Class Note:

Subject : BDT

Faculty : V. Divya

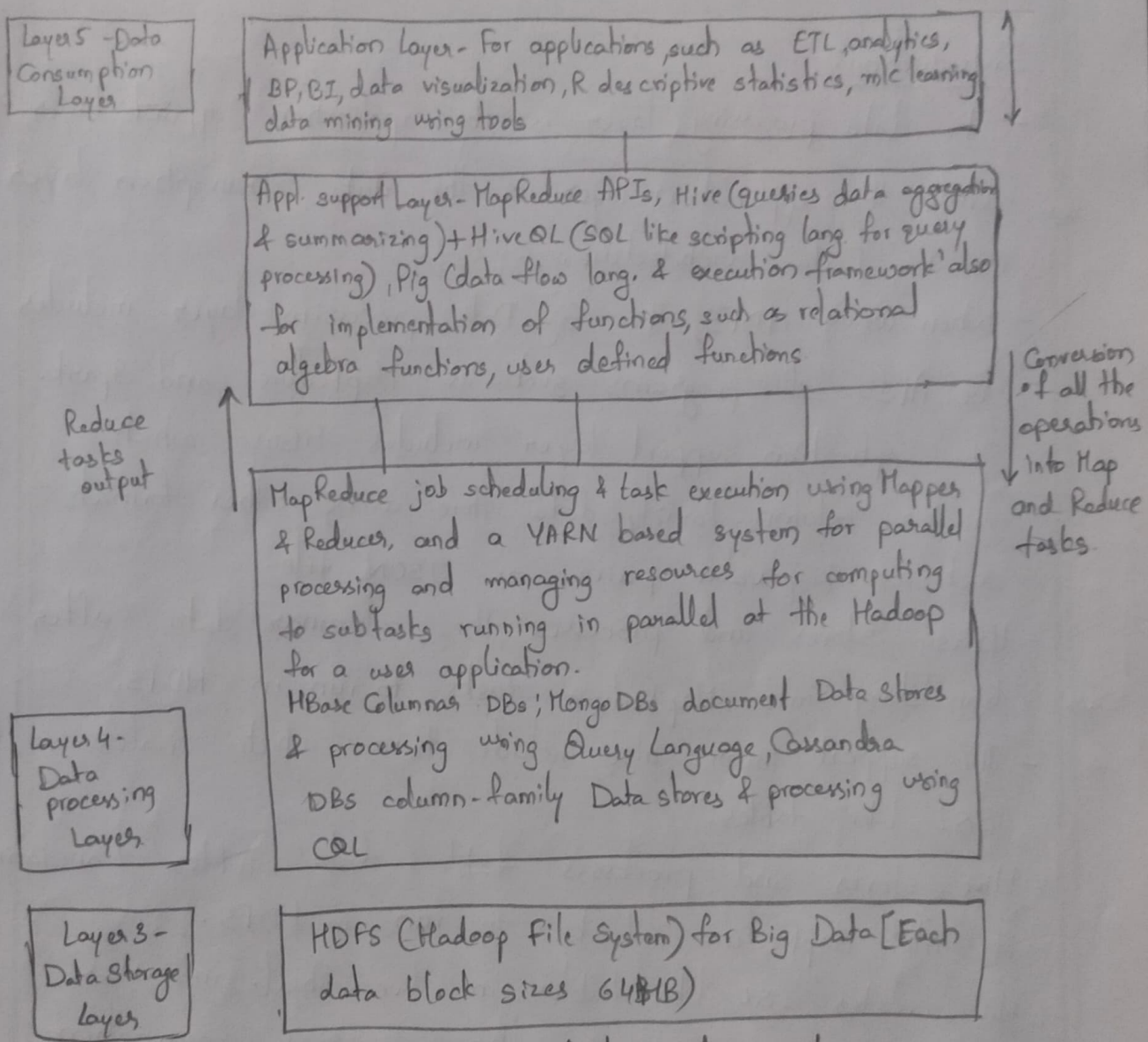
Topic: MapReduce Map Tasks, Reduce Tasks & MapReduce Execution

Unit No. : 3
Lecture No.: 1, 2, 3
Link to Session Planner (SP) S.No. .....of SP
Date Conducted:
Page No. 1

Introduction:

- The data processing layer is the application support layer, while the application layer is the data consumption layer in Big Data architecture.
- When using HDFS, the Big Data processing layer includes the APIs of programs such as MapReduce and Spark.
- The application support layer includes HBase which creates column-family data store using other formats such as key-value pairs or JSON file.
- HBase stores and processes the columnar data after translating into MapReduce tasks to run in HDFS.
- The support layer also includes Hive which creates SQL like tables.
- Hive stores and processes table data after translating it into MapReduce tasks to run in HDFS.
- Hive creates SQL like tables in Hive shell and uses HiveQL processes, queries, adhoc queries, aggregation functions and summarizing functions, such as functions to compute maximum, minimum, average of selected or grouped datasets, HiveQL is a restricted form of SQL.

- The support layer also includes Pig. Pig is a data flow language and an execution framework. which enables the usage of relational algebra in HDFS. MapReduce is the processing framework and YARN is the resource managing framework.



Big data architecture design layers.

- The above figure shows Big data architecture design layers:-

i, data storage

ii, data processing and data consumption

iii, support layer APIs for MapReduce, Hive and Pig running on top of the HDFS Data store.

Class Note:

Subject : .....B.D.T.....

Faculty : ....V.Divya.....

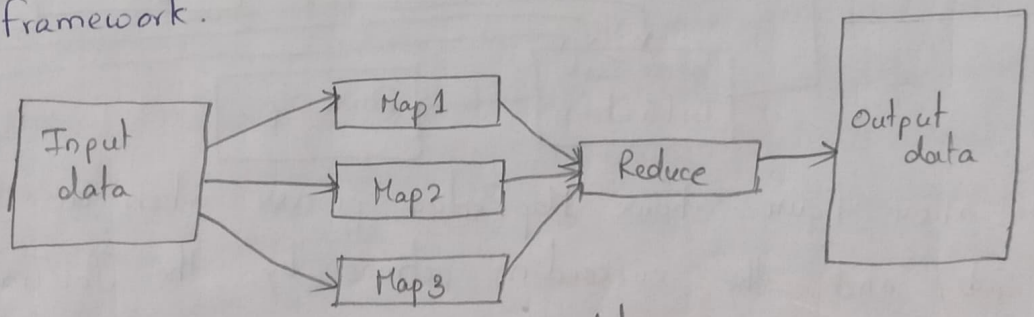
Topic: Map Reduce Map Tasks, Reduce Tasks and MapReduce Execution.

Unit No. : 3
Lecture No.: 4
Link to Session Planner (SP) S.No. .....of SP
Date Conducted:
Page No. 2

iv, application tasks.

MapReduce Map Tasks, Reduce Tasks & MapReduce Execution:

- Big data processing employs the MapReduce programming model.
- A job means a MapReduce program and each job consists of several units, called MapReduce tasks.
- A software execution framework in MapReduce programming defines the parallel tasks. The tasks give the required result. The Hadoop MapReduce implementation uses Java framework.

