# Chapter 6
# Big Data Tools for Computing on Clouds and Grids

**Forest Jay Handford**
*Affectiva Inc., USA*

## ABSTRACT

*The number of tools available for Big Data processing have grown exponentially as cloud providers have introduced solutions for businesses that have little or no money for capital expenditures. The chapter starts by discussing historic data tools and the evolution to those of today. With Cloud Computing, the need for upfront costs has been removed, costs are continuing to fall and costs can be negotiated. This chapter reviews the current types of Big Data tools, and how they evolved. To give readers an idea of costs, the chapter shows example costs (in today's market) for a sampling of the tools and relative cost comparisons of the other tools like the Grid tools used by the government, scientific communities and academic communities. Readers will take away from this chapter an understanding of what tools work best for several scenarios and how to select cost effective tools (even tools that are unknown today).*

## INTRODUCTION

To understand the present and future of Big Data it is important to review the past. In 1970, Edgar F. Codd of IBM (International Business Machines) published a paper about his idea of creating a relational database that would allow people to store and access data based on key value pairs stored in tables. This led to IBM's Donald Chamberlain and Raymond Boyce developing SEQUEL (Structured English QUEry Language). They changed the acronym to SQL (Structured Query Language) when they discovered it was a registered trademark of another company. Relational Software Inc (later renamed Oracle) and IBM were the first companies to make SQL products. Presumably because IBM makes money from selling computers, they helped make SQL an open standard which was adopted by ISO and later ANSI. Microsoft and SAP also have created very popular commercial SQL products (Wikipedia, 2016).

When Relational Database Management Systems (RDBMS) were sufficient for Big Data, corporations, schools and governments made capital expenditures to pay for the hardware and software. The first savings Big Data users encountered was the release of MySQL in 1995 (Wikipedia, 2015). Rather than

DOI: 10.4018/978-1-5225-3142-5.ch006

paying Oracle, IBM, SAP or Microsoft to license the database software, they could download and install RDBMSs for free. Commercial vendors who prefer not to be under the GNU General Public License (GPL) have to buy a commercial license of MySQL from Oracle (Oracle, 2015).

The next cost revolution was from the introduction of public cloud computing by Amazon.com in 2006 (Amazon Web Services, 2015). Public cloud computing allows organizations to bypass the upfront capital expense of servers by using operational expenses to pay for the computing resources. The public cloud allows for short term projects which would be far shorter than the lifespan of a server. These projects can now pay for server time as needed.

Public cloud computing has also reduced the startup costs and risks of buying servers. If a startup fails, the company does not have to go through the hassle of trying to sell their servers for a price to help recover the original cost.

## STORAGE SYSTEMS

The most crucial component to Big Data tools is the underlying storage system. The exception is that some organizations do not need the data once it is processed and they process the data as they get it. These organizations are fortunate in that they never have to save the data to storage.

### Limitations of RDBMS With Current Big Data Definition

Big Data is often unstructured which makes it difficult to store in a relational database. Some organizations format their unstructured data in a key value pair format which allows storage in a relational database. This was sufficient until the size of Big Data outgrew the performance capabilities of the traditional row and column format that RDBMS had. Cloud services like Amazon Redshift allow customers to have a relational database in a columnar format.

When you need more storage, you have two dimensions you can scale in. Scaling vertically is upgrading existing hardware without increasing the number of computers (ie: adding storage or upgrading an existing processor). Scaling horizontally means adding additional computers which often have the same specifications as the original computer.

Vertical scaling has several downsides. In order to scale vertically, the server usually needs to be off thus requiring downtime. There is a small risk that during an upgrade the server will be accidentally damaged, thus increasing downtime, labor costs and equipment costs. Vertical scaling usually requires adding or replacing parts. For performance improvements, replacement parts are exponentially more expensive than their predecessors. While redundant systems can be built for vertically scaling servers, the cost is usually prohibitive and companies opt to rely on backups instead. A redundant system is usually nearline or online whereas a backup is often in cold storage which requires far more time to restore. Cold storage often requires the physical interaction of finding the storage medium and connecting it to a server. The network adapter is a bottleneck for communicating with the server that cannot be overcome via upgrades.

While vertical expansion incurs exponential costs, horizontal scaling incurs linear costs. As horizontal scaling adds servers, the existing server can be operational the entire time the additional server is getting prepared. If the new server's data was copied from an existing server, workload can immediately be balanced between the two servers. Organizations with horizontal scaling are more apt to store redundant data