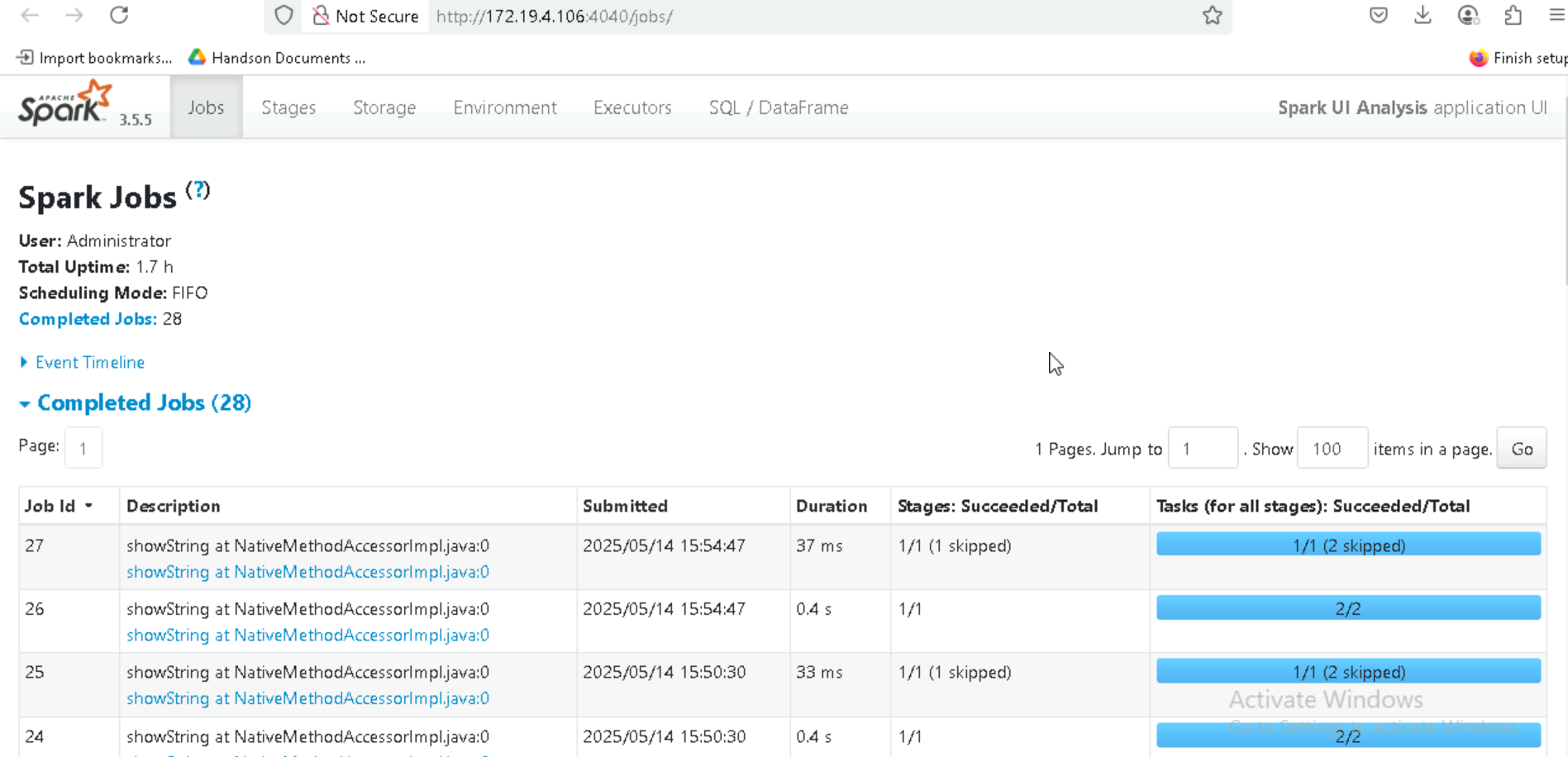
**SPARK UI ANALYSIS**

Spark UI can be accessed in development mode at localhost:4040 by default

**Jobs View**

Displays summary page of all jobs in the Spark application and details page for each job



