PySpark Certification T...

Playing with Spark RDDs > Presentation

Course Content

- Presentation
- Case Study I - Perfor...
- Case Study II - Load ...
- In-Class Demo - Inpu...
- Dataset
- Dataset
- RDD Practicals
- Apache Sqoop
- Word_Count_Class

Personal Library

Presentation 3 / 100

COURSE OUTLINE

MODULE 05

Introduction to Big Data Hadoop and Spark

Introduction to Python for Apache Spark

Functions, OOPs, Modules in Python

Deep Dive into Apache Spark Framework

Playing with Spark RDDs

Data Frames and Spark SQL

Machine Learning using Spark MLlib

Deep Dive into Spark MLlib

Understanding Apache Kafka and Apache Flume

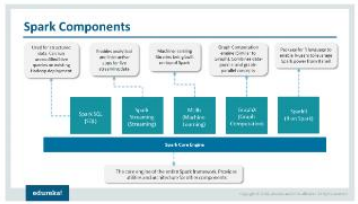
Apache Spark Streaming - Processing Multiple Batches

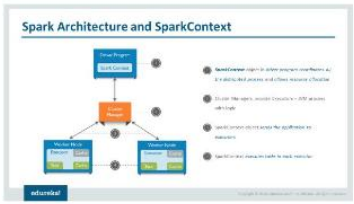
Apache Spark Streaming - Data Sources

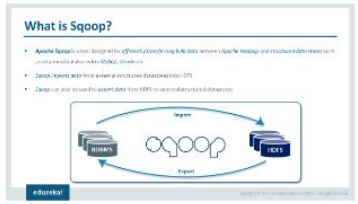
Implementing an End-to-End Project

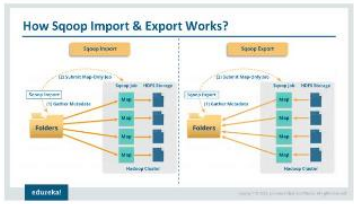
Presentation 4 / 100

Recap







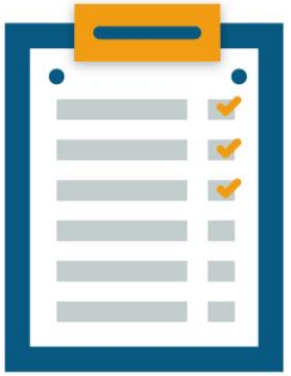


edureka!

Presentation 5 / 100

Topics

- Challenges in the Existing Computing Methods
- RDD
  - RDD Overview
  - RDD Workflow
  - RDD transformations and actions
  - Shuffling
  - Data Loading and Saving through RDD
  - Key-Value pair in RDD
- RDD Functions
- RDD Persistance
- RDD Lineage



edureka!

Presentation 6 / 100

Objectives

After completing this module, you should be able to:

- Analyze the Challenges in Existing Computing Methods
- Describe Probable Solution and How RDD Solves the Problem
- Describe What is RDD, It's Functions, Transformations and Actions
- Describe Data Loading and Saving Through RDDs
- Describe Key-Value Pair RDDs, Other Pair RDDs and Two-pair RDDs
- Describe RDD Lineage and RDD Persistence
- Explain a Spark Program Using RDD Concepts
- Describe RDD Partitioning and How It Helps Achieve Parallelization

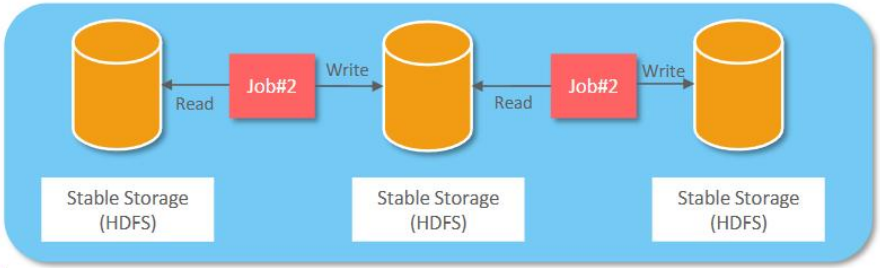
edureka!

Challenges in Existing Computing Methods

When it comes to iterative distributed computing, i.e. processing data over multiple jobs in computations, we need to reuse or share data among multiple jobs.

There are problems with the current computing method (***MapReduce***).

- We need to store data in some intermediate stable distributed storage such as HDFS
- Multiple I/O operations makes the overall computations of ***jobs slower***
- Replications and serializations which in turn makes the ***process slower***

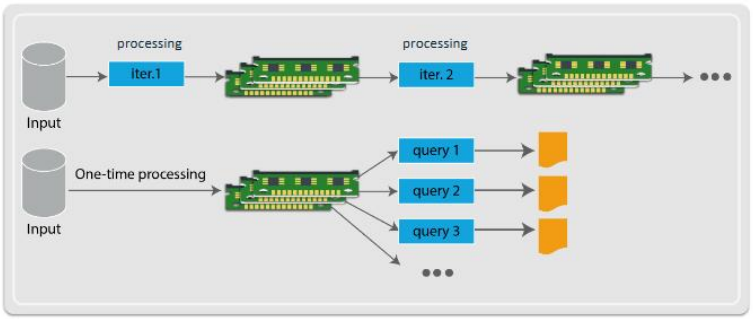


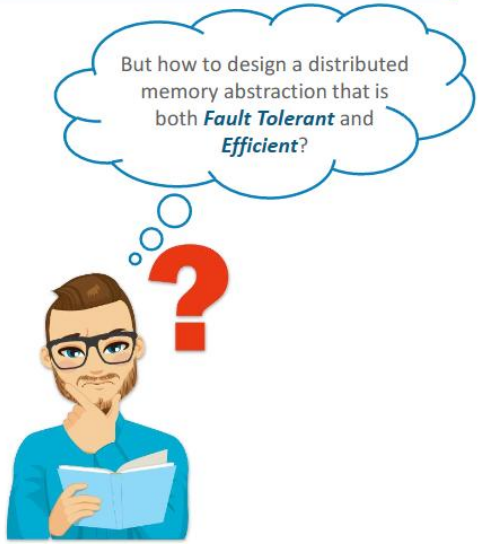
edureka!

Presentation 8 / 100

Probable Solution

- Our Goal here is to reduce the number of I/O operations through HDFS
- This can only be achieved through "***In-Memory Data Sharing***"
- The ***In-Memory Data Sharing*** is 10-100x faster than ***Network/Disk*** sharing



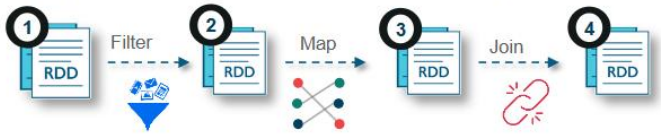


edureka!

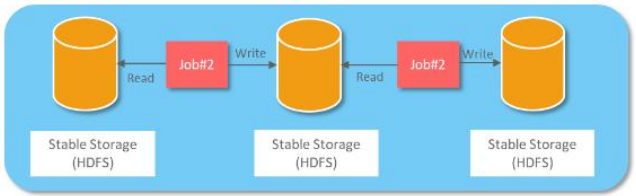
RDDs : The Perfect Solution

- RDDs try to solve all the problems by enabling ***fault tolerant distributed In-memory computations***
- Since RDDs are created over a set of transformations, it logs those transformations, rather than actual data

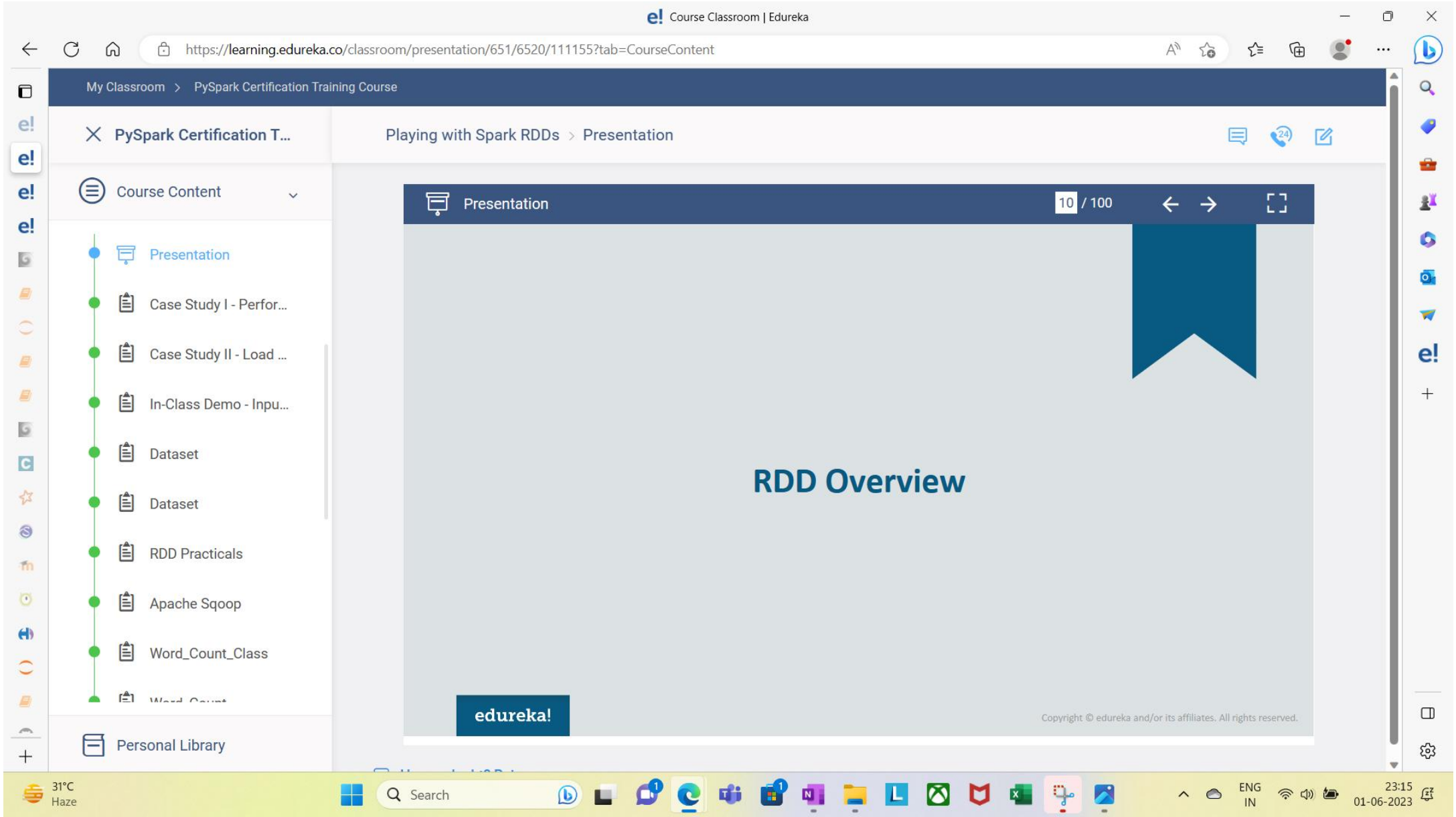
For example:



- ***In case if we lose some partition of RDD, we can replay the transformation on that partition in LINEAGE to achieve the same computation, rather than doing data replication across multiple nodes***
- This characteristic is biggest benefit of RDD, because it *saves* a lot of *efforts in data management* and *replication* and thus achieves *faster computations*

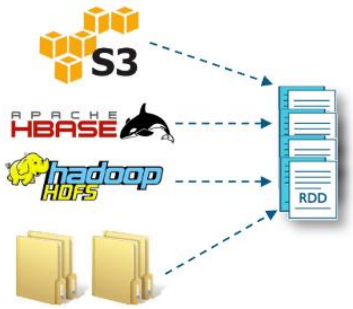


edureka!


Course Classroom | Edureka
https://learning.edureka.co/classroom/presentation/651/6520/111155?tab=CourseContent
My Classroom > PySpark Certification Training Course
PySpark Certification T...
Playing with Spark RDDs > Presentation
Course Content
Presentation
Case Study I - Perfor...
Case Study II - Load ...
In-Class Demo - Inpu...
Dataset
Dataset
RDD Practicals
Apache Sqoop
Word_Count_Class
Personal Library
Presentation
10 / 100
RDD Overview
edureka!
Copyright © edureka and/or its affiliates. All rights reserved.
31°C
Haze
Search
ENG
IN
23:15
01-06-2023

What is RDD?

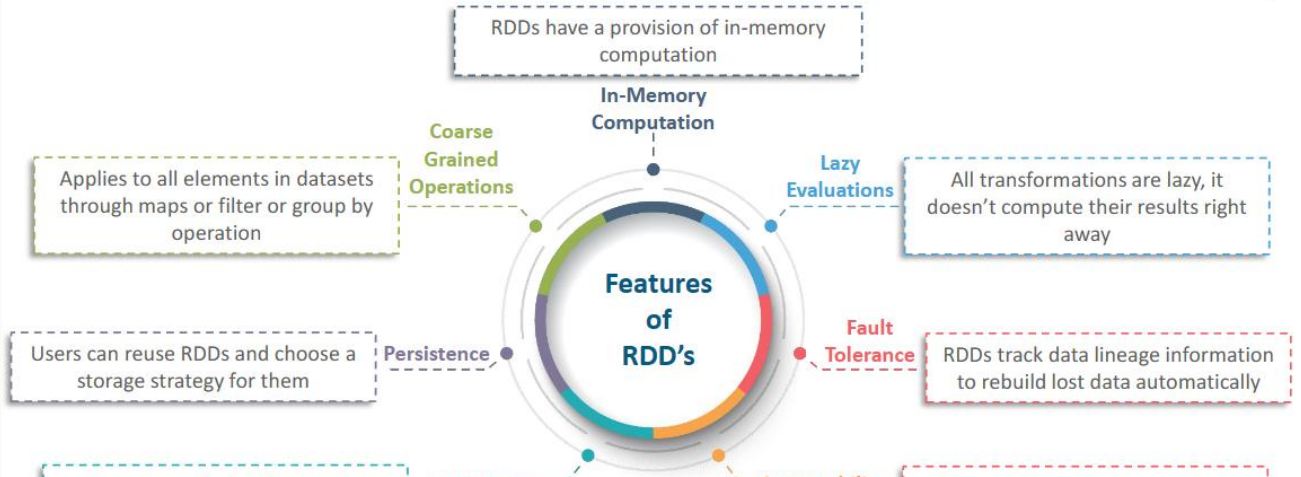
- *RDD = Resilient Distributed Datasets*
- RDD is a *distributed memory abstraction* which lets programmers perform *in-memory computations* on large clusters in a *fault-tolerant manner*
- They are *read-only collection* of objects *partitioned across* a set of machines that *can be rebuilt* if a partition is lost
- RDDs can be *created from multiple data sources* e.g. Python collection, local file system, Hadoop, Amazon S3, HBase table etc.
- There are several operations performed on RDDs:
  - ***Functions***
  - ***Transformations***
  - ***Actions***



edureka!

Presentation 12 / 100

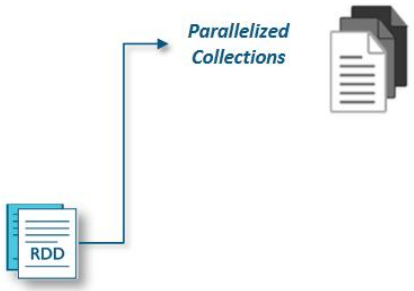
Features of RDD's



edureka!

Presentation 13 / 100

Ways to Create RDD

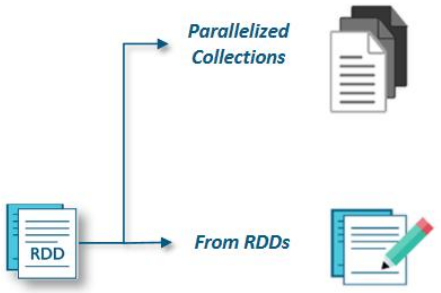



```
data="Hello, Welcome to Edureka!".split(" ")
modified_data=sc.parallelize(data, 2)
modified_data.collect()
```

edureka!

Presentation 14 / 100

Ways to Create RDD




```
data="Hello, Welcome to Edureka!".split(" ")
modified_data=sc.parallelize(data, 2)
modified_data.collect()
```

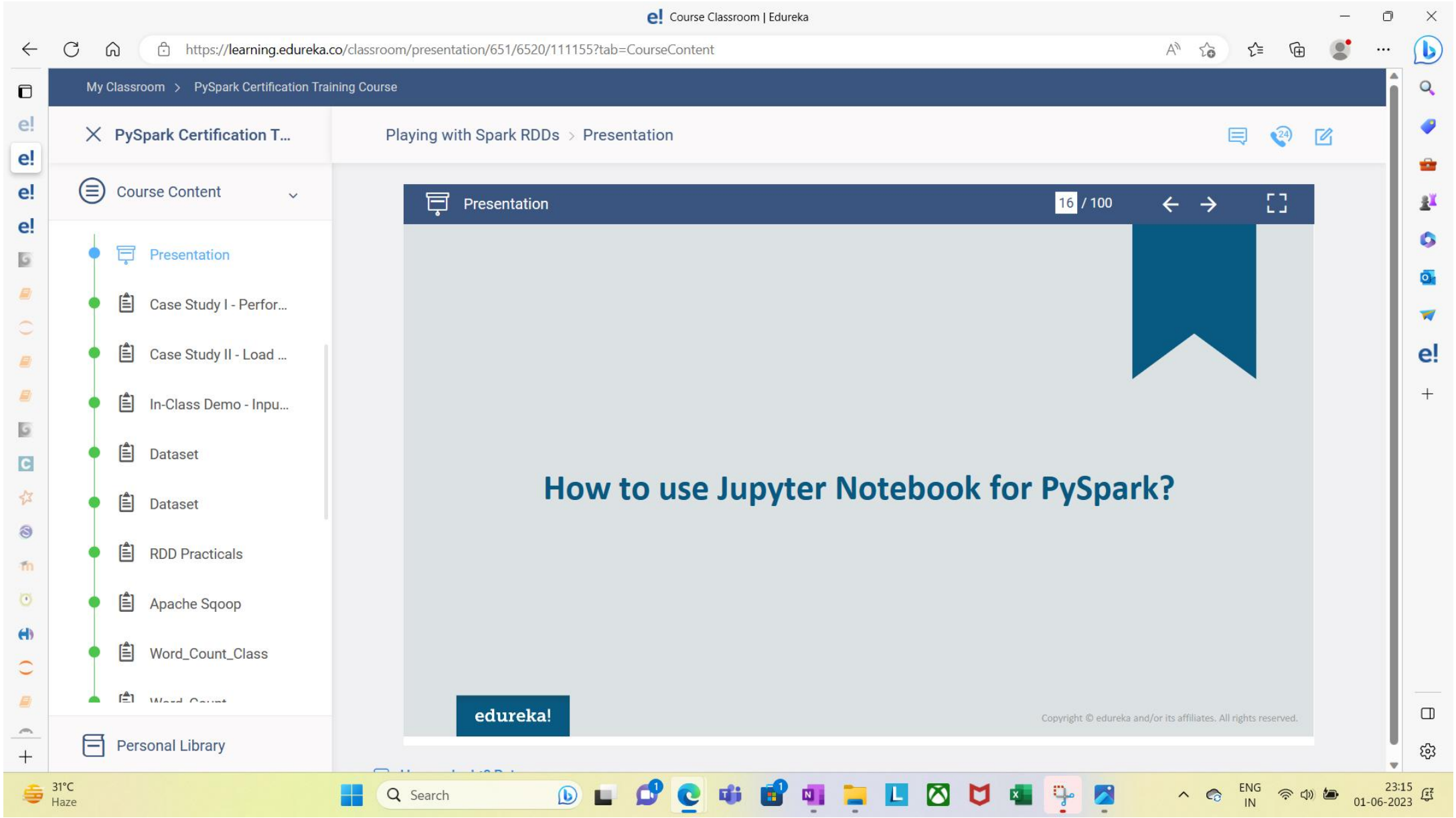
```
r1=sc.parallelize("Happy Learning".split(" "))
r2=r1.map(lambda line: line.upper())
r2.collect()
```

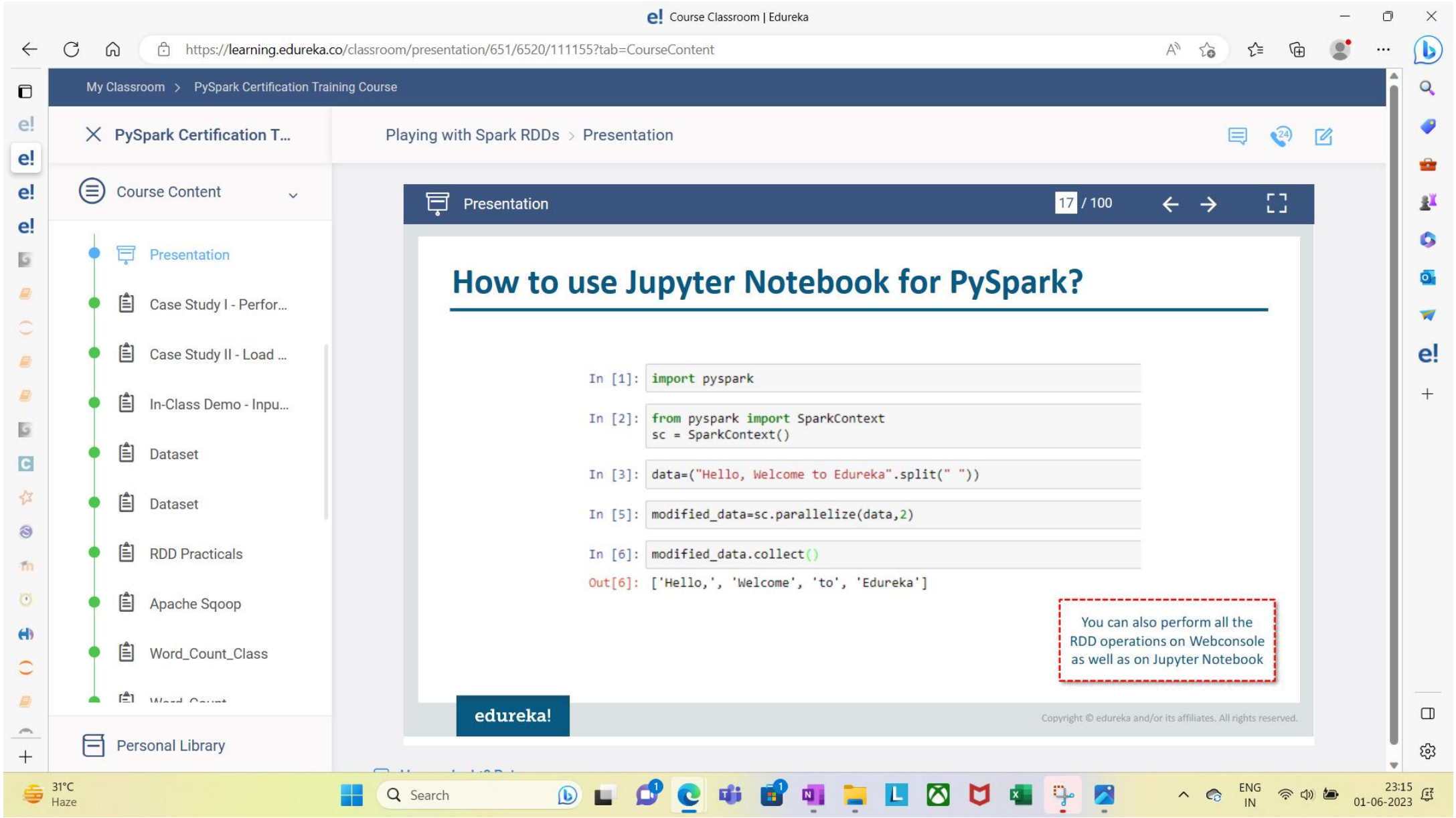
edureka!

