DW-On-Demand: The Data Warehouse Redefined in the Cloud

Ravi Chandran

Ravi Chandran is the CTO of XtremeData, Inc., which provides a massively scalable, high-performance SQL engine for big data analytics that is deployable on-premises and in the cloud.
ravi.chandran@xtremedata.com

Abstract

The rapid growth in big data volumes from new sources is orders of magnitude larger than the growth of traditional enterprise operational data. Data volumes are also growing much faster than traditional IT budgets. Because no enterprise can afford to ignore the value of big data analysis, this growth has created tremendous challenges and has spurred many new technologies. This article describes a combination of technologies that has enabled the redefinition of the traditional data warehouse.

Introduction

The primary data challenges facing today's enterprises fall into three categories: capital investment, lead time to deployment, and technology. Choices made in each category have a significant impact on the business and can dramatically help or hinder the execution of corporate strategy.

Each of these three categories also naturally falls into the primary domain of the chief financial officer (CFO), chief marketing officer (CMO), and chief information officer (CIO), respectively.

The advent of public clouds has fundamentally changed both the cost-benefit analysis and the lead-time parameters of these data challenges. Since the financial crisis in 2008, CFOs have become cautious about their capital expenses; they are not ready to approve million-dollar expenses for hardware and software. However, no CFO can afford to ignore the opportunities presented by the cloud.

Every CFO is pushing their enterprise's CIO to identify workloads that can be moved to the cloud. At the same time, every CMO is eager to leverage big data insights; CMOs have no stomach for protracted IT projects. Cloud-based infrastructure-as-a-service solutions solve both the capital investment and lead-time challenges. CIOs can readily assess the viability and cost-effectiveness of the cloud with small, incremental investments of time and resources.

Technology

Big data has spawned many new and rapidly evolving technologies, and enterprises now face a bewildering set of choices. The focus of this article is limited to data warehouses in the domain of well-defined, structured data. Loosely structured or unstructured data, although constituting the majority of digital data generated, still remains largely outside the domain of mainstream enterprise analytics and has its own specialized solutions.

The challenge of structured big data analytics is to deploy solutions that scale affordably in the context of loading, joining, aggregating, analyzing, and reporting on billions of records with sufficient speed to meet business demands.

The past few years have seen the rise of many Hadoop and NoSQL solutions that are still evolving. However, a consensus is emerging that Hadoop-like solutions need to be augmented by a full-featured SQL engine that can operate at large scale. SQL skills are ubiquitous in the market, and a massively parallel processing (MPP) SQL engine that hides the complexity of parallel execution is an attractive solution that can be deployed immediately.

Cloud Architecture

All components of a big data solution need to meet one common criterion: the ability to be deployed on today's converged, sharable hardware infrastructure. This converged architecture applies equally to public clouds and to private clouds within the enterprise data center. Hardware infrastructure in a converged data center comprises the usual three layers: server, network, and storage.

By definition, the server layer is a virtualized, x86 Linux machine; no other CPUs or operating systems are significant contenders. Therefore, there is little differentiation in terms of hardware performance or functionality among vendors of x86 machines.

The network layer offers more interesting choices. Unlike the server layer, the performance of the network tends to be divided into steps rather than being continuous: 1 or 10 gigE, double data rate (DDR), or quad data rate (QDR) InfiniBand. Network switches also have a range of capabilities that can significantly impact data-intensive processing such as bi-section bandwidth and lane throttling or bandwidth guarantees. The most prevalent network infrastructure today is based on 10 gigE.

We will discuss the storage layer in more depth.

Storage

The storage configuration for data-intensive processing has diverged from the traditional SAN/NAS (storage area network/network-attached storage) and has coalesced around two choices: object store and distributed file system. This shift was driven by the need for scalability in MPP systems, which shared solutions such as SAN/NAS cannot provide.

The new offerings in storage are exemplified by OpenStack: Cinder (block store file system) and Swift (object store). These offerings are similar to those of public cloud providers, such as Amazon Web Services (AWS), Google Compute Engine, and IBM SoftLayer.

Underlying both object store and distributed file system solutions is the same hardware infrastructure: a cluster of machines with locally attached storage. This cluster is typically built with the same x86 Linux machines in the server layer, but other choices may also be employed in the future, such as low-power ARM CPUs. The storage medium itself uses rotating magnetic disks, flash memory, or some combination of these technologies.

The block store could be local storage attached to each Linux server or a distributed solution spread across multiple servers. The object store is a redundant storage system that offers the features of a traditional SAN/NAS system in a more scalable fashion, leveraging commodity Linux clusters.

This combination of object and block store enables a redefinition of the traditional data warehouse architecture. Figure 1 shows a simplified block diagram of such a traditional data warehouse (DW), comprising two major components: a reliable storage layer and a SQL querying layer.

In modern cloud architecture, storage can be decoupled from the SQL query engine, as shown in Figure 2. The reliable storage layer is provided by the cloud object store. The SQL query layer is configured as a set of computing resources in the cloud, populated with data from the object store. The populated data can reside either in a cloud block store or locally in storage associated with the computing resources during the lifetime of the SQL query engine. When the engine is shut down, changed data can be uploaded back to the object store and repopulated upon restart.

