MODULE 3

__/__/____

Apache Pig –

It is an abstraction over mapReduce. It is a tool/platform which is used to analyze larger sets of data representing them as data flows. Pig is generally used with Hadoop, to perform all kinds of data manipulation operations.

To analyze data using Apache Pig, programmers need to write scripts using Piglatin language.

• Advantages –

- Using piglatin, programmers can perform MapReduce tasks easily without having to write complex codes in JAVA.
- It uses multi-query approach, thereby reducing the size of code.
- It provides many built-in operators to support data operations like joins, filters, ordering, etc.

• Features –

1. Rich set of operators – like join, sort, filer, filter, etc.
2. Ease of programming – similar to SQL, so easy to implement.
3. Optimization opportunities – tasks in apache pig are optimized automatically
4. Extensibility – Using existing operators, users can develop their own functions to read, process and write data.
5. UDF's – it provides facility of User Defined Functions in other languages, and invoke or embed them in pig scripts.
6. Handles all kinds of data – it analyzes both structured and unstructured data and stores them in HDFS.

• Pig Latin –

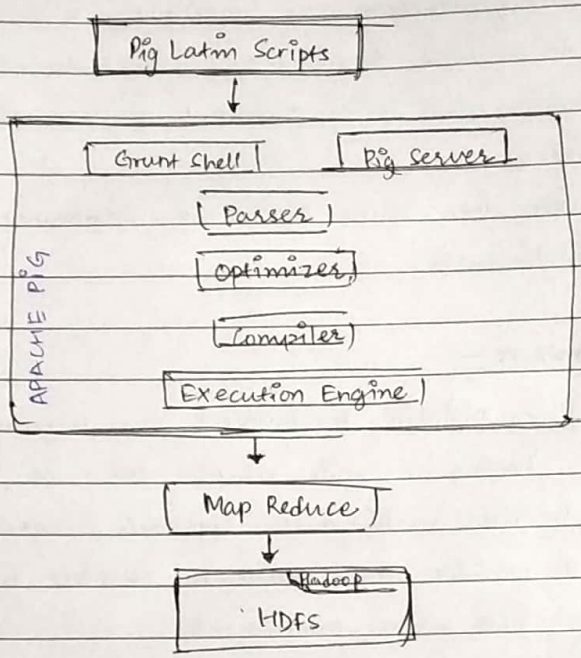
It is a high-level programming language useful for analyzing large data sets. It provides various operators using which programmers can develop their own functions for reading, writing, and processing data.

(2)

__/__/____

It is a bag where -
→ a bag is a collection of tuples
→ a tuple is an ordered set of fields
→ a field is a piece of data.

- Pig Architecture -



- Pig Latin - the language used to analyze data in Hadoop using Pig is known as piglatin.

- Parser - Pig scripts are handled by parser. it checks the syntax of the script, does type checking and outputs it as DAG (Directed Acyclic Graph) which represents the pig latin statements & logical operators.

- Optimizer - The logical plan (DAG) is passed to the logical optimizer, which carries out logical optimizations such as projections & pushdowns.

- Compiler - it compiles the optimized logical plan into a series of MapReduce jobs.

- Execution Engine - Finally, the MapReduce jobs are submitted to Hadoop in a sorted order. These MapReduce jobs are then executed on Hadoop producing the desired results.

__/__/____

- Execution Modes of Pig -

1. Local mode - Pig runs in a single JVM and makes use of local file system. This mode is suitable only for analysis of small datasets using pig.

2. MapReduce mode - In this mode, queries written in PigLatin are translated into MapReduce jobs and are run on a Hadoop cluster. MapReduce mode with the fully distributed cluster is useful of running Pig on large datasets.

→ Analyzing Datasets Using Pig -

1. Create database and database tables in Hive.
2. Import data into Hive Tables
3. Call Hive SQL in shell script
4. View database architecture.
5. Load & store Hive data into pig relation
6. Call pig script in shell script
7. Apply pivot concept in Hive SQL.
8. View output.

