JUNAID GIRKAR

60004190057

TE COMPS A4

**EXPERIMENT 6**

**AIM**: Execute PIG built in commands and run pig scripts on HDFS

THEORY:

While it comes to analyze large sets of data, as well as to represent them as data flows, we use Apache Pig. It is nothing but an abstraction over MapReduce.

Apart from its Introduction, it also includes History, need, its Architecture as well as its Features. Moreover, we will see, some Comparisons like Pig Vs Hive, Apache Pig Vs SQL and Hadoop Pig Vs MapReduce.

**What is Hadoop Pig?**

Hadoop Pig is nothing but an abstraction over MapReduce. While it comes to analyze large sets of data, as well as to represent them as data flows, we use Apache Pig. Generally, we use it with Hadoop. By using Pig, we can perform all the data manipulation operations in Hadoop.

In addition, Pig offers a high-level language to write data analysis programs which we call as Pig Latin. One of the major advantages of this language is, it offers several operators.

Through them, programmers can develop their own functions for reading, writing, and processing data.  
It has following key properties such as:

* **Ease of programming**

Basically, when all the complex tasks comprised of multiple interrelated data transformations are explicitly encoded as data flow sequences, that makes them easy to write, understand, and maintain.

* **Optimization opportunities**

It allows users to focus on semantics rather than efficiency, to optimize their execution automatically, in which tasks are encoded permits the system.

* **Extensibility**

In order to do special-purpose processing, users can create their own functions.  
Hence, programmers need to write scripts using Pig Latin language to analyze data using Apache Pig.

However, all these scripts are internally converted to Map and Reduce tasks. It is possible with a component, we call as Pig Engine. That accepts the Pig Latin scripts as input and further convert those scripts into MapReduce jobs.

**History of Pig**

Apache Pig was developed as a research project, in 2006, at Yahoo. Basically, to create and execute MapReduce jobs on every dataset it was created. By Apache incubator, Pig was open sourced, in 2007.

Then the first release of Apache Pig came out in 2008. Further, Hadoop Pig graduated as an Apache top-level project, in 2010.

**Why Do We Need Apache Pig?**

While performing any MapReduce tasks, there is a case Programmers who are not so good at Java normally used to struggle to work with Hadoop. Thus, we can say, Pig is a boon for all such programmers because:

* Without having to type complex codes in Java, using Pig Latin, programmers can perform MapReduce tasks easily.
* It also helps in reduce the length of codes, since Pig uses multi-query approach. Let’s understand it with an example. Here an operation that would require us to type 200 lines of code (LoC) in Java can be easily done by typing as less as just 10 LoC in Apache Pig. Hence, it shows, Pig reduces the development time by almost 16 times.
* When you are familiar with SQL, it is easy to learn Pig. Because Pig Latin is SQL-like language.
* It offers many built-in operators, in order to support data operations such as joins, filters, ordering, and many more. Also, it offers nested data types that are missing from MapReduce such as tuples, bags, and maps.

There are several scenarios, where we can use Pig. Such as:

* While data loads are time sensitive.
* Also, while processing various data sources.
* While we require analytical insights through sampling.

