**Pig: Pig is a data flow language**

Pig has two components:

* **Pig Latin:** Data flow language for PIG
* **Environment**: pig-x.y.z.tar. It can be set to run in Local and MapR mode. In local mode it runs on a single JVM, while in MR mode it runs on Hadoop cluster. Pig can run on TEZ too.

Pig Latin can be coded in three ways:

* **Scripts**: Bunch of commands for specific task. Save in file and run using pig command.
* **Grunt Mode**: Interactive Grunt shell prompt, type pig at master node CLI to start it.
* **Embedded Mode:** Embedded in java program

Pig can be run in following modes:

* **Local Mode:** *pig –x local*
* **MapReduce mode:** Pig need to know location of cluster NameNode and JobTracker *$ pig or*

*$ pig –x mapreduce*

MR takes time and effort in coding Pig is an alternative to it, and it is quite extensible using UDF’s. It is very efficient in case of complex join operation.

MR are highly optimized, so if we need to write a reporting job on larger data set and execute it frequently then Map reduce is ideal choice.

Starting pig: Type pig at command line from master node and grunt will start.

**ratings** = **Load** ‘/usr/maria\_dev/ml-100k/u.data’ AS (userID:int, rating:int, ratingTime:int);

**metadata**= **Load** ‘/usr/maria\_dev/ml-100k/u.item’ **USING PigStorage (‘|’)** AS (movieID: int, movieTitle: chararray, releaseDate: chararray, videoRelease: chararray, imdbLink:chararray);

**DUMP** metadata**;**

**nameLookup** = FOREACH metadata GENERATE movieID, movieTitle, ToUnixTime(ToDate(releaseDate, ‘dd-MMM-yyyy’)) AS releaseTime;

**ratingsByMovie =** GROUP ratings BY movieID;

**DUMP** ratingsByMovie**;**

**avgRatings = FOREACH** ratingsByMovieGENERATE group AS movieID, AVG(ratings.rating) AS avgRating;

**DUMP avgRatings;**

**FiveSatrMovies = FILTER** avgRatings BY avgRating > 4.0;

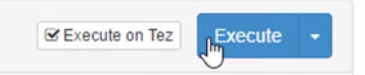
**DESCRIBE ratings;**

**DESCRIBE ratingsByMovie;**

**DESCRIBE avgRatings;**

**fiveStarWithData** = **JOIN** fiveStarMovies BY movieID, nameLookup BY movieID;

**oldestFiveStarMovies = ORDER fiveStarWithData BY nameLookup :: releaseTime;**



We can run above script form Ambari new script option. PIG can run faster on TEZ because TEZ uses DAG to execute it in best order.

**Important PIG Latin commands:**

**LOAD:** To read data

**STORE:** STORE ratings INTO ‘outRatings’ USING PigStore(‘:’);

**DUMP:** To display data to console

**FILTER:** Filtering

**DISTINCT:**

**FOREACH/GENERATE:**

**MAPREDUCE:**

**STREAM:** Stream result to processes

**SAMPLE:** create random sample

**JOIN, COGROUP, GROUP**

**CROSS:** Cartesian product

**CUBE:** Complex than CROSS

**ORDER**

**RANK**

**LIMIT**

**UNION, SPLIT**

**DESCRIBE, EXPLAIN, ILLUSTRATE**

**Relations, Tuple, Bag and field**

Pig Latin statements work with relations. A relation is a bag (more specifically, an outer bag). In above example ratings, metadata etc. are relations

A bag is a collection of tuples and a tuple is an ordered set of fields. A field is a piece of data.

Above code creates a relation named ratings with a given schema **USING PigStorage (‘|’)** is used to define delimiter. **FOREACH/ GENERATE**

Is used to create a relation from another relation.

UDF’s, Loaders and other functions.

**REGISTER: Registers jar file with UDF for importing.**

**DEFINE: Assigns names to function to be used in Pig scripts.**

**IMPORT: Used for importing macros.**

**AVG, CONCAT, COUNT, MAX, MIN, SIZE, SUM.**

* **PigStorage**
* **TextLoader**
* **JsonLoader**
* **AvroStorage**
* **ParaquetLoader**
* **OrcStorage**
* **HBaseStorage**

All UDF should extend a Filter function and has to contain a method called exec, which contains a Tuple.

public class IsOfAge extends FilterFunc {

@Override

publicBoolean exec(Tuple tuple) throwsIOException {

if(tuple == null|| tuple.size() == 0) {

returnfalse;

}

try{

Object object= tuple.get(0);

if(object == null) {

returnfalse;

}

inti = (Integer) object;

if(i == 18 || i == 19 || i == 21 || i == 23 || i == 27) {

returntrue;

} else{

returnfalse;

}

} catch(ExecExceptione) {

thrownewIOException(e);

}

}

}

Once a UDF is created, the following command has to be used to register the JAR file.

register myudf.jar;

X = filter A by IsOfAge(age);

**Pig Latin Example**: Get the list of pages visited by users whose age is between 20 and 25 years.

1. **users** = load 'users' as (name, age);
2. **users\_18to\_25** = filter users by age > 20 and age< 25;
3. **page\_views** = load 'pages' as (user, url);
4. **page\_views\_u18to25** = join users\_18to25 by name, page views by user;

What is DAG?

UDF vs UDAF?

<https://acadgild.com/blog/writing-udaf-in-pig>

How to distinguish tuple from Bag?

Bag will be inside curly braces and tuple within bracket.

COUNT VS COUNT\_STAR functions?

What is Map in Pig?

How to run pig Latin in script mode?

How to run pig in interactive mode?

Where pig is best suited?

Processing weblogs

Quick prototyping

Where to avoid Pig?

Unstructured data as Audio, Video etc.

How to use STREAM in pig?

Can we run HDFS commands in grunt shell?

Yes we can run HDFS commands in grunt shell.

Can pig be run without HDFS? Yes

**Hive**

Hive was developed at Facebook and is known for its SQL like language. Hive can be written in

* **Script mode:** hive -f
* **Interactive mode:** hive -e
* **Embedded mode:** embedded to java code

From Ambari: http://127.0.0.1:8888 menu click on Hive View to interact with hive.

Schema on Read.

Meta-store

**Managed table**

**External Table:** Late Binding of data (Lazy)

**Hive Warehouse**

**Partitioning and Buckets**

**Storage Format in Hive**

**Row Format**

**File Format:** Sequence Files and RC Files