**Session 10: Oozie and Sqoop**

Assignment 10.1

Student Name: Abarajithan SA

Course: Big Data Hadoop & Spark Training

Start Date:  2017-09-09

End Date:  2017-11-26

**Assignment 10.1**– Explain in Brief:

* The workflow of Oozie and its Benefits
* The workflow of Sqoop and its Benefits

Contents

[Introduction 2](#_Toc499634269)

[Problem Statement 2](#_Toc499634270)

[1. The workflow of Oozie and its Benefits 2](#_Toc499634271)

[1.1 Oozie Workflow Nodes: 2](#_Toc499634272)

[1.2 Oozie Workflow Diagram 3](#_Toc499634273)

[1.3 Sample Workflow 4](#_Toc499634274)

[1.4 Some of the Benefits of using Oozie 4](#_Toc499634275)

[2 The workflow of Sqoop and its Benefits 5](#_Toc499634276)

[2.1 Sqoop Architecture 6](#_Toc499634277)

[2.2 Sample Sqoop Workflow 8](#_Toc499634278)

[2.3 Benefits of using Sqoop 9](#_Toc499634279)

# Introduction

In this assignment, we are going to see the concepts of Oozie and Sqoop.

# Problem Statement

Explain in Brief:

* The workflow of Oozie and its Benefits
* The workflow of Sqoop and its Benefits

# The workflow of Oozie and its Benefits

Apache Oozie is a server-based workflow scheduling system to manage Hadoop jobs. Workflows in Oozie are defined as a collection of control flow and action nodes in a directed acyclic graph (DAG).

It is a real time scheduler and workflow engine that blends well with large production environments.

#### Oozie Architecture Diagram:



### 1.1 Oozie Workflow Nodes:

Control Flow:

* Start/end/kill
* Decision
* Fork/join

Actions:

* Map-reduce
* PIG
* HDFS
* Sub-Workflow
* Java-run custom java code

There are 2 files needed to run Oozie workflow,

1. **Workflow.xml** – contains the structure of the workflow
2. **Job.properties** – contains the configuration properties

### 1.2 Oozie Workflow Diagram



### 1.3 Sample Workflow

<workflow-app name=”**Name of Workflow”** xmlns=”uri:oozie:workflow:0.1”>

<start to=”**Start Node”**/>

<action name=”**Name of Node**”>

<hive xmlns=”uri:oozie:hive-action:0.2”>

<job-tracker>**Job Tracker Address**</job-tracker>

<name-node>**Name Node address**</name-node>

<job-xml>**Path of hive-site.xml in HDFS**</job-xml>

<configuration>

<property>

<name>**Configuration Name**</name>

<value>**Configuration Value**</value>

</property>

</configuration>

<Script>**Hive Query File**</Script>

</hive>

<ok to=”end”>

<error to=”end”>

**</action>**

### 1.4 Some of the Benefits of using Oozie

We have already many workflow schedulers in the market, however Oozie does have some benefits, and they are,

* Oozie is designed to scale in a Hadoop cluster. Each job will be launched from a different data node. This means that the workflow load will be balanced and no single machine will become overburdened by launching workflows. This also means that the capacity to launch workflows will grow as the cluster grows.
* Oozie is well integrated with Hadoop security. This is especially important in a kerberized cluster. Oozie knows which user submitted the job and will launch all actions as that user, with the proper privileges. It will handle all the authentication details for the user as well.
* Oozie is the only workflow manager with built-in Hadoop actions, making workflow development, maintenance and troubleshooting easier.
* Oozie UI makes it easier to drill down to specific errors in the data nodes. Other systems would require significantly more work to correlate jobtracker jobs with the workflow actions.
* Can handle 1250 job submissions a minute.
* Oozie gets callbacks from MapReduce jobs so it knows when they finish and whether they hang without expensive polling. No other workflow manager can do this.
* Oozie Coordinator allows triggering actions when files arrive at HDFS. This will be challenging to implement anywhere else.

# The workflow of Sqoop and its Benefits

Sqoop is a command-line interface application for transferring data between relational databases and Hadoop, Sqoop allows easy import and export of data from structured data stores such as relational databases, enterprise data warehouses and NoSql systems.

Using Sqoop, we can provision the data from external system on to HDFS and populate tables in HIVE and HBase.

Sqoop integrates with Oozie, allowing you to schedule and automate import and export tasks.

Sqoop uses a connector based architecture which supports plugins that provide connectivity to new external systems.

The following **command** is used to import all data from table called ORDERS from a MySQL DB,

***Sqoop import --connect jdbc:mysql://localhost/acmedb --table ORDERS --username test --password\*\*\*\****

**The import is done in 2 steps:**

* In the first step, sqoop introspects the database to gather the necessary metadata for the data being imported.
* The second step is a map-only Hadoop jobe that Sqoop submits to the cluster.