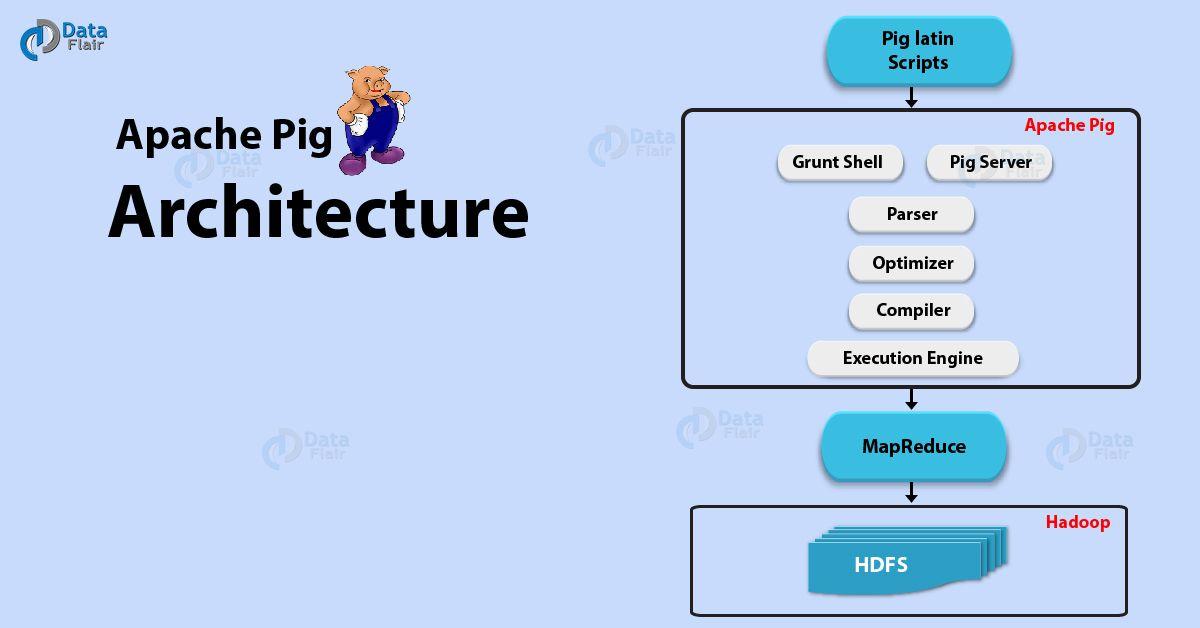
**Where Not to Use Pig?**

Also, there are some Scenarios, where we can not use. Such as:

* While the data is completely unstructured. Such as video, audio, and readable text.
* Where time constraints exist. Since Pig is slower than MapReduce jobs.
* Also, when more power is required to optimize the codes, we cannot use Pig.

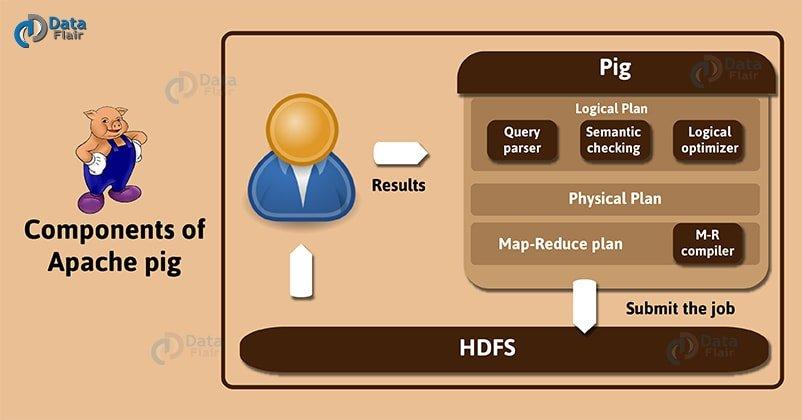
Architecture of Hadoop Pig

Here, the image, which shows the architecture of Apache Pig.



**Components of Apache Pig**

There are various components in Apache Pig Architecture which makes its execution faster as discussed below:



*Components of Apache Pig*

**a. Parser**

The Parser handles the Pig Scripts and checks the syntax of the script. It includes type checking with other checks. Therefore, an output of the parser will be a Directed Graph. However, it represents the Pig Latin statements and logical operators.

In the **DAG**, the script operators are actually represented as the nodes. Moreover, the data flows are eventually represented as edges.

**b. Optimizer**

The logical optimizer then receives the logical plan (DAG). In fact, it carries out the logical optimization such as projection and push down.

**c. Compiler**

The compiler converts the logical plan into a series of **MapReduce jobs**.

**d. Execution Engine**

In the end, the MapReduce jobs get submitted to Hadoop in a sorted order. Therefore these MapReduce jobs execute on the Hadoop and produce the desired results.

