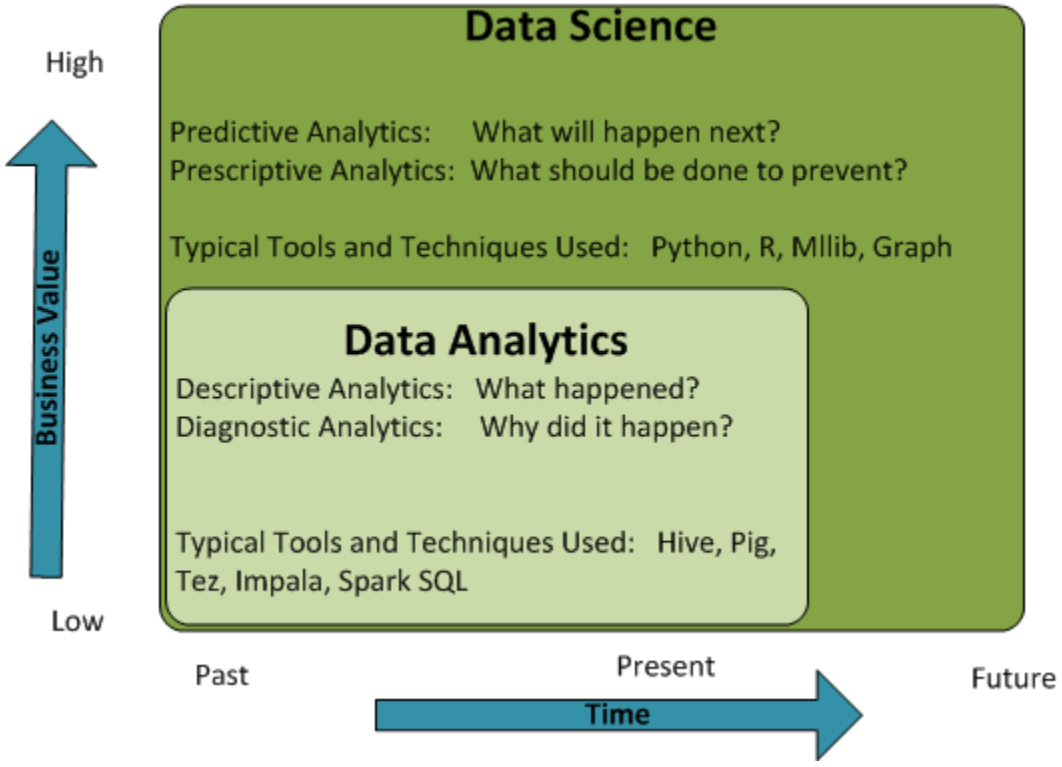
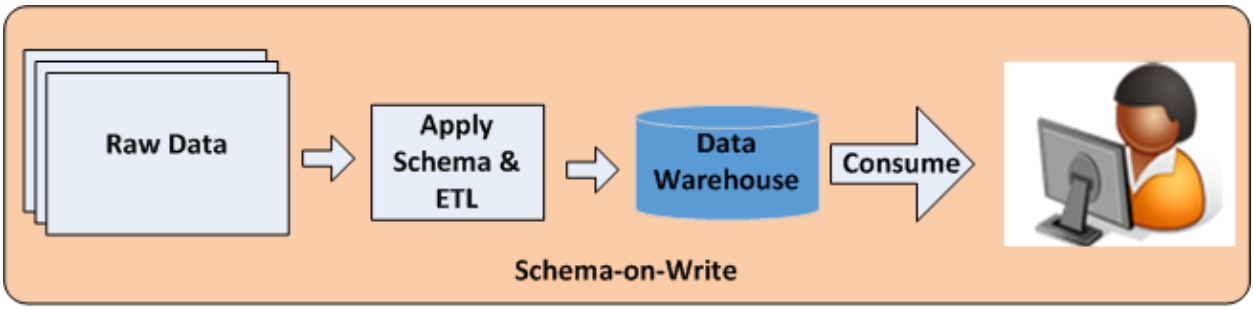
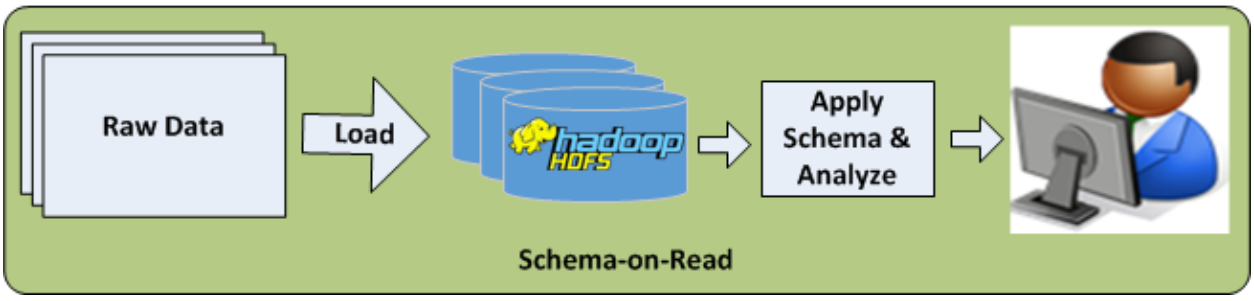
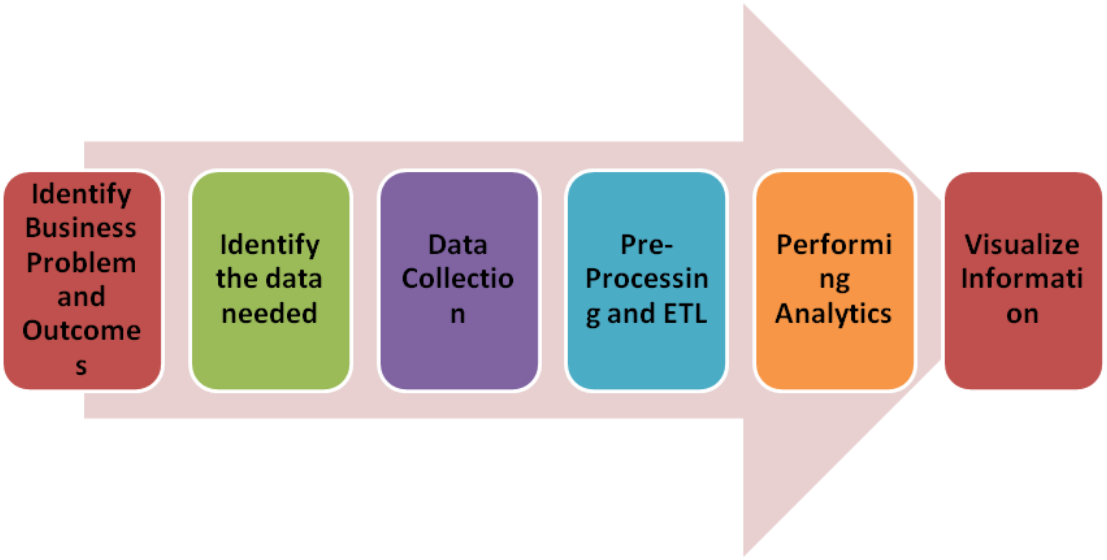
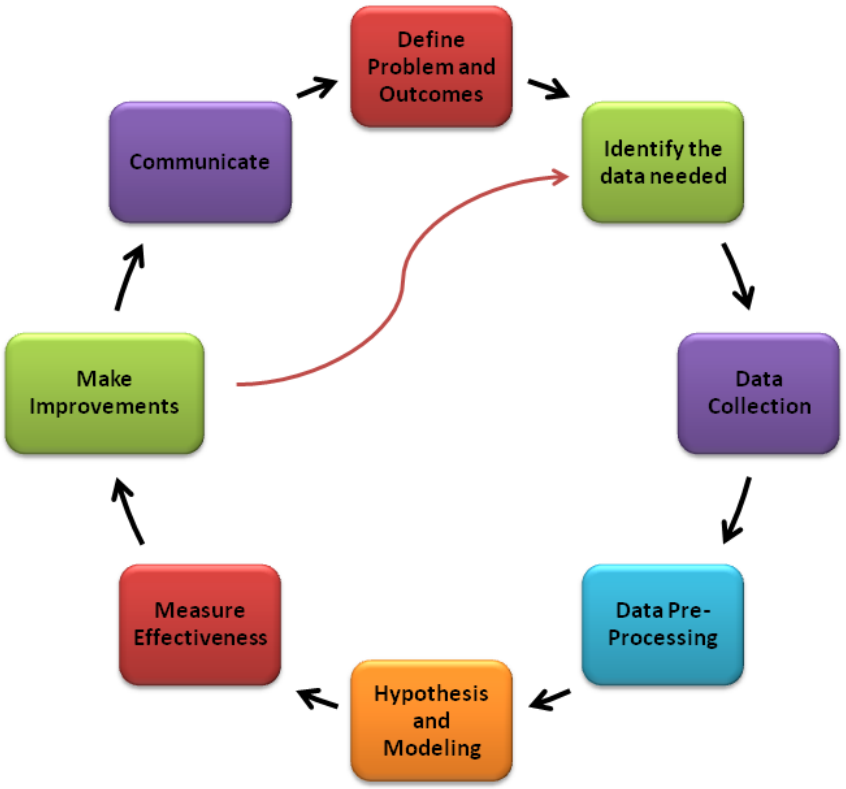
Chapter 1: Big Data Analytics at a 10,000-Foot View



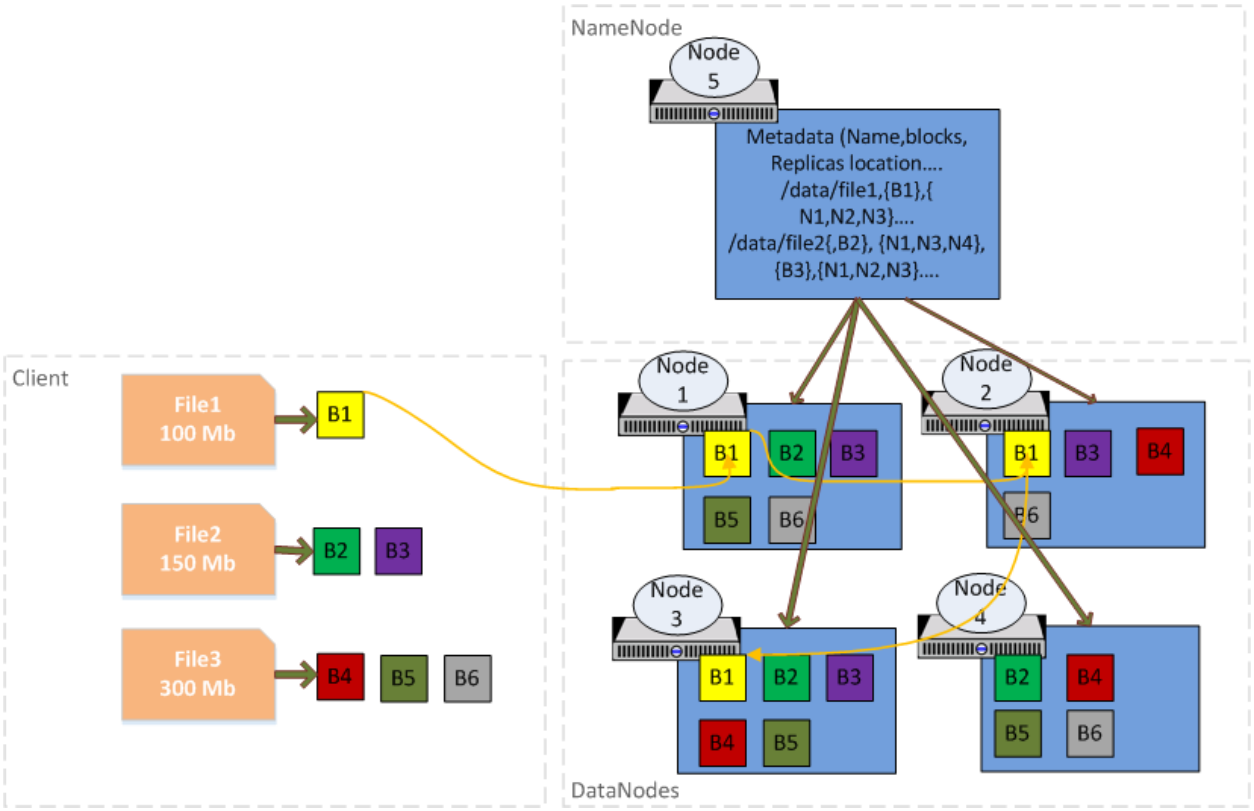


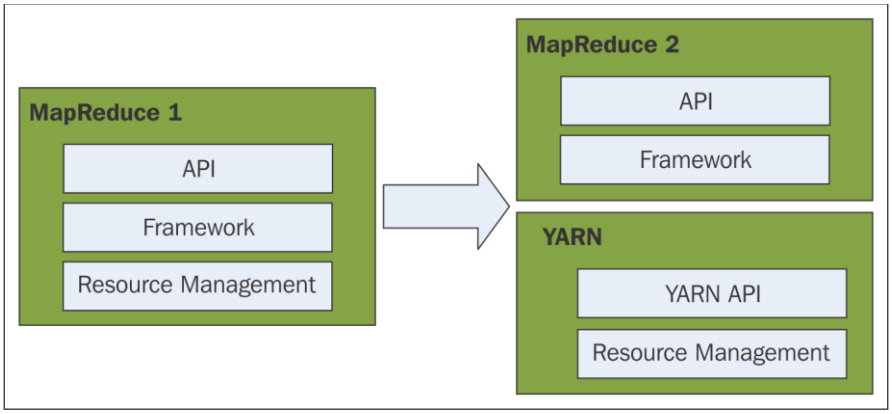


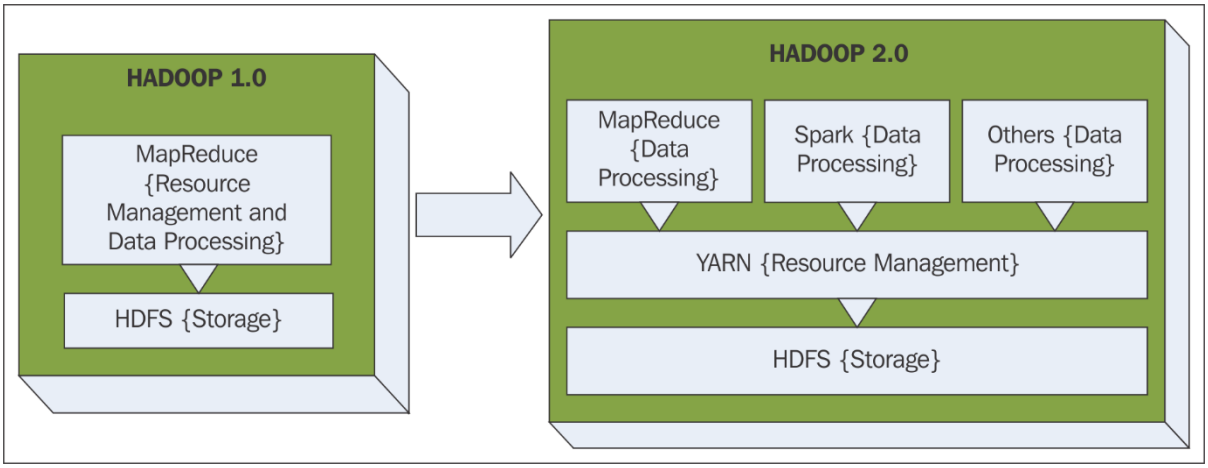

Identify Business Problem and Outcomes
Identify the data needed
Data Collection
Pre-Processing and ETL
Performing Analytics
Visualize Information

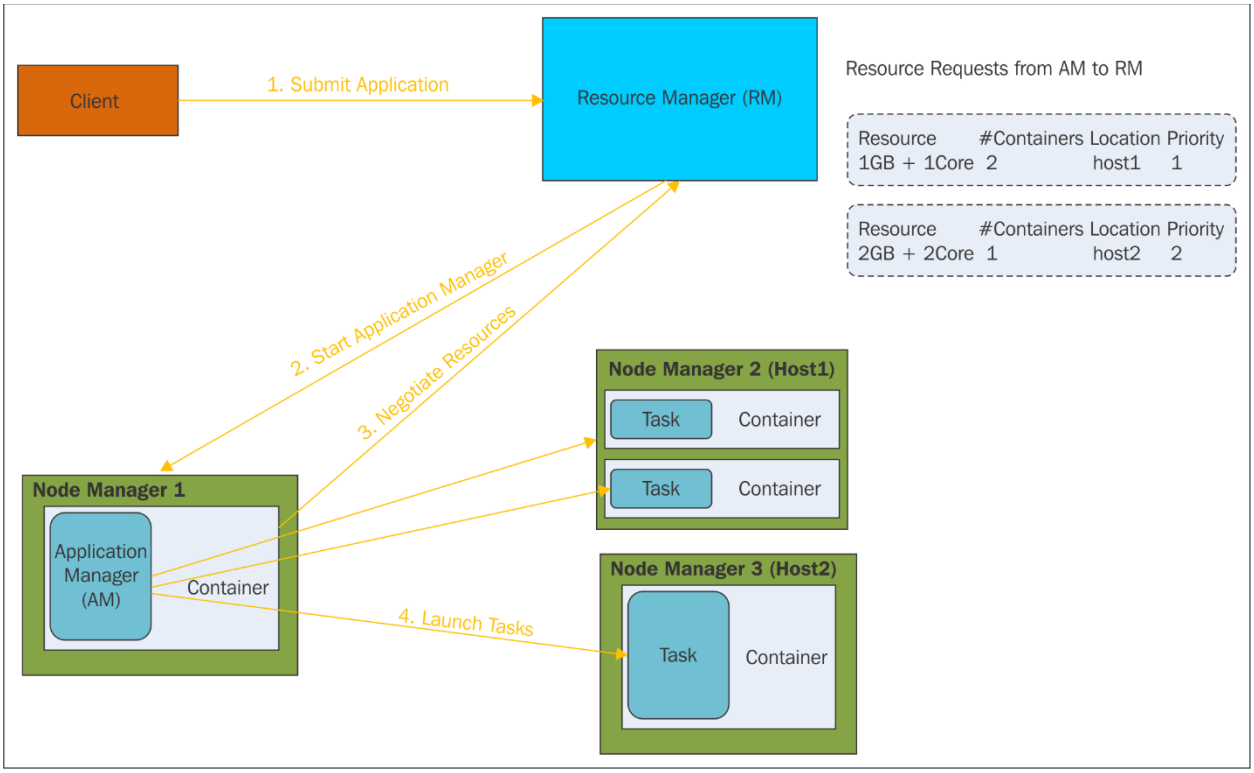

Define Problem and Outcomes
Identify the data needed
Data Collection
Data Pre-Processing
Hypothesis and Modeling
Measure Effectiveness
Make Improvements
Communicate