HIVE -

It is an open source data warehousing system, which is exclusively used to query and analyze huge datasets stored in Hadoop. It is built on top of Apache Hadoop that facilitates easy data summarization, ad-hoc queries, and the analysis of large datasets stored in various databases and file systems that integrate with Hadoop.

→ three important functionalities of Hive are -

- Data Summarization
- Data analysis
- Data Query.

→ The query language, exclusively supported by Hive is HiveQL. it translates SQL like queries into MapReduce jobs for deploying them on Hadoop.

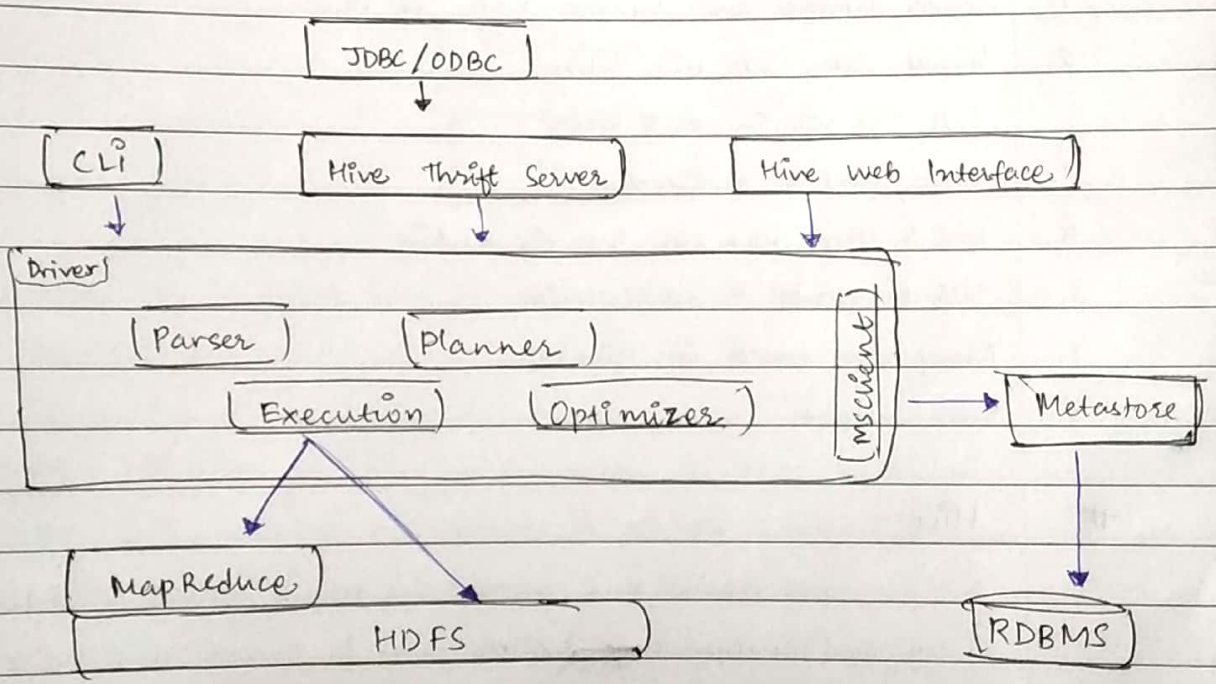
(4)

__/__/____

Reasons for Demand of Hive –

1. Its the low level interface requirement of Hadoop perfectly.
2. Supports external tables which makes it possible to process data without actually storing it in HDFS.
3. It has a rule based optimizer for optimizing logical plans.
4. Supports partitioning of data at the level of tables to improve performance.
5. Metastore or metadata store makes the lookup easy.
6. Ad hoc quering is easier.

HIVE ARCHITECTURE –



→ Metastore - It is the repository of metadata, it consists of data for each table like its location and schema, along with the information for partition metadata which lets you monitor various distributed data progress in the cluster. the metadata keeps track of the data, replicates it, and provides a backup in the case of data loss.