As illustrated, the decoupling of persistent storage from the on-demand SQL engine enables a fundamental redefinition of the data warehouse. This decoupling also enables the storage and query layers to scale independently.

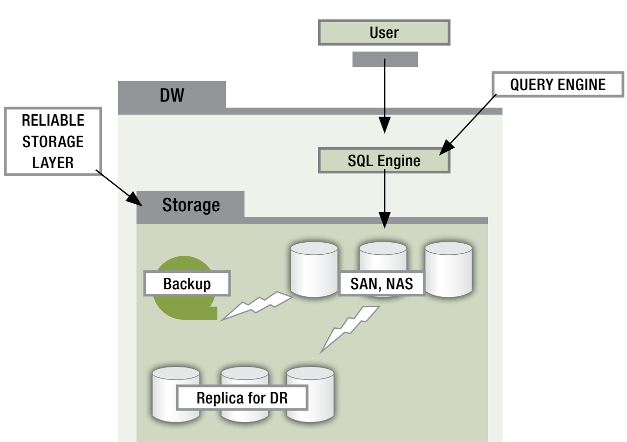

Figure 1: Architecture of the traditional data warehouse.

Object Store as the Data Repository

The cloud object store offers all the same features and benefits of the storage layer of a traditional data warehouse, typically a SAN/NAS:

- **Reliability.** Cloud object stores are designed for resiliency via replication and offer very durable data protection. For example, according to the company, Amazon S3 is designed to provide 99.999999999% durability of objects over a given year.

- **Concurrent access.** Just like a traditional SAN/NAS, cloud object stores support multiple concurrent accesses, with a scalable, high-aggregate bandwidth.

Concurrent access is critical. Cloud object stores are typically designed to support a certain bandwidth for each independent connection to the object. This feature can be leveraged very effectively by an MPP SQL engine. If each node of the parallel SQL engine cluster can independently connect to the object store, then aggregate bandwidth will scale linearly with the number of nodes.

This scalable and high-aggregate bandwidth between the object store and the MPP SQL engine is the key enabling technology that permits the decoupling of persistent storage and the SQL query engine in the data warehouse. To be practically feasible and economically attractive, the aggregate bandwidth should be on the order of gigabytes per second.

Load from Object Store

Consider a 10 TB (terabyte) data store as reasonably representative of an enterprise data warehouse or data mart. If this data is maintained in a cloud object store as the persistent and durable storage of record, how quickly can a SQL query engine be spun up and loaded with the data for analysis?

In the cloud, a SQL engine on a cluster of servers can be spun up in a matter of minutes. Data load is another matter: this could consume hours to days, depending on the capabilities of the SQL engine. Some SQL engines (for instance, column-store engines such as AWS Redshift) impose certain ordering constraints and require

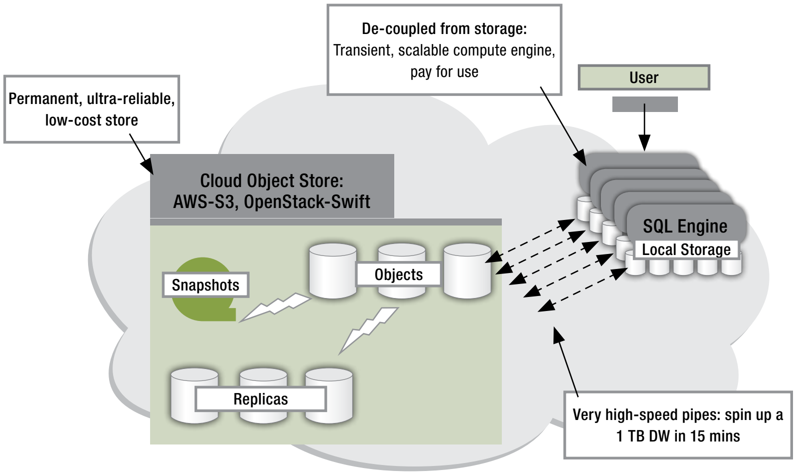

Figure 2: New data warehouse architecture enabled by the cloud.

data sorting that drastically slows down the effective data load time.

Long load times can negate the benefits of decoupling the persistent storage layer from the SQL query layer. First, consider the inconvenience factor: as mentioned, lead time and agility are important concerns for the CMO and CEO, and no one has the patience to wait 12 to 24 hours for a system to be available for analysis. Second, long load times will also rule out the significant cost savings from the on-demand, pay-for-use model for the SQL query engine, as described later in this article.

On the other hand, if the SQL engine can ingest data at high speeds in parallel, with a scalable and high-aggregate bandwidth, then all the benefits listed here will accrue.

