What's the Big Deal with Big Data?

By Gwen Shapira

Learn what big data is, how it is used specifically inside an Oracle Big Data Appliance and how you can use it effectively.

What is Big Data?

Big data is a trendy marketing term, similar to *cloud* and *web 2.0*. Different kinds of problems, architectures and data storage solutions are often referred to as "big data." At its core, the term big data describes data sets that for one or more reasons don't fit well within the traditional relational database model. Forrester Research coined the "Four Vs" as the main reasons a data set may not be a good fit for a traditional relational database.

Volume: This is the main reason big data is called exactly that. It requires more storage space compared to traditional relational databases. Big data solutions are often more than a few terabytes.

Variety: Highly structured data collected into traditional OLTP systems is no longer the only type of data businesses want to handle. Emails, tweets, website log files, XML, images and video are all additional sources of data that should be stored, processed and analyzed. The high variety of data contributes to the volume of data — unstructured data is less efficient to store than traditional structured data. In addition, it typically requires more processing power to get it into useful form.

Velocity: Large volumes of data typically arrive at very high rates and must be stored by a system with very high availability and very low latency. Sometimes there is also a requirement to process the stream of data as it arrives in addition to the more traditional batch processing. This additional requirement is sometimes called "real-time analytics."

Value: Information that is valuable to the business is mixed with large quantities of irrelevant data. This adds to the storage and processing requirements, as more data must be stored than will be eventually used and more processing power must be applied to store and clean the data.

More Data Beats Smarter Algorithms

Let's explore this concept with an example. Netflix is a provider of DVD-by-mail rentals and streaming video. When registered members browse the Netflix website, they receive recommendations for movies they are likely to enjoy. This feature is valuable to Netflix, as good recommendations will be accepted by their members, driving more DVD rentals and more business. On Oct. 2, 2006, Netflix announced it would award a prize of US$ 1 million to the team that could improve its recommendations algorithm by more than 10 percent.

In March 2008, two teams of students in professor Anand Rajamaran's data mining class in Stanford tried their hand at the Netflix challenge. The two teams took very different approaches: One team came up with very sophisticated data mining algorithms. The second team used a simple algorithm but added data about movies from the IMDb website that was not part of the original Netflix data set. The second team got much better results and, at the time, became one of the leading contenders for the prize. [Source: **http://anand.typepad.com/datawocky/2008/03/more-data-usual.html**]

The moral of this story is simple: analyzing more data — either additional data sets or larger samples of the same data set — will typically yield better results than analyzing the same data with better tools.

Google, another company many attempt to learn from or emulate its success, bases many of its products on this approach. Its original search engine was so much better than its more established competitors simply because it took into account data that its competitors ignored: the links leading to each website. Google eventually made impressive breakthroughs in areas that were thought nearly impossible, such as automated translation. It was not because it had better AI algorithms than the algorithms produced by AI developers in the previous half century, but simply because it could process every word in every webpage in the world.

Google is famous for being a data-driven company, which means collecting and analyzing abundant data to support crucial business decisions. Instead of using small focus groups to decide on a change in website design, data-driven companies run several versions of the website for a while and collect detailed metrics on the way users interact with the site: how long they stay, how many products they look at, how likely they are to eventually buy a product and so on. These metrics are used to compare the various website designs and choose the one that brings the most value to the business.

Valuable Data from Unexpected Sources

While looking at more data is very valuable, looking at more data sources increases the likelihood that the new data will bring added value to your business.

One of our customers specializes in integrating information from social media with the traditional customer support infrastructure. Their system follows conversations in social media sites such as Twitter, filters them for relevance for their customer companies and processes the content of the conversation using natural language algorithms. Whenever a tweet is determined to be a complaint or request for help, their system can automatically create a customer support ticket that can be managed and tracked using traditional customer support systems.

