



**Overview of Big Data, Data Ingestion and Processing**

In the era of the Internet of Things and Mobility, with a huge volume of data becoming available at a fast velocity, there must be the need for an efficient Analytics System.

Also, the variety of data is coming from various sources in different formats, such as sensors, logs, structured data from an RDBMS, etc. In the past few years, the generation of new data has drastically increased. More applications are being built, and they are generating more data at a faster rate.

Earlier, Data Storage was costly, and there was an absence of technology which could process the data in an efficient manner. Now the storage costs have become cheaper, and the availability of technology to transform [Big Data](https://www.xenonstack.com/services/big-data-consulting-services/) is a reality.

**What is Big Data Technology?**

According to the Author [Dr. Kirk Borne](https://twitter.com/kirkdborne), Principal Data Scientist, **Big Data Definition is Everything, Quantified, and Tracked**.Let’s pick that apart -

* **Everything** – Means every aspect of life, work, consumerism, entertainment, and play is now recognized as a source of digital information about you, your world, and anything else we may encounter.
* **Quantified** – Means we are storing those "everything” somewhere, mostly in digital form, often as numbers, but not always in such formats. The quantification of features, characteristics, patterns, and trends in all things is enabling Data Mining, Machine Learning, statistics, and discovery at an unprecedented scale on an unprecedented number of things. The Internet of Things is just one example, but the Internet of Everything is even more impressive.
* **Tracked** – Means we don’t directly quantify and measure everything just once, but we do so continuously. It includes - tracking your sentiment, your web clicks, your purchase logs, your geolocation, your social media history, etc. or tracking every car on the road, or every motor in a manufacturing plant or every moving part on an airplane, etc. Consequently, we see the emergence of smart cities, smart highways, personalized medicine, personalized education, precision farming, and so much more.

**Advantages of Big Data**

* Smarter Decisions
* Better Products
* Deeper Insights
* Greater Knowledge
* Optimal Solutions
* Customer-Centric Products
* Increased Customer Loyalty
* More Automated Processes, more accurate Predictive and Prescriptive Analytics
* Better models of future behaviors and outcomes in Business, Government, Security, Science, Healthcare, Education, and more.

**D2D Communication Meets Big Data**

* Data-to-Decisions
* Data-to-Discovery
* Data-to-Dollars

**10 Vs of Big Data**

**Big Data Architecture & Patterns**

The Best Way to a solution is to "Split The Problem."Big Data Solution can be well understood using Layered Architecture. The Layered Architecture is divided into different Layers where each layer performs a  particular function.

This Architecture helps in designing the Data Pipeline with the various requirements of either Batch Processing System or Stream Processing System. This architecture consists of 6 layers which ensure a secure flow of data.

* **Data Ingestion Layer**

This layer is the first step for the data coming from variable sources to start its journey. Data here is prioritized and categorized which makes data flow smoothly in further layers.

* **Data Collector Layer**

In this Layer, more focus is on the transportation of data from ingestion layer to rest of data pipeline. It is the Layer, where components are decoupled so that analytic capabilities may begin.

* **Data Processing Layer**

In this primary layer, the focus is to specialize the data pipeline processing system, or we can say the data we have collected in the previous layer is to be processed in this layer. Here we do some magic with the data to route them to a different destination, classify the data flow and it’s the first point where the analytic may take place.

* **Data Storage Layer**

Storage becomes a challenge when the size of the data you are dealing with, becomes large. Several possible solutions can rescue from such problems. Finding a storage solution is very much important when the size of your data becomes large. This layer focuses on "where to store such a large data efficiently."

* **Data Query Layer**

This is the layer where active analytic processing takes place. Here, the primary focus is to gather the data value so that they are made to be more helpful for the next layer.

* **Data Visualization Layer**

The visualization, or presentation tier, probably the most prestigious tier, where the data pipeline users may feel the VALUE of DATA. We need something that will grab people’s attention, pull them into, make your findings well-understood.

You May also Love to Read [Why We Need Modern Big Data Integration Platform & Data Ingestion Patterns](https://www.xenonstack.com/blog/big-data-engineering/streaming-integration-solution-platform/)

**Big Data Ingestion Architecture**

Data ingestion is the first step for building Data Pipeline and also the toughest task in the System of Big Data. In this layer we plan the way to ingest data flows from hundreds or thousands of sources into Data Center. As the Data is coming from Multiple sources at variable speed, in different formats.

