**Big Data**

Herramientas, tecnologías y procesos utilizados para recolección, almacenamiento, proceso y análisis de grandes volúmenes de datos. Conjunto de datos es tan grande y complejo que los medios tradicionales de procesamiento son ineficaces.

•  Alta escalabilidad • Procesamiento paralelo • Baja latencia • Datos no estructurados y estructurados • NoSQL

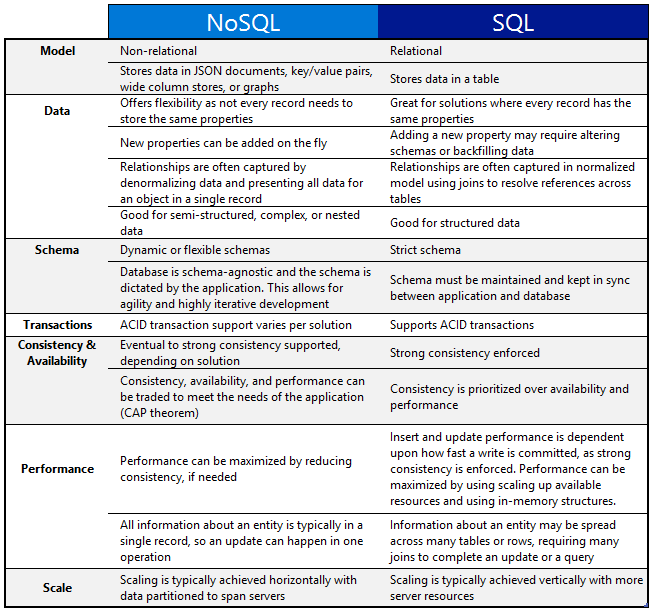
**Volumen**: grandes volúmenes de información.  Tan grande que se deben almacenar de forma distribuida.

**Variedad**: información de tipos muy diversos.

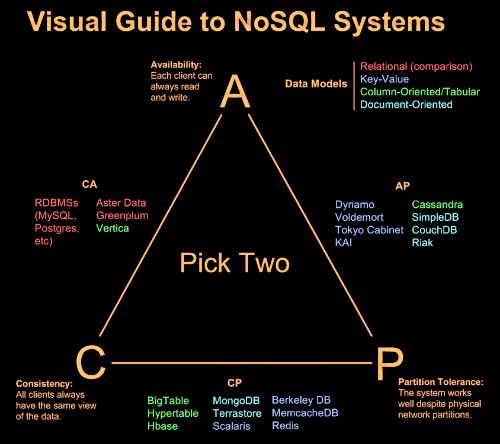
**Velocidad**: velocidad con la que se genera la información. Tal que se debe analizar de forma paralela y distribuida.

