## C3\_W2 quiz

100%

1/ If you want to merge two splits 'train' and 'test' together using Splits API, how would you be able to do so?	
0	tfds.load('mnist', split = pd.concat('train', 'test'))
0	tfds.load('mnist',split = np.concat('train+test'))
•	tfds.load('mnist', split = 'train + test') → Passing both train and test in the string is the proper way to get both the splits.
0	tfds.load('mnist', merge = 'train+test')
to cı	he MNISTv3 dataset supports the Splits API. The train split has 70000 records in it. If you just want reate a subsplit of the first 7000 records and want to use the python slicing notation instead of Splits, what would be the answer?
0	tfds.load('mnist:3.*.*', split='train[7000:]')
0	tfds.load('mnist:3.*.*', subsplit='train[:7000]')
C the	Read the entire train split, create a new dataset, iterate over the first 7000 of the 70000, and copy records one-by-one to the new dataset.
• inde	tfds.load('mnist:3.*.*', split='train[:7000]') ->train[:7000] technically takes records from 0 to 6999 ex value.
	you want a subsplit of the first 10% of the MNISTv3 training records, what would the code look like g the Splits API?
0	tfds.load('mnist:3.*.*', split='train[:10%]')
0	tfds.load('mnist:3.*.*', split='train[10%:]')
•	tfds.load('mnist:3.*.*', subsplit='train[:10%]') → 'train[:10%]'in string format represents that we want the first 10% of the records from the train split.
0	tfds.load('mnist:3.*.*', subsplit='train[10%:]')
$\underline{4/}$ How many validation splits will this code generate? val_ds = tfds.load('mnist:3.*.*', split = ['train[{}%:{}%]'.format(int(k/4),int((k+40)/4)) for k in range(0, $\frac{400}{400}$ , $\frac{40}{40}$ )]) $\rightarrow \frac{10}{10}$	
<u>Hint</u> : As k is incremented by 40, you get the values like $(0,40)$ , $(40,80)$ until the last one, $(360:400)$ . Dividing each value by 4 as you have $(k/4,(k+40)/4)$ , it will get converted to $[0\%:10\%]$ , $[10\%:20\%]$ $[90\%:100\%]$ which is 10 splits. Note that the indices should be integers so the $int()$ function was used inside the list comprehension to do that conversion.	
<u>5/</u> True or False : The TFRecord shards are only created for the training data.	
•	False →If your validation and test data size is bigger, they also can get sharded.

6/ Which of the following could be used to investigate how the TFRecords files look like?

filename = "your\_tf\_record\_file"raw\_file = tf.data.TFRecordDataset (filename)for raw\_record in raw\_file.take(1): print(repr(raw\_record))

## Hint:

tf.data.TFRecordDataset is used to read your raw TFRecord files and convert it to an object. take(1) gives you 1 record from the raw\_file TFRecordDataset object. repr() function returns a printable representation of the given tfrecord object.

<u>7/</u> Below is an example of how raw TFRecord files look like when you properly read and print them with TFRecordDataset methods.

To parse them into proper format, which of the following options need to be implemented?

- Creating a feature description dictionary
- →This is Step 1. You need to define your feature descriptions properly based on the dataset and its metadata. This Is necessary here because datasets use graph-execution, and need description to build their shape and type signature
- Apply the parsing function to each item in the dataset using the tf.data.Dataset.map method.
- →This is Step 3. You need to parse each file individually and map them to the parsing function to create clean, readable standard tensors.
- Apply the parsing function to each item in the dataset using the keras.dataset.map method
- Creating a parsing function using tfds.load()
- Creating a parsing function using tf.io.parse\_single\_example()
- → This is Step 2. *tf.io.parse\_single\_example()* is used to parse examples one by one as the raw TFRecordDataset files contain serialized tf.train.Example objects.

You can also use *tf.parse* example to parse the whole batch at once.