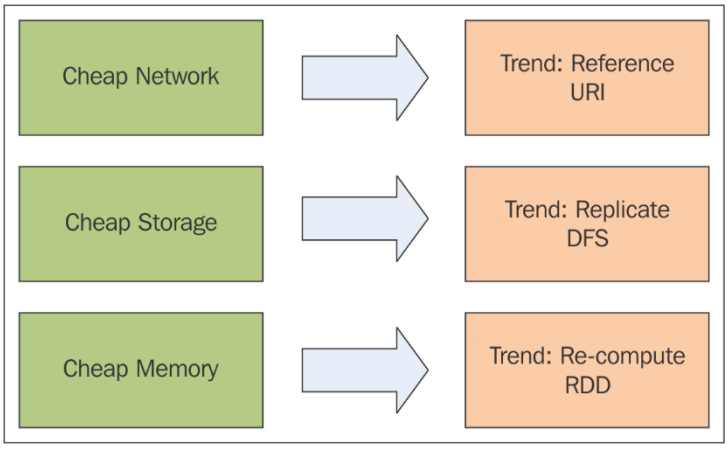
Chapter 2: Getting Started with Apache Hadoop and Apache Spark

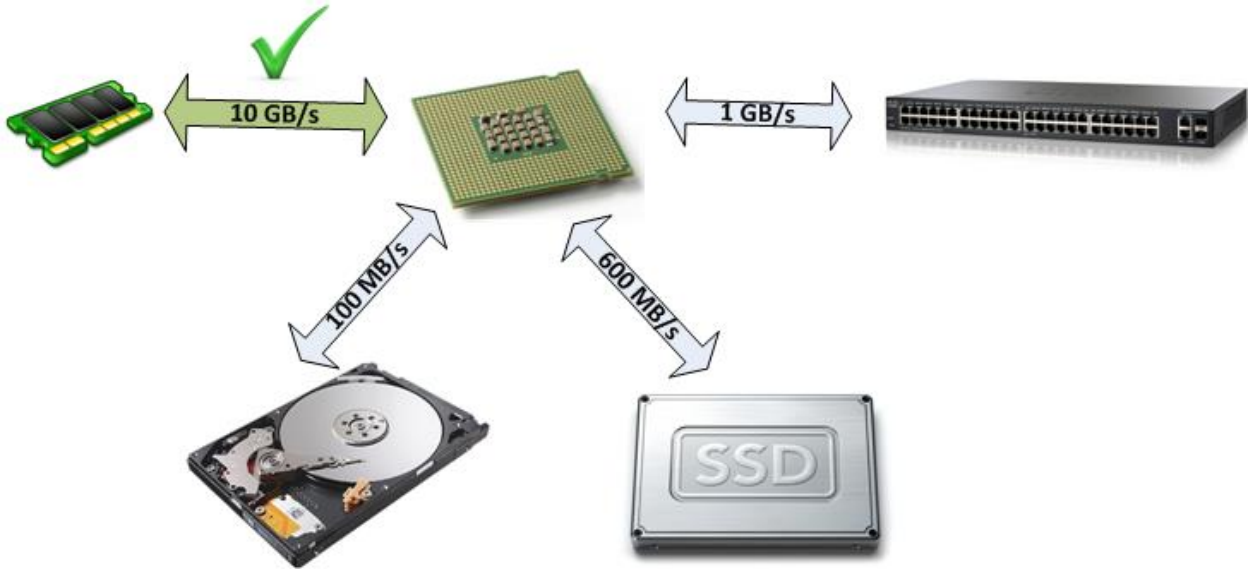





HADOOP 1.0
MapReduce {Resource Management and Data Processing}
HDFS {Storage}
HADOOP 2.0
MapReduce {Data Processing}
Spark {Data Processing}
Others {Data Processing}
YARN {Resource Management}
HDFS {Storage}


Client
1. Submit Application
Resource Manager (RM)
Resource Requests from AM to RM
Resource 1GB + 1Core
#Containers 2
Location host1
Priority 1
Resource 2GB + 2Core
#Containers 1
Location host2
Priority 2
2. Start Application Manager
3. Negotiate Resources
Node Manager 1
Application Manager (AM)
Container
Node Manager 2 (Host1)
Task
Container
Task
Container
Node Manager 3 (Host2)
Task
Container
4. Launch Tasks



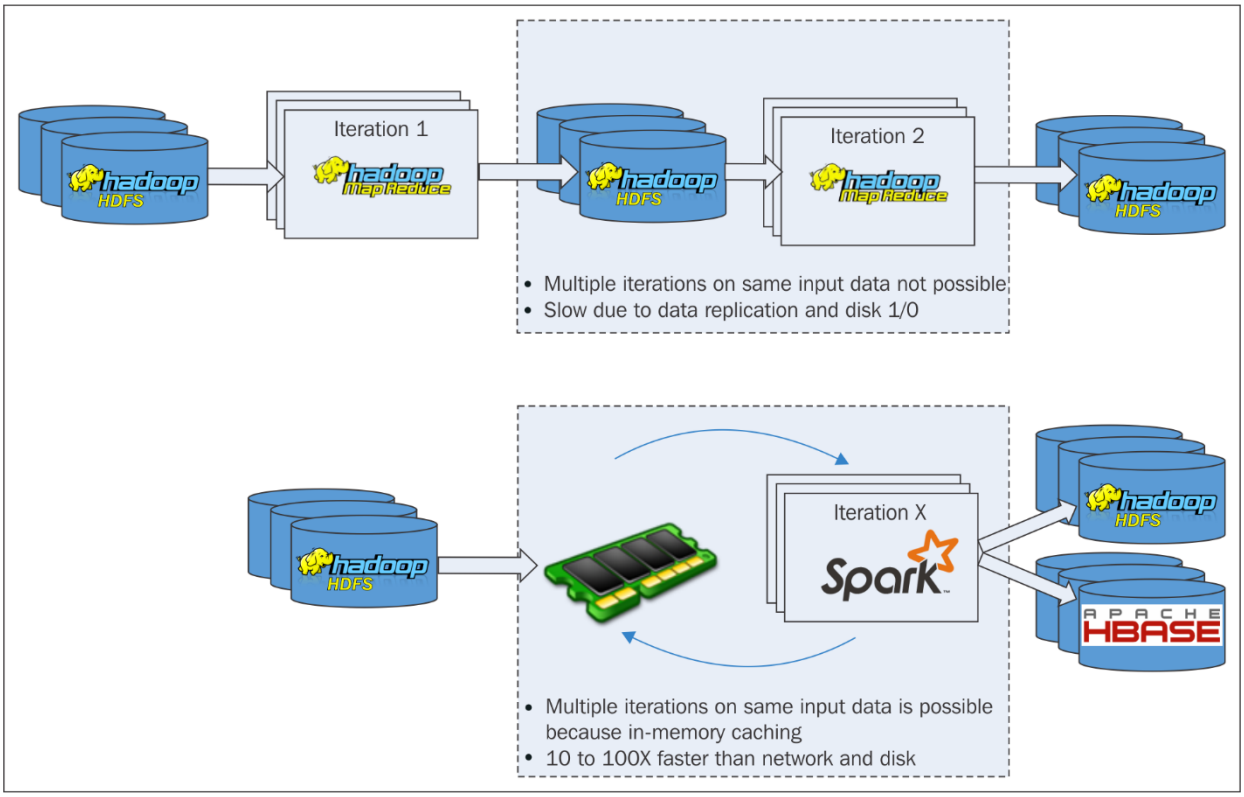


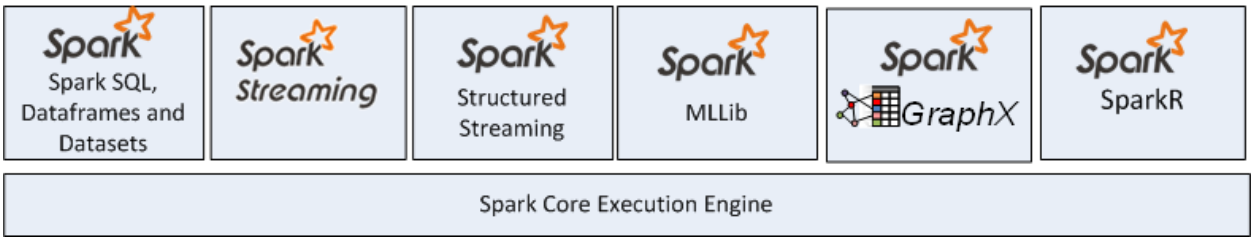
2010
- Spark Paper released
- Open Sourced

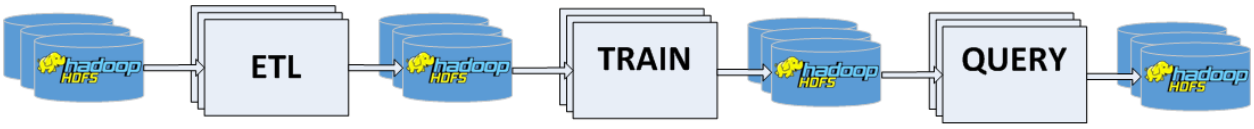
2013
- Spark goes to Apache Software Foundation

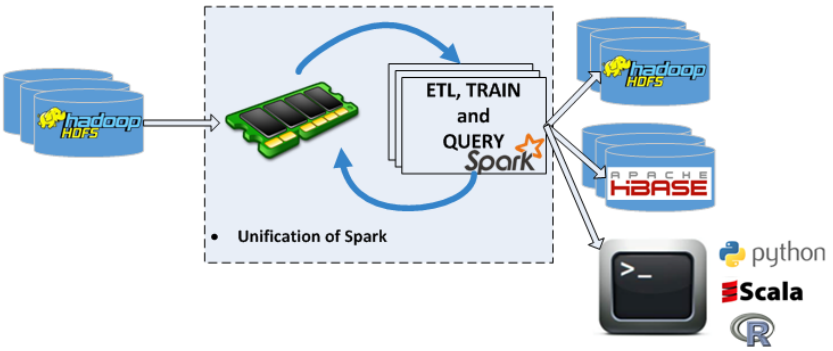
2014
- Top level project at Apache Software Foundation.
- Databricks setup 100 TB Sort new record in 23 Mins with Spark.
- Spark v 1.0 released.

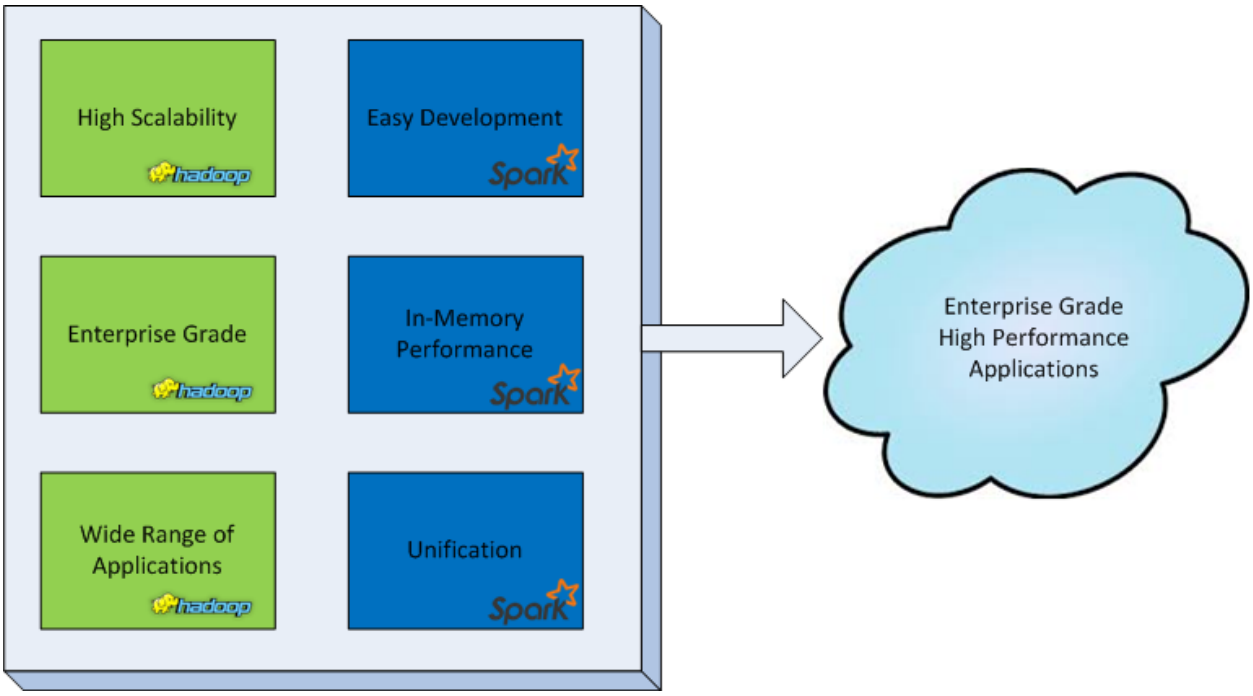
2015 & 2016
- Spark v 1.5 released
- Spark v 1.6 released
- Spark v 2.0 released
- New APIs, Features and Performance Improvements

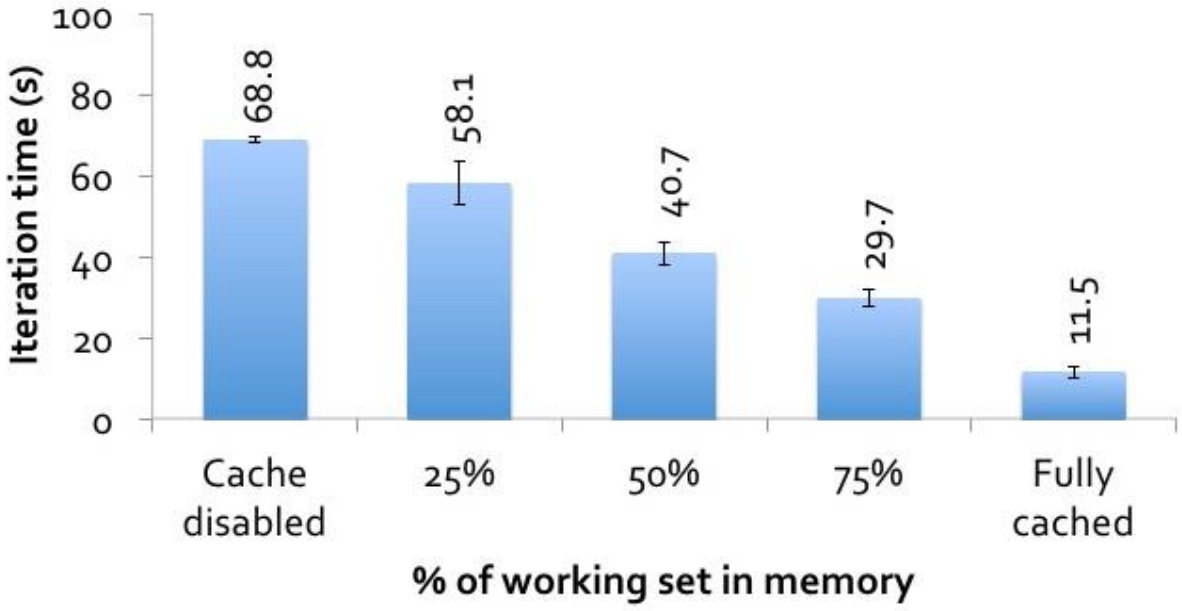

Iteration 1
Iteration 2
Iteration X
Spark
HBASE
• Multiple iterations on same input data not possible
• Slow due to data replication and disk 1/0
• Multiple iterations on same input data is possible because in-memory caching
• 10 to 100X faster than network and disk

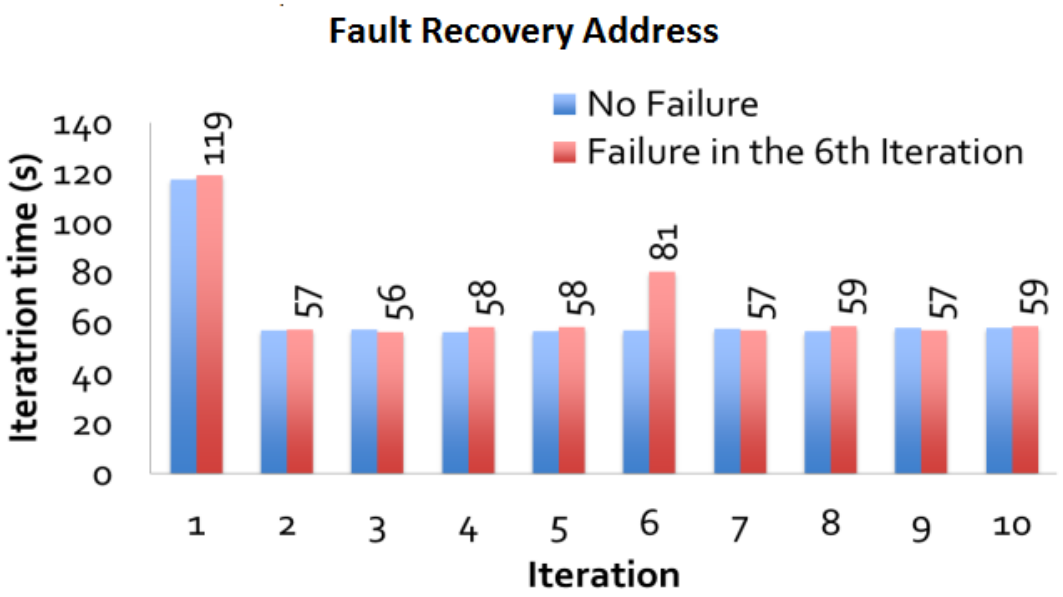

Spark SQL, Dataframes and Datasets
Spark Streaming
Structured Streaming
MLLib
GraphX
SparkR
Spark Core Execution Engine


ETL
TRAIN
QUERY


ETL, TRAIN and QUERY
Spark
HBASE
• Unification of Spark
python
Scala


High Scalability
hadoop
Easy Development
Spark
Enterprise Grade
hadoop
In-Memory Performance
Spark
Wide Range of Applications
hadoop
Unification
Spark
Enterprise Grade High Performance Applications

