type: Built-In

Schema: Built-In

□ Use an existing connection

Version

Distribution: HortonWorks * Hadoop version: Hortonworks Data Platform V1 *

Connection

NameNode URI: "hdfs://localhost:9000/" *

Authentication

□ Use kerberos authentication

User name: "anonymous" *

File Name: "" *

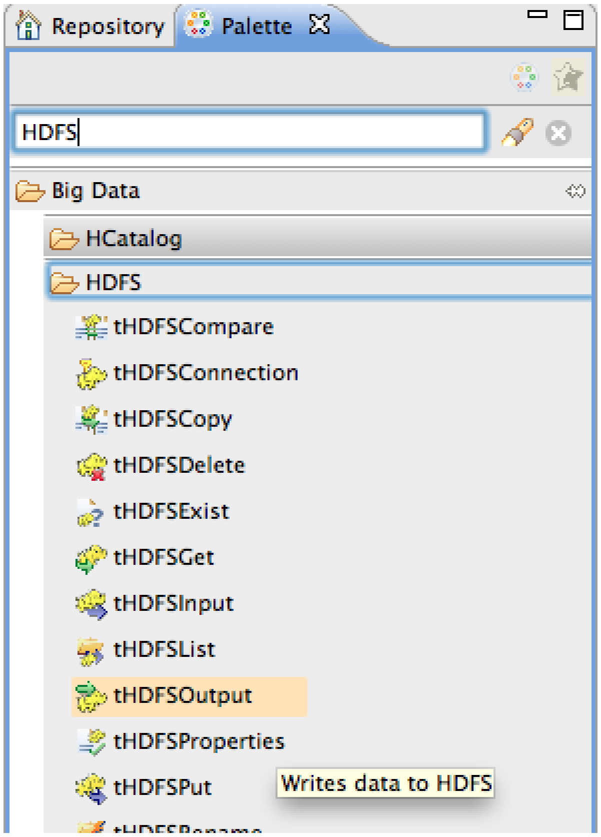
File Type

Type: Text File *

Action: Create

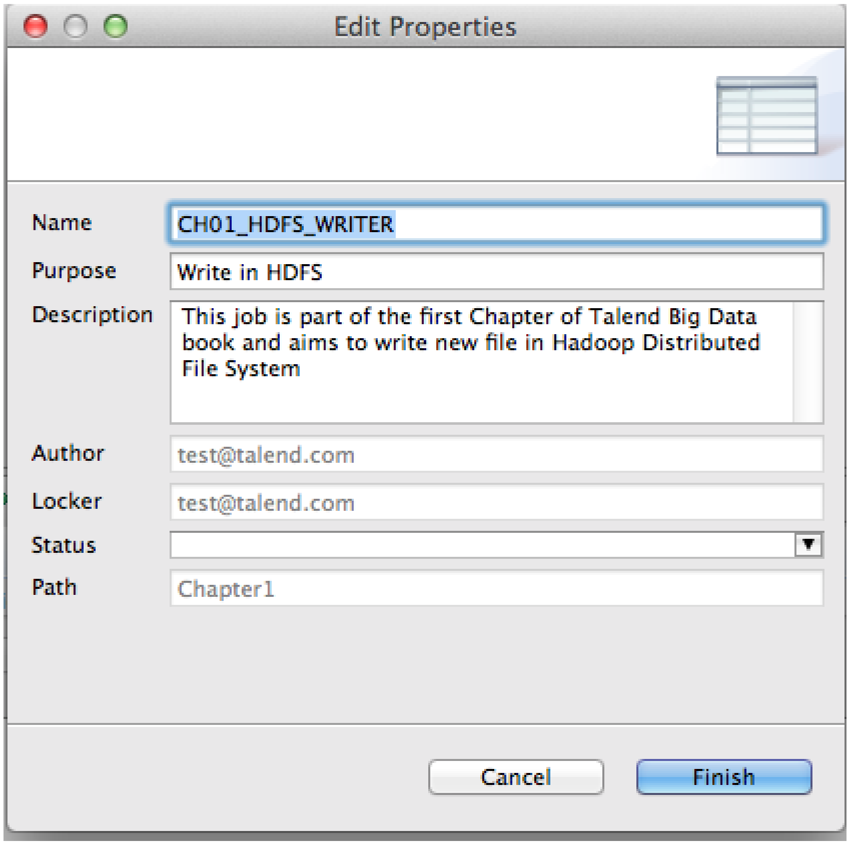
Row Separator: "\n" * Field Separator: ";" *

□ Custom encoding

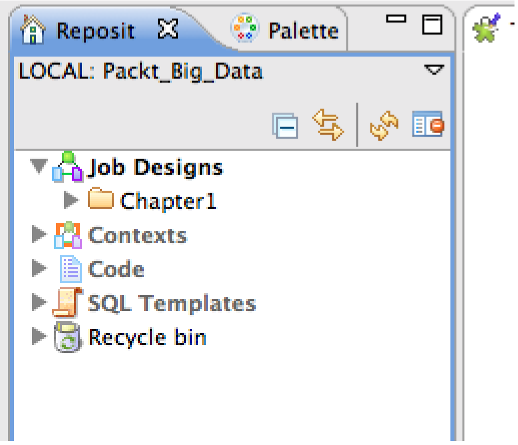


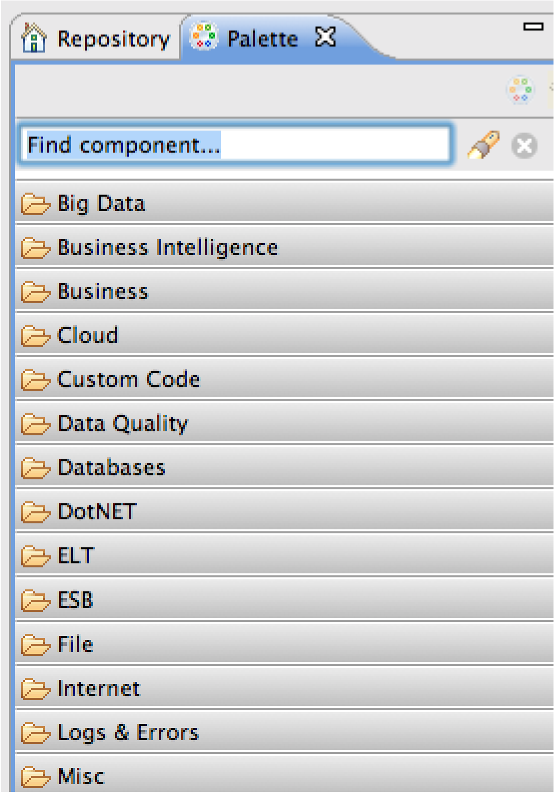
HDFS

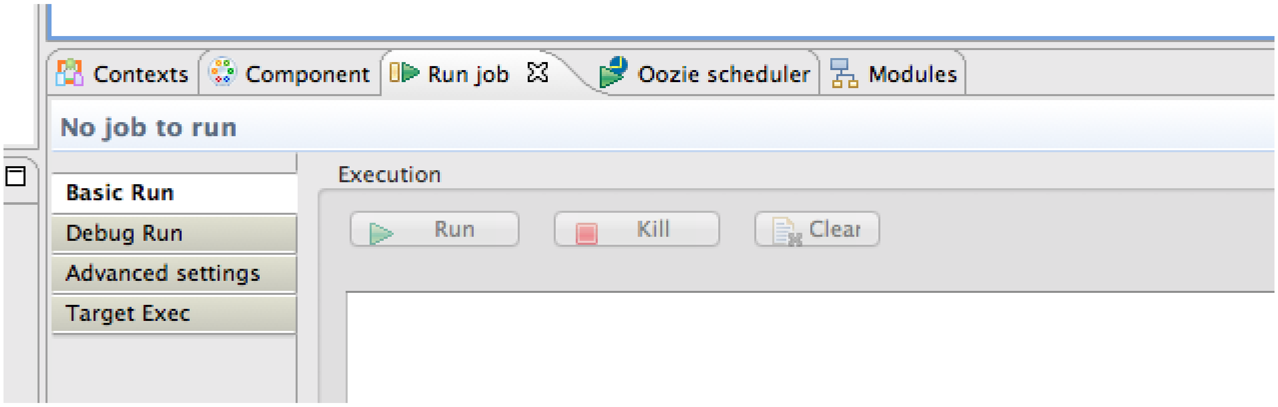
Writes data to HDFS

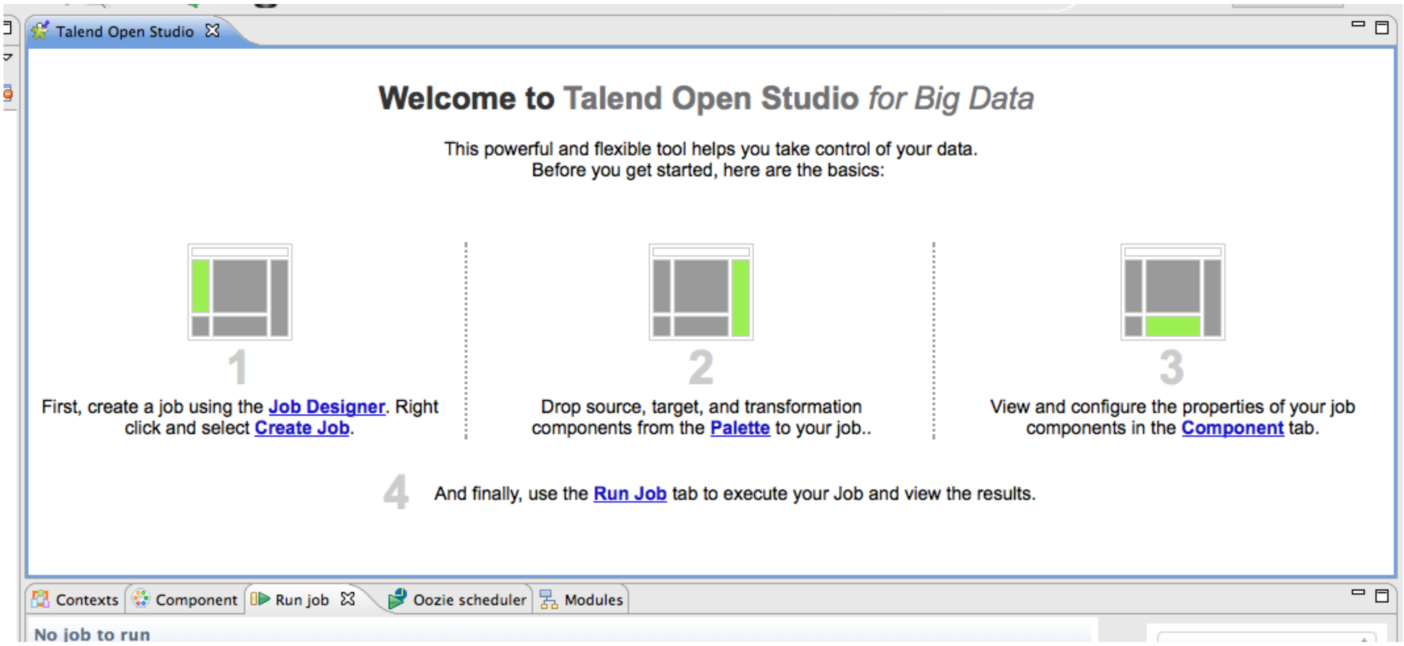


| | |
|---|---|
| Name | CH01_HDFS_WRITER |
| Purpose | Write in HDFS |
| Description | This job is part of the first Chapter of Talend Big Data book and aims to write new file in Hadoop Distributed File System |
| Author | test@talend.com |
| Locker | test@talend.com |
| Status | |
| Path | Chapter1 |


Reposit
Palette
LOCAL: Packt_Big_Data
Job Designs
Chapter1
Contexts
Code
SQL Templates
Recycle bin


Repository
Palette
Find component...
Big Data
Business Intelligence
Business
Cloud
Custom Code
Data Quality
Databases
DotNET
ELT
ESB
File
Internet
Logs & Errors
Misc