That's why we should properly ingest the data for the successful business decisions making. It's rightly said that "If starting goes well, then, half of the work is already done."

**What is Ingestion in Big Data?**

**Big Data Ingestion involves connecting to various data sources, extracting the data, and detecting the changed data.** It's about moving data - and especially the unstructured data - from where it is originated, into a system where it can be stored and analyzed.

We can also say that Data Ingestion means taking data coming from multiple sources and putting it somewhere it can be accessed. It is the beginning of Data Pipeline where it obtains or import data for immediate use.

Data can be streamed in real time or ingested in batches, When data is ingested in real time then, as soon as data arrives it is ingested immediately. When data is ingested in batches, data items are ingested in some chunks at a periodic interval of time. Ingestion is the process of bringing data into Data Processing system.

Effective Data Ingestion process begins by prioritizing data sources, validating individual files and routing data items to the correct destination.

**Challenges in Data Ingestion**

As the number of IoT devices increases, both the volume and variance of Data Sources are expanding rapidly. So, extracting the data such that it can be used by the destination system is a significant challenge regarding time and resources. Some of the other problems faced by Data Ingestion are -

* When numerous Big Data sources exist in the different format, it's the biggest challenge for the business to ingest data at the reasonable speed and further process it efficiently so that data can be prioritized and improves business decisions.
* Modern Data Sources and consuming application evolve rapidly.
* Data produced changes without notice independent of consuming application.
* Data Semantic Change over time as same Data Powers new cases.
* Detection and capture of changed data - This task is difficult, not only because of the semi-structured or unstructured nature of data but also due to the low latency needed by individual business scenarios that require this determination.

That's why it should be well designed assuring following things -

* Able to handle and upgrade the new data sources, technology and applications
* Assure that consuming application is working with correct, consistent and trustworthy data.
* Allows rapid consumption of data
* Capacity and reliability - The system needs to scale according to input coming and also it should be fault tolerant.
* Data volume - Though storing all incoming data is preferable; there are some cases in which aggregate data is stored.

**Data Ingestion Parameters**

* **Data Velocity -** Data Velocity deals with the speed at which data flows in from different sources like machines, networks, human interaction, media sites, social media. The movement of data can be massive or continuous.
* **Data Size -** Data size implies enormous volume of data. Data is generated from different sources that may increase timely.
* **Data Frequency (Batch, Real-Time) -** Data can be processed in real time or batch, in real time processing as data received on same time, it further proceeds but in batch time data is stored in batches, fixed at some time interval and then further moved.
* **Data Format (Structured, Semi-Structured, Unstructured) -** Data can be in different formats, mostly it can be the structured format, i.e., tabular one or unstructured format, i.e., images, audios, videos or semi-structured, i.e., JSON files, CSS files, etc.

**Big Data Ingestion Key Principles**

To complete the process of Data Ingestion, we should use right tools for that and most important that tools should be capable of supporting some of the fundamental principles written below -

* **Network Bandwidth -** Data Pipeline must be able to compete with business traffic. Sometimes traffic increases or sometimes decreases, so Network bandwidth scalability is biggest Data Pipeline challenge. Tools are required for bandwidth throttling and compression capabilities.
* **Unreliable Network -** Data Ingestion Pipeline takes data with multiple structures, i.e., images, audios, videos, text files, tabular files data, XML files, log files, etc. and due to the variable speed of data coming, it might travel through the unreliable network. Data Pipeline should be capable of supporting this also.
* **Heterogeneous Technologies and System -** Tools for Data Ingestion Pipeline must be able to use different data sources technologies and different operating system.
* **Choose Right Data Format -** Tools must provide data serialization format, that means as data comes in the variable format so converting them into single format will provide an easier view to understand or relate the data.
* **Streaming Data -** It depends upon business necessity whether to process the data in batch or streams or real time. Sometimes we may require both processing. So, tools must be capable of supporting both.

You May also Love to Read [Data Ingestion Using Apache Nifi For Building Data Lake Using Twitter Data](https://www.xenonstack.com/blog/big-data-engineering/streaming-lake-analytics-nifi/)

**Data Serialization in Big Data**

Different types of users have various types of data consumer needs. Here we want to share variable data, so we must plan how the user can access data in a meaningful way. That's why a single image of variable data optimize the data for human readability.

Approaches used for this are -

* [**Apache Thrift**](https://thrift.apache.org/)

It's an RPC Framework containing Data Serialization Libraries.

