TRENDY TECH

UPLIFT YOUR CAREER!

Assignment

Week10: Apache Spark - in Depth

Apache Spark Assignment

Finding Out Popular Courses for a Online Training Company

Problem Statement:

We have been approached by an Online Training Company which provides training videos.The company provides different online courses to the users and each course comprises of multiple chapters.Courses have been assigned **courseIds** and Chapters in a course have **chapterIDs**.Note that chapterIds are unique.No two courses can have same chapterId.Each Course has a **title** also. Also, Users viewing the courses are given **UserIds.**

The company wants us to prepare an **Apache Spark** based report which will enable them to figure out which are the **most popular courses** on their training portal.

The popularity of a course will depend on the total score it receives.The score which a course will get will be based on few business rules which are explained later in the document in detail.Please Refer the **Business Rules**** Section.To give a brief overview, consider if a course has 20 chapters and a user views 18 chapters out of those, then the user has watched 90% of the course and the course will get a good score of 10 say.Like that each course can be watched by multiple users and scores will accumulate for that course.

Final Ranking Chart Report will look like below, Starting with most popular course on the top.

Score	Title
X1	Y1
X2	Y2
….	….
….	….

Dataset Details:

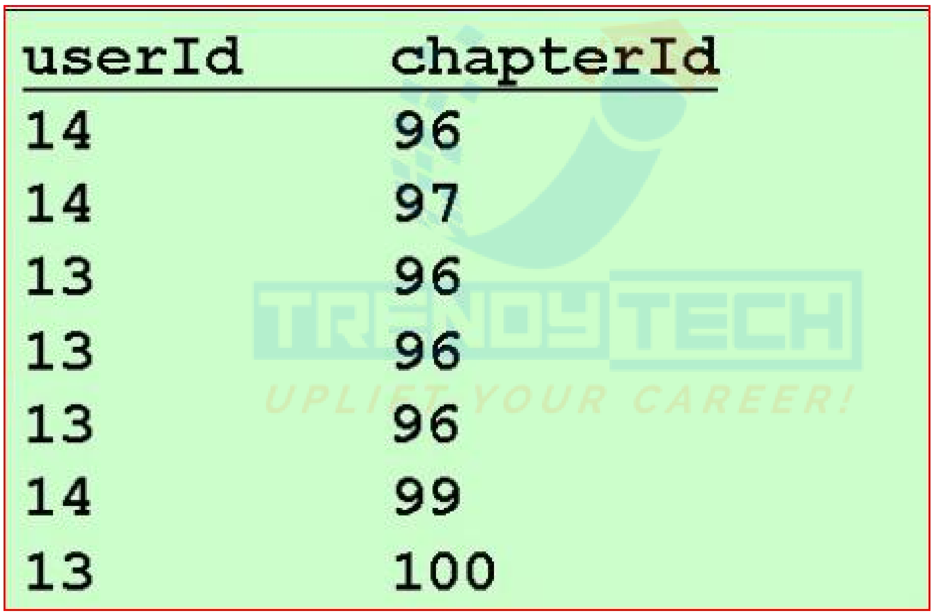
Dataset1:

We have three csv files **views-1.csv**, **views-2.csv** and **views-3.csv**
Also we have a **viewsHeader.csv** file which has the header information:

userId, chapterId, dateAndTime

This will be a log of viewings, showing which user has viewed which chapter. These chapterIds belong to courses. **Also chapterIds are unique.**

Here is the sample raw data:

