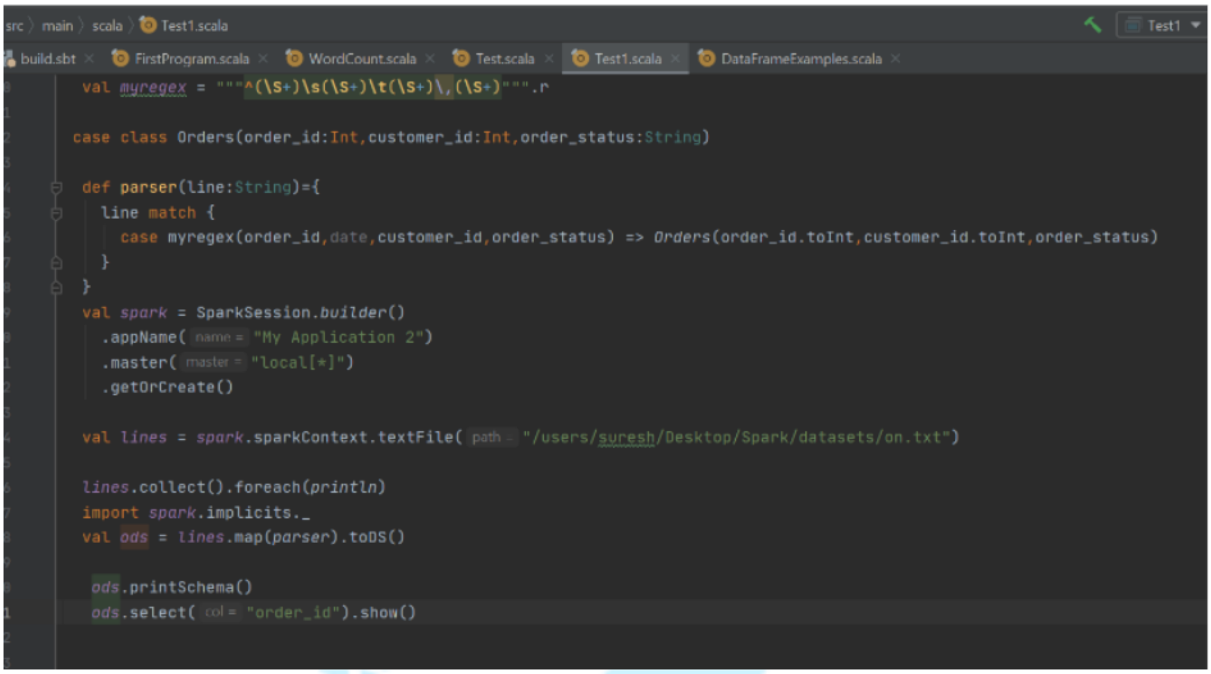
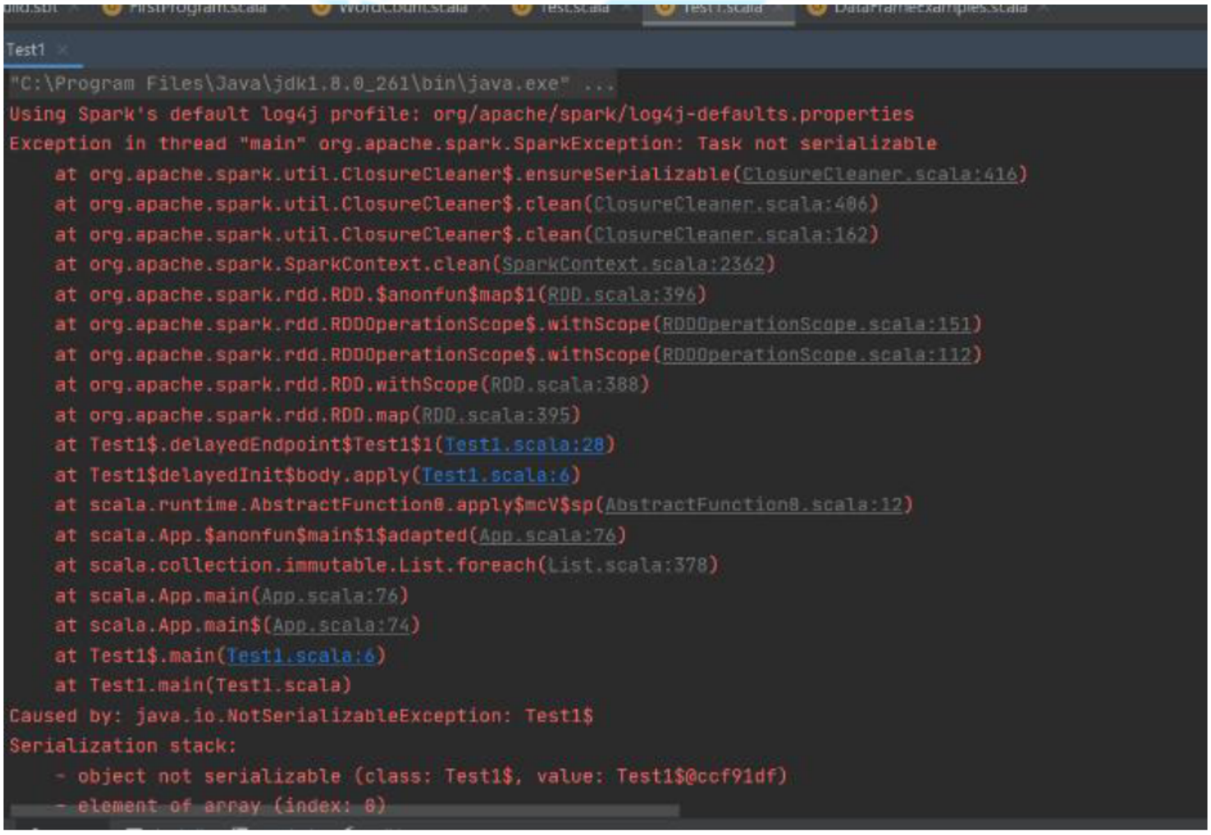
Week 13 FAQs

