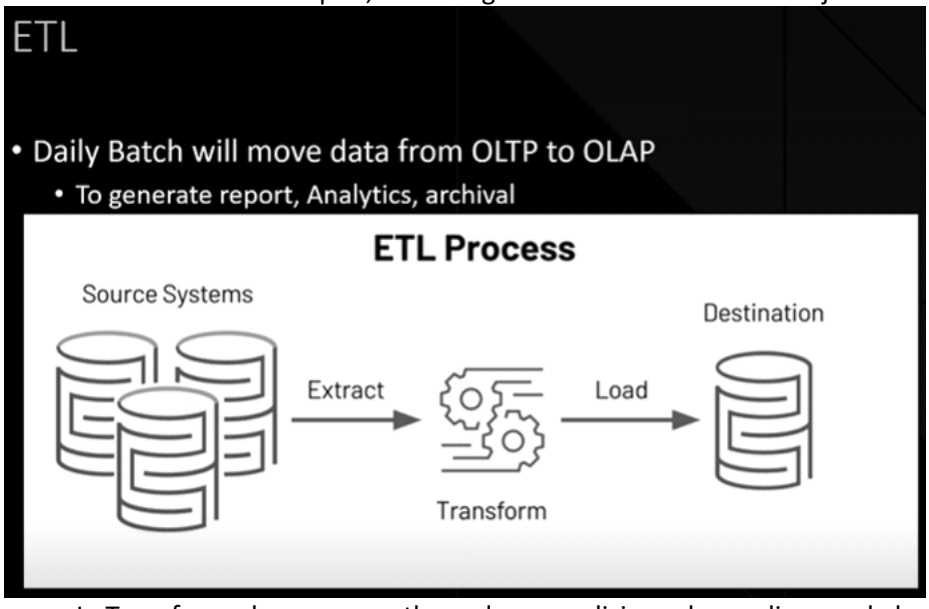
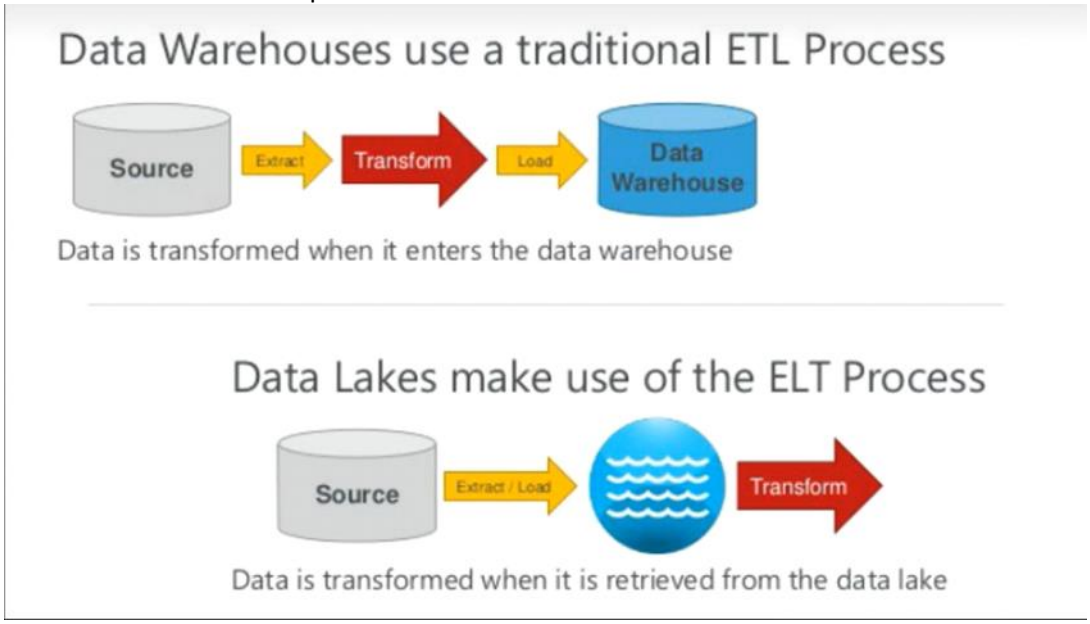
Spark Introduction