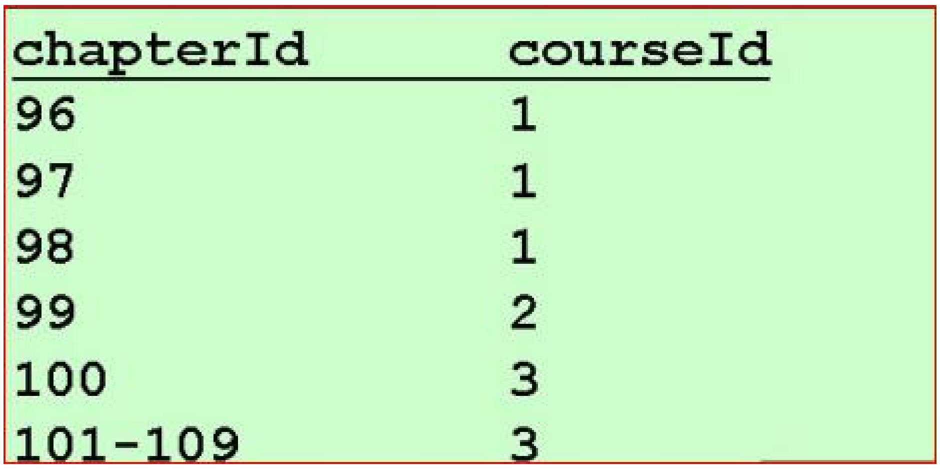


userId	chapterId
14	96
14	97
13	96
13	96
13	96
14	99
13	100

But as we can see from this raw data ,there is no reference to courses in this raw data.

Dataset 2: So we have a much smaller dataset, **chapters.csv** which is a mapping of chapterIds to courseId.

Sample Raw Data:



chapterId	courseId
96	1
97	1
98	1
99	2
100	3
101-109	3

Dataset 3:

There is another dump file (**titles.csv**) containing courseIds against their titles - we will use this at the end to print the results,so that we also get the title of each course when we get the popular courses result in the end.

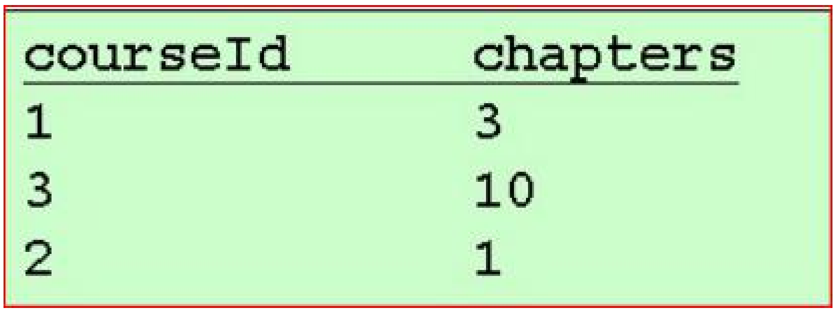
Note: For every exercise ,the expected output of the sample raw data is provided .But this is not the actual output. This has been provided for understanding purpose only.
You will work on actual data and output will be based on the csv files provided to you.

Solve the following Exercises:

Exercise 1:

Find Out how many Chapters are there per course.

Expected Output of Sample Raw Data:



courseId	chapters
1	3
3	10
2	1

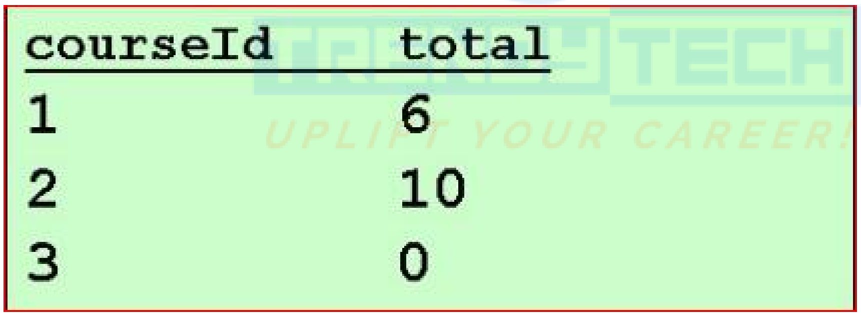
Exercise 2: **Your job is to produce a ranking chart detailing based on which are the most popular courses by score.**

****Business Rules:**

Scores which each courseId gets, will be based on few rules: Like ,If a user sticks through most of the course, that's more deserving of "points" than if someone bails out just a quarter way through the course. Following is the scoring system algorithm followed:

- If a user watches more than 90% of the course, the course gets 10 points
- If a user watches > 50% but <90% , it scores 4
- If a user watches > 25% but < 50% it scores 2
- Less than 25% is no score

Expected Output of Sample Raw Data:



courseId	total
1	6
2	10
3	0

Exercise 3: As a final touch, get the titles using a final (small data) join. You can then sort by total score (descending) to get the results in a pleasant format with most popular score on top.

