Problem Statement:

Implement the concept given in below blog link and share the complete steps along with screenshots.

https://acadgild.com/blog/oozie-job-scheduling-in-hive/

Solution:

The purpose of this document is to learn how to schedule a Hive job using Oozie. In production, where we need to run the same job for multiple times, or, we have multiple jobs that should be executed one after another, we need to schedule our job using some scheduler. There are multiple ways to automate jobs, however, here we will work with Oozie. We will begin with understanding what Oozie is and Oozie job scheduling.

Oozie, an open source Apache project is a job scheduler that manages Hadoop jobs. In short, Oozie schedules long list of works sequentially into one job.

To schedule Hive job using Oozie, we need to write a Hive-action. Our Oozie job will consist of mainly three things.

- **workflow.xml**
- **job. properties**
- **Hive script**

```
[cloudera@quickstart ~]$ cd /home/cloudera/ankita
[cloudera@quickstart ankita]$ vi job.properties
[cloudera@quickstart ankita]$ vi create_table.hql
[cloudera@quickstart ankita]$ vi workflow.xml
[cloudera@quickstart ankita]$ ls
create_table.hql  job.properties  workflow.xml
[cloudera@quickstart ankita]$ hadoop fs -mkdir -p /user/oozie/workflows
```

job. properties

This file consists of all the variable definition that we will use in our workflow.xml. Let's say, in workflow.xml, we have mentioned a property as below:

<name-node>${nameNode}</name-node>

So, in our job. properties file, we must declare *$nameNode* and assign the relative path.

oozie.libpath=${nameNode}/user/oozie/share/lib/hive
Indicates the path (in hdfs) where all the respective jars are present.

oozie.wf.application.path=${nameNode}/user/${user.name}/workflows
This is the place where from your application will get the dependent files.

```
[cloudera@quickstart ankita]$ cat job.properties
nameNode=hdfs://localhost:8020
jobTracker=localhost:8032
oozie.libpath=hdfs://localhost:8020/user/oozie/share/lib/hive
oozie.use.system.libpath=true
oozie.wf.application.path=hdfs://localhost:8020/user/oozie/workflows
appPath=hdfs://localhost:8020/user/oozie/workflows
```

Workflow.xml

This is the place where we write our Oozie action. It contains all the details of files, scripts, required to schedule and run Oozie job. As the name suggests, it is an XML file where we need to mention the details in a proper tag.

```
[cloudera@quickstart ankita]$ cat workflow.xml
<workflow-app name="HiveOozieDemo" xmlns="uri:oozie:workflow:0.1">
<start to="demo-hive"/>
<action name="demo-hive">
<hive xmlns="uri:oozie:hive-action:0.2">
<job-tracker>localhost:8032</job-tracker>
<name-node>hdfs://localhost:8020</name-node>
<job-xml>hdfs://localhost:8020/user/oozie/workflows/hive-site.xml</job-xml>
<configuration>
<property>
<name>oozie.hive.defaults</name>
<value>hdfs://localhost:8020/user/oozie/workflows/hive-site.xml</value>
</property>
<property>
<name>hadoop.proxyuser.oozie.hosts</name>
<value>*</value>
</property>
<property>
<name>hadoop.proxyuser.oozie.groups</name>
<value>*</value>
</property>
</configuration>
<script>create_table.hql</script>
</hive>
<ok to="end"/>
<error to="end"/>
</action>
<end name="end"/>
</workflow-app>
[cloudera@quickstart ankita]$
```

Now let us try to understand what exactly the content of workflow.xml means.

The first line creates a workflow app and we assign a name (according to our convenience) to recognize the job.

<workflow-app name="HiveOozieDemo">
Indicates, we are creating a workflow app whose name is 'HiveOozieDemo'. All the other properties will remain inside this main tag.

<start to="demo-hive"/>
<action name="demo-hive">
Quite self-explanatory are the above two tags which says, give a name to our action (here 'demo-hive') and when <action name> matches, start our oozie job.

<hive xmlns="uri:oozie:hive-action:0.2">
The line above is very important as, it says what kind of action we are going to run. It can be a MR action, or a Pig action, or Hive. Here we have given the name as Hive-action.

<job-tracker>${jobTracker}</job-tracker>
<name-node>${nameNode}</name-node>
<job-xml>${appPath}/hive-site.xml</job-xml>
All the above tags point to the variable where our job-tracker, NameNode, and Hive-site.xml is present. The exact declaration of these variables is done in Job.properties file.