* [**Google Protocol Buffers**](https://developers.google.com/protocol-buffers/)

It can use the specially generated source code to easily write and read structured data to and from a variety of data streams and using a variety of languages.

* [**Apache Avro**](https://avro.apache.org/)

The more recent Data Serialization format that combines some of the best features which previously listed. Avro Data is self-describing and uses a JSON-schema description. This schema is included with the data itself and natively support compression. Probably it may become a de facto standard for Data Serialization.

**Big Data Ingestion Tools**

**Apache Flume Architecture**

Apache Flume is a distributed, reliable, and available service for efficiently collecting, aggregating, and moving large amounts of log data.

It has a straightforward and flexible architecture based on streaming data flows. It is robust and faults tolerant with tunable reliability mechanisms and many failovers and recovery mechanisms.

It uses a simple, extensible data model that allows for an online analytic application.

**Functions of Apache Flume**

* **Stream Data -** Ingest streaming data from multiple sources into Hadoop for storage and analysis.
* **Insulate System -** Buffer storage platform from transient spikes, when the rate of incoming data exceeds the rate at which data can be written to the destination
* **Scale Horizontally** - To ingest new data streams and additional volume as needed.

**Apache Nifi Overview**

[Apache Nifi](https://nifi.apache.org/) provides an easy to use, the powerful, and reliable system to process and distribute data. **Apache NiFi supports robust and scalable directed graphs** of data routing, transformation, and system mediation logic. Its functions are -

* Track data flow from beginning to end
* Seamless experience between design, control, feedback, and monitoring
* Secure because of SSL, SSH, HTTPS, encrypted content.

**Integrating Elasticsearch with Logstash**

[Elastic Logstash](https://www.elastic.co/products/logstash) is an open source, server-side data processing pipeline that ingests data from a multitude of sources simultaneously transforms it, and then sends it to your “stash, " i.e., Elasticsearch.

It easily ingests from your logs, metrics, web applications, data stores, and various AWS services and done in continuous, streaming fashion. It can Ingest Data of all Shapes, Sizes, and Sources.

**Big Data Pipeline Architecture**

In this Layer, more focus is on transportation data from ingestion layer to rest of Data Pipeline. Here we use a messaging system that will act as a mediator between all the programs that can send and receive messages.

Here the tool used is Apache Kafka. It's a new approach in message-oriented middleware.

**Getting Started with Big Data Pipeline**

* Data Pipeline the main component of Data Integration. **All transformation of data happens in Data Pipeline**.
* It is a Python-based tool that streams and transforms real-time data to service that need it.
* Data Pipeline Automate the movement and transformation of data. Data Pipeline is a Data Processing engine that runs inside your application.
* It is used to transform all the incoming data in a standard format so that we can prepare it for analysis and visualization. Data Pipeline is built on Java Virtual Machine (JVM).
* So, a Data Pipeline is a series of steps that your data moves through. The output of one step in the process becomes the input of the next. Data, typically raw data, goes on one side, passes through a series of steps.
* The steps of a Data Pipeline can include cleaning, transforming, merging, modelling and more, in any combination.

**Big Data Pipeline Functions**

* **Data Ingestion**

Data Pipeline Helps in bringing data into your system. It means taking unstructured data from where it is originated into a system where it can be stored and analyzed for making business decisions

* **Data Integration**

Data Pipeline also helps in bringing different types of data together.

* **Data Organization**

Organizing data means an arrangement of data; this arrangement is also made in Data Pipeline.

* **Data Refining**

It's also one of the processes where we can enhance, clean, improve the raw data.

* **Data Analytics**

After improving the useful data, Data Pipeline provides us with the processed data on which we can apply the operations on raw data and can make business decisions accurately.

**Need Of Big Data Pipeline**

A Data Pipeline is software that takes data from multiple sources and makes it available to be used strategically for making business decisions.

Primarily reasons for the need of data pipeline is because it's tough to monitor Data Migration and manage data errors. Other reasons for this are below -

* **Business Decisions**- Critical Analysis is only possible when combining data from multiple sources. For making business decisions, we should have a single image of all the data coming.
* **Connections -** All the time data keeps on increasing, new data came and old data modified, so, each new integration can take anywhere from a few days to a few months to complete.
* **Accuracy -** The only way to build trust with data consumers is to make sure that your data is auditable. One best practice that’s easy to implement is never to discard inputs or intermediate forms when altering data.
* **Latency -** The fresher your data, the agiler your company’s decision-making can be. Extracting data from APIs and databases in real-time can be difficult, and many target data sources, including large object stores like Amazon S3 and analytics databases like Amazon Redshift, are optimized for receiving data in chunks rather than a stream.
* **Scalability -** Data can be increased or decreased with time we can't say for on Monday data will come less and rest of days comes a lot for processing. So, usage of data is not uniform. What we can do is making our pipeline so scalable that able to handle any amount of data coming at variable speed.