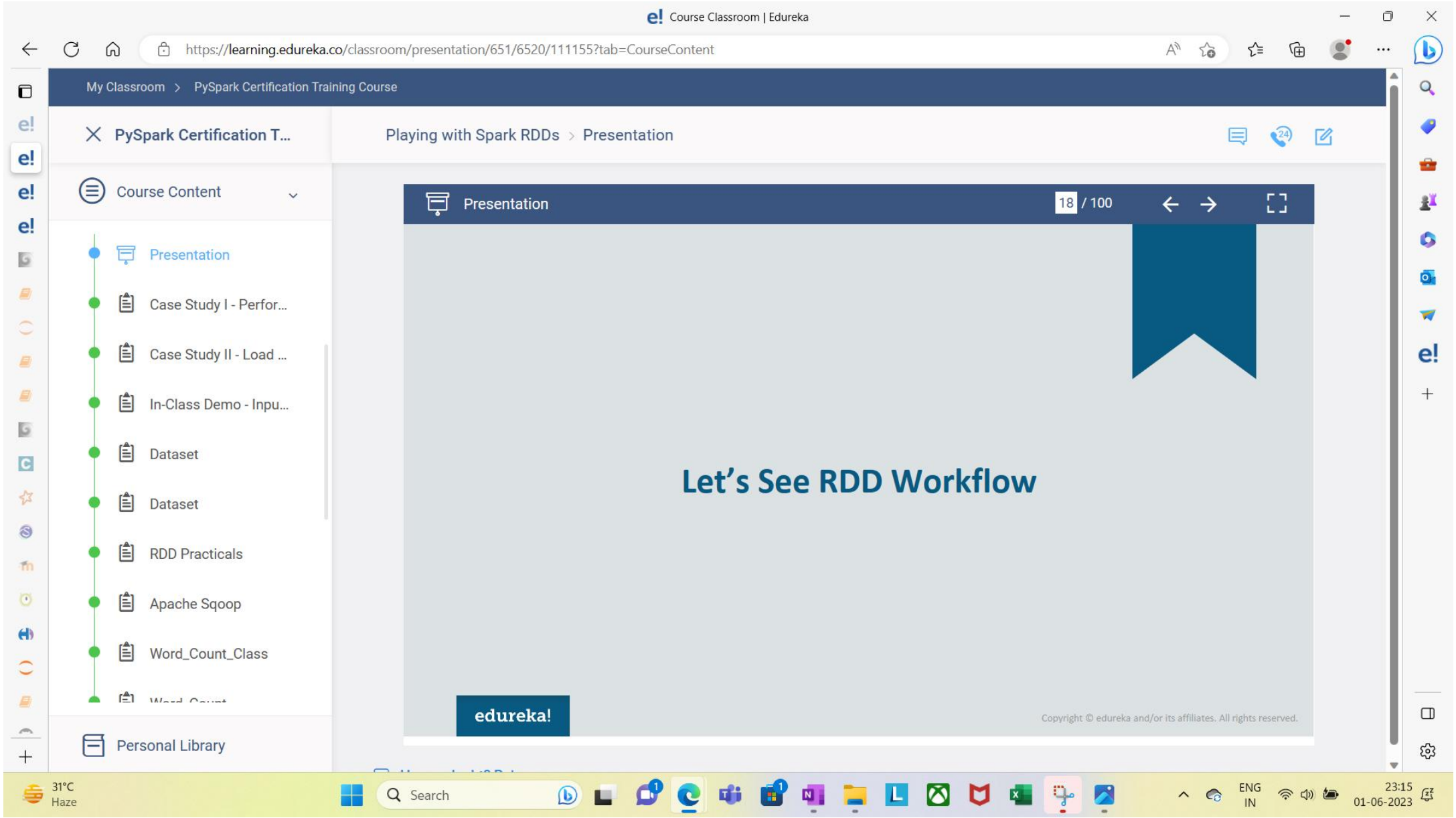
Presentation 15 / 100

Ways to Create RDD

edureka!

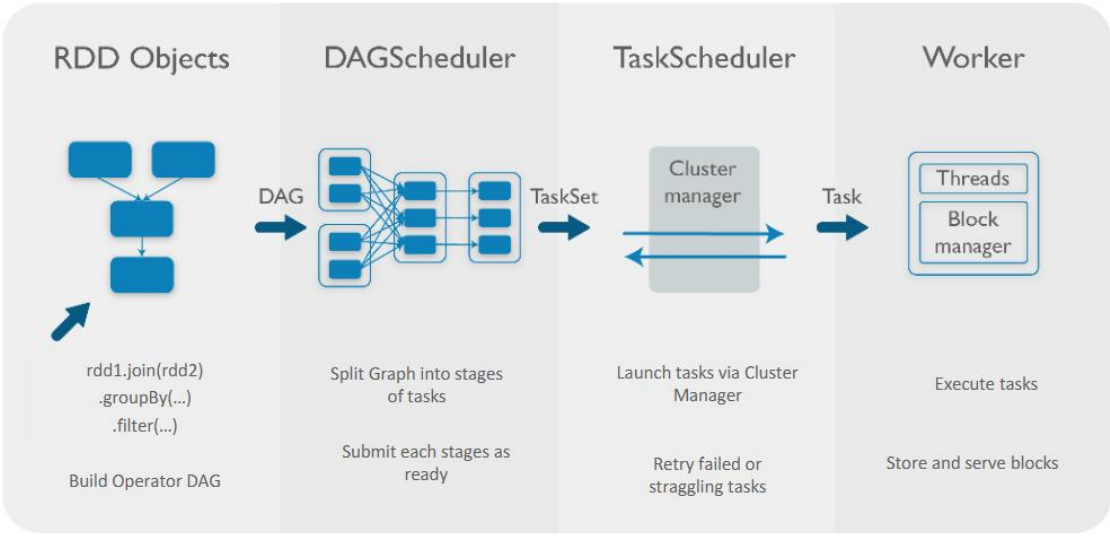

Course Classroom | Edureka
https://learning.edureka.co/classroom/presentation/651/6520/111155?tab=CourseContent
My Classroom > PySpark Certification Training Course
PySpark Certification T...
Playing with Spark RDDs > Presentation
Course Content
Presentation
Case Study I - Perfor...
Case Study II - Load ...
In-Class Demo - Inpu...
Dataset
Dataset
RDD Practicals
Apache Sqoop
Word_Count_Class
Personal Library
Presentation
16 / 100
How to use Jupyter Notebook for PySpark?
edureka!
Copyright © edureka and/or its affiliates. All rights reserved.
31°C
Haze
Search
ENG
IN
23:15
01-06-2023


Course Classroom | Edureka
https://learning.edureka.co/classroom/presentation/651/6520/111155?tab=CourseContent
My Classroom > PySpark Certification Training Course
PySpark Certification T...
Playing with Spark RDDs > Presentation
Course Content
Presentation
Case Study I - Perfor...
Case Study II - Load ...
In-Class Demo - Inpu...
Dataset
Dataset
RDD Practicals
Apache Sqoop
Word_Count_Class
Personal Library
Presentation
17 / 100
How to use Jupyter Notebook for PySpark?
In [1]: import pyspark
In [2]: from pyspark import SparkContext
sc = SparkContext()
In [3]: data=("Hello, Welcome to Edureka".split(" "))
In [5]: modified_data=sc.parallelize(data,2)
In [6]: modified_data.collect()
Out[6]: ['Hello,', 'Welcome', 'to', 'Edureka']
You can also perform all the RDD operations on Webconsole as well as on Jupyter Notebook
edureka!
Copyright © edureka and/or its affiliates. All rights reserved.
31°C Haze
Search
ENG IN
23:15 01-06-2023

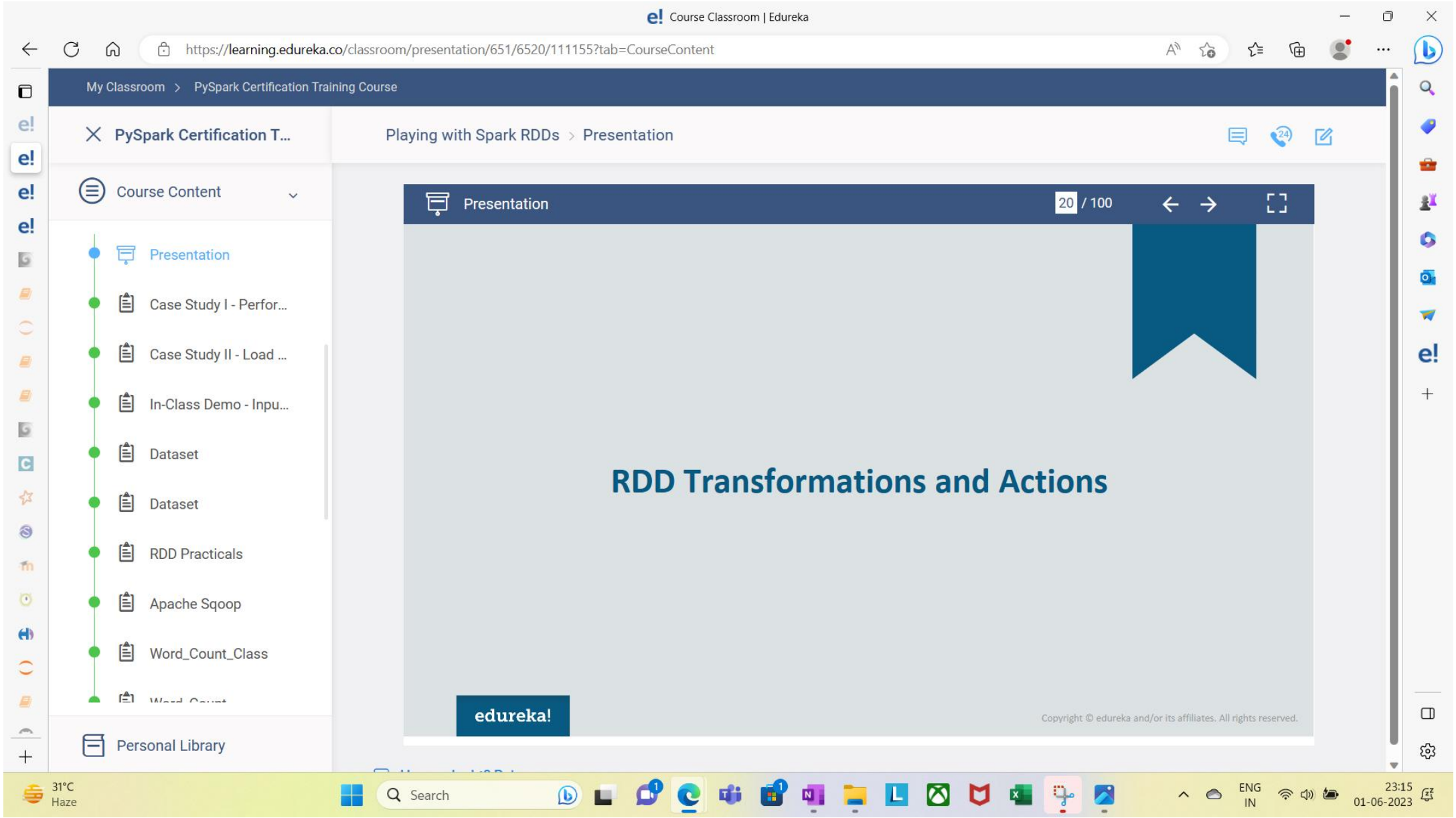

Course Classroom | Edureka
https://learning.edureka.co/classroom/presentation/651/6520/111155?tab=CourseContent
My Classroom > PySpark Certification Training Course
PySpark Certification T...
Playing with Spark RDDs > Presentation
Course Content
Presentation
Case Study I - Perfor...
Case Study II - Load ...
In-Class Demo - Inpu...
Dataset
Dataset
RDD Practicals
Apache Sqoop
Word_Count_Class
Personal Library
Presentation
18 / 100
Let's See RDD Workflow
edureka!
Copyright © edureka and/or its affiliates. All rights reserved.
31°C
Haze
Search
ENG
IN
23:15
01-06-2023

Presentation 19 / 100

RDD Workflow

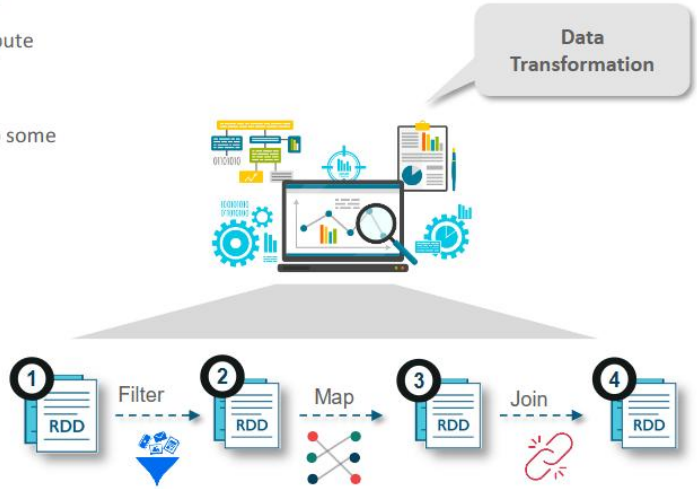


edureka!


Course Classroom | Edureka
https://learning.edureka.co/classroom/presentation/651/6520/111155?tab=CourseContent
My Classroom > PySpark Certification Training Course
PySpark Certification T...
Playing with Spark RDDs > Presentation
Course Content
Presentation
Case Study I - Perfor...
Case Study II - Load ...
In-Class Demo - Inpu...
Dataset
Dataset
RDD Practicals
Apache Sqoop
Word_Count_Class
Personal Library
Presentation
20 / 100
RDD Transformations and Actions
edureka!
Copyright © edureka and/or its affiliates. All rights reserved.
31°C
Haze
Search
ENG
IN
23:15
01-06-2023

Transformations

- Transformations *create* a *new dataset from* an *existing one*
- All *transformations* in Spark *are lazy*: they do not compute their results right away
- Instead they remember the transformations applied to some base dataset
- This helps in:
  - *Optimizing* the *required calculations*
  - *Recovery* of *lost data partitions*



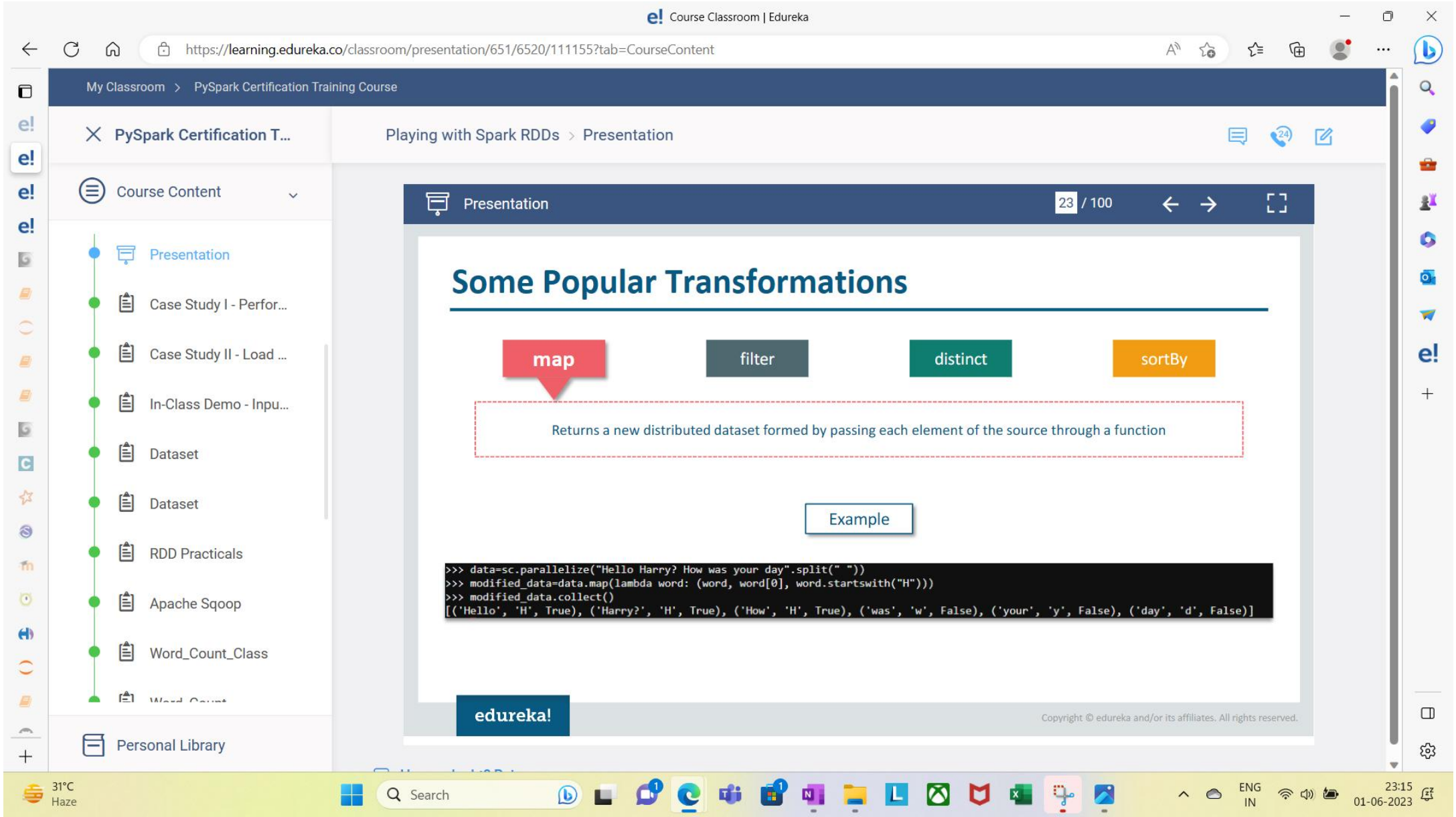
edureka!

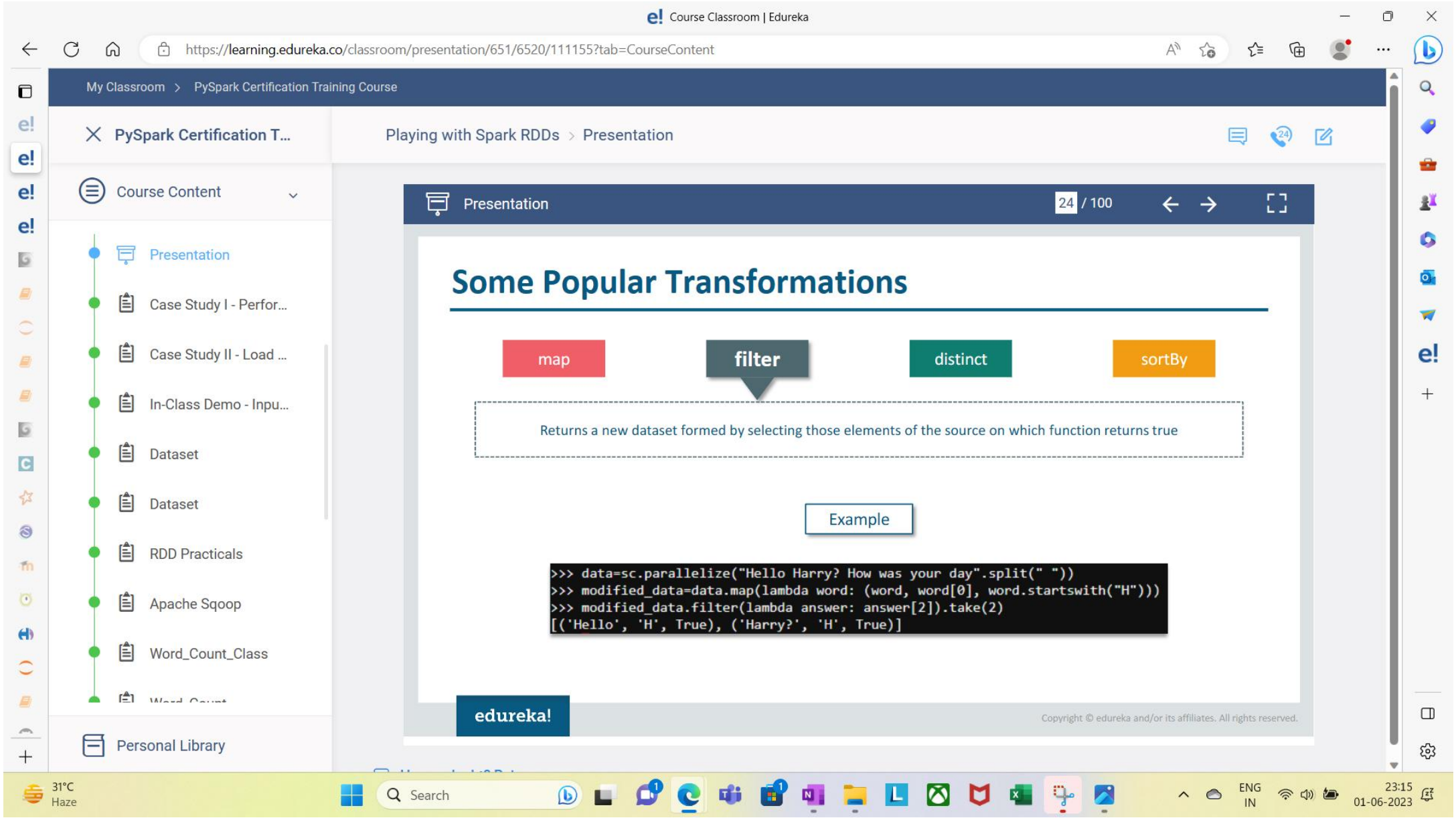
Presentation 22 / 100

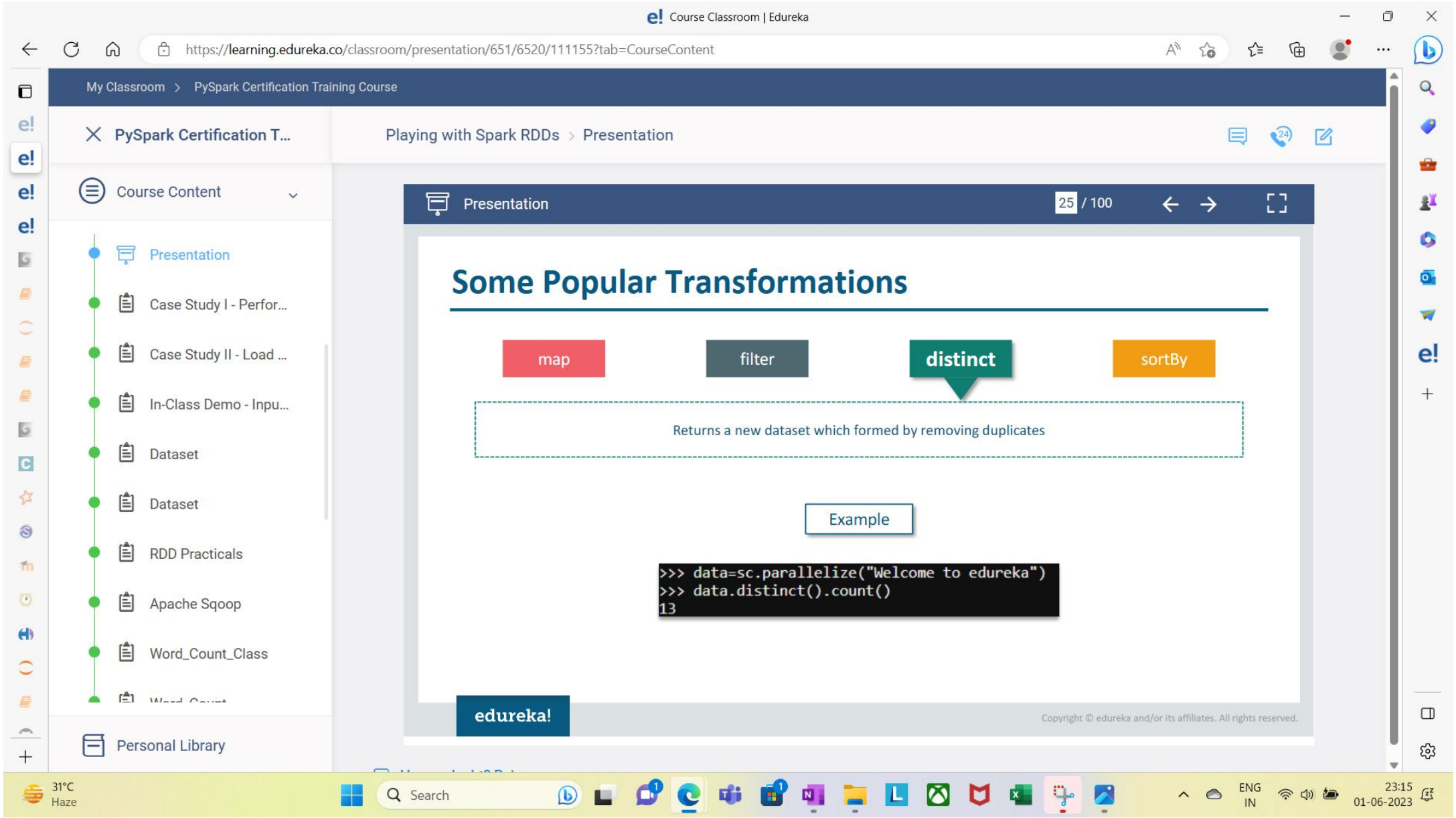
Transformations (Contd.)