**Hints

<u>Exercise 1:</u>

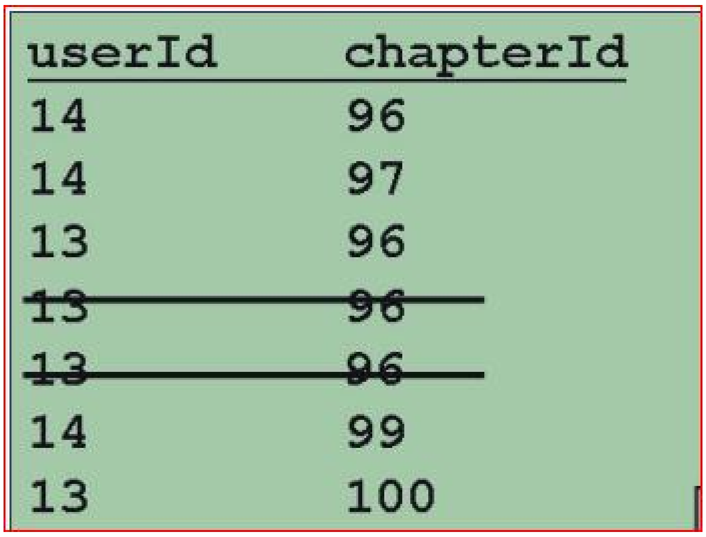
Build an RDD containing a key of courseId together with the number of chapters on that course. You can use a map transformation here.

courseId	count
1	1
1	1
2	1
3	1
3	1

<u>Exercise 2:</u>

Step 1: Removing Duplicate Views .

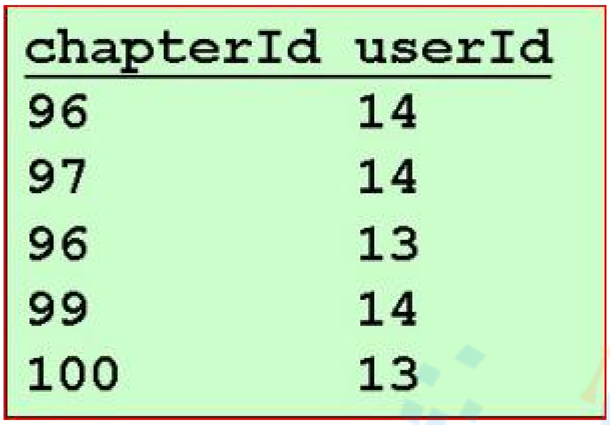
the same user watching the same chapter doesn't count, so we can call distinct to remove duplications.



userId	chapterId
14	96
14	97
13	96
~~13~~	~~96~~
~~13~~	~~96~~
14	99
13	100

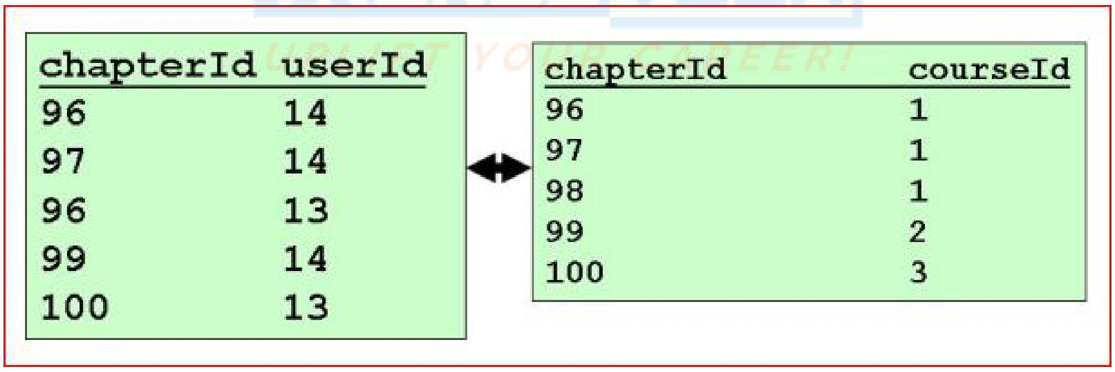
Step 2: Joining to get Course Id in the RDD

