**ASSIGNMENT-10.1**

1. The workflow of Oozie and its Benefits

Oozie is a ***server based Workflow Engine*** specialized in running workflow jobs with actions that run Hadoop Map/Reduce and Pig jobs.

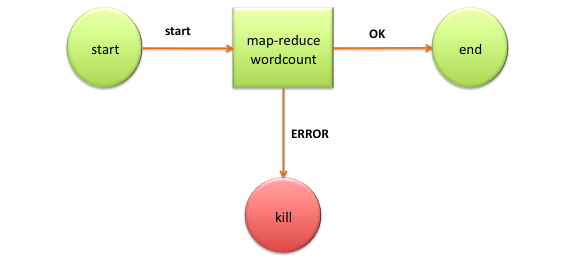
Oozie is a Java Web-Application that runs in a Java servlet-container.

For the purposes of Oozie, a workflow is a collection of actions (i.e. Hadoop Map/Reduce jobs, Pig jobs) arranged in a control dependency DAG (Direct Acyclic Graph). "control dependency" from one action to another means that the second action can't run until the first action has completed.Oozie workflows definitions are written in hPDL (a XML Process Definition Language similar to [JBOSS JBPM](http://www.jboss.org/jbossjbpm/) jPDL).

Oozie workflow actions start jobs in remote systems (i.e. Hadoop, Pig). Upon action completion, the remote systems callback Oozie to notify the action completion, at this point Oozie proceeds to the next action in the workflow.Oozie workflows contain control flow nodes and action nodes. Control flow nodes define the beginning and the end of a workflow ( start , end and fail nodes) and provide a mechanism to control the workflow execution path ( decision , fork and join nodes). Action nodes are the mechanism by which a workflow triggers the execution of a computation/processing task. Oozie provides support for different types of actions: Hadoop map-reduce, Hadoop file system, Pig, SSH, HTTP, eMail and Oozie sub-workflow. Oozie can be extended to support additional type of actions. Oozie workflows can be parameterized (using variables like ${inputDir} within the workflow definition). When submitting a workflow job values for the parameters must be provided. If properly parameterized (i.e. using different output directories) several identical workflow jobs can concurrently.

* ***Oozie Workflow jobs*** — Represented as directed acyclical graphs to specify a sequence of actions to be executed.
* ***Oozie Coordinator jobs*** — Represent Oozie workflow jobs triggered by time and data availability.
* ***Oozie Bundle***— Facilitates packaging multiple coordinator and workflow jobs, and makes it easier to manage the life cycle of those jobs.

**Workflow Diagram**



## Running the Workflow

A topology runs in a distributed manner, on multiple worker nodes. Storm spreads the tasks evenly on all the worker nodes. The worker node’s role is to listen for jobs and start or stop the processes whenever a new job arrives.

Note − The workflow and hive scripts should be placed in HDFS path before running the workflow.

oozie job --oozie http://host\_name:8080/oozie -D

oozie.wf.application.path=hdfs://namenodepath/pathof\_workflow\_xml/workflow.xml-run

This will run the workflow once.

To check the status of job you can go to Oozie web console -- http://host\_name:8080/

By clicking on the job you will see the running job. You can also check the status using Command Line Interface (We will see this later). The possible states for workflow jobs are: PREP, RUNNING, SUSPENDED, SUCCEEDED, KILLED and FAILED.

In the case of an action start failure in a workflow job, depending on the type of failure, Oozie will attempt automatic retries. It will request a manual retry or it will fail the workflow job. Oozie can make HTTP callback notifications on action start/end/failure events and workflow end/failure events. In the case of a workflow job failure, the workflow job can be resubmitted skipping the previously completed actions. Before doing a resubmission the workflow application could be updated with a patch to fix a problem in the workflow application code.

**BENEFITS:**

One advantage of the Oozie framework is that it is fully integrated with the Apache Hadoop stack and supports Hadoop jobs for Apache MapReduce, Pig, Hive, and Sqoop. In addition, it can be used to schedule jobs specific to a system, such as Java programs.

1. The workflow of Sqoop and its Benefits

**Apache Sqoop** is a tool designed for efficiently transferring bulk data between Apache Hadoop and structured data stores such as relational databases. Ex: MySql,Oracle.PostgreSql etc.

Apache Sqoop automates most of this process,relying on the database to describe the schema for the data to be imported.Sqoop uses MapReduce to import and export the data, which provides parallel operation as well as fault tolerance.

When we execute any Apache Sqoop command it converts in to a Java jar file which executes and inside the jar you can see the Map and Reduce execution.

***Sqoop is “SQL to Hadoop and Hadoop to SQL”***

With Apache Sqoop, you can import data from a relational database system or a mainframe into HDFS.The input to the import process is either database table or mainframe datasets. For databases, Apache Sqoop will read the table row-by-row in to HDFS. The output of this import process is a set of files containing a copy of the imported table or datasets.The import process is performed in parallel.For this reason, the output will be in multiple files.These files may be delimited text files(for example with commas or tabs separating each field), or binary Avro or Sequence Files containing serialized record data.

**Apache Sqoop**is a collection of related tools. To use **Apache Sqoop**, you specify the tool you want to use and the arguments that control the tool.

Syntax: **$ sqoop tool-name [tool-arguments]**

Now to move further it’s a good practice to know the tools **Apache Sqoop**provides. Open the putty / cmd and type in **sqoop**help , you will see a number of tool-name **Apache Sqoop**provides.

Few sample Sqoop commands are:

**--connect** => This argument takes the jdbc connector

**--driver** => This argument tells **Apache Sqoop**which jdbc driver we will be using.

**--username** => This argument takes the value of the database user from which we are going to connect with the database.

**-- password**=> This argument takes the value of the database user password to Authenticate the DB user.

**--table**     => The name of the table we are going to sqoop into.

**--m**     => Number of mapper we want to create while processing.

**--target-dir**  => HDFS location where you want to store the imported Data.

# Advantages of Sqoop:

# Below are the advantages of Apache Sqoop, which is also the reason for choosing this technology in this layer.

* 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.

**Apache Sqoop Limitations**

* The Change Data Capture and incremental imports are bit challenging and needs external logic for implementation.
* Failures needs special handling like in case of partial import or export.