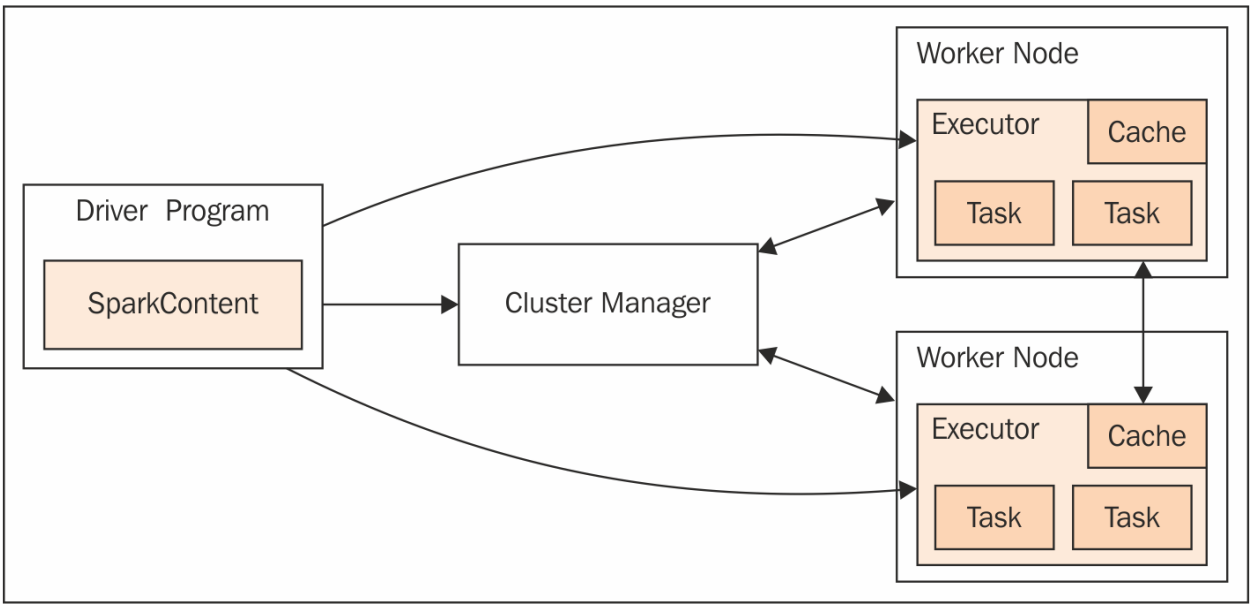
Problems with big data - Store, Process, Scale
Hadoop - to store(HDFS-stored data in distributed manner) and process(MR-Map Reduce for distributed processing) big data
Limitation of hadoop - 1) on disk computation 2) It could process only batch data
3) If we want to do 10 different things, we need to learn 10 different frameworks
Apache Spark - it is a multi-language engine for executing data engineering, data science and machine learning on single-node machines or clusters.

- Can be considered as the successor of Hadoop as it overcomes the drawbacks of it. Unlike hadoop, it supports both real-time as well as batch processing. It is a general clustering system.
- It also supports in-memory calculations, which makes it 100 times faster than hadoop. This is made possible by reducing the number of read/write operations into the disk.
- It offers high-level APIs in Java, Python, Scala, R and SQL.



Four types of cluster manager: Standalone, Apache Mesos, Hadoop YARN, Kubernetes
RDD, DataFrame, Dataset
Spark Context

In [1]:
```
import pyspark
```

In [6]:
```
import findspark
```

In [7]:
```
findspark.init('/usr/local/spark')
```

In [8]:
```
from pyspark import SparkContext
```

In [11]:
```
sc.stop()
```

In [12]:
```
conf=pyspark.SparkConf().setMaster("local").setAppName("first")
```

In [13]:
```
sc=SparkContext(conf=conf)
```

In [14]:
```
rdd=sc.parallelize([1,2,3])
```

In [15]:
```
rdd.collect()
```

Out[15]:
```
[1, 2, 3]
```

In [1]:
```
import findspark
```

In [2]:
```
findspark.init()
```

In [3]:
```
from pyspark.sql import SparkSession
```

In [5]:
```
spark=SparkSession.builder.appName("RDDExample").getOrCreate()
```

```
Setting default log level to "WARN".
To adjust logging level use sc.setLogLevel(newLevel). For SparkR, use setLogLevel(newLevel).
23/09/21 08:38:50 WARN NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-j
ava classes where applicable
```

In [7]:
```
df=spark.createDataFrame([(1,2,3)])
```

In [8]:
```
df.show()
```

```
+---+---+---+
| _1| _2| _3|
+---+---+---+
|  1|  2|  3|
+---+---+---+
```

In [9]:
```
from datetime import datetime, date
import pandas as pd
from pyspark.sql import Row

df = spark.createDataFrame([
    Row(a=1, b=2., c='string1', d=date(2000, 1, 1), e=datetime(2000, 1, 1, 12, 0)),
    Row(a=2, b=3., c='string2', d=date(2000, 2, 1), e=datetime(2000, 1, 2, 12, 0)),
    Row(a=4, b=5., c='string3', d=date(2000, 3, 1), e=datetime(2000, 1, 3, 12, 0))
])
df
```

Out[9]:
```
DataFrame[a: bigint, b: double, c: string, d: date, e: timestamp]
```

In [10]:
```
df.printSchema()
```

```
root
 |-- a: long (nullable = true)
 |-- b: double (nullable = true)
 |-- c: string (nullable = true)
 |-- d: date (nullable = true)
 |-- e: timestamp (nullable = true)
```

In [11]:
```
df.show()
```

```
+---+---+-------+----------+-------------------+
|  a|  b|      c|         d|                  e|
```

In [14]:
```
df.select("a","b").show()
```

```
+---+---+
|  a|  b|
+---+---+
|  1|2.0|
|  2|3.0|
|  4|5.0|
+---+---+
```

In [13]:
```
from pyspark.sql.functions import col
```

In [15]:
```
df.select(col("a").alias("first name")).show()
```

```
+----------+
|first name|
+----------+
|         1|
|         2|
|         4|
+----------+
```

In [16]:
```
df.select("a",col("b"),df["c"]).show()
```

```
+---+---+-------+
|  a|  b|      c|
+---+---+-------+
|  1|2.0|string1|
|  2|3.0|string2|
|  4|5.0|string3|
+---+---+-------+
```

In [18]:
```
df.withColumnRenamed("a","First Name").show()
```

```
+----------+---+-------+----------+-------------------+
|First Name|  b|      c|         d|                  e|
+----------+---+-------+----------+-------------------+
|         1|2.0|string1|2000-01-01|2000-01-01 12:00:00|
|         2|3.0|string2|2000-02-01|2000-01-02 12:00:00|
|         4|5.0|string3|2000-03-01|2000-01-03 12:00:00|
+----------+---+-------+----------+-------------------+
```

In [20]:
```
from pyspark.sql.functions import *
```

In [24]:
```
df.select(concat("b",lit(" & "),"c")).show()
```

```
+-----------------+
|concat(b,  & , c)|
+-----------------+
|    2.0 & string1|
|    3.0 & string2|
|    5.0 & string3|
+-----------------+
```

In [26]:
```
df\
.select(concat("b",lit(" & "),"c"))\
.show()
```

```
+-----------------+
|concat(b,  & , c)|
+-----------------+
|    2.0 & string1|
|    3.0 & string2|
|    5.0 & string3|
+-----------------+
```

In [27]:
```
dfffinal=df.drop("d")
```

In [28]:
```
dfffinal.write.csv("/home/labuser/1PySpark/Day1/finalemp")
```