→ Driver - the driver recieves HiveQL statements and works like a controller. It monitors the progress & life cycle of various executions by creating

__/__/____

sessions. It stores the metadata that is generated while executing the HiveQL statement. When the reducing operation is completed by the MapReduce job, the driver collects the data points and query results.

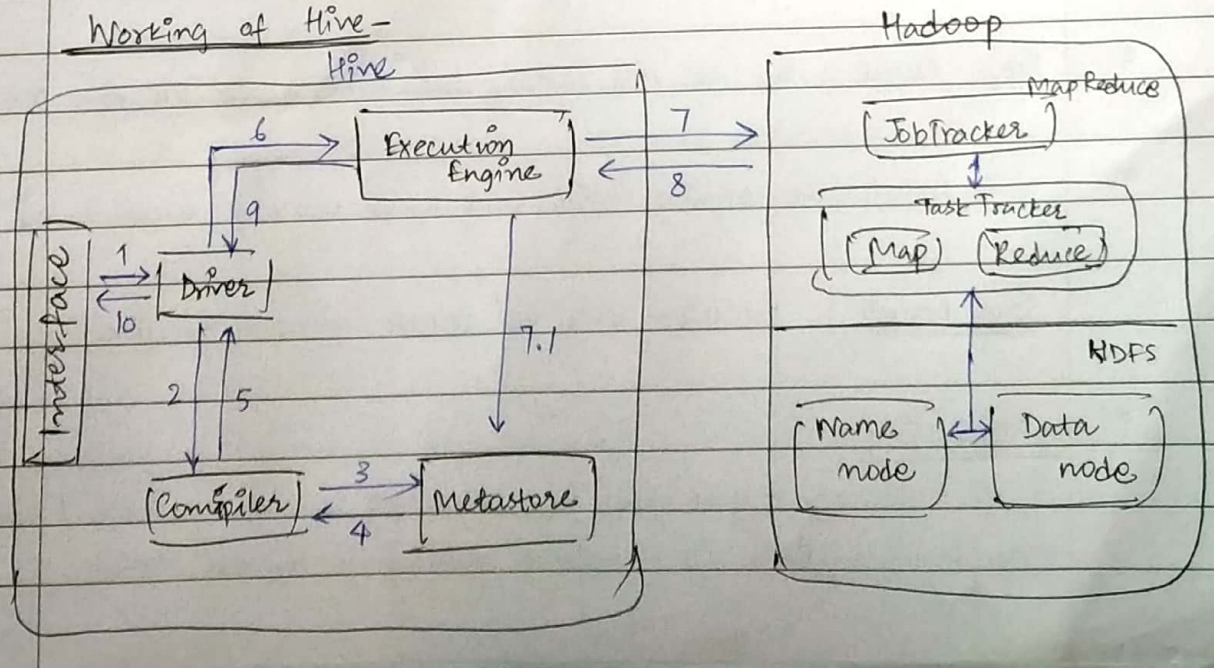
→ Compiler - It is assigned with the task of converting a HiveQL query into a MapReduce input. It includes a method to execute the steps & tasks needed to let HiveQL output as needed by MapReduce.

→ Optimizer - This performs various transformation steps for aggregation and pipeline conversion by a single join for multiple joins. It also is assigned to split a task while transforming data, before the reduce operations, for improved efficiency and scalability.

→ Executor - It executes tasks after the compilation and optimization steps. It directly interacts with Hadoop Job Tracker for scheduling the tasks to be run.

→ CLI, UI & Thrift Servers - the command line interface (CLI) and the user interface (UI) submit queries and process monitoring and instructions so that the external users can interact with Hive. Thrift server lets other clients interact with Hive.

Working of Hive -



__/__/____

1. Execute query: the Hive interface such as command line or web UI sends query to driver (JDBC, ODBC, etc) to execute.

2. Get Plan: the driver takes the help of query compiler that parses the query to check the syntax and query plan or the requirement of query.