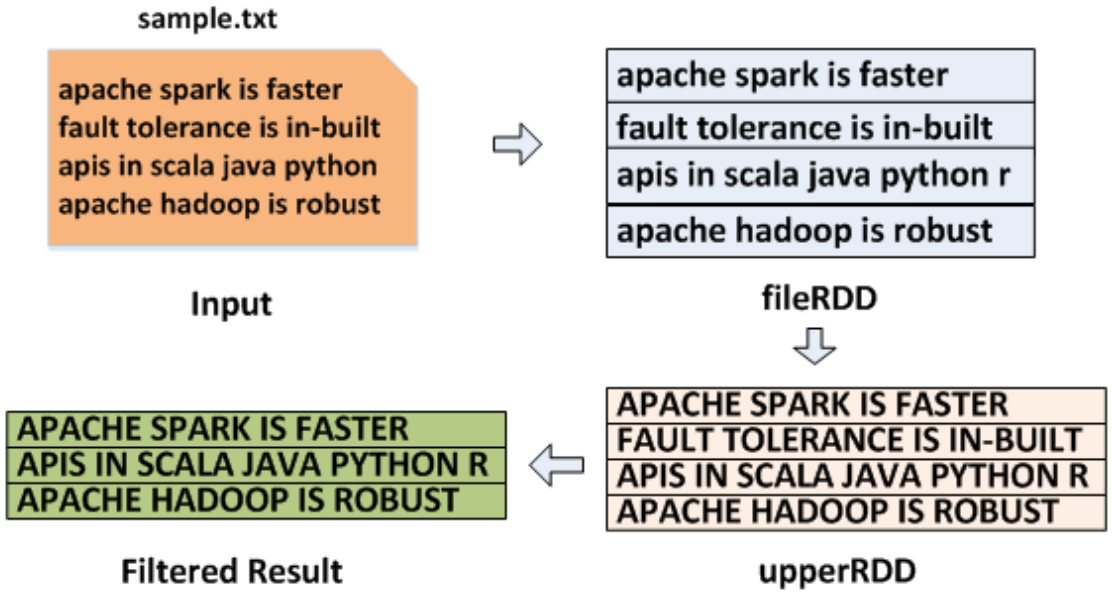

Iteration time (s)
100
80
60
40
20
0
68.8
58.1
40.7
29.7
11.5
Cache disabled
25%
50%
75%
Fully cached
% of working set in memory

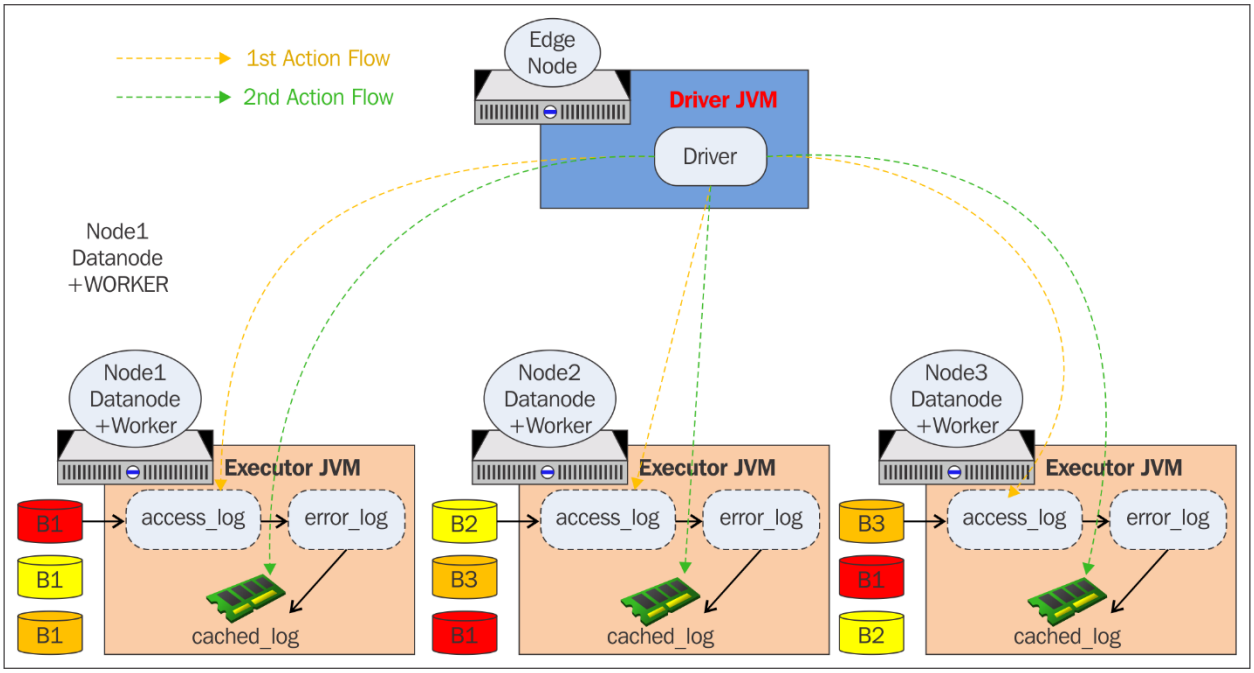

Fault Recovery Address
No Failure
Failure in the 6th Iteration
Iteratrion time (s)
140
120
100
80
60
40
20
0
119
57
56
58
58
81
57
59
57
59
1
2
3
4
5
6
7
8
9
10
Iteration

Chapter 3: Deep Dive into Apache Spark

```
[cloudera@quickstart spark-2.0.0-bin-hadoop2.7]$ bin/spark-shell
Spark context Web UI available at http://192.168.139.175:4040
Spark context available as 'sc' (master = local[*], app id = local-147052
7900398).
Spark session available as 'spark'
Welcome to
      ____              __
     / __/__  ___ _____/ /__
    _\ \/ _ \/ _ `/ __/  '_/
   /___/ .__/\_,_/_/ /_/\_\   version 2.0.0
      /_/

Using Scala version 2.11.8 (Java HotSpot(TM) 64-Bit Server VM, Java 1.7.0
_67)
Type in expressions to have them evaluated.
Type :help for more information.

scala>
```

```
fileRDD = sc.textFile("/data/sample.txt")
upperRDD = fileRDD.map(lambda line: line.upper())
upperRDD.filter(lambda line: line.startswith('A')).collect()
```





Spark 1.6.0 Jobs Stages Storage Environment Executors **PySparkShell** application UI

Spark Jobs (?)

Total Uptime: 1.2 min
Scheduling Mode: FIFO
Completed Jobs: 1

▸ Event Timeline

Completed Jobs (1)

Job Id	Description	Submitted	Duration	Stages: Succeeded/Total	Tasks (for all stages): Succeeded/Total
0	count at <stdin>:1	2016/04/07 04:01:06	1 s	1/1	4/4

Summary Metrics for 4 Completed Tasks

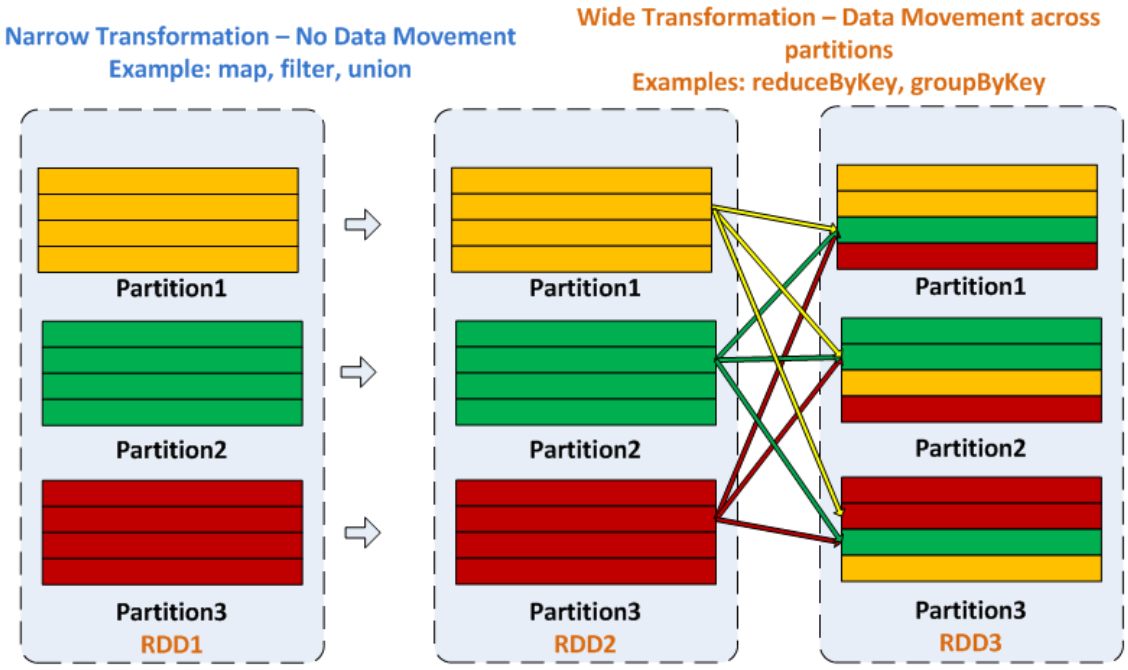
Metric	Min	25th percentile	Median	75th percentile	Max
Duration	0.4 s	0.4 s	0.4 s	0.4 s	0.4 s
GC Time	0.3 s	0.3 s	0.3 s	0.3 s	0.3 s

Aggregated Metrics by Executor

Executor ID ▲	Address	Task Time	Total Tasks	Failed Tasks	Succeeded Tasks
0	192.168.139.175:43785	5 s	4	0	4

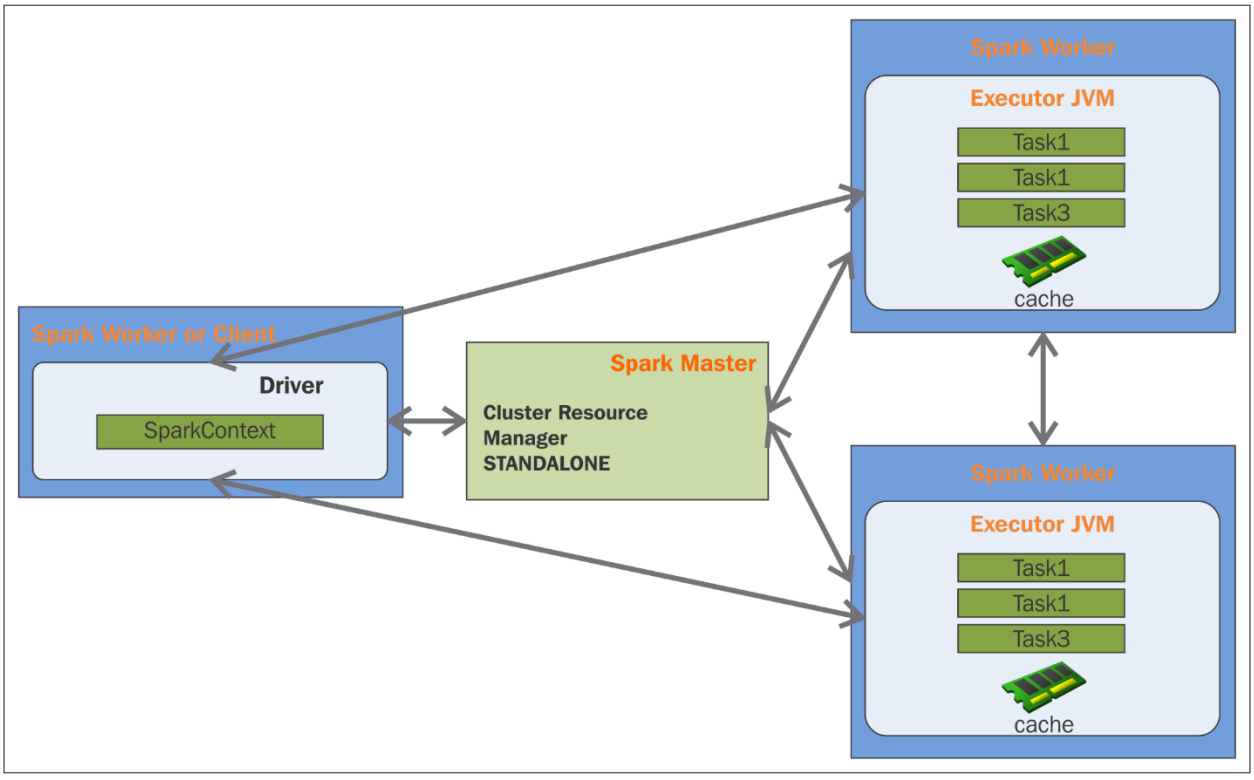
Tasks

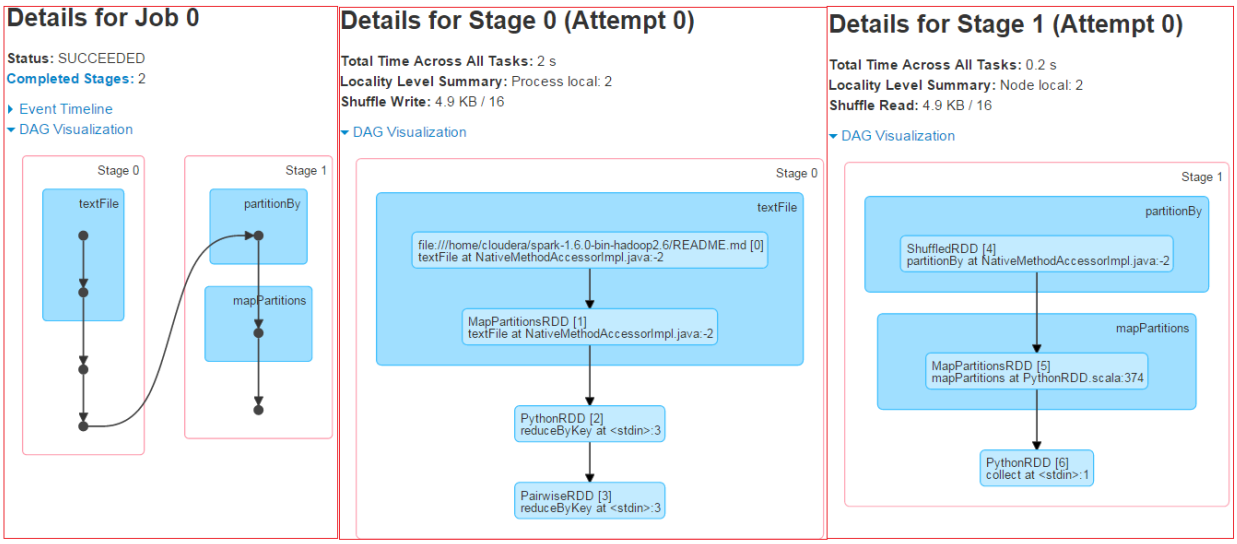
Index ▲	ID	Attempt	Status	Locality Level	Executor ID / Host	Launch Time	Duration	GC Time	Errors
0	0	0	SUCCESS	PROCESS_LOCAL	0 / 192.168.139.175	2016/08/06 18:06:19	0.4 s	0.3 s	
1	1	0	SUCCESS	PROCESS_LOCAL	0 / 192.168.139.175	2016/08/06 18:06:19	0.4 s	0.3 s	
2	2	0	SUCCESS	PROCESS_LOCAL	0 / 192.168.139.175	2016/08/06 18:06:19	0.4 s	0.3 s	
3	3	0	SUCCESS	PROCESS_LOCAL	0 / 192.168.139.175	2016/08/06 18:06:19	0.4 s	0.3 s	



Tasks

Index ▲	ID	Attempt	Status	Locality Level	Executor ID / Host
0	0	0	SUCCESS	PROCESS_LOCAL	0 / 192.168.139.164
1	1	0	SUCCESS	PROCESS_LOCAL	0 / 192.168.139.164
2	2	0	SUCCESS	PROCESS_LOCAL	0 / 192.168.139.164
3	3	0	SUCCESS	PROCESS_LOCAL	0 / 192.168.139.164

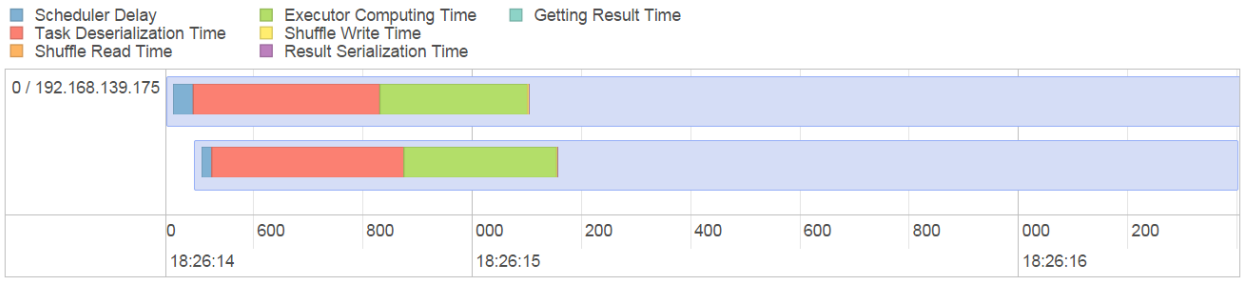


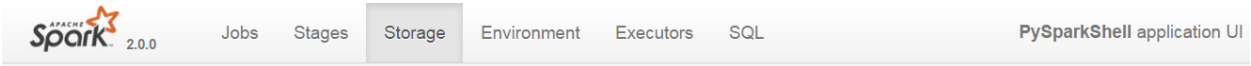


Details for Stage 0 (Attempt 0)

Total Time Across All Tasks: 2 s
Locality Level Summary: Process local: 2
Input Size / Records: 1914.0 B / 99
Shuffle Write: 5.0 KB / 16