Wednesday, March 23, 2022 12:25 PM

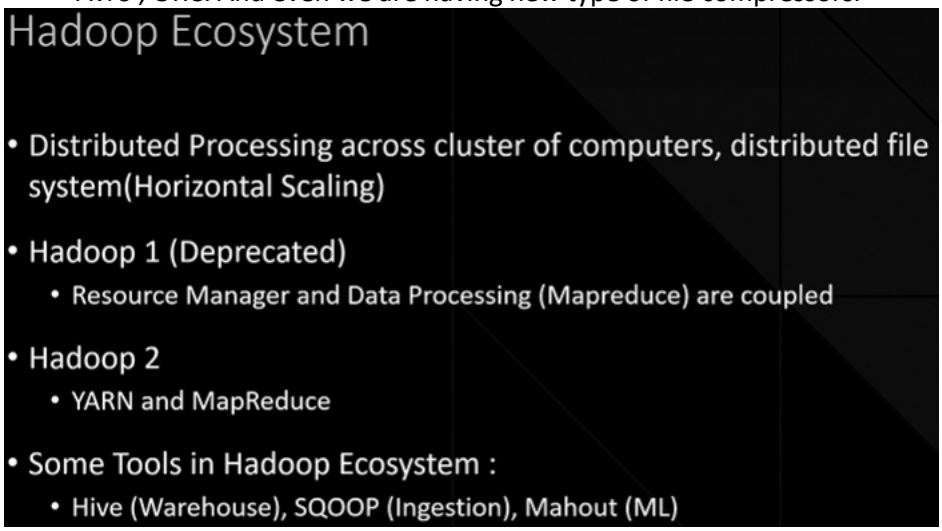
- Data engineering means designing and building systems for collecting , storing , analyzing data. It's main tasks are data pipeline creation , data validation and data quality and other tasks like data archival.
- Data Engineering skills (Programming languages like Scala and python, SQL languages like RDBS or NoSQL , Cloud tools)
- We connect database to the front end using local host number and specific connection username and password.
- OLTP stores only transactional data, it doesn't store any other un important details.
- To generate the report oriented data we use OLAP database. OLAP is used to store the summary type of the dataset.
- In OLTP they do multiple vertical bisection to split the data.
- In transactional level at RDBMS we do normalization to stop the repetition data.
- In OLAP we try to minimizing the need for joins, reducing the number of tables, queries to be retrieved can be simpler, retrieving data is faster due to fewer joins.



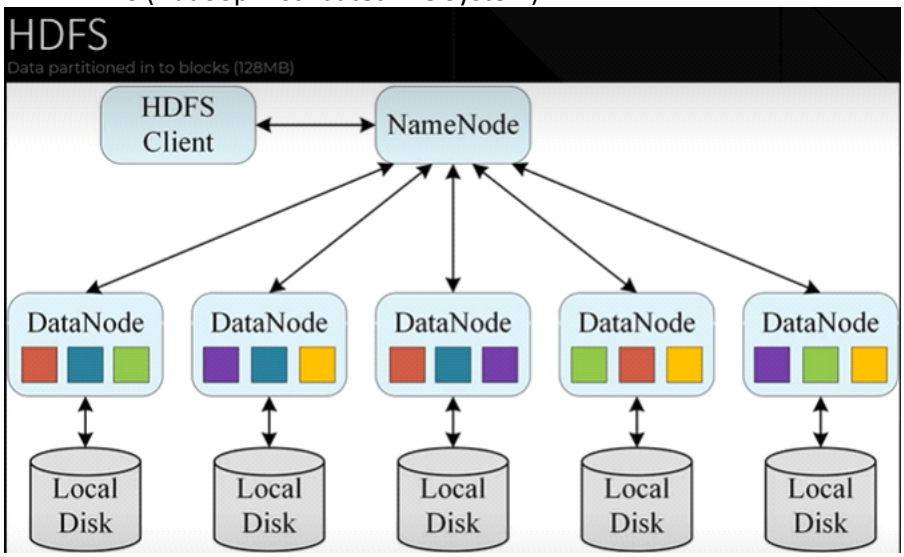
- In Transform phase we use those de-normalizing rules as discussed above.
- Data Lake: We have dumped raw data across the organization
- Data Warehouse: Here we will have data as summarized way. We will have current and historic data in this place.