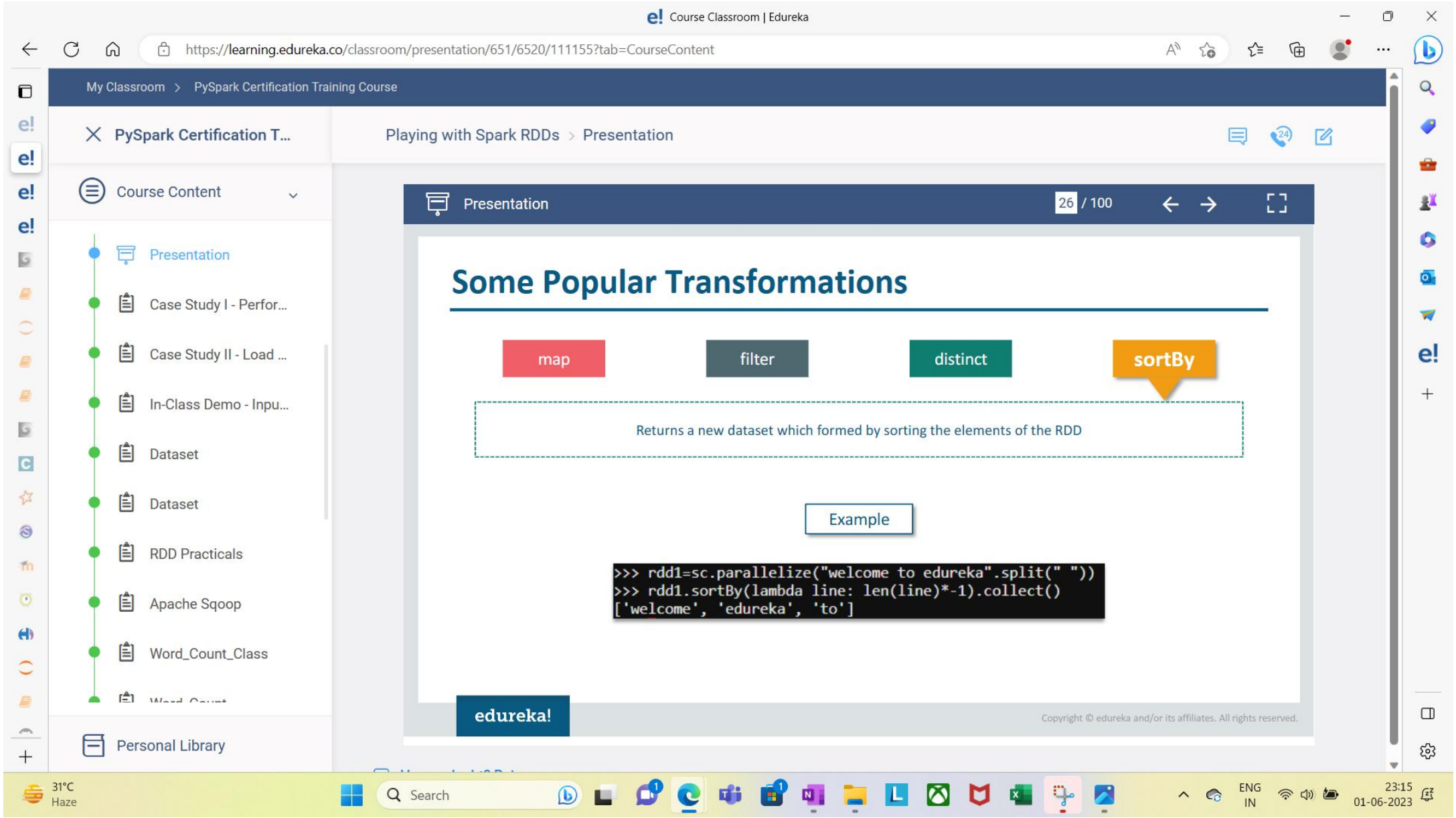
Transformation	Meaning
map(func)	Return a new distributed dataset formed by passing each element of the source through a function func
filter(func)	Return a new dataset formed by selecting those elements of the source on which func returns true
distinct(func)	Return a RDD containing only distinct elements
sortBy ([*ascending*], [*numTasks*])	Returns a dataset of (K, V) pairs sorted by keys in ascending or descending order, as specified in the boolean ascending argument
intersection(otherDataset)	Return a new RDD that contains the intersection of elements in the source dataset and the argument
flatMap()	Similar to map, but allows emitting more than one item in the map function
union(Dataset)	Return a new RDD that contains the intersection of elements in the source dataset and the argument
groupBy()	When called on a dataset of (K, V) pairs, returns a dataset of (K, Iterable<V>) pairs

edureka!


Course Classroom | Edureka
https://learning.edureka.co/classroom/presentation/651/6520/111155?tab=CourseContent
My Classroom > PySpark Certification Training Course
PySpark Certification T...
Playing with Spark RDDs > Presentation
Course Content
Presentation
Case Study I - Perfor...
Case Study II - Load ...
In-Class Demo - Inpu...
Dataset
Dataset
RDD Practicals
Apache Sqoop
Word_Count_Class
Personal Library
Presentation
23 / 100
Some Popular Transformations
map
filter
distinct
sortBy
Returns a new distributed dataset formed by passing each element of the source through a function
Example
>>> data=sc.parallelize("Hello Harry? How was your day".split(" "))
>>> modified_data=data.map(lambda word: (word, word[0], word.startswith("H")))
>>> modified_data.collect()
[('Hello', 'H', True), ('Harry?', 'H', True), ('How', 'H', True), ('was', 'w', False), ('your', 'y', False), ('day', 'd', False)]
edureka!
Copyright © edureka and/or its affiliates. All rights reserved.
31°C
Haze
Search
ENG
IN
23:15
01-06-2023


Course Classroom | Edureka
https://learning.edureka.co/classroom/presentation/651/6520/111155?tab=CourseContent
My Classroom > PySpark Certification Training Course
PySpark Certification T...
Playing with Spark RDDs > Presentation
Course Content
Presentation
Case Study I - Perfor...
Case Study II - Load ...
In-Class Demo - Inpu...
Dataset
Dataset
RDD Practicals
Apache Sqoop
Word_Count_Class
Personal Library
Presentation
24 / 100
Some Popular Transformations
map
filter
distinct
sortBy
Returns a new dataset formed by selecting those elements of the source on which function returns true
Example
>>> data=sc.parallelize("Hello Harry? How was your day".split(" "))
>>> modified_data=data.map(lambda word: (word, word[0], word.startswith("H")))
>>> modified_data.filter(lambda answer: answer[2]).take(2)
[('Hello', 'H', True), ('Harry?', 'H', True)]
edureka!
Copyright © edureka and/or its affiliates. All rights reserved.
31°C
Haze
Search
ENG
IN
23:15
01-06-2023


Course Classroom | Edureka
https://learning.edureka.co/classroom/presentation/651/6520/111155?tab=CourseContent
My Classroom > PySpark Certification Training Course
PySpark Certification T...
Playing with Spark RDDs > Presentation
Course Content
Presentation
Case Study I - Perfor...
Case Study II - Load ...
In-Class Demo - Inpu...
Dataset
Dataset
RDD Practicals
Apache Sqoop
Word_Count_Class
Personal Library
Presentation
25 / 100
Some Popular Transformations
map
filter
distinct
sortBy
Returns a new dataset which formed by removing duplicates
Example
>>> data=sc.parallelize("Welcome to edureka")
>>> data.distinct().count()
13
edureka!
Copyright © edureka and/or its affiliates. All rights reserved.
31°C
Haze
Search
ENG
IN
23:15
01-06-2023


e! Course Classroom | Edureka
https://learning.edureka.co/classroom/presentation/651/6520/111155?tab=CourseContent
My Classroom > PySpark Certification Training Course
PySpark Certification T...
Playing with Spark RDDs > Presentation
Course Content
Presentation
Case Study I - Perfor...
Case Study II - Load ...
In-Class Demo - Inpu...
Dataset
Dataset
RDD Practicals
Apache Sqoop
Word_Count_Class
Personal Library
Presentation
26 / 100
Some Popular Transformations
map
filter
distinct
sortBy
Returns a new dataset which formed by sorting the elements of the RDD
Example
>>> rdd1=sc.parallelize("welcome to edureka".split(" "))
>>> rdd1.sortBy(lambda line: len(line)*-1).collect()
['welcome', 'edureka', 'to']
edureka!
Copyright © edureka and/or its affiliates. All rights reserved.
31°C
Haze
Search
ENG
IN
23:15
01-06-2023

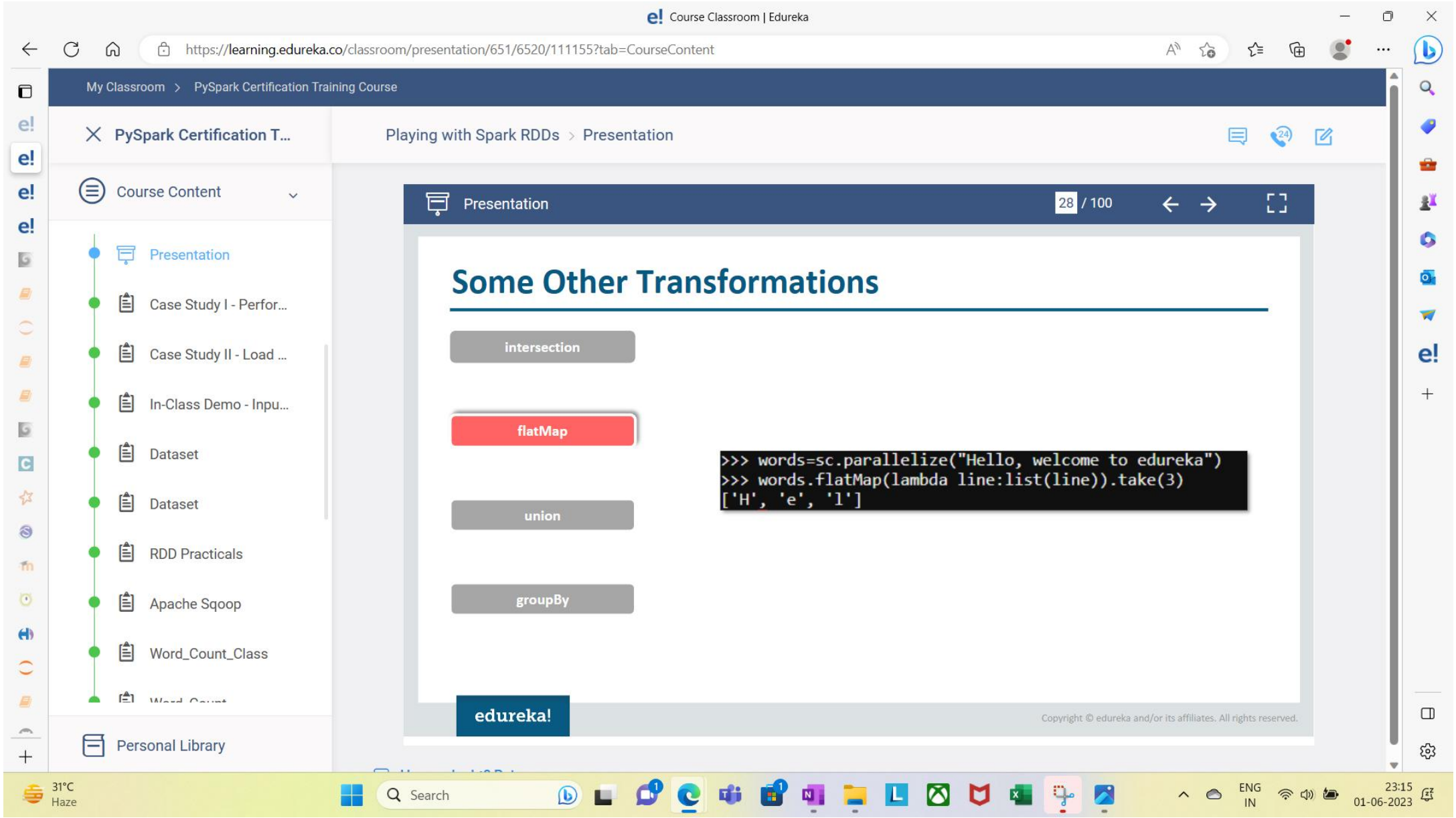
Presentation 27 / 100

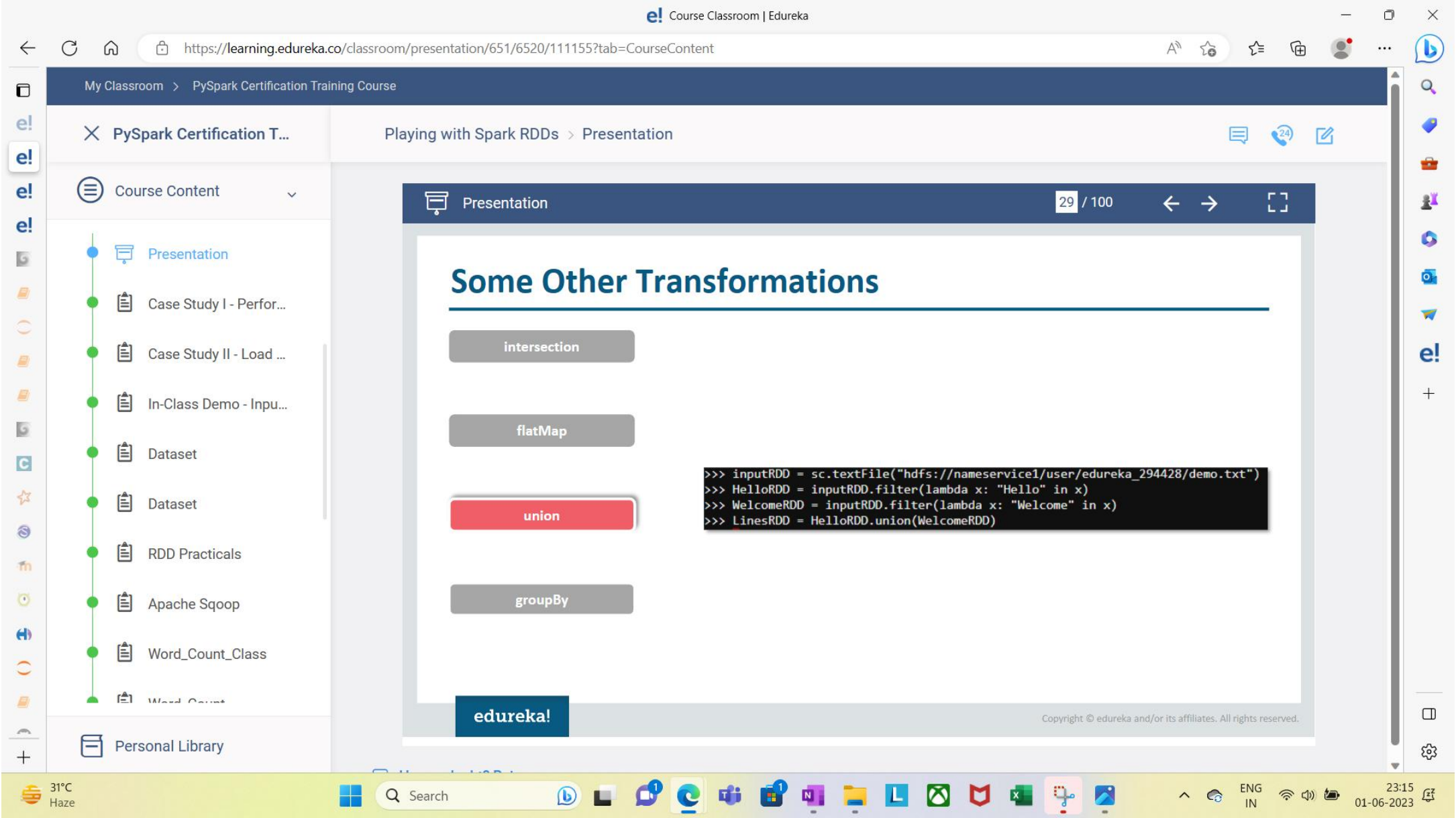
Some Other Transformations

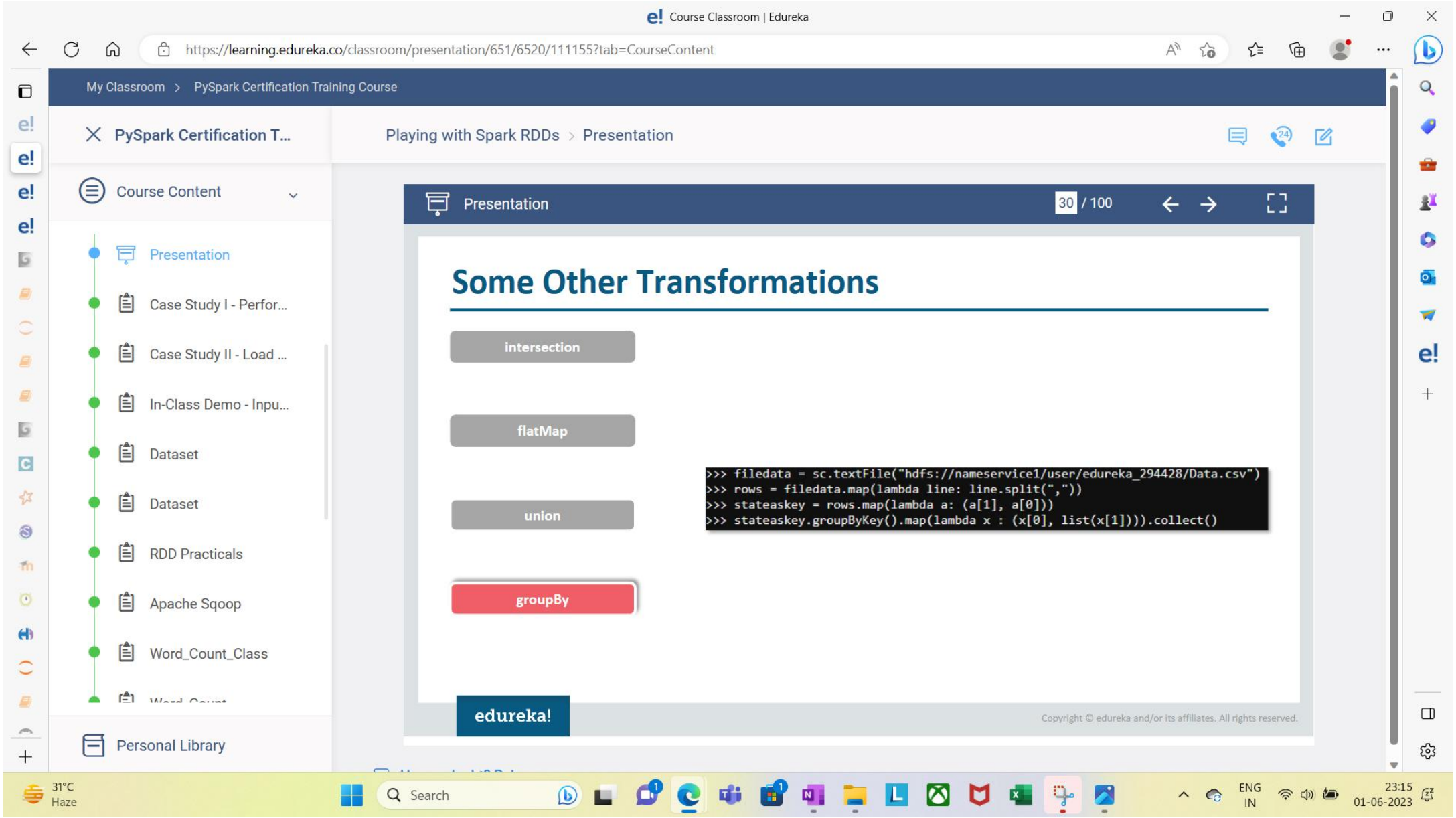
- intersection
- flatMap
- union
- groupBy

```
>>> rdd1 = sc.parallelize(((1,"jan",2016),(3,"nov",2014), (16,"feb",2014)))
>>> rdd2 = sc.parallelize(((5,"dec",2014),(1,"jan",2016)))
>>> comman = rdd1.intersection(rdd2)
>>> comman.collect()
[(1, 'jan', 2016)]
```

edureka!


e! Course Classroom | Edureka
https://learning.edureka.co/classroom/presentation/651/6520/111155?tab=CourseContent
My Classroom > PySpark Certification Training Course
PySpark Certification T...
Playing with Spark RDDs > Presentation
Course Content
Presentation
Case Study I - Perfor...
Case Study II - Load ...
In-Class Demo - Inpu...
Dataset
Dataset
RDD Practicals
Apache Sqoop
Word_Count_Class
Personal Library
Presentation
28 / 100
Some Other Transformations
intersection
flatMap
union
groupBy
>>> words=sc.parallelize("Hello, welcome to edureka")
>>> words.flatMap(lambda line:list(line)).take(3)
['H', 'e', 'l']
edureka!
Copyright © edureka and/or its affiliates. All rights reserved.
31°C
Haze
Search
ENG
IN
23:15
01-06-2023


e! Course Classroom | Edureka
https://learning.edureka.co/classroom/presentation/651/6520/111155?tab=CourseContent
My Classroom > PySpark Certification Training Course
PySpark Certification T...
Playing with Spark RDDs > Presentation
Course Content
Presentation
Case Study I - Perfor...
Case Study II - Load ...
In-Class Demo - Inpu...
Dataset
Dataset
RDD Practicals
Apache Sqoop
Word_Count_Class
Personal Library
Presentation
29 / 100
Some Other Transformations
intersection
flatMap
union
groupBy
>>> inputRDD = sc.textFile("hdfs://nameservice1/user/edureka_294428/demo.txt")
>>> HelloRDD = inputRDD.filter(lambda x: "Hello" in x)
>>> WelcomeRDD = inputRDD.filter(lambda x: "Welcome" in x)
>>> LinesRDD = HelloRDD.union(WelcomeRDD)
edureka!
Copyright © edureka and/or its affiliates. All rights reserved.
31°C Haze
Search
ENG IN
23:15 01-06-2023


e! Course Classroom | Edureka
https://learning.edureka.co/classroom/presentation/651/6520/111155?tab=CourseContent
My Classroom > PySpark Certification Training Course
PySpark Certification T...
Playing with Spark RDDs > Presentation
Course Content
Presentation
Case Study I - Perfor...
Case Study II - Load ...
In-Class Demo - Inpu...
Dataset
Dataset
RDD Practicals
Apache Sqoop
Word_Count_Class
Personal Library
Presentation
30 / 100
Some Other Transformations
intersection
flatMap
union
groupBy
>>> filedata = sc.textFile("hdfs://nameservice1/user/edureka_294428/Data.csv")
>>> rows = filedata.map(lambda line: line.split(","))
>>> stateaskey = rows.map(lambda a: (a[1], a[0]))
>>> stateaskey.groupByKey().map(lambda x : (x[0], list(x[1]))).collect()
edureka!
Copyright © edureka and/or its affiliates. All rights reserved.
31°C Haze
Search
ENG IN
23:15 01-06-2023

Actions

Spark forces the calculations for execution only when actions are invoked on the RDDs

Action	Meaning
reduce(func)	Aggregate the elements of the dataset using a function func (which takes two arguments and returns one). The function should be commutative and associative so that it can be computed correctly in parallel
first()	Return the first element of the dataset (similar to take(1))
takeOrdered(n, [ordering])	Return the first n elements of the RDD using either their natural order or a custom comparator
count()	Return the number of elements in the dataset
collect()	Return all the elements of the dataset as an array at the driver program. This is usually useful after a filter or other operation that returns a sufficiently small subset of the data
saveAsSequenceFile(path)	Write the elements of the dataset as a Hadoop SequenceFile in a given path in the local file system, HDFS or any other Hadoop-supported file system
foreach(func)	Run a function func on each element of the dataset. This is usually done for side effects such as updating an accumulator variable
countByValue()	Counts the number of values in a given RDD

edureka!

Presentation 32 / 100

Some Popular Actions

- reduce
- count
- first
- countByValue

Aggregate the elements of the dataset using a function func (which takes two arguments and returns one). The function should be commutative and associative so that it can be computed correctly in parallel

```
>>> sc.parallelize(range(1,50)).reduce(lambda a, b: a+b)
1225
```

edureka!

Presentation 33 / 100

Some Popular Actions

- reduce
- count — Returns the number of rows in the dataset(i.e. RDD)
- first
- countByValue

```
>>> words_data=sc.parallelize("Welcome to PySpark Training".split(" "))
>>> words_data.count()
4
```

edureka!

Presentation 34 / 100

Some Popular Actions

- reduce
- count
- first — Returns the first value/element from the dataset
- countByValue

```
>>> words_data=sc.parallelize("Welcome to PySpark Training".split(" "))
>>> words_data.first()
'Welcome'
```

edureka!