W13:1 Sam is getting Task not serializable Error.



```
  val myregex = """^(\S+)\s(\S+)\t(\S+)\,(\S+)""".r

 case class Orders(order_id:Int,customer_id:Int,order_status:String)

  def parser(line:String)={
    line match {
      case myregex(order_id,date,customer_id,order_status) => Orders(order_id.toInt,customer_id.toInt,order_status)
    }
  }
  val spark = SparkSession.builder()
    .appName( name = "My Application 2")
    .master( master = "local[*]")
    .getOrCreate()

  val lines = spark.sparkContext.textFile( path = "/users/suresh/Desktop/Spark/datasets/on.txt")

  lines.collect().foreach(println)
  import spark.implicits._
  val ods = lines.map(parser).toDS()

   ods.printSchema()
   ods.select( col = "order_id").show()
```

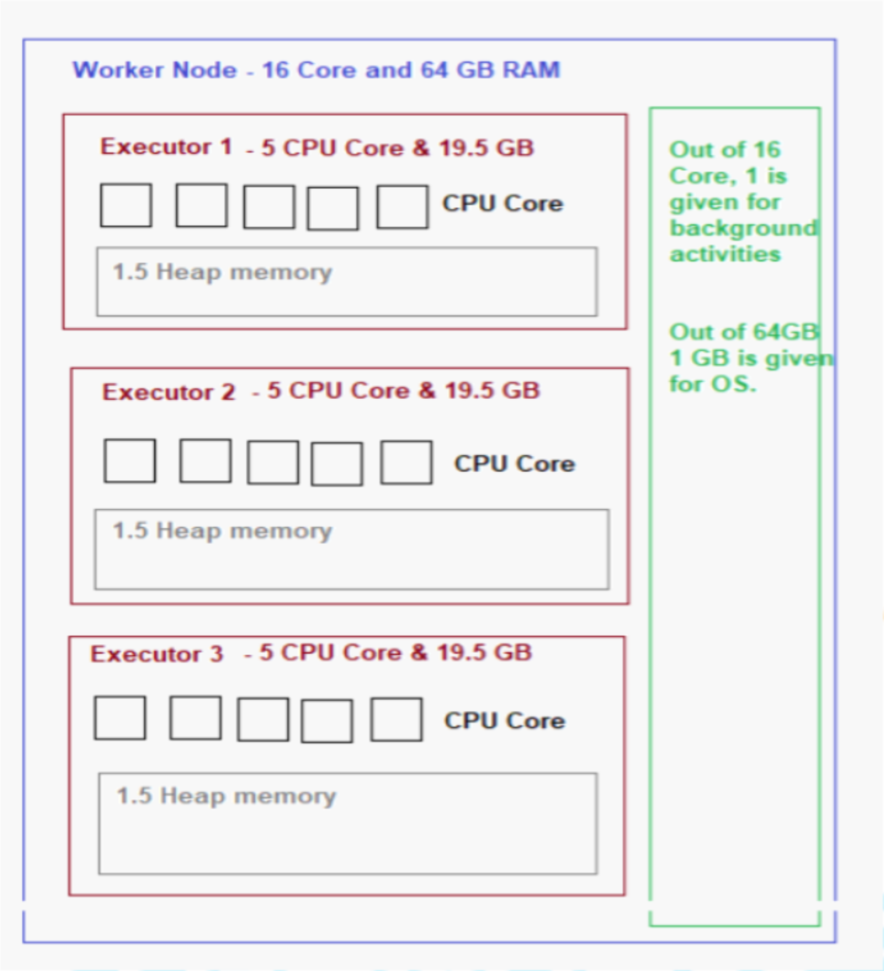


```
"C:\Program Files\Java\jdk1.8.0_261\bin\java.exe" ...
Using Spark's default log4j profile: org/apache/spark/log4j-defaults.properties
Exception in thread "main" org.apache.spark.SparkException: Task not serializable
    at org.apache.spark.util.ClosureCleaner$.ensureSerializable(ClosureCleaner.scala:416)
    at org.apache.spark.util.ClosureCleaner$.clean(ClosureCleaner.scala:406)
    at org.apache.spark.util.ClosureCleaner$.clean(ClosureCleaner.scala:162)
    at org.apache.spark.SparkContext.clean(SparkContext.scala:2362)
    at org.apache.spark.rdd.RDD.$anonfun$map$1(RDD.scala:396)
    at org.apache.spark.rdd.RDDOperationScope$.withScope(RDDOperationScope.scala:151)
    at org.apache.spark.rdd.RDDOperationScope$.withScope(RDDOperationScope.scala:112)
    at org.apache.spark.rdd.RDD.withScope(RDD.scala:388)
    at org.apache.spark.rdd.RDD.map(RDD.scala:395)
    at Test1$.delayedEndpoint$Test1$1(Test1.scala:28)
    at Test1$delayedInit$body.apply(Test1.scala:6)
    at scala.runtime.AbstractFunction0.apply$mcV$sp(AbstractFunction0.scala:12)
    at scala.App.$anonfun$main$1$adapted(App.scala:76)
    at scala.collection.immutable.List.foreach(List.scala:378)
    at scala.App.main(App.scala:76)
    at scala.App.main$(App.scala:74)
    at Test1$.main(Test1.scala:6)
    at Test1.main(Test1.scala)
Caused by: java.io.NotSerializableException: Test1$
Serialization stack:
    - object not serializable (class: Test1$, value: Test1$@ccf91df)
    - element of array (index: 0)
```

