**1)It is configured in hadoop/conf/core-site.xml**

**hdfs://<x.x.x.x>:1234/test/</user/logger/python/logs>/test.log**

**2)**

**from pyspark.sql impot Row**

**from pyspark.sql import HiveContext**

**sqlContext = HiveContext(sc)**

**spark.sql(“SET hive.merge.mapredfiles = false”)**

**spark.sql (“SET hive.merge.small.files.avgsize = 16000000”)**

**spark.sql(“SET hive.execution.engine = mr”)**

**3)**

**#Using Pandas**

**Assuming some data**

**import pandas as pd**

**data = {'Name': ['Sana', 'Arun', 'Anusha', 'Priya'],**

**‘email': [‘hctambadgmail.com’, ‘ xx@gmail.com’, ‘yy@gmail.com’],**

**'Qualification': ['MS', 'BE’, 'Msc', 'MS']}**

**data[:3]**

**data{email\_valid\_flag} = (“valid email id”)**

**if flag**

**print(‘valid email id’)**

**else**

**print ‘not valid’)**

**# Using Spark Dataframe**

**from .pyspark.sql.functions import col**

**from pyspark.sql.functions import udf**

**df = {'Name': ['Sana', 'Arun', 'Anusha', 'Priya'],**

**‘email': [‘hctambadgmail.com’, ‘ xx@gmail.com’, ‘yy@gmail.com’],**

**'Qualification': ['MS', 'BE’, 'Msc', 'MS']}**

**#inserting new column to existing table**

**df = df.insert[3, 'email\_valid\_flag')**

**df = df.filter(col("email\_valid\_flag"),if not valid mail id)**

**print(df)**

**4)**

**#Step1: Convert a dataframe to list of dicts**

**from pyspark.sql.functions import col**

**from pyspark.sql.functions import udf**

**df.T.to\_dict().value()**

**#Step2: read the same into a another dataframe with prefix of "a\_" to the column**

**df = pd.DataFrame({“A”:[1,2,3,4], “B”:[5,3,2,1]})**

**df = df.add.prefix(‘a\_’)**

**df = pd.DataFrame(data)**

**df**

**5)**

**from pyspark­\_seq\_id**

**df\_len = 50**

**freq = 1**

**ref= spark.range(5, df\_len, freq). toDF(“id”)**

**ref.show(50)**

**#There is an another example using monotonically increasing method**

**from pyspark.sql.function import desc, row\_number,monotonically\_increasing\_id**

**df\_with\_seq\_id= df.withColumn(“index\_coloumn\_name”,**

**row\_number().over (Window.orderBy(montomically\_increased\_id()))-1)**

**7)**

**Import pandas as pd**

**Import nump as ny**

**df = pd.dataframe ({‘A’: [‘Ishan’, ‘Usha’, ‘Rohan’],**

**‘B’: [‘Masters’, ‘Graduate’, ‘Masters’],**

**‘C’: [‘25’, ‘23’, ‘27] })**

**df.show()**

**8)**

**import sqlite3**

**import pandas as pd**

**cnx= sqlite3.connect(‘file.Data’)**

**df= pd.read\_sql\_query(“SELECT \* FROM table\_name”, cnx)**

**9)**

**import pandas, numpy**

**test\_read = pd.read\_csv(‘pandas\_text\_read.csv’,**

**delimiter = “,” , names = [‘user id’, ‘name’ , ‘country’] )**

**#Now Converting above pandadataframe into spark dataframe**

**Import pyspark**

**From pyspark.sql import SparkSession**

**Import panda as pd**

**Spark = SparkSession,builder,appName(‘pandasToSparkDF),getorCreate()**

**pdDF = pd.read\_csv(test\_read)**

**pdDF**

**from pyspark.sql.types import\***

**myschema = StructType ( [“Col1”, longType(),True)\**

**“Col2”, IntegerType(), True])**

**df = spark.createDataFrame(pdDF,Scheam=myShema)**

**type(df)**

**pyspark,sql.dataframe.DataFrame**

**10)**

**from pyspark.sql import SQLCONTEXT**

**from pyspark.sql.types import**

**sqlContext = SQLContext(sc)**

**Data= sqlcontext.read.csv(“/path”, header= True/False, Schema= “infer”, sep =”delimeter”)**

**Data= Data.withColumn(“Country”, “Code goes here”)**

**Data.printSchema()**