**Why Learn Spark?**

Spark is currently one of the most popular tools for big data analytics. You might have heard of other tools such as Hadoop. Hadoop is a slightly older technology although still in use by some companies. Spark is generally faster than Hadoop, which is why Spark has become more popular over the last few years.

There are many other big data tools and systems, each with its own use case. For example, there are database system like [Apache Cassandra](http://cassandra.apache.org/) and SQL query engines like [Presto](https://prestodb.io/). But Spark is still one of the most popular tools for analyzing large data sets.

**CPU (Central Processing Unit)**  
The CPU is the "brain" of the computer. Every process on your computer is eventually handled by your CPU. This includes calculations and also instructions for the other components of the compute.

**Memory (RAM)**  
When your program runs, data gets temporarily stored in memory before getting sent to the CPU. Memory is ephemeral storage - when your computer shuts down, the data in the memory is lost.

**Storage (SSD or Magnetic Disk)**  
Storage is used for keeping data over long periods of time. When a program runs, the CPU will direct the memory to temporarily load data from long-term storage.

**Network (LAN or the Internet)**  
Network is the gateway for anything that you need that isn't stored on your computer. The network could connect to other computers in the same room (a Local Area Network) or to a computer on the other side of the world, connected over the internet.

**Other Numbers to Know?**  
You may have noticed a few other numbers involving the L1 and L2 Cache, mutex locking, and branch mispredicts. While these concepts are important for a detailed understanding of what's going on inside your computer, you don't need to worry about them for this course. If you're curious to learn more, check out [Peter Norvig's original blog post](http://norvig.com/21-days.html) from a few years ago, and [an interactive version](http://people.eecs.berkeley.edu/~rcs/research/interactive_latency.html) for today's current hardware.

**CPU:**

The CPU is the brains of a computer. The CPU has a few different functions including directing other components of a computer as well as running mathematical calculations. The CPU can also store small amounts of data inside itself in what are called **registers**. These registers hold data that the CPU is working with at the moment.

For example, say you write a program that reads in a 40 MB data file and then analyzes the file. When you execute the code, the instructions are loaded into the CPU. The CPU then instructs the computer to take the 40 MB from disk and store the data in memory (RAM). If you want to sum a column of data, then the CPU will essentially take two numbers at a time and sum them together. The accumulation of the sum needs to be stored somewhere while the CPU grabs the next number.

This cumulative sum will be stored in a register. The registers make computations more efficient: the registers avoid having to send data unnecessarily back and forth between memory (RAM) and the CPU.

A 2.5 Gigahertz CPU means that the CPU processes 2.5 billion operations per second. Let's say that for each operation, the CPU processes 8 bytes of data. How many bytes could this CPU process per second?

20 billion bytes per second

2.5 billion operations per second x 8 bytes per operation = 20 billion bytes per second

Twitter generates about 6,000 tweets per second, and each tweet contains 200 bytes. So in one day, Twitter generates data on the order of:

(6000 tweets / second) x (86400 seconds / day) x (200 bytes / tweet) ~ 104 billion bytes / day

Knowing that tweets create approximately 104 billion bytes of data per day, how long would it take the 2.5 GigaHertz CPU to analyze a full day of tweets?

5.2 seconds

That's right! 104 billion bytes \* (1 second / 20 billion bytes) = 5.2 seconds

If a dataset is larger than the size of your RAM, you might still be able to analyze the data on a single computer. By default, the Python pandas library will read in an entire dataset from disk into memory. If the dataset is larger than your computer's memory, the program won't work.

However, the Python pandas library can read in a file in smaller chunks. Thus, if you were going to calculate summary statics about the dataset such as a sum or count, you could read in a part of the dataset at a time and accumulating the sum or count.

Here is an example of how [this works](http://pandas.pydata.org/pandas-docs/stable/io.html#io-chunking).