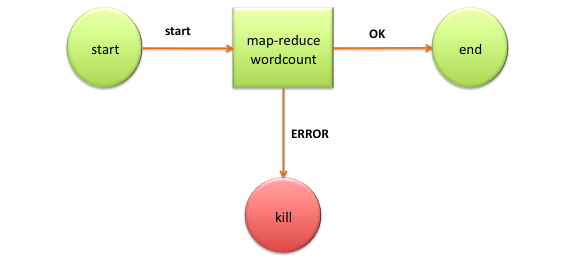
**1 Oozie Work Flow**

* Oozie is a workflow scheduler system to manage Apache Hadoop jobs and it is server based work flow specialized in running workflow jobs with actions that run Hadoop Map/Reduce and Pig jobs.
* Oozie Workflow jobs are Directed Acyclical Graphs (DAGs) of actions.
* Oozie Coordinator jobs are recurrent Oozie Workflow jobs triggered by time (frequency) and data availability.
* Oozie is integrated with the rest of the Hadoop stack supporting several types of Hadoop jobs out of the box (such as Java map-reduce, Streaming map-reduce, Pig, Hive, Sqoop and Distcp) as well as system specific jobs (such as Java programs and shell scripts).
* Oozie is a scalable, reliable and extensible system.
* 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.



### Fig:10.1 WordCount Workflow Example

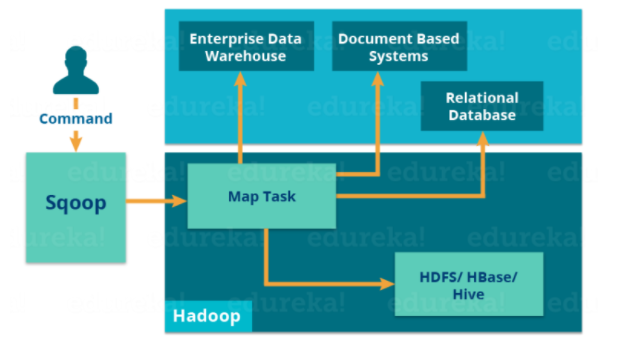
**Benefits Of Oozie**

1. Oozie is designed to scale in a Hadoop cluster. Each job will be launched from a different datanode. 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.
2. 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.
3. Oozie is the only workflow manager with built-in Hadoop actions, making workflow development, maintenance and troubleshooting easier.
4. 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.
5. Oozie is proven to scale in some of the world’s largest clusters. The [**white paper**](https://docs.google.com/viewer?a=v&pid=sites&srcid=ZGVmYXVsdGRvbWFpbnxzd2VldHdvcmtzaG9wMjAxMnxneDo1NzRhYjZlNzdmNTM1Yjgw#!)discusses a deployment at Yahoo! that can handle 1250 job submissions a minute.
6. 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.
7. Oozie Coordinator allows triggering actions when files arrive at HDFS. This will be challenging to implement anywhere else.
8. Oozie is supported by Hadoop vendors. If there is ever an issue with how the workflow manager integrates with Hadoop

**2 Work Flow of Sqoop**

The import tool imports individual tables from RDBMS to HDFS. Each row in a table is treated as a record in HDFS.

When we submit Sqoop command, our main task gets divided into sub tasks which is handled by individual Map Task internally. Map Task is the sub task, which imports part of data to the Hadoop Ecosystem. Collectively, all Map tasks imports the whole data.

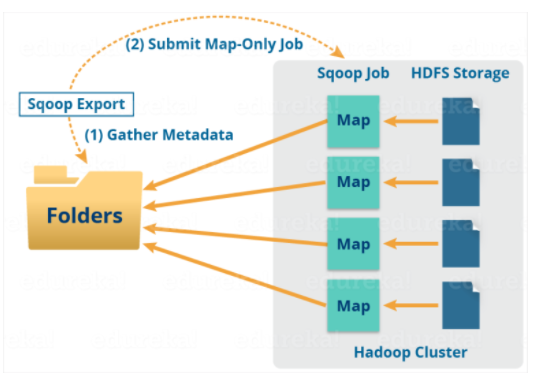


Export also works in a similar manner.

The export tool exports a set of files from HDFS back to an RDBMS. The files given as input to Sqoop contain records, which are called as rows in table.

When we submit our Job, it is mapped into Map Tasks which brings the chunk of data from HDFS. These chunks are exported to a structured data destination. Combining all these exported chunks of data, we receive the whole data at the destination, which in most of the cases is an RDBMS (MYSQL/Oracle/SQL Server).

Reduce phase is required in case of aggregations. But, Apache Sqoop just imports and exports the data; it does not perform any aggregations. Map job launch multiple mappers depending on the number defined by user. For Sqoop import, each mapper task will be assigned with a part of data to be imported. Sqoop distributes the input data among the mappers equally to get high performance. Then each mapper creates connection with the database using JDBC and fetches the part of data assigned by Sqoop and writes it into HDFS or Hive or HBase based on the arguments provided in the CLI.



**Benefits of 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