This isn't very meaningful as it is - we know for example that user 14 has watched 96 and 97, but are they from the same course, or are they different courses???? We need to join the data together. As the common column in both RDDs is the chapterId, we need both Rdds to be keyed on that.



chapterId	userId
96	14
97	14
96	13
99	14
100	13

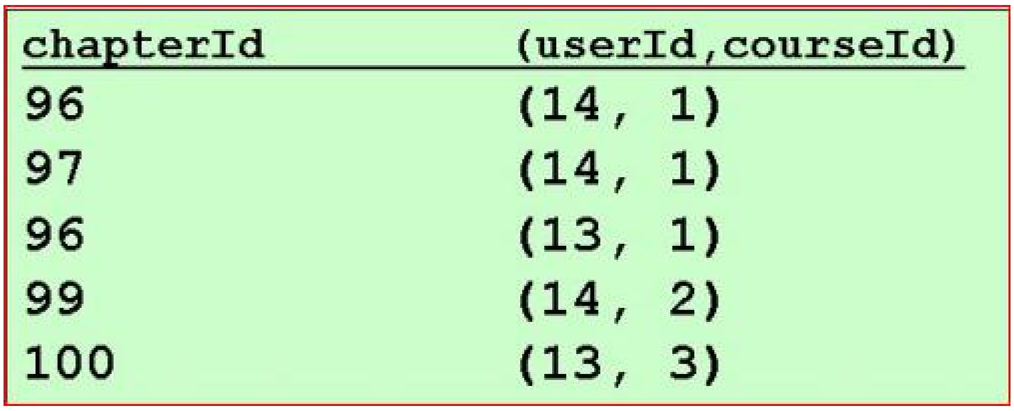
You need to do a map to switch the key and the value. Now that you have two RDDs, each keyed on chapterId, you can join them together.



chapterId	userId
96	14
97	14
96	13
99	14
100	13

chapterId	courseId
96	1
97	1
98	1
99	2
100	3

And now after joining you will have:

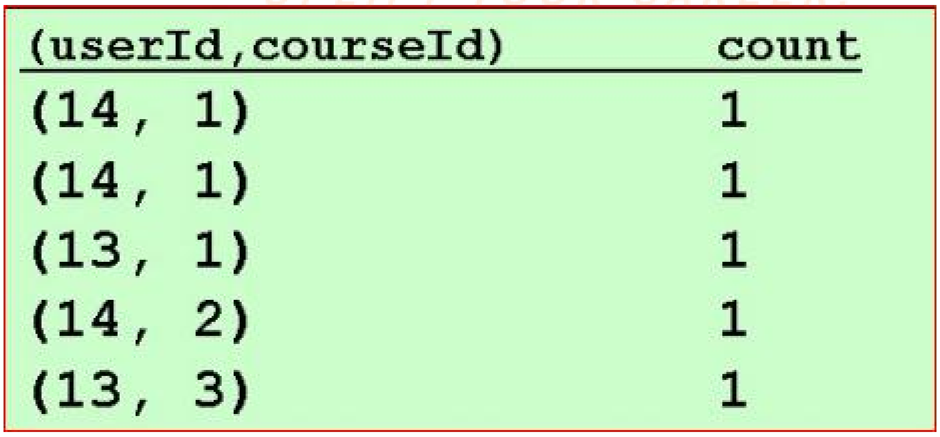


chapterId	(userId,courseId)
96	(14, 1)
97	(14, 1)
96	(13, 1)
99	(14, 2)
100	(13, 3)

Step 3: Drop the Chapter Id

As each "row" in the RDD now represents "a chapter from this course has been viewed", the chapterIds are no longer relevant. You can get rid of them at this point - this will avoid dealing with tricky nested tuples later on. At the same time, given we're counting how many chapters of a course each user watched, we will be counting shortly, so we can drop a "1" in for each user/course combination.