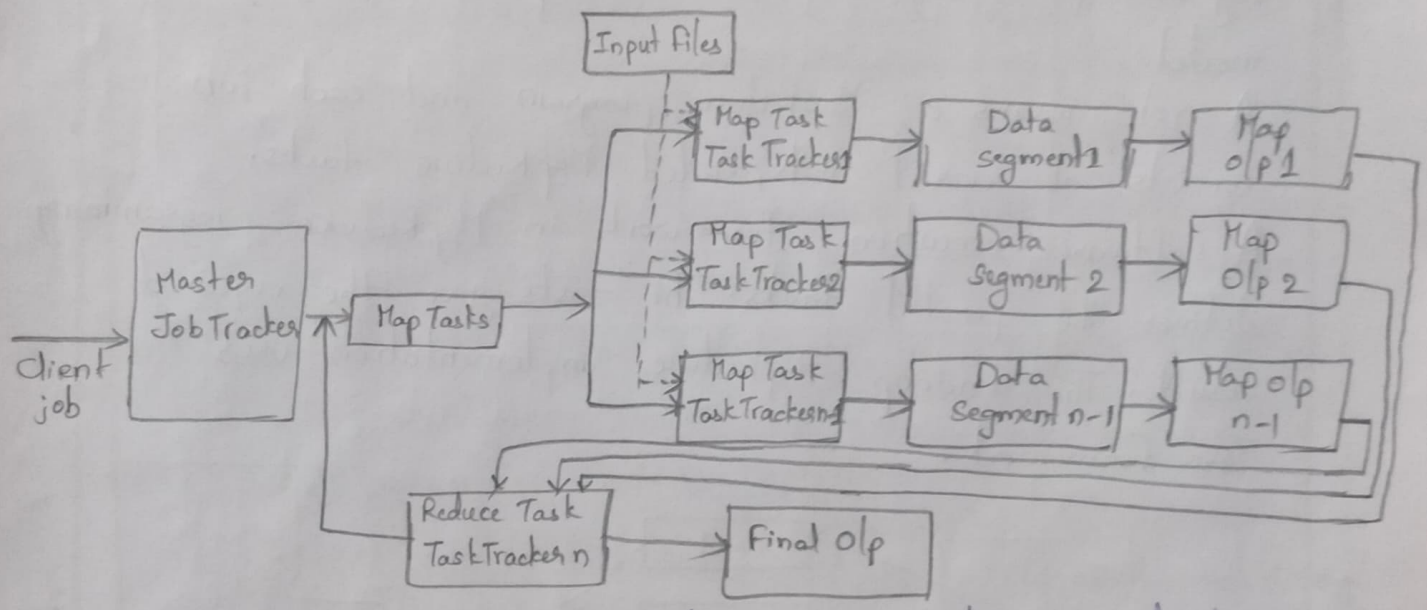


MapReduce Programming model

- The model defines two important tasks, namely Map and Reduce. Map takes input data set as pieces of data and maps them on various nodes for parallel processing. The reduce task which takes the output from the maps as an input and combines those data

pieces into a smaller set of data. A reduce task always run after the map task.

- The input data is in the form of an HDFS file. & the output of the task also gets stored in the HDFS.
- The compute nodes and the storage nodes are the same at a cluster, that is, the Map Reduce program and the HDFS are running on the same set of nodes.
- A user application specifies locations of i/p data and translates into map and reduces functions.



- The above figure shows Map Reduce process when a client submits a job, and the succeeding actions by the Job Tracker and Task Tracker.

i) Map Tasks :

- It means a task that implements a map(), which runs user application codes for each key-value pair (k1, v1). Key v1 is a set of keys. Key k1 maps to a group of data values. Values V1 are a large string which is read from the input file.
- The o/p of map() would be zero & intermediately key value pairs (k2, v2). The value v2 is the information for the transformation operation at the reduce task using aggregation or other reducing functions.

Class Note:

Subject : ....BDT.....

Faculty : .....V. Divya.....

Topic: ....Key Value Pair, Grouping by key, Partitioning, Combiners.

Unit No. : 3
Lecture No.: 5
Link to Session Planner (SP) S.No. .....of SP
Date Conducted:
Page No. 3

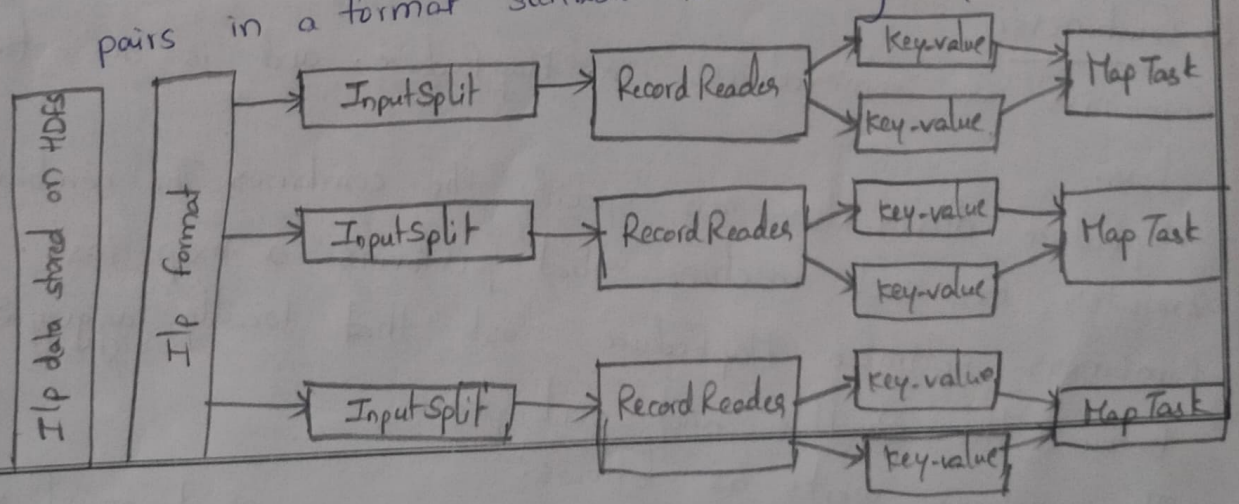
- Reduce Task refers to a task which take the output v2 from the map as an input and combines those data pieces into a smaller set of data using a combiner.
- Mapper performs a function on individual values in a dataset irrespective of the data size of the input.

ii. Key Value Pair:

- Each phase of MapReduce has key-value pairs as input and output. Data should be converted first into key-value pairs before it is passed to the Mapper, as the Mapper only understands key-value pairs of data.
- Key value pairs in Hadoop MapReduce are generated as:

→ Input Split - Defines a logical representation of data and presents a split data for processing at individual map().

→ Record Reader - Communicates with the Input Split and converts the split into records which are in the form of key-value pairs in a format suitable for reading by Mapper.



iii, Grouping by key:

- When a map tasks completes, shuffle process aggregates all the Mapper outputs by grouping the key-values of the Mapper output and the value V2 append in a list of values.
A "GroupBy" operation on intermediat keys creates V2.