Ans: Extend your object with Serializable.

WEEK 13 FAQS

W13:2 John has drawn this picture for Balanced approach design. Is it correct?


Worker Node - 16 Core and 64 GB RAM
Executor 1 - 5 CPU Core & 19.5 GB
CPU Core
1.5 Heap memory
Executor 2 - 5 CPU Core & 19.5 GB
CPU Core
1.5 Heap memory
Executor 3 - 5 CPU Core & 19.5 GB
CPU Core
1.5 Heap memory
Out of 16 Core, 1 is given for background activities
Out of 64GB 1 GB is given for OS.

Ans: Yes, this is a balanced approach, neither thin executors nor fat executors.

W13:3 Is it possible to launch more than 1 executors in the local mode?

Ans: Cloudera has single mode installation hence we can run it in local mode only. ITVersity is a cluster hence you can see multiple executors and cores working, used to optimize the program.

One alternative way is do have your own spark standalone cluster installation where you configure master and slave. Also, this may make your machine slow and run into memory error.

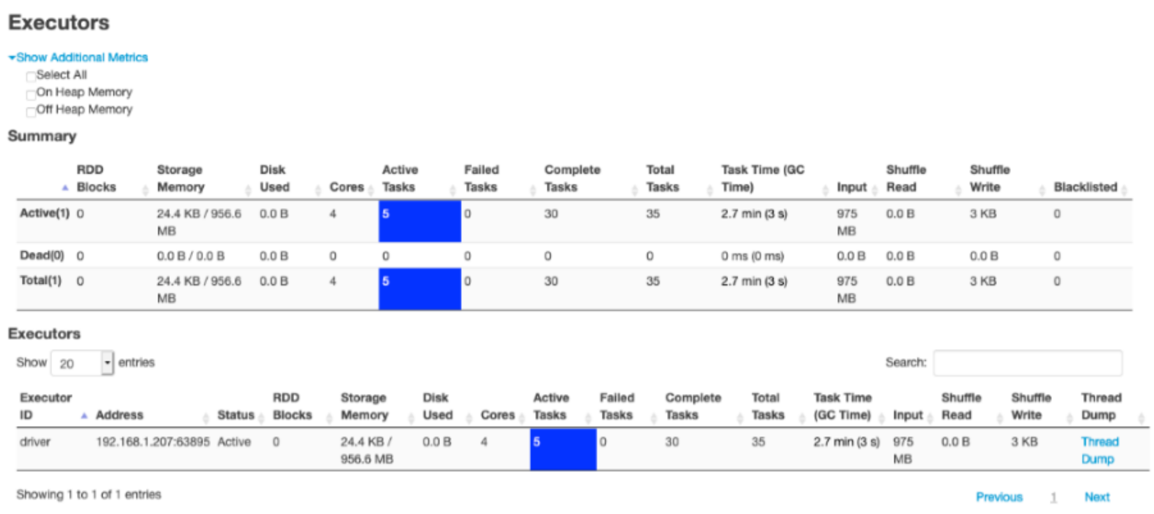
WEEK 13 FAQS

W13:4 Is it possible to allocate the executor cores and memory from Eclipse/ IDE? Specifying the same in the $SPARK_HOME/conf/spark-defaults.conf will set for all applications to use the same configuration. What if I want my application 1 to use 4 cores and 3GB memory and App2 to 2 cores and 2GB memory for each executor without dynamic resource allocation?

Ans: Ee can achieve this using spark submit.

When we run our jar through spark submit, there we can pass our class file name along with the executor arguments like we pass to spark-shell

W13:5 In spark UI, for executors it shows storage memory as 956MB (as shown in the figure 1) Is this actually the spark storage and further divided into execution and storage memory? (as per figure 2)