**Event Timeline**

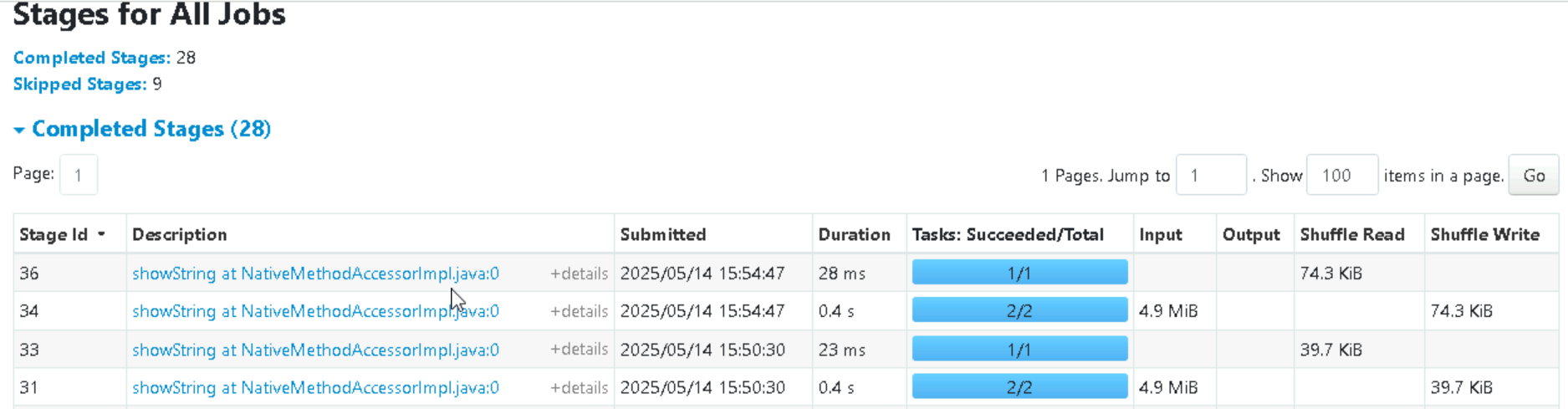
Displays in chronological order the events related to the executors and the jobs

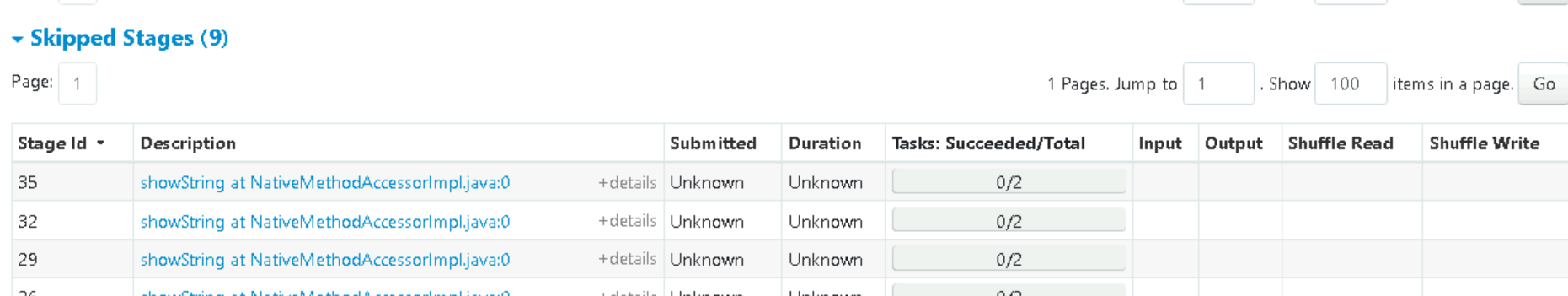


**Stages View**

Shows the input data size of each stage,shuffle read and write data size,and number of tasks.

Additionally skipped stages also mentioned.

