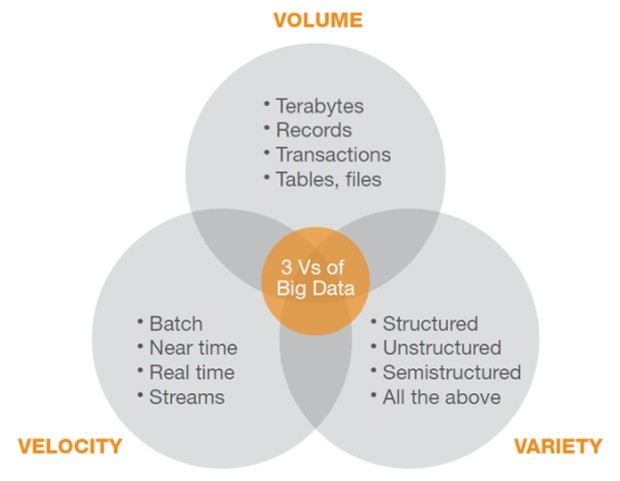
1. **Describe the characteristics of Big data in detail.**

****

**1) Volume** (amount of data the size of the data set):

Volume Refers to the vast amounts of data generated every second. We are not talking Terabytes but Zettabytes or Brontobytes. If we take all the data generated in the world between the beginning of time and 2008, the same amount of data will soon be generated every minute. This makes most data sets too large to store and analyze using traditional database technology. New big data tools use distributed systems so that we can store and analyze data across databases that are dotted around anywhere in the world.

90% of all data ever created, was created in the past 2 years. From now on, the amount of data in the world will double every two years. By 2020, we will have 50 times the amount of data as that we had in 2011. The sheer volume of the data is enormous and a very large contributor to the ever expanding digital universe is the Internet of Things with sensors all over the world in all devices creating data every second. The era of a trillion sensors is upon us.

If we look at airplanes they generate approximately 2.5 billion Terabyte of data each year from the sensors installed in the engines. Self-driving cars will generate 2 Petabyte of data every year. Also the agricultural industry generates massive amounts of data with sensors installed in tractors. Shell uses super-sensitive sensors to find additional oil in wells and if they install these sensors at all 10,000 wells they will collect approximately 10 Exabyte of data annually. That again is absolutely nothing if we compare it to the Square Kilometer Array Telescope that will generate 1 Exabyte of data per day.

In the past, the creation of so much data would have caused serious problems. Nowadays, with decreasing storage costs, better storage solutions like Hadoop and the algorithms to create meaning from all that data this is not a problem at all.

**2) Velocity** (speed of data in and out or data in motion):  
 Velocity Refers to the speed at which new data is generated and the speed at which data moves around. Just think of social media messages going viral in seconds. Technology allows us now to analyze the data while it is being generated (sometimes referred to as in-memory analytics), without ever putting it into databases.

The Velocity is the speed at which the data is created, stored, analyzed and visualized. In the past, when batch processing was common practice, it was normal to receive an update from the database every night or even every week. Computers and servers required substantial time to process the data and update the databases. In the big data era, data is created in real-time or near real-time. With the availability of Internet connected devices, wireless or wired, machines and devices can pass-on their data the moment it is created.

The speed at which data is created currently is almost unimaginable: Every minute we upload 100 hours of video on YouTube. In addition, every minute over 200 million emails are sent, around 20 million photos are viewed and 30,000 uploaded on Flickr, almost 300,000 tweets are sent and almost 2.5 million queries on Google are performed.

The challenge organizations have is to cope with the enormous speed the data is created and used in real-time.

**3) Variety** (range of data types, domains and sources)  
 Variety Refers to the different types of data we can now use. In the past we only focused on structured data that neatly fitted into tables or relational databases, such as financial data. In fact, 80% of the world’s data is unstructured (text, images, video, voice, etc.) With big data technology we can now analyze and bring together data of different types such as messages, social media conversations, photos, sensor data, video or voice recordings.

In the past, all data that was created was structured data, it neatly fitted in columns and rows but those days are over. Nowadays, 90% of the data that is generated by an organization is unstructured data. Data today comes in many different formats: structured data, semi-structured data, unstructured data and even complex structured data. The wide variety of data requires a different approach as well as different techniques to store all raw data.

There are many different types of data and each of those types of data require different types of analyses or different tools to use. Social media like Facebook posts or Tweets can give different insights, such as sentiment analysis on your brand, while sensory data will give you information about how a product is used and what the mistakes are.

1. **Explain the possible solutions to handle Big data.**

Big data has great potential to produce useful information for companies which can benefit the way they manage their problems. Big data analysis is becoming indispensable for automatic discovering of intelligence that is involved in the frequently occurring patterns and hidden rules. These massive data sets are too large and complex for humans to effectively extract useful information without the aid of computational tools. Emerging technologies such as the Hadoop framework and MapReduce offer new and exciting ways to process and transform big data, defined as complex, unstructured, or large amounts of data, into meaningful knowledge.