Shuffle and Sorting Phase:

- Here, all pairs with the same group key (k2) collect and group together, creating one group for each key.
- So the shuffle op format will be a List of <k2, List (V2)>, Thus a different subset of intermediate key space assigns to each reduce node.

iv, Partitioning:

- The partitioner does the Partitioning. The partitions are the semi-mappers in MapReduce.
- Partitioner is an optional class. MapReduce Driver class can specify the Partitioner. A partition processes the output of map tasks before submitting it to Reducer Tasks.
- Partitioner function executes on each machine that performs a map task. Partitioner is an optimization in MapReduce that allows local partitioning before reduce-task phase.

v, Combiners:

- Combiners are semi reducers in MapReduce. and is an optional class.
- MapReduce driver class can specify the combiner. The combiner() executes on each machine that performs a map task.
- Combiners optimize MapReduce task that locally aggregates before the shuffle and sort phase.
- The combiner works as follows:
  → It doesnot have its own interface and it must implement the interface at reduce().

Class Note:

Subject : .....B.D.T.....

Faculty : .....V.Divya.....

Topic: ...Reduce Tasks, Composing MapReduce for Calculations & Algorithms.

Unit No. : 3
Lecture No.: 6
Link to Session Planner (SP) S.No. .....of SP
Date Conducted:
Page No. 4

→ It operates on each map output key. It must have the same input and output key-value types as the Reducer class

→ It can produce summary information from a large dataset because it replaces the original Map output with fewer records or smaller records.

vi, Reduce Tasks:

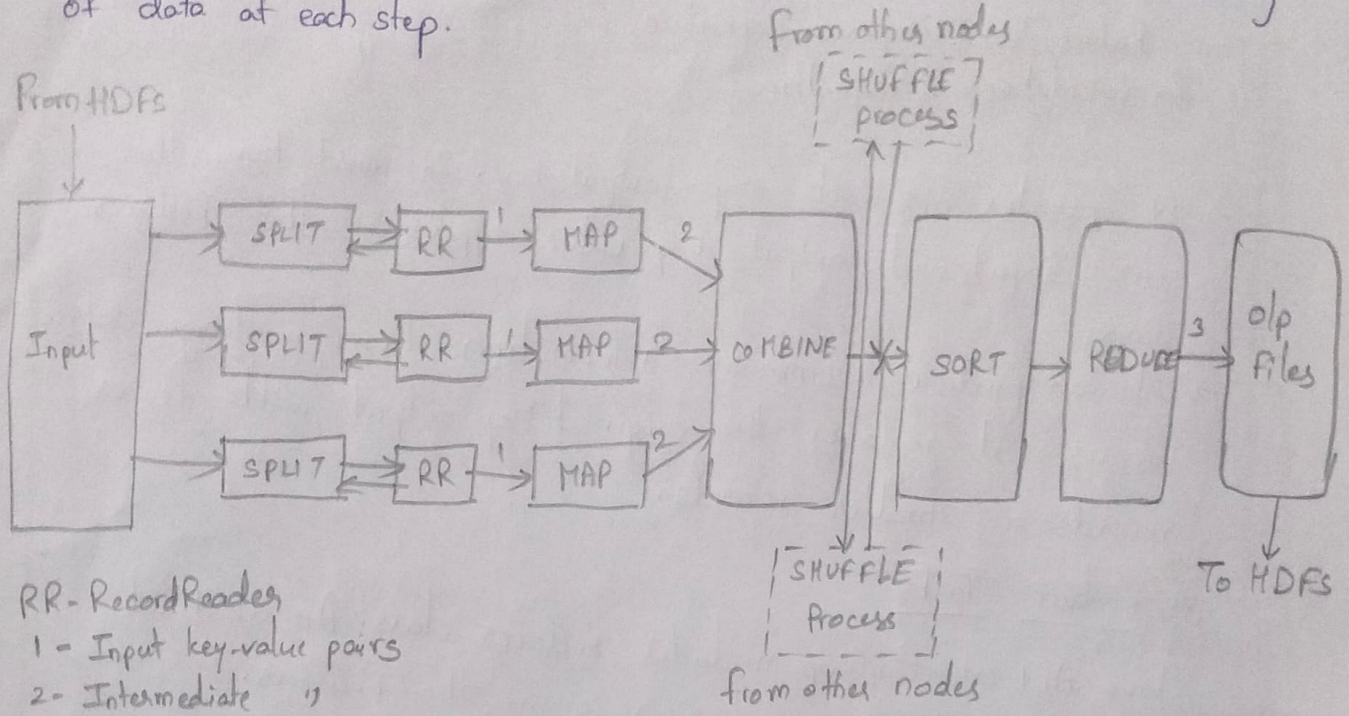
- Java API at Hadoop includes Reducer class. An abstract ~~information~~ function, reduce() is in the Reducer.
- Reduce task implements reduce() that takes the Mapper output, which is grouped by key-values (k2, v2) and applies it in parallel to each group.

Sample code for Reducer class

```
public class ExampleReducer extends Reducer <k2, v2, k3, v3>
{ void reduce (k2 key, Iterable <v2> values, Context context)
        throws
        IOException, InterruptedException
        {...}
}
```

vii) Details of MapReduce Processing steps:

- Execution of MapReduce job does not consider how the distributed processing implements. Rather, the execution involves the formatting of data at each step.



RR - RecordReader
1 - Input key-value pairs
2 - Intermediate "
3 - Final "

Map Reduce Execution steps.

Composing MapReduce for Calculations & Algorithms:

i) Composing MapReduce for Calculations:

- The calculations for various operations compose are:

Counting and summing: Assume that the no. of alerts or messages generated during a specific maintenance activity of vehicles need counting for a month.

Sorting: Many applications need sorted values in a certain order by some rule or process. Mappers just emit all items as values associated with the sorting keys which assemble as a function of items. Reducers combine all emitted parts into a final list.

Finding Distinct values (Counting unique values): Applications such as web log analysis need counting of unique users. Evaluation

Class Note:

Subject : ....B.D.T.....

Faculty : ....V.Divya.....

Topic: ..Composing MapReduce for Calculations, Matrix - vector Multiplication by MapReduce.

Unit No. : 3
Lecture No.: 7
Link to Session
Planner (SP) S.No. .....of SP
Date Conducted:
Page No. 5

is performed for the total no. of unique values in each field for each set of records that belongs to same group.

Collating: It is a way ~~of~~ to collect all items which have the same value of function in one document & file & a way to process items with the same value of function together.

Filtering & Parsing: It collects only those items which satisfy some condition & transform each item into some other representation. which include tasks such as text parsing, value extraction and conversion from one format to another.

Distributed Task Execution: Large computations divide into multiple partitions and combine the ~~large~~ results from all partitions for the final ~~is~~ result.

Graph processing using Iterative Message Passing: Graph is a network of entities and relationships b/w them. A node corresponds to an entity. An edge joining two nodes corresponds to a ~~simulations, numerical analysis and performance testing~~. relationship. Path traversal method processes a graph.

Cross Correlation: This involves calculation using number of tuples where the items co occur in a set of tuples of items. If the total no. of items is N then the total no. of values = N×N.

ii, Matrix - Vector Multiplication by MapReducer

Numbers of applications need multiplication of nxn matrix A with vector B of dimension 'n'. Each element of the product is the element of vector C of dimension n. The elements of C calculate by relation.