3. Get metadata: the compiler sends metadata request to metastore.

4. Send metadata: metastore sends metadata as a response to the compiler.

5. Send Plan: the compiler checks the requirement and resends the plan to the driver. Upto here, the parsing and compiling of query is complete.

6. Execute Plan: the driver sends the execute plan to execution engine.

7. Execute Job: Internally, the process of execution job is a MapReduce job. the execution engine sends the job to JobTracker, which is in NameNode and it assigns this job to TaskTracker, which is in Data node. Here, the query executes MapReduce job.

7.1 Metadata Ops: Meanwhile in execution, the execution engine can execute metadata operations with Metastore.

8. Fetch Result: The execution engine ~~sends those~~ recieves the results from Data nodes

9. Send Results: the execution engine sends those resultant values to the driver

10. Send Results: The driver sends the results to Hive Interfaces.

o Applications

- log processing - text mining - document indexing - Google Analytics
- Sentiment analytics - predictive modeling - hypothesis testing.

__/__/____

HBase –

It is a distributed column-oriented database built on top of the Hadoop file system. It is an open source project and is horizontally scalable. It is a data model that provides quick random access to huge amounts of structured data. It leverages the fault tolerance provided by HDFS.
It is a part of Hadoop ecosystem that provides random real-time read/write access to data in Hadoop file system.

- **Storage Mechanism in HBase / HBase Data Model –**

HBase is a column-oriented database and the tables in it are sorted by row. The table schema defines only column families, which are key-value pairs. Subsequent column values are stored contiguously on the disk. Each cell value of the table has a timestamp. In short, in an HBase
  - table is a collection of rows.
  - row is a collection of column families
  - Column family is a collection of columns
  - Column is a collection of key-value pair.

~~HBase~~ HDFS	~~HDFS~~ HBase
– it is distributed file system suitable for storing large files.	it is database built on top of HDFS.
– it doesn't support fast individual record lookups.	it provides fast lookups for larger tables.
– it provides high latency batch processing	it provides low latency access to single rows.
– it provides only sequential access of data.	it uses Hash tables and provides random access of data.

→ **Need of HBase –**

1. Hadoop cannot handle high ~~volume of multi-structured data~~ velocity of random writes and reads and also cannot change a file without completely rewriting it. HBase is NOSQL, it allows fast random reads and writes in an optimized way.

(B)

__/__/____

2. With exponentially growing data, relational databases cannot handle the variety of data to render better performance. HBase provides scalability and partitioning for efficient storage and retrieval.