DAG Visualization
Show Additional Metrics
Event Timeline
Enable zooming

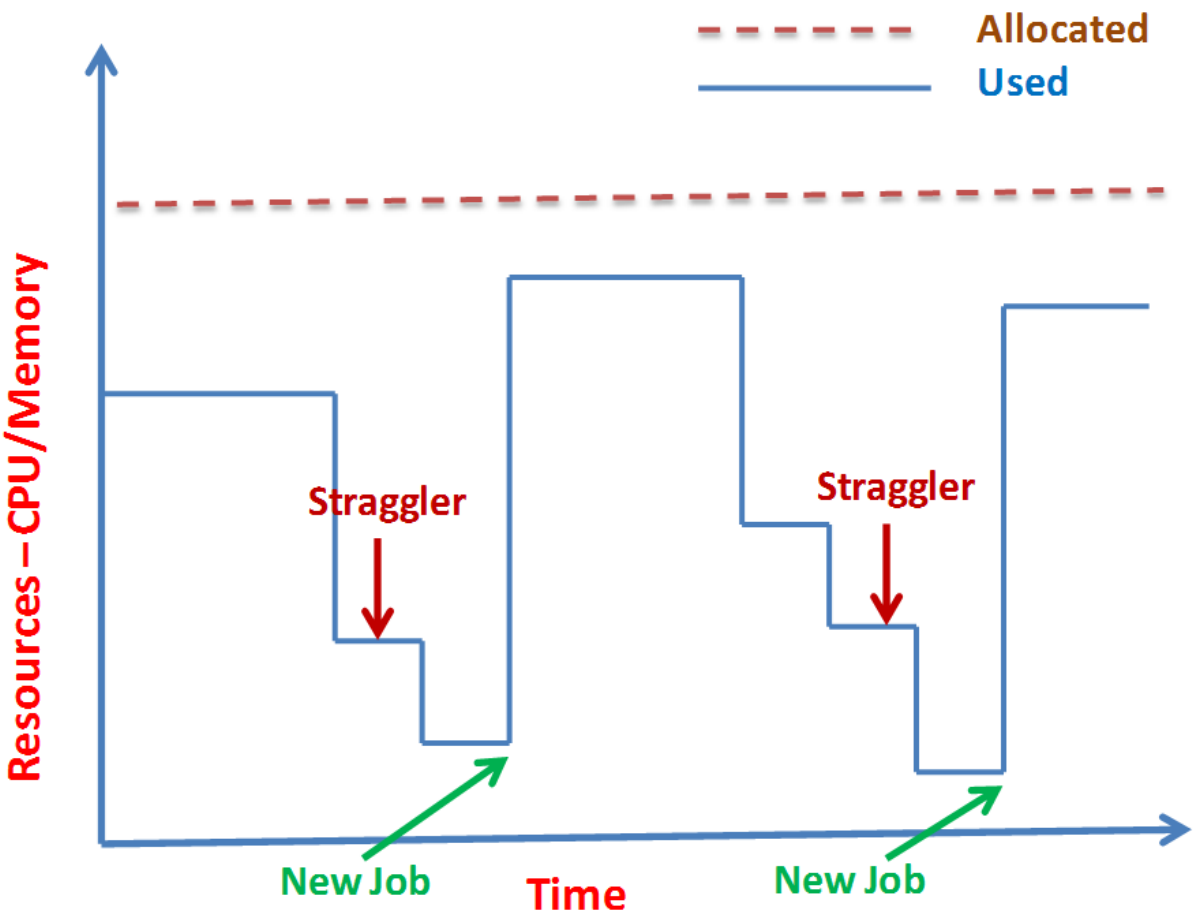


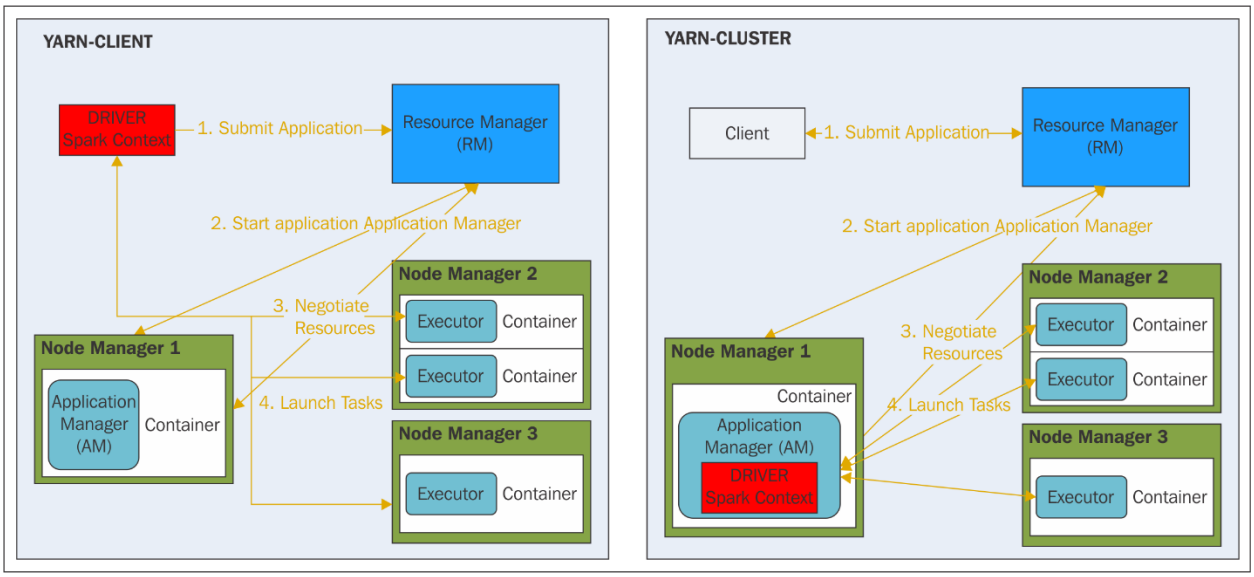


Storage

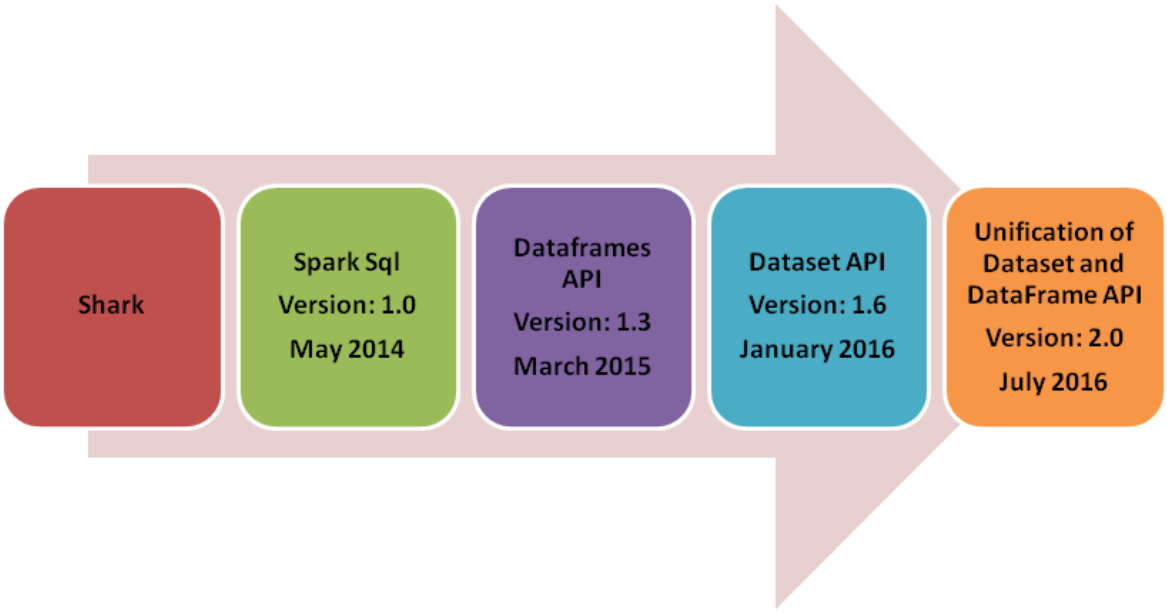
RDDs

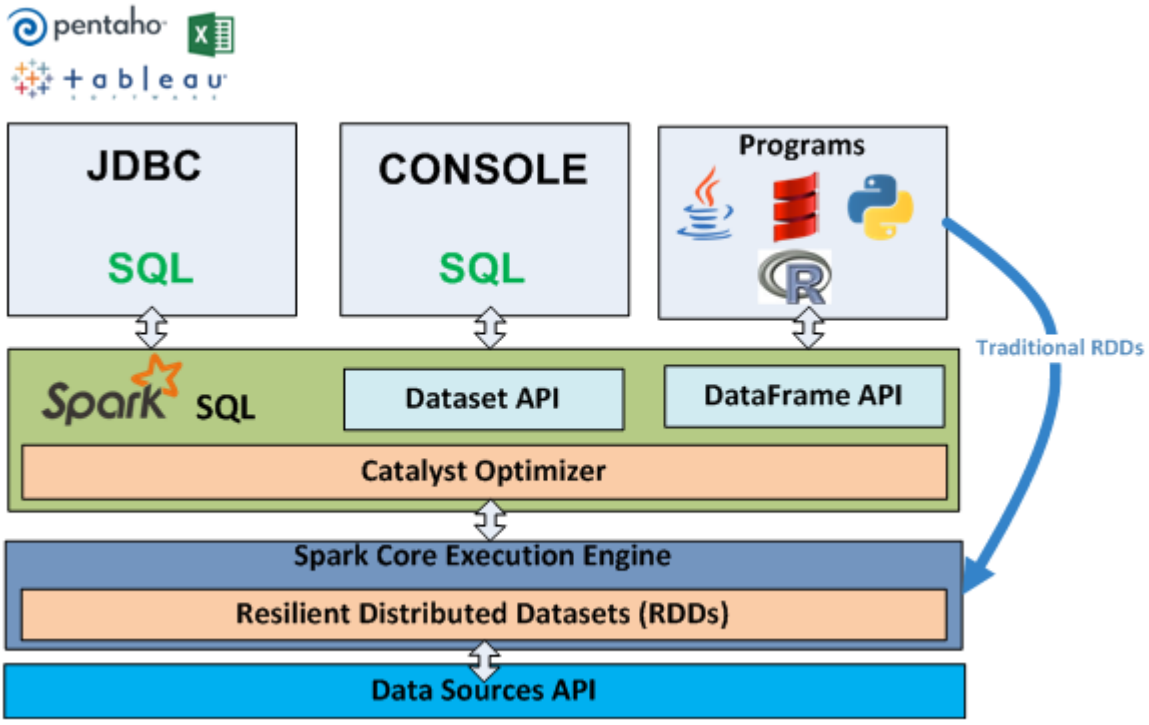
RDD Name	Storage Level	Cached Partitions	Fraction Cached	Size in Memory	Size on Disk
PythonRDD	Memory Serialized 1x Replicated	2	100%	3.9 KB	0.0 B

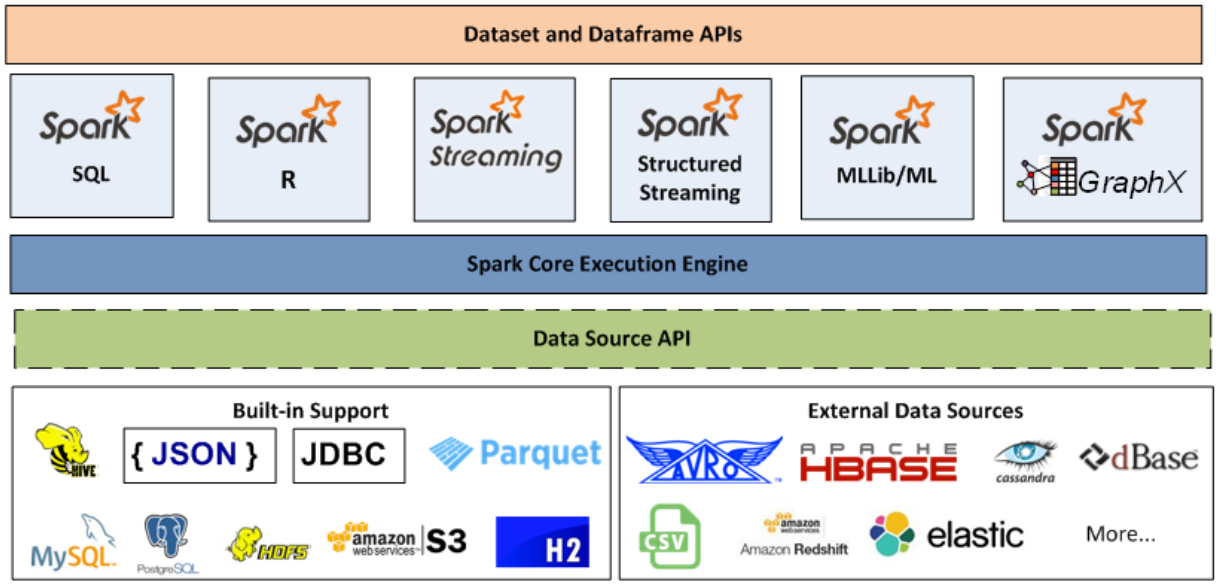

Allocated
Used
Resources – CPU/Memory
Straggler
Straggler
New Job
Time
New Job

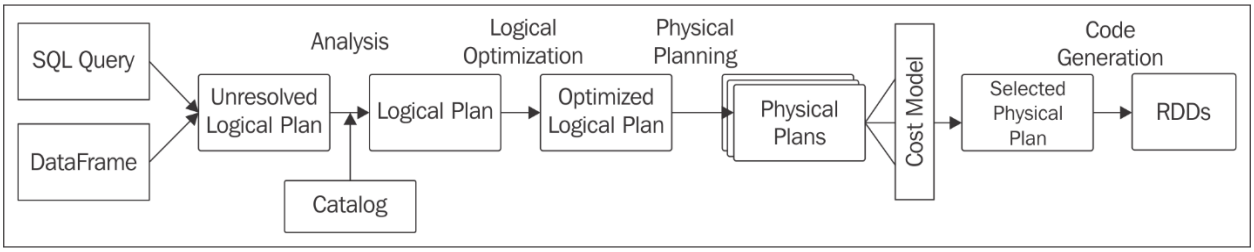

YARN-CLIENT
DRIVER
Spark Context
1. Submit Application
Resource Manager
(RM)
2. Start application Application Manager
Node Manager 2
3. Negotiate
Resources
Executor
Container
Executor
Container
Node Manager 1
Application
Manager
(AM)
Container
4. Launch Tasks
Node Manager 3
Executor
Container
YARN-CLUSTER
Client
1. Submit Application
Resource Manager
(RM)
2. Start application Application Manager
Node Manager 2
3. Negotiate
Resources
Executor
Container
Executor
Container
Node Manager 1
Container
Application
Manager (AM)
DRIVER
Spark Context
4. Launch Tasks
Node Manager 3
Executor
Container

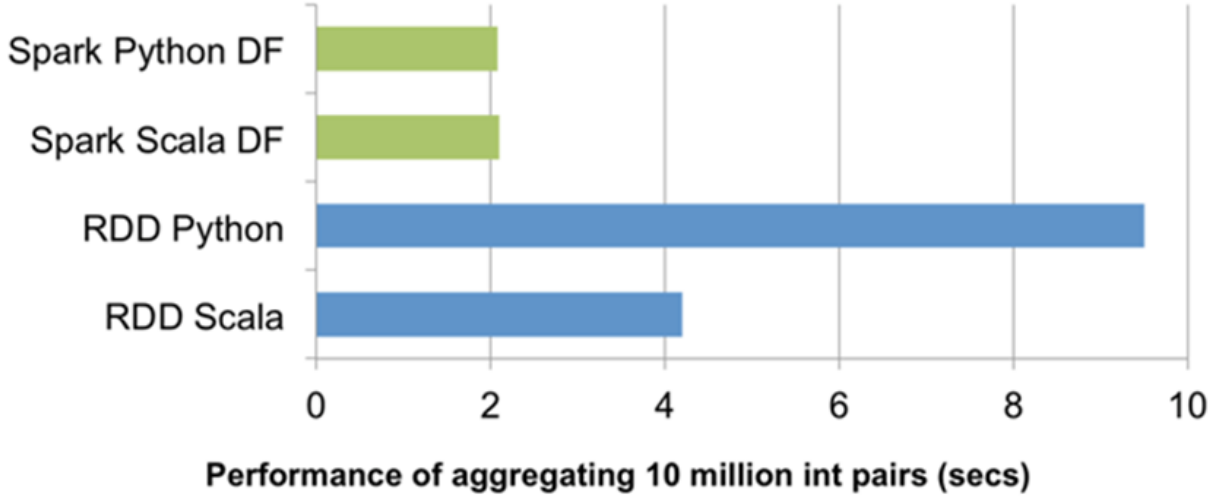
Chapter 4: Big Data Analytics with Spark SQL, DataFrames, and Datasets

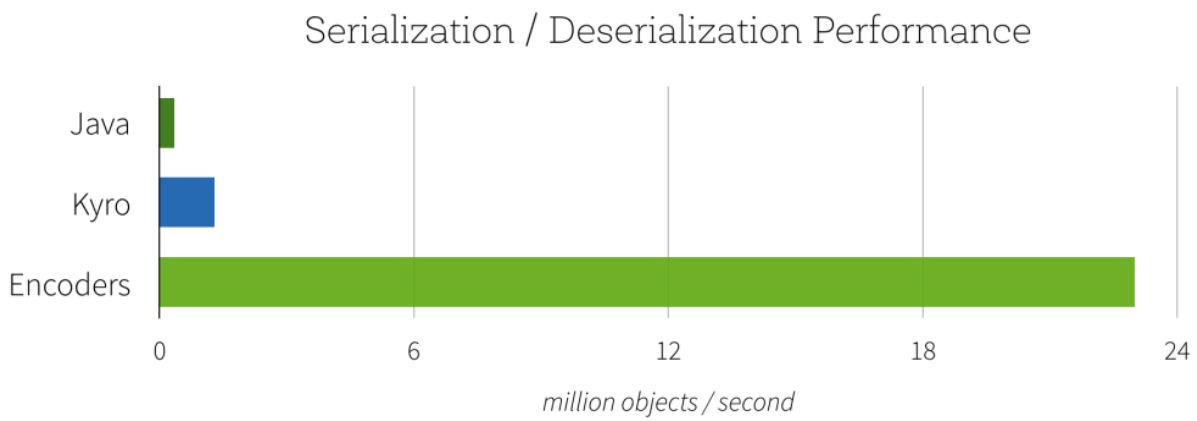





Dataset and Dataframe APIs
Spark SQL
Spark R
Spark Streaming
Spark Structured Streaming
Spark MLLib/ML
Spark GraphX
Spark Core Execution Engine
Data Source API
Built-in Support
{ JSON }
JDBC
Parquet
MySQL
PostgreSQL
HDFS
amazon web services S3
H2
External Data Sources
AVRO
APACHE HBASE
cassandra
dBase
CSV
Amazon Redshift
elastic
More...