Executors

▾Show Additional Metrics

□ Select All
□ On Heap Memory
□ Off Heap Memory

Summary

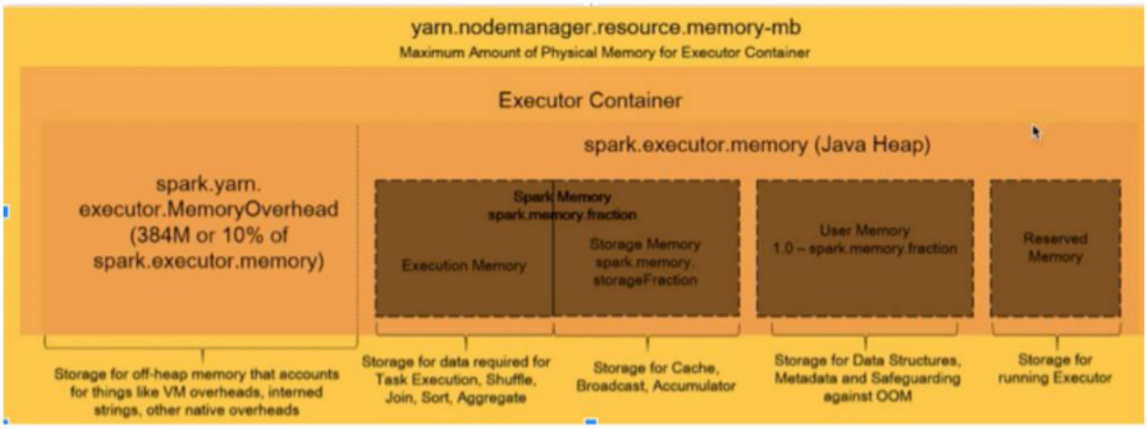
	RDD Blocks	Storage Memory	Disk Used	Cores	Active Tasks	Failed Tasks	Complete Tasks	Total Tasks	Task Time (GC Time)	Input	Shuffle Read	Shuffle Write	Blacklisted
Active(1)	0	24.4 KB / 956.6 MB	0.0 B	4	5	0	30	35	2.7 min (3 s)	975 MB	0.0 B	3 KB	0
Dead(0)	0	0.0 B / 0.0 B	0.0 B	0	0	0	0	0	0 ms (0 ms)	0.0 B	0.0 B	0.0 B	0
Total(1)	0	24.4 KB / 956.6 MB	0.0 B	4	5	0	30	35	2.7 min (3 s)	975 MB	0.0 B	3 KB	0

Executors

Show 20 entries Search:

Executor ID	Address	Status	RDD Blocks	Storage Memory	Disk Used	Cores	Active Tasks	Failed Tasks	Complete Tasks	Total Tasks	Task Time (GC Time)	Input	Shuffle Read	Shuffle Write	Thread Dump
driver	192.168.1.207:63895	Active	0	24.4 KB / 956.6 MB	0.0 B	4	5	0	30	35	2.7 min (3 s)	975 MB	0.0 B	3 KB	Thread Dump

Showing 1 to 1 of 1 entries



yarn.nodemanager.resource.memory-mb
Maximum Amount of Physical Memory for Executor Container
Executor Container
spark.executor.memory (Java Heap)
spark.yarn. executor.MemoryOverhead (384M or 10% of spark.executor.memory)
Spark Memory spark.memory.fraction
Execution Memory
Storage Memory spark.memory. storageFraction
User Memory 1.0 – spark.memory.fraction
Reserved Memory
Storage for off-heap memory that accounts for things like VM overheads, interned strings, other native overheads
Storage for data required for Task Execution, Shuffle, Join, Sort, Aggregate
Storage for Cache, Broadcast, Accumulator
Storage for Data Structures, Metadata and Safeguarding against OOM
Storage for running Executor

Ans: I believe you had given executor memory as 2GB that's why storage memory is showing as 956 MB.

On the DAG it is both storage and execution memory.

Basically, when you assign memory to an executor for instance 2GB in this case this is called heap memory (handled by JVM) and is further subdivided into multiple parts

1. Reserved Memory: used for running the executor, it is 300 MB always
2. Storage and Execution Memory: they both share a common region

Storage and execution memory is 60 % of your remaining memory after taking out reserved memory space

2GB = 2048 MB

2048 - 300 (Reserved Memory) = 1748MB

60 % of this 1748 MB is used for storage and execution

1748 * 0.6 = 1048.8 MB (this is close to what you see on the DAG)

1048.8 = Spark Memory (Storage + Execution)

however, 50 % of this shared memory can be taken up by Storage if needed

1048 * 0.5 = 524.4 MB for storage (Execution can't evict this space if you keep your data under this threshold)

In short, yes, its spark memory which gets divided into storage and execution

W13:6 If John has 1 billion rows of log files with only 2 distinct keys. Is it not better to always use salting and then reduce? or it is necessary only when we use groupBy.

Ans: reduceByKey will try to do local aggregation first before shuffling data however, if you have 2 distinct keys, after aggregation you will end up getting the data in 2 partitions therefore Salting is not dependent on groupBy, whenever your group by column has very low cardinality, majority of the partitions will be empty while remaining have all of the data making large partitions. That's why the number of executors won't help as only 2 cores would be doing the major work while others being idle.

W13:7 In the example where we are using salting, we know the key in advance. But my question is like if suppose we didn't have any clue on the key. sometime the is 100 and sometime the is 1 (here key means ERROR, WARN). how to tackle these kinds of situation.

Ans: salting can only help if you know your data in advance. ideally a data analyst can help you plan such things well in advance.

WEEK 13 FAQS

W13:8 what is the use of offheap memory? even if I persist rdd with storage level as offheap the data is getting stored at disk only? Any use case of using offheap memory?