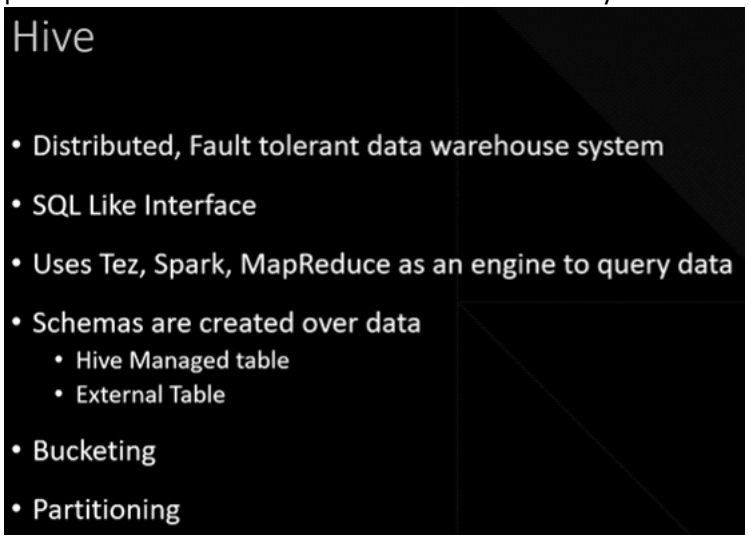
- New data formats that we are using now to share the data in big data operations are Parquet, Avro , ORC. And even we are having new type of file compressors.



- HDFS (Hadoop Distributed File System)



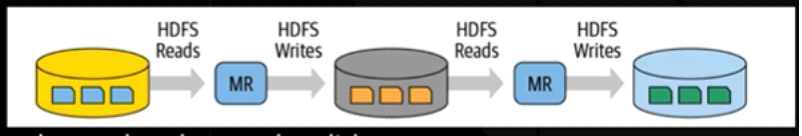
MapReduce: Map does the operations which happens inside a data block or partition (where we are having every record assigned with a key) And reduce does the aggregation part.
Cluster Manager: Manages cluster of computer and their resources like CPU, memory , storage, ports and other available resources. YARN is widely used cluster manager.



- If we make any change on hive managed table we change the original data, where as external table the original data will not be affected.