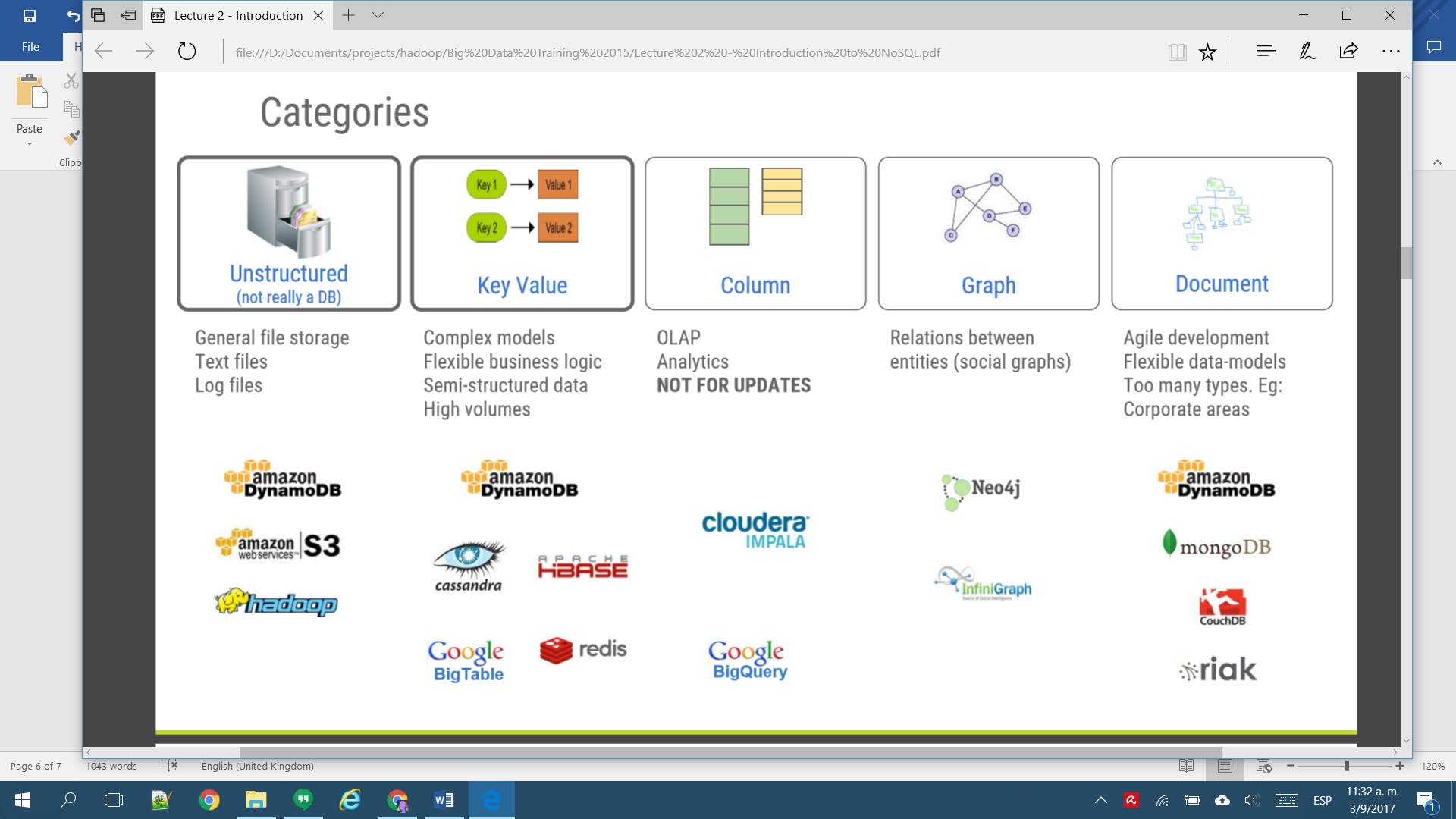
**NoSQL**

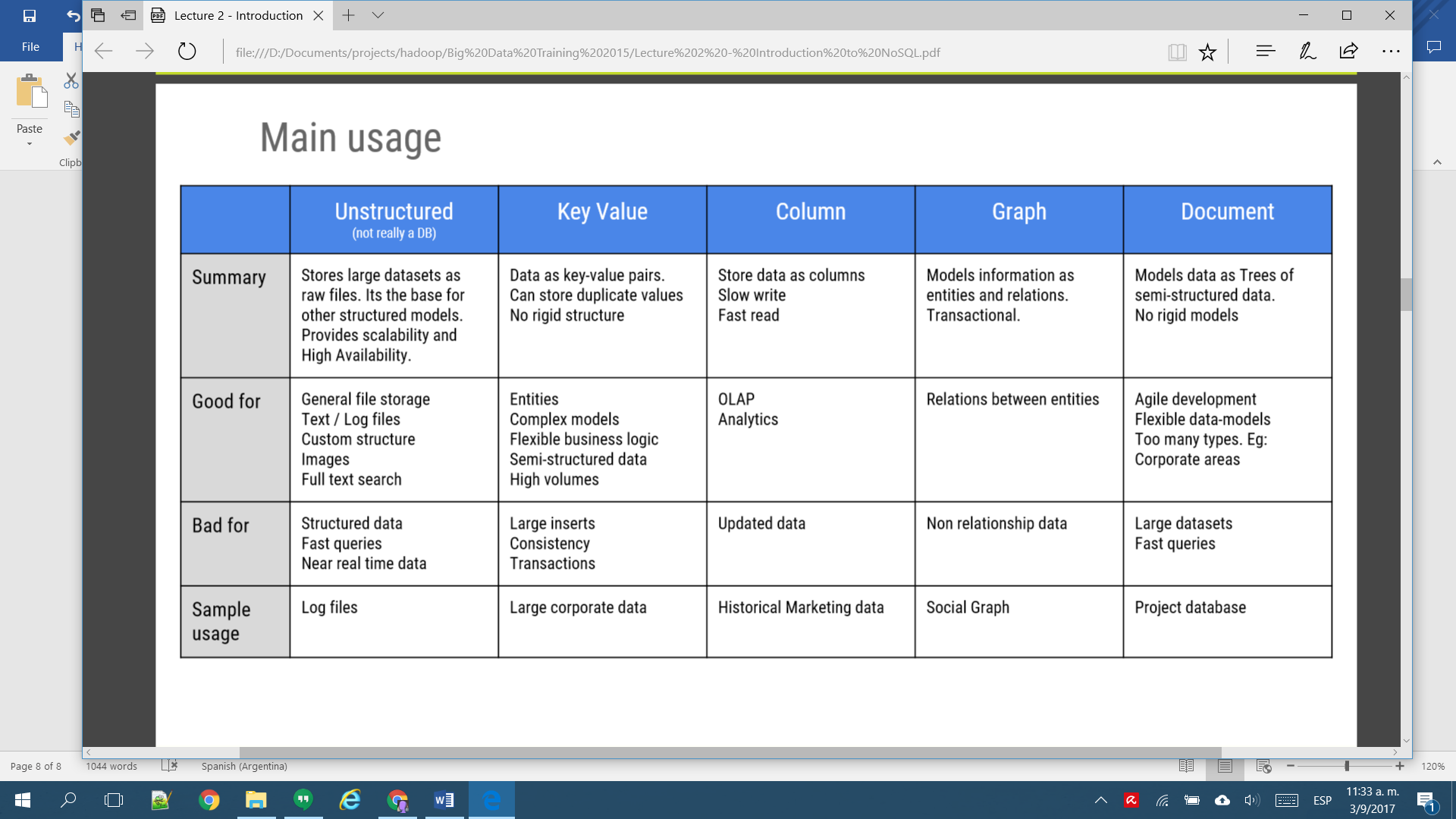
NoSQL vs Relational:



Teorema CAP: sistema puede tener no más de dos de estas tres características simultáneamente.