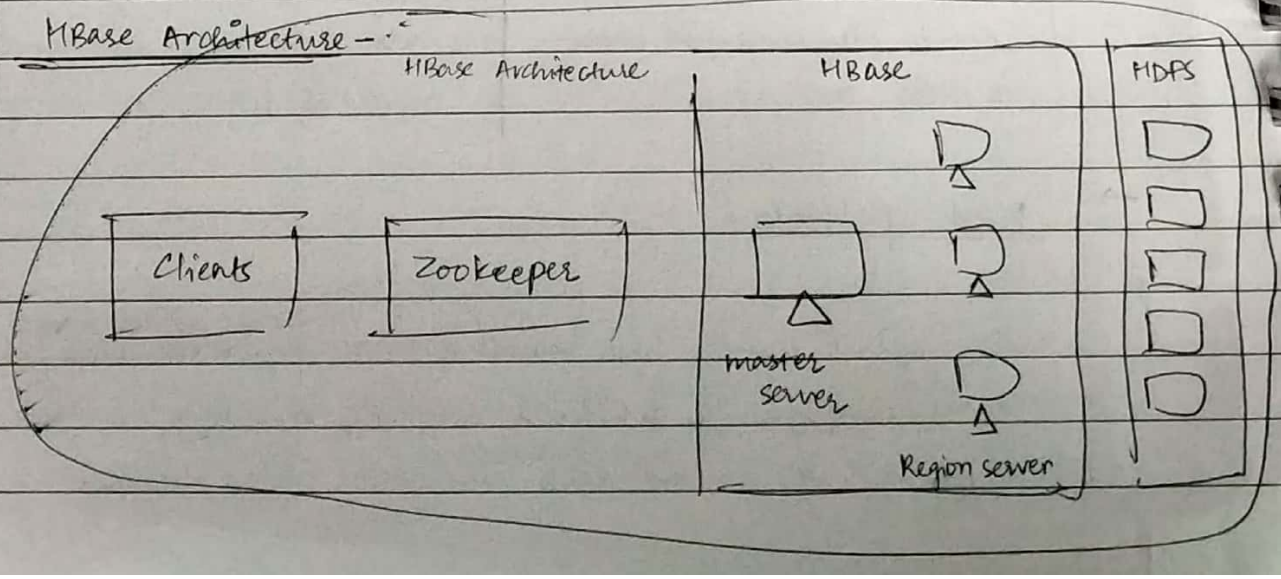
8 ~~Hadoop~~

o Features of HBase -

1. it is linearly scalable.
2. it has automatic failure support
3. it provides consistent reads & writes.
4. it has easy Java API for clients.
5. it provides data replication across clusters.
6. it integrates with Hadoop, both as a source and a destination.

	HBase	RDBMS
1.	it is schema-less, it's defined by only column families	it is governed by its schema, which describes the whole structure of tables.
2.	it is built for wide tables	for small tables
3.	it is horizontally scalable	Hard to scale
4.	No transactions are there in HBase	RDBMS is transactional.
5.	it has de-normalised data.	it will have normalized data.
6.	its good for semi-structured & structured data.	its good for structured data.

11 HBase Architecture -



- HBase Master Server -
  - it assigns regions to region servers and takes the help of Apache Zookeeper for this task.
  - Handles load balancing of the regions across region servers.
  - It unloads the busy servers and shifts the regions to less occupied servers
  - maintains the state of the cluster by negotiating the load balancing.
  - it is responsible for schema changes and other metadata operations such as creation of tables and column families.

- Region Server -
  a region contains all the rows between the start key and the end key assigned to that region. HBase tables can be divided into no. of regions in such a way that all the columns of column family is stored in one region.
  Many regions are assigned to Region Server, which is responsible for handling, managing, executing reads and writes operations on that set of regions.
  → Functionality:
  1. A region has a default size of 256 MB which can be configured according to the need.
  2. A group of regions is served to clients by a region server.
  3. A region server can serve approximately 1000 regions to the client.

- Components of Region Server are -
  1. WAL - Write Ahead Log (WAL) is a file attached to every Region Server. It stores the new data that hasn't been persisted or committed to permanent storage.
  2. Block cache - it resides at top of Region Server. It stores the frequently read data in the memory.
  3. Mem Store - it is the write cache. It stores all the incoming data before committing it to the disk or permanent memory.
  4. HFile - it is stored on HDFS. Thus, it stores the actual cells on the disk. It commits the data to HFile when the size of MemStore exceeds.

(10)

__/__/____

- Zookeeper (The Co-ordinator) –

it is an open-source project that provides services like maintaining configuration information, naming, providing distributed synchronizations, etc. It has emphemeral nodes representing different region servers. Master servers use these nodes to discover available servers.

In addition to availability, the nodes are used to track server failures or network partitions. Clients communicate with regions via zookeeper.

- Functions –
  - establishing client communication with region servers.
  - tracking server failure and network partitions
  - maintain configuration information
  - provides emphemeral nodes, which represent diff region servers.

- META table –

It is a special HBase catalog table. It maintains a list of all the regions servers in the HBase storage system. .META file maintains the table in form of keys and values. Key represents the start key of the region and its id and the value contains the path of region server.

→ Search Mechanism –

1. the client retrieves the location of the META table from zookeeper.
2. the client then requests for the location of region server of the corresponding row key from the META table to access it. the client caches this information with the location of the META table.
3. Then it will get the row location by requesting from the corresponding Region Server.

→ Write Mechanism –

1. Whenever the client has a write request, the client writes the data to the WAL the edits are then appended at the end of WAL file.