Ans: Off-heap storage refers to the objects that are not managed by JVM Garbage Collector. It is managed by the operating system but stored outside the JVM process. It is slightly slower than accessing the on-heap storage.
In order to make it work we need to explicitly enable off-heap storage with spark.memory.offHeap.enabled and also specify the amount of off-heap memory in spark.memory.offHeap.size

W13:9 Is it necessary to have a Itversity cloud lab to undergo and implement week 13 contents? Because till now Urvii was working on her local machine. She has not opted for Itversity lab.

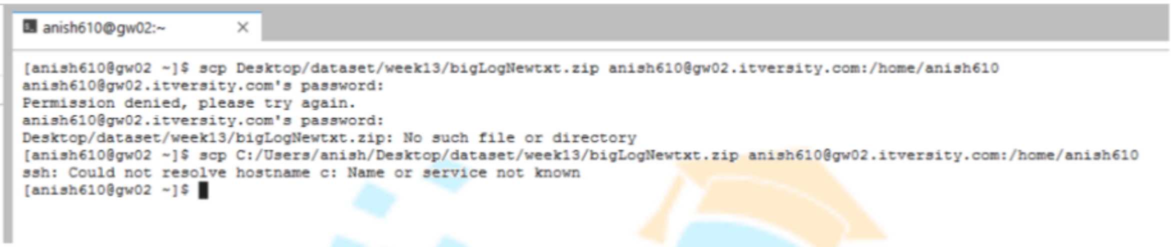
Ans: It is definitely helpful for some concepts like OOM and optimization which need Realtime cluster access.

W13:10 it was discussed that spark has internal optimization where in between different stages, instead of executing all the transformations since the beginning stage, it will actually skip executing the earlier stage and will read the output of that earlier stage from the disk to the current stage(exchange). my doubt is for every stage does spark store the data again on the disk? If so, is this a local disk? But why again on disk?

Ans: spark saves each stage output on the local disk when it encounters a wide transformation because on HDFS/ distributed storage there is a replication factor in place and it doesn't want that. However only the last stage output is actually kept and other stages will be skipped if you call an action again.
It keeps these data in its temp directory and deletes it once spark application stops.
All intermediate data is stored on local disk

WEEK 13 FAQS

W13:11 Chris is unable to transfer BigLogNewtxt.zip from my local to itversity gateway node. I tried these scp commands with respective paths:
scp C:/Users/anish/Desktop/dataset/week13/bigLogNewtxt.zip anish610@gw02.itversity.com:/home/anish610
scp Desktop/dataset/week13/bigLogNewtxt.zip anish610@gw02.itversity.com:/home/anish610
It is showing No such file or directory as output . Am I missing something?



```
[anish610@gw02 ~]$ scp Desktop/dataset/week13/bigLogNewtxt.zip anish610@gw02.itversity.com:/home/anish610
anish610@gw02.itversity.com's password:
Permission denied, please try again.
anish610@gw02.itversity.com's password:
Desktop/dataset/week13/bigLogNewtxt.zip: No such file or directory
[anish610@gw02 ~]$ scp C:/Users/anish/Desktop/dataset/week13/bigLogNewtxt.zip anish610@gw02.itversity.com:/home/anish610
ssh: Could not resolve hostname c: Name or service not known
[anish610@gw02 ~]$ 
```

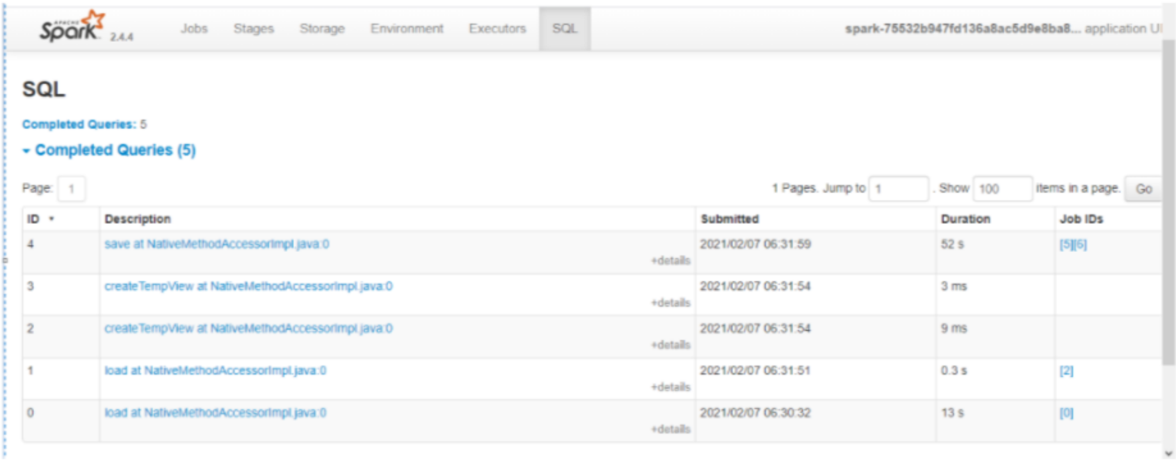
Ans: do ls and cross check your file, if it exists.

Solution 2: use itversity upload option to transfer file from windows to itversity.

W13:12 How to check the type of join in spark UI.