SQL Query
DataFrame
Analysis
Unresolved Logical Plan
Catalog
Logical Optimization
Logical Plan
Physical Planning
Optimized Logical Plan
Physical Plans
Cost Model
Code Generation
Selected Physical Plan
RDDs

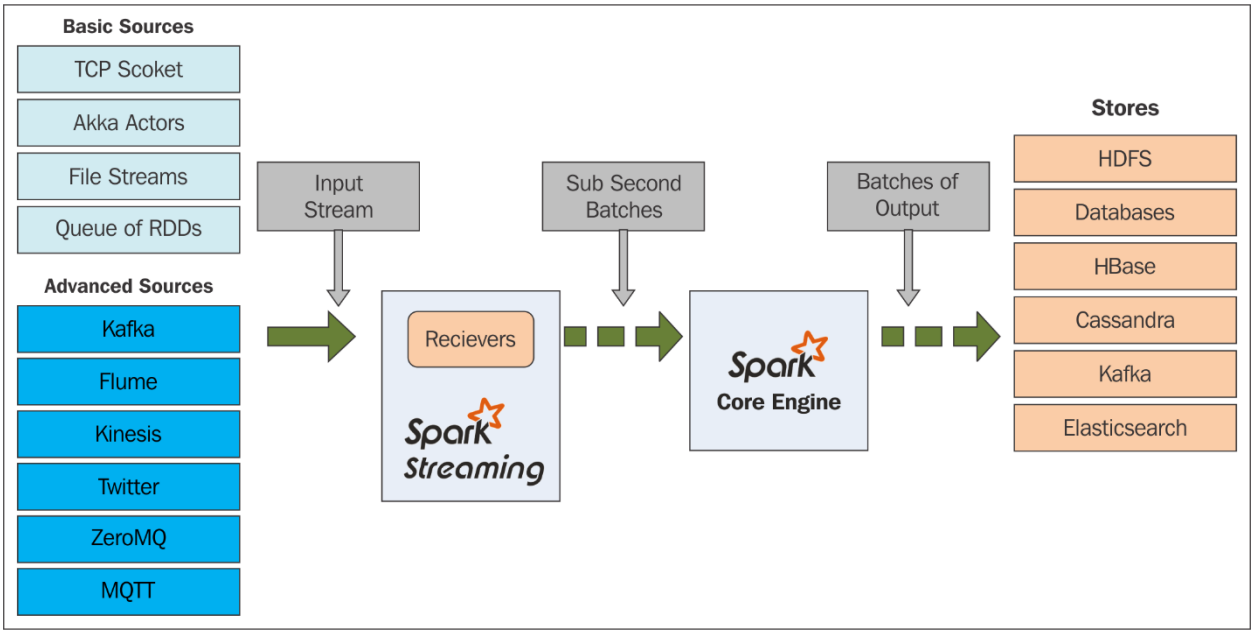

Spark Python DF
Spark Scala DF
RDD Python
RDD Scala
0
2
4
6
8
10
Performance of aggregating 10 million int pairs (secs)

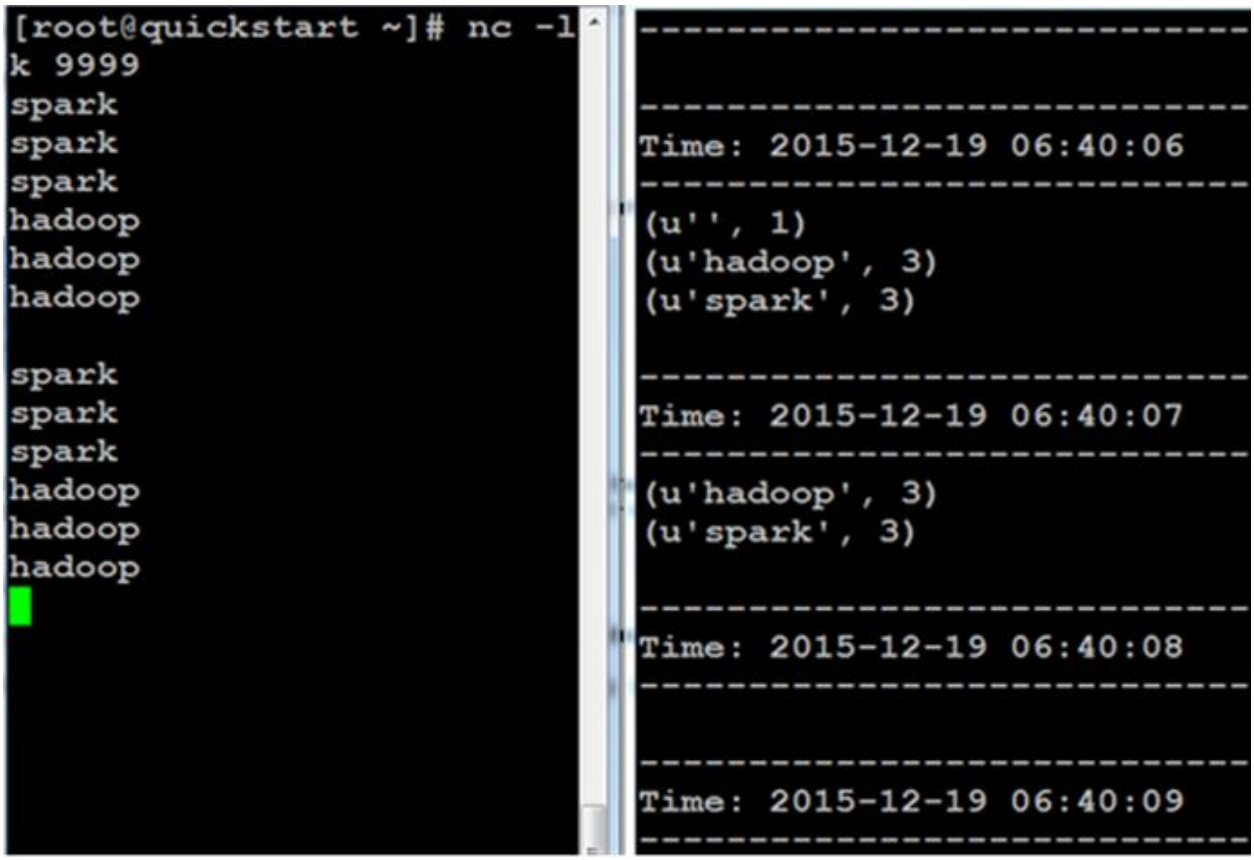



```
>>> mylist = [(50, "DataFrame"),(60, "pandas")]
>>> myschema = ['col1', 'col2']
>>> df = sc.parallelize(mylist).toDF(myschema)
>>> df.printSchema()
root
 |-- col1: long (nullable = true)
 |-- col2: string (nullable = true)

>>> df.show()
+----+---------+
|col1|     col2|
+----+---------+
|  50|DataFrame|
|  60|   pandas|
+----+---------+
```

```
>>> df = spark.read.format('jdbc').options(url='jdbc:mysql://loc
alhost:3306/retail_db?user=root&password=cloudera', dbtable='dep
artments').load()
>>> df.show()
+-------------+---------------+
|department_id|department_name|
+-------------+---------------+
|            2|        Fitness|
|            3|       Footwear|
|            4|        Apparel|
|            5|           Golf|
|            6|       Outdoors|
|            7|       Fan Shop|
+-------------+---------------+

>>> df2rdd = df.rdd
>>> df2rdd.take(2)
[Row(department_id=2, department_name=u'Fitness'), Row(departmen
t_id=3, department_name=u'Footwear')]
```

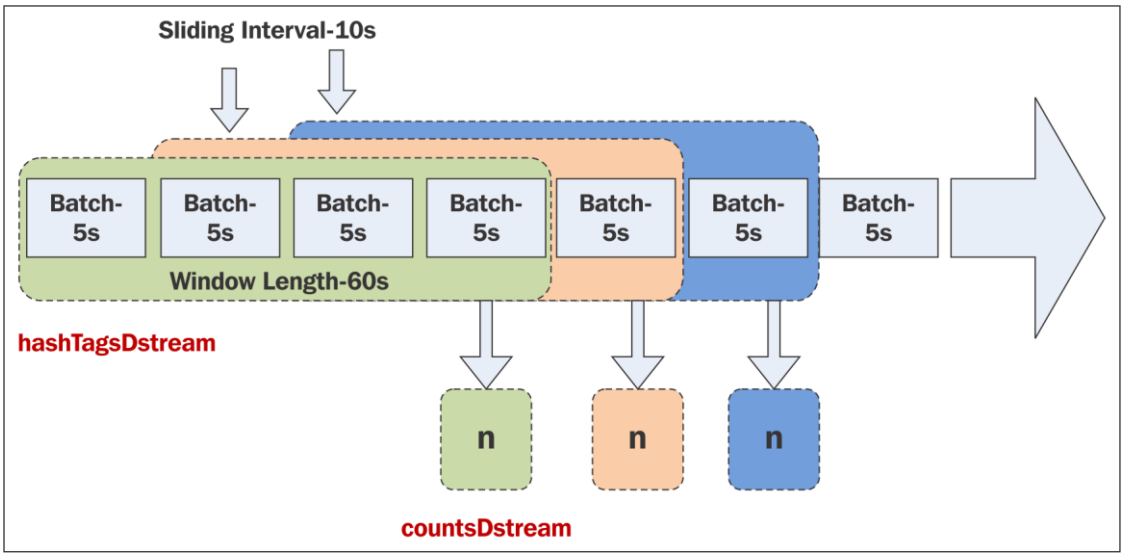
Chapter 5: Real-Time Analytics with Spark Streaming and Structured Streaming

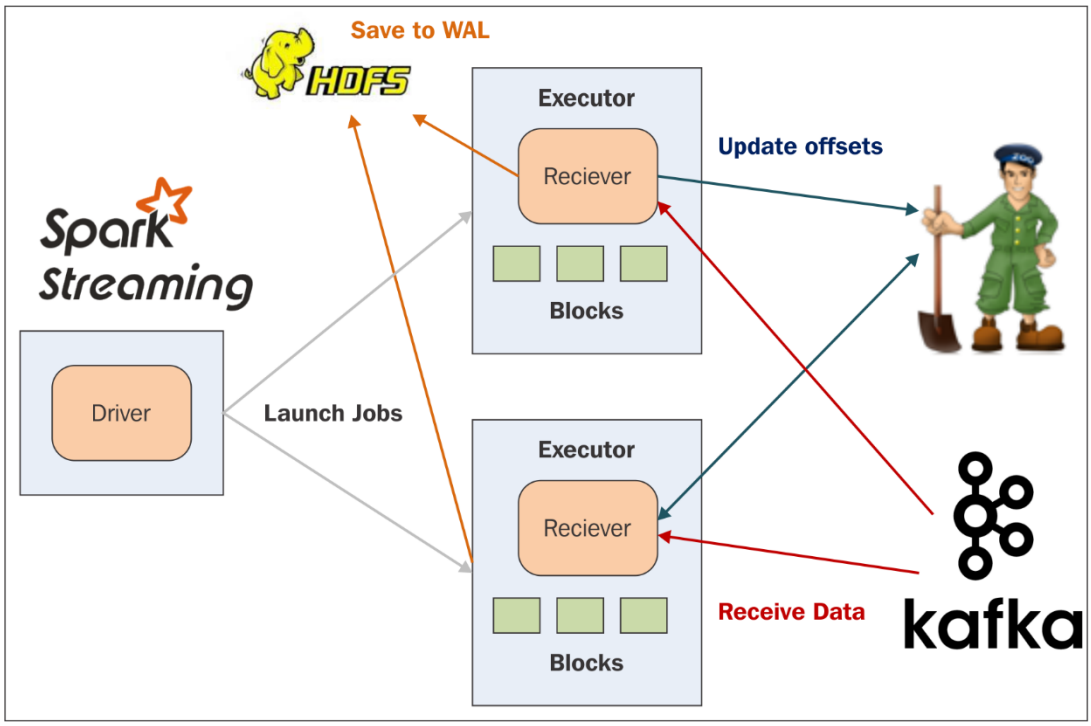


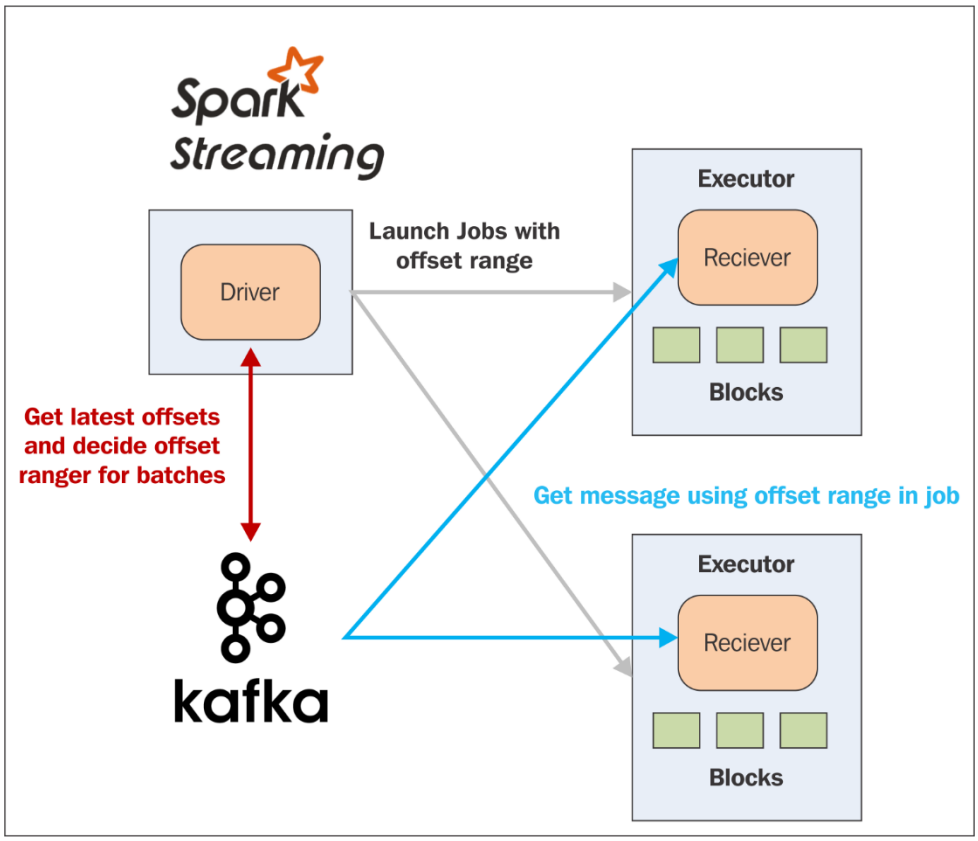


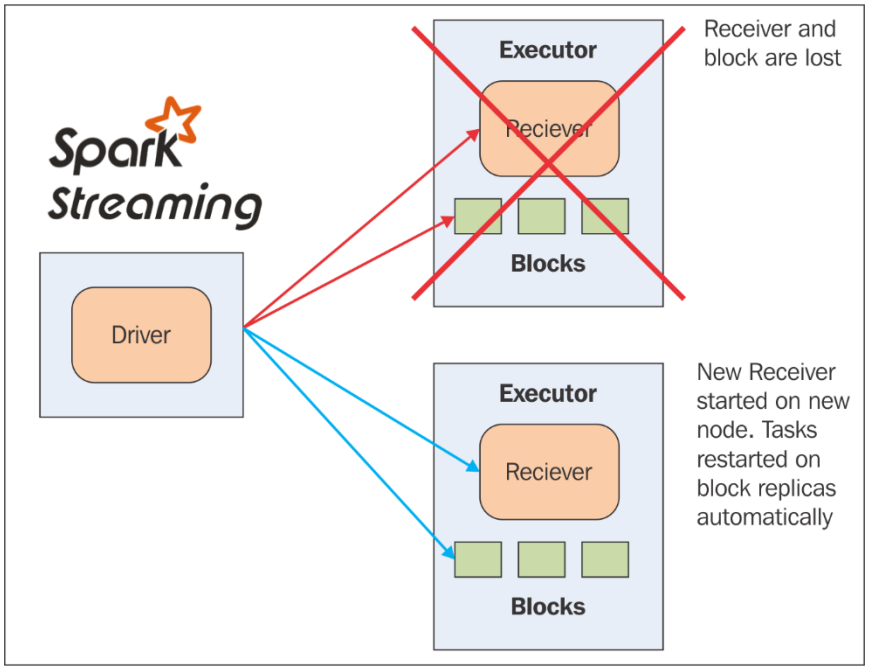
```
[root@quickstart ~]# nc -l
k 9999
spark
spark
spark
hadoop
hadoop
hadoop
spark
spark
spark
hadoop
hadoop
hadoop
```

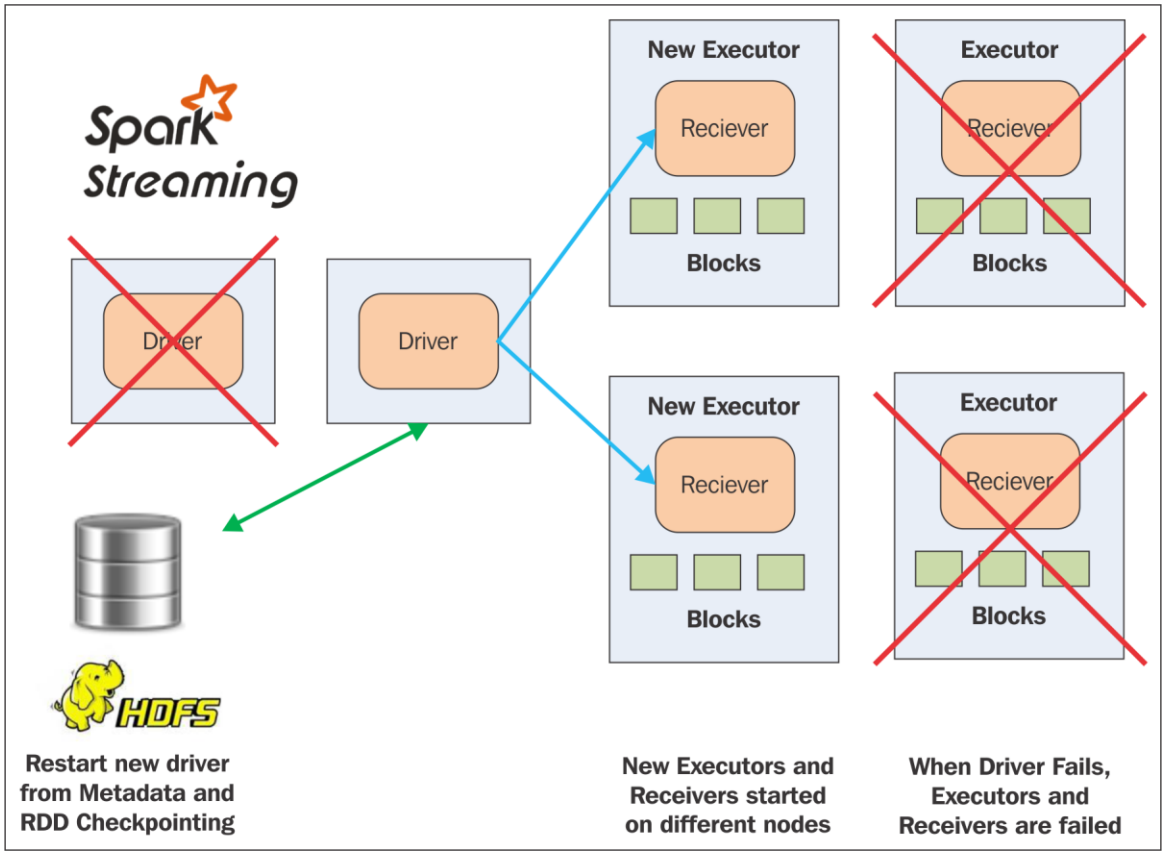
```
Time: 2015-12-19 06:44:54
-------------------------------------------
(u'hadoop', 3)
(u'spark', 3)

-------------------------------------------
Time: 2015-12-19 06:44:55
-------------------------------------------
(u'hadoop', 5)
(u'spark', 6)

-------------------------------------------
Time: 2015-12-19 06:44:56
-------------------------------------------
(u'hadoop', 6)
(u'spark', 6)

-------------------------------------------
Time: 2015-12-19 06:44:57
-------------------------------------------
(u'hadoop', 6)
(u'spark', 6)
```