Welcome to Talend Open Studio *for Big Data*

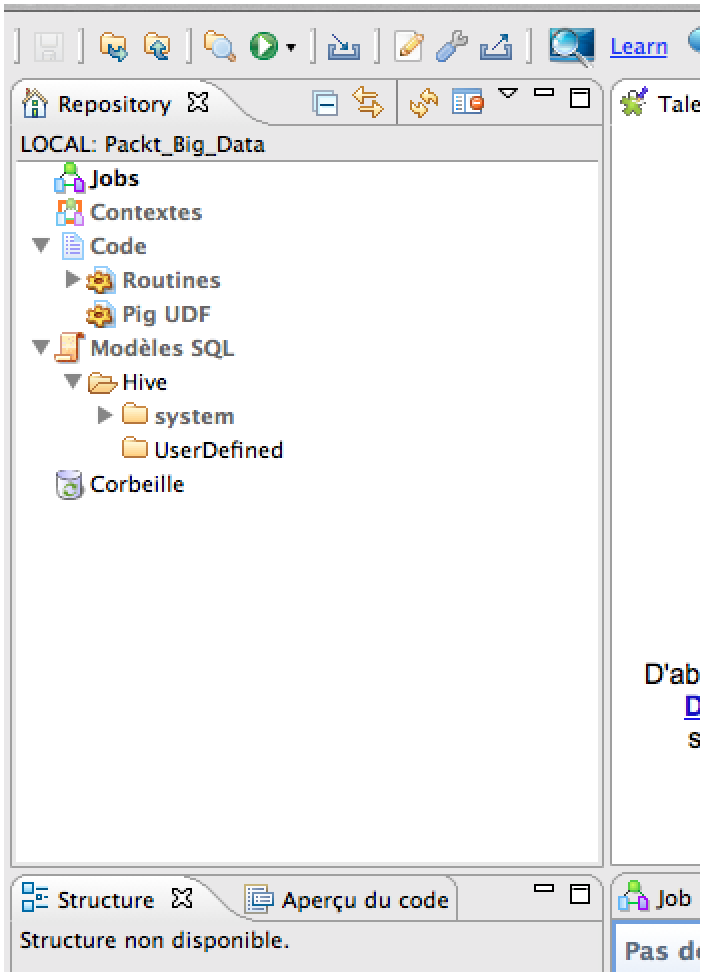
This powerful and flexible tool helps you take control of your data.
Before you get started, here are the basics:

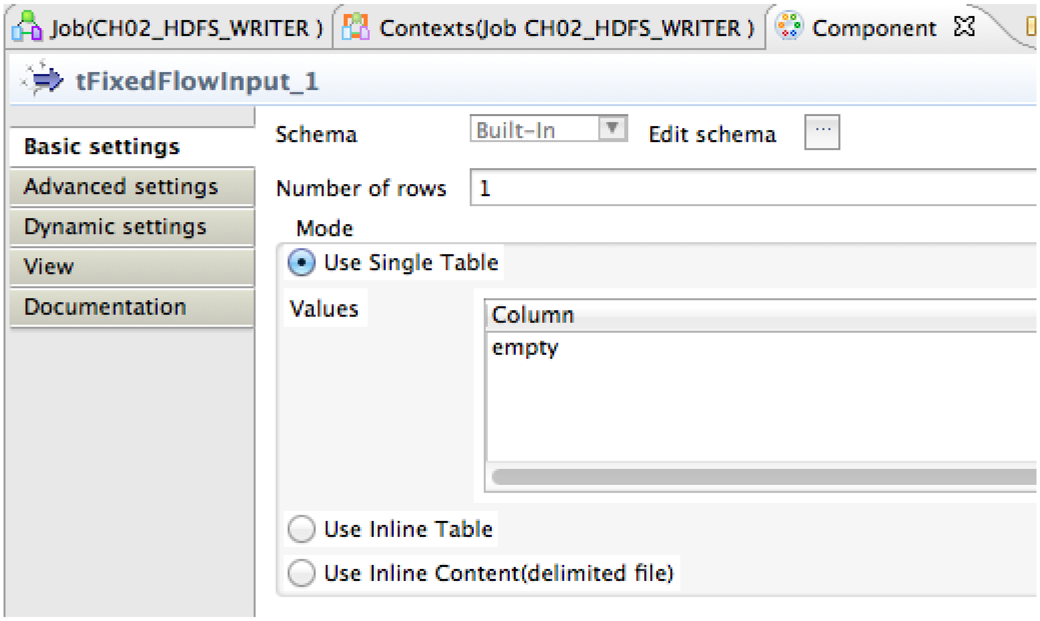
First, create a job using the Job Designer. Right click and select Create Job.

Drop source, target, and transformation components from the Palette to your job..

View and configure the properties of your job components in the Component tab.

And finally, use the Run Job tab to execute your Job and view the results.





Schema: Built-In

Number of rows: 1

Mode

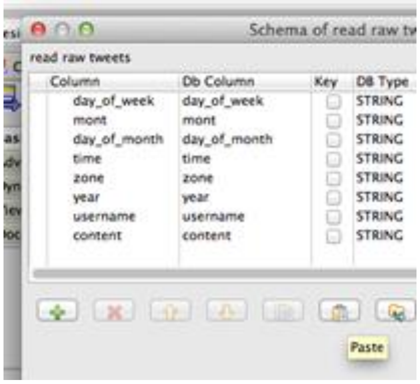
● Use Single Table

| Column |
|---|
| empty |

○ Use Inline Table

○ Use Inline Content(delimited file)

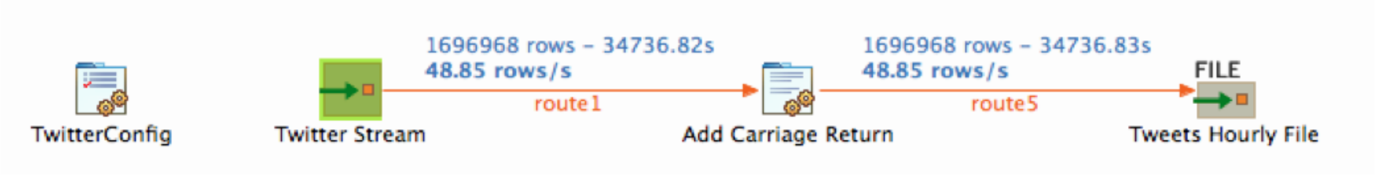
## 3. Formatting Data

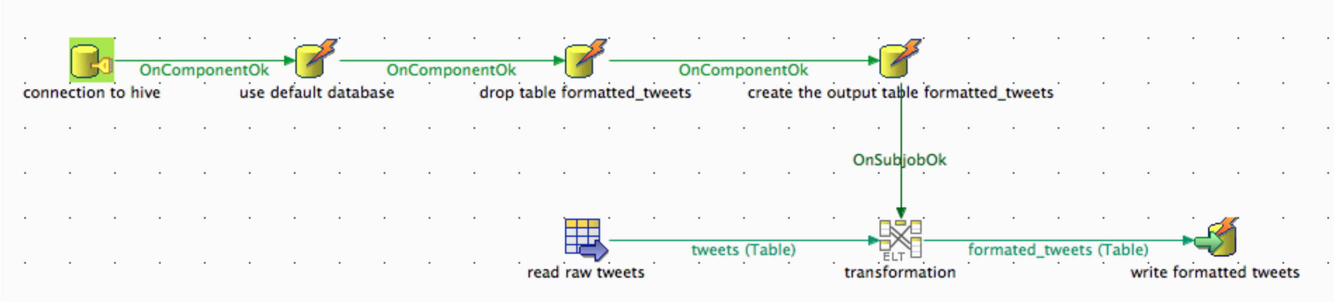


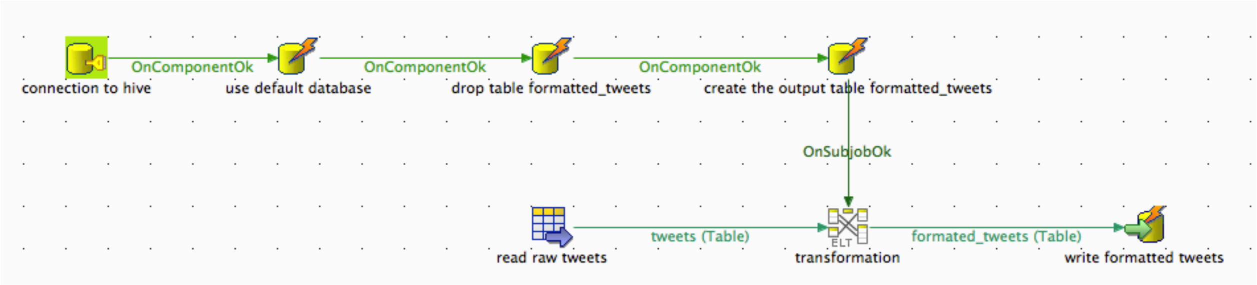
Schema of read raw tw

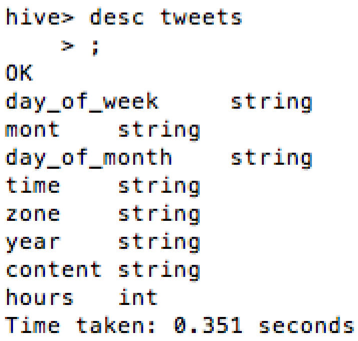
read raw tweets

| Column | Db Column | Key | DB Type |
|---|---|---|---|
| day_of_week | day_of_week | □ | STRING |
| mont | mont | □ | STRING |
| day_of_month | day_of_month | □ | STRING |
| time | time | □ | STRING |
| zone | zone | □ | STRING |
| year | year | □ | STRING |
| username | username | □ | STRING |
| content | content | □ | STRING |