Clasificación NoSQL por estructuras de datos



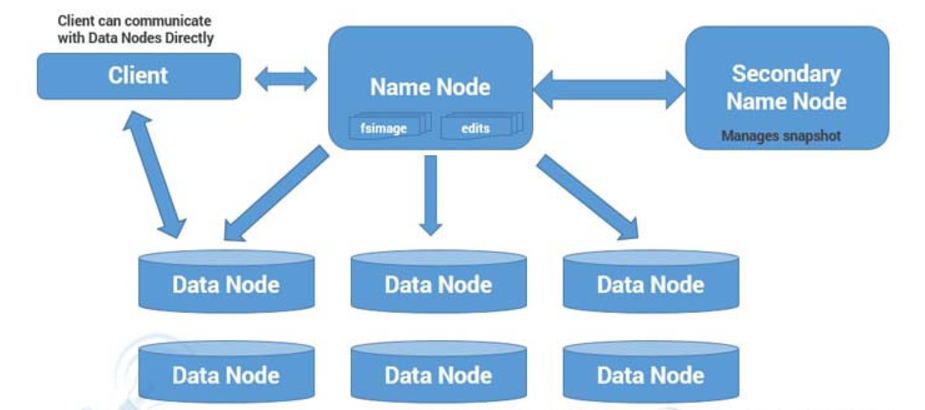
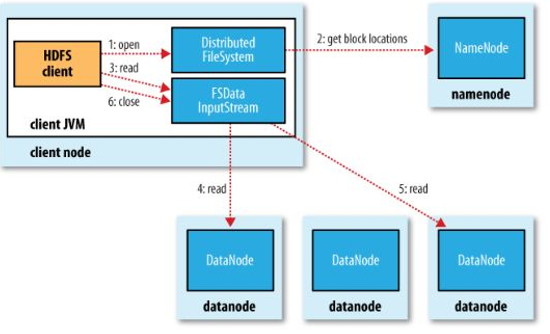
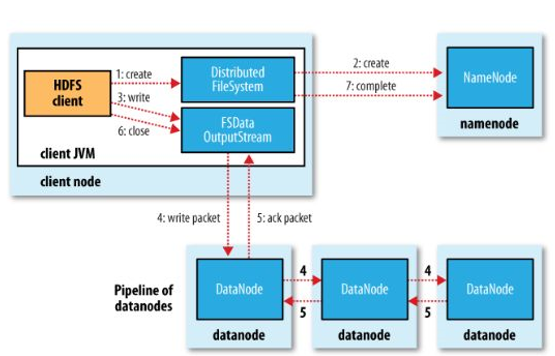
**Hadoop**