__/__/____

2. Once the data is written on to the WAL, then it is copied to MemStore.
3. Once data is placed in MemStore, then the client recieves the acknowledgement.
4. When MemStore reaches the threshold, it dumps or commits the data into a HFile.

→ **Read Mechanism** –

1. The client retrieves the location of Region server from .META server if the client does not have it in its cache memory.
2. For reading the data, the scanner first looks for row cell in block cache, Here, all the recently read key value pairs are stored.
3. If ~~scan~~ Scanner fails to find required results, it moves to MemStore, as we know this is write cache memory. there, it searches for the most recently written files, which has not been dumped yet into HFile.
4. At last, it will use bloom filters and block cache to load the data from HFile.

Compaction –

HBase combines HFiles to reduce the storage and reduce the no. of disks seeks needed for a read. this process is called compaction.

- Minor Compaction: HBase automatically picks smaller HFiles and recommits them to bigger HFiles (by merge sort). This is minor compaction. This helps in storage space optimization.
- Major Compaction: HBase merges and recommits same column families and place them together to form new HFile. It increases read performances. It drops deleted and expired cells in the process.

Crash and Data Recovery –

1. Whenever a region server fails, Zookeeper notifies to the HMaster about the failure.
2. Then HMaster distributes and allocates the regions of crashed region server to active region servers. and to recover the data, it distributes WAL to region servers.

(12)

__/__/____

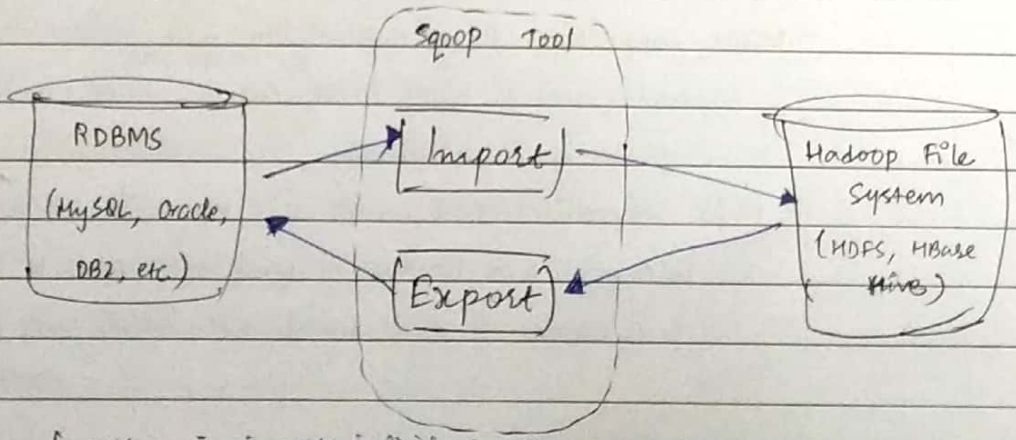
3. Each region server re-executes the WAL to build the MemStore for that failed region's column family.
4. The data is written in chronological order in WAL.
5. So, after all the region servers executes the WAL, the MemStore data for all ~~the~~ column family is recovered.

SQOOP -

It is a tool designed to transfer data between Hadoop and relational database servers. It is used for importing data from RDBMS, like MySQL, Oracle, etc into HBase, Hive or HDFS. It is also used for exporting data from HDFS into RDBMS. It is a command line interpreter.

- Features -
  1. it makes data analysis efficient.
  2. it helps in mitigating the excessive loads to external systems.
  3. it provides data interaction programmatically by generating Java classes.
  4. it parallizes data transfer for optimal systems utilization and fast performance.
  5. it supports bulk imports, and direct inputs.

Sqoop Workflow



→ Sqoop Import - it imports individual tables from RDBMS to HDFS. Each row in a table is treated as a record in HDFS. All records are stored as text data in text files or as binary data in Avro & sequence files.

__/__/____

→ Sqoop Export - it exports a set of files from HDFS to RDBMS. The files given as input to sqoop contains records. Those are read and parsed into a set of records and delimited with user-specific delimiter.