****

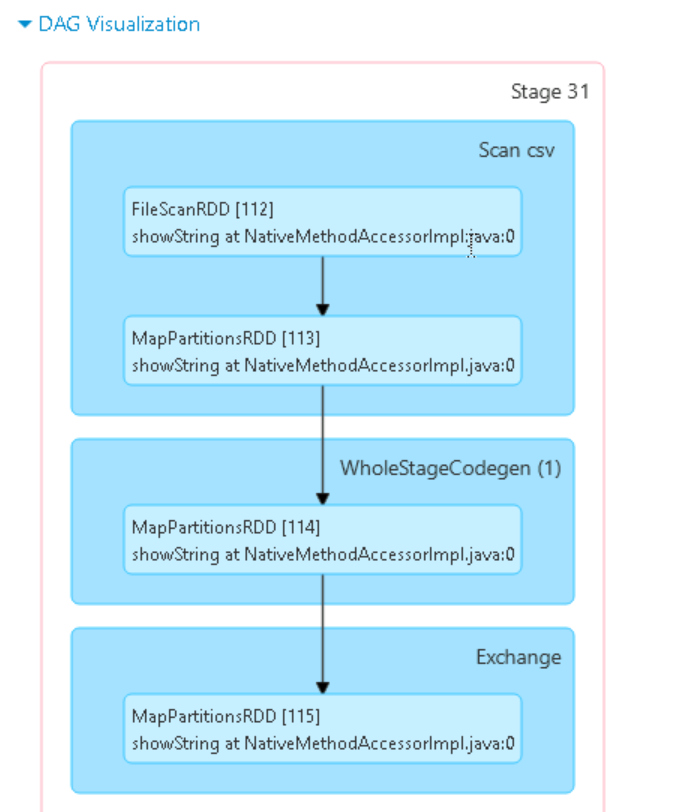


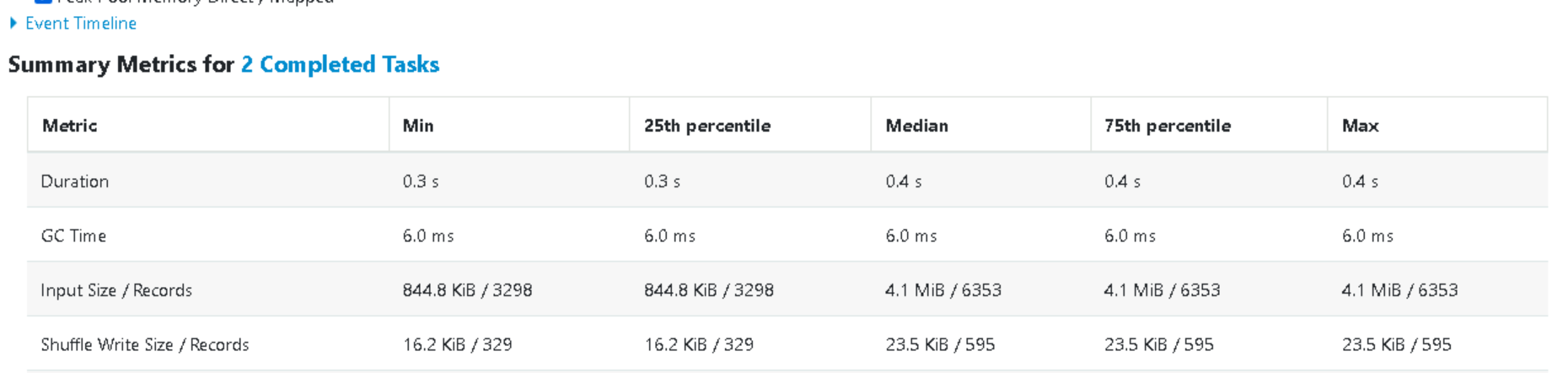
**DAG Visualisation**

It is the visual representation steps breakdown of each stage.