$C_i = \sum_{j=1}^{n} a_{ij} b_j$ An example of calculation is given below:

Assume $A = \begin{bmatrix} 1 & 5 & 4 \\ 2 & 1 & 3 \\ 4 & 2 & 1 \end{bmatrix}$ $B = \begin{bmatrix} 4 \\ 1 \\ 3 \end{bmatrix}$

Multiplication $C = A \times B = \begin{bmatrix} 1\times4+5\times1+4\times3 \\ 2\times4+1\times1+3\times3 \\ 4\times4+2\times1+1\times3 \end{bmatrix}$ Hence $C = \begin{bmatrix} 21 \\ 18 \\ 21 \end{bmatrix}$

iii, Relational-Algebra Operations:

→ Selection
→ Projection
→ Union
→ Intersection and difference
→ Natural Join
→ Grouping and Aggregation by MapReduce.

iv, Matrix Multiplication:

→ Matrix multiplication with Cascading of Two MapReduce Steps:

Step	Step description	Matrix A	Matrix B	Matric C = A·B
1.	Name	A	B	C
2.	Specify attributes of (key, value pairs of each element [row no, column no., value])	(I, J, v_a)	(J, K, v_b)	(I, K, v_c)
3.	Specify relations	$R_A = A(I, J, v_a)$	$R_B = A(J, K, v_b)$	$R_C = A(I, K, v_c)$
4.	Consider tuples of A,B,C	(i, j, a_{ij})	(j, k, b_{jk})	(i, k, c_{ik})
5.	find natural Join of R_A and R_B = Matrix elements (a_{ij}, b_{jk})			tuples (i, j, k, v_a, v_b)
6.	Get tuples for finding Product C			four component tuple $(i, j, k, v_a \times v_b)$

Class Note:

Subject : ...B.D.T.....

Faculty : ...V.Divya.....

Topic: .....Hive.....

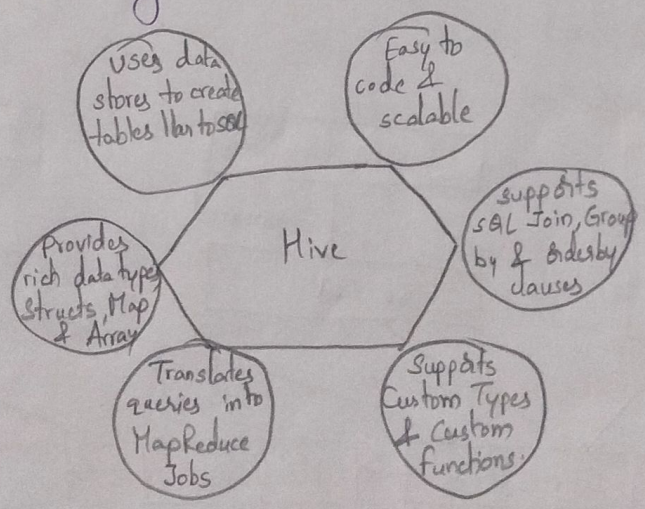
Unit No. : 3
Lecture No.: 8
Link to Session Planner (SP) S.No. .....of SP
Date Conducted:
Page No. 6

<I, K> 3SUM (Va×Vb)

7. Grouping & aggregation of tuples with attributes I & K

HIVE:

- Hive was created by Facebook. & is a data warehousing tool and is also a data store on the top of Hadoop.
- An enterprise uses a datawarehouse as large data repositories that are designed to enable the tracking, managing, & analyzing the data.
- Hive processes structured data & integrates data from multiple heteragenous sources.
-



Main Features of Hive.

Hive characteristics:

- Has the capability to translate queries into MapReduce jobs. This makes Hive scalable, able to handle data warehouse applications, and therefore, suitable for analysis of static data of an extremely large size, where the fast response-time is not a criterion.
- Supports web interfaces as well. Application APIs as well as web-browser, clients, can access the Hive DB server.
- Provides an SQL dialect (Hive Query Language, abbreviated HiveQL & HQL)

Limitations:

Hive is:

- Not a full database. Main disadvantage is that Hive doesnot provide update, alter and deletion of records in the database.
- Not developed for unstructured data.
- Not designed for realtime queries.
- Performs the partition always from the last column.

i, Hive Architecture:

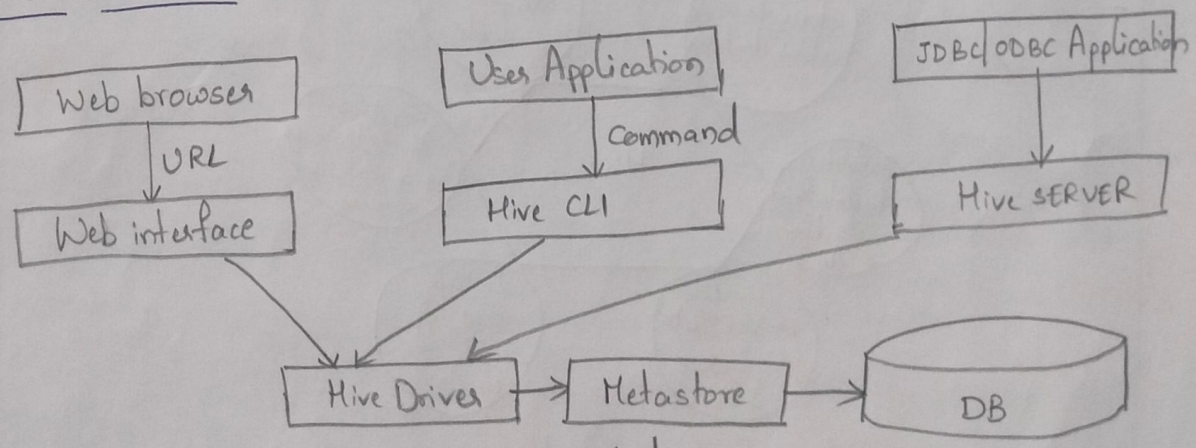


fig: Hive Architecture.

Compoonen

Class Note:

Subject : BDT

Faculty : V.Divya

Topic: Hive

Unit No. : 3
Lecture No.: 9
Link to Session
Planner (SP) S.No. .....of SP
Date Conducted:
Page No. 7

Components of Hive Architecture are:

- Hive Server (Thrift) :- An optional service that allows a remote client to submit requests to Hive and retrieve results. Requests can use a variety of programming languages. Thrift server exposes a very simple client API to execute HiveQL stmts.
- Hive CLI (Command Line Interface): Popular interface to interact with Hive. Hive runs in local mode that uses local storage when running the CLI on a Hadoop cluster instead of HDFS.
- Web Interface:- Hive can be accessed using a web browser as well. This requires a HWI Server running on some designated code. The URL http:// hadoop:-port no> / hwi command can be used to access Hive through the web.
- Metastore:- It is the system catalog. All other components of Hive interact with the Metastore. It stores the schema or metadeta of tables, databases, columns in a table, their datatypes and HDFS mapping.

ii) Hive Installation:

- Hive can be installed on Windows 10, Ubuntu 16.04 and MySQL. It requires three slw packages.
  - → Java Development Kit for Java compiler (JavaC) and interpreter
  - → Hadoop
  - → Compatible version of Hive with Java. Hive 1.2 onward supports Java 1.7 or newer.
- Steps for installation of Hive in a Linux based OS are as follows:
  - → Install Javac and Java from Oracle Java download site. Download jdk 7 or a later version and extract the compressed file. All users can access Java by MakeJava available to all users. The user has to move it to the location "/usr/local/" using the required commands.
  - → Set the path by the commands for jdk 1.7.0_71, export JAVA_HOME = /usr/local/jdk1.7.0_71, export PATH = $ PATH : $JAVA_HOME/bin.
  - → Install Hadoop http://apache.claz.org/hadoop/common/hadoop-2.4.1/
  - → Make shared HADOOP, MAPRED, COMMON, HDFS and all related files, configure HADOOP and set property such as replication parameter.
  - → Name the yarn.nodemanager.aux-services. Assign value to mapreduce_shuffle. Set namenode and datanode paths.
  - → Download http://apache.petsads.us/hive/hive-0.14.0/. Use 'ls' command to verify the files $ tar zxvf apache-hive-0.14.0-bin.tar.gz, $ ls
  - → Use an external database server. Configure metastore for server.