FLUME

It is a distributed and reliable service for collecting and aggregating huge amount of log data. It also has a tunable reliability mechanisms and several recovery and failover mechanisms.

• Features -

1. It ingests log data from multiple web servers into centralized store efficiently.
2. Using flume, we can get data from multiple servers immediately into Hadoop.
3. Its also used to import huge volumes of events produced by social networking sites.
4. It supports a large set of sources and destination types.
5. It can be scaled horizontally.
6. It support multi-hop flows, contextual routing, etc.

• Advantages -

1. We can store the data into any of centralized stores (HBase, HDFS)
2. It provides feature of contextual routing.
3. It is reliable, fault tolerant, scalable and customizable.
4. The transactions are channel based where 2 transactions are maintained for each msg, which guarantees msg delivery.

• Architecture -

1. Source - defines where the data is coming from, for instance a msg queue or file
2. Sinks - defined the destination of the data pipelined from various sources.
3. Channels - are pipes which establish connect b/w sources and sinks.

↳ Concepts -

1. The master acts like a reliable config service which is used by nodes for retrieving their configurations.

__/__/____

2. If the config for a particular node changes on the master then it will dynamically be updated by the master.

	SQOOP	FLUME
1.	It handles batch data	It handles real time data.
2.	It works on high volume of data.	It works on low volume of data.
3.	Sqoop load is not driven by events.	Data loading is completely event driven
4.	It has a connector based architecture.	It has agent based architecture.
5.	It supports data compression	It doesn't support data compression.
6.	main fn is transfer of static data from RDBMS to HDFS.	main fn is to transfer streaming data to HDFS.

OOZIE -

It is a scheduler system to run and manage Hadoop jobs in a distributed environment. It allows to combine multiple complex jobs to be run in a sequential order to achieve a bigger task.

It is responsible for triggering the workflow actions, which in turn uses the Hadoop execution engine to actually execute the task.

- Features -

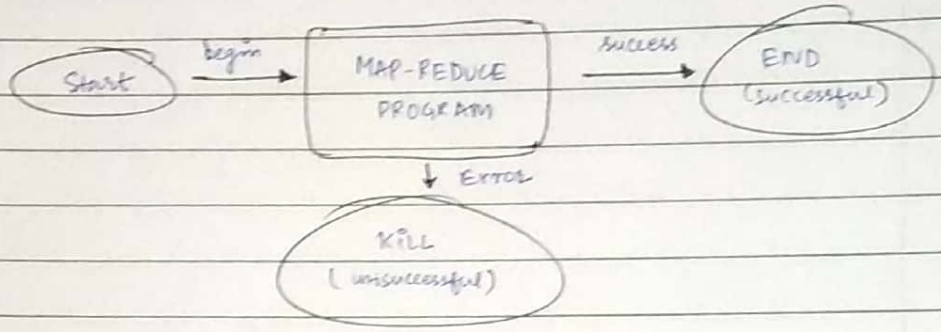
1. It has client API and command line interface which can be used to launch, control and monitor job from Java Application.
2. Using its web service APIs one can control jobs from anywhere.
3. It has provision to execute jobs which are scheduled to run periodically.
4. It has provision to send email notifications upon completing of jobs.

- Architecture -

1. Workflow engine - Responsibility of a workflow engine is to store and run workflows composed of Hadoop jobs. eg: MapReduce, Pig, Hive.
2. Coordinator engine - it runs workflow jobs based on predefined schedules and availability of data

__/__/__

- **Oozie Workflow -**



- Action Node - It represents workflow task. eg. running a MapReduce, Pig or Hive jobs, moving files into HDFS, etc.

- Control flow node - It controls the workflow execution b/w actions by allowing constructs like conditional logic wherein different branches may be followed depending on action of earlier action node.
  - Start node : designates the start of workflow job.
  - End node : signals the end of the job.
  - Error node : designates the occurrence of an error and corresponding error msg to be printed.