**Pig Latin Data Model**

There is a complete nested data model of Pig Latin. Meanwhile, it allows complex non-atomic data types such as map and tuple.

**a. Field and Atom**

Atom is a single value in Pig Latin, with any data type. The storage occurs in form of string and we can also use it as string and number. Various atomic values of Pig are int, long, float, double, chararray, and byte array.

Furthermore, any simple atomic value or data is actually considered as a field.  
For Example − ‘dataflair’ or ‘12’

**d. Tuples**

A record which contains an ordered set of fields is a Tuple. Thus, the fields can be of any type. A tuple is same as the row in a table of RDBMS.  
For Example − (Dataflair, 12)

**c. Bag**

A bag contains an unordered set of tuples. Therefore, a collection of tuples (non-unique) is can be a bag. Each tuple may have any number of fields. We can represent the bag as ‘{}’. It is same as a table in RDBMS.

However, it is not necessary that every tuple contains the same fields. Hence,  the fields in the same position (column) may not have the same type.

Example − {(Dataflair, 12), (Training, 11)}  
While a bag can be a field in a relation which is an inner bag.  
Example − {Dataflair, 12, {1212121212, dt@gmail.com,}}

**d. Map**

A map (or data map) contains the set of many key-value pairs. Meanwhile, the key has to be of type chararray and unique. The value can be of any type. We can represent it by ‘[]’.  
Example − [name#Dataflair, age#11]

**e. Relation**

Furthermore, a relation contains the bag of tuples. There may be no serial order of processing in the relations.

There are several features of Pig. Such as:

**i. Rich set of operators**

In order to perform several operations, Pig offers many operators. Such as join, sort, filer and many more.

**ii. Ease of programming**

Since you are good at SQL,  it is easy to write a Pig script. Because of Pig Latin as same as SQL.

**iii. Optimization opportunities**

In Apache Pig, all the tasks optimize their execution automatically. As a result, the programmers need to focus only on the semantics of the language.

**iv. Extensibility**

Through Pig, it is easy to read, process, and write data. It is possible by using the existing operators. Also, users can develop their own functions.

**v. UDFs**

By using Pig, we can create User-defined Functions in other programming languages like Java. Also, can invoke or embed them in Pig Scripts.

vi. Handles all kinds of data

Pig generally analyzes all kinds of data. Even both structured and unstructured. Moreover, it stores the results in HDFS.

Recommended Skills prior to learning Pig

Such as:

* Basic knowledge of Linux Operating System
* Fundamental programming skills

Applications of Pig

For performing tasks involving ad-hoc processing and quick prototyping, data scientists generally use Apache Pig. More of its applications are:

1. In order to process huge data sources like weblogs.
2. Also, to perform data processing for search platforms.
3. Moreover, to process time sensitive data loads.

Observations:

**Uploading Data to HDFS:**

Text

Description automatically generated

**Grunt Script 1:**

| grunt> data = LOAD '/user/cloudera/data.txt' USING PigStorage(',') as (eid:int,name:chararray,salary:int,address:chararray); 2022-06-13 10:30:39,497 [main] INFO org.apache.hadoop.conf.Configuration.deprecation - fs.default.name is deprecated. Instead, use fs.defaultFS grunt> dept = LOAD '/user/cloudera/dept.csv' USING PigStorage(',') as (did:int,dname:chararray,dhead:chararray); grunt> emp = LOAD '/user/cloudera/emp.csv' USING PigStorage(',') as (eid:int,ename:chararray,did:int,address:chararray); grunt> c = join emp by did, dept by did; grunt> dump c; |
| --- |