Unstructured data such as blog posts, job postings and real-estate property listings can be a source of information about trends. Which technologies will be hot in the next six month, which cities will grow, and which political candidate is becoming more popular are all questions that hold enormous value in the human conversations taking place all over the web. The problem is that this data is not well defined. The information needed is hidden beneath layers of duplicated information, needless words, misspelled names and other problems that make it expensive to store and difficult to parse.

While relational databases can be made to deal with the challenge, they are often the more expensive and less efficient solution. Relational databases simply weren't built to process large amounts of unstructured data in the form of texts, photos and video.

continued on page 22

Not Every Problem is a Big Data Problem

Because big data is a trendy buzzword, many system architects are happy to call their medium data problems "big data" and attempt to use the same solutions we will describe as big data solutions. This is not the best idea. Big data solutions make trade-offs in usability, maintainability and consistency in order to solve problems that are specific to big data for which no other solutions exist. If traditional IT systems can adequately service your requirements, you are probably much better off with well-known systems that your employees are familiar with.

Having a very large number of distinct data sets is not big data problem because it can be solved by large number of traditional databases. Suboptimal queries, lack of partitioning, wasteful use of PL/SQL code, lack of high availability and old hardware are all are problems with solutions far easier than declaring the system as "big data" and migrating to a completely new architecture.

Big Data Solutions

As we discussed, there are five main requirements from big data solutions:

- **Scalable storage** – to store large data volumes. It should be cost effective, too, because we will want to store large quantities of data on the chance that small parts of it will eventually become valuable.
- **Scalable processing power** – to support the amount of processing unstructured data requires
- **Storage flexibility** – data is unstructured so it can't be forced into a rigid schema, or the data structure is still not completely known when the solution is designed.
- **Low latency** – to support the high insert rates and real time stream processing
- **High availability** – must have for any system critical to the business

Currently, there is no system that will support all those requirements, so big data architectures tend to be heterogeneous and combine several database types to achieve the business goals. Data storage and processing solutions that were developed for big data are often described as "NoSQL" since they are nonrelational databases and do not support the SQL language that is common in a relational databases.

They typically share the following characteristics:

- **Distributed** – In order to store large amounts of data and to scale efficiently in terms of both data storage and processing power, the data store is distributed across a large cluster of cheap machines, each using its own disks to store the data.
- **Replicated** – In order to maintain high availability in a system that is distributed across large number of machines, each block of data will be replicated to a number of servers. In this case, a loss of each specific machine will not impact the availability of the system as a whole since the data will still be usable from the replicas.
- **Eventually consistent** – If each read or write operation will require synchronization with all the machines that store the data, the system would have difficulties maintaining consistently low latency and high availability. Instead, most big data solutions allow the developers to choose the way they trade off between latency and consistency of the data.

Despite these similarities, there is a wide variety of solutions for big data systems, broadly fitting into the following categories:

Distributed Key-Value Stores

These data stores are designed to store very simple data types, typically a pair of objects. They emphasize the ability to store and retrieve those pairs very efficiently but do not allow any processing more complex than simple set/get operations. Because there is little or no structure imposed on the objects stored, these solutions are a good fit for unstructured data.

Typical use cases include monitoring information, shopping basket, user session data for websites, lists of recommended items and prepared reports for users. Cassandra, Voldermort, Riak, Redis and Oracle NoSQL are all examples of this type of data store.

Distributed Document Stores

Documents in the document stores are similar to rows in a database, but the structure is not as rigid. They do not need to adhere to the same schema and have the same columns or fields. The format within each document is a set of keys and values, and each value can be its own set of keys and values.

Here is an example of such a document:
[source: http://en.wikipedia.org/wiki/Document-oriented_database]:

```
FirstName:"Jonathan", Address:"15 Wanamassa Point Road",
Children:[{Name:"Michael",Age:10}, {Name:"Jennifer", Age:8},
{Name:"Samantha", Age:5}, {Name:"Elena", Age:2}]
```

Because document stores are slightly more structured than key-value stores, they typically support the creation of views and secondary indexes on fields that are not the primary key.