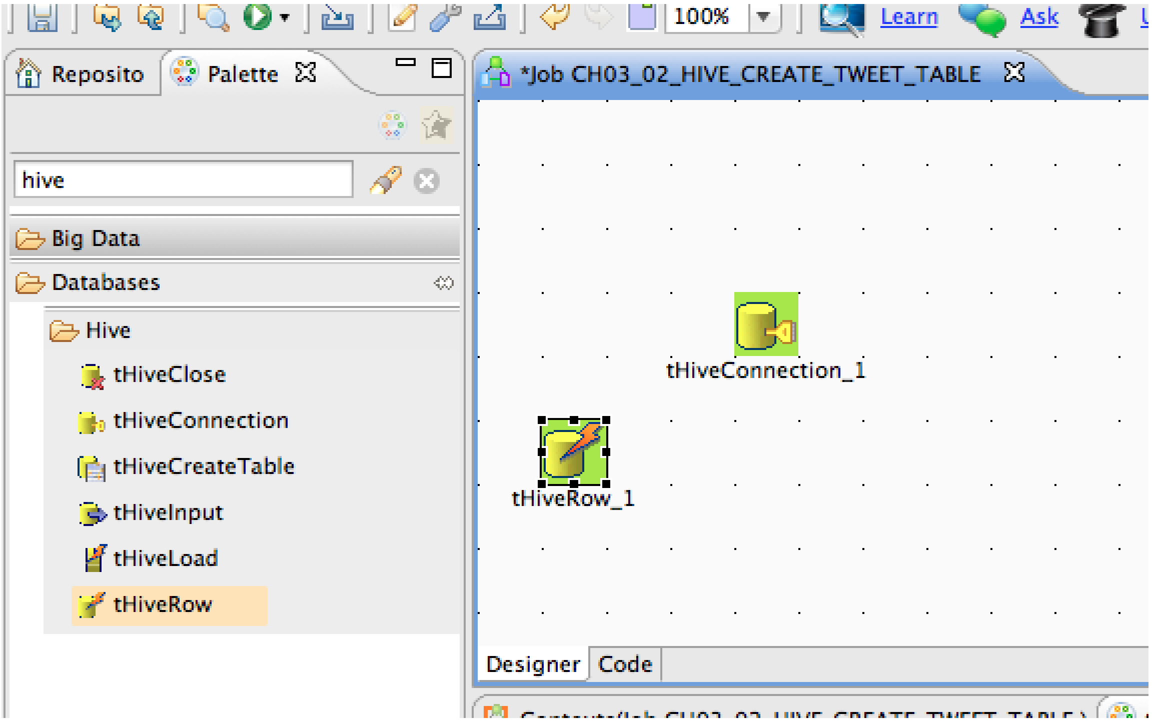


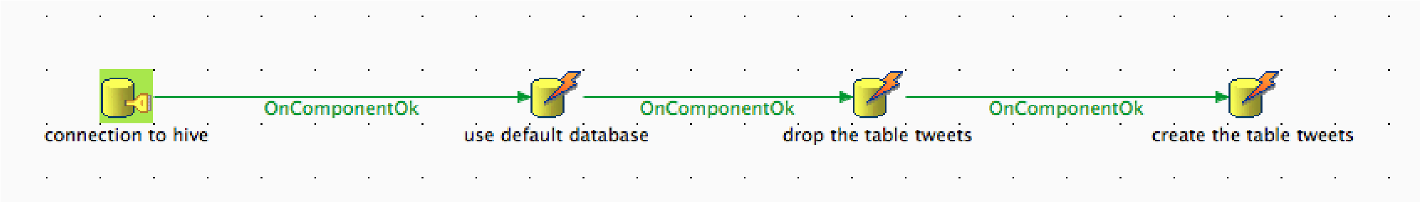


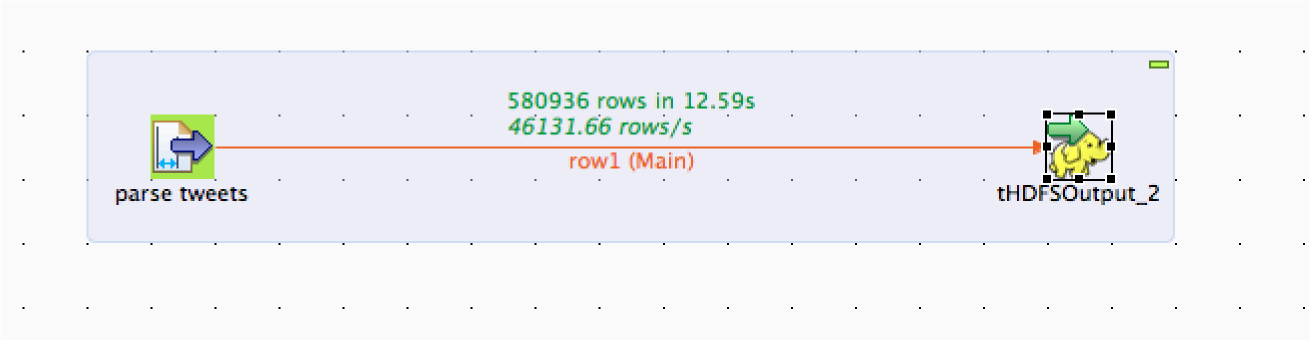


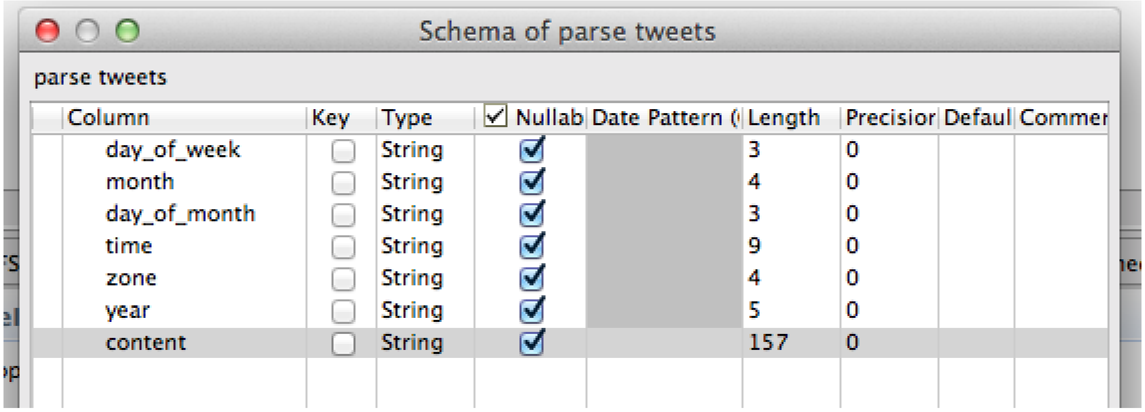


```
hive> desc tweets
    > ;
OK
day_of_week     string
mont    string
day_of_month    string
time    string
zone    string
year    string
content string
hours   int
Time taken: 0.351 seconds
```





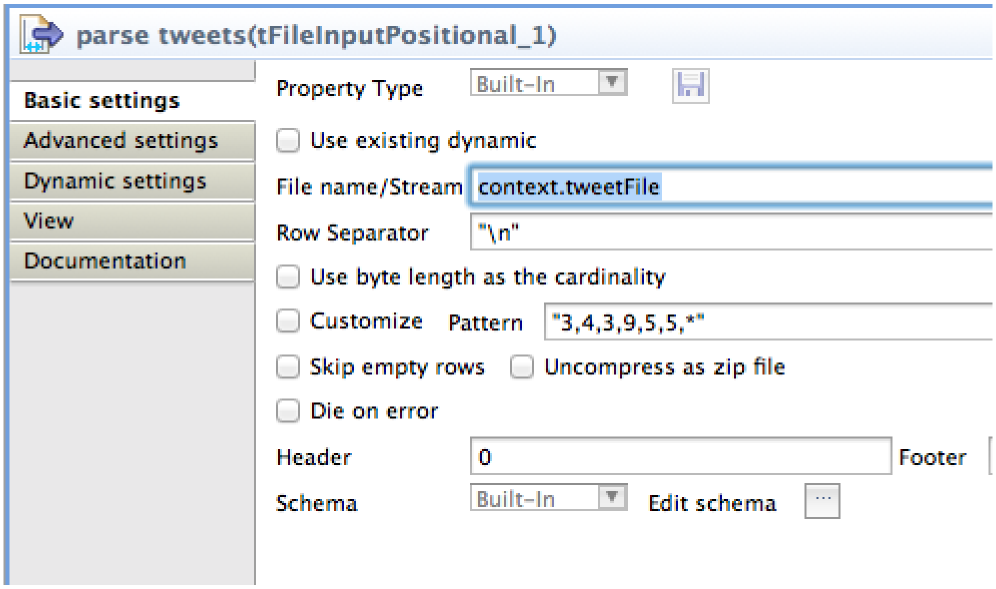




Schema of parse tweets

parse tweets

| Column | Key | Type | ☑ Nullab | Date Pattern ( | Length | Precisior | Defaul | Commer |
|---|---|---|---|---|---|---|---|---|
| day_of_week | □ | String | ☑ | | 3 | 0 | | |
| month | □ | String | ☑ | | 4 | 0 | | |
| day_of_month | □ | String | ☑ | | 3 | 0 | | |
| time | □ | String | ☑ | | 9 | 0 | | |
| zone | □ | String | ☑ | | 4 | 0 | | |
| year | □ | String | ☑ | | 5 | 0 | | |
| content | □ | String | ☑ | | 157 | 0 | | |



parse tweets(tFileInputPositional_1)

Property Type: Built-In

□ Use existing dynamic

File name/Stream: context.tweetFile

Row Separator: "\n"

□ Use byte length as the cardinality

□ Customize Pattern: "3,4,3,9,5,5,*"

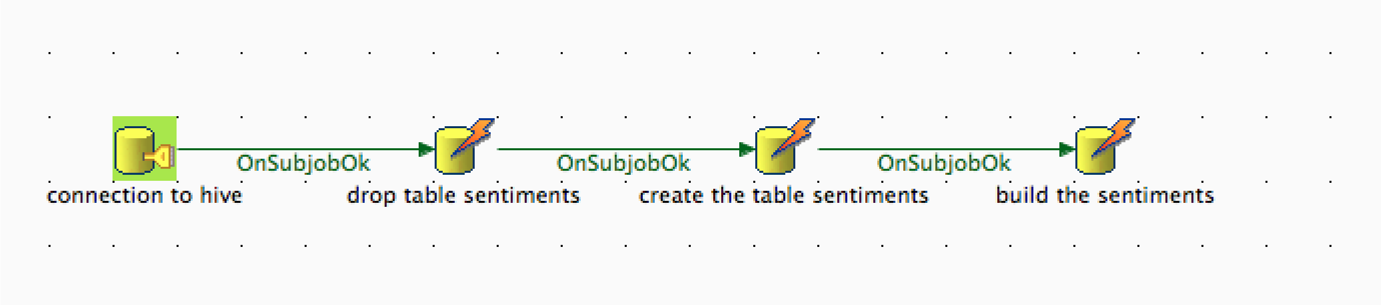
□ Skip empty rows □ Uncompress as zip file

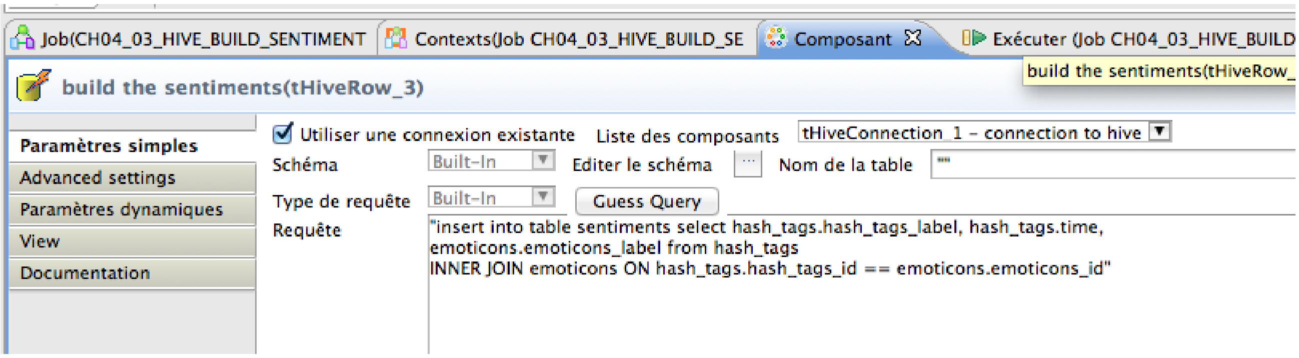
□ Die on error

Header: 0 Footer

Schema: Built-In Edit schema ...

## 4. Processing Tweets with Apache Hive





Job(CH04_03_HIVE_BUILD_SENTIMENT
Contexts(Job CH04_03_HIVE_BUILD_SE
Composant
Exécuter (Job CH04_03_HIVE_BUILD
build the sentiments(tHiveRow_
Paramètres simples
Advanced settings
Paramètres dynamiques
View
Documentation
Guess Query

build the sentiments(tHiveRow_3)

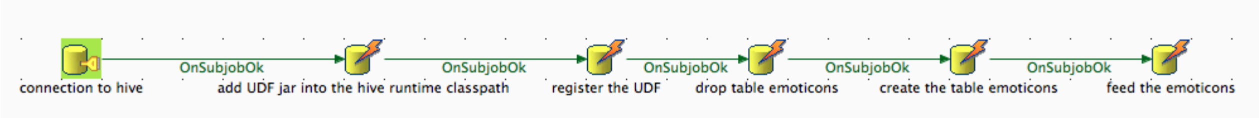
☑ Utiliser une connexion existante — Liste des composants: tHiveConnection_1 - connection to hive

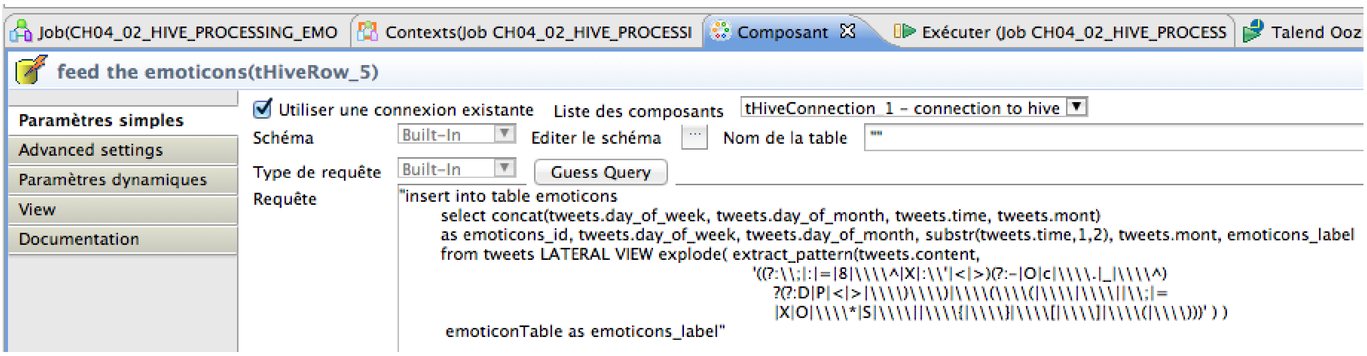
Schéma: Built-In — Editer le schéma — Nom de la table: ""

Type de requête: Built-In

Requête:

```
"insert into table sentiments select hash_tags.hash_tags_label, hash_tags.time,
emoticons.emoticons_label from hash_tags
INNER JOIN emoticons ON hash_tags.hash_tags_id == emoticons.emoticons_id"
```





feed the emoticons(tHiveRow_5)

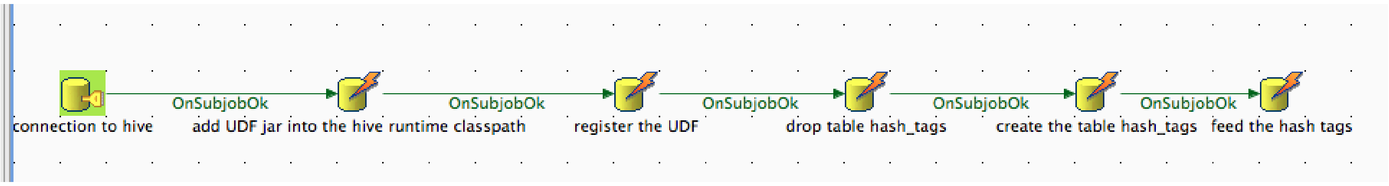
☑ Utiliser une connexion existante — Liste des composants: tHiveConnection_1 - connection to hive