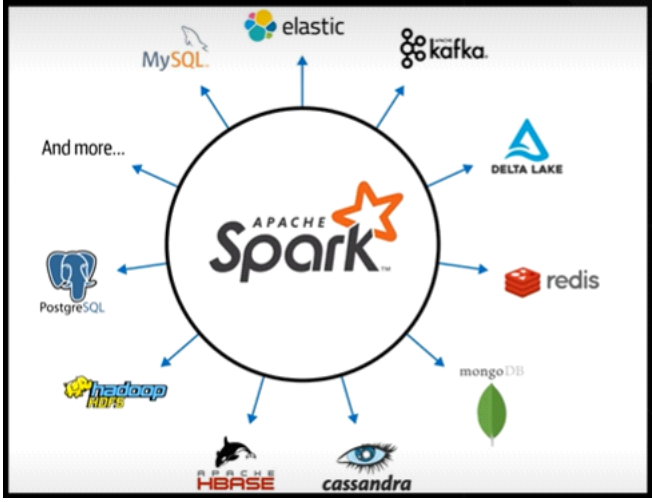
Problem with Hadoop

- No Caching



  - Intermediate results needs to be saved to disk
- No Support for Streaming

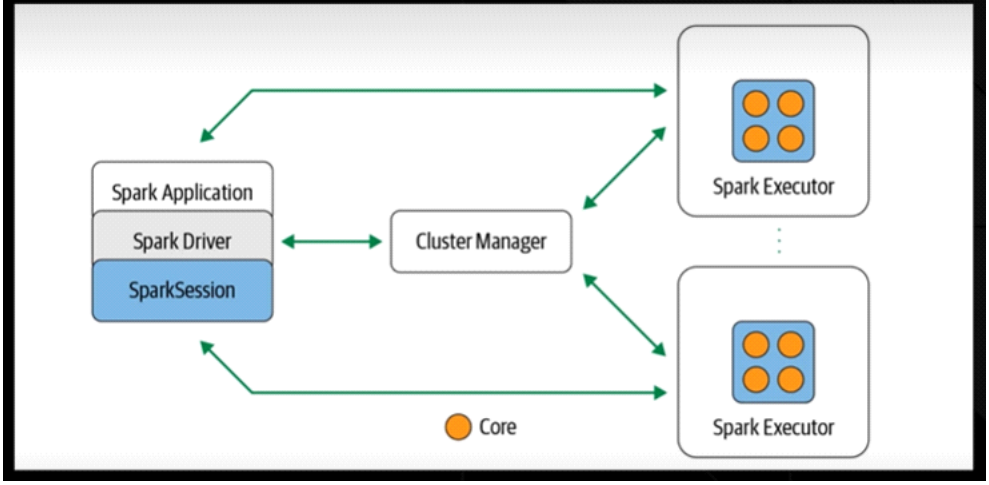
Apache Spark – Supported Datasources



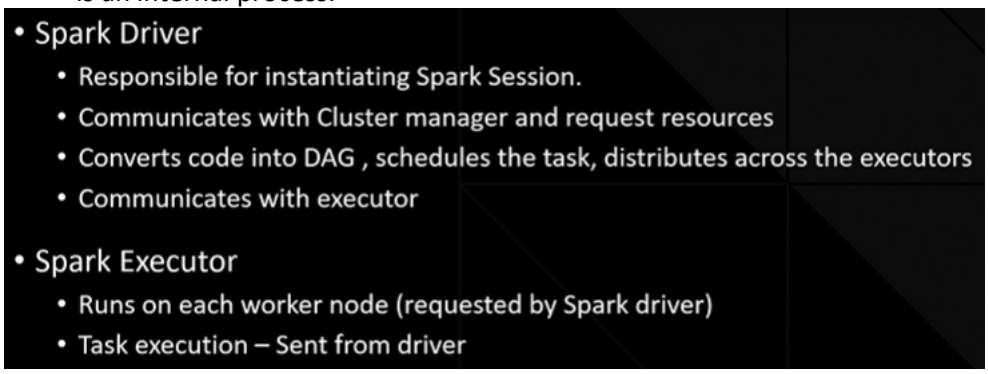
Spark App – Package & Execution

- Packaging
  - Java & Scala Spark – jar
  - PySpark – Zip file, Egg file
  - Can be executed in Notebooks
- Execution
  - Local mode – IDE Run
  - Cluster mode - Trigger Spark app with API/schedule with tools like oozie/airflow