Document stores can be used to store catalogs of e-commerce sites, tracking and reporting on website usage, and for storing data for online games. MongoDB and CouchDB are the best known document stores.

Hadoop

Hadoop is not a database. It is an infrastructure like an operating system.

Hadoop consists of two parts: a distributed file system (HDFS) and programing model with a job scheduling system (MapReduce). Hadoop's file system was built to support large files, so the default block size is 64M. This makes the disk seek time a small percentage of the time it takes to retrieve the data. You can store smaller files in HDFS, but you can't store too many small files — there is memory overhead. Each block is replicated on several servers, so if any single server fails, the data is not lost and processing can continue. You can configure the number of servers each block is replicated on.

MapReduce is a parallel job-processing framework. Each MapReduce job splits the data into independent chunks (usually block-sized), each chunk is processed by a map task in parallel to all other tasks. Map tasks usually do independent transformations and filtering of the data. The output of the map tasks is the input to reduce tasks that aggregate the data and generate the final output. Hadoop places the tasks on the server that contains the data each task is required to process in order to reduce network traffic between servers. If a task hangs or stalls on one server, it can be started on additional servers to speed up processing.

Hadoop Add-ons

Hadoop is a very basic and low-level framework, so while it provides infinite capabilities for developers, it is not easy to use or manage by default. The Hadoop ecosystem consists of a variety of management and query tools that are intended to make life easier for developers and administrators. The popular tools include data integration tools such as Sqoop and Oracle Hadoop Loader, high level query languages such as Pig and Hive and databases on top of

Hadoop such as HBase. Several companies including EMC, IBM, Cloudera and Hortonworks have their own Hadoop distributions, which include management tools and sometimes an improved version of Hadoop. Oracle, EMC and Netapp also sell a hardware-software integrated solution as a Hadoop appliance.

Oracle Big Data Appliance

Oracle revealed its Big Data Appliance (BDA) at Oracle OpenWorld 2011. It comes in a full rack configuration with 18 Sun servers for a total storage capacity of 432 TB. Every server in the rack has two CPUs, each with six cores, for a total of 216 cores per full rack. Each server has 48 GB memory for a total of 864 GB of memory per full rack. [source: **http://www.oracle.com/us/products/database/big-data-for-enterprise-519135.pdf**]

Connections between the BDA nodes are via InfiniBand, and the same IB network can be used to connect BDA to external systems such as Exadata. This enables high-speed data transfer for batch processing and solves one of the major bottlenecks in integrating big data systems: the time it takes to transfer the data between the processing and storage units.

From a software perspective, Oracle's BDA includes:

- **Cloudera's Hadoop Distribution**, CDH3. This includes Hadoop itself and the add-ons mentioned above to assist in querying and job management.
- **Cloudera's enterprise manager** gives you a dashboard view of the cluster, provides central configuration for the cluster and includes a wide variety of reports and diagnostics tools.
- **Oracle NoSQL** is a key-value distributed database, optimized for low-latency data collection of large volumes of unstructured or semi-structured data. It supports simple get/set query patterns but is intended for data collection. Deeper data analysis has to be done with other tools.
- **Oracle Hadoop Loader** is a high speed data loader from Hadoop to Oracle databases. It uses data pump technologies and massive parallel processing to load data from Hadoop to Oracle as fast as possible.
- **Oracle Data Integrator Application Adapter for Hadoop** enables Oracle Data Integrator to generate Hadoop MapReduce programs through an easy-to-use graphical interface.
- **Oracle Direct Connector for Hadoop Distributed File System** (ODCH) enables the Oracle Database SQL engine to access data seamlessly from the Hadoop Distributed File System.
- Many analysts and data scientists in organizations use the R to analyze and chart their data. **Oracle Connector for R** gives R users a native, high-performance access to Hadoop Distributed File System (HDFS) and MapReduce programming framework. Oracle also has products that integrate R into the Oracle database and its BI products, but these are not part of the big data machine.