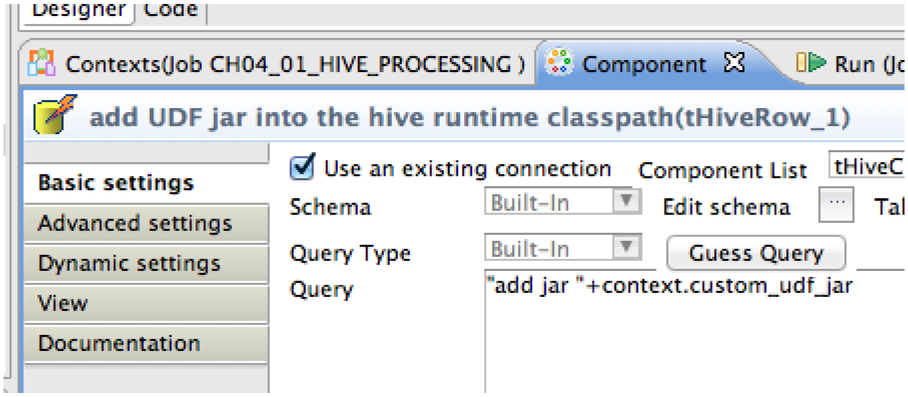
Schéma: Built-In — Editer le schéma — Nom de la table: ""

Type de requête: Built-In

Requête:

```
"insert into table emoticons
        select concat(tweets.day_of_week, tweets.day_of_month, tweets.time, tweets.mont)
        as emoticons_id, tweets.day_of_week, tweets.day_of_month, substr(tweets.time,1,2), tweets.mont, emoticons_label
        from tweets LATERAL VIEW explode( extract_pattern(tweets.content,
                                             '((?:\\;|:|=|8|\\\\^|X|:\\'|<|>)(?:-|O|c|\\\\.|_|\\\\^)
                                                ?(?:D|P|<|>|\\\\)\\\\)|\\\\(\\\\(|\\\\|\\\\||\\;|=
                                                |X|O|\\\\*|S|\\\\||\\\\{|\\\\}|\\\\[|\\\\]|\\\\(|\\\\)))' ) )
         emoticonTable as emoticons_label"
```





add UDF jar into the hive runtime classpath(tHiveRow_1)

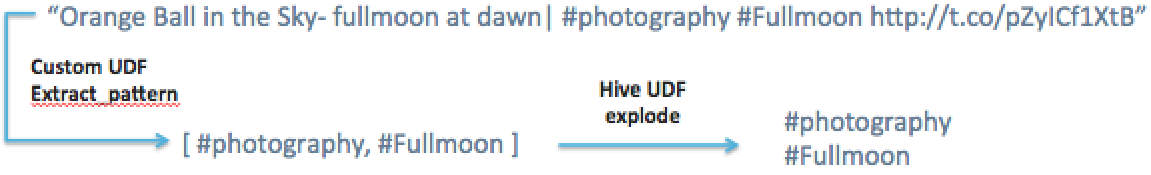
☑ Use an existing connection — Component List: tHiveC

Schema: Built-In — Edit schema

Query Type: Built-In

Query:

```
"add jar "+context.custom_udf_jar
```

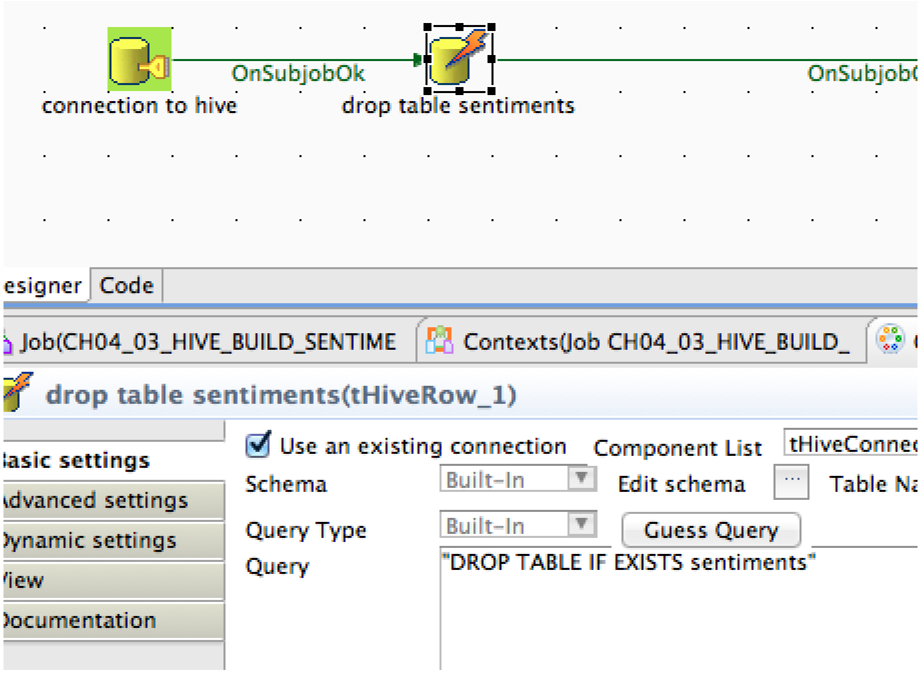

“Orange Ball in the Sky- fullmoon at dawn| #photography #Fullmoon http://t.co/pZyICf1XtB”

Custom UDF
Extract_pattern

[ #photography, #Fullmoon ]

Hive UDF
explode

#photography
#Fullmoon



drop table sentiments(tHiveRow_1)

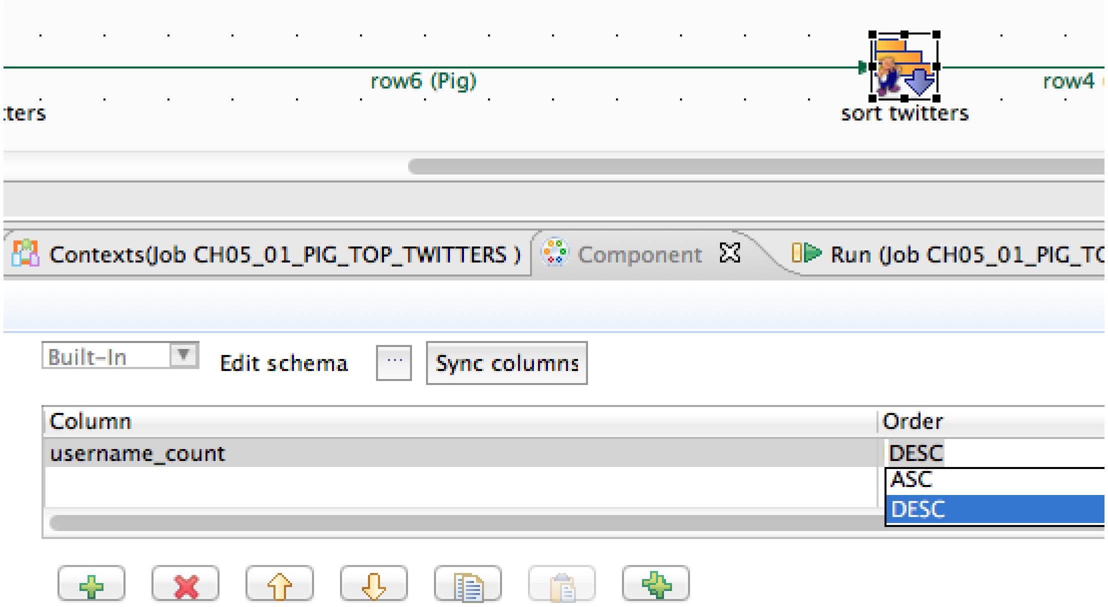
☑ Use an existing connection Component List tHiveConnec

Schema Built-In

Query Type Built-In

Query "DROP TABLE IF EXISTS sentiments"

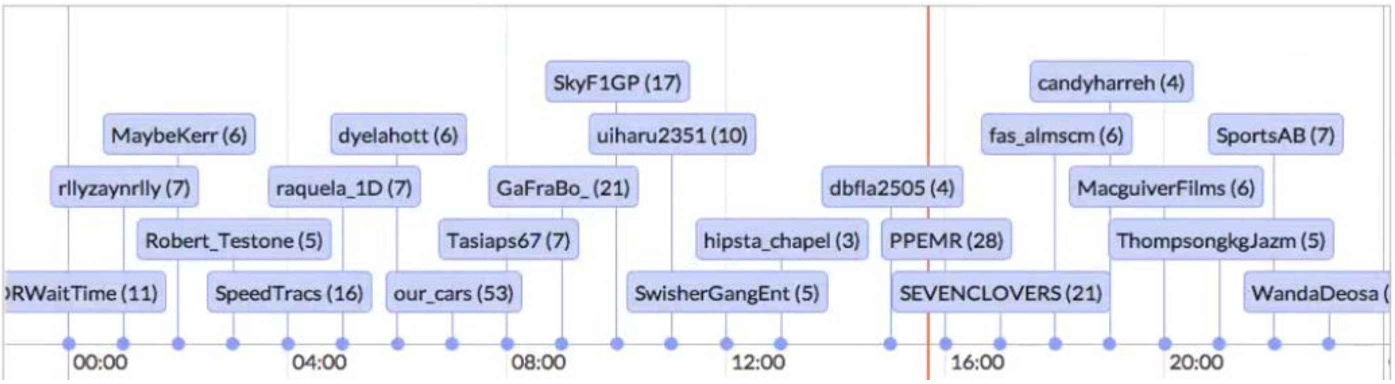
## 5. Aggregating Data with Apache Pig

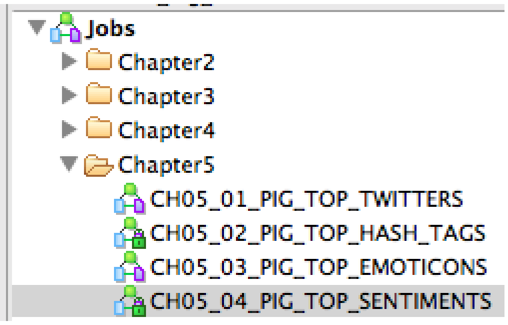


Built-In

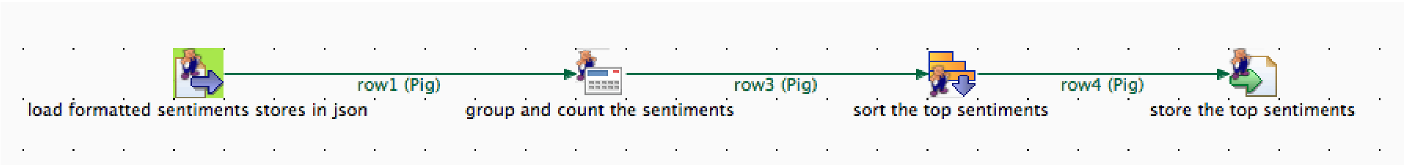
| Column | Order |
|---|---|
| username_count | DESC |

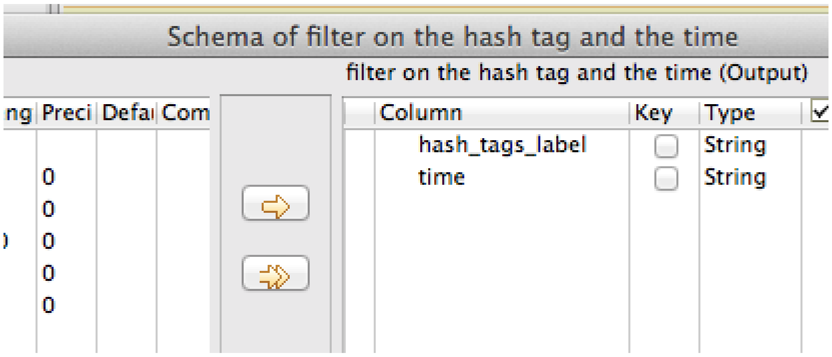
Top Twitters Timeline





- Jobs
  - Chapter2
  - Chapter3
  - Chapter4
  - Chapter5
    - CH05_01_PIG_TOP_TWITTERS
    - CH05_02_PIG_TOP_HASH_TAGS
    - CH05_03_PIG_TOP_EMOTICONS
    - CH05_04_PIG_TOP_SENTIMENTS