Class Note:

Subject: BDT

Faculty: V. Divya

Topic: Hive

Unit No.: 3
Lecture No.: 9
Link to Session Planner (SP) S.No. ....of SP
Date Conducted:
Page No. 8

iii. Comparison with RDBMS (Traditional Database):

- Hive is a DB system which defines databases and tables.
- Hive analyzes structured data in DB. Hive has certain differences with RDBMS.

Characteristics	Hive	RDBMS
- Record level queries	No update & delete	Insert, update & delete
- Transaction support	No	Yes.
- Latency	Minutes or more	In fractions of a sec.
- Data size	Petabytes	Terabytes.
- Data per query	Petabytes	Gigabytes.
- Query language	HiveQL	SQL
- Support JDBC/ODBC	Limited	Full.

iv. Hive Datatypes and file formats:

- Hive defines various primitive, complex, string, date/time, collection datatypes and file formats for handling and storing different data formats.

Datatype Name	Description
TINYINT	1 byte signed integer, Postfix letter is Y.
SMALLINT	2 byte signed integer, Postfix letter is S.
INT	4 byte signed integer.
BIGINT	8 byte signed integer. Postfix letter is L.
FLOAT	4 byte single-precision floating point number.
DOUBLE	8 byte double-precision floating point number.
BOOLEAN	True or false.
DATE	YYYY-MM-DD format
VARCHAR	1 to 65355 bytes. Use single quotes (") & double quotes (" ")
CHAR	255 bytes
DECIMAL	Used for representing immutable arbitrary precision. DECIMAL (Precision, scale) format.
UNION	Collection of heterogeneous data types. Create union
NULL	Missing values representation.
STRUCT	Similar to 'C' struct, a collection of fields of different data types. An access to field uses dot notation. Ex: struct ('a', 'b')
MAP	A collection of key-value pairs. Fields access using [ ] notation.
ARRAY	Ordered sequence of same types. Accesses to fields using array index

Class Note:

Subject : BDT

Faculty : V. Divya

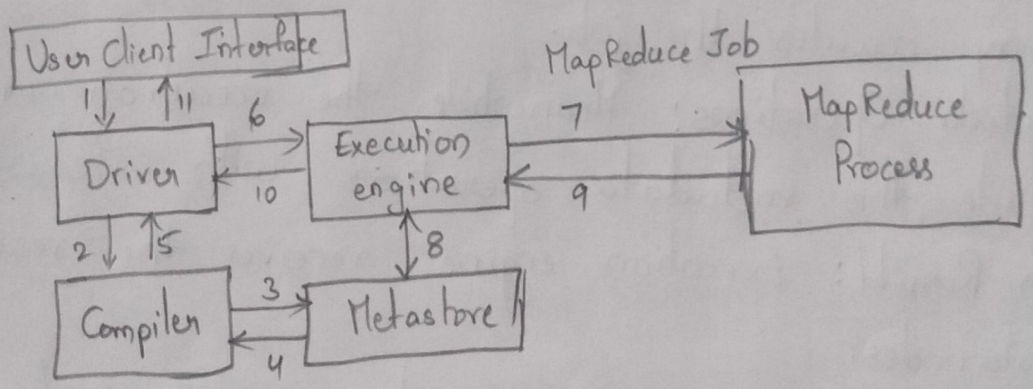
Topic: Hive

Unit No. : 3
Lecture No.: 9
Link to Session Planner (SP) S.No. ....of SP
Date Conducted:
Page No. 9

v, Hive Data Model:

Name	Description
Database	Namespace for tables.
Tables	Similar to tables in RDBMS. Support filter, projection, join and union operations. The table data stores in a directory in HDFS.
Partitions	Table can have one or more partition keys that tell how the data stores.
Buckets	Data in each partition further divides into buckets based on hash of a column in the table. Stored as a file in the partition directory.

vi, Hive Integration and Workflow steps:



Steps 1 to 11 are as follows:

Step No:	OPERATION

1. Execute Query: Hive interface (CLI or Web Interface) sends a query to Database Driver to execute the query.
2. Get plan: Driver sends the query to query compiler that parses the query to check the syntax and query plan or the requirement of the query.
3. Get Metadata: Compiler sends metadata request to Metastore (of any database, such as MySQL).
4. Send Metadata: Metastore sends metadata as a response to compiler.
5. Send Plan: Compiler checks the requirement and resends the plan to driver. The parsing and compiling of the query is complete at this place.
6. Execute Plan: 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 DataNode. Then, the query executes the job.
8. Metadata Operations: Meanwhile the execution engine can execute the metadata operations with Metastore.
9. Fetch Result: Execution engine receives the results from Datanodes.
10. Send Results: ~~Driver sen~~ Execution engine sends the result to Driver.
11. Send Results: Driver sends the results to Hive interfaces.

Class Note:

Subject : BDT

Faculty : V. Divya

Topic: Hive.

Unit No. : 3
Lecture No.: 9
Link to Session Planner (SP) S.No. .....of SP
Date Conducted:

vii, Hive Built-in functions:

- Hive supports a number of builtin functions.
- Below table gives the return types, syntax and descriptions of examples of these functions.

Return Type	~~Syntax~~ Syntax	Description
BIGINT	- round(double a).	Returns the rounded BIGINT (8 byte integer) value of 8 byte double precision floating point number a.
BIGINT	- floor(double a)	- Returns the maximum BIGINT value that is equal to or less than the double.
BIGINT	- ceil(double a)	- Returns the minimum BIGINT value that is equal to or greater than the double.
double	- rand(), rand(int seed)	- Returns a random number (double) that distributes uniformly from 0 to 1 and that changes in each row. Integer seed ensured that random number sequence is deterministic.