<script>create_table.hql</script>
We need to fill in the exact name of our script file (here, it is a Hive script file) which will be looked for and the query will get executed.

create_table.hql

This is the Hive script which we want to schedule in Oozie.

```
[cloudera@quickstart ankita]$ cat create_table.hql
USE default;
CREATE TABLE HiveOozie
(
id INT,
name STRING
)
ROW FORMAT DELIMITED FIELDS TERMINATED BY ',';
```

Now we will create a directory in hdfs and put the workflow.xml, create_table.hql and hive-site.xml files in that /user/oozie/workflows directory.

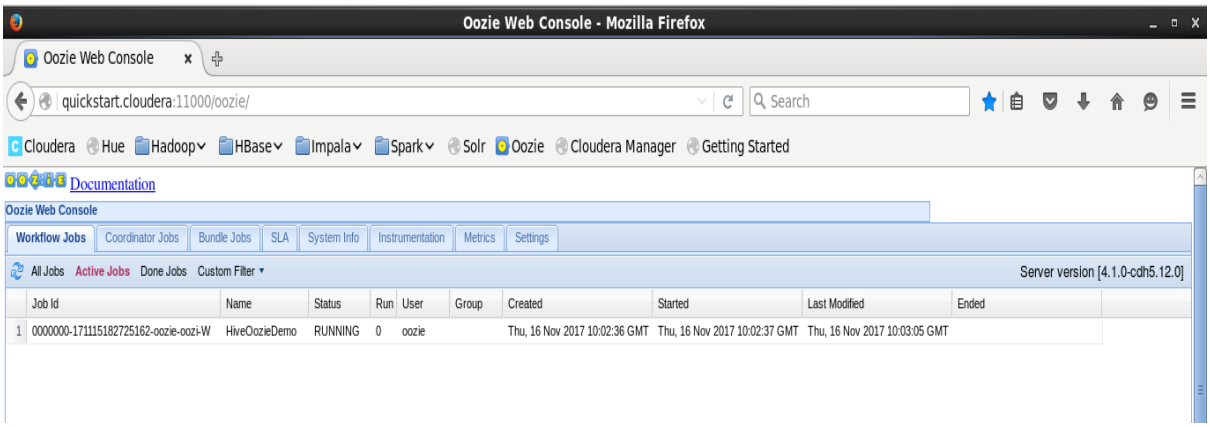
After successfully putting the files we will now run the oozie job by command:

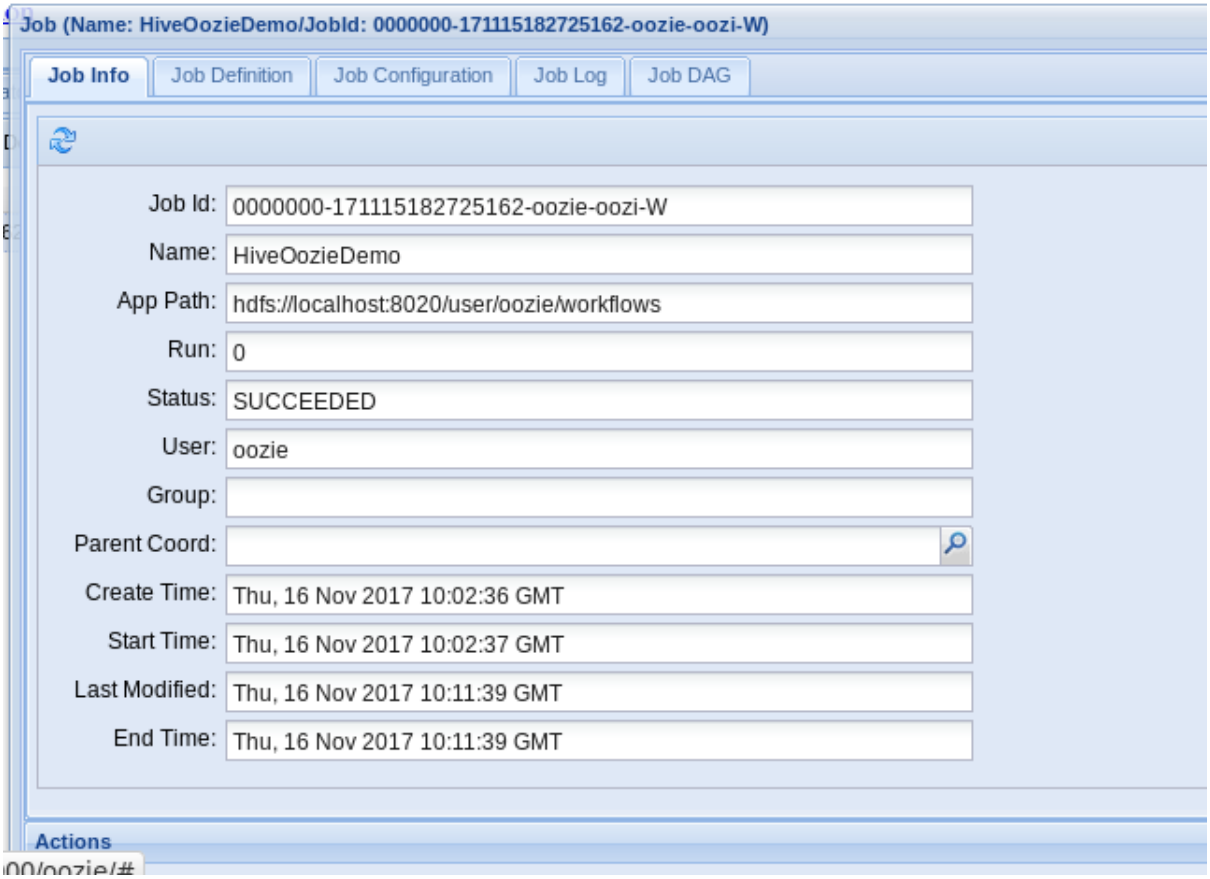
sudo -u oozie oozie job -oozie http://localhost:11000/oozie -config /home/cloudera/ankita/job.properties –run