**Big Data Pipeline Use Cases**

Data Pipeline is useful to some roles, including CTOs, CIOs, Data Scientists, Data Engineers, BI Analysts, SQL Analysts, and anyone else who derives value from a unified real-time stream of user, web, and mobile engagement data. So, use cases for data pipeline are given below -

* For Business Intelligence Teams
* For SQL Experts
* For Data Scientists
* For Data Engineers
* For Product Teams

**Apache Kafka Overview**

It is used for buildingreal-time data pipelinesand streaming apps. It can process streams of data in real-time and store streams of data safely in a distributed replicated cluster.

Kafka works in combination with Apache Storm, Apache HBase and Apache Spark for real-time analysis and rendering of streaming data.

* Building Real-Time streaming Data Pipelines that reliably get data between systems or applications
* Building Real-Time streaming applications that transform or react to the streams of data.

**Apache Kafka Use Cases**

* Stream Processing
* Website Activity Tracking
* Metrics Collection and Monitoring
* Log Aggregation

**Apache Kafka Features**

* One of the features of Apache Kafka is durable Messaging.
* Apache Kafka relies heavily on the file system for storing and caching messages: rather than maintain as much as possible in memory and flush it all out to the filesystem, all data is immediately written to a persistent log on the filesystem without necessarily flushing to disk.
* Apache Kafka solves the situation where the producer is generating messages faster than the consumer can consume them in a reliable way.

**Apache Kafka Architecture**

Apache Kafka System design act as Distributed commit log, where incoming data is written sequentially on disk. There are four main components involved in moving data in and out of Apache Kafka -

* **Topics -** Topic is a user-defined category to which messages are published.
* **Producers -** Producers post messages to one or more topics
* **Consumers -** Consumers subscribe to topics and process the posted messages.
* **Brokers -** Brokers that manage the persistence and replication of message data.

**Big Data Processing Layer**

In the previous layer, we gathered the data from different sources and made it available to go through rest of pipeline.

In this layer, our task is to do magic with data, as now data is ready we only have to route the data to different destinations.

In this main layer, the focus is to specialize Data Pipeline processing system or we can say the data we have collected by the last layer in this next layer we have to do processing on that data.

**Big Data Batch Processing System**

A simple batch processing system for offline analytics. For doing this tool used is Apache Sqoop.

**What is Apache Sqoop?**

It efficiently transfers bulk data between Apache Hadoop and structured datastores such as relational databases. Apache Sqoop can also be used to extract data from Hadoop and export it into external structured data stores.

Apache Sqoop works with relational databases such as Teradata, Netezza, Oracle, MySQL, Postgres, and HSQLDB.

**Functions of Apache Sqoop**

* Import sequential data sets from mainframe
* Data imports
* Parallel Data Transfer
* Fast data copies
* Efficient data analysis
* Load balancing

**Near Real-Time Processing System**

A pure online processing system for online analytics. For this type of processing [**Apache Storm**](http://storm.apache.org/) is used**.** The Apache Storm cluster makes decisions about the criticality of the event and sends the alerts to the warning system (dashboard, e-mail, other monitoring systems).

**What is Apache Storm?**

It is a system for processing streaming data in real time. It adds reliable real-time data processing capabilities to Enterprise Hadoop. Storm on YARN is powerful for scenarios requiring real-time analytics, machine learning and continuous monitoring of operations.

**6 Key Features of Apache Storm**

* **Fast** – It can process one million 100 byte messages per second per node.
* **Scalable** – It can do parallel calculations that run across a cluster of machines.
* **Fault-tolerant** – When workers die, Storm will automatically restart them. If a node dies, the worker will be restarted on another node.
* **Reliable** – Storm guarantees that each unit of data (tuple) will be processed at least once or exactly once. Messages are only replayed when there are failures.
* **Easy to operate** – It consists of Standard configurations that are suitable for production on day one. Once deployed, Storm is easy to work.
* **Hybrid Processing system -** This consist of Batch and Real-time processing System capabilities. For this type of processing tool used is Apache Spark and Apache Flink.