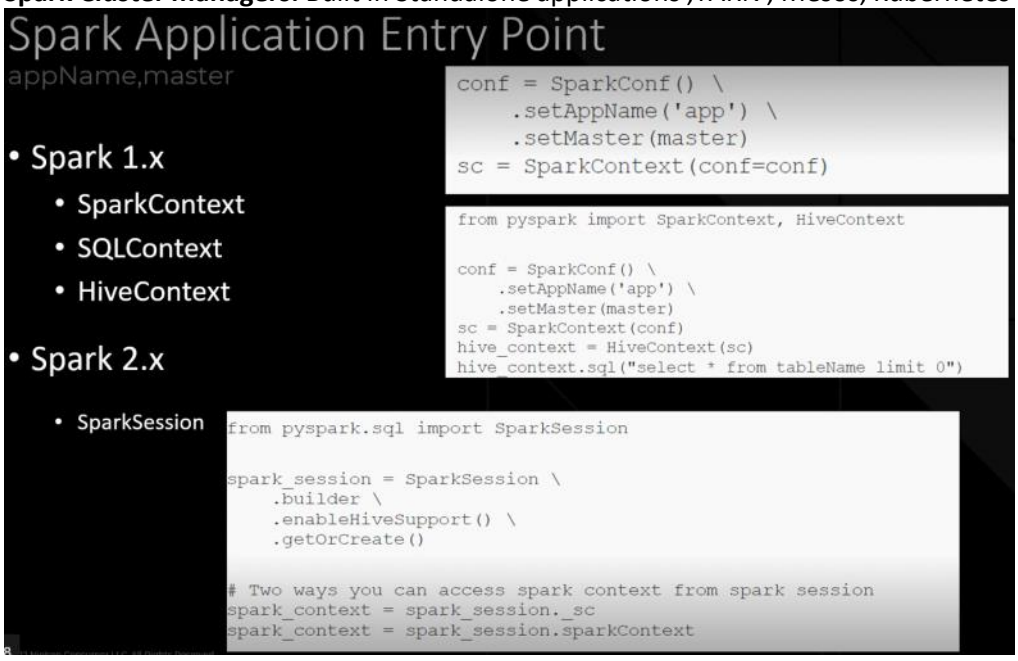
Spark Architecture

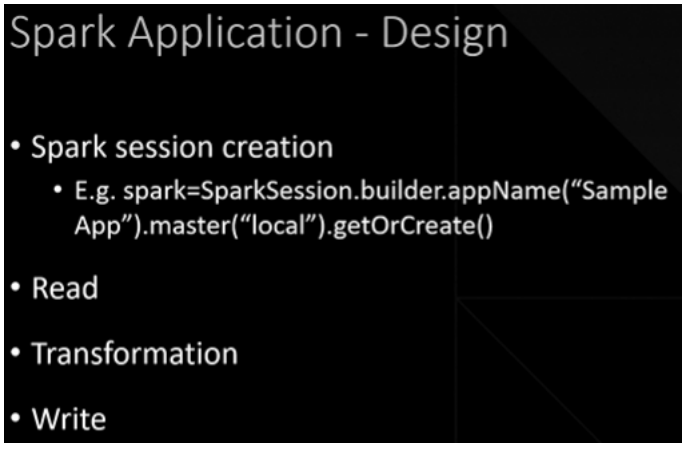


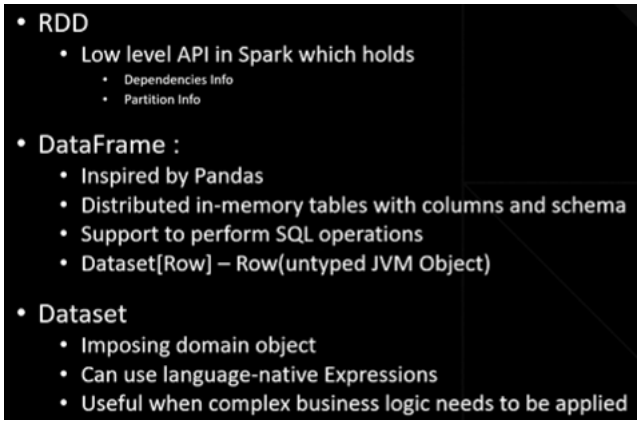
- Driver instructs the executor with the help of cluster manager. The working of cluster manager is an internal process.



Spark Cluster managers: Built in Standalone applications ,YARN , Mesos, Kubernetes .







Transformations	Actions
orderBy()	show()
groupBy()	take()
filter()	count()
select()	collect()
join()	save()

- Can be created from sc (SparkContext)
  - sc.parallelize(List()/Seq())
  - sc.textFile()
  - sc.wholeTextFiles()
- Or from <dataframe>.rdd
- RDD should be use when we want to use accumulator (counter) and broadcast variables.
- Benefits from Spark Optimizations aren't available.

RDD: Row delimiter

RDD Transformations

Transformation	Meaning
map(*func*)	Return a new distributed dataset formed by passing each element of the source through a function *func*.
filter(*func*)	Return a new dataset formed by selecting those elements of the source on which *func* returns true.
flatMap(*func*)	Similar to map, but each input item can be mapped to 0 or more output items (so *func* should return a Seq rather than a single item).
mapPartitions(*func*)	Similar to map, but runs separately on each partition (block) of the RDD, so *func* must be of type Iterator<T> => Iterator<U> when running on an RDD of type T.
coalesce(*numPartitions*)	Decrease the number of partitions in the RDD to numPartitions. Useful for running operations more efficiently after filtering down a large dataset.
repartition(*numPartitions*)	Reshuffle the data in the RDD randomly to create either more or fewer partitions and balance it across them. This always shuffles all data over the network.
reduceByKey(*func*, [*numPartitions*])	When called on a dataset of (K, V) pairs, returns a dataset of (K, V) pairs where the values for each key are aggregated using the given reduce function *func*, which must be of type (V,V) => V. Like in `groupByKey`, the number of reduce tasks is configurable through an optional second argument.

RDD Actions

Action	Meaning
reduce(*func*)	Aggregate the elements of the dataset using a function *func* (which takes two arguments and returns one). The function should be commutative and associative so that it can be computed correctly in parallel.
collect()	Return all the elements of the dataset as an array at the driver program. This is usually useful after a filter or other operation that returns a sufficiently small subset of the data.
count()	Return the number of elements in the dataset.
first()	Return the first element of the dataset (similar to take(1)).
take(*n*)	Return an array with the first *n* elements of the dataset.
foreach(*func*)	Run a function *func* on each element of the dataset. This is usually done for side effects such as updating an Accumulator or interacting with external storage systems.
saveAsTextFile(*path*)	Write the elements of the dataset as a text file (or set of text files) in a given directory in the local filesystem, HDFS or any other Hadoop-supported file system. Spark will call toString on each element to convert it to a line of text in the file.

Dataset samples