This is a very attractive package intended to complement Oracle's traditional relational database and give organizations the tools to collect, organize and analyze big data in a way that integrates with the existing Oracle tool sets and the data that is already stored in the corporate warehouse.

Use Cases for Big Data and BDA

From our discussions with customers, we identified three very common use cases for big data software or the appliance. We'll discuss the scenarios, their challenges and how big data software solves them.

Log Analysis

Modern data centers generate huge amounts of logs from applications and web services. These logs contain very specific information about how users are using the application and how the application performs. Since the information is so detailed to the granularity of single clicks, the collected data set is both massive and nearly useless without significant post-processing. We wanted to store the data on cheap storage and process it using cheap processors.

Using Hadoop, the web and application servers store the log files directly to the HDFS file system and use MapReduce jobs to extract information from those logs. The jobs run on the nodes that hold the data, so there is never a need to copy the logs between servers.

Using MapReduce, the data analysis team can answer questions such as:

- How many users use each feature in my site?
- Which page do users usually go to after visiting a specific page?
- Do people return more often to my site after I made changes?
- What usage patterns correlate with people who eventually buy a product?
- What is the correlation between slow performance and purchase rates?

Recommendation Systems

A lot of the modern web experience revolves around websites being about to predict what you'll do next or what you'd like to do but don't know about yet.

- Social media sites suggest people that you may want to connect with.
- Retail sites suggest products that you may want to buy.
- Advertising services attempts to show you advertisements for products and services you will eventually buy.
- Web-based email systems such as Gmail suggest which emails should have higher priority.

Website owners have a strong interest in making excellent suggestions. If the suggestions are useful, consumers will buy more and generate more revenue. The best way to make excellent suggestions is by using as much data as possible. To recommend a product to you, a website needs to look at your purchase history, the products you looked at but didn't buy, what your friends bought, topics you are interested in, your geographical location, etc.

This data is extracted from the business OLTP systems, social networks, census information and web logs. Since the data is unstructured and requires massive processing, collecting it in Hadoop is the natural choice. MapReduce jobs are then used to apply analysis algorithms, either in batch jobs or interactively using R. The results are loaded into the operational relational database and served to customers through the website. If the amounts of data in the result set is huge and latency is critical, the data can also be loaded to a NoSQL database and served from there. NoSQL is a good fit because the data is already processed and the application only needs to retrieve the right recommendations for a specific customer.

Typically, the analysis task starts out as a daily batch job, but soon users expect more interactive experience, such as new recommendations every time they visit the website, and to take their feedback into account in real time. More storage and processing resources are needed to collect user feedback and use it as part of the analysis and recommendation generation process. The quick rate of growth makes the use of cheap and scalable systems critical.

continued on page 24

ETL and Other Batch Processing Jobs

In many organizations, data from OTLP system is extracted, processed and loaded into the enterprise data warehouse. The processing phase often happens within the OLTP database, takes a long time to run and impacts the performance of the OLTP system. This means that the data warehouse is usually updated once a day and the data analysts never have access to the most recent operational data.

Even in organizations that don't have an enterprise data warehouse and its formal ETL process, there are batch data processing and report generation jobs that run during the night due to the time they take and performance impact.

These jobs can run much faster and without any impact to production when the data is extracted in parallel from Oracle database to the Hadoop cluster. Then Hadoop tasks can process the data and the results can be loaded back into an Oracle database. One of our customers took a 12 hour Oracle job and turned it into one hour Hadoop job that can run multiple times a day.

Oracle Big Data Appliance is a great fit for this use case for the following reasons:

- Hadoop is very scalable; the processing step can be scaled simply by adding more Hadoop nodes.
- The bottleneck to the process is usually the link through which the data moves between the database and the Hadoop cluster.
- The InfiniBand network and the high-performance Oracle Hadoop Loader will make the data loading and unloading steps faster than they can be done in any other system, driving down the time of the entire process.