**Grunt Output 1:**

| 2022-06-13 10:31:22,095 [main] INFO org.apache.pig.tools.pigstats.ScriptState - Pig features used in the script: HASH\_JOIN 2022-06-13 10:31:22,095 [main] INFO org.apache.pig.newplan.logical.optimizer.LogicalPlanOptimizer - {RULES\_ENABLED=[AddForEach, ColumnMapKeyPrune, DuplicateForEachColumnRewrite, GroupByConstParallelSetter, ImplicitSplitInserter, LimitOptimizer, LoadTypeCastInserter, MergeFilter, MergeForEach, NewPartitionFilterOptimizer, PushDownForEachFlatten, PushUpFilter, SplitFilter, StreamTypeCastInserter], RULES\_DISABLED=[FilterLogicExpressionSimplifier, PartitionFilterOptimizer]} 2022-06-13 10:31:22,101 [main] INFO org.apache.pig.backend.hadoop.executionengine.mapReduceLayer.MRCompiler - File concatenation threshold: 100 optimistic? false 2022-06-13 10:31:22,106 [main] INFO org.apache.pig.backend.hadoop.executionengine.mapReduceLayer.MRCompiler$LastInputStreamingOptimizer - Rewrite: POPackage->POForEach to POJoinPackage 2022-06-13 10:31:22,106 [main] INFO org.apache.pig.backend.hadoop.executionengine.mapReduceLayer.MultiQueryOptimizer - MR plan size before optimization: 1 2022-06-13 10:31:22,106 [main] INFO org.apache.pig.backend.hadoop.executionengine.mapReduceLayer.MultiQueryOptimizer - MR plan size after optimization: 1 2022-06-13 10:31:22,118 [main] INFO org.apache.hadoop.yarn.client.RMProxy - Connecting to ResourceManager at quickstart.cloudera/127.0.0.1:8032 2022-06-13 10:31:22,119 [main] INFO org.apache.pig.tools.pigstats.ScriptState - Pig script settings are added to the job 2022-06-13 10:31:22,127 [main] INFO org.apache.pig.backend.hadoop.executionengine.mapReduceLayer.JobControlCompiler - mapred.job.reduce.markreset.buffer.percent is not set, set to default 0.3 2022-06-13 10:31:22,127 [main] INFO org.apache.pig.backend.hadoop.executionengine.mapReduceLayer.JobControlCompiler - Reduce phase detected, estimating # of required reducers. 2022-06-13 10:31:22,128 [main] INFO org.apache.pig.backend.hadoop.executionengine.mapReduceLayer.JobControlCompiler - Using reducer estimator: org.apache.pig.backend.hadoop.executionengine.mapReduceLayer.InputSizeReducerEstimator 2022-06-13 10:31:22,131 [main] INFO org.apache.pig.backend.hadoop.executionengine.mapReduceLayer.InputSizeReducerEstimator - BytesPerReducer=1000000000 maxReducers=999 totalInputFileSize=640 2022-06-13 10:31:22,131 [main] INFO org.apache.pig.backend.hadoop.executionengine.mapReduceLayer.JobControlCompiler - Setting Parallelism to 1 2022-06-13 10:31:22,237 [main] INFO org.apache.pig.backend.hadoop.executionengine.mapReduceLayer.JobControlCompiler - creating jar file Job9137013878959109241.jar 2022-06-13 10:31:24,326 [main] INFO org.apache.pig.backend.hadoop.executionengine.mapReduceLayer.JobControlCompiler - jar file Job9137013878959109241.jar created 2022-06-13 10:31:24,332 [main] INFO org.apache.pig.backend.hadoop.executionengine.mapReduceLayer.JobControlCompiler - Setting up single store job 2022-06-13 10:31:24,332 [main] INFO org.apache.pig.data.SchemaTupleFrontend - Key [pig.schematuple] is false, will not generate code. 2022-06-13 10:31:24,332 [main] INFO org.apache.pig.data.SchemaTupleFrontend - Starting process to move generated code to distributed cache 2022-06-13 10:31:24,332 [main] INFO org.apache.pig.data.SchemaTupleFrontend - Setting key [pig.schematuple.classes] with classes to deserialize [] 2022-06-13 10:31:24,358 [main] INFO org.apache.pig.backend.hadoop.executionengine.mapReduceLayer.MapReduceLauncher - 1 map-reduce job(s) waiting for submission. 2022-06-13 10:31:24,360 [JobControl] INFO org.apache.hadoop.yarn.client.RMProxy - Connecting to ResourceManager at quickstart.cloudera/127.0.0.1:8032 2022-06-13 10:31:24,368 [JobControl] INFO org.apache.hadoop.conf.Configuration.deprecation - fs.default.name is deprecated. Instead, use fs.defaultFS 2022-06-13 10:31:24,864 [JobControl] INFO org.apache.hadoop.mapreduce.lib.input.FileInputFormat - Total input paths to process : 1 2022-06-13 10:31:24,864 [JobControl] INFO org.apache.pig.backend.hadoop.executionengine.util.MapRedUtil - Total input paths to process : 1 2022-06-13 10:31:24,866 [JobControl] INFO org.apache.pig.backend.hadoop.executionengine.util.MapRedUtil - Total input paths (combined) to process : 1 2022-06-13 10:31:24,874 [JobControl] INFO org.apache.hadoop.mapreduce.lib.input.FileInputFormat - Total input paths to process : 1 2022-06-13 10:31:24,874 [JobControl] INFO org.apache.pig.backend.hadoop.executionengine.util.MapRedUtil - Total input paths to process : 1 2022-06-13 10:31:24,876 [JobControl] INFO org.apache.pig.backend.hadoop.executionengine.util.MapRedUtil - Total input paths (combined) to process : 1 2022-06-13 10:31:24,894 [JobControl] INFO org.apache.hadoop.mapreduce.JobSubmitter - number of splits:2 2022-06-13 10:31:24,913 [JobControl] INFO org.apache.hadoop.mapreduce.JobSubmitter - Submitting tokens for job: job\_1655141303825\_0002 2022-06-13 10:31:24,943 [JobControl] INFO org.apache.hadoop.yarn.client.api.impl.YarnClientImpl - Submitted application application\_1655141303825\_0002 2022-06-13 10:31:24,945 [JobControl] INFO org.apache.hadoop.mapreduce.Job - The url to track the job: http://quickstart.cloudera:8088/proxy/application\_1655141303825\_0002/ 2022-06-13 10:31:24,946 [main] INFO org.apache.pig.backend.hadoop.executionengine.mapReduceLayer.MapReduceLauncher - HadoopJobId: job\_1655141303825\_0002 2022-06-13 10:31:24,946 [main] INFO org.apache.pig.backend.hadoop.executionengine.mapReduceLayer.MapReduceLauncher - Processing aliases c,dept,emp 2022-06-13 10:31:24,946 [main] INFO org.apache.pig.backend.hadoop.executionengine.mapReduceLayer.MapReduceLauncher - detailed locations: M: dept[14,7],dept[-1,-1],c[16,4],emp[15,6],emp[-1,-1],c[16,4] C: R:  2022-06-13 10:31:24,963 [main] INFO org.apache.pig.backend.hadoop.executionengine.mapReduceLayer.MapReduceLauncher - 0% complete 2022-06-13 10:31:32,268 [main] INFO