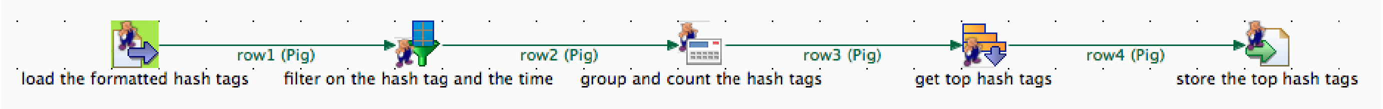


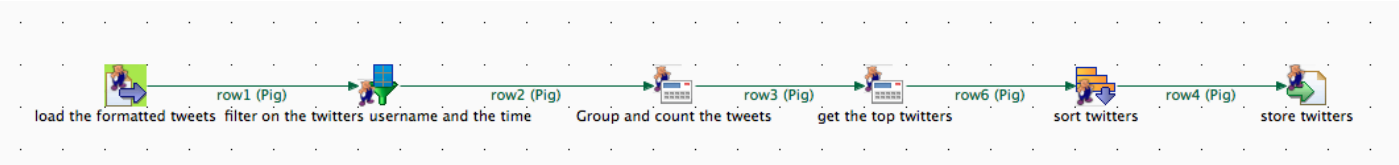
Schema of filter on the hash tag and the time

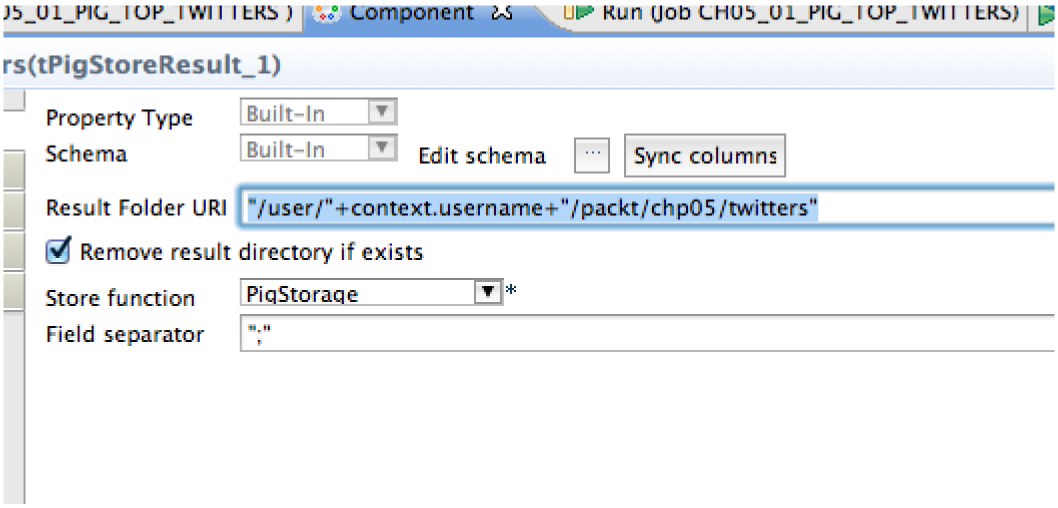
filter on the hash tag and the time (Output)

| ng | Preci | Defa | Com |
|---|---|---|---|
| | | | |
| | 0 | | |
| | 0 | | |
| ) | 0 | | |
| | 0 | | |
| | 0 | | |

| Column | Key | Type | ☑ |
|---|---|---|---|
| hash_tags_label | □ | String | |
| time | □ | String | |


load the formatted hash tags
row1 (Pig)
filter on the hash tag and the time
row2 (Pig)
group and count the hash tags
row3 (Pig)
get top hash tags
row4 (Pig)
store the top hash tags





rs(tPigStoreResult_1)

Property Type: Built-In

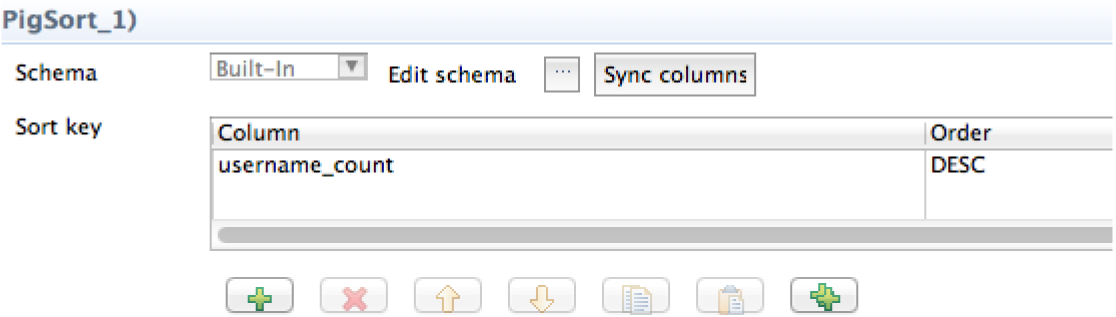
Schema: Built-In

Result Folder URI: "/user/"+context.username+"/packt/chp05/twitters"

☑ Remove result directory if exists

Store function: PigStorage *

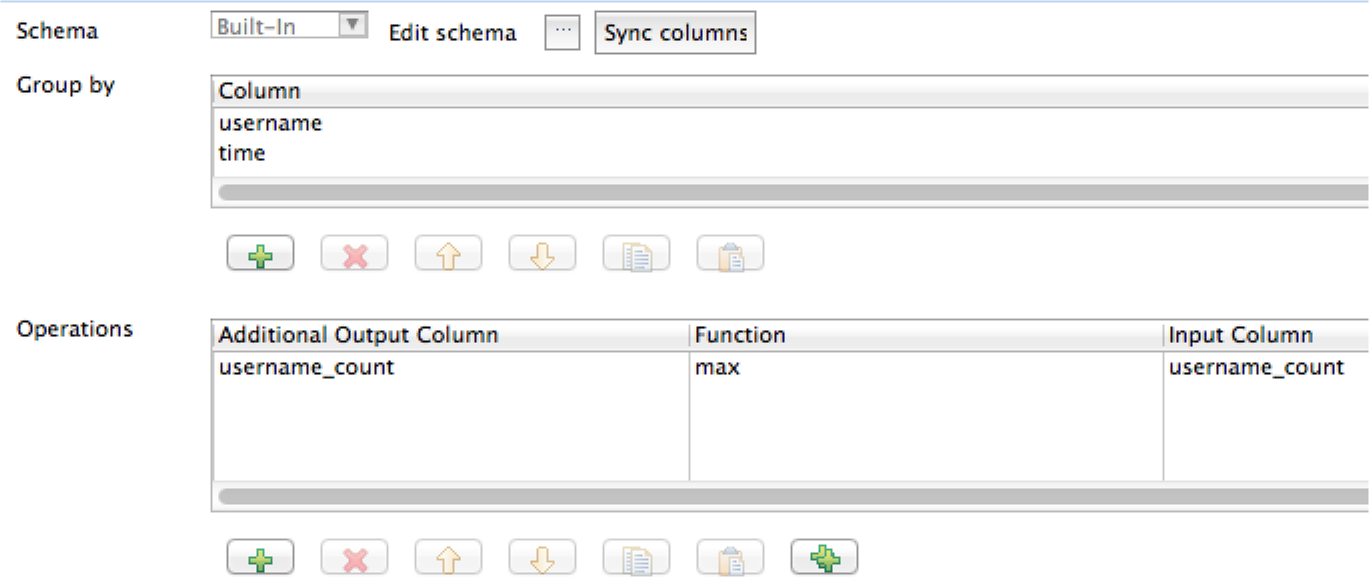
Field separator: ";"


PigSort_1)

Schema: Built-In

Sort key

| Column | Order |
|---|---|
| username_count | DESC |



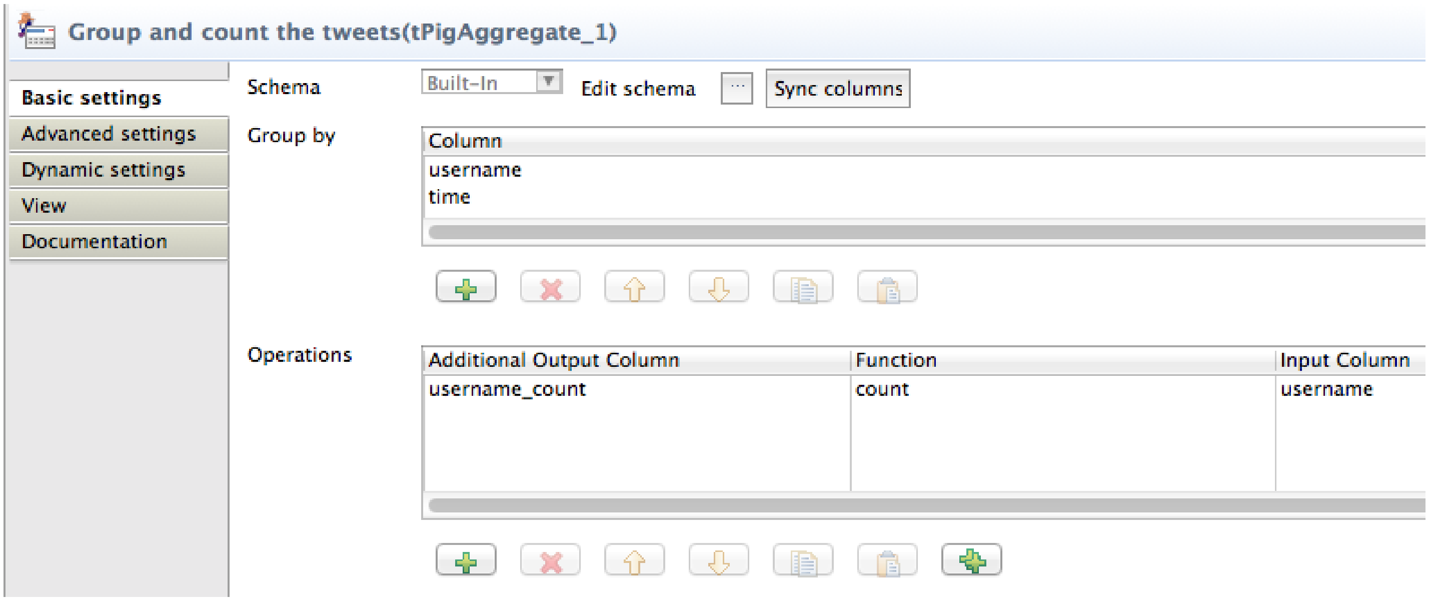
Schema: Built-In

Group by

| Column |
|---|
| username |
| time |

Operations

| Additional Output Column | Function | Input Column |
|---|---|---|
| username_count | max | username_count |



Group and count the tweets(tPigAggregate_1)

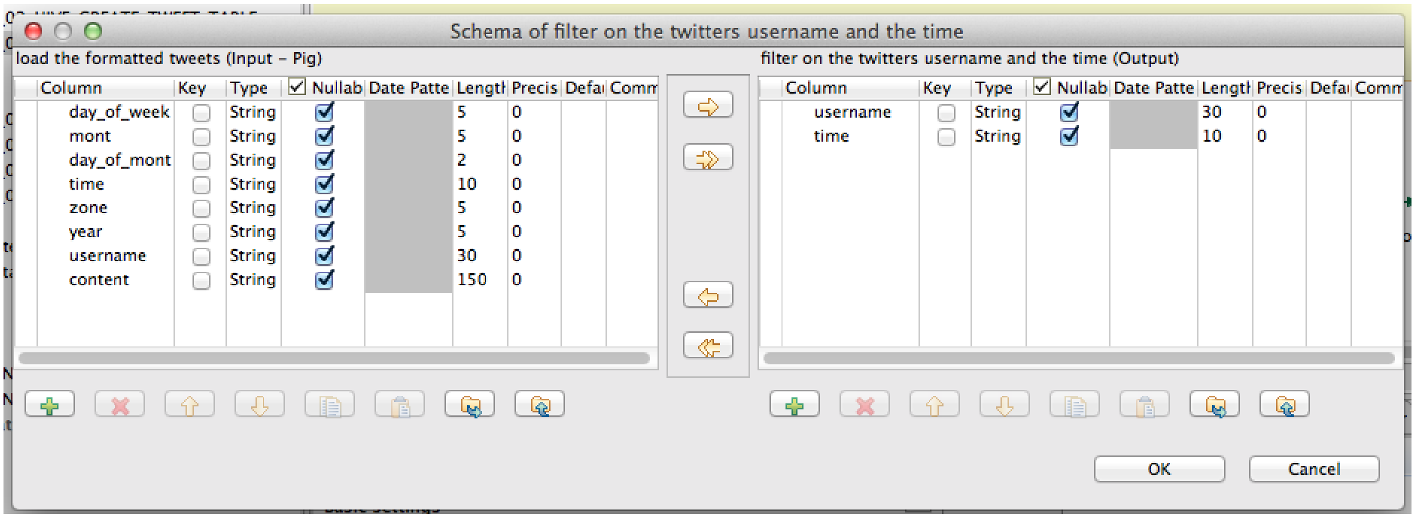
Schema: Built-In

Group by

| Column |
|---|
| username |
| time |

Operations

| Additional Output Column | Function | Input Column |
|---|---|---|
| username_count | count | username |