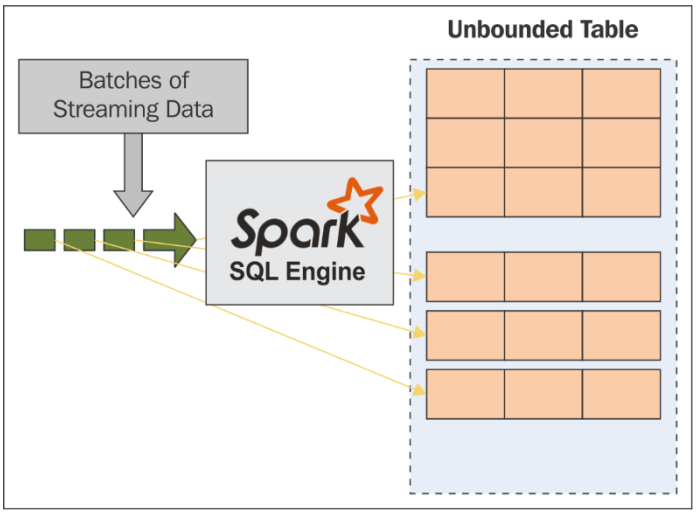


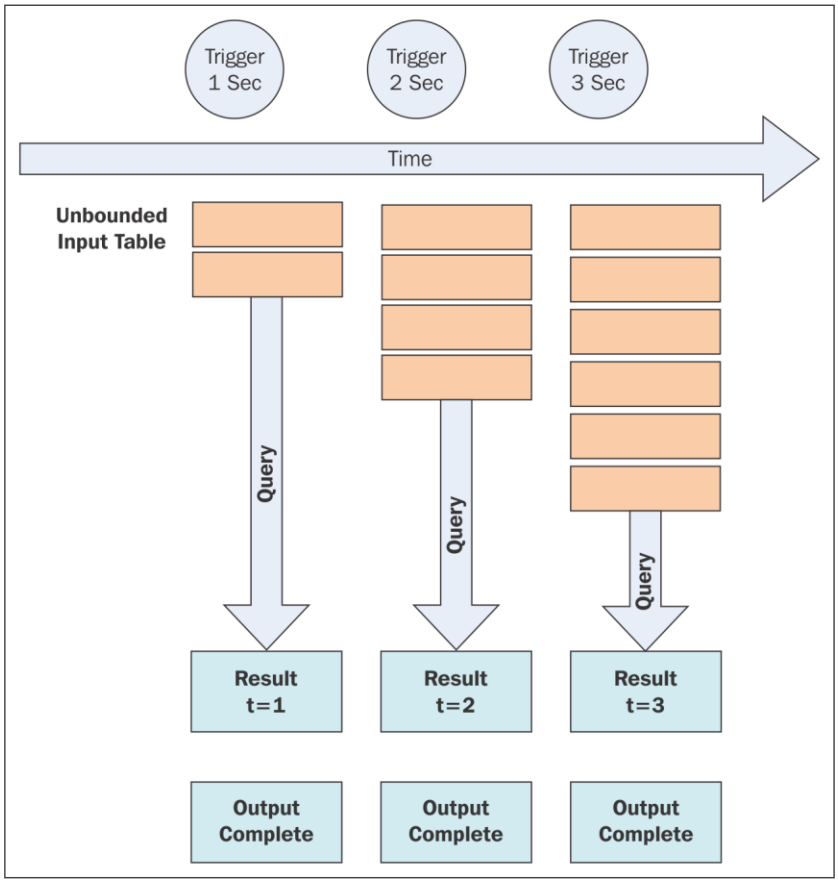

Save to WAL
HDFS
Executor
Reciever
Blocks
Update offsets
Spark Streaming
Driver
Launch Jobs
Executor
Reciever
Blocks
Receive Data
kafka


Spark Streaming
Driver
Launch Jobs with offset range
Executor
Reciever
Blocks
Get latest offsets and decide offset ranger for batches
Get message using offset range in job
kafka
Executor
Reciever
Blocks

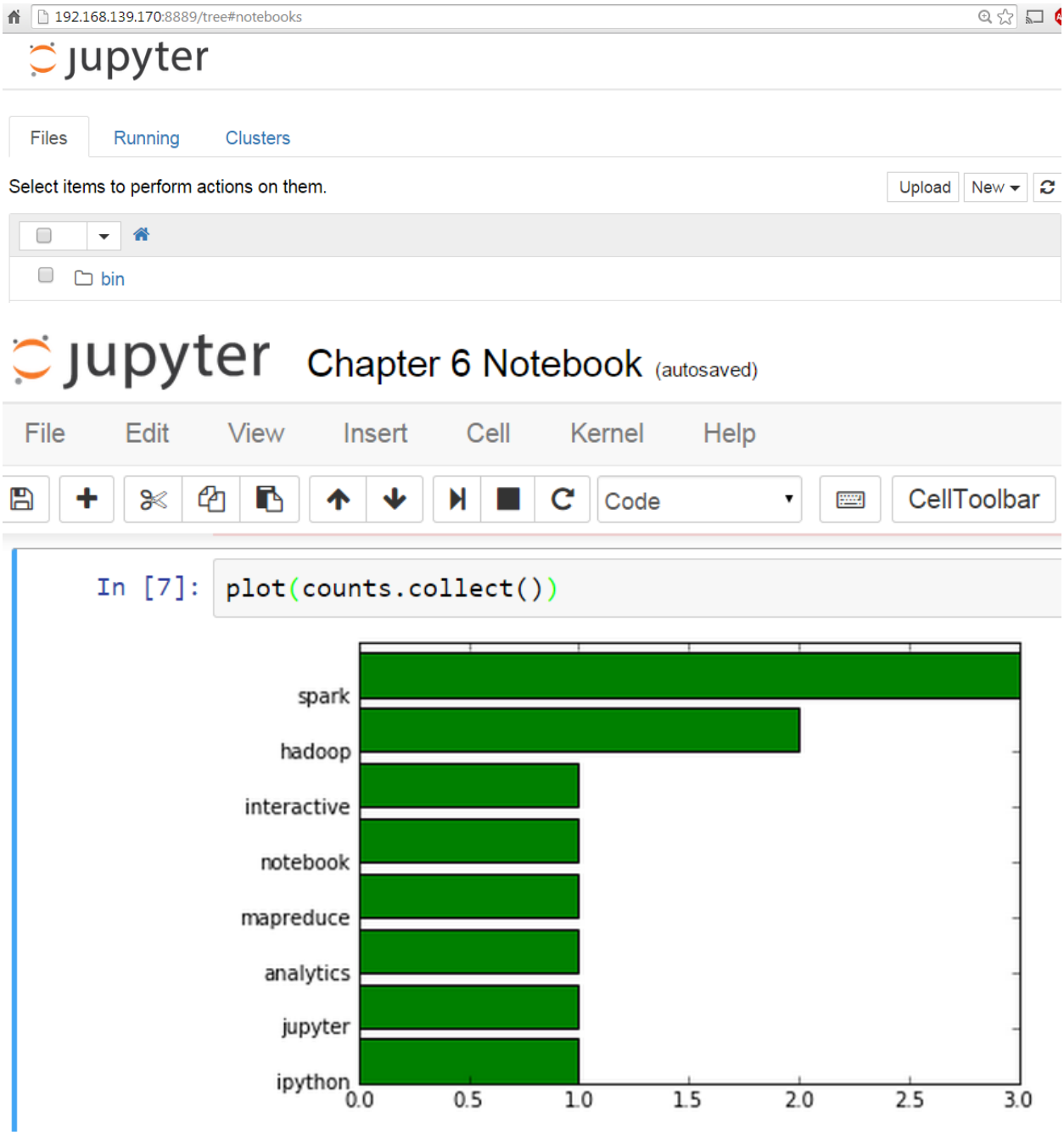

Spark Streaming
Driver
Executor
Reciever
Blocks
Receiver and block are lost
Executor
Reciever
Blocks
New Receiver started on new node. Tasks restarted on block replicas automatically

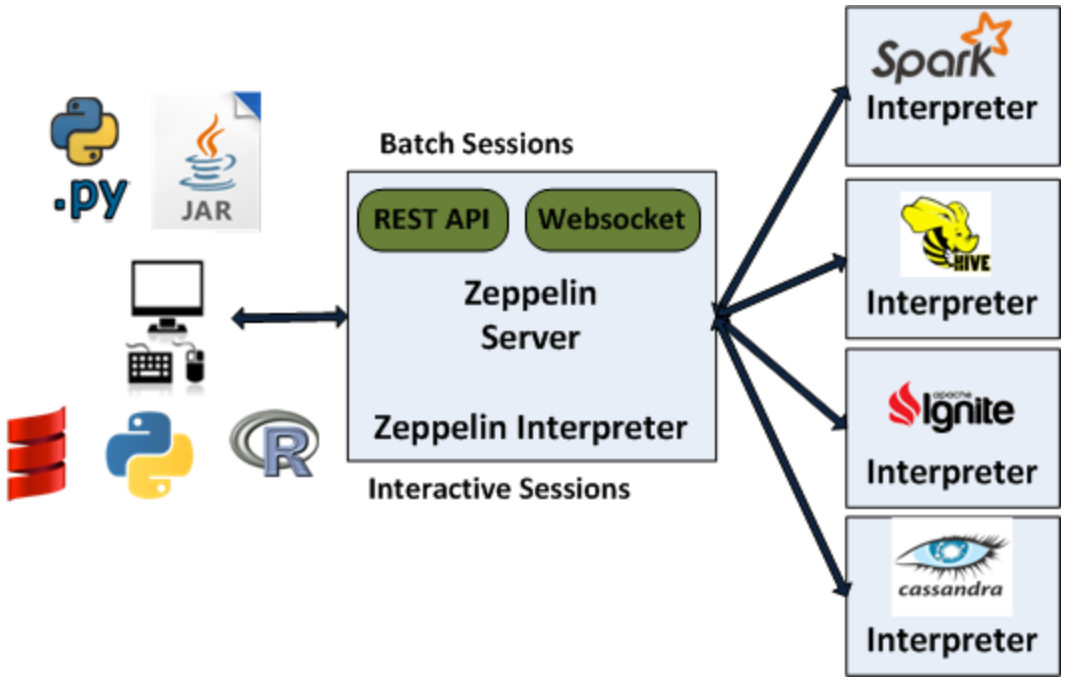

Spark Streaming
Driver
Driver
HDFS
New Executor
Reciever
Blocks
Executor
Reciever
Blocks
New Executor
Reciever
Blocks
Executor
Reciever
Blocks
Restart new driver from Metadata and RDD Checkpointing
New Executors and Receivers started on different nodes
When Driver Fails, Executors and Receivers are failed


Unbounded Table
Batches of Streaming Data
Spark
SQL Engine


Trigger 1 Sec
Trigger 2 Sec
Trigger 3 Sec
Time
Unbounded Input Table
Query
Query
Query
Result t=1
Result t=2
Result t=3
Output Complete
Output Complete
Output Complete

Chapter 6: Notebooks and Dataflows with Spark and Hadoop





Zeppelin Notebook Interpreter Configuration Search in your notebooks

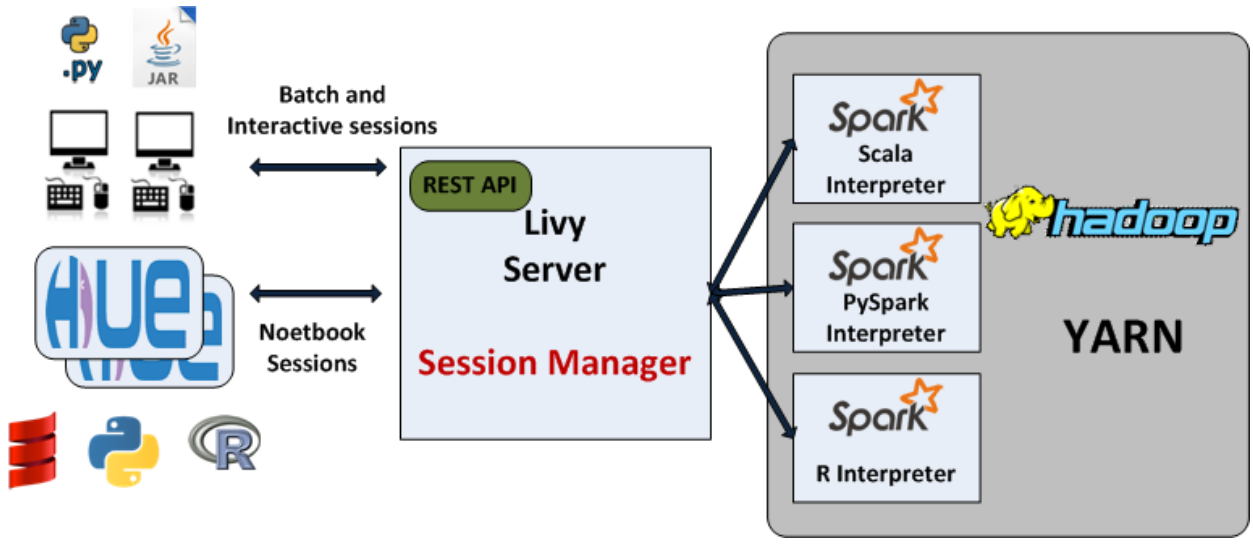
Chapter6ZepNote1

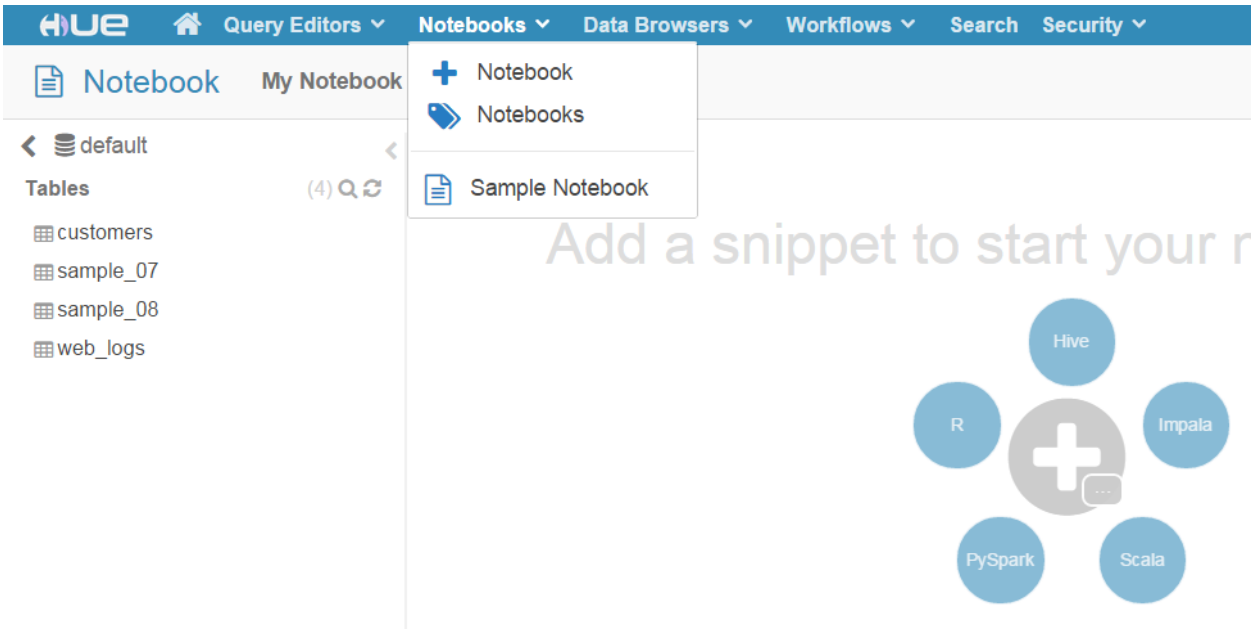
```
%sql select word, max(count) from words group by word
```

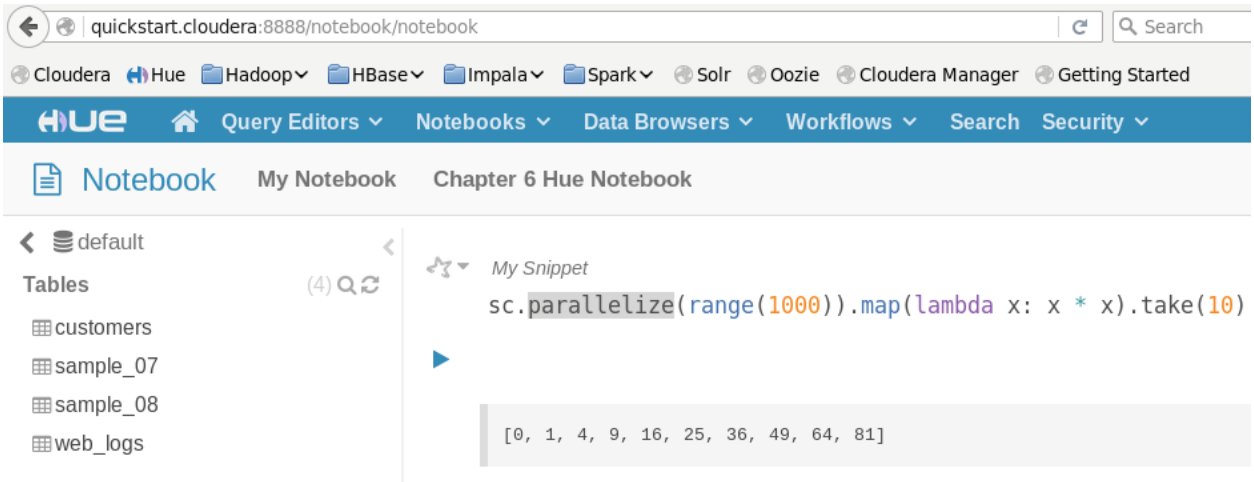
FINISHED

word	_c1
json	62
nodemanager	15
during	25
port	3
so	21
flume	178
configurations	149
nimbus	13
removed	60

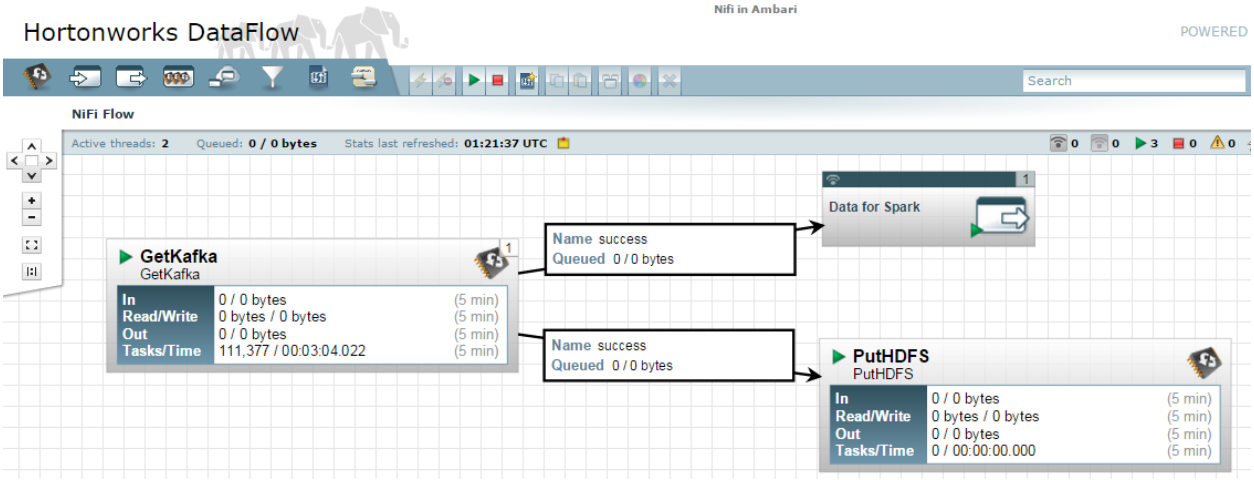
Took 16 seconds (outdated)


