Types of Big Data

1. **Unstructured Data** : Images and videos data
2. **Structured Data** : RDBMS, Excel Data, Spreadsheet Data
3. **Semi-structured Data** : Xml, JSON, Csv, (ORC(256mb default), Parquet, AURO) -> Big data file format which is much more compressed and used for getting better query responses.
4. **Quasi-structured Data** : Web stream file , click stream file.

Spark help you to acheieve *Realtime processing* or *Stream processing*. *kafka* can generate data from any source at a regular interval.

Uber Old Infrastructure

1. Data Sources -> kafka, Key-val Db, RDBMS
2. Storagge -> Vertica (landing zone) : where the data is generally stored in the DB which can be acessed faster.
3. Processing -> ETL(adhoc analytics) : Extract Transform and Load to a permanent storage

Uber Infrastructure: Solution

- More data warehouse options such as Hive(warehouse), Presto and spark which enabled easy access data.
- Hadoop Upserts and Incremental(Hudi) was introduced, an open-source spark library.
- Hudi provies an abstraction layer on top of HDFS and **Parquet** (Columnar storage) to support the required update and delete operations.
- Hudi enables users to take out just modified data incrementally, boosting query efficiency.

Data processing

1. Bulk processing -> processing the data(**BULK**) and analysis.
2. Real-time processing ->

- take the data at regular intervals of time and process it.
- data time is marked and for each marked time eg: T1 , we do process for T1.
- Eg : Fraud Detection

Spark Advantages :

1. **Batch processing** : Spark batch can be used instead of Hadoop Map Reduce.
2. **Structured Data Analysis** : Spark DataFrames are simple and can be used for quick analysis of data.
3. **Machine Learning Analysis** : MLib can be used for clustering , recommendations and classification.
4. **Interactive SQL Analysis** : Spark SQL can be used over Stringer, Tez, or Impala.
5. **Real-time Streaming Data Analysis** : Spark Streaming can be used instead of specialized libraries, such as storm.

Features :

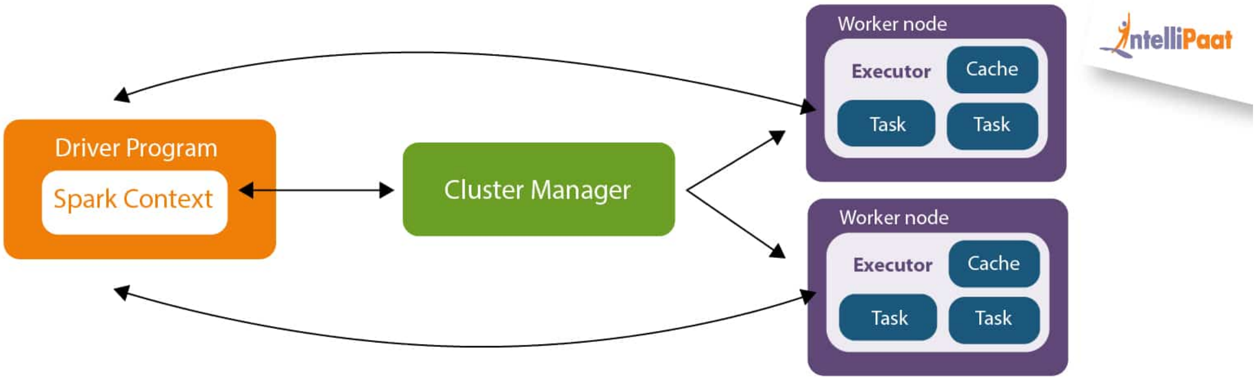
- Spark is suitable for real-time operations and to process larger data on a network.
- Apache spark is an open-source cluster computing framework.

- Apache spark provides you 100 times faster performance for a few applications (100 times in memmory and 10 times in disk).
- Apache spark is suitale for machine learning algorithms as it allows programs to load and query data repreatedly.

Components of a Spark Project :

1. **Spark SQL** : provides capability to read from various structured data like CSV, Avro, Parquet
2. **Spark Streaming** : Enables fault tolerant processing of real-time data streams.
3. **MLLib** : Provides tools like ML algorithms and other utilities.
4. **GraphX** : Perform and manipulate graph parallel computations.
5. **SparkR** : Enables R language users to leverage power of Apache
6. **Spark Core** : Performs the I/O operations

Architecture of Spark



- **Driver Program** : it is the exe cution point of the program. It creates SparkContext object.
- **spark context** :
  - Spark context is crutial component which ensures the connection to the *spark cluster*.
  - It is entry point for spark api
  - It is responsible for managing the spark application life cycle
  - It handles allocatin of resources(CPU,MEMORY,etc) required by executors
  - Manages configuration of spark application , eg: memory.
- **Cluster Manager** :
  - Manages the cluster, by default it is Spark stand alone, but YARN is also there.
  - It schedules and dispatches the task to the worker nodes.
  - It distrubutes the works or tasks to the worker nodes
  - Cluster manager is responsible for the scaling of the resources up or down, depending on the demand
  - Handles the fault tolerance. If the job fails it reshedules it to maintain reliabiiity of the cluster.
  - Monitors ans manges the cluster's resources to ensure that the resources are managed efficiently.
- **Executers** : It uses cache slots to keep data in memory and task slots are java threads that run the code Executors run tasks scheduled by the driver.
- **Worker Node** : They are the slave nodes that execute the tasks. Contains tasks and cache(storage). If any resource needed it asks the Cluster Mangager.