Presentation 35 / 100

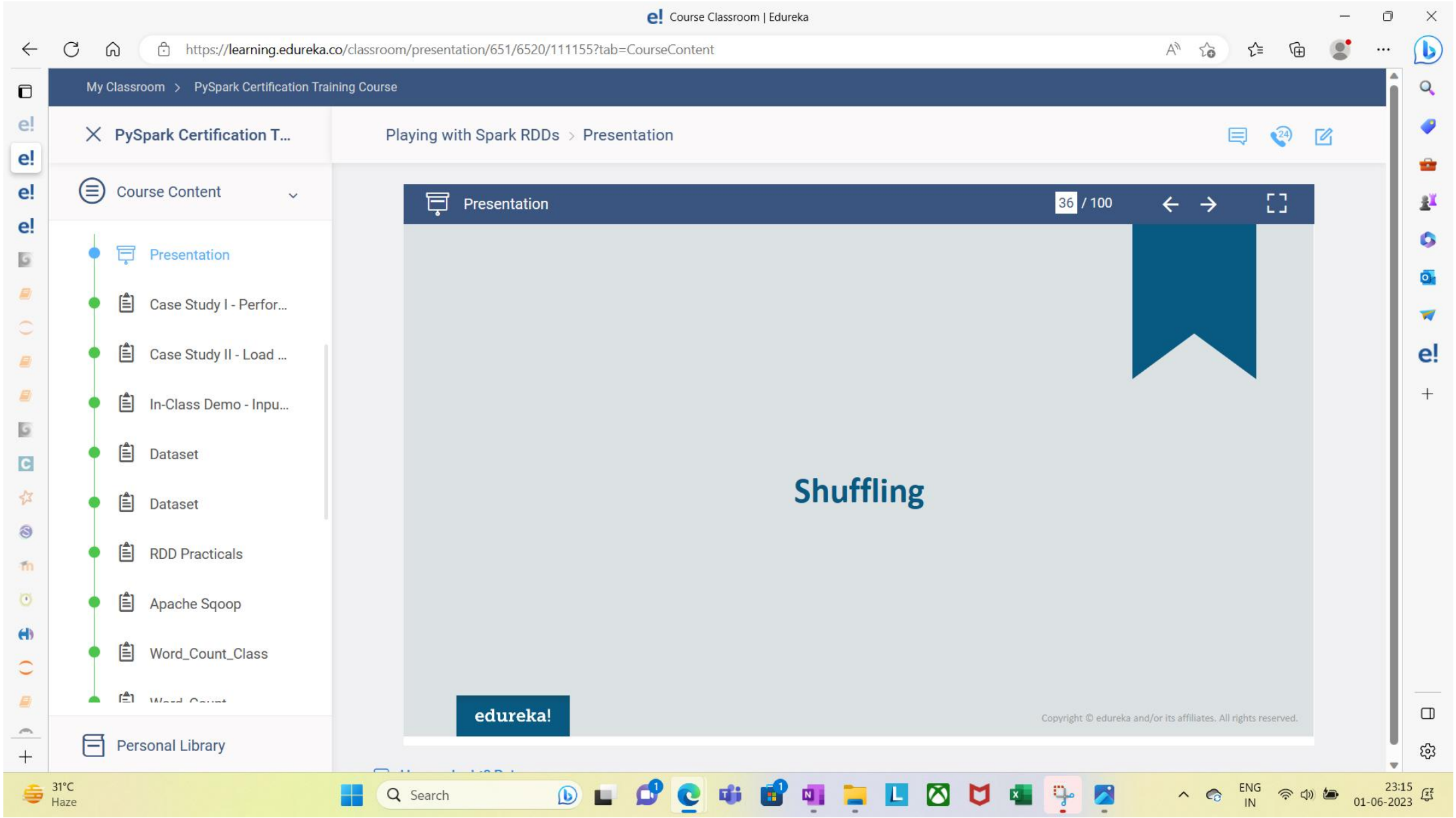
Some Popular Actions

- reduce
- count
- first
- countByValue

Returns the values present in the RDD

```
>>> words_data=sc.parallelize("Welcome to PySpark Training".split(" "))
>>> words_data.countByValue()
defaultdict(<type 'int'>, {'to': 1, 'Training': 1, 'Welcome': 1, 'PySpark': 1})
```

edureka!


Course Classroom | Edureka
https://learning.edureka.co/classroom/presentation/651/6520/111155?tab=CourseContent
My Classroom > PySpark Certification Training Course
PySpark Certification T...
Playing with Spark RDDs > Presentation
Course Content
Presentation
Case Study I - Perfor...
Case Study II - Load ...
In-Class Demo - Inpu...
Dataset
Dataset
RDD Practicals
Apache Sqoop
Word_Count_Class
Personal Library
Presentation
36 / 100
Shuffling
edureka!
Copyright © edureka and/or its affiliates. All rights reserved.
31°C
Haze
Search
ENG
IN
23:15
01-06-2023

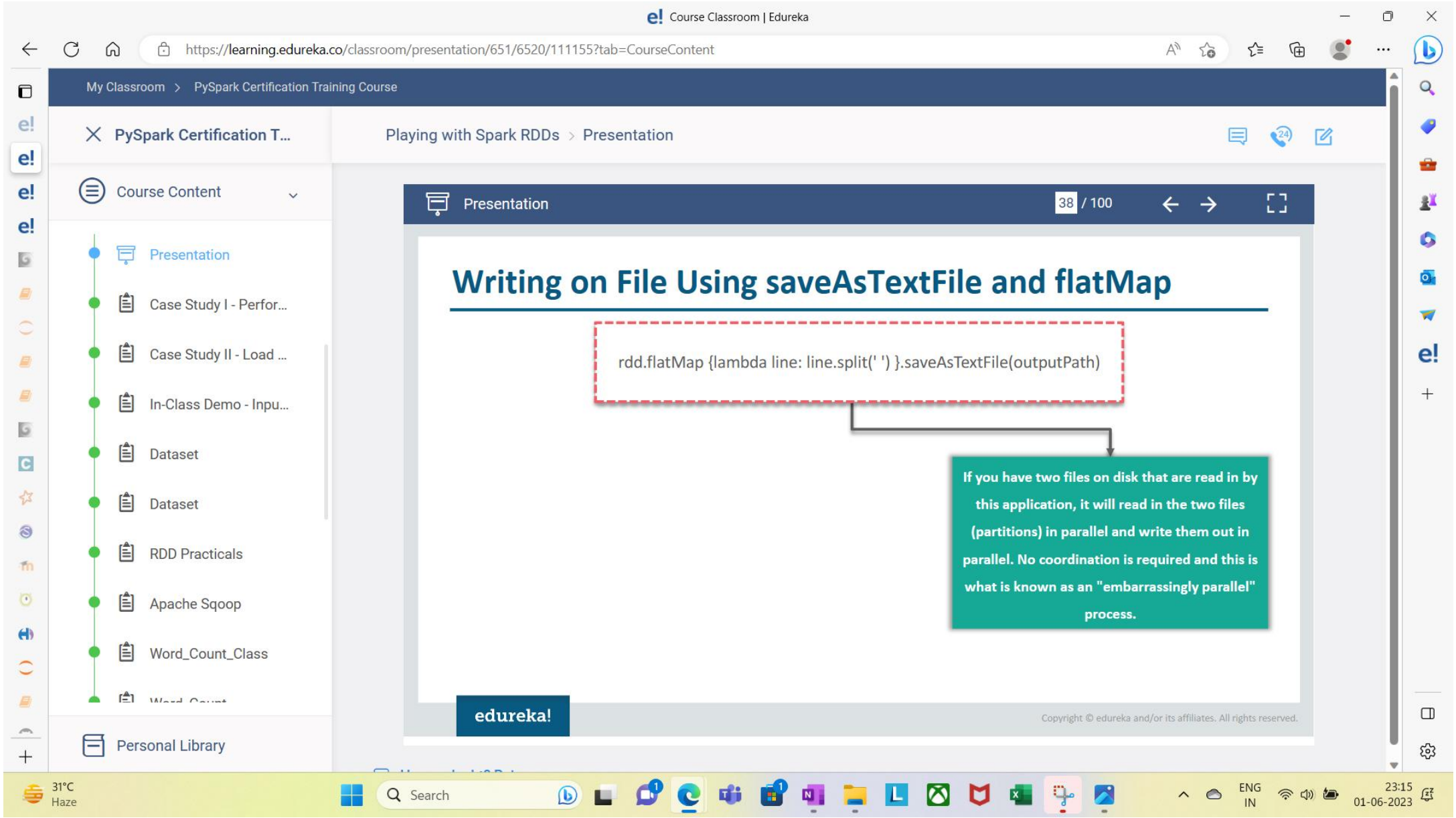
Presentation 37 / 100

Writing on File Using saveAsTextFile and flatMap

```
rdd.flatMap {lambda line: line.split(' ') }.saveAsTextFile(outputPath)
```

The memory usage of this is actually fairly straightforward. In the case of a text file, it streams the file in, line by line applies your flatMap operation and immediately writes (possibly buffering) it out to disk. Therefore you're not really doing anything terribly memory intensive and Spark doesn't really do anything fancy in terms of coordination

edureka!


Course Classroom | Edureka
https://learning.edureka.co/classroom/presentation/651/6520/111155?tab=CourseContent
My Classroom > PySpark Certification Training Course
PySpark Certification T...
Playing with Spark RDDs > Presentation
Course Content
Presentation
Case Study I - Perfor...
Case Study II - Load ...
In-Class Demo - Inpu...
Dataset
Dataset
RDD Practicals
Apache Sqoop
Word_Count_Class
Personal Library
Presentation
38 / 100
Writing on File Using saveAsTextFile and flatMap
rdd.flatMap {lambda line: line.split(' ') }.saveAsTextFile(outputPath)
If you have two files on disk that are read in by this application, it will read in the two files (partitions) in parallel and write them out in parallel. No coordination is required and this is what is known as an "embarrassingly parallel" process.
edureka!
Copyright © edureka and/or its affiliates. All rights reserved.
31°C
Haze
Search
ENG
IN
23:15
01-06-2023

Presentation 39 / 100

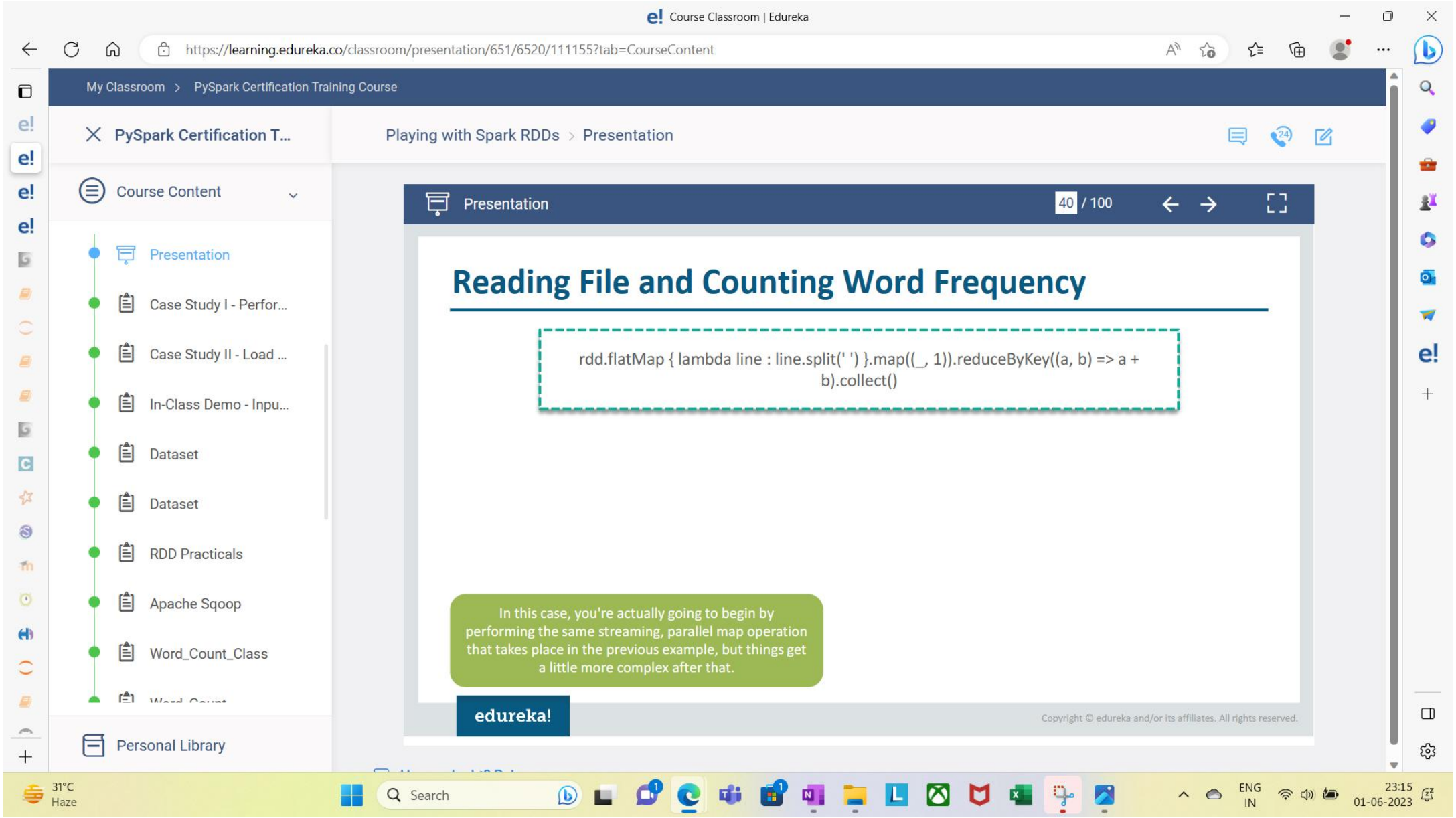
Reading File and Counting Word Frequency

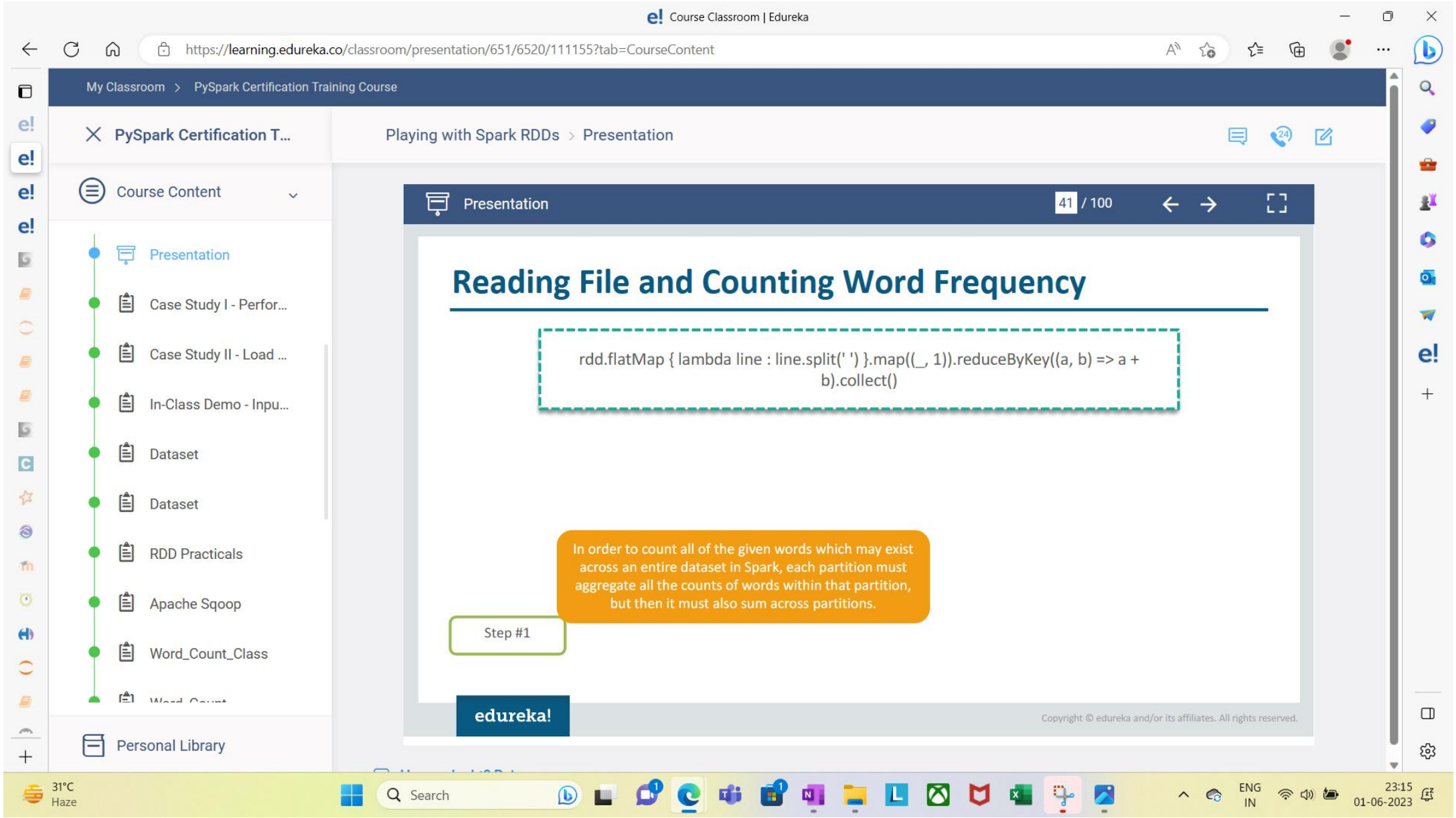
```
rdd.flatMap { lambda line : line.split(' ') }.map((_, 1)).reduceByKey((a, b) => a + b).collect()
```

Lets see the working of this line of code

Note: This code splits each line by whitespace, turns each word into a row and each row into a tuple of (word, 1) which then gets aggregated to result in (word, # of occurrences for that word)

edureka!


Course Classroom | Edureka
https://learning.edureka.co/classroom/presentation/651/6520/111155?tab=CourseContent
My Classroom > PySpark Certification Training Course
PySpark Certification T...
Playing with Spark RDDs > Presentation
Course Content
Presentation
Case Study I - Perfor...
Case Study II - Load ...
In-Class Demo - Inpu...
Dataset
Dataset
RDD Practicals
Apache Sqoop
Word_Count_Class
Personal Library
Presentation
40 / 100
Reading File and Counting Word Frequency
rdd.flatMap { lambda line : line.split(' ') }.map((_, 1)).reduceByKey((a, b) => a + b).collect()
In this case, you're actually going to begin by performing the same streaming, parallel map operation that takes place in the previous example, but things get a little more complex after that.
edureka!
Copyright © edureka and/or its affiliates. All rights reserved.
31°C Haze
Search
ENG IN
23:15 01-06-2023


Course Classroom | Edureka
https://learning.edureka.co/classroom/presentation/651/6520/111155?tab=CourseContent
My Classroom > PySpark Certification Training Course
PySpark Certification T...
Playing with Spark RDDs > Presentation
Course Content
Presentation
Case Study I - Perfor...
Case Study II - Load ...
In-Class Demo - Inpu...
Dataset
Dataset
RDD Practicals
Apache Sqoop
Word_Count_Class
Personal Library
Presentation
41 / 100
Reading File and Counting Word Frequency
rdd.flatMap { lambda line : line.split(' ') }.map((_, 1)).reduceByKey((a, b) => a + b).collect()
In order to count all of the given words which may exist across an entire dataset in Spark, each partition must aggregate all the counts of words within that partition, but then it must also sum across partitions.
Step #1
edureka!
Copyright © edureka and/or its affiliates. All rights reserved.
31°C Haze
Search
ENG IN
23:15 01-06-2023

Presentation 42 / 100

Reading File and Counting Word Frequency

```
rdd.flatMap { lambda line : line.split(' ') }.map((_, 1)).reduceByKey((a, b) => a + b).collect()
```

The process of moving the data from partition to partition in order to aggregate, join, match up, or spread out in some other way, is known as **shuffling.**

Step #1

Step #2

edureka!

Presentation 43 / 100

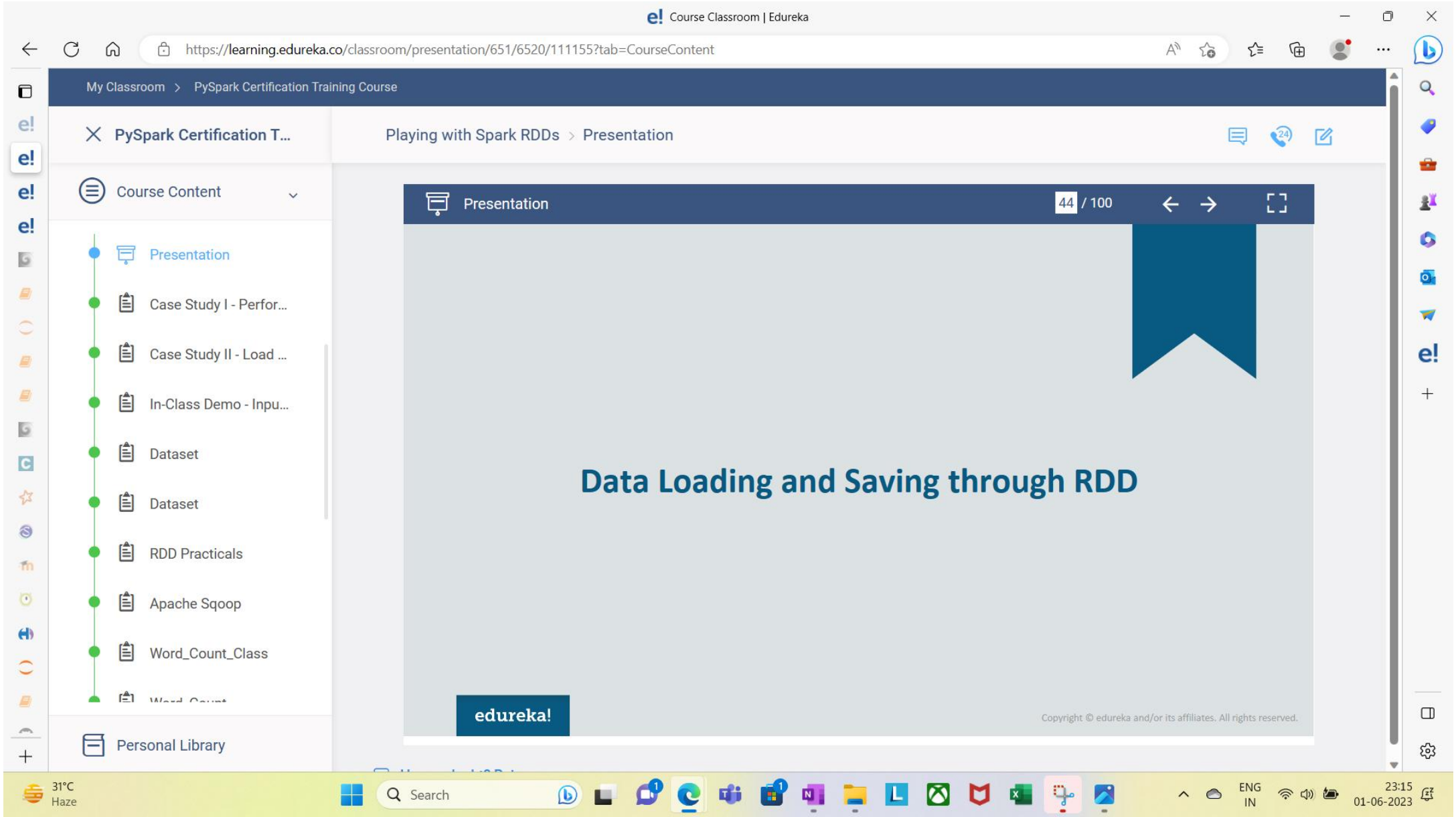
Reading File and Counting Word Frequency

```
rdd.flatMap { lambda line : line.split(' ') }.map((_, 1)).reduceByKey((a, b) => a + b).collect()
```

The following operations are examples of shuffle inducing operations for RDDs:

- **groupBy/subtractByKey/foldByKey/aggregateByKey/reduceByKey**
- **cogroup**
- **any of the join transformations**
- **distinct**

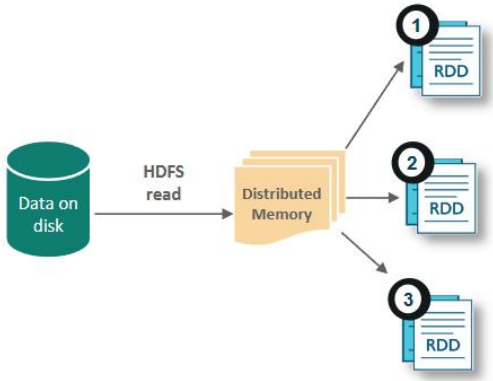
edureka!