org.apache.pig.backend.hadoop.executionengine.mapReduceLayer.MapReduceLauncher - 50% complete 2022-06-13 10:31:40,115 [main] INFO org.apache.pig.backend.hadoop.executionengine.mapReduceLayer.MapReduceLauncher - 100% complete 2022-06-13 10:31:40,115 [main] INFO org.apache.pig.tools.pigstats.SimplePigStats - Script Statistics:   HadoopVersion PigVersion UserId StartedAt FinishedAt Features 2.6.0-cdh5.13.0 0.12.0-cdh5.13.0 cloudera 2022-06-13 10:31:22 2022-06-13 10:31:40 HASH\_JOIN  Success!  Job Stats (time in seconds): JobId Maps Reduces MaxMapTime MinMapTIme AvgMapTime MedianMapTime MaxReduceTime MinReduceTime AvgReduceTime MedianReducetime Alias Feature Outputs job\_1655141303825\_0002 2 1 1 1 1 1 1 1 1 1 c,dept,emp HASH\_JOIN hdfs://quickstart.cloudera:8020/tmp/temp-1655937455/tmp596918191,  Input(s): Successfully read 7 records from: "/user/cloudera/dept.csv" Successfully read 17 records from: "/user/cloudera/emp.csv"  Output(s): Successfully stored 16 records (969 bytes) in: "hdfs://quickstart.cloudera:8020/tmp/temp-1655937455/tmp596918191"  Counters: Total records written : 16 Total bytes written : 969 Spillable Memory Manager spill count : 0 Total bags proactively spilled: 0 Total records proactively spilled: 0  Job DAG: job\_1655141303825\_0002   2022-06-13 10:31:40,165 [main] WARN org.apache.pig.backend.hadoop.executionengine.mapReduceLayer.MapReduceLauncher - Encountered Warning FIELD\_DISCARDED\_TYPE\_CONVERSION\_FAILED 3 time(s). 2022-06-13 10:31:40,165 [main] INFO org.apache.pig.backend.hadoop.executionengine.mapReduceLayer.MapReduceLauncher - Success! 2022-06-13 10:31:40,167 [main] INFO org.apache.hadoop.conf.Configuration.deprecation - fs.default.name is deprecated. Instead, use fs.defaultFS 2022-06-13 10:31:40,167 [main] INFO org.apache.pig.data.SchemaTupleBackend - Key [pig.schematuple] was not set... will not generate code. 2022-06-13 10:31:40,177 [main] INFO org.apache.hadoop.mapreduce.lib.input.FileInputFormat - Total input paths to process : 1 2022-06-13 10:31:40,177 [main] INFO org.apache.pig.backend.hadoop.executionengine.util.MapRedUtil - Total input paths to process : 1 (11,Maninder Kahlon,1,Delhi,1,Sales,Maninder Kahlon) (1,Jagjit Singh,1,Delhi,1,Sales,Maninder Kahlon) (2,Hardeep Kakkar,1,Delhi,1,Sales,Maninder Kahlon) (3,Akshay Singhaniya,2,Delhi,2,Marketing,Manjot Dhillon) (12,Manjot Dhillon,2,Mumbai,2,Marketing,Manjot Dhillon) (4,Shalini Mittal,2,Punjab,2,Marketing,Manjot Dhillon) (6,Sakshi Malhotra,3,Mumbai,3,Technical,Mustafa Zahid) (5,Abeer Arora,3,Mumbai,3,Technical,Mustafa Zahid) (13,Mustafa Zahid,3,Banglore,3,Technical,Mustafa Zahid) (10,Jaspinder Narula,4,Punjab,4,Operations,Neha Sharma) (14,Neha Sharma,4,Delhi,4,Operations,Neha Sharma) (7,Ramesh Kumar,4,Banglore,4,Operations,Neha Sharma) (8,Harjinder Kumar,5,Banglore,5,Legal,Kareem Khurana) (15,Kareem Khurana,5,Punjab,5,Legal,Kareem Khurana) (16,Anubhav Bassi,6,Delhi,6,Customer Care,Anubhav Bassi) (9,Harneet Kaur,6,Punjab,6,Customer Care,Anubhav Bassi) |
| --- |