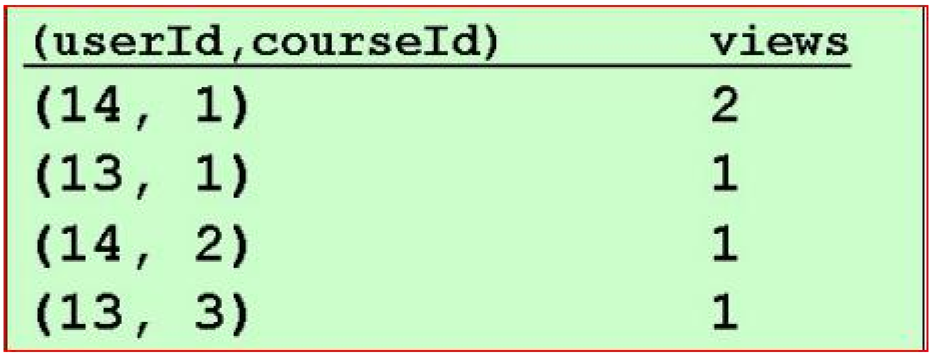
Expected RDD after this step:



(userId,courseId)	count
(14, 1)	1
(14, 1)	1
(13, 1)	1
(14, 2)	1
(13, 3)	1

Step 4 - Count Views for User/Course

We can now run a reduce as usual..

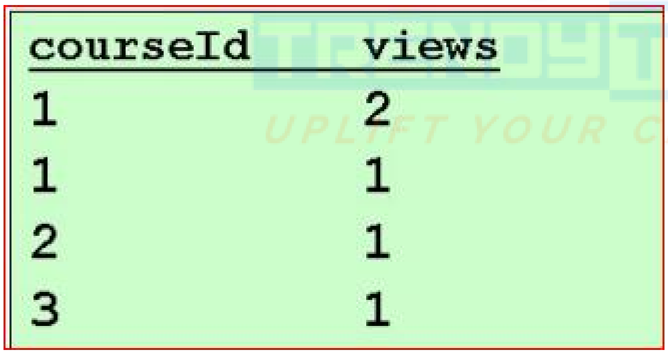


(userId,courseId)	views
(14, 1)	2
(13, 1)	1
(14, 2)	1
(13, 3)	1

We can read these results as "user 14 watched 2 chapters of 1; user 13 watched 1 chapter of course 1; user 14 watched 2 chapters of course 2. etc".

Step 5 - Drop the userId

The user id is no longer relevant. We've needed it up until now to ensure that for example, 10 users watching 1 chapter doesn't count the same as 1 user watching 10 chapters! Another map transformation should get you...

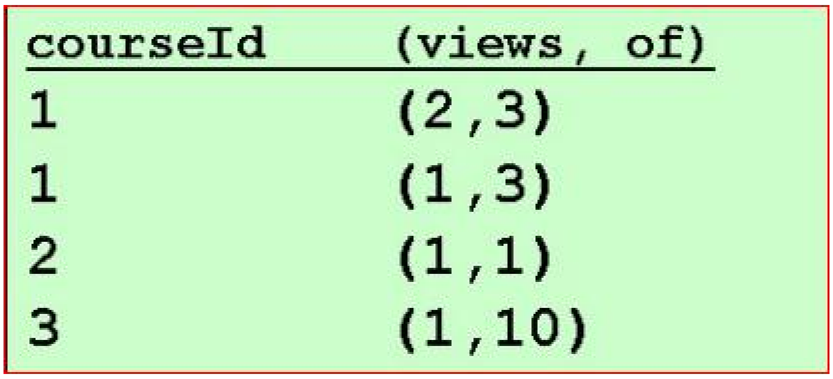


courseId	views
1	2
1	1
2	1
3	1

To rephrase the previous step, we now read this as "someone watched 2 chapters of course 1. Somebody different watched 1 chapter of course 1; someone watched 1 chapter of course 2, etc".