e! Course Classroom | Edureka
https://learning.edureka.co/classroom/presentation/651/6520/111155?tab=CourseContent
My Classroom > PySpark Certification Training Course
PySpark Certification T...
Playing with Spark RDDs > Presentation
Course Content
Presentation
Case Study I - Perfor...
Case Study II - Load ...
In-Class Demo - Inpu...
Dataset
Dataset
RDD Practicals
Apache Sqoop
Word_Count_Class
Personal Library
Presentation
44 / 100
Data Loading and Saving through RDD
edureka!
Copyright © edureka and/or its affiliates. All rights reserved.
31°C
Haze
Search
ENG
IN
23:15
01-06-2023

Data Loading in RDD

- RDD is *re-computed* every time when it is *materialized*
- So it is a good idea to *improve performance* by *caching* a *RDD* if it is accessed frequently
- One of the easiest way to *load data in RDD* is to load from a *Python collection*:

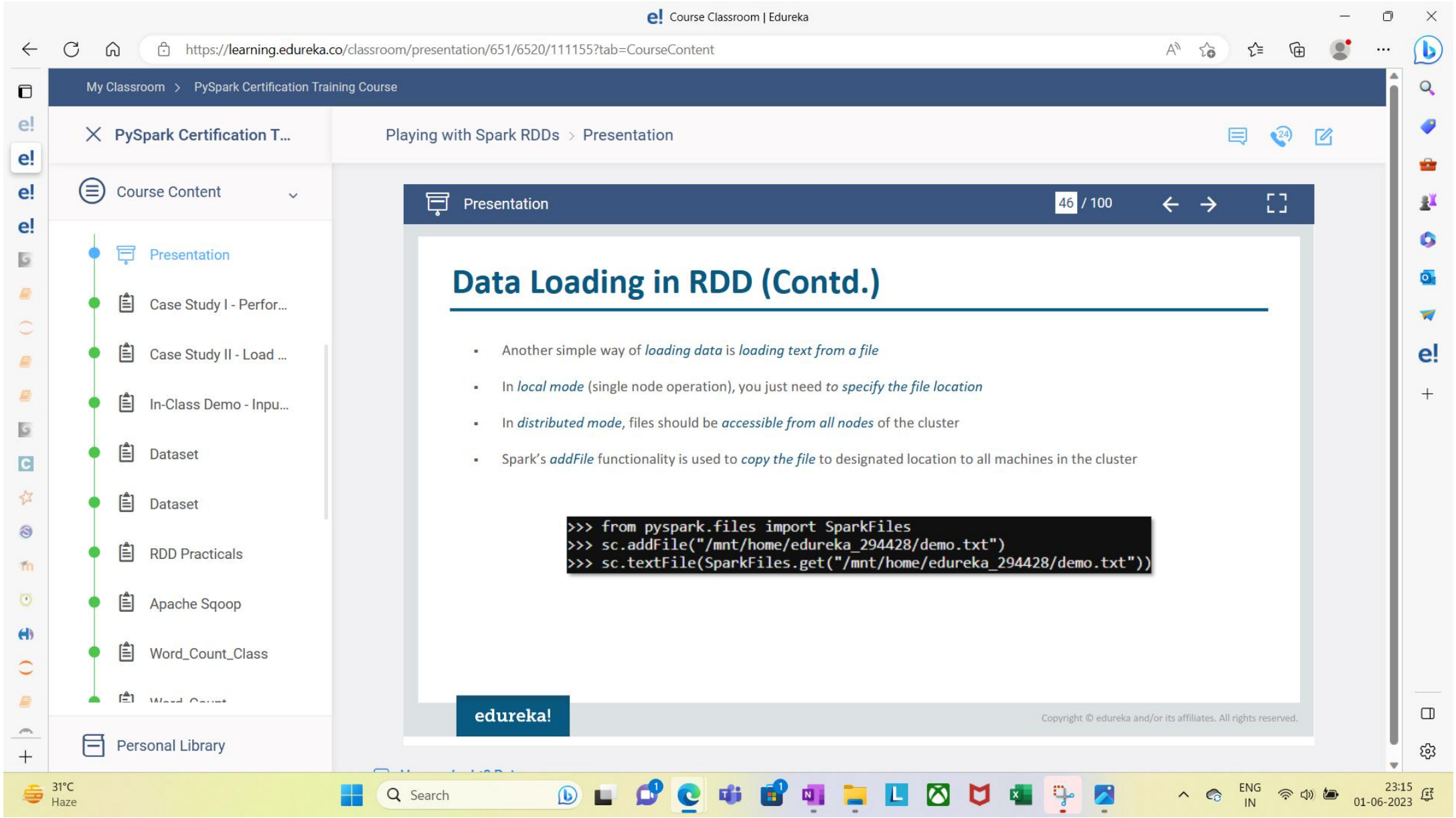
```
>>> rdd=sc.parallelize([1, 2, 3, 4, 5])
>>> rdd.collect()
[1, 2, 3, 4, 5]
```



- SparkContext provides parallelize function, which converts the Python collection into the RDD of the same type:

```
dataRDD: org.apache.spark.rdd.RDD[Int] = ParallelCollectionRDD[0] at parallelize at <console>:12
```

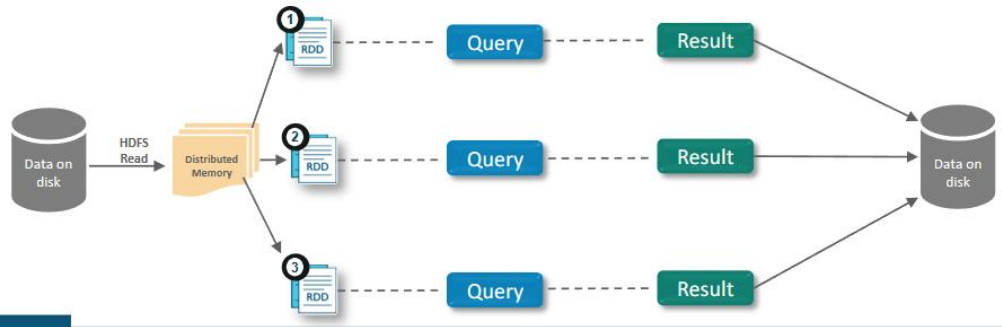
edureka!


Course Classroom | Edureka
https://learning.edureka.co/classroom/presentation/651/6520/111155?tab=CourseContent
My Classroom > PySpark Certification Training Course
PySpark Certification T...
Playing with Spark RDDs > Presentation
Course Content
Presentation
Case Study I - Perfor...
Case Study II - Load ...
In-Class Demo - Inpu...
Dataset
Dataset
RDD Practicals
Apache Sqoop
Word_Count_Class
Personal Library
Presentation
46 / 100
Data Loading in RDD (Contd.)
Another simple way of loading data is loading text from a file
In local mode (single node operation), you just need to specify the file location
In distributed mode, files should be accessible from all nodes of the cluster
Spark's addFile functionality is used to copy the file to designated location to all machines in the cluster
>>> from pyspark.files import SparkFiles
>>> sc.addFile("/mnt/home/edureka_294428/demo.txt")
>>> sc.textFile(SparkFiles.get("/mnt/home/edureka_294428/demo.txt"))
edureka!
Copyright © edureka and/or its affiliates. All rights reserved.
31°C Haze
Search
ENG IN
23:15 01-06-2023

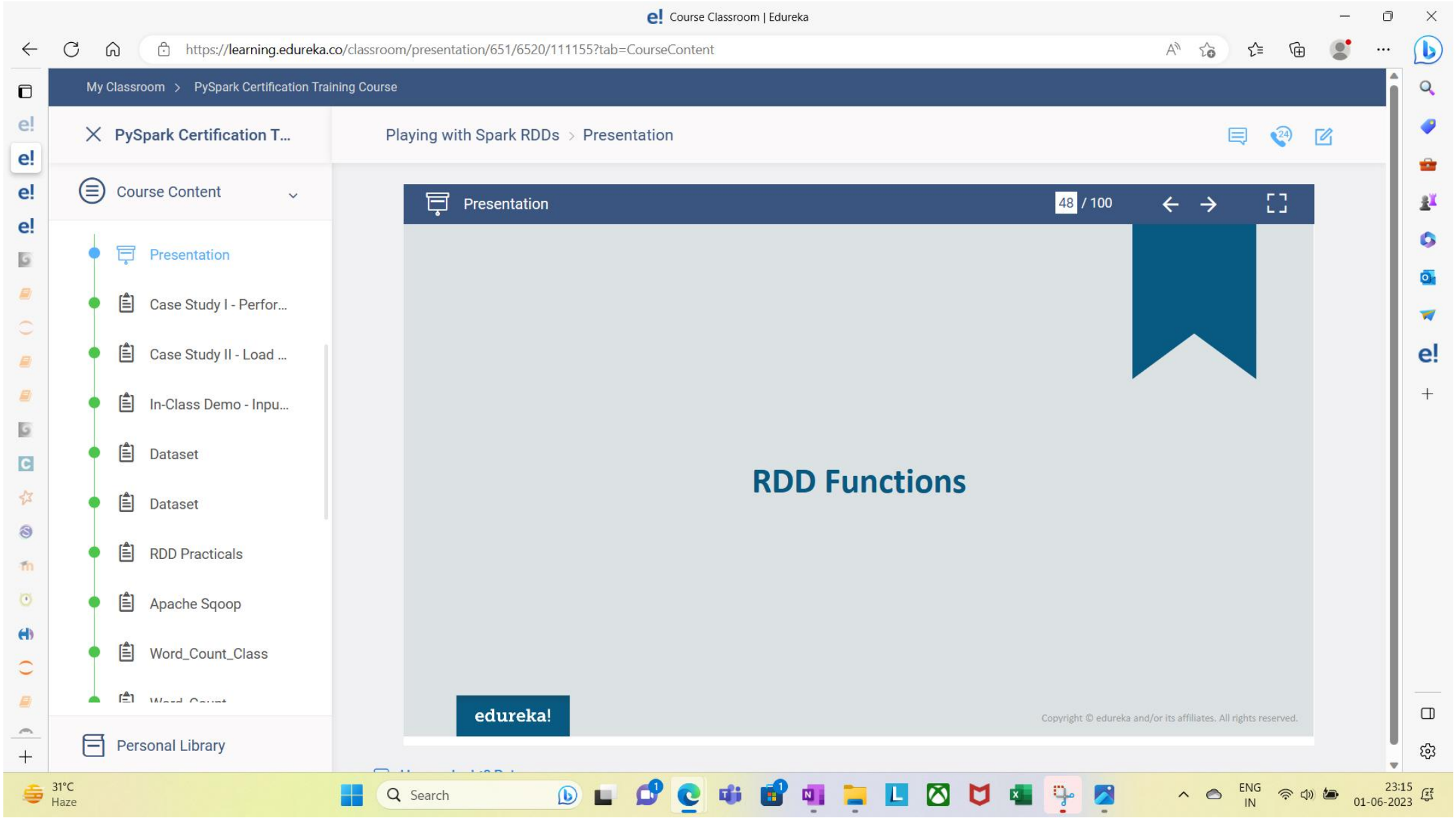
Saving Data Through RDD

- While the methods for loading an RDD are largely found in the *SparkContext* class
- Methods for saving an RDD are defined on the RDD classes
- In Python, implicit conversion exists so that an RDD that can be saved as a sequence file is converted to the appropriate type

```
>>> words=sc.parallelize("Hello, welcome to edureka")
>>> words.saveAsTextFile("/user/edureka_294428/output.txt")
```



edureka!


Course Classroom | Edureka
https://learning.edureka.co/classroom/presentation/651/6520/111155?tab=CourseContent
My Classroom > PySpark Certification Training Course
PySpark Certification T...
Playing with Spark RDDs > Presentation
Course Content
Presentation
Case Study I - Perfor...
Case Study II - Load ...
In-Class Demo - Inpu...
Dataset
Dataset
RDD Practicals
Apache Sqoop
Word_Count_Class
Personal Library
Presentation
48 / 100
RDD Functions
edureka!
Copyright © edureka and/or its affiliates. All rights reserved.
31°C
Haze
Search
ENG
IN
23:15
01-06-2023

Presentation 49 / 100

General RDD Functions

Function	Argument	Returns
countApprox	()	It is an approximation of count()
coalesce	()	Collapses partitions on the same worker in order to avoid a shuffle of the data when repartitioning
zip	()	zip allows you to "zip" together two RDDs, assuming that they have the same length
foreach	(f: T => Unit)	Applies the function f to each element
persist	() ,(newLevel: StorageLevel)	Sets the RDD storage level, which can cause the RDD to be stored after it is computed. Different StorageLevels can be seen in StorageLevel
sample	(fraction: double)	RDD of that fraction
toDebugString	()	A handy function that outputs the recursive deps of the RDD
unpersist	()	Remove all the persistent blocks of the RDD from the memory/disk

edureka!

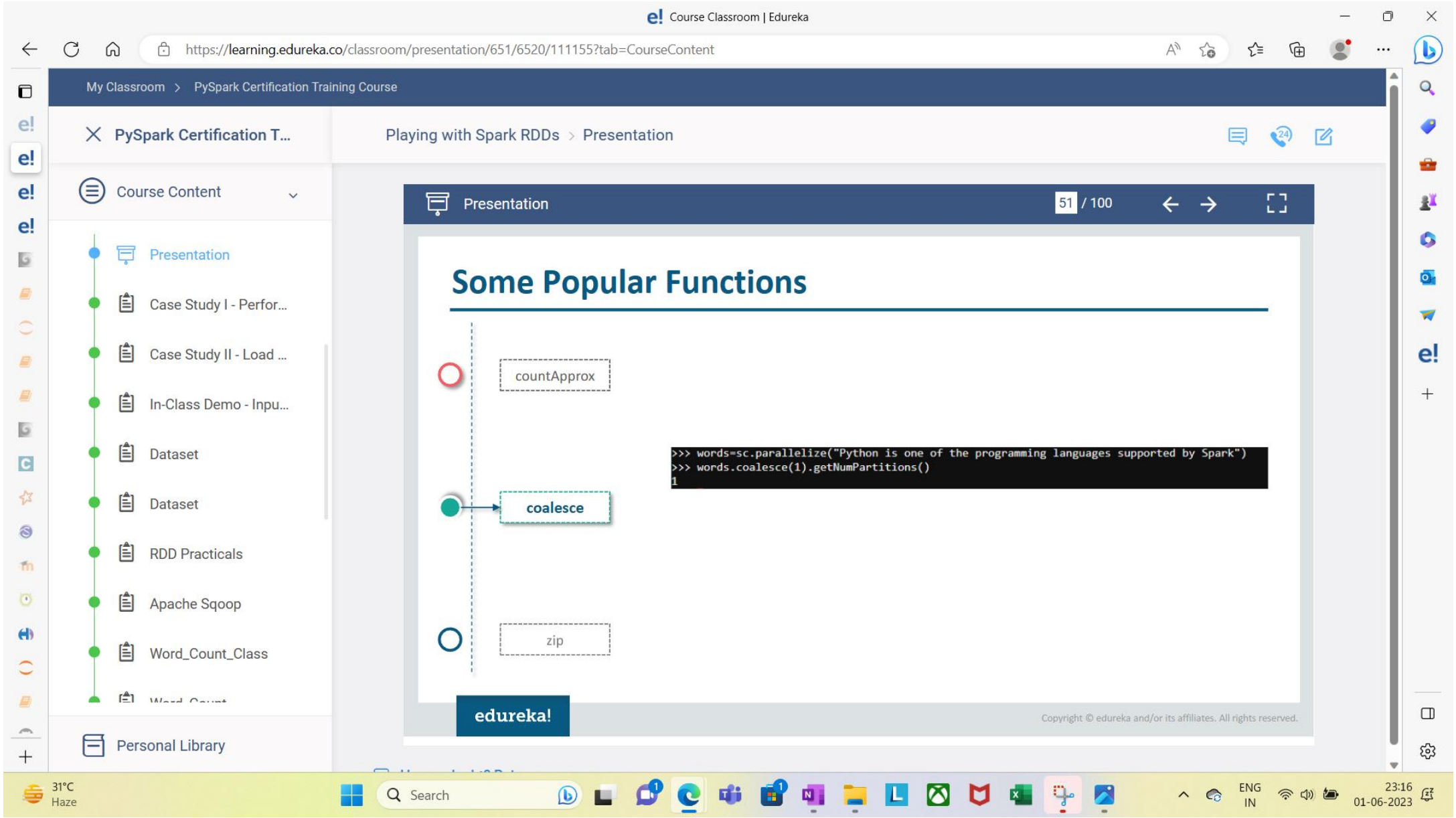
Presentation 50 / 100

Some Popular Functions

- countApprox
- coalesce
- zip

```
>>> words=sc.parallelize("hello".split(" "))
>>> words.countApprox(400,0.95)
1
```

edureka!


e! Course Classroom | Edureka
https://learning.edureka.co/classroom/presentation/651/6520/111155?tab=CourseContent
My Classroom > PySpark Certification Training Course
PySpark Certification T...
Playing with Spark RDDs > Presentation
Course Content
Presentation
Case Study I - Perfor...
Case Study II - Load ...
In-Class Demo - Inpu...
Dataset
Dataset
RDD Practicals
Apache Sqoop
Word_Count_Class
Personal Library
Presentation
51 / 100
Some Popular Functions
countApprox
coalesce
zip
>>> words=sc.parallelize("Python is one of the programming languages supported by Spark")
>>> words.coalesce(1).getNumPartitions()
1
edureka!
Copyright © edureka and/or its affiliates. All rights reserved.
31°C
Haze
Search
ENG
IN
23:16
01-06-2023

Presentation 52 / 100

Some Popular Functions

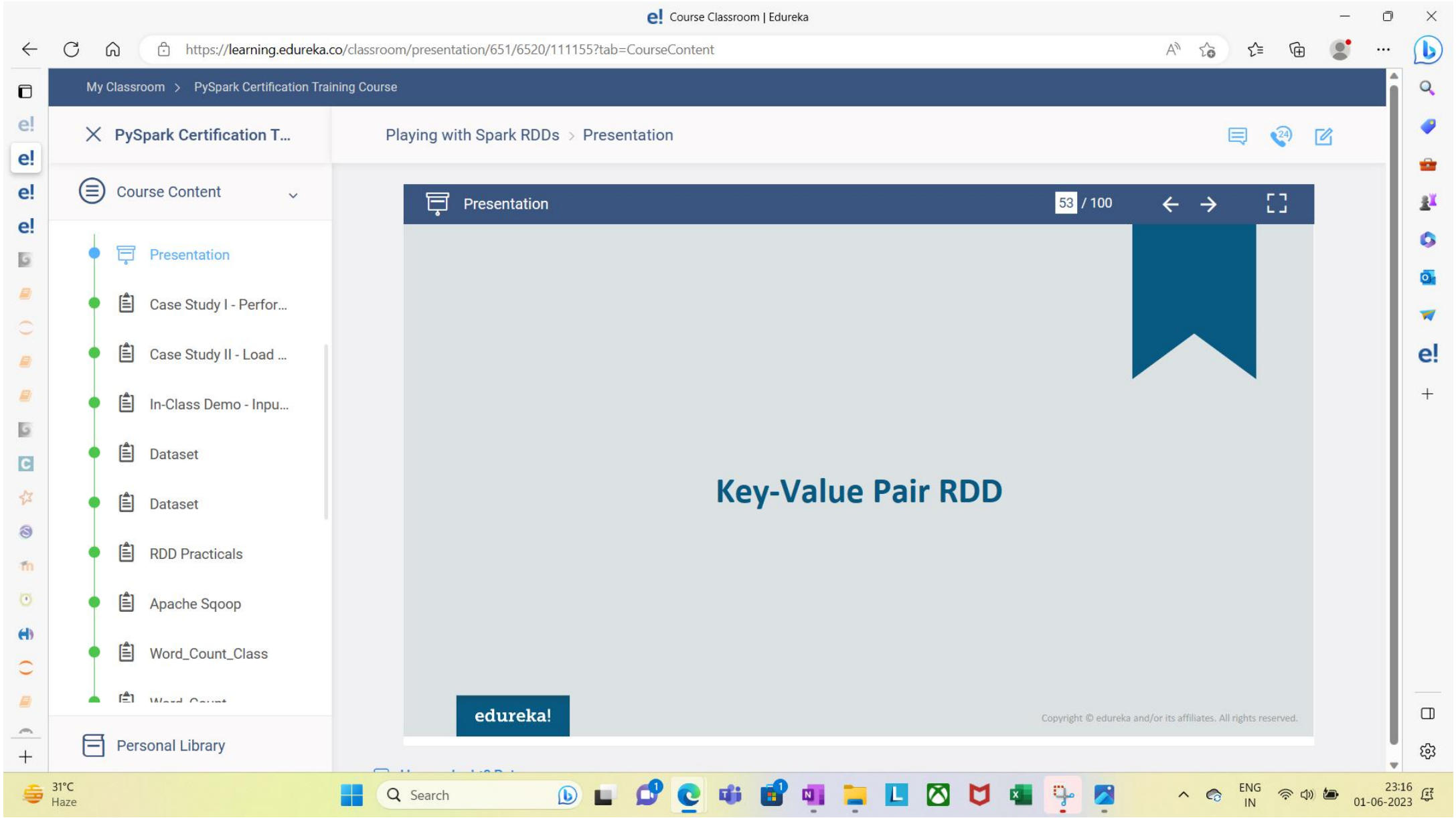
countApprox

coalesce

zip

```
>>> words=sc.parallelize("Python is one of the programming languages supported by Spark".split(" "))
>>> numRange = sc.parallelize(range(10), 2)
>>> words.zip(numRange).collect()
```

edureka!


e! Course Classroom | Edureka
https://learning.edureka.co/classroom/presentation/651/6520/111155?tab=CourseContent
My Classroom > PySpark Certification Training Course
PySpark Certification T...
Playing with Spark RDDs > Presentation
Course Content
Presentation
Case Study I - Perfor...
Case Study II - Load ...
In-Class Demo - Inpu...
Dataset
Dataset
RDD Practicals
Apache Sqoop
Word_Count_Class
Personal Library
Presentation
53 / 100
Key-Value Pair RDD
edureka!
Copyright © edureka and/or its affiliates. All rights reserved.
31°C
Haze
Search
ENG
IN
23:16
01-06-2023

Presentation 54 / 100

Key-Value Pair RDD

- Pair RDDs can be created by running a map() function that returns key-value pairs. The procedure to build the key-value RDDs differs by language
- In Python, for the functions on keyed data to be available, we also need to return tuples

Function	Parameter options	Explanation	Return Type
foldByKey	(zeroValue) (func(V,V)=>V)	Merges the values using the provided function. Unlike a traditional fold function over a list, the zeroValue can be added an arbitrary number of times.	RDD[K,V]
reduceByKey	(func(V,V)=>V, numTasks)	Parallel version of reduce that merges the values for each key using the provided function and returns an RDD.	RDD[K,V]
groupByKey	(numPartitions)	Groups elements together by key.	RDD[K,Seq[V]]

edureka!

Presentation 55 / 100

Other Pair RDD Functions

Function	Parameter options	Explanation	Return Type
lookup	(key: K)	Looks up a specific element in the RDD. Uses the RDD's partitioner to figure out which partition(s) to look at.	Seq[V]
mapValues	(f: v=>u)	A specialized version of map for PairRDD when you only want to change the value of the key-value pair. If you need to make your changes based on both key and value, you must use one of the normal RDD Map functions.	RDD[K,U]
collectAsMap	()	Takes an RDD and returns a concrete map. Your RDD must be able to fit into the memory.	Map[K, V]
countByKey	()	Counts the number of elements for each key in RDD.	Map[K, Long]
partitionBy	(partitioner:P, mapSideCombine: Boolean)	Returns a new RDD with the same data but partitioned by the new Partitioner, and mapSideCombine controls Spark group values with the same key together before repartitioning. Defaults to false.	RDD[K,V]
flatMapValues	(f:V => TraversableOnce[U])	Similar to mapValues. A specialized version of flatMap for PairRDDs when you only want to change the value of the key-value pair. Takes the provided Map function and applies it to the value. The resulting sequence is then "flattened".	RDD[K, U]

edureka!

Presentation 56 / 100

Some Popular Functions

- keyBy
- lookup
- mapValues
- sampleBykey

```
>>> words=sc.parallelize("Hello World, Welcome to edureka".split(" "))
>>> keyword = words.keyBy(lambda word: word.lower()[0])
>>> keyword.collect()
[('h', 'Hello'), ('w', 'World,'), ('w', 'Welcome'), ('t', 'to'), ('e', 'edureka')]
```

edureka!