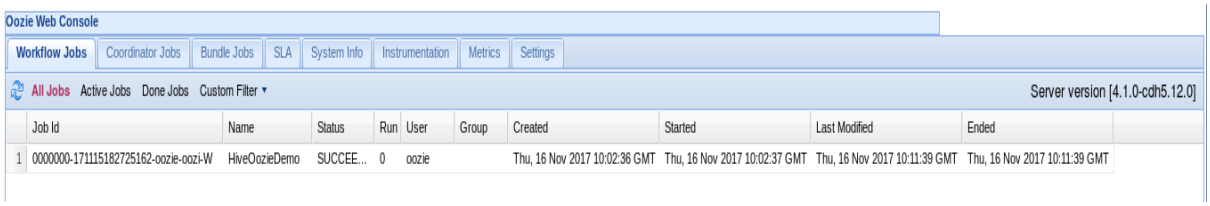
```
[cloudera@quickstart ankita]$ hadoop fs -copyFromLocal /etc/alternatives/hive
-conf/hive-site.xml /user/oozie/workflows
[cloudera@quickstart ankita]$ hadoop fs -copyFromLocal workflow.xml /user/ooz
ie/workflows
[cloudera@quickstart ankita]$ hadoop fs -copyFromLocal create_table.hql /user
/oozie/workflows
[cloudera@quickstart ankita]$ hadoop fs -ls /user/oozie/workflows
Found 3 items
-rw-r--r--   1 cloudera supergroup        109 2017-11-16 01:58 /user/oozie/wo
rkflows/create_table.hql
-rw-r--r--   1 cloudera supergroup       1937 2017-11-16 01:57 /user/oozie/wo
rkflows/hive-site.xml
-rw-r--r--   1 cloudera supergroup        766 2017-11-16 01:58 /user/oozie/wo
rkflows/workflow.xml
[cloudera@quickstart ankita]$ sudo -u oozie oozie job -oozie http://localhost
:11000/oozie -config /home/cloudera/ankita/job.properties -run
job: 0000000-171115182725162-oozie-oozi-W
[cloudera@quickstart ankita]$
```

After we run the job, we can check the status by using Oozie console.

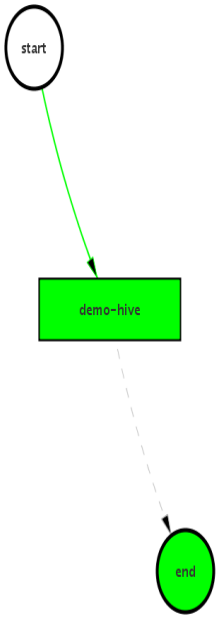
http://quickstart.cloudera:11000/oozie/







DAG for the HiveOozieDemo



Lastly we will check in hive whether the table is created or not.

```
[cloudera@quickstart ankita]$ hive

Logging initialized using configuration in file:/etc/hive/conf.dist/hive-log4j.properties
WARNING: Hive CLI is deprecated and migration to Beeline is recommended.
hive> use default;
OK
Time taken: 2.111 seconds
hive> show tables;
OK
hiveoozie
Time taken: 1.706 seconds, Fetched: 1 row(s)
hive> describe hiveoozie;
OK
id                      int
name                    string
Time taken: 1.044 seconds, Fetched: 2 row(s)
hive>
```