Vertices-RDD or Dataframes

Edges-operations applied on RDD

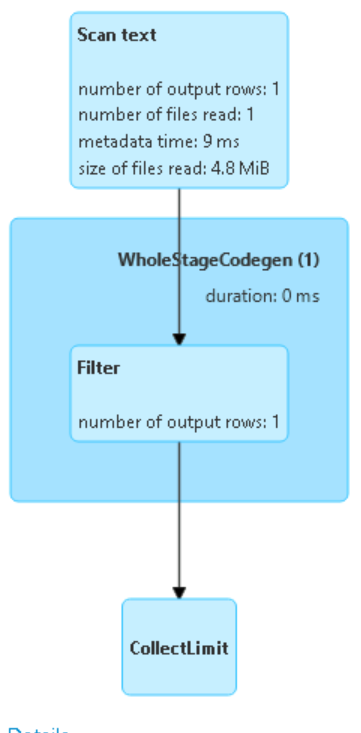
****

****

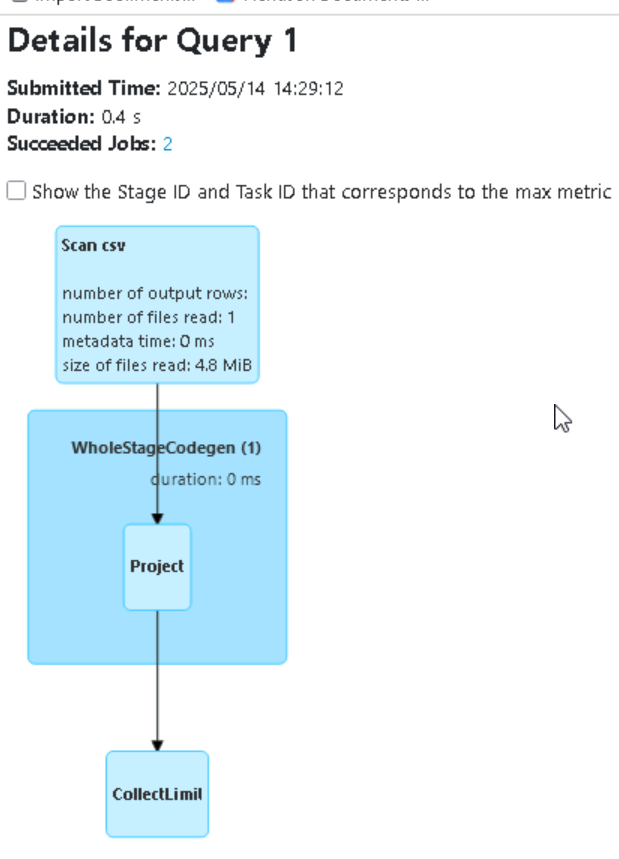
**SQL/DataFrame**

This show each query in the overall job and its duration as well as job IDs associated with each query.