Ans: you can check that in SQL tab whether its Broadcast join or Hash join

W13:13 Yash have this under sql tab, can you please confirm what has to be checked. under this


Spark 2.4.4
Jobs
Stages
Storage
Environment
Executors
SQL
spark-75532b947fd136a8ac5d9e8ba8... application UI
Page: 1
1 Pages. Jump to 1 . Show 100 items in a page. Go

SQL

Completed Queries: 5

Completed Queries (5)

ID	Description	Submitted	Duration	Job IDs
4	save at NativeMethodAccessorImpl.java:0 +details	2021/02/07 06:31:59	52 s	[5][6]
3	createTempView at NativeMethodAccessorImpl.java:0 +details	2021/02/07 06:31:54	3 ms	
2	createTempView at NativeMethodAccessorImpl.java:0 +details	2021/02/07 06:31:54	9 ms	
1	load at NativeMethodAccessorImpl.java:0 +details	2021/02/07 06:31:51	0.3 s	[2]
0	load at NativeMethodAccessorImpl.java:0 +details	2021/02/07 06:30:32	13 s	[0]

Ans: Click on links in the given table

WEEK 13 FAQS

W13:14 John has partitioned the data based on some date column and I have around 10 partitions, In this case, the number of tasks would still be equal to the number of partitions i.e., 10 or would it be equal to the number of blocks (considering the block size of 128 MB)?

Ans: the number of tasks is equal to the number of partitions.
If you don't have partition then it will be equal to number of blocks.

W13:15 There is word count program. Text File size is 1GB. 2 executers are there having memory 100 MB and 200 MB respectively and driver having 100MB. Would the program run.

Ans: The memory mentioned is too less and never used in Big data. Partition size is one block 128 mb and memory is 100 mb, so surely it will fail

W13:16 how to parse date from string format to date time e.g.: "1612432793000"

Ans: I am considering this string as Unix time , from_unixtime method can be used. Below code for reference ,

```
df = (empdf
    .select("date")
    # Convert timestamp to unix timestamp.
    .withColumn("unix_timestamp", unix_timestamp("date", "yyyy-MM-dd hh:mm:ss"))
    # Convert unix timestamp to timestamp.
    .withColumn("date_from_unixtime", from_unixtime("unix_timestamp")))
df.show(2)
```

this is also another way

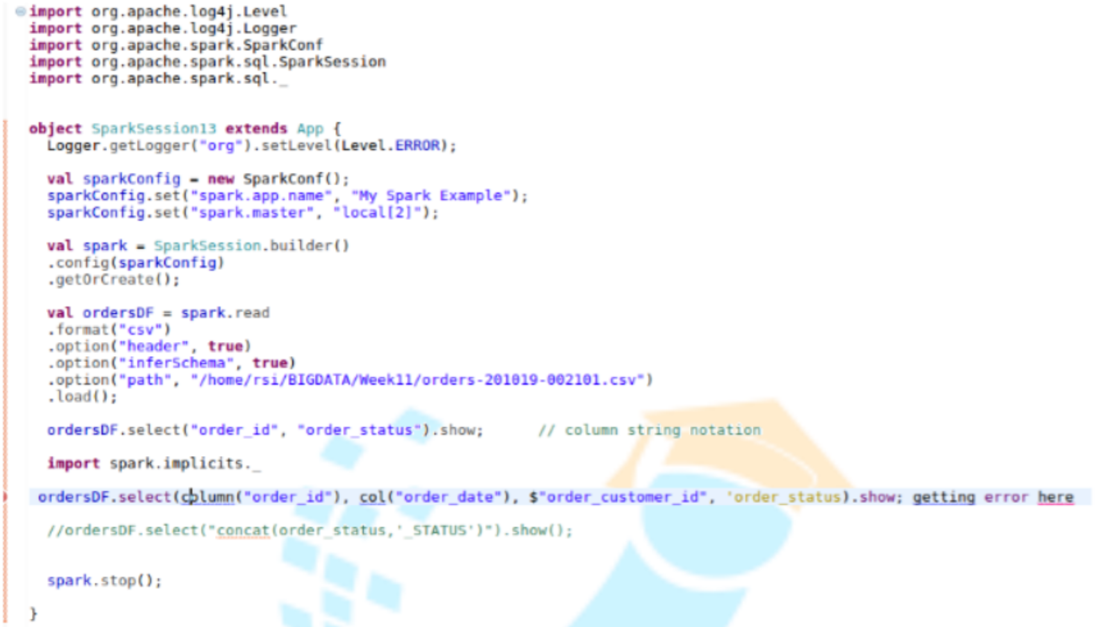
```
df. select (to_date (col ("unix_timestamp_seconds"). cast (TimestampType)). as
("current_date")). show (false)
```

W13:17 What is the difference between spark broadcast join and hive map join.