Step 6: of how many chapters?

The scoring is based on what percentage of the total course they watched. So we will need to get the total number of chapters for each course into our RDD:

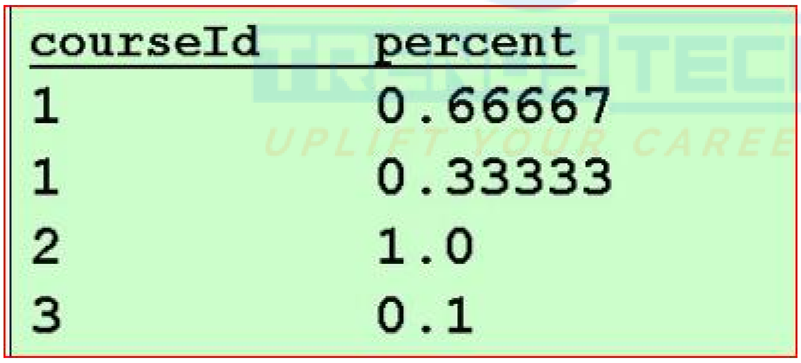


courseId	(views, of)
1	(2,3)
1	(1,3)
2	(1,1)
3	(1,10)

"Someone watched 2 chapters of 3. Somebody different watched 1 chapter of 3".

Step 7: Convert to percentages

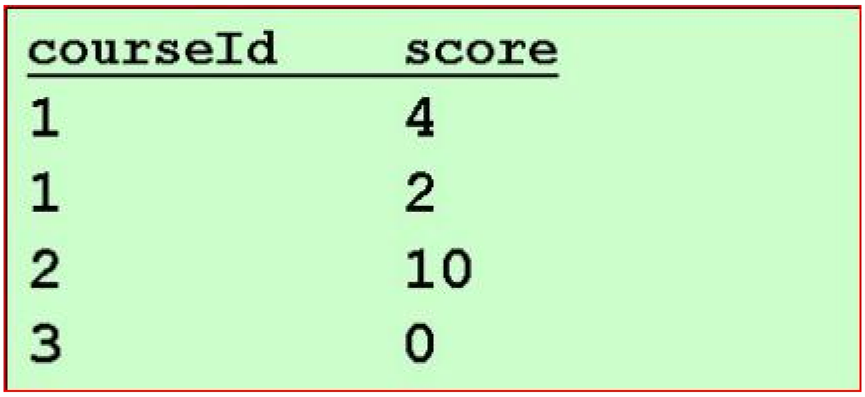
This should be an easy mapping.



courseId	percent
1	0.66667
1	0.33333
2	1.0
3	0.1

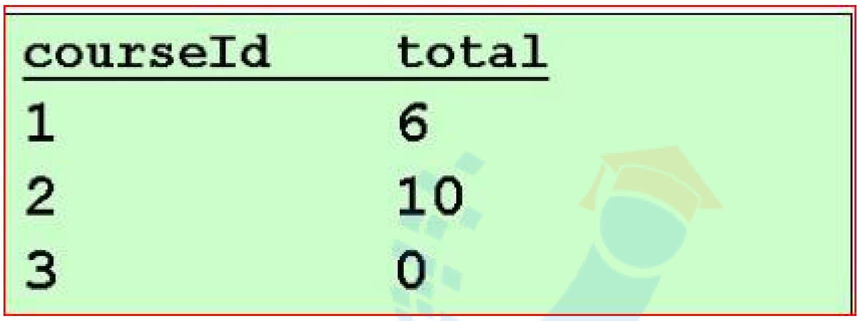
Step 8: Convert to scores

As described earlier, percentages are converted to scores:



courseId	score
1	4
1	2
2	10
3	0

Step 9: Add up the total scores



courseId	total
1	6
2	10
3	0

Exercise 3:

You have to associate titles with the course.Use Inner Join.Then later as a finishing job you will not require **courseIds** any more as the **titles** are available.You can sort the data by most popular course to display data in format - **Score, Title** .

****

.