Sql query 1st Action

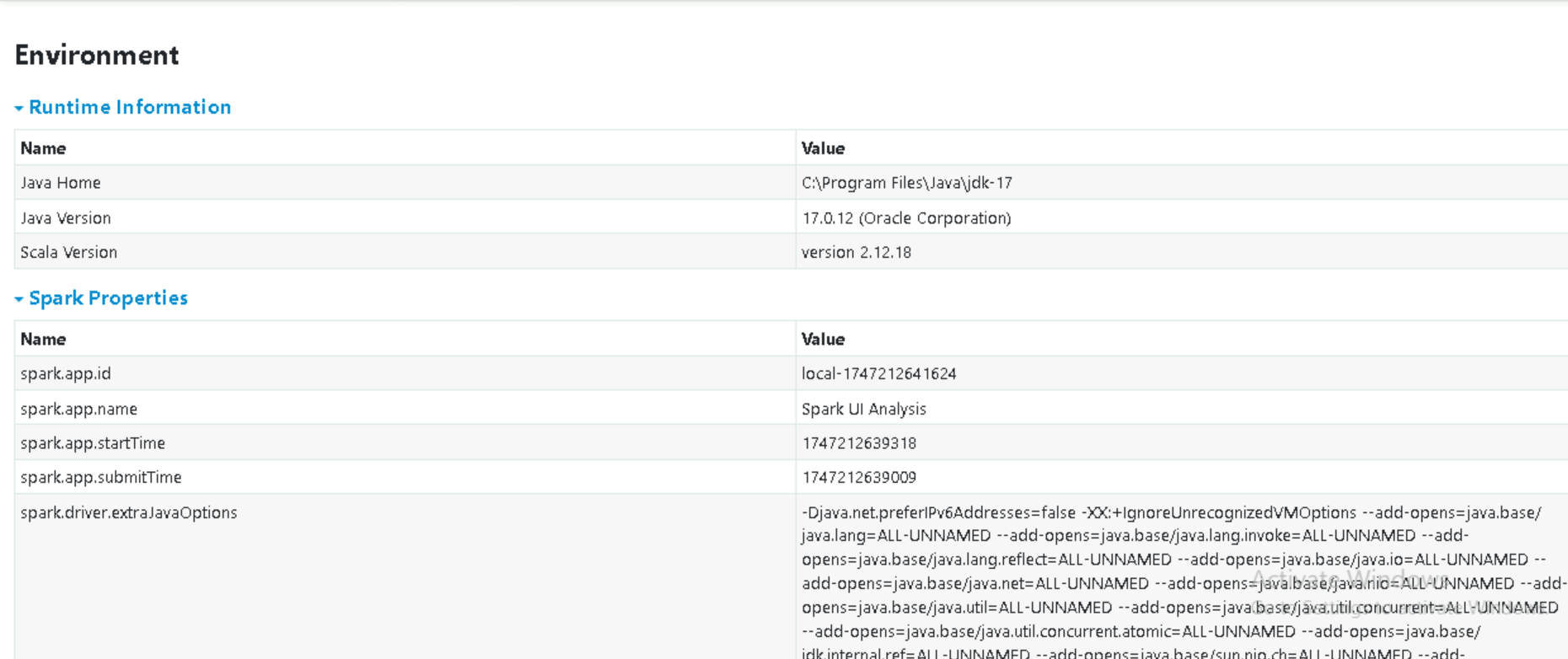


Sql query 1



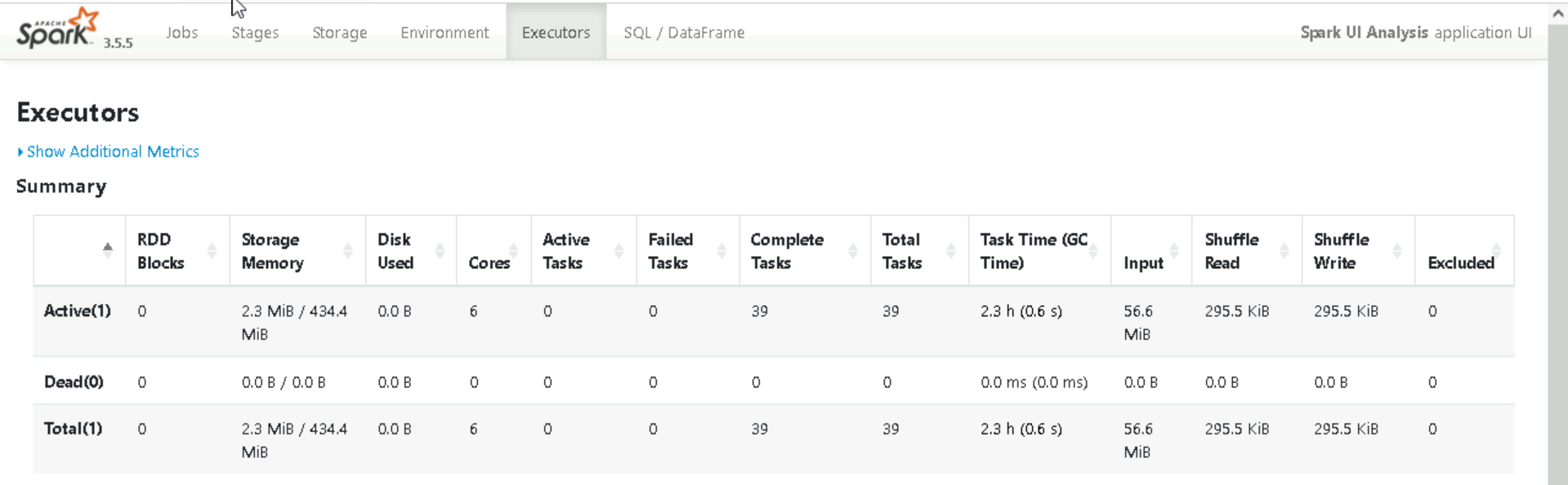
**Enviroment View**

Displays the values for different environment and configuration variables.