*open-source* software for reliable, *scalable*, *fault-tolerant*, a *framework* that allows for the *distributed* *storing* (HDFS)and *processing* (MR/YARN) of large data sets across clusters of commodity hardware.

Hadoop Common, que proporciona acceso a los sistemas de archivos soportados por Hadoop. El paquete de software The Hadoop Common contiene los archivos .jar y los scripts necesarios para ejecutar Hadoop.

Hadoop Distributed File System (HDFS) es un sistema de archivos distribuido, escalable y portátil. HDFS almacena archivos particionándolos en blocks (of 64 or 128 MB size) distribuidos en múltiples máquinas.

(HA) Fiabilidad replicado de datos a través de múltiples hosts, por defecto, los datos se almacenan en 3 nodos: dos en el mismo rack, y otro en un rack distinto.

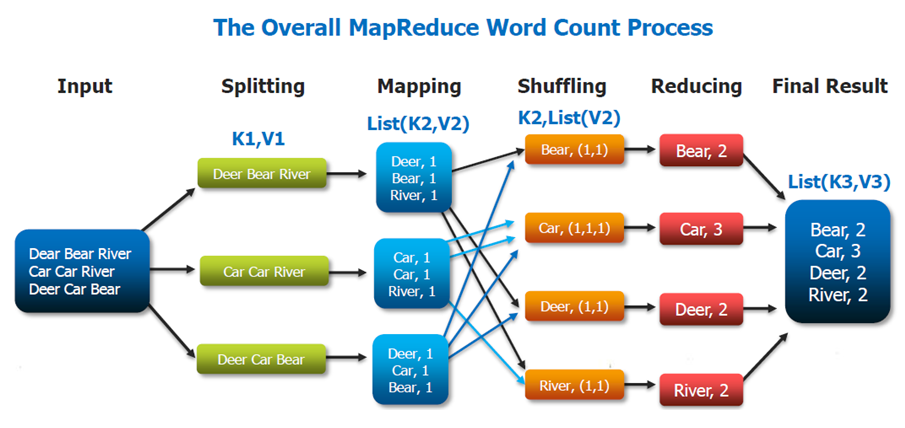


*Name Node*: single instance. Responsible for manage metadata about files distributed across the cluster: location of file blocks across cluster, permission. Metadata stored in file named fsimage, updated and kept in memory of NN. Changes in one file called edits as edit logs. SPOF in H1.

*Secondary Name Node*: manages the metadata for the Name Node. In the sense, it reads the information written in edit logs (by Name Node) and creates an updated file of current cluster metadata. Then it transfers that file back to Name Node so that fsimage file can be updated