Presentation 57 / 100

Some Popular Functions

- keyBy
- lookup
- mapvalues
- sampleBykey

```
>>> keyword.lookup("w")
['World,', 'Welcome']
```

edureka!

Presentation 58 / 100

Some Popular Functions

- keyBy
- lookup
- **mapValues**
- sampleByKey

```
>>> keyword.mapValues(lambda word: word.upper()).collect()
[('h', 'HELLO'), ('w', 'WORLD,'), ('w', 'WELCOME'), ('t', 'TO'), ('e', 'EDUREKA')]
```

edureka!

Presentation 59 / 100

Some Popular Functions

- keyBy
- lookup
- mapValues
- **sampleByKey**

```
>>> import random
>>> distinctChars = words.flatMap(lambda word: list(word.lower())).distinct().collect()
>>> sampleMap = dict(map(lambda c: (c, random.random()), distinctChars))
>>> words.map(lambda word: (word.lower()[0], word)).sampleByKey(True, sampleMap, 6).collect()
[('e', 'edureka')]
```

edureka!

Presentation 60 / 100

Double RDD Functions

Function	Argument	Returns
mean	()	Average of RDDs elements
sum	()	Sum of Elements
variance	()	Variance of RDD elements
stats	()	Mean, Variance and Count as Stats Counter

edureka!

Presentation 61 / 100

Double RDD Functions

mean

sum

variance

stats

Example

```
>>> a = sc . parallelize ([11, 12, 9, 7, 5, 10])
>>> a.mean()
9.0
```

edureka!

Presentation 62 / 100

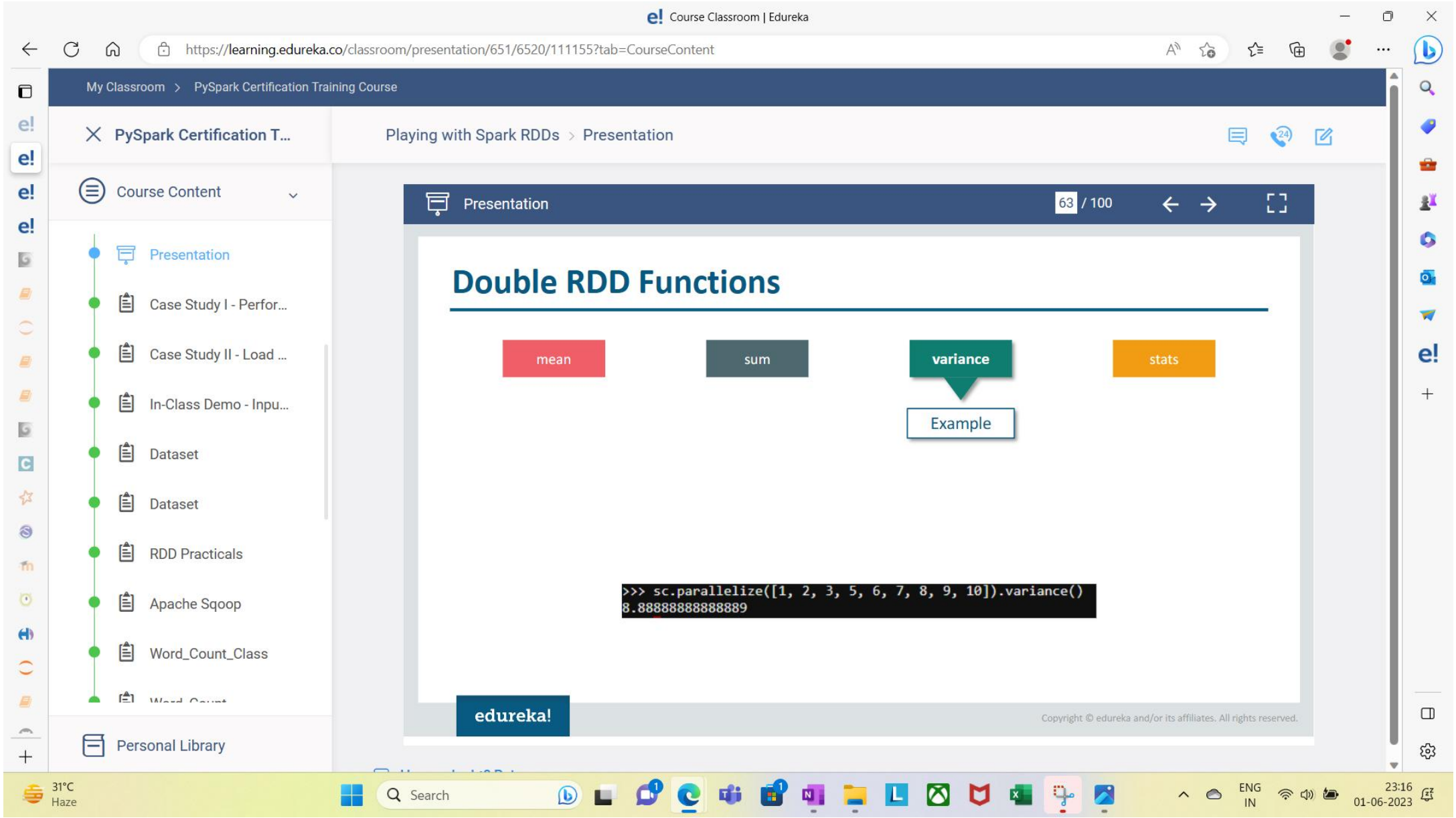
Double RDD Functions

mean | sum | variance | stats

Example

```
>>> sc.parallelize([39, 21, 34, 11, 5, 70]).sum()
180
```

edureka!


Course Classroom | Edureka
https://learning.edureka.co/classroom/presentation/651/6520/111155?tab=CourseContent
My Classroom > PySpark Certification Training Course
PySpark Certification T...
Playing with Spark RDDs > Presentation
Course Content
Presentation
Case Study I - Perfor...
Case Study II - Load ...
In-Class Demo - Inpu...
Dataset
Dataset
RDD Practicals
Apache Sqoop
Word_Count_Class
Personal Library
Presentation
63 / 100
Double RDD Functions
mean
sum
variance
stats
Example
>>> sc.parallelize([1, 2, 3, 5, 6, 7, 8, 9, 10]).variance()
8.88888888888889
edureka!
Copyright © edureka and/or its affiliates. All rights reserved.
31°C
Haze
Search
ENG
IN
23:16
01-06-2023

Presentation 64 / 100

Double RDD Functions

mean | sum | variance | stats

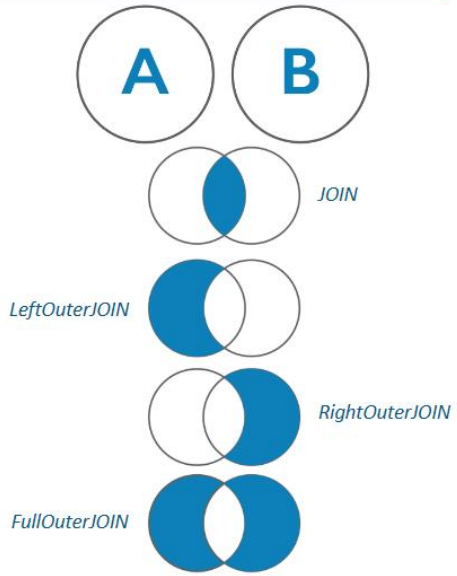
Example

```
>>> sc.parallelize([10, 20, 30, 40, 50, 60, 70, 80, 90, 100]).stats()
(count: 10, mean: 55.0, stdev: 28.7228132327, max: 100, min: 10)
```

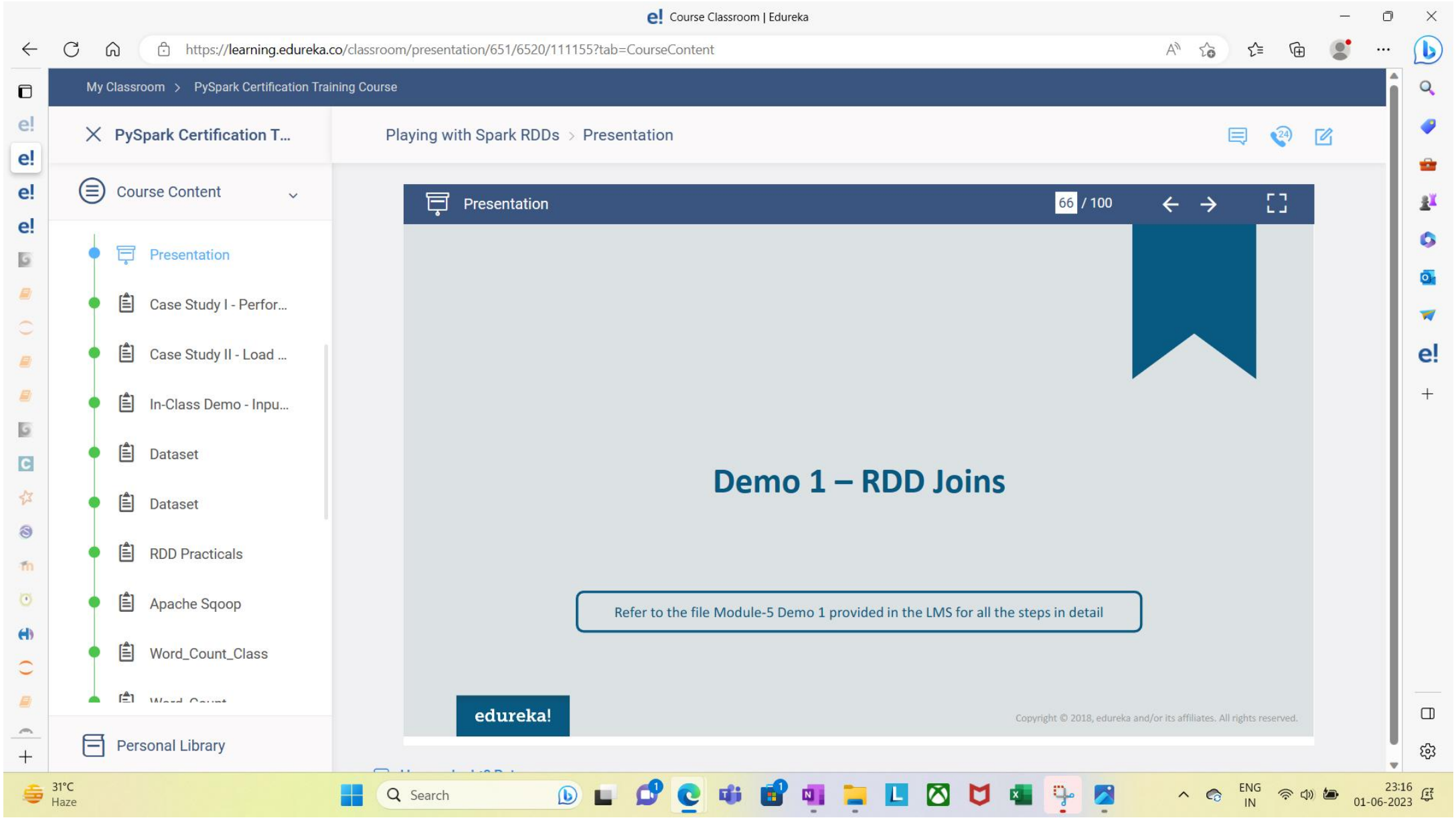
edureka!

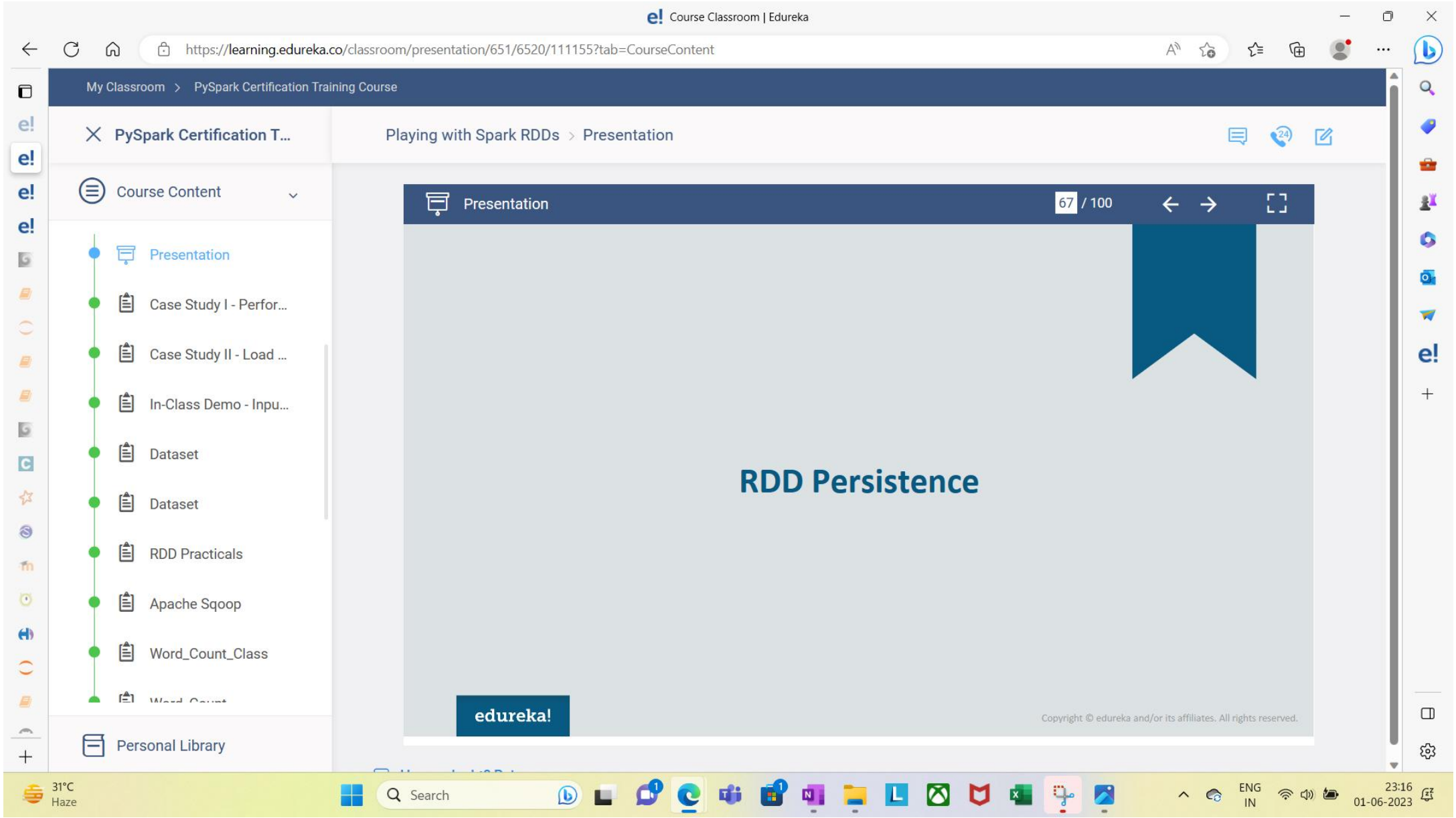
RDD Joins

- Spark allows *easy manipulation* of *multiple data sets* with out-of-the-box join operators
- Spark Pair RDDs provide ***join***, ***leftOuterJoin***, ***rightOuterJoin***, ***fullOuterJoin*** methods to perform respective joins
- Programmers need to create the RDDs carefully, as the *keys* are *automatically chosen by* the *join operation*
- Pair RDDs also provide the *cartesian* and *cogroup methods* to *perform* the *respective operations*



edureka!


Course Classroom | Edureka
https://learning.edureka.co/classroom/presentation/651/6520/111155?tab=CourseContent
My Classroom > PySpark Certification Training Course
PySpark Certification T...
Playing with Spark RDDs > Presentation
Course Content
Presentation
Case Study I - Perfor...
Case Study II - Load ...
In-Class Demo - Inpu...
Dataset
Dataset
RDD Practicals
Apache Sqoop
Word_Count_Class
Personal Library
Presentation
66 / 100
Demo 1 – RDD Joins
Refer to the file Module-5 Demo 1 provided in the LMS for all the steps in detail
edureka!
Copyright © 2018, edureka and/or its affiliates. All rights reserved.
31°C
Haze
Search
ENG
IN
23:16
01-06-2023


Course Classroom | Edureka
https://learning.edureka.co/classroom/presentation/651/6520/111155?tab=CourseContent
My Classroom > PySpark Certification Training Course
PySpark Certification T...
Playing with Spark RDDs > Presentation
Course Content
Presentation
Case Study I - Perfor...
Case Study II - Load ...
In-Class Demo - Inpu...
Dataset
Dataset
RDD Practicals
Apache Sqoop
Word_Count_Class
Personal Library
Presentation
67 / 100
RDD Persistence
edureka!
Copyright © edureka and/or its affiliates. All rights reserved.
31°C
Haze
Search
ENG
IN
23:16
01-06-2023

Presentation 68 / 100

RDD Persistence

- Spark RDDs are *lazily evaluated*, thus when we may wish to use the same RDD multiple times, it results in *recomputing* the RDD
- To avoid computing an RDD multiple times, we can ask Spark to *persist the data*
- In this case, the node that computes the RDD *stores* their *partitions*
- If a node that has data persisted on it fails, Spark will *recompute the lost partitions* of the data when needed
- When you attempt to *cache too much data* to fit in memory, Spark will automatically *evict old partitions* using a *Least Recently Used (LRU)* cache policy
- RDDs come with a method called ***unpersist()*** that lets you *manually remove* them from the cache

edureka!

Presentation 69 / 100

Persistence Level

Level	Space used	CPU time	In memory	On disk	Comments
MEMORY_ONLY	High	Low	Y	N	RDD is stored as deserialized JAVA object in JVM. If the RDD does not fit in memory, some partitions will not be cached and will be recomputed when needed
MEMORY_ONLY_SER	Low	High	Y	N	Stores RDDs as serialized JAVA object. It stores one-byte array per partition
MEMORY_AND_DISK	High	Medium	Some	Some	RDD is stored as deserialized JAVA object in JVM. If the full RDD does not fit in memory then the remaining partition is stored on disk, instead of recomputing it
MEMORY_AND_DISK_SER	Low	High	Some	Some	Spills to disk if there is too much data to fit in memory. Stores serialized representation in memory.
DISK_ONLY	Low	High	N	Y	Store the RDD partitions only on disk

edureka!

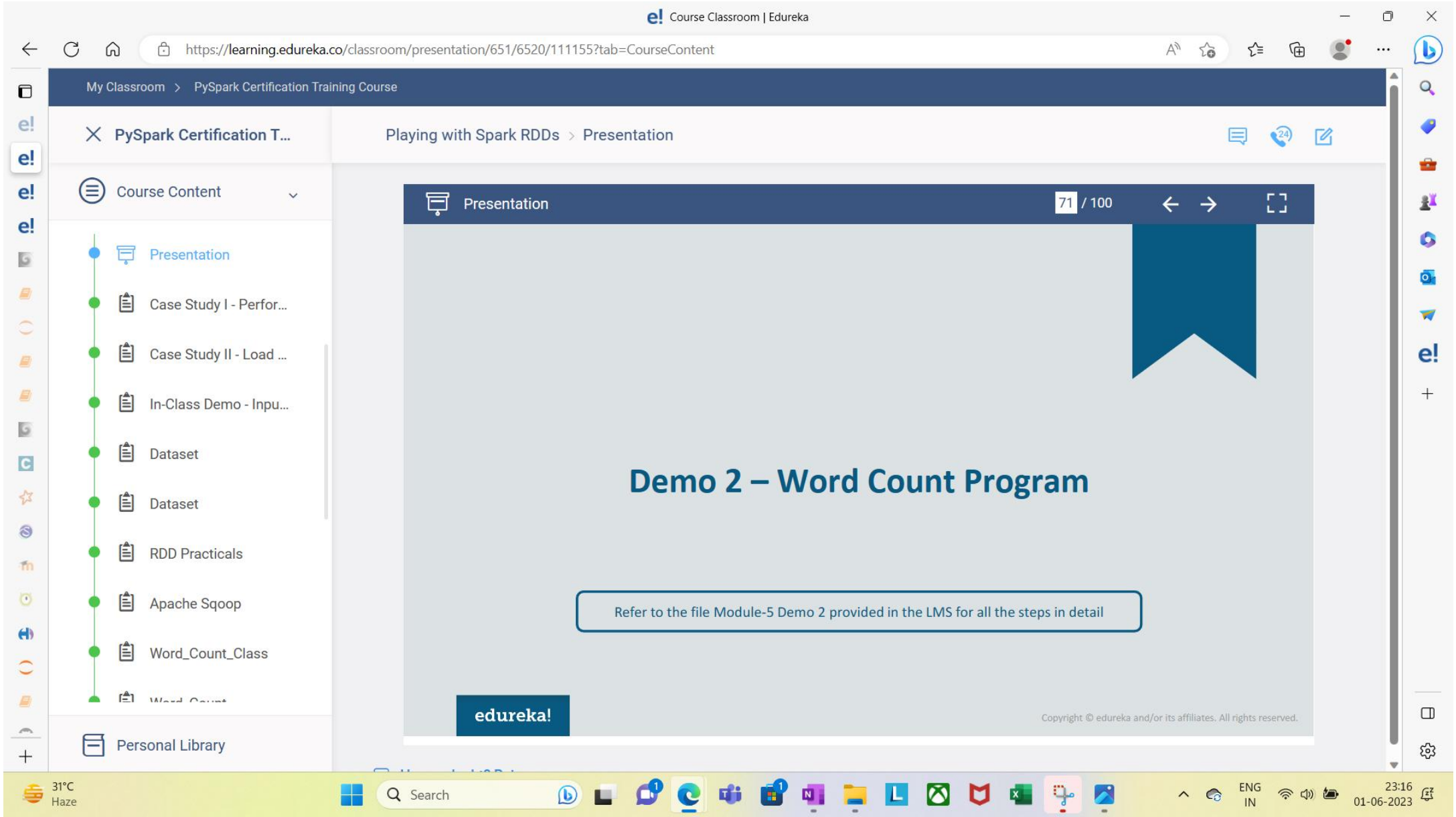
RDD Persistence (Contd.)

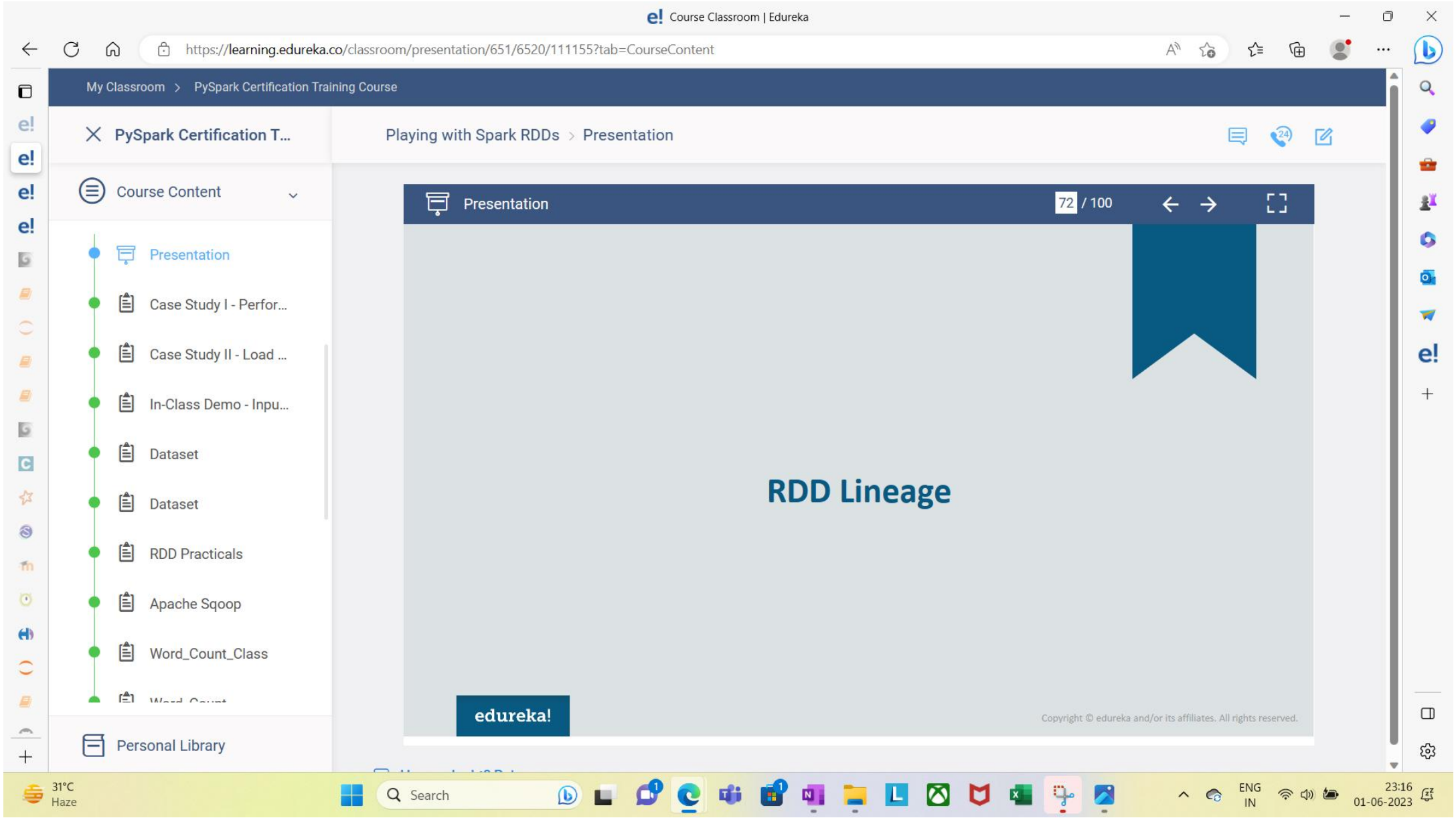
- *Caching* is very *useful* when *data* is *accessed repeatedly*, such as when querying a small "hot" dataset or when running an iterative algorithm
- The cache is *fault-tolerant*
- As a simple example, let's revisit our Scala example and mark our linesWithSpark dataset to be cached:

Example:

```
>>> rdd1=sc.textFile("/user/edureka_294428/susmita_demo/Rishabh.txt")
>>> rdd1.first()
u'Happy Learning!'
>>> rdd1.count()
2
>>> rdd2=rdd1.map(lambda word:word.upper())
>>> rdd2.collect()
[u'HAPPY LEARNING!', u'WELCOME TO EDUREKA']
>>> rdd2.cache()
PythonRDD[33] at collect at <stdin>:1
```

edureka!