**Grunt Script 2:**

| grunt> p = foreach c generate emp::address; grunt> n = filter emp by (address == 'Delhi'); grunt> m = filter emp by (address matches '\*.l\*.'); grunt> x = foreach emp Generate (int)$0 +1, ename; grunt> c = cross emp, dept; grunt> STORE x into 'finaloutput'; |
| --- |

**Grunt Output 2:**

| 2022-06-13 10:34:49,466 [main] INFO org.apache.pig.tools.pigstats.ScriptState - Pig features used in the script: UNKNOWN 2022-06-13 10:34:49,467 [main] INFO org.apache.pig.newplan.logical.optimizer.LogicalPlanOptimizer - {RULES\_ENABLED=[AddForEach, ColumnMapKeyPrune, DuplicateForEachColumnRewrite, GroupByConstParallelSetter, ImplicitSplitInserter, LimitOptimizer, LoadTypeCastInserter, MergeFilter, MergeForEach, NewPartitionFilterOptimizer, PushDownForEachFlatten, PushUpFilter, SplitFilter, StreamTypeCastInserter], RULES\_DISABLED=[FilterLogicExpressionSimplifier, PartitionFilterOptimizer]} 2022-06-13 10:34:49,467 [main] INFO org.apache.pig.newplan.logical.rules.ColumnPruneVisitor - Columns pruned for emp: $2, $3 2022-06-13 10:34:49,470 [main] INFO org.apache.pig.backend.hadoop.executionengine.mapReduceLayer.MRCompiler - File concatenation threshold: 100 optimistic? false 2022-06-13 10:34:49,470 [main] INFO org.apache.pig.backend.hadoop.executionengine.mapReduceLayer.MultiQueryOptimizer - MR plan size before optimization: 1 2022-06-13 10:34:49,470 [main] INFO org.apache.pig.backend.hadoop.executionengine.mapReduceLayer.MultiQueryOptimizer - MR plan size after optimization: 1 2022-06-13 10:34:49,482 [main] INFO org.apache.hadoop.yarn.client.RMProxy - Connecting to ResourceManager at quickstart.cloudera/127.0.0.1:8032 2022-06-13 10:34:49,483 [main] INFO org.apache.pig.tools.pigstats.ScriptState - Pig script settings are added to the job 2022-06-13 10:34:49,487 [main] INFO org.apache.pig.backend.hadoop.executionengine.mapReduceLayer.JobControlCompiler - mapred.job.reduce.markreset.buffer.percent is not set, set to default 0.3 2022-06-13 10:34:49,590 [main] INFO org.apache.pig.backend.hadoop.executionengine.mapReduceLayer.JobControlCompiler - creating jar file Job8565394627470248938.jar 2022-06-13 10:34:51,678 [main] INFO org.apache.pig.backend.hadoop.executionengine.mapReduceLayer.JobControlCompiler - jar file Job8565394627470248938.jar created 2022-06-13 10:34:51,682 [main] INFO org.apache.pig.backend.hadoop.executionengine.mapReduceLayer.JobControlCompiler - Setting up single store job 2022-06-13 10:34:51,682 [main] INFO org.apache.pig.data.SchemaTupleFrontend - Key [pig.schematuple] is false, will not generate code. 2022-06-13 10:34:51,682 [main] INFO org.apache.pig.data.SchemaTupleFrontend - Starting process to move generated code to distributed cache 2022-06-13 10:34:51,682 [main] INFO org.apache.pig.data.SchemaTupleFrontend - Setting key [pig.schematuple.classes] with classes to deserialize [] 2022-06-13 10:34:51,687 [main] INFO org.apache.pig.backend.hadoop.executionengine.mapReduceLayer.MapReduceLauncher - 1 map-reduce job(s) waiting for submission. 