### Sqoop Architecture



Oozie’s sqoop action helps users run Sqoop jobs as part of the workflow. The following elements are part of the Sqoop action

1. job-tracker (required)
2. name-node (required)
3. prepare
4. job-xml
5. configuration
6. command (required if arg is not used)
7. arg (required if command is not used)
8. file
9. archive

The action needs to know the JobTracker and the NameNode of the underlying Hadoop cluster where Oozie has to run the sqoop action .The <prepare> section is optional and is typically used as a preprocessor to delete output directories or HCatalog table partitions or to create some directories required for the action. This delete helps make the action repeatable and enables retries after failure.

The <job-xml> element or the <configuration> section can be used to capture all of the Hadoop job configuration properties.For hive action we will be using the <job-xml> tag to pass the hive-site.xml.This way, the hive-site.xml is just reused in its entirety and no additional configuration settings or special files are necessary.

Oozie also supports the <file> and <archive> elements for actions that need them. This is the native, Hadoop way of packaging libraries, archives, scripts, and other data files that jobs need, and Oozie provides the syntax to handle them.

The arguments to Sqoop are sent either through the <command> element in one line or broken down into many <arg> elements.

<action name="**mySqoopAction**">

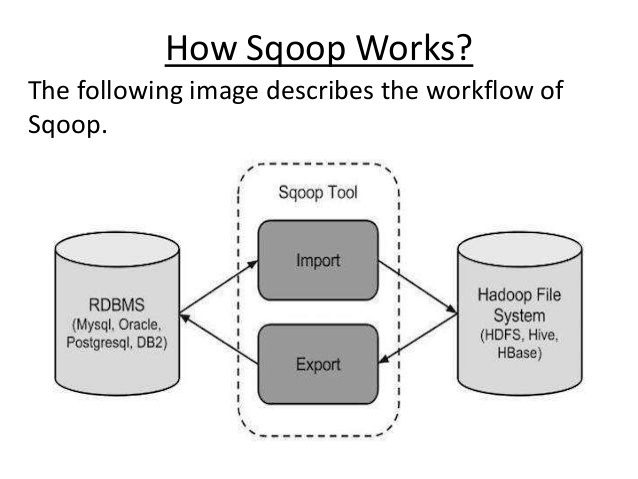
<sqoop>

<command>**import --connect jdbc:hsqldb:file:db.hsqldb** --table

test\_table --target-dir hdfs://localhost:8020/user/joe/sqoop\_tbl -m 1</command>

</sqoop>

</action>



### Sample Sqoop Workflow

Below are the steps to run sample sqoop action to get data from Mysql table on HDFS.

1. Configure **job.properties**

Example:

nameNode=hdfs://<namenode-host>:8020

jobTracker=<rm-host>:8050

queueName=default

examplesRoot=examples

oozie.use.system.libpath=true

oozie.wf.application.path=${nameNode}/user/${user.name}

oozie.libpat=/user/root

1. Configure **Workflow.xml**

Example:

1. <?xml version="1.0" encoding="UTF-8"?>
2. <workflow-app xmlns="uri:oozie:workflow:0.2" name="sqoop-wf">
3. <start to="sqoop-node"/>
4. <action name="sqoop-node">
5. <sqoop xmlns="uri:oozie:sqoop-action:0.2">
6. <job-tracker>${jobTracker}</job-tracker>
7. <name-node>${nameNode}</name-node>
8. <configuration>
9. <property>
10. <name>mapred.job.queue.name</name>
11. <value>${queueName}</value>
12. </property>
13. </configuration>
14. <command>import --connect jdbc:mysql://<mysql-server-hostname>:3306/<database-name> --username <mysql-database-username> --table <table-name> --driver com.mysql.jdbc.Driver --m 1</command>
15. </sqoop>
16. <ok to="end"/>
17. <error to="fail"/>
18. </action>
19. <kill name="fail">
20. <message>Sqoop failed, error message[${wf:errorMessage(wf:lastErrorNode())}]</message>
21. </kill>
22. <end name="end"/>
23. </workflow-app>
24. Upload **workflow.xml** and shell script to "**oozie.wf.application.path**" defined in **job.properties**
25. Follow below command to run Oozie workflow

“oozie job -oozie http://<oozie-server-hostname>:11000/oozie -config /$PATH/job.properties –run”

.

### Benefits of using Sqoop

Below are the advantages of Apache Sqoop,

* Allows the transfer of data with a variety of structured data stores like Postgres, Oracle, Teradata, and so on.
* Since the data is transferred and stored in Hadoop, Sqoop allows us to offload certain processing done in the **ETL** (**Extract**, **Load** and **Transform**) process into low-cost, fast, and effective Hadoop processes.
* Sqoop can execute the data transfer in parallel, so execution can be quick and more cost effective.
* Helps to integrate with sequential data from the mainframe. This helps not only to limit the usage of the mainframe, but also reduces the high cost in executing certain jobs using mainframe hardware.