**What is Apache Spark?**

[Apache Spark](https://spark.apache.org/) is a fast, in-memory data processing engine with elegant and expressive development APIs to allow data workers to efficiently execute streaming, machine learning or SQL workloads that require fast iterative access to data sets.

With Spark running on Apache Hadoop YARN, developers everywhere can now create applications to exploit Spark’s power, derive insights, and enrich their data science workloads within a single, shared data set in Hadoop.

**Real-Time Processing System**

**What is Apache Flink?**

Apache Flink is an open-source framework for distributed stream processing that Provides results that are accurate, even in the case of out-of-order or late-arriving data. Some of its features are -

* It is stateful and fault-tolerant and can seamlessly recover from failures while maintaining exactly-once application state.
* Performs at large scale, running on thousands of nodes with excellent throughput and latency characteristics.
* It's streaming data flow execution engine, APIs and domain-specific libraries for Batch, Streaming, Machine Learning, and Graph Processing.

**Apache Flink Use Cases**

* Optimization of e-commerce search results in real-time
* Stream processing-as-a-service for data science teams
* Network/Sensor monitoring and error detection
* ETL for Business Intelligence Infrastructure

**Big Data Storage Layer**

Next, the major issue is to keep data in the right place based on usage. We have relational Databases that were a successful place to store our data over the years.

But with the new big data strategic enterprise applications, you should no longer be assuming that your persistence should be relational.

We need different databases to handle the different variety of data, but using different databases creates overhead. That's why there is an introduction to the new concept in the database world, i.e., the **Polyglot Persistence**.

**What is Polyglot Persistence?**

Polyglot persistence is the idea of using multiple databases to power a single application. Polyglot persistence is the way to share or divide your data into multiple databases and leverage their power together.

It takes advantage of the strength of different database. Here various types of data are arranged in a variety of ways. In short, it means picking the right tool for the right use case.

It’s the same idea behind Polyglot Programming, which is the idea that applications should be written in a mix of languages to take advantage of the fact that different languages are suitable for tackling different problems.

**Advantages of Polyglot Persistence -**

* **Faster response times -** Here we leverage all the features of databases in one app, which makes the response times of your app very quickly.
* **Helps your app to scale well -** Your app scales exceptionally well with the data. All the NoSQL databases scale well when you model databases correctly for the data that you want to store.
* **A rich experience -** You have a vibrant experience when you harness the power of multiple databases at the same time. For example, if you want to search for Products in an e-commerce app, then you use ElasticSearch, which returns the results based on relevance, which MongoDB cannot do.

**Big Data Storage Tools**

**HDFS : Hadoop Distributed File System**

* [HDFS](https://hadoop.apache.org/docs/r1.2.1/hdfs_design.html) is a Java-based file system that provides scalable and reliable data storage, and it was designed to span large clusters of commodity servers.
* HDFS holds a huge amount of data and provides easier access.
* To store such massive data, the files are stored on multiple machines. These files are stored redundantly to rescue the system from possible data losses in case of failure.
* HDFS also makes applications available for parallel processing. HDFS is built to support applications with large data sets, including individual files that reach into the terabytes.
* It uses a master/slave architecture, with each cluster consisting of a single NameNode that manages file system operations and supporting DataNodes that manage data storage on individual compute nodes.
* When HDFS takes in data, it breaks the information down into separate pieces and distributes them to different nodes in a cluster, allowing for parallel processing.
* The file system also copies each piece of data multiple times and distributes the copies to individual nodes, placing at least one copy on a different server rack
* HDFS and YARN from the data management layer of Apache Hadoop.

**Features of HDFS**

* It is suitable for the distributed storage and processing.
* Hadoop provides a command interface to interact with HDFS.
* The built-in servers of name node and data node help users to quickly check the status of the cluster.
* Streaming access to file system data.
* HDFS provides file permissions and authentication.

**GlusterFS: Dependable Distributed File System**

As we know good storage solution must provide elasticity in both storage and performance without affecting active operations.

Scale-out storage systems based on GlusterFS are suitable for unstructured data such as documents, images, audio and video files, and log files. [GlusterFS](https://www.gluster.org/) is a scalable network filesystem.

Using this, we can create large, distributed storage solutions for media streaming, data analysis, and other data- and bandwidth-intensive tasks.

* It's Open Source.
* You can deploy GlusterFS with the help of commodity hardware servers.
* Linear scaling of performance and storage capacity.
* Scale storage size up to several petabytes, which can be accessed by thousands for servers.