Return type	Syntax	Description
string	concate (string str1, string str2, ---)	Returns the string resulting from condatenating str1 with str2, ---
String	substr (string str, int start)	Returns the substring of str starting from a start position till the end of string str.
string	substr (string str, int start, int length)	Returns the substring of str starting from the start position with the given length.
string	~~substr~~ upper (string str), ucase (string str)	Returns the string resulting from converting all characters of str to upper case.
string	lower (string str), lcase (string str)	Returns the string resulting from converting all characters of str to lower case.
string	trim (string str)	Returns the string resulting from trimming spaces from both ends, trim ('123456') ~~fro~~ returns '123456'
string	ltrim (string str); rtrim (string str).	Returns the string resulting from trimming spaces (only one end, left or right hand side or right-hand side spaces trimmed). ltrim ('12A3456') returns '12A3456' and rtrim ('12A3456') returns '12A3456'.
string	rtrim (string str)	Returns the string resulting from trimming spaces from the end of str.
int	year (string date)	Returns the year part of a date or a timestamp string.
int	month (string date)	Returns the month part of a date or a timestamp string.
int	day (string date)	Returns the day part of a date or a timestamp string.

Class Note:

Subject : BDT

Faculty : V. Divya

Topic: Hive QL

Unit No. : 3
Lecture No.: 10
Link to Session
Planner (SP) S.No. .....of SP
Date Conducted:
Page No. 11

Hive QL :-

- Hive Query Language is for querying the large datasets which reside in the HDFS environment.
- Hive QL script commands enable data definition, data manipulation and query processing.
- Hive QL supports a large base of SQL users who are using SQL to extract information from data warehouse.

HiveQL Process Engine - Hive QL is similar to SQL for querying on schema information at Metastore. It is one of the replacements of traditional approach for MapReduce program. Instead of writing MapReduce program in Java, we can write a query for MapReduce job and process it.

Execution Engine - The bridge between HiveQL process engine and MapReduce is Hive Execution Engine. Execution engine processes the query and generates results same as MapReduce results. It uses the flavor of MapReduce.

i) Hive QL DDL :-

- Hive QL database commands for data definition for DBs and tables are:

CREATE DATABASE,

SHOW DATABASE (list of all DBs),

CREATE SCHEMA,

CREATE TABLE.

Following are HiveQL commands which create a table:

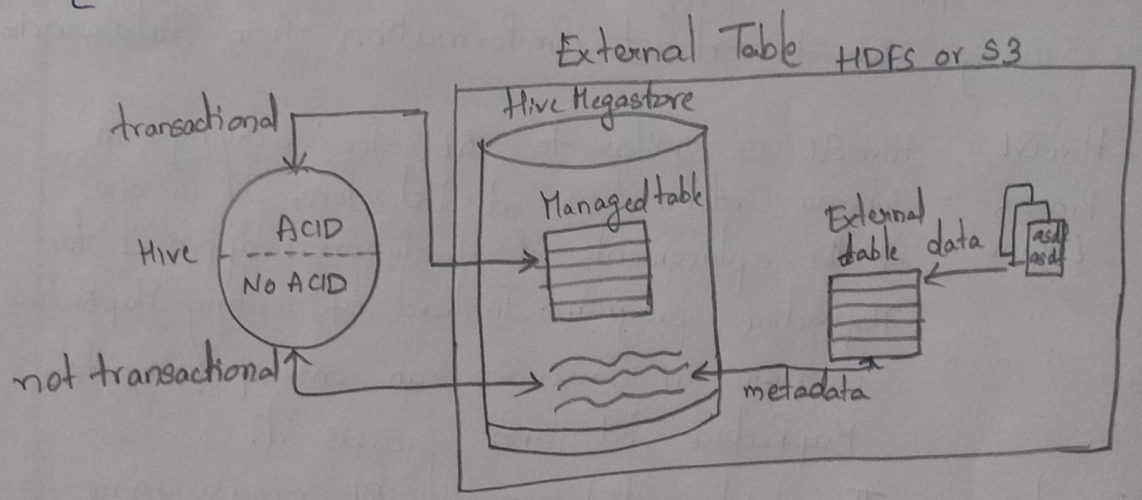
CREATE [TEMPORARY] [EXTERNAL] TABLE [IF NOT EXISTS] [<database name> < table name>

[(< column name > <data type> [COMMENT <column comment>], ----)]

[COMMENT <table comment>]

[ROW FORMAT <row format>]

[STORED AS <file format>]



Example: on Creating a Table

- How do you create a table toy_products with the following fields.

Field	Datatype
Product Category -	string
ProductId -	int
ProductName -	string
ProductPrice -	float.

Class Note:

Subject : BDT

Faculty : V. Divya

Topic: Hive QL

Unit No. : 3
Lecture No.: 10
Link to Session Planner (SP) S.No. ..... of SP
Date Conducted:
Page No. 12

Solution:

```
CREATE TABLE IF NOT EXISTS toy-products (ProductCategory
string, ProductId int, ProductName string, ProductPrice float)
COMMENT 'Toy details'
ROW FORMAT DELIMITED
FIELDS TERMINATED BY '\t'
LINES TERMINATED BY '\n'
STORED AS TEXTFILE;
```

The option IF NOT EXISTS, Hive ignores the statement in case the table already exists.

Creating a database:-

- A command is

```
CREATE DATABASE|SCHEMA [IF NOT EXISTS] <database
name>;
```

IF NOT EXISTS is an optional clause. This clause notifies the user that a database with the same name already exists. SCHEMA can be also created in place of DATABASE using this command.

A command is written to get the list of all existing databases.

```
SHOW DATABASES;
```

A command is written to delete an existing database.

DROP (DATABASE|SCHEMA) [IF EXISTS] <database name>
[RESTRICT|CASCADE];

Examples

Usages of databases commands for CREATE, SHOW and DROP

CREATE DATABASE IF NOT EXISTS toys_company DB;

SHOW DATABASES;

default, toys_company DB

Drop Database toys_companyDB.

ii, HiveQL DML:

HiveQL commands for data manipulation are:

USE <database name>,

DROP DATABASE,

DROP SCHEMA,

ALTER TABLE,

DROP TABLE, and

LOAD DATA.

Loading Data into HIVE DB:

The following is a command for inserting (loading) data into Hive DBs.

LOAD DATA [LOCAL] INPATH '<file path>' [OVERWRITE]
INTO TABLE <table name> [PARTITION (part col1=val1,
partcol2 = val2 ....)].

LOCAL is an identifier to specify the local path. It is optional. OVERWRITE is optional to overwrite the data in the table, PARTITION is optional. val1 is value assigned to

Class Note:

Subject : BDT

Faculty : V. Divya

Topic: Hive QL

Unit No. : 3
Lecture No.: 11
Link to Session
Planner (SP) S.No. .....of SP
Date Conducted:

partition column 1 (part col 1) and val2 is value assigned to partition column2 (partcol2).

iii, HiveQL for querying the data:-

- For any data analysis application there is a need for partitioning and storing the data.
- A data warehouse should have a large no. of partitions where the tables, files and databases store.
- Querying then requires sorting, aggregating and joining functions.

```
SELECT [ALL | DISTINCT] <select expression>, <select
    expression>, ----
FROM <table name>
[WHERE <where condition>]
[GROUP BY <column List>]
[HAVING <having condition>]
[CLUSTER BY <column List> | [DISTRIBUTE BY <column
    LISE>] [SORT BY <column List>]]
[LIMIT number];
```

a, Partitioning:

- Table partitioning refers to divide the table data into some parts based on the values of particular set of columns.
- Hive organizes tables into partitions. and Partition makes querying easy and fast.
- This is because SELECT is then from the smaller number of column fields.
- The following example explains the concept of partitioning, columnar and file records formats:

→ Table partitioning:

~~Create a~~ table with Partition using command:

```
CREATE [EXTERNAL] TABLE <table name> (<column name 1>
  <data type 1>,----)
PARTITIONED BY (<column name n><data type n> [COMMENT
  <column comment>], ....);
```

→ Renaming the partition:

Rename a partition in the existing table using the following command:

```
ALTER TABLE <table name> PARTITION partition_spec
  RENAME TO PARTITION partition_spec;
```

→ Add a partition to the existing table:

Add a partition in the existing table using the following command:

```
ALTER TABLE <table name> ADD [IF NOT EXISTS] PARTITION
  partition_spec.
[LOCATION 'location1'] partition_spec [LOCATION 'location2'] ...;
```

Class Note:

Subject : BDT

Faculty : V. Divya

Topic: HiveQL

Unit No. : 3
Lecture No.: 11
Link to Session Planner (SP) S.No. .....of SP
Date Conducted:
Page No. 14

partition_spec : (p_column " " p_col_value, p_column = p_col_value, ...)