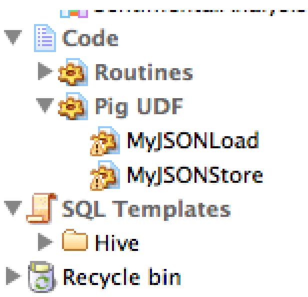
Schema of filter on the twitters username and the time

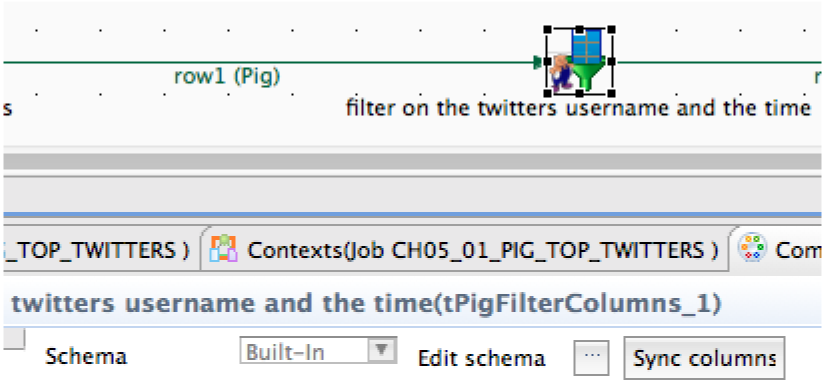
load the formatted tweets (Input – Pig)

| Column | Key | Type | ☑ Nullab | Date Patte | Lengtl | Precis | Defa | Comm |
|---|---|---|---|---|---|---|---|---|
| day_of_week | ☐ | String | ☑ | | 5 | 0 | | |
| mont | ☐ | String | ☑ | | 5 | 0 | | |
| day_of_mont | ☐ | String | ☑ | | 2 | 0 | | |
| time | ☐ | String | ☑ | | 10 | 0 | | |
| zone | ☐ | String | ☑ | | 5 | 0 | | |
| year | ☐ | String | ☑ | | 5 | 0 | | |
| username | ☐ | String | ☑ | | 30 | 0 | | |
| content | ☐ | String | ☑ | | 150 | 0 | | |

filter on the twitters username and the time (Output)

| Column | Key | Type | ☑ Nullab | Date Patte | Lengtl | Precis | Defa | Comm |
|---|---|---|---|---|---|---|---|---|
| username | ☐ | String | ☑ | | 30 | 0 | | |
| time | ☐ | String | ☑ | | 10 | 0 | | |

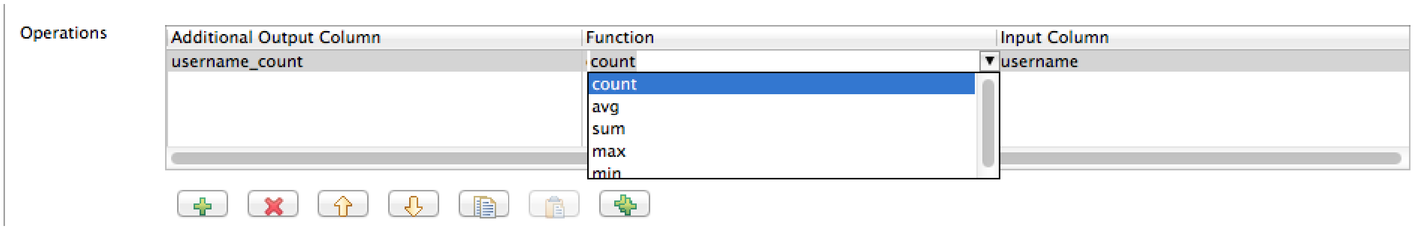




row1 (Pig)
filter on the twitters username and the time
_TOP_TWITTERS )
Contexts(Job CH05_01_PIG_TOP_TWITTERS )
Com

twitters username and the time(tPigFilterColumns_1)

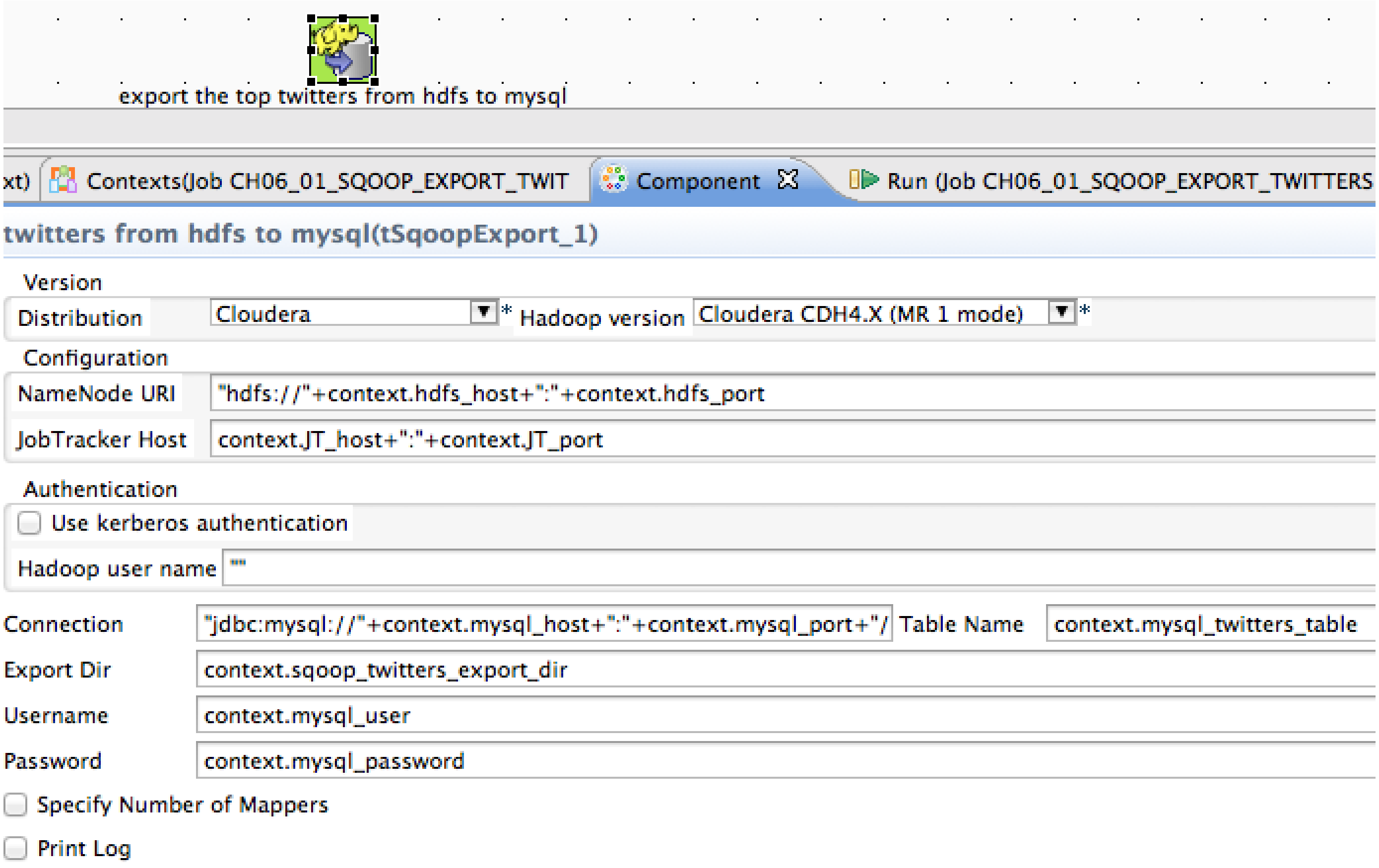
Schema Built-In Edit schema ... Sync columns



Operations

| Additional Output Column | Function | Input Column |
|---|---|---|
| username_count | count | username |

## 6. Using the SQL Database



xt)
Contexts(Job CH06_01_SQOOP_EXPORT_TWIT
Component
Run (Job CH06_01_SQOOP_EXPORT_TWITTERS

export the top twitters from hdfs to mysql

twitters from hdfs to mysql(tSqoopExport_1)

Version

Distribution: Cloudera * Hadoop version: Cloudera CDH4.X (MR 1 mode) *

Configuration

NameNode URI: "hdfs://"+context.hdfs_host+":"+context.hdfs_port

JobTracker Host: context.JT_host+":"+context.JT_port

Authentication

☐ Use kerberos authentication

Hadoop user name: ""

Connection: "jdbc:mysql://"+context.mysql_host+":"+context.mysql_port+"/ Table Name: context.mysql_twitters_table

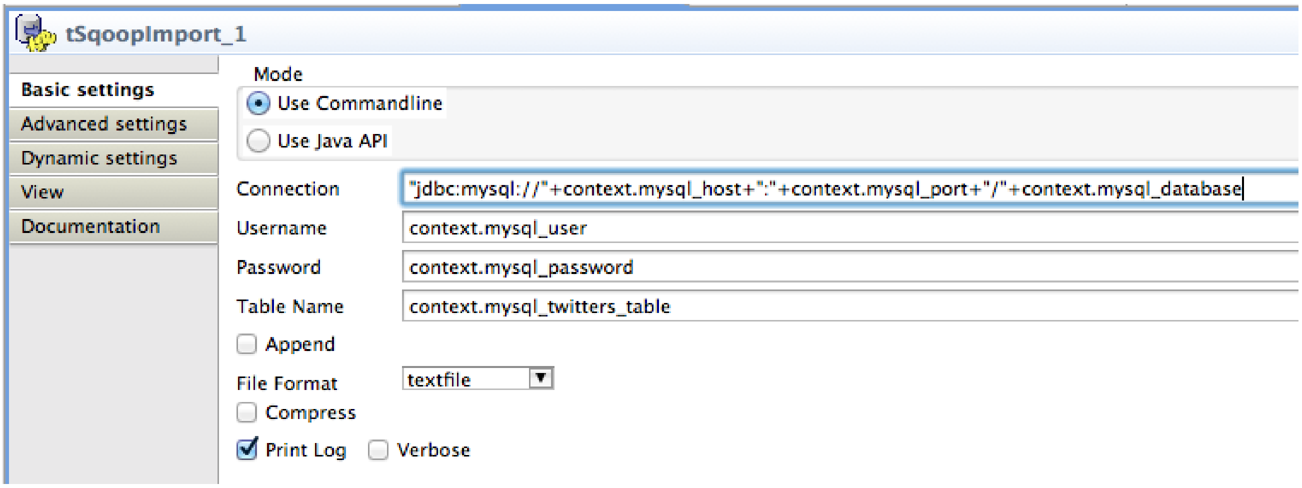
Export Dir: context.sqoop_twitters_export_dir

Username: context.mysql_user

Password: context.mysql_password

☐ Specify Number of Mappers

☐ Print Log



tSqoopImport_1

Mode

● Use Commandline

○ Use Java API

Connection: "jdbc:mysql://"+context.mysql_host+":"+context.mysql_port+"/"+context.mysql_database

Username: context.mysql_user

Password: context.mysql_password

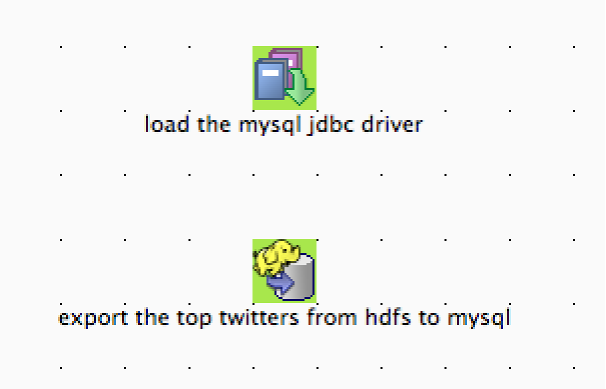
Table Name: context.mysql_twitters_table

