1. Why Map-reduce program is needed in Pig Programming?

There are certain jobs which can be executed more effectively using Hadoop MapReduce coding approach rather than using Pig or Hive scripts and vice versa. MapReduce is a powerful programming model for parallelism based on rigid procedural structure. Hadoop MapReduce allows programmers to filter and aggregate data from HDFS to gain meaningful insights from big data. There are certain circumstances when hadoop developers can choose to use Hadoop MapReduce over Pig and Hive-

* When hadoop developers need definite driver program control then they should make use of Hadoop MapReduce instead of Pig and Hive.
* Whenever the job requires implementing a custom partitioner, hadoop developers can choose MapReduce over Pig and Hive.
* If there already exists pre-defined library of Java Mappers or Reducers for a job then it is a wise decision to use Hadoop MapReduce instead of Pig and Hive.
* If the hadoop developers require good amount of testability when combining lots of large data sets then they should use MapReduce instead of Pig and Hive.
* If the application demands legacy code requirements that command physical structure then Hadoop MapReduce is a better option.
* If the job requires optimization at a particular stage of processing by making the best use of tricks like in-mapper combining then Hadoop MapReduce can prove to be a better coding approach over Pig and Hive.
* If the job has some tricky usage of distributed cache (replicated join), cross products, groupings or joins then Hadoop MapReduce is a better programming approach over Pig and Hive

1. What are advantages of pig over MapReduce?

* Decrease in development time. This is the biggest advantage especially considering vanilla map-reduce jobs' complexity, time-spent and maintenance of the programs.
* Learning curve is not steep, anyone who does not know how to write vanilla map-reduce or SQL for that matter could pick up and can write map-reduce jobs; not easy to master, though.
* Procedural, not declarative unlike SQL, so easier to follow the commands and provides better expressiveness in the transformation of data every step. Comparing to vanilla map-reduce, it is much more like an english language. It is concise and unlike Java but more like Python.
* I really liked the idea of dataflow where everything is about data even though we sacrifice control structures like for loop or if structures. This enforces the developer to think about the data but nothing else. In Python or Java, you create the control structures(for loop and ifs) and get the data transformation as a side effect. In here, data and because of data, data transformation is a first class citizen. Without data, you cannot create for loops, you need to always transform and manipulate data. But if you are not transforming data, what are you doing in the very first place?
* Since it is procedural, you could control of the execution of every step. If you want to write your own UDF(User Defined Function) and inject in one specific part in the pipeline, it is straightforward.
* Speaking of UDFs, you could write your UDFs in Python thanks to Jython. How awesome is that!
* Lazy evaluation: unless you do not produce an output file or does not output any message, it does not get evaluated. This has an advantage in the logical plan, it could optimize the program beginning to end and optimizer could produce an efficient plan to execute.
* Enjoys everything that Hadoop offers, parallelization, fault-tolerancy with many relational database features.
* It is quite effective for unstructured and messy large datasets. Actually, Pig is one of the best tool to make the large unstructured data to structured.
* You have UDFs which you want to parallellize and utilize for large amounts of data, then you are in luck. Use Pig as a base pipeline where it does the hard work and you just apply your UDF in the step that you want.

1. What is pig engine and what is its importance?
2. What are the modes of Pig execution?
3. /\* local mode \*/
4. $ pig -x local ...
6. /\* Tez local mode \*/
7. $ pig -x tez\_local ...
9. /\* mapreduce mode \*/
10. $ pig ...
11. or
12. $ pig -x mapreduce ...
13. /\* Tez mode \*/
14. $ pig -x tez ...

5. What is grunt shell in Pig?

You can run Pig in interactive mode using the Grunt shell. Invoke the Grunt shell using the "pig" command (as shown below) and then enter your Pig Latin statements and Pig commands interactively at the command line.

***Example***

These Pig Latin statements extract all user IDs from the /etc/passwd file. First, copy the /etc/passwd file to your local working directory. Next, invoke the Grunt shell by typing the "pig" command (in local or hadoop mode). Then, enter the Pig Latin statements interactively at the grunt prompt (be sure to include the semicolon after each statement). The DUMP operator will display the results to your terminal screen.

grunt> A = load 'passwd' using PigStorage(':');

grunt> B = foreach A generate $0 as id;

grunt> dump B;

6. What are the features of Pig Latin language?

Apache Pig comes with the following features −

* **Rich set of operators** − It provides many operators to perform operations like join, sort, filer, etc.
* **Ease of programming** − Pig Latin is similar to SQL and it is easy to write a Pig script if you are good at SQL.
* **Optimization opportunities** − The tasks in Apache Pig optimize their execution automatically, so the programmers need to focus only on semantics of the language.
* **Extensibility** − Using the existing operators, users can develop their own functions to read, process, and write data.
* **UDF’s** − Pig provides the facility to create **User-defined Functions** in other programming languages such as Java and invoke or embed them in Pig Scripts.
* **Handles all kinds of data** − Apache Pig analyzes all kinds of data, both structured as well as unstructured. It stores the results in HDFS.

7. Is Pig latin commands case sensitive?

The names (aliases) of relations and fields are case sensitive. The names of Pig Latin functions are case sensitive. The names of parameters (see Parameter Substitution) and all other Pig Latin keywords are case insensitive.

In the example below, note the following:

1. The names (aliases) of relations A, B, and C are case sensitive.
2. The names (aliases) of fields f1, f2, and f3 are case sensitive.
3. Function names PigStorage and COUNT are case sensitive.
4. Keywords LOAD, USING, AS, GROUP, BY, FOREACH, GENERATE, and DUMP are case insensitive. They can also be written as load, using, as, group, by, etc.
5. In the FOREACH statement, the field in relation B is referred to by positional notation ($0).

8. What is a data flow language?

In a dataflow language, you have a stream of data which is passed from instruction to instruction to be processed. Conditional execution, jumps and procedure calls route the data to different instructions. This could be seen as data flowing through otherwise static instructions like how electrical signals flow through circuits or water flows through pipes. A dataflow "if" statement would route the data to the correct branch.