→ Drop a partition:

Drop a partition in existing table using the following command:

ALTER TABLE <table name> DROP [IF EXISTS] PARTITION partition_spec, PARTITION partition_spec;

→ Partitioning : Query fast processing:

Advantages of Partition:

- Distributes execution load horizontally.
- Query response time becomes faster when processing a small part of the data instead of searching the entire dataset.

Limitations of Partition:

- Creating a large no. of partitions in a table leads to a large no. of files and directories in HDFS, which is an overhead to NameNode, since it must keep all metadata for the file system in memory only.

- Partitions may optimize some queries based on Where clauses, but they may be less responsive for other important queries on grouping clauses.
- A large no. of partitions will lead to a large no. of tasks (which will run in separte JVM) in each MapReduce job, thus creating a lot of overhead in maintaining JVM start up and tear down (A separte task will be used for each file). The overhead of JVM startup and tear down can exceed the actual processing time in the worst case.

b, Bucketing:

- A partition itself may have a large no. of columns when tables are very large. Tables or partitions can be subdivided into buckets.
- Division is based on the hash of a column in table. CLUSTERED BY clause divides a table into buckets.

c, Views:

- Views provide ease of programming. Complex queries simplify using reusable Views.
- A view provisions the following:
  - → Saves the query and reduces the query complexity.
  - → Use a View like a table but a View does not store data like a table
  - → Hides the complexity by dividing the query into smaller, more manageable pieces.
  - → The Hive executes the View and then the planner combines the information in View definition with remaining actions on quer

Class Note:

Subject : BDT

Faculty : V. Divya

Topic: Hive QL, PIG

Unit No. : 3
Lecture No.: 12
Link to Session Planner (SP) S.No. .....of SP
Date Conducted:
Page No. 15

iv, Aggregation:

Hive supports the following built in aggregation functions. The usage of these functions is same as SQL aggregate functions.

Return type	Syntax	Description
BIGINT	count(*), count(expr)	Returns the total no. of retrieved rows.
DOUBLE	sum(col), sum(DISTINCT col).	Returns the sum of elements in the group or the sum of distinct values of column in the group.
DOUBLE	avg(col), avg(DISTINCT col)	Returns the average of elements in the group or the average of distinct values of the column in group.
DOUBLE	min(col)	Returns the minimum value of column in the group.
DOUBLE	max(col)	Returns the maximum value of column in the group.

v, Join:

- A JOIN clause combines columns of two or more tables, based on a relation between them.
- HiveQL Join is more or less similar to SQL JOINS.

vi, Group By clause:

GROUP BY, HAVING, ORDER BY DISTRIBUTE BY, CLUSTER BY are HiveQL clauses.

Pig:

Apache developed Pig, which:

- is an abstraction over MapReduce.
- is an execution framework for parallel processing.
- reduces the complexities of writing a MapReduce program.
- is a high level dataflow language. Dataflow language means that a Pig operation node takes the inputs and generates the output for the next node.
- is mostly used in HDFS environment.
- performs data manipulation operations at files at data nodes in Hadoop.

Applications of Pig:

- Analyzing large datasets.
- Executing tasks involving adhoc processing.
- Processing large data sources such as web logs, and streaming online data.
- Data processing for search platforms. Pig processes different types of data.

Class Note:

Subject : .....BDT.....

Faculty : .....V.Divya.....

Topic: .....Pig.....

Unit No. : 3
Lecture No.: 12
Link to Session
Planner (SP) S.No. .....of SP
Date Conducted:
Page No. 16.

- Processing time sensitive data loads; data extracts and analyzes quickly.

Features:

1. Apache PIG helps programmers write complex data transformations using scripts. Pig Latin Language is very similar to SQL and possess a rich set of built-in operators, such as groups, join, filter, limit, order by, parallel, sort and split.
2. Creates user defined functions (UDFs) to write custom functions which are not available in Pig.
3. Process any kind of data, structured, semi-structured or unstructured data, coming from various sources.
4. Reduces the length of codes using multi-query approach. Pig code of 10 lines is equal to MapReduce code of 200 lines. Thus, the processing is very fast.
5. Extracts the data, performs operations on that data and dumps the data in the required format in HDFS. The operation is called ETL
6. Performs automatic optimization of tasks before execution
7. Programmers and developers can concentrate on whole

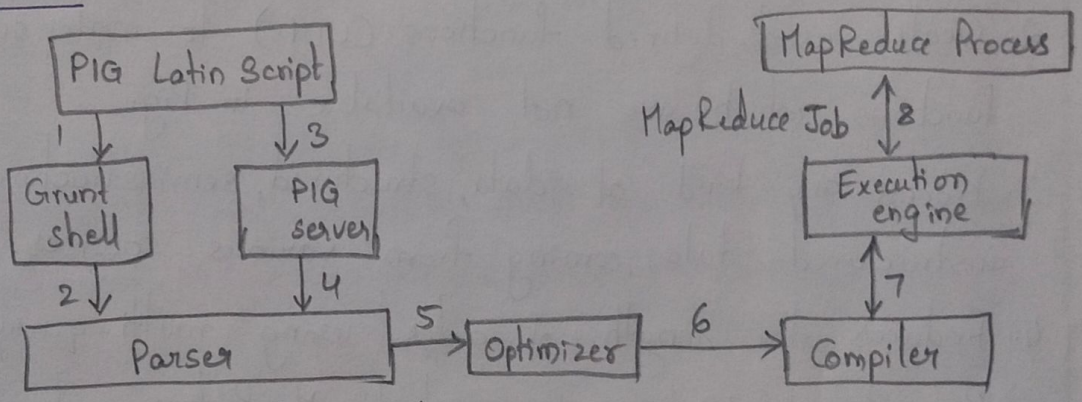
operation without a need to create mapper and reducer tasks separately.

8. Reads the input data files from HDFS or the data files from other sources such as local file system, stores the intermediate data and writes back the output in HDFS.

9. Pig characteristics are data reading, processing, programming the UDFs in multiple languages and programming multiple queries by fewer codes. This causes fast processing.

10. Pig derives guidance from four philosophies, live anywhere, take anything, domestic and run as if flying. This justifies the name Pig, as the animal pig also has these characteristics.