Ans: It's same

WEEK 13 FAQS

W13:18 Eric is getting error while using column object notation to select column. It says `not found: value column not found: value col`

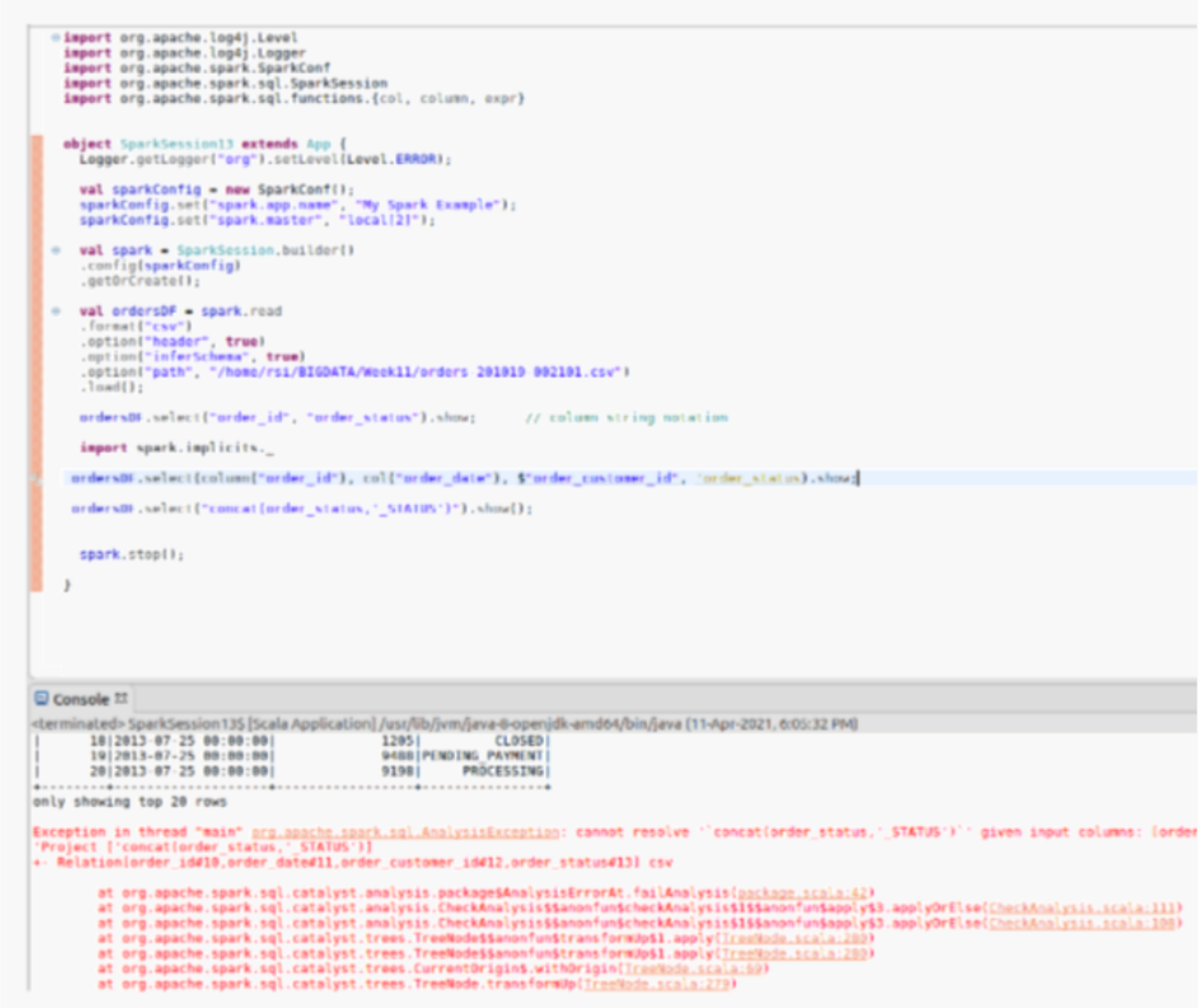


```
import org.apache.log4j.Level
import org.apache.log4j.Logger
import org.apache.spark.SparkConf
import org.apache.spark.sql.SparkSession
import org.apache.spark.sql._

object SparkSession13 extends App {
  Logger.getLogger("org").setLevel(Level.ERROR);

  val sparkConfig = new SparkConf();
  sparkConfig.set("spark.app.name", "My Spark Example");
  sparkConfig.set("spark.master", "local[2]");

  val spark = SparkSession.builder()
  .config(sparkConfig)
  .getOrCreate();

  val ordersDF = spark.read
  .format("csv")
  .option("header", true)
  .option("inferSchema", true)
  .option("path", "/home/rsi/BIGDATA/Week11/orders-201019-002101.csv")
  .load();

  ordersDF.select("order_id", "order_status").show;      // column string notation

  import spark.implicits._

 ordersDF.select(column("order_id"), col("order_date"), $"order_customer_id", 'order_status).show; getting error here

  //ordersDF.select("concat(order_status,'_STATUS')").show();


  spark.stop();

}
```

Ans: it is not org.apache.spark.sql.{col, column}

try with: - `import org.apache.spark.sql.functions. {col, column}`

W13:19 John is getting Exception while trying to use column expression, please refer screenshot. I am not mixing it with any other type (column object, column strings). It's working fine while using with expr () keyword. So, even if we are not mixing it with any other type, Still do we need to use expr keyword?



```
import org.apache.log4j.Level
import org.apache.log4j.Logger
import org.apache.spark.SparkConf
import org.apache.spark.sql.SparkSession
import org.apache.spark.sql.functions.{col, column, expr}


object SparkSession13 extends App {
  Logger.getLogger("org").setLevel(Level.ERROR);

  val sparkConfig = new SparkConf();
  sparkConfig.set("spark.app.name", "My Spark Example");
  sparkConfig.set("spark.master", "local[2]");

  val spark = SparkSession.builder()
  .config(sparkConfig)
  .getOrCreate();

  val ordersDF = spark.read
  .format("csv")
  .option("header", true)
  .option("inferSchema", true)
  .option("path", "/home/rsi/BIGDATA/Week11/orders-201019-002101.csv")
  .load();

  ordersDF.select("order_id", "order_status").show;      // column string notation

  import spark.implicits._

 ordersDF.select(column("order_id"), col("order_date"), $"order_customer_id", 'order_status).show;

 ordersDF.select("concat(order_status,'_STATUS')").show();


  spark.stop();

}
```

