(pyspark-env) student@paka:~$ cat p1.py

from pyspark.sql import SparkSession

spark = SparkSession.builder.appName('demo1').getOrCreate()

data = [("pA",101),("pB",102),("pC",103)]

columns = ["pName","pID"]

df = spark.createDataFrame(data,columns)

df.show()

(pyspark-env) student@paka:~$ cat p2.py

from pyspark.sql import SparkSession

spark = SparkSession.builder.appName('demo1').getOrCreate()

#data = [("pA",101),("pB",102),("pC",103)]

#columns = ["pName","pID"]

#df = spark.createDataFrame(data,columns)

#df.show()

df = spark.read.csv('prod.csv',header=True)

df.show()

df.show(2)

(pyspark-env) student@paka:~$ cat p3.py

from pyspark.sql import SparkSession

spark = SparkSession.builder.appName('demo1').getOrCreate()

data = [("pA",101),("pB",102),("pC",103)]

columns = ["pName","pID"]

df1 = spark.createDataFrame(data,columns)

df2 = spark.read.csv('prod.csv',header=True)

df1.printSchema()

df2.printSchema()

(pyspark-env) student@paka:~$ cat p4.py

from pyspark.sql import SparkSession

spark = SparkSession.builder.appName('demo1').getOrCreate()

df = spark.read.csv('prod.csv',header=True)

df.select("productName").show()

df.filter(df.productCost >1250).show()

df.groupBy("productID").count().show()

(pyspark-env) student@paka:~$ cat p5.py

from pyspark.sql import SparkSession

spark = SparkSession.builder.appName('demo1').getOrCreate()

df = spark.read.csv('prod.csv',header=True)

df.select("productName").show()

df.filter(df.productCost >1250).show()

df.groupBy("productID").count().show()

df.createOrReplaceTempView("prod")

result = spark.sql("select \*from prod where productCost >3000")

result.show()

(pyspark-env) student@paka:~$ cat p6.py

from pyspark.sql import SparkSession

spark = SparkSession.builder.appName("demo2").getOrCreate()

sc = spark.sparkContext

rdd = sc.parallelize([10,20,30,40,50])

print("partition count:",rdd.getNumPartitions())

rdd = sc.parallelize([10,20,30,40,50],4)

print("partition count:",rdd.getNumPartitions())

(pyspark-env) student@paka:~$ cat p7.py

from pyspark.sql import SparkSession

spark = SparkSession.builder.appName("demo2").getOrCreate()

sc = spark.sparkContext

# create an RDD with 2 partitions

rdd = sc.parallelize([10,20,30,40,50,60],2)

# To get partitions count

print(rdd.getNumPartitions())

rdd1 = rdd.map(lambda a:a+100)

########################### immutable - not changable - initialize to another rdd

print(rdd1.collect()) # collect and display results

(pyspark-env) student@paka:~$ cat p8.py

from pyspark.sql import SparkSession

spark = SparkSession.builder.appName("demo2").getOrCreate()

sc = spark.sparkContext

# create an RDD with 2 partitions

rdd = sc.parallelize([10,20,30,40,50,60],2)

mapped\_obj = rdd.map(lambda a:[a,a+100])

print(mapped\_obj.collect())

print('') # empty line

rdd = sc.parallelize([10,20,30,40,50,60],2)

flat\_mapped\_obj = rdd.flatMap(lambda a:[a,a+100])

print(flat\_mapped\_obj.collect())

(pyspark-env) student@paka:~$ cat p9.py

from pyspark.sql import SparkSession

spark = SparkSession.builder.appName("demo2").getOrCreate()

sc = spark.sparkContext

rdd1 = sc.parallelize([1,2,3])

rdd2 = sc.parallelize([4,5,6])

rdd\_result = rdd1.union(rdd2)

print(rdd\_result.collect())

print("")

rdd1 = sc.parallelize([1,2,3])

rdd2 = sc.parallelize([3,5,6])

rdd\_result = rdd1.union(rdd2)

print(rdd\_result.collect())

(pyspark-env) student@paka:~$ cat p10.py

from pyspark.sql import SparkSession

spark = SparkSession.builder.appName("demo2").getOrCreate()

sc = spark.sparkContext

rdd = sc.parallelize([("row1",10),("row2",20),("row3",30),("row1",40),("row2",50)])

print(rdd.reduceByKey(lambda a,b:a+b).collect())