Course Classroom | Edureka
https://learning.edureka.co/classroom/presentation/651/6520/111155?tab=CourseContent
My Classroom > PySpark Certification Training Course
PySpark Certification T...
Playing with Spark RDDs > Presentation
Course Content
Presentation
Case Study I - Perfor...
Case Study II - Load ...
In-Class Demo - Inpu...
Dataset
Dataset
RDD Practicals
Apache Sqoop
Word_Count_Class
Personal Library
Presentation
71 / 100
Demo 2 – Word Count Program
Refer to the file Module-5 Demo 2 provided in the LMS for all the steps in detail
edureka!
Copyright © edureka and/or its affiliates. All rights reserved.
31°C
Haze
Search
ENG
IN
23:16
01-06-2023

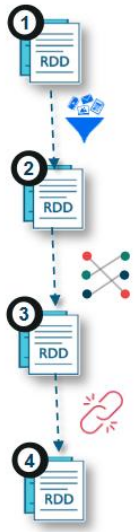

Course Classroom | Edureka
https://learning.edureka.co/classroom/presentation/651/6520/111155?tab=CourseContent
My Classroom > PySpark Certification Training Course
PySpark Certification T...
Playing with Spark RDDs > Presentation
Course Content
Presentation
Case Study I - Perfor...
Case Study II - Load ...
In-Class Demo - Inpu...
Dataset
Dataset
RDD Practicals
Apache Sqoop
Word_Count_Class
Personal Library
Presentation
72 / 100
RDD Lineage
edureka!
Copyright © edureka and/or its affiliates. All rights reserved.
31°C
Haze
Search
ENG
IN
23:16
01-06-2023

RDD Lineage

- ***RDD Lineage*** (RDD dependency graph) is a *graph of* all the *parent RDDs* of *a RDD*. It is built as a result of applying transformations to the RDD and creates a logical execution plan

The given RDD graph could be the result of the following series of transformations:

- A RDD lineage graph is hence a graph of what transformations need to be executed after an action has been called



edureka!

Course Classroom | Edureka

https://learning.edureka.co/classroom/presentation/651/6520/111155?tab=CourseContent

My Classroom > PySpark Certification Training Course

PySpark Certification T...

Playing with Spark RDDs > Presentation

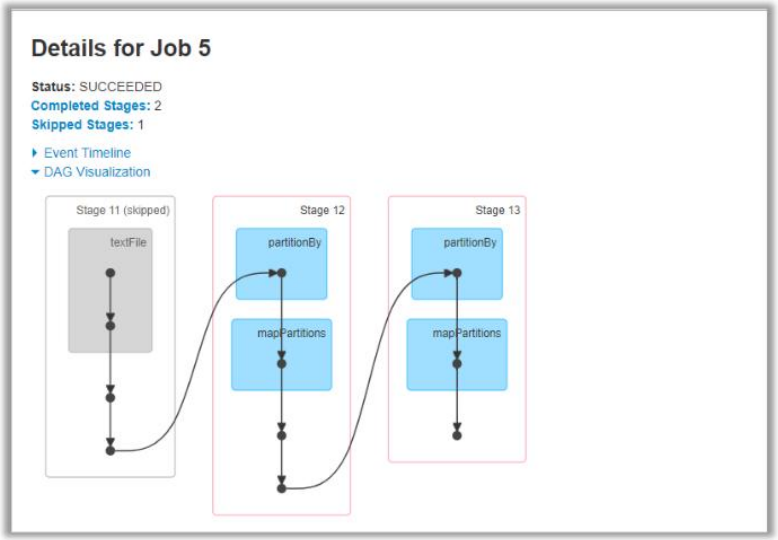
Course Content

- Presentation
- Case Study I - Perfor...
- Case Study II - Load ...
- In-Class Demo - Inpu...
- Dataset
- Dataset
- RDD Practicals
- Apache Sqoop
- Word_Count_Class

Personal Library

Presentation 74 / 100

RDD Lineage



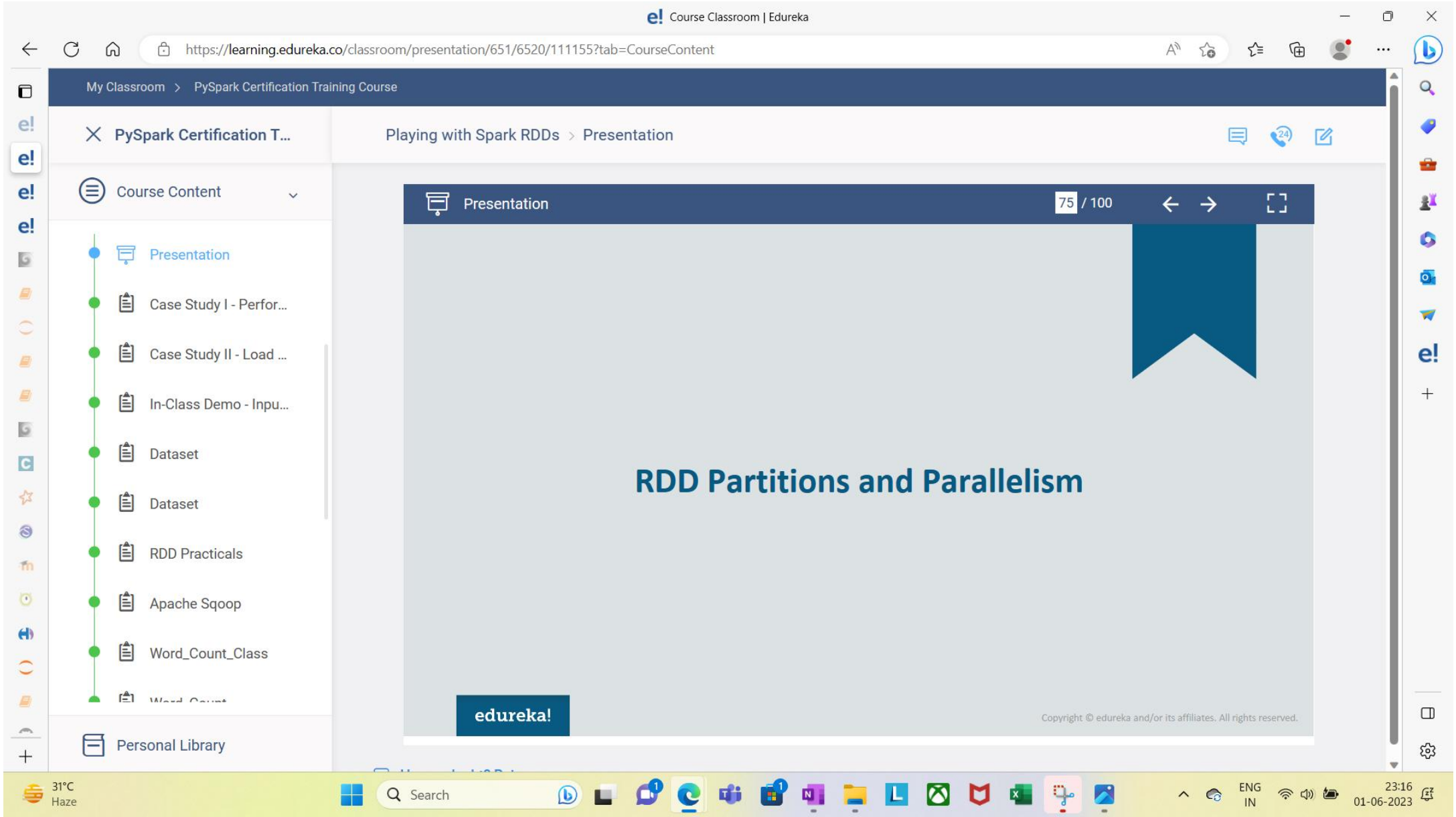
edureka!

31°C Haze

Search

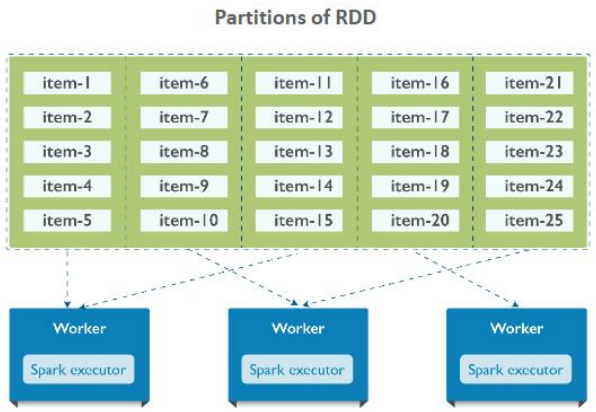
ENG IN

23:16 01-06-2023


e! Course Classroom | Edureka
https://learning.edureka.co/classroom/presentation/651/6520/111155?tab=CourseContent
My Classroom > PySpark Certification Training Course
PySpark Certification T...
Playing with Spark RDDs > Presentation
Course Content
Presentation
Case Study I - Perfor...
Case Study II - Load ...
In-Class Demo - Inpu...
Dataset
Dataset
RDD Practicals
Apache Sqoop
Word_Count_Class
Personal Library
Presentation
75 / 100
RDD Partitions and Parallelism
edureka!
Copyright © edureka and/or its affiliates. All rights reserved.
31°C
Haze
Search
ENG
IN
23:16
01-06-2023

Partitioning RDD and Achieving Parallelism

- A ***partition*** is a *logical chunk* of a *large distributed data set*
- Spark *manages data using partitions* that helps *parallelize distributed data processing* with *minimal network traffic* for sending data between executors
- By default, Spark tries to *read data into an RDD* from the *nodes* that are *close to it*



edureka!

Presentation 77 / 100

Partitioning RDD and Achieving Parallelism

- Since Spark usually accesses distributed partitioned data, to optimize transformation operations it creates partitions to hold the data chunks
- RDDs get partitioned automatically without programmer intervention
- However, there are times when we'd like to adjust the size and number of partitions or the partitioning scheme according to the needs of our application
- We use the given method on an RDD to know the number of partitions in this RDD: **rdd.getNumPartitions()**

```
>>> rdd3=sc.parallelize(range(1,25,3))
>>> rdd3.collect()
[1, 4, 7, 10, 13, 16, 19, 22]
>>> rdd3.getNumPartitions()
2
```

edureka!

Executing a Parallel Task on Shell

```
>>> rdd1=sc.parallelize(range(1,100,5))
>>> rdd1.collect()
[1, 6, 11, 16, 21, 26, 31, 36, 41, 46, 51, 56, 61, 66, 71, 76, 81, 86, 91, 96]
>>> rdd1.getNumPartitions()
2
>>>
```

```
sc.parallelize(range (1,100,2)).collect()
```

2 here, represents the number of partitions the RDD is split into which in turn Runs them parallelly

edureka!

Viewing Partitions (Job History Server)

Go to Job History Server and in Search bar enter your **app id**

Spark 2.1.0.cloudera2 **History Server**

Event log directory: hdfs://nameservice1/user/spark/applicationHistory

Last updated: 6/21/2018, 1:04:28 PM

Show 20 entries Search: edureka_29

App ID	App Name	Started	Spark User	Last Updated	Event Log
application_1528714825862_3994	PySparkShell	2018-06-21 06:46:41	edureka_294428	2018-06-21 06:46:46	Download
application_1528714825862_3086	Spark shell	2018-06-19 13:40:48	edureka_292003	2018-06-19 13:40:47	Download
application_1528714825862_3040	Spark shell	2018-06-19 10:54:50	edureka_292003	2018-06-19 10:54:50	Download
application_1528188402551_0787	Spark shell	2018-06-06 12:55:55	edureka_292003	2018-06-06 14:02:54	Download
application_1517837014215_35537	Spark shell	2018-05-09 10:32:34	edureka_292140	2018-05-09 11:55:36	Download

Showing 1 to 5 of 5 entries (filtered from 398 total entries) Previous 1 Next

edureka!

Presentation 80 / 100

Viewing the Partitions (Job History Server)

Spark Jobs (?)

User: edureka_294428
Total Uptime:
Scheduling Mode: FIFO
Completed Jobs: 8

Event Timeline

Completed Jobs (8)

Total 2 Partitions are done

Job Id	Description	Submitted	Duration	Stages: Succeeded/Total	Tasks (for all stages): Succeeded/Total
7	collect at <stdin>:1	2018/06/21 07:06:31	0.1 s	1/1	2/2

edureka!

Presentation 81 / 100

Viewing Parallelism

Spark 2.1.0 Jobs Stages Storage Environment Executors Spark shell application UI

Details for Stage 0 (Attempt 0)

Total Time Across All Tasks: 17 ms
Locality Level Summary: Process local: 5

DAG Visualization
Show Additional Metrics
Event Timeline
Enable zooming

Scheduler Delay
Task Deserialization Time
Shuffle Read Time
Executor Computing Time
Shuffle Write Time
Result Serialization Time
Getting Result Time

Parallel Execution of 2 completed tasks

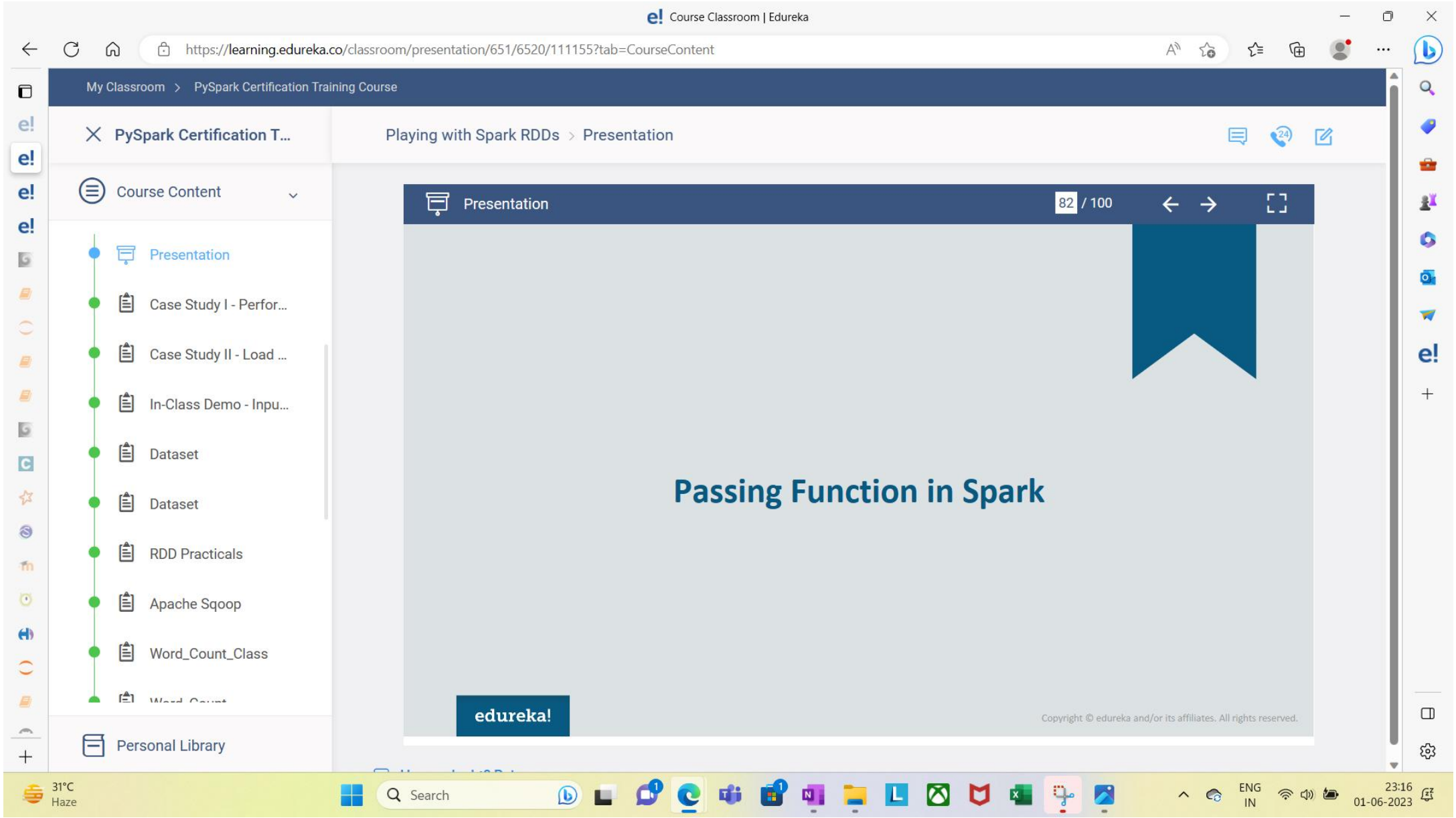
1 / ip-20-0-31-221.ec2.internal

900 950 000 050 100 150 200 250 300
13:41:42 13:41:43

Summary Metrics for 5 Completed Tasks

Metric	Min	25th percentile	Median	75th percentile	Max
Duration	1 ms	1 ms	1 ms	2 ms	12 ms
GC Time	0 ms	0 ms	0 ms	0 ms	0 ms

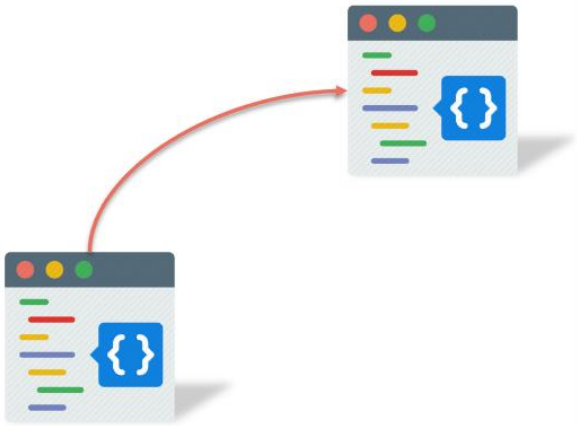
edureka!


Course Classroom | Edureka
https://learning.edureka.co/classroom/presentation/651/6520/111155?tab=CourseContent
My Classroom > PySpark Certification Training Course
PySpark Certification T...
Playing with Spark RDDs > Presentation
Course Content
Presentation
Case Study I - Perfor...
Case Study II - Load ...
In-Class Demo - Inpu...
Dataset
Dataset
RDD Practicals
Apache Sqoop
Word_Count_Class
Personal Library
Presentation
82 / 100
Passing Function in Spark
edureka!
Copyright © edureka and/or its affiliates. All rights reserved.
31°C
Haze
Search
ENG
IN
23:16
01-06-2023

Presentation 83 / 100

Function Passing

- Function Passing and Object Sharing is an inherent part of any standalone Software Application
- When you pass functions to another class or function, you implicitly provide access of the function output and members to the calling function or class

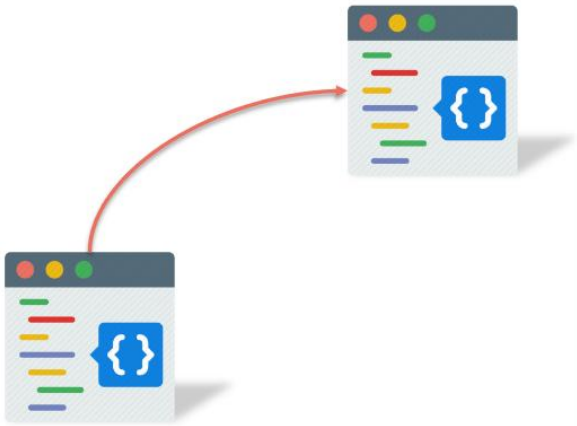


edureka!

