ASSIGNMENT 10.1

● The workflow of Oozie and its Benefits

Workflow in Oozie is a sequence of actions arranged in a control dependency **DAG (Direct Acyclic Graph)**. The actions are in controlled dependency as the next action can only run as per the output of current action. Subsequent actions are dependent on its previous action. A workflow action can be a **Hive action, Pig action, Java action, Shell action**, etc. There can be decision trees to decide how and on which condition a job should run.

A fork is used to run multiple jobs in parallel. Oozie workflows can be parameterized (variables like **${nameNode}** can be passed within the workflow definition). These parameters come from a configuration file called as property file. (More on this explained in the following chapters).

Example :

Consider we want to load a data from external hive table to an ORC Hive table.

Step 1 − DDL for Hive external table (say **external.hive**)

```
Create external table external_table(
   name string,
   age int,
   address string,
   zip int
)
row format delimited
fields terminated by ','
stored as textfile
location '/test/abc';
```

Step 2 − DDL for Hive ORC table (say orc.hive)

```
Create Table orc_table(
   name string, -- Concate value of first name and last name with space as seperator
   yearofbirth int,
   age int, -- Current year minus year of birth
   address string,
   zip int
)
STORED AS ORC
;
```

Step 3− Hive script to insert data from external table to ORC table (say Copydata.hql)

```
use ${database_name}; -- input from Oozie
```

```
insert into table orc_table
select
concat(first_name,' ',last_name) as name,
yearofbirth,
year(from_unixtime) --yearofbirth as age,
address,
zip
from external_table
;
```

Step 4 − Create a workflow to execute all the above three steps. (let’s call it workflow.xml)

<!-- This is a comment -->

```
<workflow-app xmlns = "uri:oozie:workflow:0.4" name = "simple-Workflow">

   <start to = "Create_External_Table" />


   <!—Step 1 -->


   <action name = "Create_External_Table">

      <hive xmlns = "uri:oozie:hive-action:0.4">

         <job-tracker>xyz.com:8088</job-tracker>

         <name-node>hdfs://rootname</name-node>

         <script>hdfs_path_of_script/external.hive</script>

      </hive>


      <ok to = "Create_orc_Table" />

      <error to = "kill_job" />

   </action>


   <!—Step 2 -->


   <action name = "Create_orc_Table">

      <hive xmlns = "uri:oozie:hive-action:0.4">

         <job-tracker>xyz.com:8088</job-tracker>
      <name-node>hdfs://rootname</name-node>
      <script>hdfs_path_of_script/orc.hive</script>
    </hive>


    <ok to = "Insert_into_Table" />
    <error to = "kill_job" />
  </action>
```

<!—Step 3 -->

```


  <action name = "Insert_into_Table">
    <hive xmlns = "uri:oozie:hive-action:0.4">
      <job-tracker>xyz.com:8088</job-tracker>
      <name-node>hdfs://rootname</name-node>
      <script>hdfs_path_of_script/Copydata.hive</script>
      <param>database_name</param>
    </hive>


    <ok to = "end" />
    <error to = "kill_job" />
  </action>


  <kill name = "kill_job">
    <message>Job failed</message>
  </kill>


  <end name = "end" />


</workflow-app>
```

Benefits :

Oozie has client API and command line interface which can be used to launch, control and monitor job from Java application.

- Using its Web Service APIs one can control jobs from anywhere.
- Oozie has provision to execute jobs which are scheduled to run periodically.
- Oozie has provision to send email notifications upon completion of jobs.

● The workflow of Scoop and its Benefits :

The sqoop action runs a Sqoop job.

The workflow job will wait until the Sqoop job completes before continuing to the next action.

To run the Sqoop job, you have to configure the sqoop action with the =job-tracker=, name-node and Sqoop command or arg elements as well as configuration.

A sqoop action can be configured to create or delete HDFS directories before starting the Sqoop job.

Sqoop configuration can be specified with a file, using the job-xml element, and inline, using the configuration elements.

Oozie EL expressions can be used in the inline configuration. Property values specified in the configuration element override values specified in the job-xml file.

Note that Hadoop mapred.job.tracker and fs.default.name properties must not be present in the inline configuration.

As with Hadoop map-reduce jobs, it is possible to add files and archives in order to make them available to the Sqoop job. Refer to the [WorkflowFunctionalSpec#FilesAchives][Adding Files and Archives for the Job] section for more information about this feature.

The sqoop action runs a Sqoop job.

The workflow job will wait until the Sqoop job completes before continuing to the next action.

To run the Sqoop job, you have to configure the sqoop action with the =job-tracker=, name-node and Sqoop command or arg elements as well as configuration.

A sqoop action can be configured to create or delete HDFS directories before starting the Sqoop job.

Sqoop configuration can be specified with a file, using the job-xml element, and inline, using the configuration elements.

Oozie EL expressions can be used in the inline configuration. Property values specified in the configuration element override values specified in the job-xml file.

Note that Hadoop mapred.job.tracker and fs.default.name properties must not be present in the inline configuration.

As with Hadoop map-reduce jobs, it is possible to add files and archives in order to make them available to the Sqoop job. Refer to the [WorkflowFunctionalSpec#FilesAchives][Adding Files and Archives for the Job] section for more information about this feature.

Syntax:

```
<workflow-app name="[WF-DEF-NAME]" xmlns="uri:oozie:workflow:0.1">
    ...
    <action name="[NODE-NAME]">
        <sqoop xmlns="uri:oozie:sqoop-action:0.2">
            <job-tracker>[JOB-TRACKER]</job-tracker>
            <name-node>[NAME-NODE]</name-node>
            <prepare>
               <delete path="[PATH]"/>
               ...
               <mkdir path="[PATH]"/>
               ...
            </prepare>
            <configuration>
                <property>
                    <name>[PROPERTY-NAME]</name>
                    <value>[PROPERTY-VALUE]</value>
                </property>
                ...
            </configuration>
            <command>[SQOOP-COMMAND]</command>
            <arg>[SQOOP-ARGUMENT]</arg>
            ...
            <file>[FILE-PATH]</file>
            ...
            <archive>[FILE-PATH]</archive>
            ...
        </sqoop>
        <ok to="[NODE-NAME]"/>
        <error to="[NODE-NAME]"/>
    </action>
    ...
</workflow-app>
```

The prepare element, if present, indicates a list of paths to delete or create before starting the job. Specified paths must start withhdfs://HOST:PORT .

The job-xml element, if present, specifies a file containing configuration for the Sqoop job. As of schema 0.3, multiple job-xml elements are allowed in order to specify multiple job.xmlfiles.

The configuration element, if present, contains configuration properties that are passed to the Sqoop job.