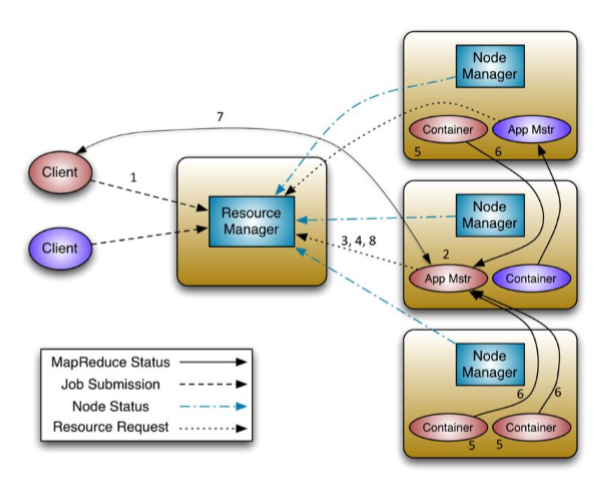
*Data Node*: responsible for storing the individual file blocks on the slave nodes in Hadoop cluster. Periodically sends heart bits to Name Node to make Name Node aware that slave process is running.

MapReduce is the algorithm of executing any task on distributed system. Using MapReduce one can process a large file in parallel manner. MapReduce framework executes any task on different nodes as full file is distributed across the cluster in a form of various blocks.



YARN is the resource manager (negotiator). Has three main components:

*ResourceManager*: responsible for getting job submitted from client and schedule it on cluster, monitoring running jobs on cluster and allocating proper resources on the slave node. Uses two other processes named *Application Manager* (Negotiating first container for executing application specific task with suitable ApplicationMaster on slave node)and *Scheduler* (Scheduling the job execution as per submission request received by ResourceManager , Allocating resources to applications submitted to the cluster, Coordinating with ApplicationManager daemon and keeping track of resources of running applications)

*NodeManager*: responsible for coordinating with Resource Manager for task scheduling and tracking the resource utilization on the slave node, also reports the resource utilization back to the Resource Manager

*ApplicationMaster:* instance of this daemon is per application. Negotiating suitable resource containers on slave node from ResourceManager.

Limitations of Hadoop 1.x Architecture

1. 1.x – has a Single Point of Failure as there is no backup Name Node.
2. Job scheduling, resource management and job monitoring are being done by Job Tracker which is tightly coupled with Hadoop. So Job Tracker is not able to manage resources outside Hadoop. Job Tracker has to coordinate with all task tracker so in a very big cluster it will be difficult to manage huge number of task trackers altogether.
3. Due to single Name Node, there is no concept of namespaces in Hadoop 1.x. Everything is being managed under single namespace.
4. In Hadoop 1.x, Hadoop Cluster can be scaled upto ~4000 nodes. Scalablity beyond that may cause performance degradation and increasing task failure ratio.
5. Hadoop 2.x has some common Hadoop API which can easily be integrated with any third party applications to work with Hadoop
6. New architecture has added the architectural features like HDFS High Availability (Name Node HA: StandBy Name Node) and HDFS Federation
7. Hadoop 2.x not using Job Tracker and Task Tracker daemons for resource management now on-wards, it is using YARN (Yet Another Resource Negotiator) for Resource Management

Hadoop not good for:

1. Low-latency Data Access: Applications that require real-time query, and low-latency access to data in tens of milliseconds will not work well with Hadoop.
2. Structured Data: Hadoop is not fit for structured data with strong relationship. Hadoop works well for semi-structured and unstructured data. It stores data in files, doesn’t index them like
3. When data isn’t that big
4. Too many small files: When there are too many small files, the NameNode will hit its memory limit where the block map and the metadata are hosted. And to handle the NameNode bottleneck, Hadoop introduces HDFS Federation.
5. MapReduce may not the best choice

**Sqoop**

Tool designed to transfer data between Hadoop and relational databases. You can use Sqoop to import data from a relational database management system (RDBMS) such as MySQL or Oracle into the Hadoop Distributed File System (HDFS), transform the data in Hadoop MapReduce, and then export the data back into an RDBMS.

Uses MapReduce to import and export the data, which provides parallel operation as well as fault tolerance.

Import process is a database table. Sqoop will read the table row-by-row into HDFS. The output of this import process is a set of files containing a copy of the imported table. The import process is performed in parallel. Each row from a table is represented as a separate record in HDFS. Records can be stored as text, or in binary representation as Avro or SequenceFiles.