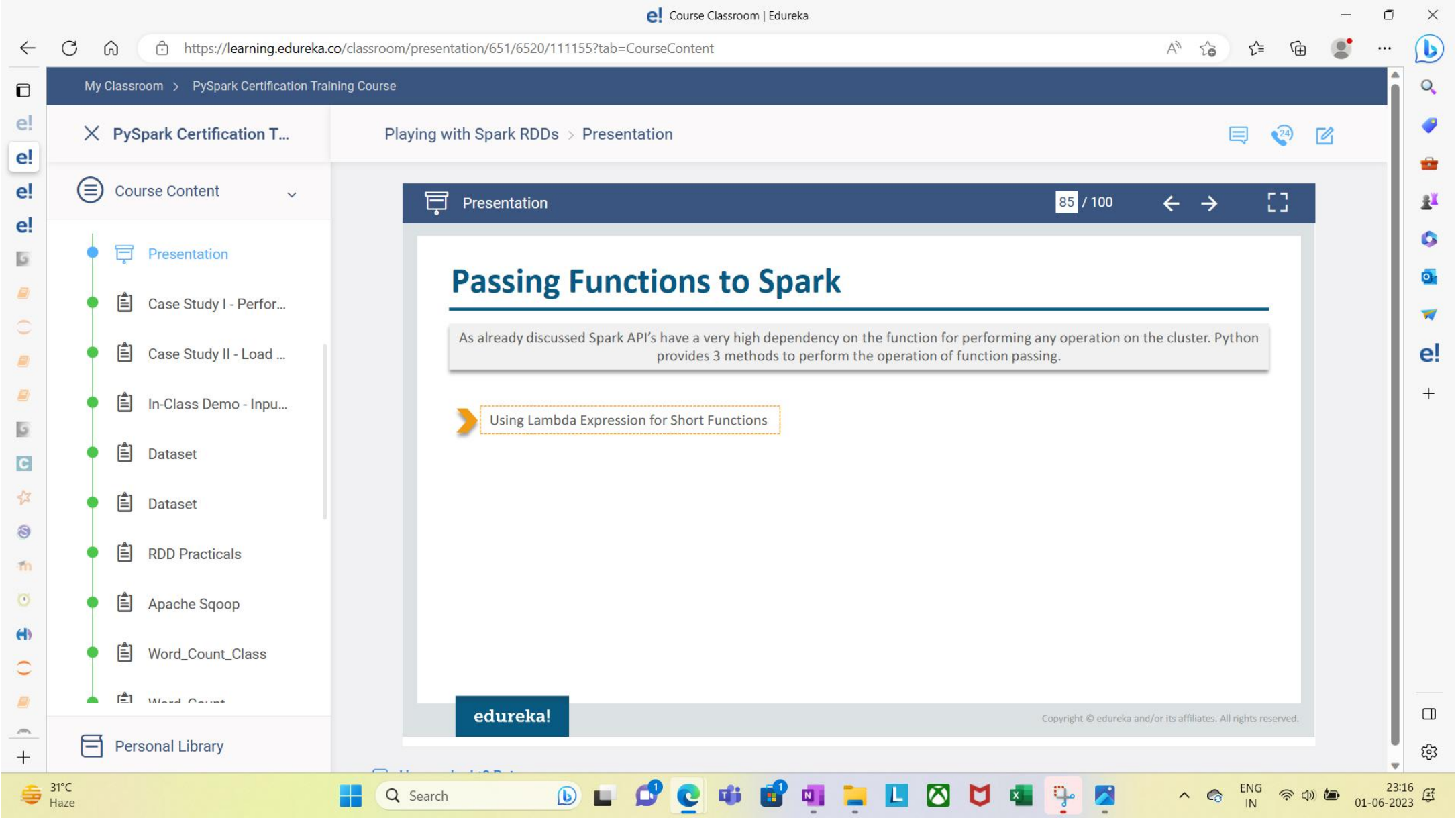
Presentation 84 / 100

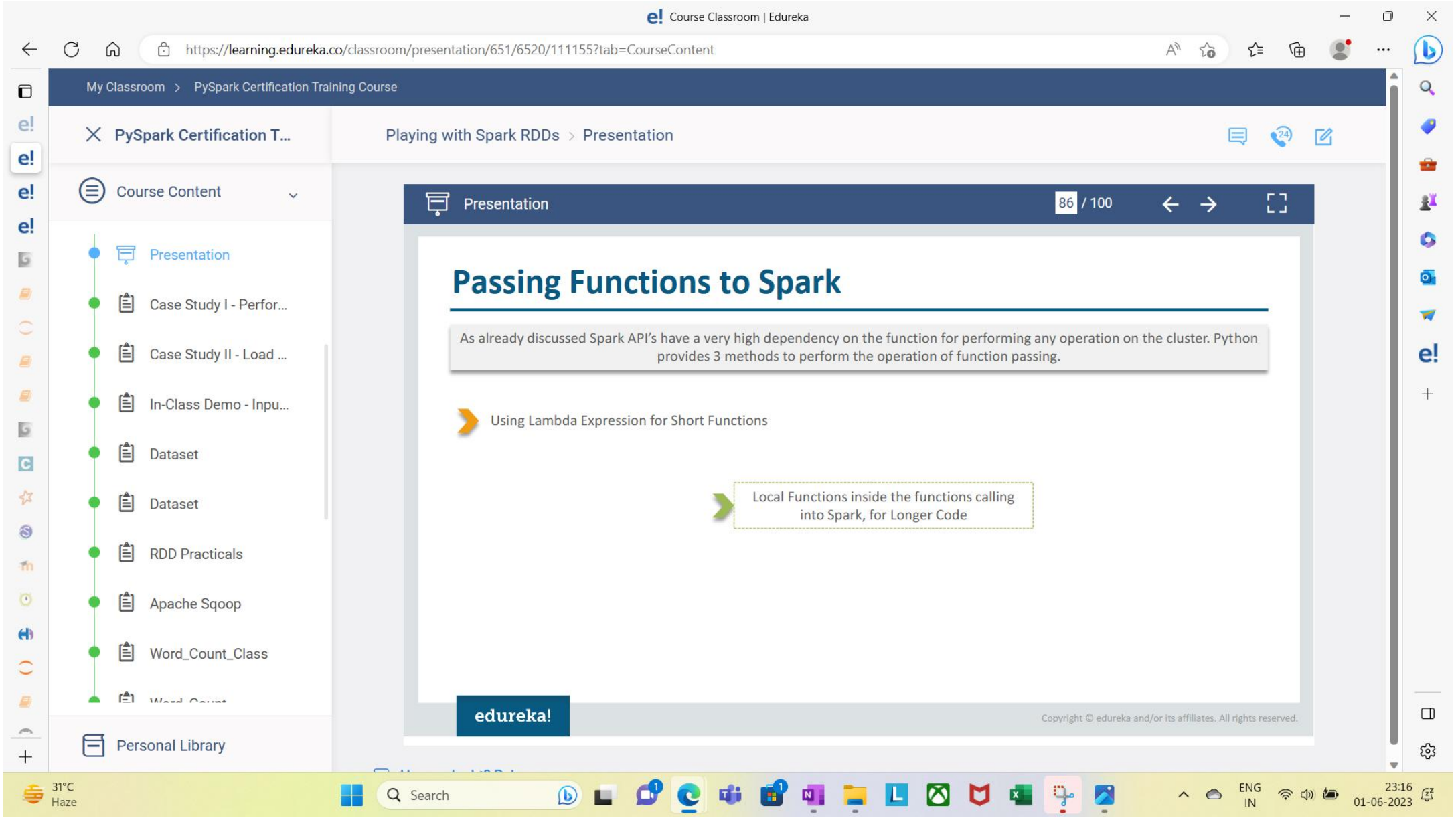
Function Passing

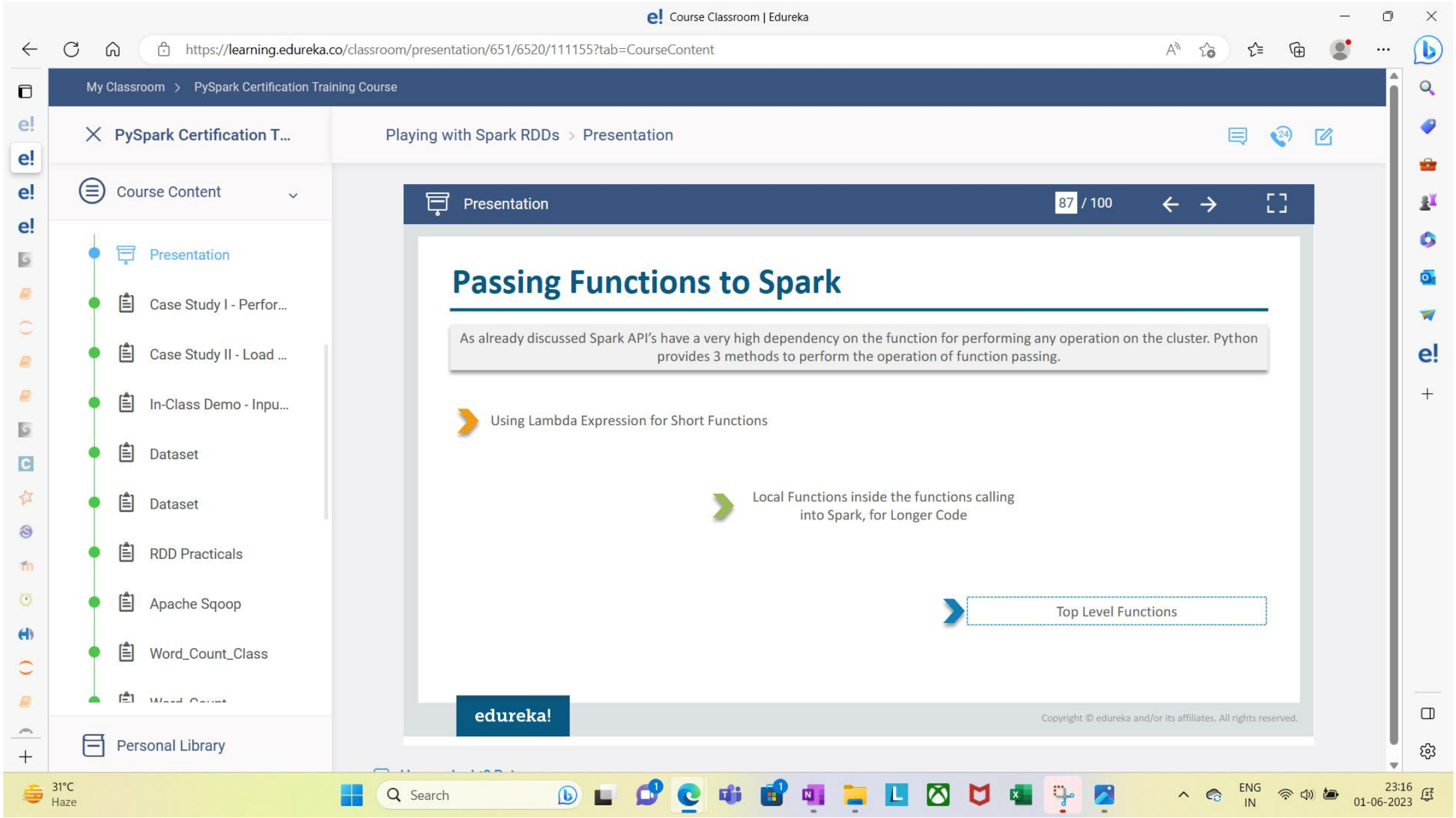
- In Spark, most of the Transformation operations and a considerable number of Action Operations depend on the Function Passing
- Function Passing performs an incomprehensible role in performing Data Computation and defining Program Flow



edureka!


Course Classroom | Edureka
https://learning.edureka.co/classroom/presentation/651/6520/111155?tab=CourseContent
My Classroom > PySpark Certification Training Course
PySpark Certification T...
Playing with Spark RDDs > Presentation
Course Content
Presentation
Case Study I - Perfor...
Case Study II - Load ...
In-Class Demo - Inpu...
Dataset
Dataset
RDD Practicals
Apache Sqoop
Word_Count_Class
Personal Library
Presentation
85 / 100
Passing Functions to Spark
As already discussed Spark API's have a very high dependency on the function for performing any operation on the cluster. Python provides 3 methods to perform the operation of function passing.
Using Lambda Expression for Short Functions
edureka!
Copyright © edureka and/or its affiliates. All rights reserved.
31°C
Haze
Search
ENG
IN
23:16
01-06-2023


Course Classroom | Edureka
https://learning.edureka.co/classroom/presentation/651/6520/111155?tab=CourseContent
My Classroom > PySpark Certification Training Course
PySpark Certification T...
Playing with Spark RDDs > Presentation
Course Content
Presentation
Case Study I - Perfor...
Case Study II - Load ...
In-Class Demo - Inpu...
Dataset
Dataset
RDD Practicals
Apache Sqoop
Word_Count_Class
Personal Library
Presentation
86 / 100
Passing Functions to Spark
As already discussed Spark API's have a very high dependency on the function for performing any operation on the cluster. Python provides 3 methods to perform the operation of function passing.
Using Lambda Expression for Short Functions
Local Functions inside the functions calling into Spark, for Longer Code
edureka!
Copyright © edureka and/or its affiliates. All rights reserved.
31°C Haze
Search
ENG IN
23:16 01-06-2023


Course Classroom | Edureka
https://learning.edureka.co/classroom/presentation/651/6520/111155?tab=CourseContent
My Classroom > PySpark Certification Training Course
PySpark Certification T...
Playing with Spark RDDs > Presentation
Course Content
Presentation
Case Study I - Perfor...
Case Study II - Load ...
In-Class Demo - Inpu...
Dataset
Dataset
RDD Practicals
Apache Sqoop
Word_Count_Class
Personal Library
Presentation
87 / 100
Passing Functions to Spark
As already discussed Spark API's have a very high dependency on the function for performing any operation on the cluster. Python provides 3 methods to perform the operation of function passing.
Using Lambda Expression for Short Functions
Local Functions inside the functions calling into Spark, for Longer Code
Top Level Functions
edureka!
Copyright © edureka and/or its affiliates. All rights reserved.
31°C Haze
Search
ENG IN
23:16 01-06-2023

Presentation 88 / 100

Using Lambda Expression for Short Function

```
lines = sc.textFile("README.md")
pythonLines = lines.filter(lambda line: "Python" in line)
pythonLines.first()
```

1. Lambda expressions for simple functions that can be written as an expression
2. Lambdas do not support multi-statement functions or statements that do not return a value

edureka!

Presentation 89 / 100

Using Local Functions

```
def containsError(s):
    return "error" in s
word = rdd.filter(containsError)
```

1. We pass the locally created functions inside the pre-defined functions as parameters to create RDDs
2. One of the major characteristics of local functions is that they are not interpreted with the other Top-Level Functions
3. These functions are at non-zero indentation level

edureka!

Presentation 90 / 100

Using Top Level Functions

```
def __init__(self):
        self.field = "Hello"
class MyClass(object):
def doStuff(self, rdd):
                return rdd.map(lambda s: self.field + s)
```

1. We can very clearly see that both the function and the class are at same indentation level and therefore will be executed at the same time

edureka!

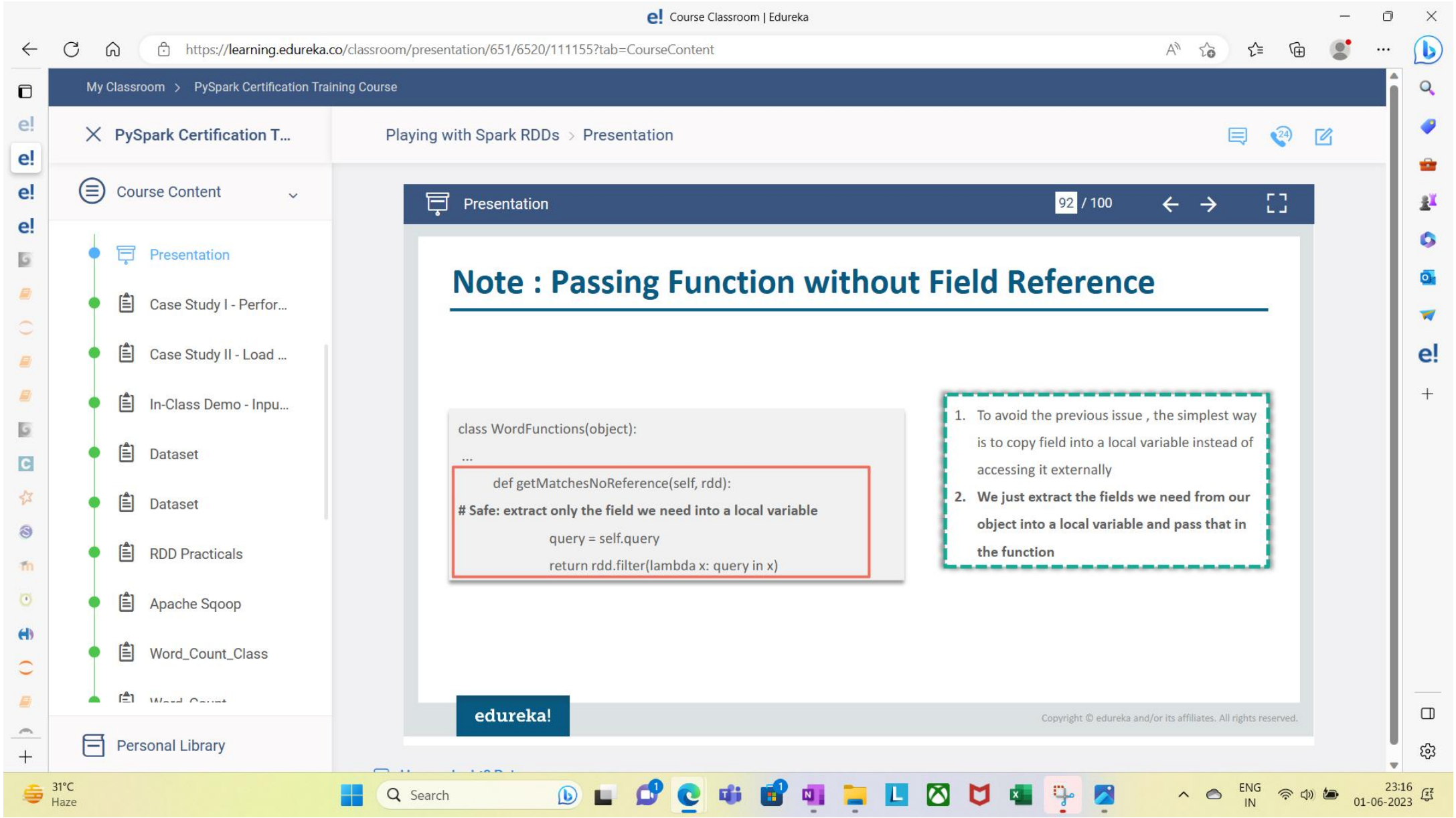
Presentation 91 / 100

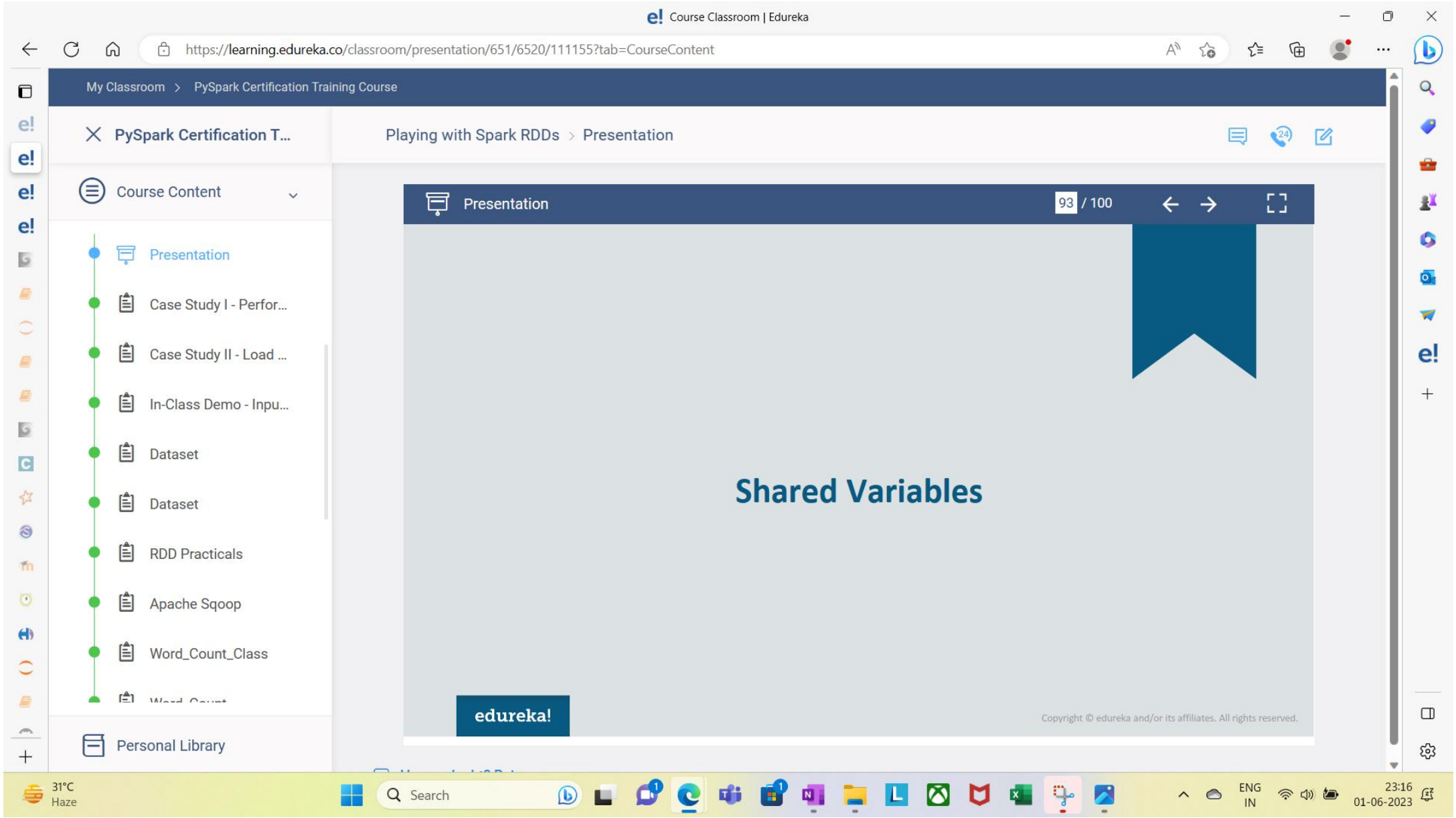
Note : Passing Function with Field Reference

```
class SearchFunctions(object):
    def __init__(self, query):
        self.query = query
    def isMatch(self, s):
        return self.query in s
    def getMatchesFunctionReference(self, rdd):
# Problem: references all of "self" in "self.isMatch"
        return rdd.filter(self.isMatch)
    def getMatchesMemberReference(self, rdd):
# Problem: references all of "self" in "self.query"
        return rdd.filter(lambda x: self.query in x)
```

1. When you pass a function that is the member of an object, or contains references to fields in an object (e.g., self.field), Spark sends the entire object to worker nodes, which can be much larger than the bit of information you need.
2. **Sometimes this can also cause your program to fail, if your class contains objects that Python can't figure out how to pickle.**

edureka!

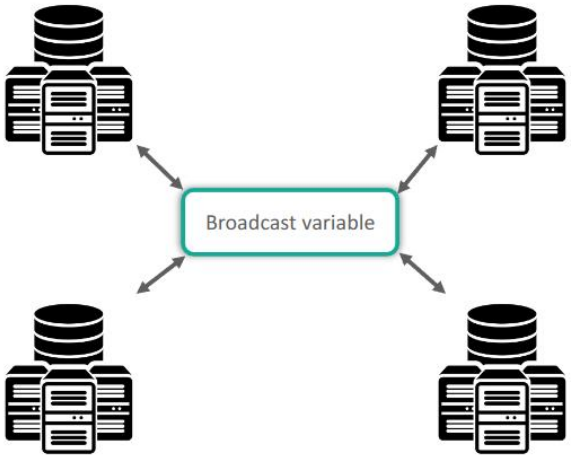

Course Classroom | Edureka
https://learning.edureka.co/classroom/presentation/651/6520/111155?tab=CourseContent
My Classroom > PySpark Certification Training Course
PySpark Certification T...
Playing with Spark RDDs > Presentation
Course Content
Presentation
Case Study I - Perfor...
Case Study II - Load ...
In-Class Demo - Inpu...
Dataset
Dataset
RDD Practicals
Apache Sqoop
Word_Count_Class
Personal Library
Presentation
92 / 100
Note : Passing Function without Field Reference
class WordFunctions(object):
...
def getMatchesNoReference(self, rdd):
# Safe: extract only the field we need into a local variable
query = self.query
return rdd.filter(lambda x: query in x)
1. To avoid the previous issue , the simplest way is to copy field into a local variable instead of accessing it externally
2. We just extract the fields we need from our object into a local variable and pass that in the function
edureka!
Copyright © edureka and/or its affiliates. All rights reserved.
31°C Haze
Search
ENG IN
23:16 01-06-2023


Course Classroom | Edureka
https://learning.edureka.co/classroom/presentation/651/6520/111155?tab=CourseContent
My Classroom > PySpark Certification Training Course
PySpark Certification T...
Playing with Spark RDDs > Presentation
Course Content
Presentation
Case Study I - Perfor...
Case Study II - Load ...
In-Class Demo - Inpu...
Dataset
Dataset
RDD Practicals
Apache Sqoop
Word_Count_Class
Personal Library
Presentation
93 / 100
Shared Variables
edureka!
Copyright © edureka and/or its affiliates. All rights reserved.
31°C
Haze
Search
ENG
IN
23:16
01-06-2023

Shared Variables – Broadcast Variable

What are Broadcast Variables?

1. Broadcast variables in Apache Spark is a mechanism for sharing variables across executors that are meant to be read-only.
2. Without broadcast variables these variables would be shipped to each executor for every transformation and action, and this can cause network overhead.
3. However, with broadcast variables, they are shipped once to all executors and are cached for future reference.

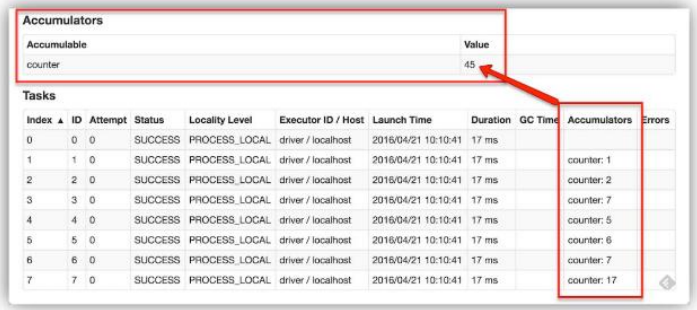


edureka!

Shared Variable – Accumulators

What are Accumulators ?

1. Accumulators are variables that are used for aggregating information across the executors.
2. For example, this information can pertain to data or API diagnosis like how many records are corrupted or how many times a particular library API was called.
3. Spark natively supports accumulators of numeric types, and programmers can add support for new types
4. As a user, you can create named or unnamed accumulators

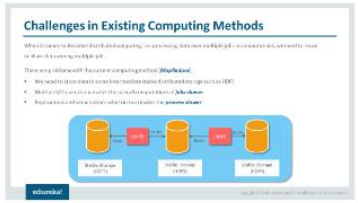


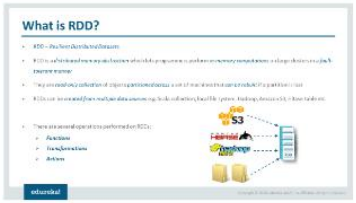
Note : Tracking accumulators in the UI can be useful for understanding the progress of running stages although this is not yet supported in Python

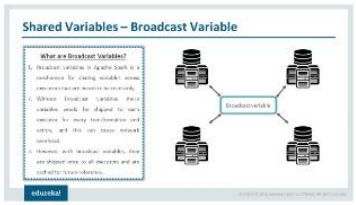
edureka!

Presentation 96 / 100

Summary







edureka!