.py
JAR
Batch and
Interactive sessions
REST API
Livy
Server
Session Manager
Noetbook
Sessions
Spark
Scala
Interpreter
Spark
PySpark
Interpreter
Spark
R Interpreter
hadoop
YARN

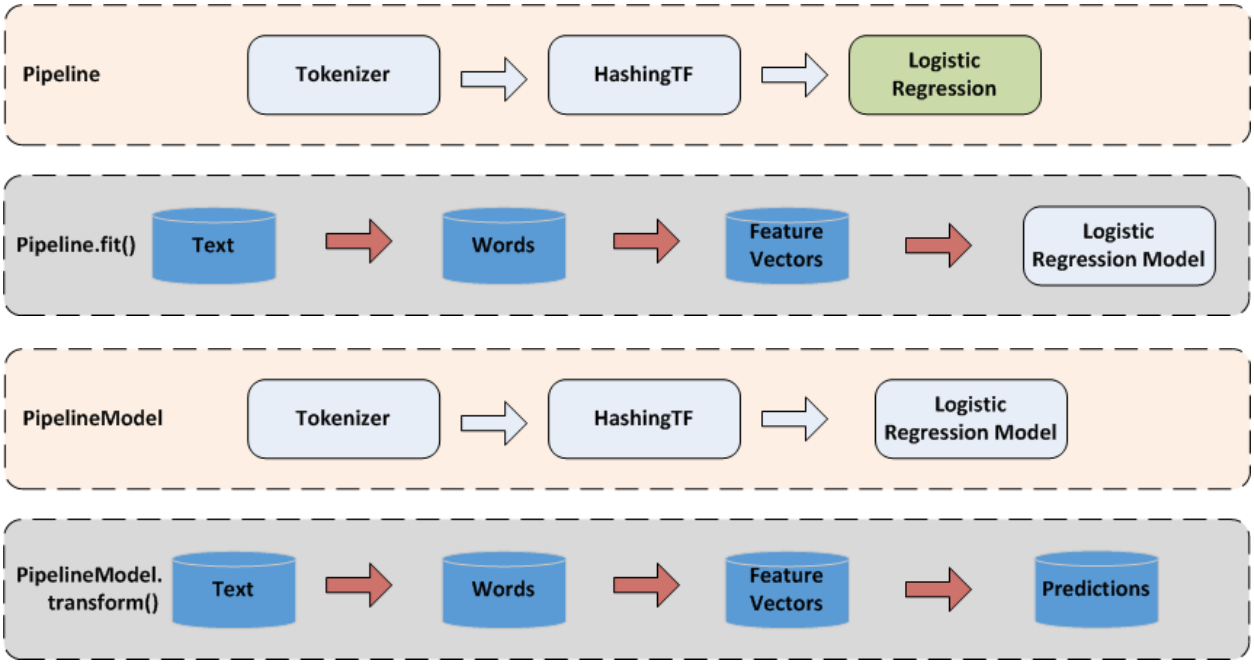

Query Editors
Notebooks
Data Browsers
Workflows
Search
Security
Notebook
My Notebook
Notebook
Notebooks
Sample Notebook
default
Tables
(4)
customers
sample_07
sample_08
web_logs
Add a snippet to start your r
Hive
R
Impala
PySpark
Scala


quickstart.cloudera:8888/notebook/notebook
Search
Cloudera
Hue
Hadoop
HBase
Impala
Spark
Solr
Oozie
Cloudera Manager
Getting Started
Query Editors
Notebooks
Data Browsers
Workflows
Search
Security
Notebook
My Notebook
Chapter 6 Hue Notebook
default
Tables
(4)
customers
sample_07
sample_08
web_logs
My Snippet
sc.parallelize(range(1000)).map(lambda x: x * x).take(10)
[0, 1, 4, 9, 16, 25, 36, 49, 64, 81]

Nifi in Ambari

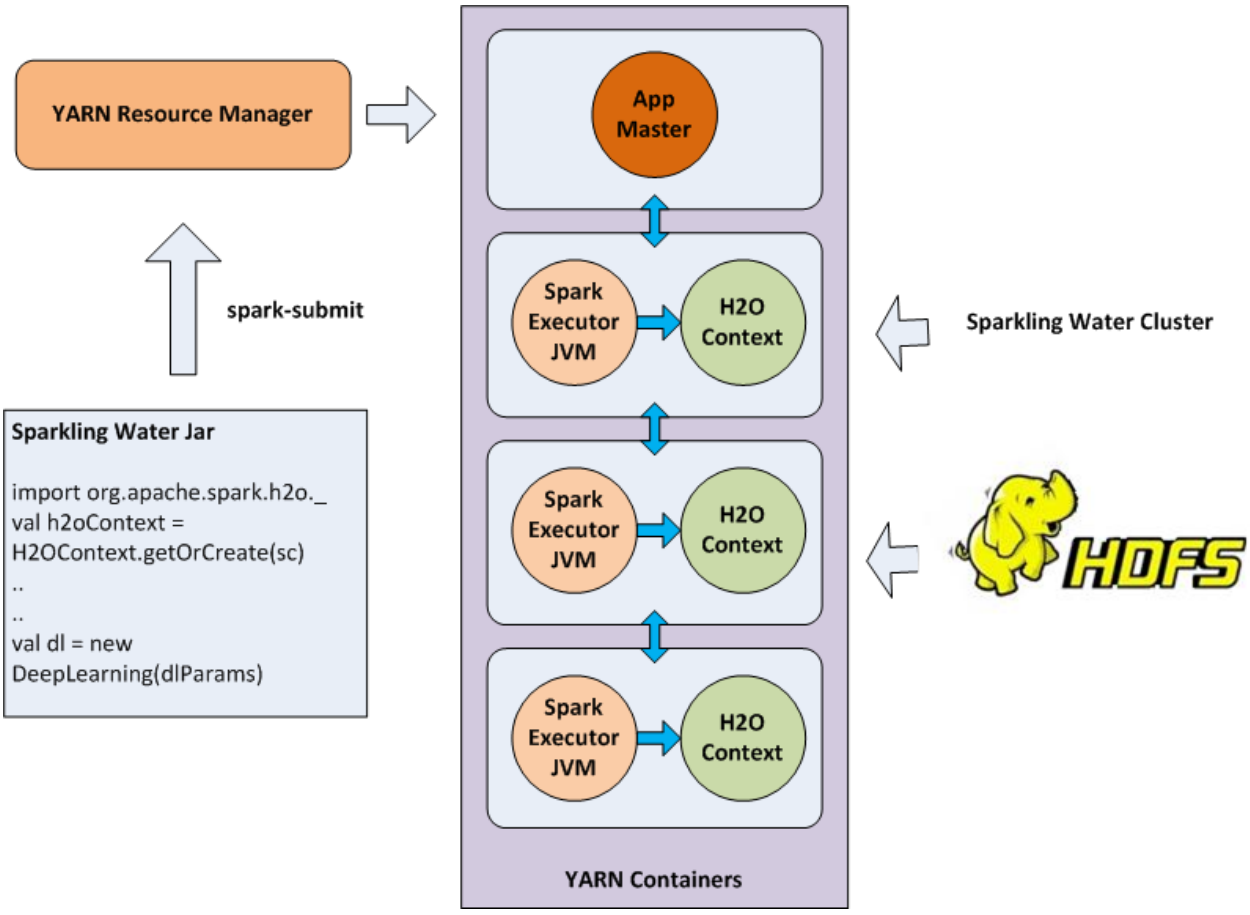


Chapter 7: Machine Learning with Spark and Hadoop

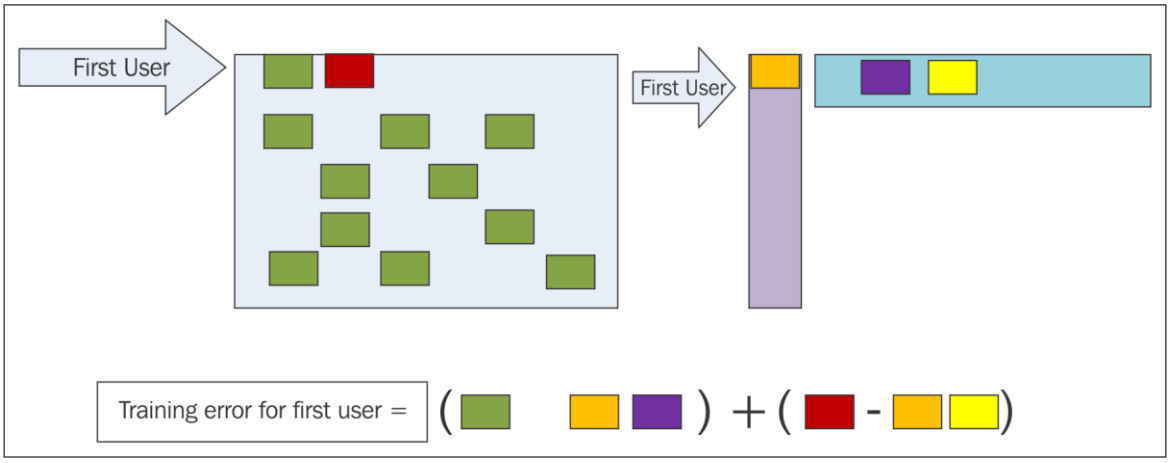



```
+---+--------------------+-----+
| id|                text|label|
+---+--------------------+-----+
|  0|apache spark rdd ...|  1.0|
|  1|       mllib pipeline|  0.0|
|  2|         hadoop mahout|  1.0|
|  3| mapreduce iterative|  0.0|
+---+--------------------+-----+
```

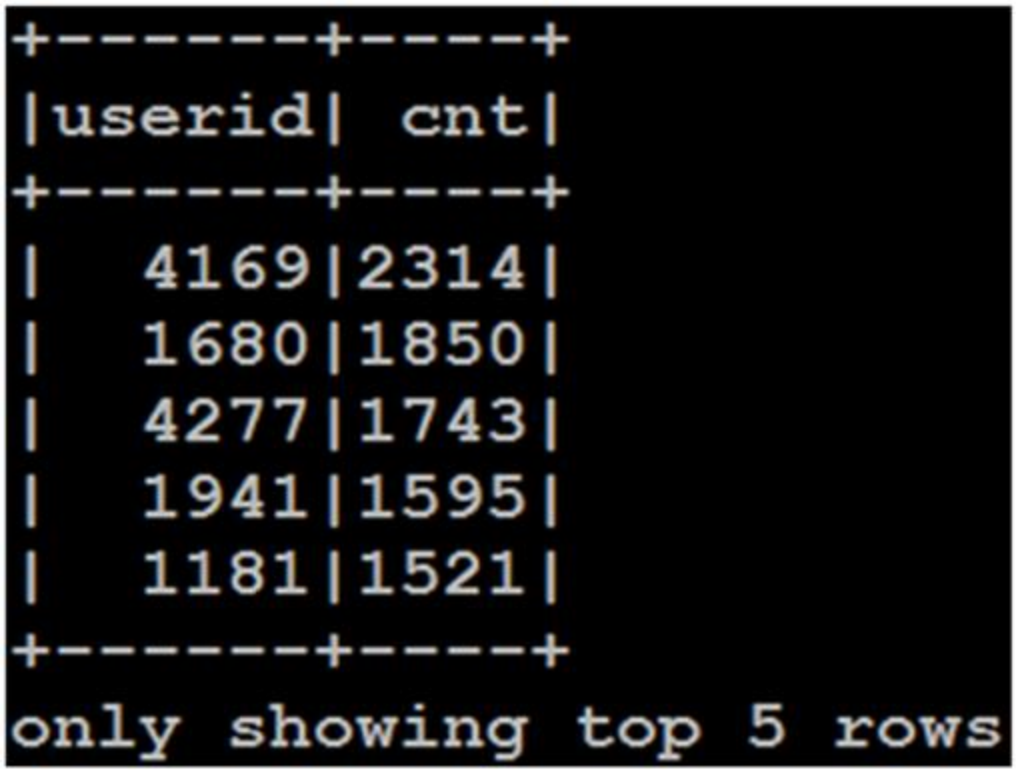
```
+---+--------------+
| id|          text|
+---+--------------+
|  4|         p q r|
|  5|mllib pipeline|
|  6|         x y z|
|  7| hadoop mahout|
+---+--------------+
```



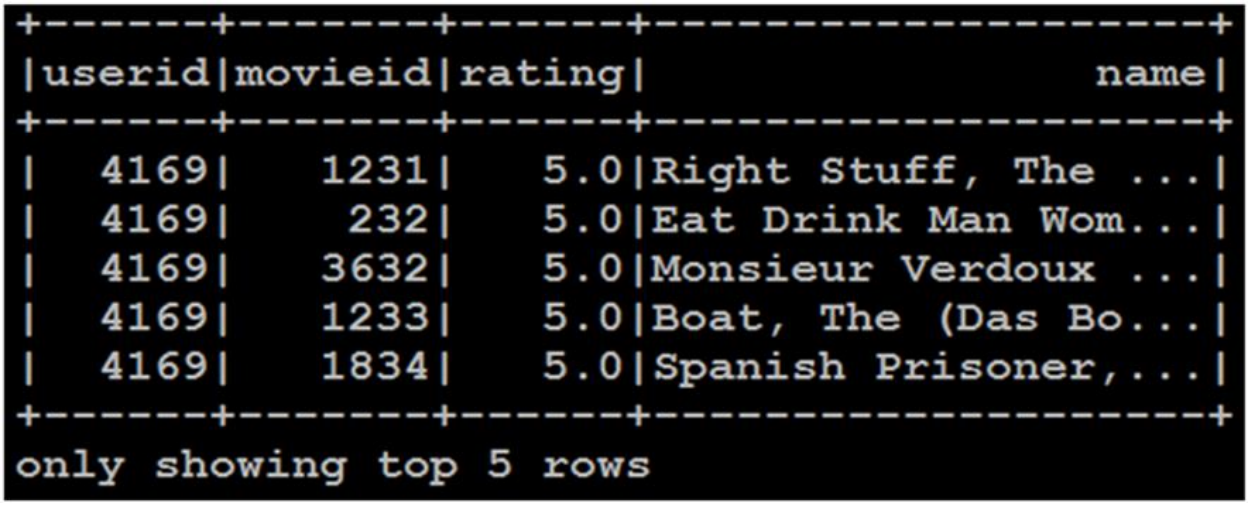
Chapter 8: Building Recommendation Systems with Spark and Mahout




```
+--------------------+-------+-------+------+
|                name|maxrtng|minrtng|cntusr|
+--------------------+-------+-------+------+
|American Beauty (...|    5.0|    1.0|  3428|
|Star Wars: Episod...|    5.0|    1.0|  2991|
|Star Wars: Episod...|    5.0|    1.0|  2990|
|Star Wars: Episod...|    5.0|    1.0|  2883|
|Jurassic Park (1993)|    5.0|    1.0|  2672|
+--------------------+-------+-------+------+
only showing top 5 rows
```

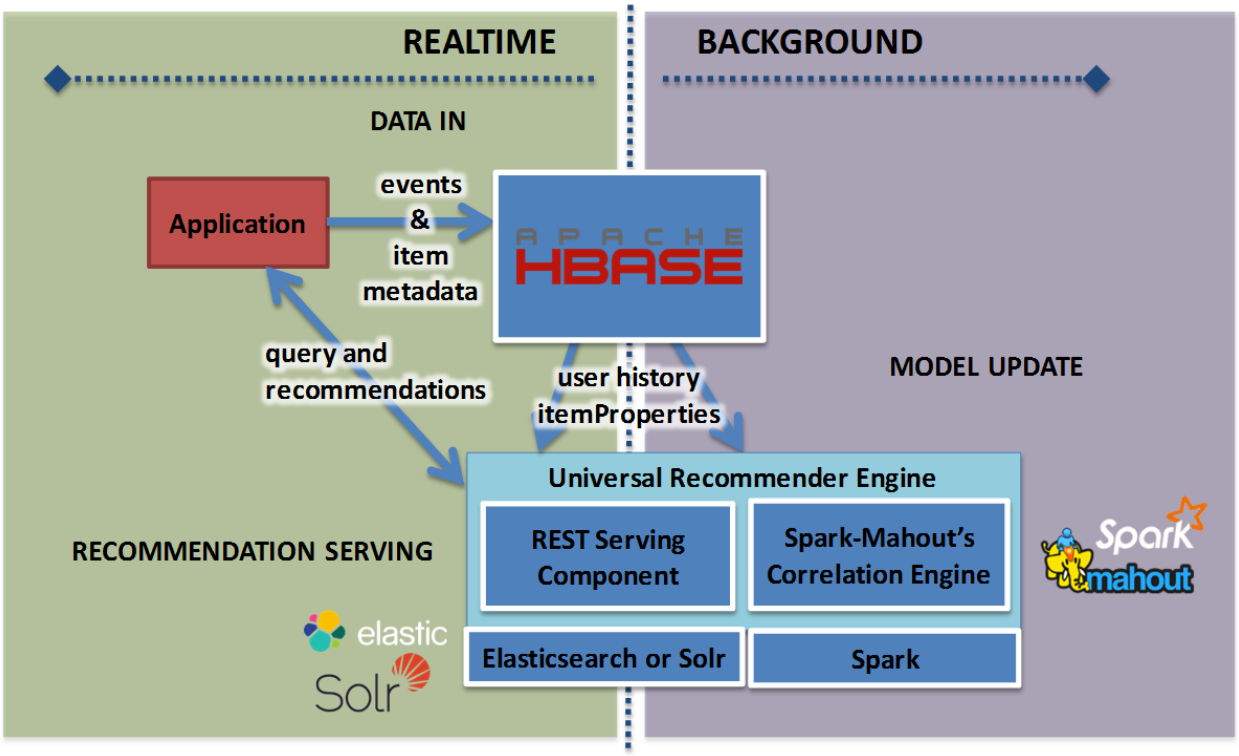


```
+------+----+
|userid| cnt|
+------+----+
|  4169|2314|
|  1680|1850|
|  4277|1743|
|  1941|1595|
|  1181|1521|
+------+----+
only showing top 5 rows
```