Oracle Database Appliance vs. Home Grown Hadoop Cluster

Reasons to roll your own Hadoop cluster:

The No. 1 reason to roll your own cluster is cost. At the time of writing of this article, Oracle's Big Data Appliance (BDA) costs around US$ 500,000.[source: **http://www.itworld.com/it-managementstrategy/239851/big-data-oracle-cloudera-about-make-it-rain**] It has 18 servers each with 12 cores, 48 GB of RAM and 36 TB of storage. A reasonably good server with six cores, 16 GB of RAM and 12 TB of storage from Dell retails for around US$ 6000. A solution with 54 of those servers cost around US$ 324,000 — cheaper than Oracle's BDA, but it has more cores and as much memory and storage. Oracle's offer is competitive, but the roll-your-own cluster is still economical.

Another good reason to roll your own is the flexibility. Appliances are called that way because they have a very specific configuration. You get a fixed number of nodes, processors, RAM and storage. Oracle BDA has an 18-node rack. What if you want 12 nodes, or 23? What if you want less RAM and more cores? Oracle's BDA, being an appliance, doesn't give you these options.

One of the very nice things about Hadoop is the fact that it does not demand a fancy hardware just to start. This property allows you to develop small scale proof-of-concept clusters running on inexpensive "test" servers in the data center, even workstations used by the development team. Once you demonstrate business value, justifying a budget for a larger system becomes easier. Several successful large-scale enterprise clusters started that way. A popular but effective trick is making sure that the business folks start relying on your data. Then you create an outage. When users ask for the reason, you explain that it was just a proof-of-concept system running on some spare hardware and that you needed the hardware for another project and hence the outage. If users want a full-blown system, well, it would cost US$ 384,000.

Reasons to go with Oracle Big Data Appliance

Cost is not the only consideration in making a decision on a system. An appliance gets you a standard configuration that a large vendor is willing to support now and later, which may come at a premium, but it has an implied reliability. It is easy to roll your own Hadoop cluster if you already know how to size a Hadoop system, which hardware to get and how to configure it for maximum performance and reliability. However, it is very common to see customers beg their software vendors for hardware advice and pay expensive consultants for advice on hardware purchases. Expensive hardware systems can become even more expensive outages later because their specifications did not satisfy the workload requirements.

Getting business value out of big data is a difficult problem without adding the extra difficulty of sizing the hardware and configuring the Hadoop cluster. In many cases, it is an excellent idea to let the vendor take care of the sizing exercise and concentrating internal IT resources of the other problems. This is even more practical since there is still a significant shortage of skilled resources with reasonable Hadoop experience. Your system administrator, while being a star in UNIX environment, probably does not know what hardware to evaluate, how to size the system, how to best configure Hadoop and how to troubleshoot when something goes wrong. A reliable vendor support becomes extra critical as a result.

Conclusion

Businesses can analyze new data streams to gain new insights into their business and markets and to provide added value to their customers. The new data streams are challenging in their volume, variety and velocity, and traditional IT infrastructures have to be augmented by new systems, built to collect, organize and analyze the new data. Hadoop, its add-ons and the various NoSQL databases exist to support the new storage and processing requirements. These solutions are low cost, scalable and flexible. Due to the increasing demand for enterprise version of these solutions, vendors such as Oracle also offer built-in appliances that integrate well with the existing IT infrastructure to offer a complete solution. In this article, you learned the basics of big data architecture, its defining components and practical considerations in deploying a successful big data project, including an overview of the solution from Oracle's Big Data Appliance.

About the Author

Gwen (Chen) Shapira spent the last 10 years in different technical positions in the IT industry. Shapira is an Oracle ACE, Oak Table member, Oracle Certified Professional, currently working as a senior Oracle DBA for Pythian and specializes in scalability and high-availability solutions such as RAC and Streams. She is a board member of North California Oracle User Group and has presented at a number of conferences. Shapira posts her ideas and opinions on the Pythian Blog: **www.pythian.com/news/author/shapira**.