Resiliant Distributed Data(RDD) : - Structure which is a core data structre of Spark. Fundamental data structure in spark. - RDD is an immutable collection of objects which defines the data structure of Spark. - *Lazy Evaluation* - *Partitioning* is by checking the number of workers. This can be controlled by *rdd.repartition()* helps you to control the number of partitions and *rdd.coelesce()* reduces the number of partitions.

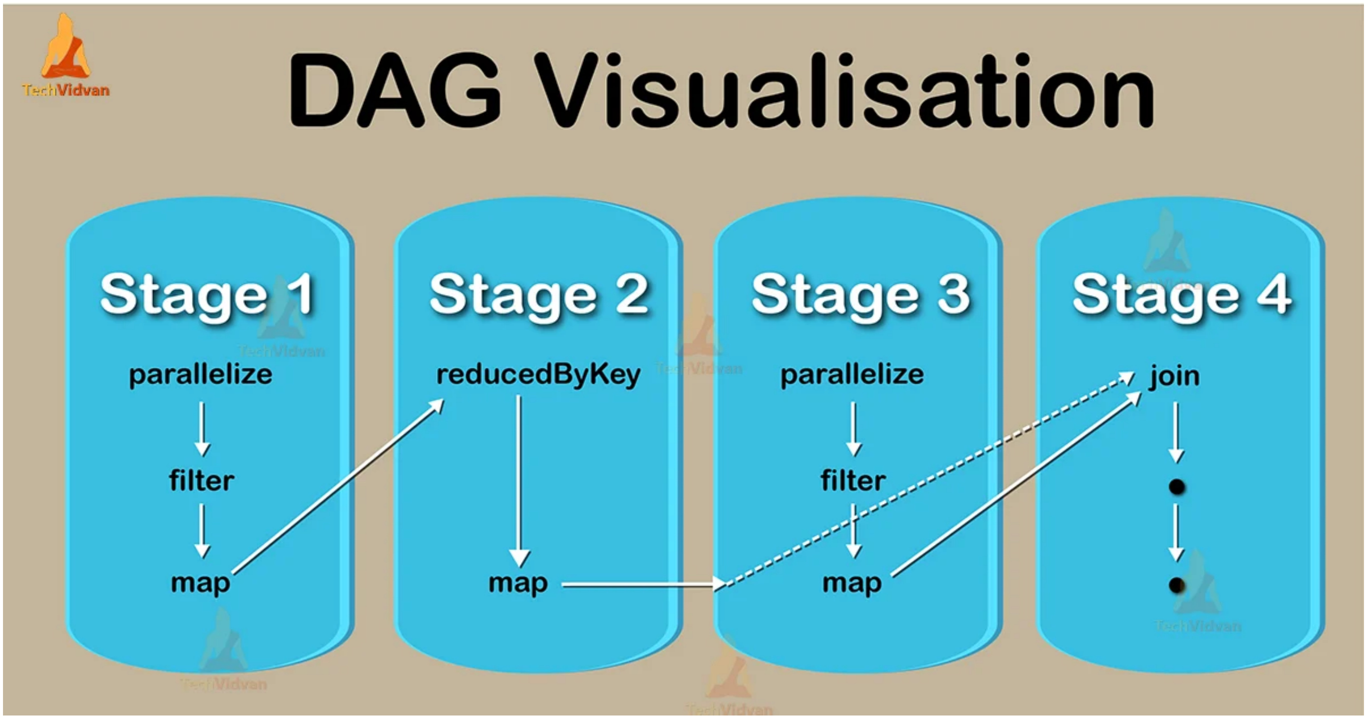
Creating Spark RDD

1. Parallelized Collections : sc.parallelize() -> Any collection
2. Existing RDDs : rdd2 = rdd1.transform()

Whenever you start a spark shell or pyspark it starts 2 session by default, 1, spark context and 2, spark session. Spark shell is scala mode of the spark, whereas the pyspark is the python version.

Directed Acycle Graph (DAG)

One node in a graph is directly connected to another . Each vertex of the graph represents a separate operation, and the edges represnet the operation dpendencies.



Vertex : stage of each application/job, In each stage you find RDD creation and transformation.

stage 1 : parallelize -> filter -> map
stage 2 : RecducedByKey -> map
stage 3 : parallelize -> filter -> map
stage 4 : join rdds

Narrow transformation is self-sufficient. It is the result of a map() an filter() functions such that the data is aggregate. Wide transformation is not self-sufficient. It isresult of GroupByKey() and ReduceByKey() functions