Tests using real-world data sets have shown excellent results on several cloud environments, including AWS, Rackspace, and IBM SoftLayer. On IBM SoftLayer, an experiment using an 8-server SQL engine cluster and 1 TB of total data from an object store (6.6 billion rows across 5 tables) had a total load time of about 375 seconds (averaged over several runs). Extrapolating to a 10 TB data store and assuming a 16x node SQL engine cluster (one full data center rack), the total load time would be about 1,875 seconds or roughly half an hour—a completely acceptable deployment time.

Economics

If a 10 TB database (compressed in object store) can be loaded and ready for querying in 30 minutes, it becomes practical to use the cloud object store as the durable storage of record. It also enables significant cost savings.

Cloud object stores are inexpensive; they average about $100 per TB per month. On the other hand, the query engine requires a cluster of computing servers, which are priced by the hour and are much more expensive. Reasonably powerful servers are priced at about $4 per server hour.

Now consider the 10 TB DW built using a cluster of 16 servers for the SQL engine (Table 1).

ITEM	COST PER MONTH	COMMENTS
Object store	$1,000	
Query engine	$46,000	Always on
	$13,000	Running 10 hours/day, 5 days/week

Table 1: Costs of a 10 TB DW built using a cluster of 16 servers for the SQL engine.

A dramatic reduction of 70 percent in the monthly cost can be achieved if the data warehouse is decomposed into separate storage and query layers, with the query layer turned on only during business hours.

A consensus is emerging that Hadoop-like solutions need to be augmented by a full-featured SQL engine that can operate at large scale.

Agility

Other additional and significant benefits of the redefined “on-demand” data warehouse include elastic scaling, disaster recovery, and non-interference of unrelated workloads.

Elastic Scaling

Loading data rapidly from the object store enables “right sizing” of the system to match analytic workloads. For example, in the 10 TB data store discussed earlier, consider workloads that vary by day of the week. Mondays are heavy with data from the weekend, while other days are light. On Mondays, the SQL engine cluster can be spun up as 32 servers, while other days, it remains at 16 servers. Monday’s peak workload can be handled with ease for a small incremental cost, leveraging all benefits of the cloud infrastructure.

Disaster Recovery

Maintaining the object store as the durable storage of record also provides a built-in disaster recovery mechanism. Because the object store itself is durable and disaster-proof by design, only the SQL query engine needs to be considered for disaster recovery. A disaster that knocks out the SQL engine cluster will still leave intact the last known good version of the data in the object store. Disaster recovery from this version is trivial—simply spin up another SQL engine cluster elsewhere in the cloud.

Of course, this implies that the last version is reasonably current. Be sure to reverse the download process and upload the latest version of the database from the SQL cluster back to the object store at periodic intervals, such as at the end of every day.

As with the download, speed is of the essence. If the upload is slow, the practical feasibility of maintaining the object store with current data becomes questionable. Our tests verified that reversing the load process is also fast and scalable.

Data in the SQL cluster also needs to be protected. The SQL engine itself should be resilient and immune to disk, server, and network failures. The SQL engine used in our tests provides multiple levels of data protection, including dual-parity local RAID (redundant array of independent disks) storage and replication across servers.

Workload Isolation

Interference from unrelated workloads in a traditional data warehouse is unavoidable. Most legacy systems in enterprise data centers have evolved and grown over the years and typically contain many disparate databases and workloads within the same physical system. The workloads contain mission-critical processing that needs to run as scheduled, intermixed with sporadic, ad hoc, or non-critical processes. Legacy vendors of data warehouse systems such as Teradata have therefore developed complex tools for workload management that allow the administrator to restrict resources to users and processes.

However, these workload management mechanisms are difficult to implement, consume valuable resources themselves, and need skilled administrators for effective implementation. It is not unusual for legacy systems to be brought down by runaway processes that escaped the management controls.

The redefined data warehouse offers a simpler and more elegant solution: implementing unrelated workloads in independent systems. In the cloud, it no longer makes sense to keep adding nodes to an existing system or keep all data in a single system, as recommended by legacy vendors of on-premises systems.

The single system in the cloud is really the single storage of record that can be maintained in the object store. This frees the SQL query engine to be sized appropriately for different workloads, and also enables isolated, separate querying environments without interference. Management of an on-demand data warehouse in the cloud is also much simpler. No complex workload management tools are necessary; deployment, monitoring, and reporting are automated via open source frameworks such as Heat in OpenStack.

The nexus of big data and the cloud has created major disruptions in the world of enterprise IT.

Summary

The nexus of big data and the cloud has created major disruptions in the world of enterprise IT. Traditional hardware and software vendors and their delivery methods are being challenged. In the particular domain of structured big data analytics, the cloud enables a redefinition of the data warehouse: storage can be decoupled from computing.

The cloud infrastructure provides two separate and distinct components: the object store and the computing servers. The object store provides a highly reliable storage layer with the durability of an enterprise-grade SAN/NAS. In the redefined data warehouse, the object store becomes the storage of record, with the query engine being spun up on demand using the servers in the cloud. This is only practical if the query engine meets key requirements of scalability, speed of ingest, and speed of querying. This article described the architecture of a redefined data warehouse, along with measured results from real-world tests to validate its feasibility. ■