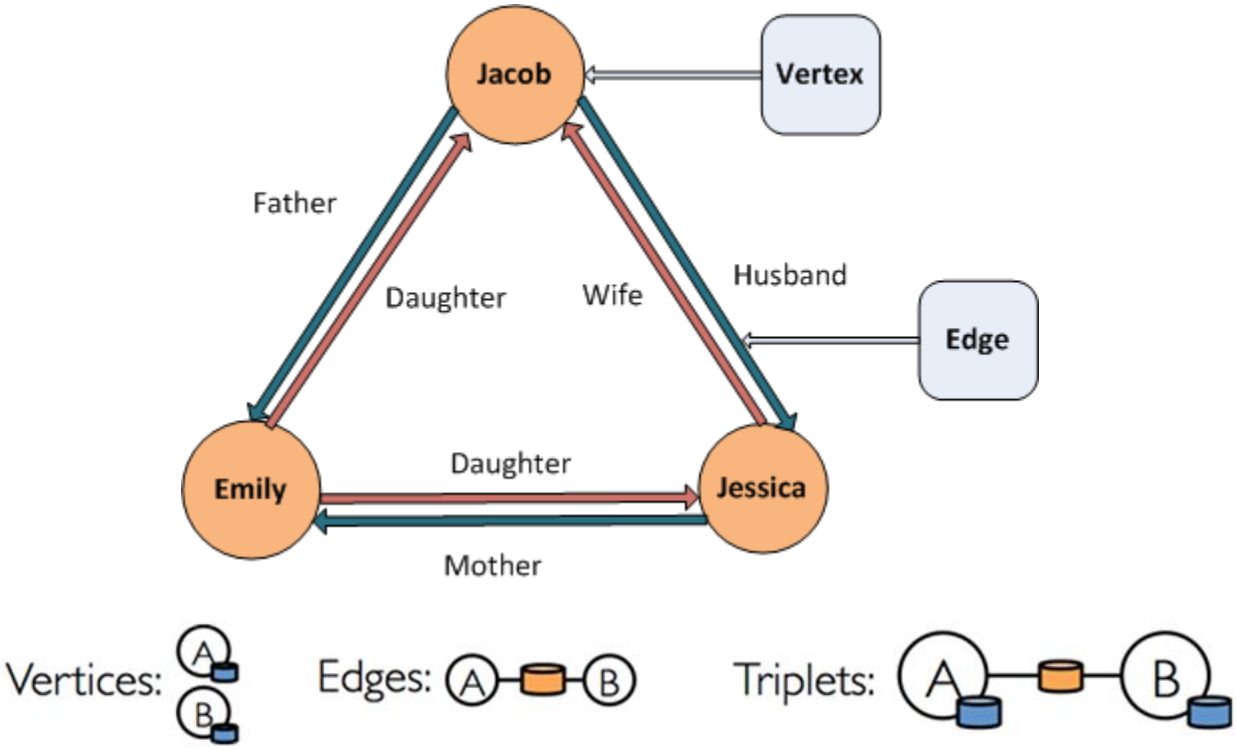


```
+------+-------+------+--------------------+
|userid|movieid|rating|                name|
+------+-------+------+--------------------+
|  4169|   1231|   5.0|Right Stuff, The ...|
|  4169|    232|   5.0|Eat Drink Man Wom...|
|  4169|   3632|   5.0|Monsieur Verdoux ...|
|  4169|   1233|   5.0|Boat, The (Das Bo...|
|  4169|   1834|   5.0|Spanish Prisoner,...|
+------+-------+------+--------------------+
only showing top 5 rows
```

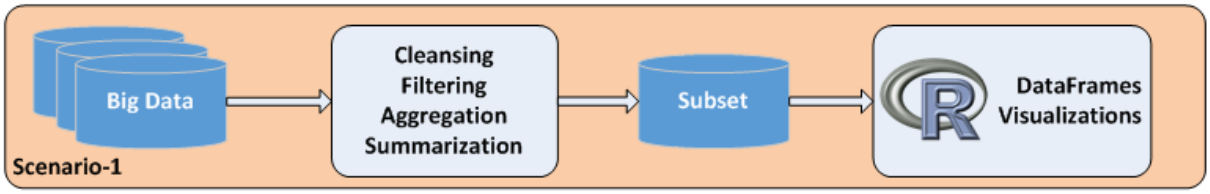
```
>>> dir(model)
['__class__', '__del__', '__delattr__', '__dict__', '__doc__', '__format
__', '__getattribute__', '__hash__', '__init__', '__module__', '__new__'
, '__reduce__', '__reduce_ex__', '__repr__', '__setattr__', '__sizeof__'
, '__str__', '__subclasshook__', '__weakref__', '_java_loader_class', '_
java_model', '_load_java', '_sc', 'call', 'load', 'predict', 'predictAll
', 'productFeatures', 'rank', 'recommendProducts', 'recommendUsers', 'sa
ve', 'userFeatures']
```

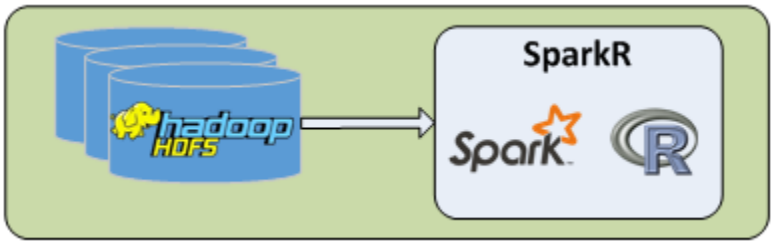

REALTIME
BACKGROUND
DATA IN
Application
events
&
item
metadata
APACHE
HBASE
query and
recommendations
user history
itemProperties
MODEL UPDATE
Universal Recommender Engine
REST Serving
Component
Spark-Mahout's
Correlation Engine
RECOMMENDATION SERVING
elastic
Solr
Elasticsearch or Solr
Spark
Spark
mahout

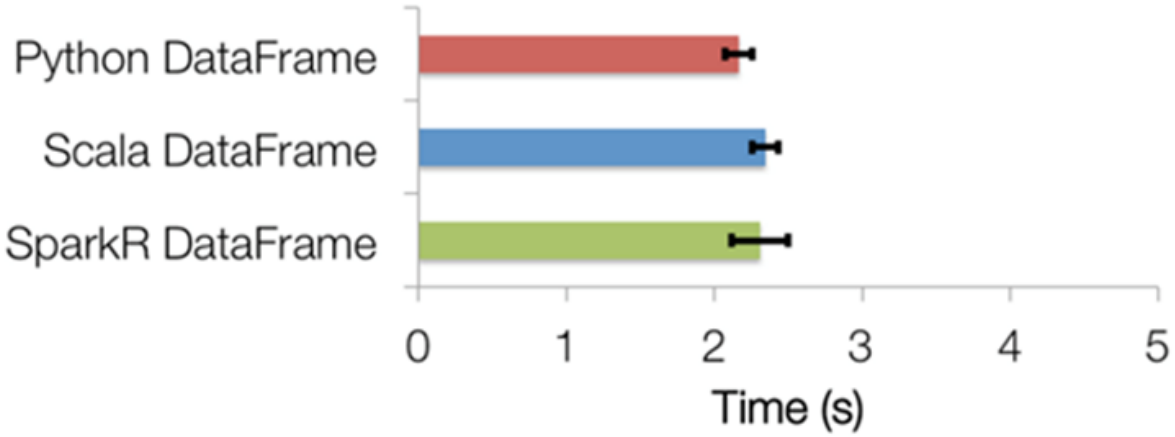
Chapter 9: Graph Analytics with GraphX

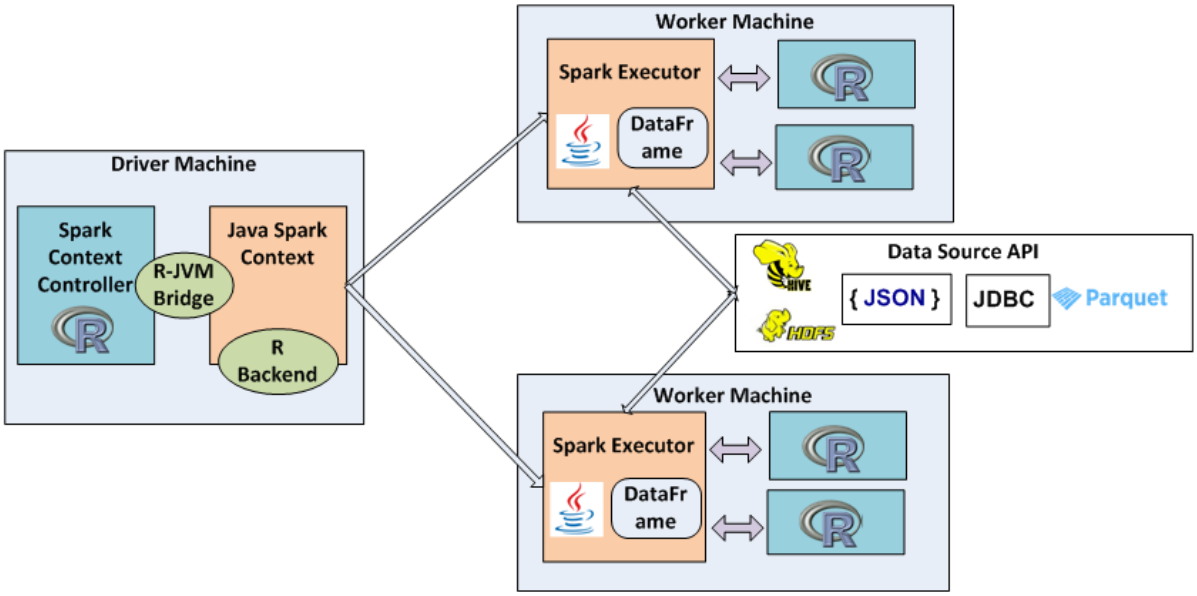


Chapter 10: Interactive Analytics with SparkR





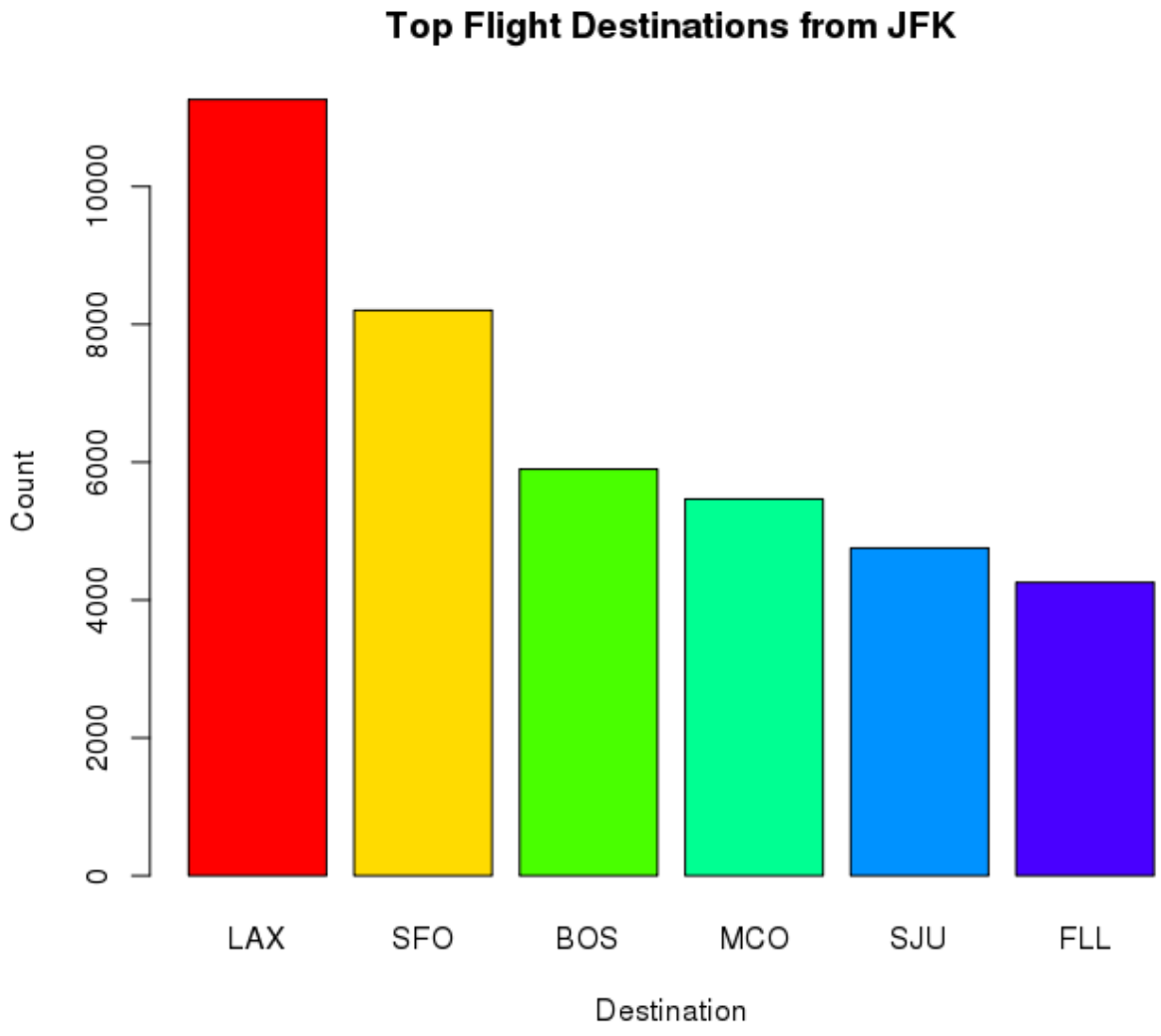


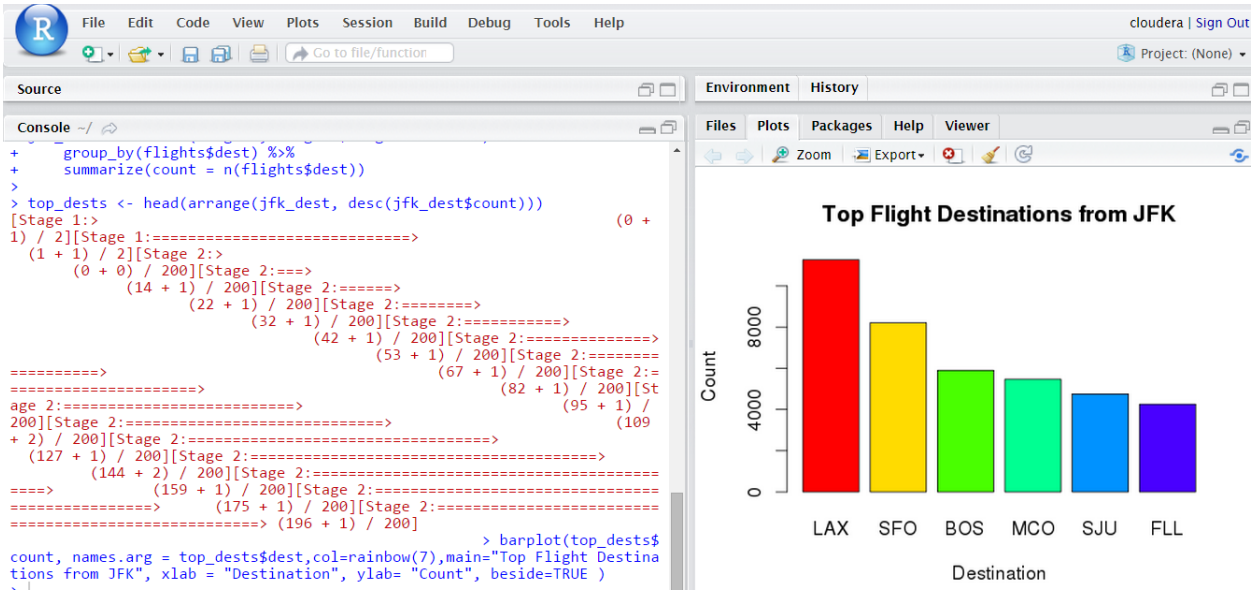



```
Launching java with spark-submit command /home/cloudera/spark-2.0.0-bin-hadoop2.7/bin/spar
k-submit   "--master" "local[2]" "sparkr-shell" /tmp/RtmpQ6DC4T/backend_port50fe71dbe06d
Setting default log level to "WARN".
To adjust logging level use sc.setLogLevel(newLevel).

 Welcome to
      ____              __
     / __/__  ___ _____/ /__
    _\ \/ _ \/ _ `/ __/  '_/
   /___/ .__/\_,_/_/ /_/\_\   version  2.0.0
      /_/


 SparkSession available as 'spark'.
>
```

```
Launching java with spark-submit command /home/cloudera/spark-2.0.0-bin-hadoop2.7/bin/spar
k-submit   "--master" "spark://quickstart.cloudera:7077" "sparkr-shell" /tmp/RtmpRknKgY/ba
ckend_port566f765d5f12
Setting default log level to "WARN".
To adjust logging level use sc.setLogLevel(newLevel).

 Welcome to
      ____              __
     / __/__  ___ _____/ /__
    _\ \/ _ \/ _ `/ __/  '_/
   /___/ .__/\_,_/_/ /_/\_\   version  2.0.0
      /_/


 SparkSession available as 'spark'.
>
```

```
Launching java with spark-submit command /home/cloudera/spark-2.0.0-bin-hadoop2.7/bin/spar
k-submit   "--master" "yarn" "sparkr-shell" /tmp/RtmprccjdS/backend_port1bc3213922d9
Setting default log level to "WARN".
To adjust logging level use sc.setLogLevel(newLevel).

 Welcome to
      ____              __
     / __/__  ___ _____/ /__
    _\ \/ _ \/ _ `/ __/  '_/
   /___/ .__/\_,_/_/ /_/\_\   version  2.0.0
      /_/

 SparkSession available as 'spark'.
>
```






```
+     group_by(flights$dest) %>%
+     summarize(count = n(flights$dest))
>
> top_dests <- head(arrange(jfk_dest, desc(jfk_dest$count)))
[Stage 1:>                                                          (0 + 1) / 2][Stage 1:==============================>
  (1 + 1) / 2][Stage 2:>
       (0 + 0) / 200][Stage 2:===>
             (14 + 1) / 200][Stage 2:======>
                    (22 + 1) / 200][Stage 2:========>
                           (32 + 1) / 200][Stage 2:===========>
                                  (42 + 1) / 200][Stage 2:===============>
                                         (53 + 1) / 200][Stage 2:==========
==========>                                     (67 + 1) / 200][Stage 2:=
=====================>                                (82 + 1) / 200][St
age 2:============================>                          (95 + 1) /
200][Stage 2:==============================>                  (109
+ 2) / 200][Stage 2:====================================>
  (127 + 1) / 200][Stage 2:=======================================>
         (144 + 2) / 200][Stage 2:=========================================
====>           (159 + 1) / 200][Stage 2:=================================
================>      (175 + 1) / 200][Stage 2:=========================
=========================> (196 + 1) / 200]
                                                    > barplot(top_dests$
count, names.arg = top_dests$dest,col=rainbow(7),main="Top Flight Destina
tions from JFK", xlab = "Destination", ylab= "Count", beside=TRUE )
```