2022-06-13 10:34:51,689 [JobControl] INFO org.apache.hadoop.yarn.client.RMProxy - Connecting to ResourceManager at quickstart.cloudera/127.0.0.1:8032 2022-06-13 10:34:51,691 [JobControl] INFO org.apache.hadoop.conf.Configuration.deprecation - fs.default.name is deprecated. Instead, use fs.defaultFS 2022-06-13 10:34:51,743 [JobControl] INFO org.apache.hadoop.mapreduce.lib.input.FileInputFormat - Total input paths to process : 1 2022-06-13 10:34:51,743 [JobControl] INFO org.apache.pig.backend.hadoop.executionengine.util.MapRedUtil - Total input paths to process : 1 2022-06-13 10:34:51,745 [JobControl] INFO org.apache.pig.backend.hadoop.executionengine.util.MapRedUtil - Total input paths (combined) to process : 1 2022-06-13 10:34:51,760 [JobControl] INFO org.apache.hadoop.mapreduce.JobSubmitter - number of splits:1 2022-06-13 10:34:51,773 [JobControl] INFO org.apache.hadoop.mapreduce.JobSubmitter - Submitting tokens for job: job\_1655141303825\_0003 2022-06-13 10:34:51,798 [JobControl] INFO org.apache.hadoop.yarn.client.api.impl.YarnClientImpl - Submitted application application\_1655141303825\_0003 2022-06-13 10:34:51,801 [JobControl] INFO org.apache.hadoop.mapreduce.Job - The url to track the job: http://quickstart.cloudera:8088/proxy/application\_1655141303825\_0003/ 2022-06-13 10:34:52,188 [main] INFO org.apache.pig.backend.hadoop.executionengine.mapReduceLayer.MapReduceLauncher - HadoopJobId: job\_1655141303825\_0003 2022-06-13 10:34:52,188 [main] INFO org.apache.pig.backend.hadoop.executionengine.mapReduceLayer.MapReduceLauncher - Processing aliases emp,x 2022-06-13 10:34:52,188 [main] INFO org.apache.pig.backend.hadoop.executionengine.mapReduceLayer.MapReduceLauncher - detailed locations: M: emp[15,6],x[-1,-1] C: R:  2022-06-13 10:34:52,211 [main] INFO org.apache.pig.backend.hadoop.executionengine.mapReduceLayer.MapReduceLauncher - 0% complete 2022-06-13 10:35:00,087 [main] INFO org.apache.pig.backend.hadoop.executionengine.mapReduceLayer.MapReduceLauncher - 50% complete 2022-06-13 10:35:02,221 [main] INFO org.apache.pig.backend.hadoop.executionengine.mapReduceLayer.MapReduceLauncher - 100% complete 2022-06-13 10:35:02,221 [main] INFO org.apache.pig.tools.pigstats.SimplePigStats - Script Statistics:   HadoopVersion PigVersion UserId StartedAt FinishedAt Features 2.6.0-cdh5.13.0 0.12.0-cdh5.13.0 cloudera 2022-06-13 10:34:49 2022-06-13 10:35:02 UNKNOWN  Success!  Job Stats (time in seconds): JobId Maps Reduces MaxMapTime MinMapTIme AvgMapTime MedianMapTime MaxReduceTime MinReduceTime AvgReduceTime MedianReducetime Alias Feature Outputs job\_1655141303825\_0003 1 0 2 2 2 2 n/a n/a n/a n/a emp,x MAP\_ONLY hdfs://quickstart.cloudera:8020/user/cloudera/finaloutput,  Input(s): Successfully read 17 records (831 bytes) from: "/user/cloudera/emp.csv"  Output(s): Successfully stored 17 records (281 bytes) in: "hdfs://quickstart.cloudera:8020/user/cloudera/finaloutput"  Counters: Total records written : 17 Total bytes written : 281 Spillable Memory Manager spill count : 0 Total bags proactively spilled: 0 Total records proactively spilled: 0  Job DAG: job\_1655141303825\_0003   2022-06-13 10:35:02,268 [main] WARN org.apache.pig.backend.hadoop.executionengine.mapReduceLayer.MapReduceLauncher - Encountered Warning FIELD\_DISCARDED\_TYPE\_CONVERSION\_FAILED 1 time(s). 2022-06-13 10:35:02,268 [main] INFO org.apache.pig.backend.hadoop.executionengine.mapReduceLayer.MapReduceLauncher - Success! |
| --- |

CONCLUSION:

Hence, we have seen the whole concept of Hadoop Pig. Apart from its usage, we have also seen where we cannot use it. In this experiment, we have also implemented Map Reduce functionality of Hadoop by using Pig and Pig Latin.