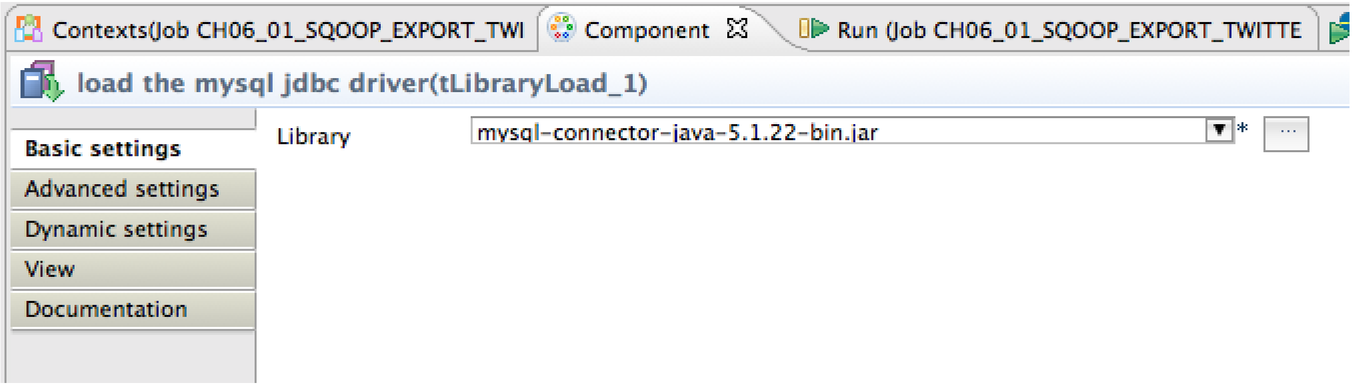
☐ Append

File Format: textfile

☐ Compress

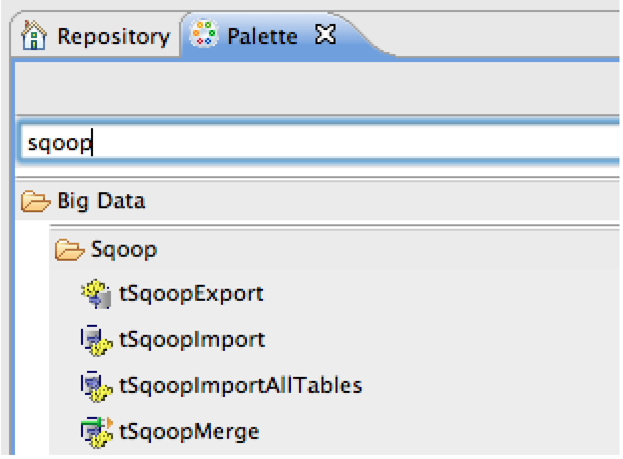
☑ Print Log ☐ Verbose


load the mysql jdbc driver
export the top twitters from hdfs to mysql


Contexts(Job CH06_01_SQOOP_EXPORT_TWI
Component
Run (Job CH06_01_SQOOP_EXPORT_TWITTE
Basic settings
Advanced settings
Dynamic settings
View
Documentation

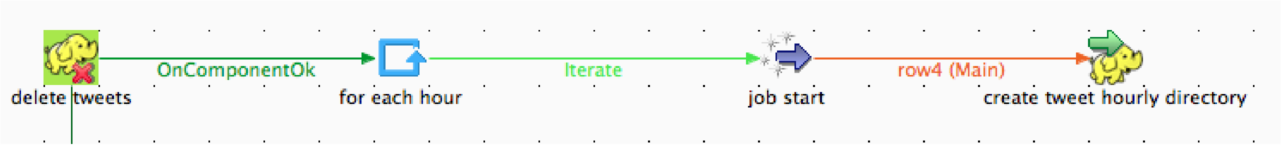
load the mysql jdbc driver(tLibraryLoad_1)

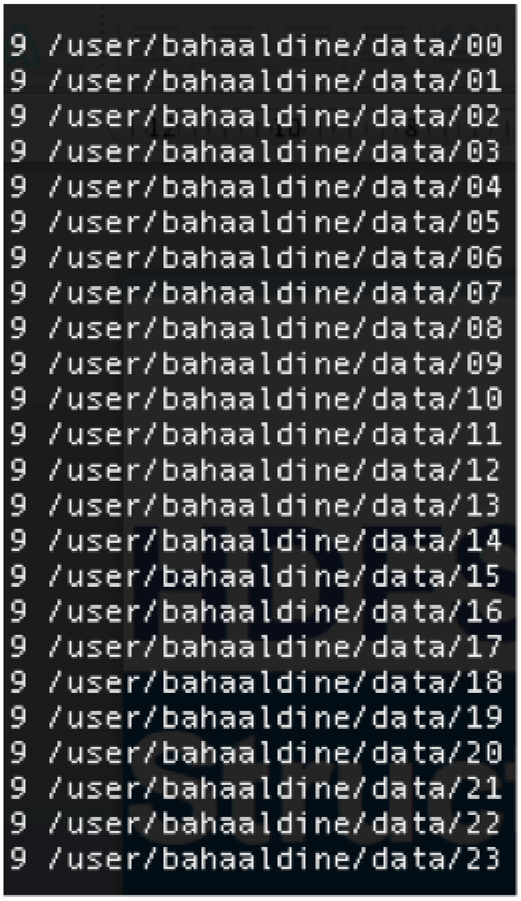
Library mysql-connector-java-5.1.22-bin.jar



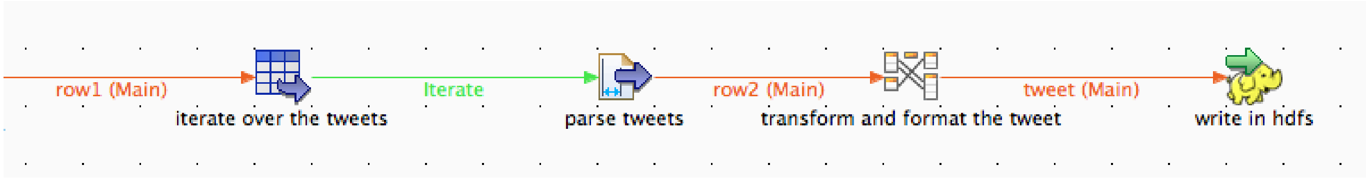
sqoop

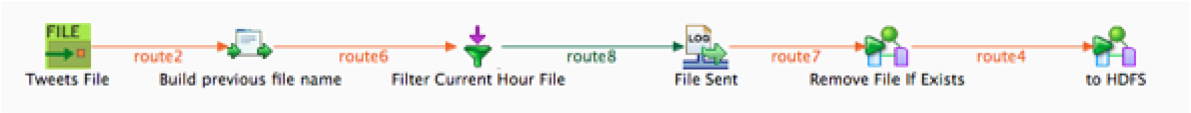
## 7. Big Data Architecture and Integration Patterns

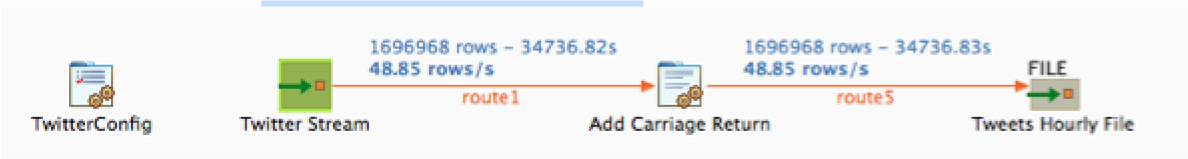


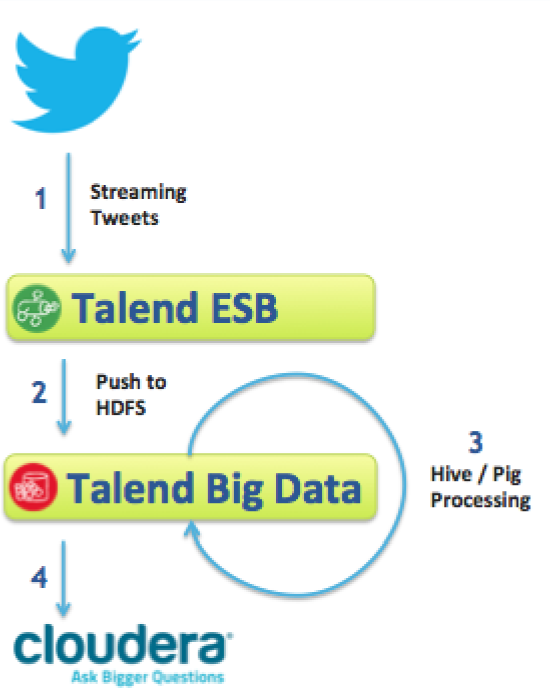


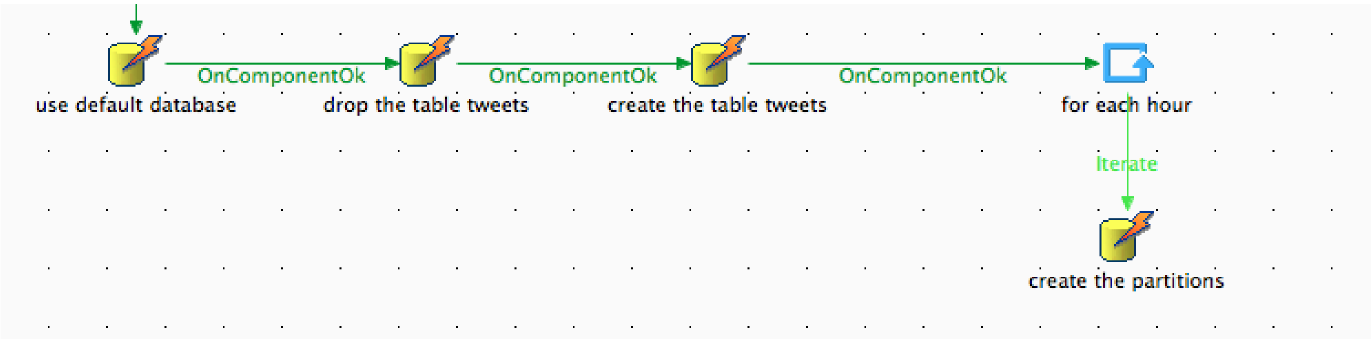
```
9 /user/bahaaldine/data/00
9 /user/bahaaldine/data/01
9 /user/bahaaldine/data/02
9 /user/bahaaldine/data/03
9 /user/bahaaldine/data/04
9 /user/bahaaldine/data/05
9 /user/bahaaldine/data/06
9 /user/bahaaldine/data/07
9 /user/bahaaldine/data/08
9 /user/bahaaldine/data/09
9 /user/bahaaldine/data/10
9 /user/bahaaldine/data/11
9 /user/bahaaldine/data/12
9 /user/bahaaldine/data/13
9 /user/bahaaldine/data/14
9 /user/bahaaldine/data/15
9 /user/bahaaldine/data/16
9 /user/bahaaldine/data/17
9 /user/bahaaldine/data/18
9 /user/bahaaldine/data/19
9 /user/bahaaldine/data/20
9 /user/bahaaldine/data/21
9 /user/bahaaldine/data/22
9 /user/bahaaldine/data/23
```



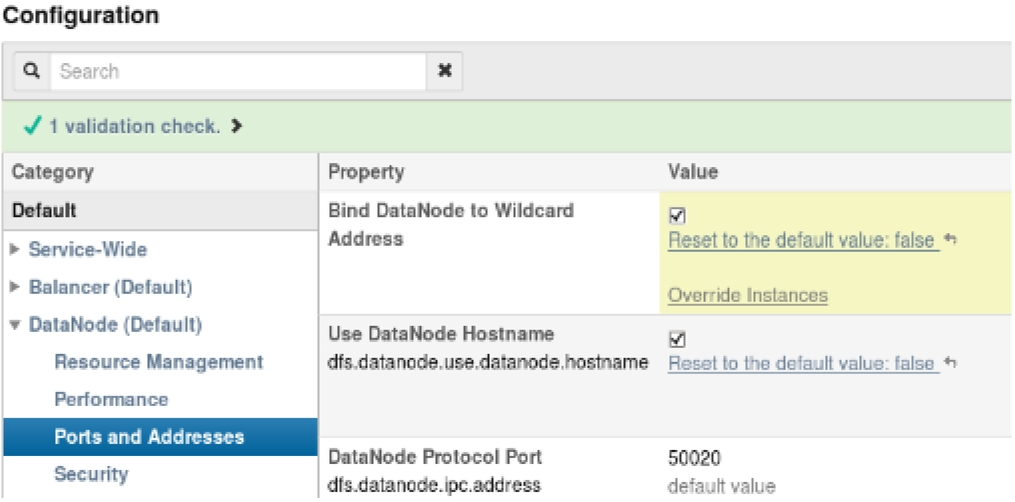



TwitterConfig
Twitter Stream
1696968 rows - 34736.82s
48.85 rows/s
route1
Add Carriage Return
1696968 rows - 34736.83s
48.85 rows/s
route5
FILE
Tweets Hourly File