Pig Architecture:



The three ways to execute scripts are:

1. Grunt shell: An interactive shell of Pig that executes the scripts.
2. Script File: Pig commands written in a script file that execute at Pig sewer.
3. Embedded Script: Create UDFs for the functions unavailable as Pig built-in operators. UDF can be in other programming languages. UDFs can embed in Pig Latin script file.

Class Note:

Subject : BDT

Faculty : V. Divya

Topic: Pig

Unit No. : 3
Lecture No.: 12
Link to Session Planner (SP) S.No. ....of SP
Date Conducted:
Page No. 17

1. Parser:
- A parser handles Pig scripts after passing through Grunt or Pig server.
- Parser performs type checking and checks the script syntax.
- Output is a Directed Acyclic Graph (DAG).
- DAG represents the Pig Latin statements and logical operators. Node represent the logical operators. Edge b/w sequentially traversed nodes represent the data flows.

2. Optimizer:-
- DAG is submitted to logical optimizer.
- The optimization activities, such as split, merge, transform and reorder operators execute in this phase.
- Optimization is an automatic feature.
- Optimizer reduces the amount of data in the pipeline at any instant of time, while processing the extracted data.
- It executes certain functions for carrying out this task, as explained as follows:

→ PushUpfilter: If there are multiple conditions in the filter and the filter can be split, Pig splits the conditions and

pushes up each condition separately. Selecting these conditions at an early stage helps in reducing the no. of records remaining in the pipeline.

→ PushDown for Each Flatten: Applying flatten, which produces a cross product between a complex type such as tuple, bag or other fields in the record, as late as possible in the plan. This keeps no. of records low in pipeline.

→ ColumnPruner: Omitts never used columns or the one no longer needed, reducing the size of record. This can be applied after each operator, so that the fields can be pruned as aggresively as possible.

→ MapKey Pruner: Omitts never used map keys, reducing the size of record

→ Limit Optimizer: If the limit operator is immediately applied after load or sort operator, Pig converts the load or sort into a limit sensitive implementation, which doesnot require processing the whole dataset.

3. Compiler: The compiler compiles after the optimization process. The optimized codes are a series of MapReduce jobs.

4. Execution Engine: Finally, the MapReduce jobs submit for execution to the engine. The MapReduce jobs execute and it outputs the final result.

i) Apache Pig-Grunt Shell :-

- Main use of Grunt shell is for writing Pig Latin scripts. Any shell command invokes using sh and ls. Syntax of sh command is:

grunt> sh shell command parameters

Syntax of ls command: grunt> sh ls

Class Note:

Subject : .....B.D.T.....

Faculty : .....V. Divya.....

Topic: .....Pig.....

Unit No. : 3
Lecture No.: 12
Link to Session
Planner (SP) S.No. .....of SP
Date Conducted:

- Grunt shell includes a set of quality utility commands. Included utility commands are clear, help, history, quit & set.
- Shell includes commands such as exec, kill and run to control Pig from Grunt shell.

ii) Installing Pig:-

- Download the latest version from - https://pig.apache.org/
- Download the tar files and create a Pig directory

```
$ cd Downloads/
$ tar zxvf pig-0.15.0-src.tar.gz
$ tar zxvf pig-0.15.0.tar.gz
$ mv pig-0.15.0-src.tar.gz/* /home/Hadoop/Pig/
```

- Configure the Pig

```
export PIG HOME /home/Hadoop/Pig
export PATH = $PATH: /home /Hadoop /Pig/bin.
export PIG CLASSPATH $HADOOP HOME/conf.
```

iii) Pig Latin Data Model:-

- Pig Latin supports primitive data types which are atomic or scalar data types.

Datatype	Description	Example
bag	Collection of tuples	{(1,1),(2,4)}
tuple	Ordered set of fields	(1,1)
map(data map)	Set of key-value pairs	[Numberst]
int	signed 32-bit integer	10
long	Signed 64-bit integer	10L & 10l
float	32-bit floating point	22.7F & 22.7f
double	64-bit floating point	3.4 & 3.4e2 & 3.4E2
char array	char[], character array	data analytics
byte array	BLOB (Byte array)	ff00

iv, Pig Latin and developing Pig Latin scripts:-

- Pig Latin enables developing the scripts for data analysis. A no. operators in Pig Latin help to develop their own functions for reading, writing and processing data.
- Pig Latin programs execute in the Pig runtime environment.
- Stmts in Pig Latin:-
  - → Basic constructs to process the data.
  - → Include schemas and expressions.
  - → End with a semicolon
  - → LOAD stmt reads the data from file system, DUMP displays the result and STORE stores the result.
  - → Single line comments begin with -- and multiline begin with /* and end with */
  - → Keywords are not case-sensitive
  - → Function names, relations and paths are case-sensitive.

Class Note:

Subject : BDT

Faculty : V. Divya

Topic: Pig

Unit No. : 3
Lecture No.: 12
Link to Session
Planner (SP) S.No. ..... of SP
Date Conducted:
Page No. 19

a, Apache Pig Execution:

- Pig execution modes: Local mode:-
  All data files install and run from a local host using the local file system. Local mode is mostly used for testing purpose.
  Command: pig -x local
- MapReduce mode: All the data files load a process that exists in the HDFS. A MapReduce job invokes in the backend to perform a particular operation on the data that exists in HDFS when a Pig Latin statement executes to process the data.
  Command: pig -x mapreduce or pig.
- Pig Latin Script Execution modes:

→ Interactive mode: using the Grunt shell.

→ Batch mode: writing the Pig Latin script in a single file with '.pig extension.

→ Embedded mode: Defining UDFs in programming languages such as Java, and using them in script.

b, Commands:

→ pig-help: to get list of pig commands

→ pig-version: to get version of pig.

→ pig - to start the Grunt shell, write the command.

→ LOAD - first step to a dataflow is to specify the input. Load stmt in Pig Latin loads the data from PigStorage.

→ store - provides the store stmt for writing the processed data after the processing is complete.

STORE processed into '/PigDemo/Data/Output/Processed';

→ Dump - provides dump command to see the processed data on the screen. DUMP processed;

→ Foreach - gives a simple way to apply transformations based on columns

→ Filter - gives a simple way to select tuples from a relation based on some specified conditions.

→ Group - collects records with same key. There is no direct connection b/w group and aggregate function in Pig Latin unlike SQL.

→ Orderby - sorts the data based on a specific field value, producing a total order of o/p data.

→ Distinct - removes duplicate tuples. It works only on entire tuples, not on individual fields.

→ Join - joins two or more relations based on values in the common field

→ Limit - gets the limited number of results.

→ Parallel - is for parallel data processing.

→ AVG - Compute the average of numeric values in a single column bag.

→ SUM - compute the sum of numeric values in a single column bag.

→ MAX - get the max. of numeric values or chararrays in a single column bag.

→ MIN - get the min. of numeric values or chararrays in a single column bag.

→ COUNT and COUNT_STAR - count the number of tuples in a bag.

→ CONCAT - concatenate two fields.

→ DIFF - compare two fields in a tuple.

→ IsEmpty - check if a bag or map is empty.

→ SIZE - compute the number of elements based on data type.

→ TOKENSIZE - split a string and output a bag of words.