****

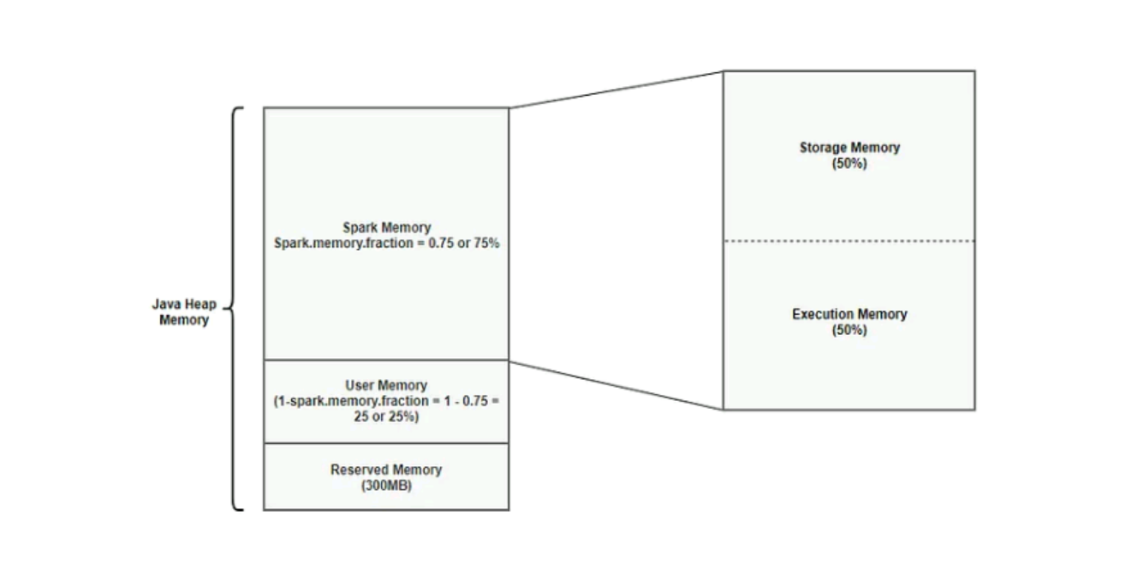
**Executors View**

The executors tab displays the summary information about the executors that were created for application,including memory and disk usage and task and shuffle information

****

**Types of Memory in Spark**

**JVM Memory**

****

Spark runs on Java Virtual Machine,and memory is used by each executor is divided into several types:

**1.Reseved Memory**

* + This is the memory reserved by the system and its size is hardcoded.
  + Used by Spark itself for internal operations(logs,metadata etc.)
  + Reserved memory is around 300MB

**2.Spark Memory(Unified Memory)**

* + Splits the memory into 2:

1.execution memory

2.storage memory

1.Execution Memory

i.Used for computations like joins,aggregations,sorting,shuffles

ii.Temporary memory ,released after task execution

2.Storage Memory

i.Use for storing all of the cache data,broadcast variables are also stored here.

ii.spark clears space for new cache requests by removing old cached objects based on LRU (Least Recently Used)mechanism,once the cache data is out of storage it is either returns to disk or recomputed based on configuration.

**3.user Memory**

* + Memory used by user defined code and spark internal datastructure.
  + Not managed by spark memory manager.

Eg:Memory used in custom functions or external libraries

**Memory Management in Spark**

Memory management in spark controls how spark uses RAM available to each executor to operations like :

Sparks memory management is divided into 2 types:

1.Static Memory management

2.Unified memory management

**Static Memory management**

* Memory splitted into 2 parts:execution and storage
* No sharing between execution and storage memory
* You have to manually set memory sizes
* If one part was idle,its memory could’nt be reused
* Less efficient and harder to tune.

**Unified Memory Management**

* Execution and storage share one memory pool
* Memory can be borrowed between them as needed
* Easier to manage with fewer settings
* Better use of available memory
* More efficient and Faster performance