1
Streaming Tweets
Talend ESB
2
Push to HDFS
Talend Big Data
3
Hive / Pig Processing
4
cloudera
Ask Bigger Questions

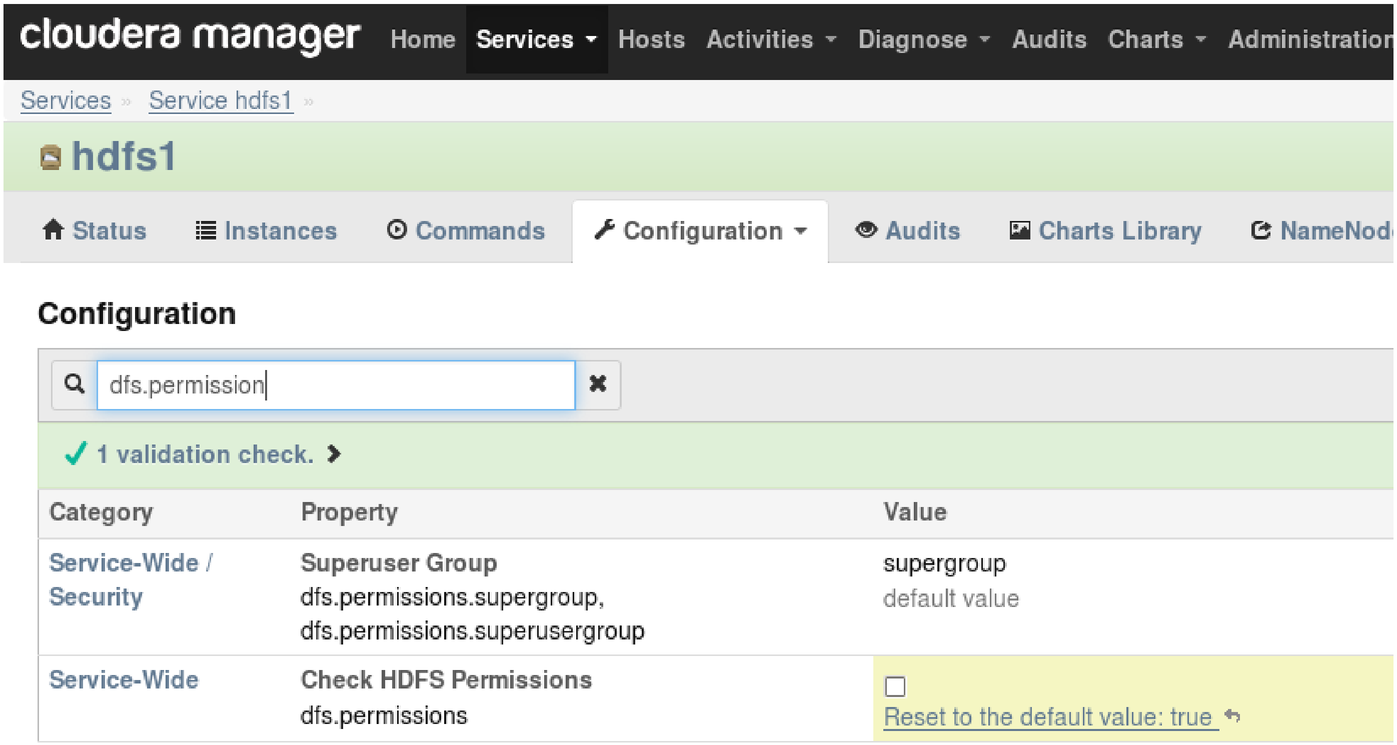

use default database
OnComponentOk
drop the table tweets
OnComponentOk
create the table tweets
OnComponentOk
for each hour
Iterate
create the partitions

# 8. Appendix-A



Configuration

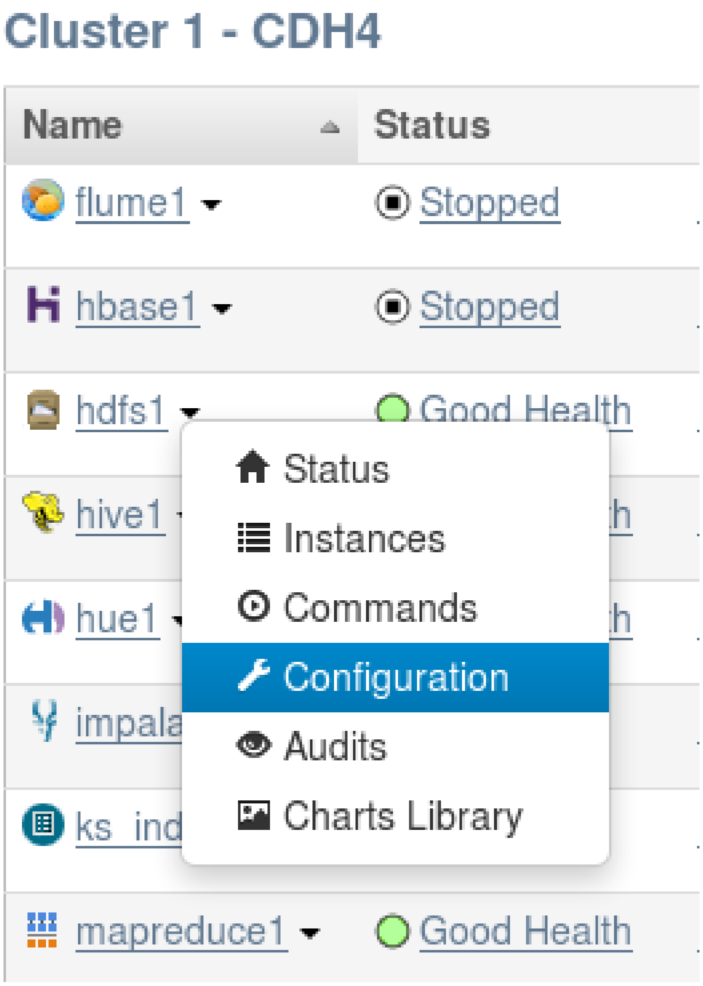
| Category | Property | Value |
|---|---|---|
| Default | Bind DataNode to Wildcard Address | ☑<br>Reset to the default value: false<br>Override Instances |
| ▸ Service-Wide<br>▸ Balancer (Default)<br>▾ DataNode (Default)<br>Resource Management<br>Performance<br>Ports and Addresses<br>Security | Use DataNode Hostname<br>dfs.datanode.use.datanode.hostname | ☑<br>Reset to the default value: false |
| | DataNode Protocol Port<br>dfs.datanode.ipc.address | 50020<br>default value |



Configuration

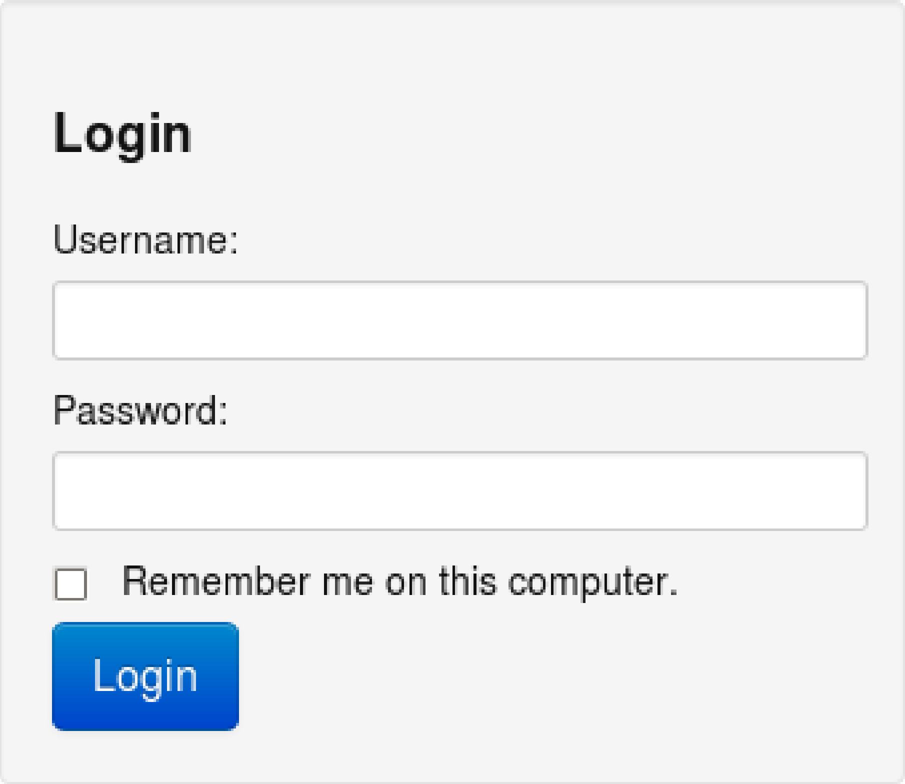
dfs.permission

| Category | Property | Value |
|---|---|---|
| Service-Wide / Security | Superuser Group<br>dfs.permissions.supergroup, dfs.permissions.superusergroup | supergroup<br>default value |
| Service-Wide | Check HDFS Permissions<br>dfs.permissions | ☐<br>Reset to the default value: true |



## Cluster 1 - CDH4

| Name | Status |
|---|---|
| flume1 | Stopped |
| hbase1 | Stopped |
| hdfs1 | Good Health |
| hive1 | h |
| hue1 | h |
| impala | |
| ks_ind | |
| mapreduce1 | Good Health |

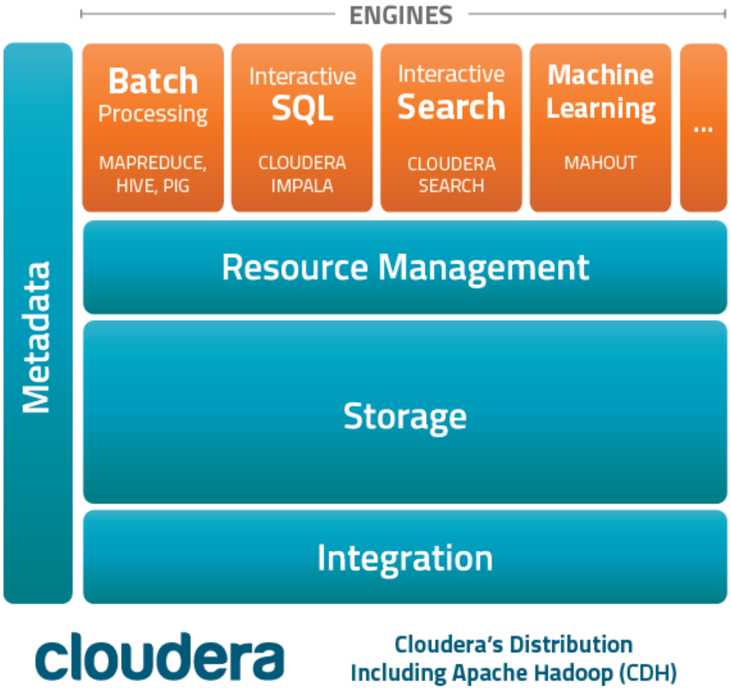


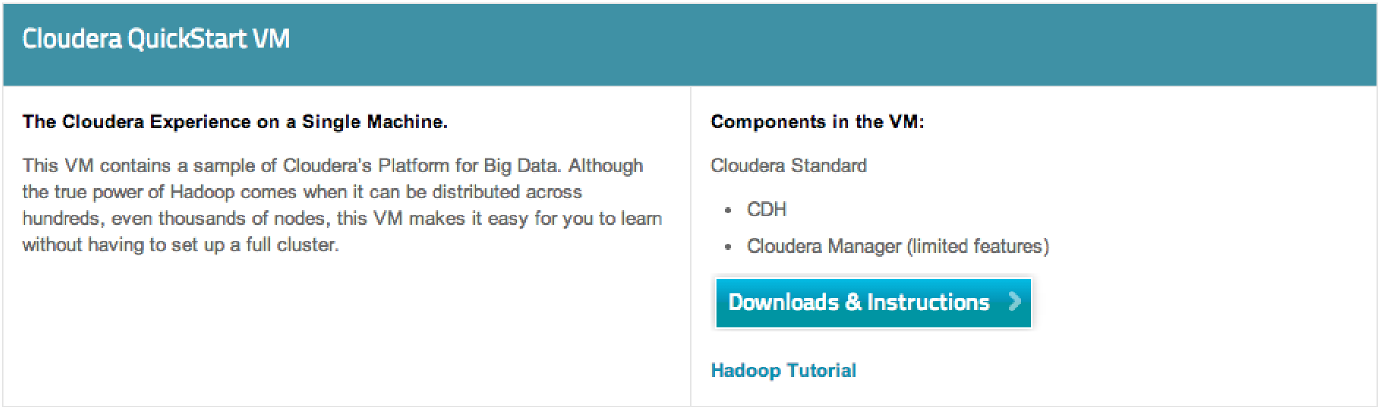
Login

Username:

Password:

□ Remember me on this computer.





## Cloudera QuickStart VM

**The Cloudera Experience on a Single Machine.**

This VM contains a sample of Cloudera's Platform for Big Data. Although the true power of Hadoop comes when it can be distributed across hundreds, even thousands of nodes, this VM makes it easy for you to learn without having to set up a full cluster.

**Components in the VM:**

Cloudera Standard

- CDH
- Cloudera Manager (limited features)

**Hadoop Tutorial**