Hadoop is a scalable, open source, fault tolerant Virtual Grid operating system architecture for data storage and processing. It runs on commodity hardware, it uses HDFS which is fault-tolerant high bandwidth clustered storage architecture. It runs MapReduce for distributed data processing and is works with structured and unstructured data. For handling the velocity and heterogeneity of data, tools like Hive, Pig and Mahout are used which are parts of Hadoop and HDFS framework. Hadoop and HDFS (Hadoop Distributed File System) by Apache is widely used for storing and managing big data.

**Hadoop Distributed File System (HDFS)**

HDFS is a distributed file system which is in particular designed and optimized for processing large data volumes and for highest availability. It spreads across the local storage of a cluster consisting of many server nodes.

HDFS includes a master server (NameNode) that partitions the original data and assign the data to the server nodes, based on defined rules. Each slave node of the cluster stores just a small fragment of the complete data set. The DataNode which is installed on each slave node is in charge of the local data management.

Each data block is replicated on more than one server for the purpose of high availability. By default, every data block exists three times. Besides the primary data block, one copy typically exists on a server in the same rack, while an additional copy will be on a server in another rack. To increase availability even more, data can be distributed to different locations. Of course, this will not protect against human errors when copying or deleting data; for this purpose, additional backup processes need to be applied.

The HDFS NameNode manages the metadata, thus being always aware of which data blocks belong to which files, where the data blocks are located, and where which storage capacities are occupied. By means of periodically transmitted signals, the NameNode will always know which DataNodes are still working. If the signal is missing, the NameNode will recognize the failure of a DataNode, will remove the failed DataNode from the Hadoop cluster, and will always try to distribute data load evenly across the available DataNodes. Furthermore, the NameNode ensures that the defined number of data copies is always available.

**Hadoop MapReduce**

Similar to HDFS, the MapReduce framework works according to the master-slave principle. The master (JobTracker) divides a given problem (job) into multiple tasks (map tasks), and distributes these tasks across the network to a number of slave nodes (TaskTracker) for parallel processing.

Typically, the map tasks run on the same cluster nodes, where the processed data resides. If that server node is already heavily loaded, another node will be selected which is close to the data, i.e. preferably a node in the same rack.

Intermediate results will be exchanged among the nodes (shuffling), and thereafter merged by the reduce tasks to a final result. As the processing tasks are moved to the data and not vice versa, in the map phase basically all I/O activities can be parallelized while network load is almost completely avoided. To avoid any bottleneck regarding scalability in the shuffle phase, a non-blocking switched network without or with only low overprovisioning between the server nodes is required.

Optionally, intermediate results of the map phase and the shuffle phase may be aggregated (combine), in other to keep data volumes transferred from map tasks to reduced tasks as low as possible.

While the input data for MapReduce as well as final results reside in HDFS, the intermediate results are deposited in the local file systems of the DataNodes.

The JobTracker is steadily in touch with the TaskTrackers, monitors the slave nodes and takes care that disrupted or aborted tasks will be executed anew.

If a task does not notify any progress for a longer period, or if a slave node fails completely, all tasks not terminated yet will be restarted on another server, usually on a server with a respective copy of the data. If a task runs extremely slowly, the JobTracker will also restart the task on another server in order to execute the overall job in good time (speculative execution).

The only weak spot is the JobTracker itself, because it represents a Single Point of Failure. Hence, this server should include as many redundant components as possible, in order to keep the probability of failure on a low level.

It is true that MapReduce can be directly applied for executing business analytics. But a frequent use case is to transform data into an optimized shape for analytics.

**YARN (Yet Another Resource Negotiator):**

YARN is a further development of the MapReduce framework. Cluster resource management and application control which in MapReduce v1 are both covered by the JobTracker, are split up into separated instances. It is just the lean ResourceManager that still exists as a central instance, while the entire job control is delegated to the ApplicationMaster which can run on any slave node. For every job an ApplicationMaster is dynamically created; it is exclusively available for this job. MapReduce v2 will then only look after the parallel data processing.

The distributed application control increases the level of parallelization, removes bottlenecks and enables clusters with 10,000s of nodes.

1. **Explain the differences between scaling up and scaling out.**

|  |  |
| --- | --- |
| **Scaling up** | **Scaling out** |
| This means increasing the size of one physical machine by increasing the amount of resources (RAM/CPU) available for processing. | This means combining multiple independent nodes/computers into one system. |
| It is expensive. | It is cheap. |
| It does make it easier to control a system, and to provide for certain data quality issues. | It does make it complex to control a system, and to provide for certain data quality issues. |
| Scaling up generally refers to purchasing and installing a more capable central control or piece of hardware. For example, when a project’s input/output demands start to push against the limits of an individual server, a scaling up approach would be to buy a more capable server with more processing capacity and RAM. | By contrast, scaling out means linking together other lower-performance machines to collectively do the work of a much more advanced one. With these types of distributed setups, it's easy to handle a larger workload by running data through different system trajectories. |