Console

```
<terminated> SparkSession13$ [Scala Application] /usr/lib/jvm/java-8-openjdk-amd64/bin/java (11-Apr-2021, 6:05:32 PM)
|      18|2013-07-25 00:00:00|             1205|         CLOSED|
|      19|2013-07-25 00:00:00|             9488|PENDING_PAYMENT|
|      20|2013-07-25 00:00:00|             9198|     PROCESSING|
+--------+-------------------+-----------------+---------------+
only showing top 20 rows

Exception in thread "main" org.apache.spark.sql.AnalysisException: cannot resolve '`concat(order_status,'_STATUS')`' given input columns: [order
'Project ['concat(order_status,'_STATUS')]
+- Relation[order_id#10,order_date#11,order_customer_id#12,order_status#13] csv

        at org.apache.spark.sql.catalyst.analysis.package$AnalysisErrorAt.failAnalysis(package.scala:42)
        at org.apache.spark.sql.catalyst.analysis.CheckAnalysis$$anonfun$checkAnalysis$1$$anonfun$apply$3.applyOrElse(CheckAnalysis.scala:111)
        at org.apache.spark.sql.catalyst.analysis.CheckAnalysis$$anonfun$checkAnalysis$1$$anonfun$apply$3.applyOrElse(CheckAnalysis.scala:108)
        at org.apache.spark.sql.catalyst.trees.TreeNode$$anonfun$transformUp$1.apply(TreeNode.scala:280)
        at org.apache.spark.sql.catalyst.trees.TreeNode$$anonfun$transformUp$1.apply(TreeNode.scala:280)
        at org.apache.spark.sql.catalyst.trees.CurrentOrigin$.withOrigin(TreeNode.scala:69)
        at org.apache.spark.sql.catalyst.trees.TreeNode.transformUp(TreeNode.scala:279)
```

Ans: "Analysis Exception" means check "Column Name"   - there is no "order_customer_id" Column in our Dataset, it's "customer_id"

W13:20 Nancy is having doubt related to storage and execution memory. Suppose, there is 50percent threshold for storage memory in the executor which is already pre-defined. Now, the executions are going on and it took almost 80prcnt of the total available (storage and execution memory). Now, if there is a requirement to cache anything, then what will happen? As we know that, storage cannot evict execution, so will the storage have to wait for execution to complete? or as the storage threshold is

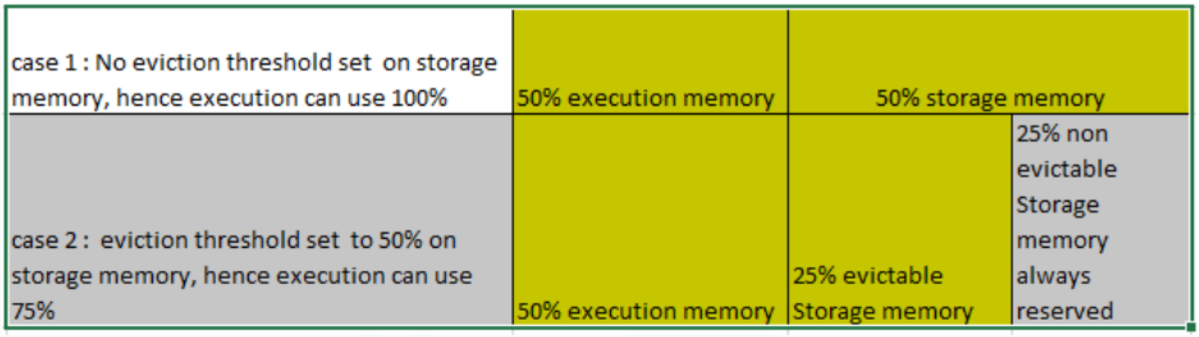
WEEK 13 FAQS

50percnt, it can easily occupy up to that 50prcnt of the memory space

Ans: case 1: if your application never does cache then you can use full storage memory for execution

Case 2: if your application does cache sometimes then you can set the threshold (beyond which the execution cannot evict the storage and hence you will always find storage for cache): problem you discussed will never occur. In your case you did not mention the evict storage memory threshold (if I understood correctly). so, I am considering case 1 and suddenly the cache requirement arises then in that case it will wait for execution memory to free up.

It cannot use 80% +50% memory of (storage and execution memory)



case 1 : No eviction threshold set on storage memory, hence execution can use 100%	50% execution memory	50% storage memory	
case 2 : eviction threshold set to 50% on storage memory, hence execution can use 75%	50% execution memory	25% evictable Storage memory	25% non evictable Storage memory always reserved