**GlusterFS Use Cases**

* Cloud Computing
* Streaming Media
* Content Delivery

**Amazon S3 Storage Service**

* Amazon Simple Storage Service (Amazon S3) is object storage with a simple web service interface to store and retrieve any amount of data from anywhere on the internet.
* It is designed to deliver 99.999999999% durability, and scale past trillions of objects worldwide.
* Customers use S3 as primary storage for cloud-native applications; as a bulk repository, or "data lake," for analytics; as a target for backup & recovery and disaster recovery; and with serverless computing.
* It's simple to move large volumes of data into or out of S3 with Amazon's cloud data migration options.
* Once data is stored on Amazon S3, it can be automatically tiered into lower cost, longer-term cloud storage classes like S3 Standard - Infrequent Access and Amazon Glacier for archiving.

**Big Data Query Layer**

It is the layer where active analytic processing takes place. This is a field where interactive queries are necessaries, and it’s a zone traditionally dominated by SQL expert developers. Before Hadoop, we had an insufficient storage due to which it takes long analytics process.

At first, it goes through a Lengthy process, i.e., ETL to get a new data source ready to be stored and after that, it puts the data in database or data warehouse. But now, data analytics became essential step which solved problems while computing such a large amount of data.

Companies from all industries use big data analytics to -

* Increase revenue
* Decrease costs
* Increase productivity

**Big Data Analytics Query Tools**

* **Apache Hive Architecture**

[Apache Hive](https://hive.apache.org/) is data warehouse infrastructure built on top of Apache Hadoop for providing data summarization, ad-hoc query, and analysis of large datasets.

Data analysts use Hive to query, summarize, explore and analyze that data, then turn it into actionable business insight.

It provides a mechanism to project structure onto the data in Hadoop and to query that data using a SQL - like a language called HiveQL (HQL).

**Features of Apache Hive**

* Query data with a SQL - based language.
* Interactive response times, even over massive datasets.
* It's scalable as data variety and volume grows, more commodity machines can be added, without a corresponding reduction in performance Works with traditional data integration and data analytics tools.
* **Apache Spark SQL**

Spark SQL includes a cost-based optimizer, columnar storage, and code generation to make queries fast.

At the same time, it scales to thousands of nodes and multi-hour queries using the Spark engine, which provides full mid-query fault tolerance.

Spark SQL is a Spark module for structured data processing. Some of the Functions performed by Spark SQL are -

* The interfaces provided by Spark SQL provide Spark with more information about the structure of both the data and the computation being performed.
* Internally, Spark SQL uses this extra information to perform additional optimizations.
* One use of Spark SQL is to execute SQL queries.
* Spark SQL can also be used to read data from an existing Hive installation.
* **Amazon Redshift**

[Amazon Redshift](https://aws.amazon.com/redshift/) is a fully managed, petabyte-scale data warehouse service in the cloud. We use Amazon Redshift to load the data and run queries on the data.

We can also create additional databases as needed by running a SQL command. Most important we can scale it from hundred gigabytes of data to a petabyte or more.

It enables you to use your data to acquire new insights for your business and customers. The Amazon Redshift service manages all of the work of setting up, operating and scaling a data warehouse.

These tasks include provisioning capacity, monitoring and backing of the cluster, and applying patches and upgrades to the Amazon Redshift engine.

* **Presto - SQL Query Engine For Big Data**

[Presto](https://prestodb.io/) is an open source distributed SQL query engine for running interactive analytic queries against data sources of all sizes ranging from gigabytes to petabytes.

Presto was designed and written for interactive analytics and approaches and the speed of commercial data warehouses while scaling to the size of organizations like Facebook.

**Presto Capabilities**

* Presto allows querying data where it lives, including Hive, [Cassandra](http://cassandra.apache.org/), relational databases or even proprietary data stores.
* A single Presto query can combine data from multiple sources, allowing for analytics across your entire organization.
* Presto is targeted at analysts who expect response times ranging from sub-second to minutes.
* Presto breaks the false choice between having fast analytics using an expensive commercial solution or using a slow "free" solution that requires excessive hardware.

**Who Uses Presto?**

* Facebook uses Presto for interactive queries against several internal data stores, including their 300PB Data Warehouse. Over 1,000 Facebook employees use Presto daily to run more than 30,000 queries that in the complete scan over a petabyte each per day.
* Leading internet companies including Airbnb and Dropbox are using Presto