**[8 reasons you should be using Apache Pig](http://blog.mortardata.com/post/33711299619/8-reasons-you-should-be-using-apache-pig)**

**October 16, 2012**

[**Scott Haylon**](http://t.umblr.com/redirect?z=http%3A%2F%2Fmortardata.com%2Fteam&t=ODViNDI3ZGU5YWIzNzkxOGIwNTliNzBjMDY5YTNjNzQ3YjNjMDViNixLeks1akdGNA%3D%3D)

As many of you know, we’re building Mortar based on a fundamental belief that big data needs to get easier.

Processing big data has made incredible strides over the past decade.  It would be hard to overstate the importance of the MapReduce programming model to this progress.  Its simple design breaks work down and recombines it in a series of parallelizable operations making it incredibly scalable – today, Yahoo, Facebook and others run MapReduce jobs on tens of thousands of machines.  Since MapReduce expects hardware failures, it can run on inexpensive commodity hardware, sharply lowering the cost of a computing cluster.

However, although MapReduce puts parallel programming within reach of most professional software engineers, developing MapReduce jobs isn’t exactly easy: (1) they require the programmer to think in terms of “map” and “reduce”, an unintuitive paradigm for most, (2) n-stage jobs can be difficult to manage, and (3) common operations (such as filters, projections, and joins) and rich data types require custom code.

This is why our friend [**Alan Gates**](http://t.umblr.com/redirect?z=http%3A%2F%2Fwww.twitter.com%2Falanfgates&t=Yzg2NThmYWY3ZWY4M2E3ZTc3ZTdiMDE4MDRjMzg3ODFkNzkyNWM4OSxLeks1akdGNA%3D%3D) and his former team at Yahoo! developed Apache Pig, which has two components:

-       PigLatin – a simple yet powerful high-level data flow language similar to SQL that executes MapReduce jobs.  PigLatin is often called simply “Pig”.

-       Pig Engine – parses, optimizes, and automatically executes PigLatin scripts as a series of MapReduce jobs on a Hadoop cluster).

So why should you consider using Pig instead of raw MapReduce?  Here are 8 big reasons:

***(1) - It’s a quick little porker.***

Pig’s multi-query approach combines certain types of operations together in a single pipeline, reducing the number of times data is scanned.  This means **1/20th the lines of code and 1/16th the development time** when compared to writing raw MapReduce. [1]

***(2) - It will eat anything.***

Pig got its name because it’s omnivorous – it will happily consume any data you feed it: structured, semi-structured, or unstructured.

***(3) - This pig does more with less.***

Pig provides the common data operations (filters, joins, ordering, etc.) and nested data types (e.g. tuples, bags, and maps) missing from MapReduce.

***(4) - It’s a pig almost anyone can ride.***

It’s easy to learn (especially if you’re familiar with SQL) and opens Hadoop to data professionals who may not be software engineers.

***(5) - It’s Pig Latin that actually makes sense.***

PigLatin *reads like a series of steps (e.g. join this data to that data, then filter the result…) so it is easy to write, and even better, it is easy to read.*

***(6) - It’s not afraid to play with snakes.***

Pig is easily extensible by UDFs – including Python, Java, JavaScript, and Ruby – so you can use them to load, aggregate, or do sophisticated analysis.

***(7) - It helps you sleep at night.***

Pig insulates your code from changes to the Hadoop Java API, so your jobs won’t suddenly break due to an update.  It also manages all details of submitting jobs and running complex data flows.

***(8) - It’s the most popular pig since***[***Babe***](http://t.umblr.com/redirect?z=http%3A%2F%2Fwww.imdb.com%2Ftitle%2Ftt0112431%2F&t=N2U2ZjlkZjM1MGY2MTY2ZDBlYmUwODc5YjZiOGU2MDhkNjIzMmI1MSxLeks1akdGNA%3D%3D)**.**

Pig is open source and actively supported by an impressive community of developers who are constantly committing back code.  It also has lots of big-time users: LinkedIn, Twitter, Salesforce, Stanford University, and many more.

Are there any downsides?  Well, because Pig translates to MapReduce, perfectly implemented MapReduce code can sometimes execute jobs slightly faster than equally well-written Pig code.  However, only the most elite MapReduce experts can optimize their code to take advantage of this performance difference, and the gap continues to shrink with each new release.

To help bring greater awareness about Pig and collaborate further with the passionate existing Pig community here in NYC, we’ve decided to share our passion by creating the [**NYC Pig User Group**](http://t.umblr.com/redirect?z=http%3A%2F%2Fwww.meetup.com%2FNYC-Pig-User-Group&t=M2JmN2VkMGZlOTMwNjg4MWRmMmVlZmM4ZWU4NzlkYjQ1YjM3YWJiNixLeks1akdGNA%3D%3D).

If you love Pig like we do, or even just think you might, we’d love to have you come join the group!  Our first meetup is Wednesday, October 24th at 6:30pm (during Strata + Hadoop World NYC) and features a talk by Apache Pig VP Daniel Dai of Hortonworks on the latest version of Pig.

Want more information about Pig?  Check out our help and tutorials site[**here**](http://t.umblr.com/redirect?z=http%3A%2F%2Fhelp.mortardata.com%2F%23%2521%2Fpig_help_and_resources&t=Y2JjYjk1MDU5YmJlZWRiZmNiMTExYjlmYjlhODA0MTk0ODc3ZjQ4OSxLeks1akdGNA%3D%3D).  If you have more questions about Pig and MapReduce or the NYC Pig User Group, please [**email us**](http://t.umblr.com/redirect?z=mailto%3Ainquiries%40mortardata.com&t=OGJlYTk0ZWFjN2I4OGJhYzJiNGZjOWNkZDFmNDhkZmQ1OTM2NTA1YixLeks1akdGNA%3D%3D)!