```
pip install pyspark
```

```
Requirement already satisfied: pyspark in /usr/local/lib/python3.10/dist-packages (3.5.1)
Requirement already satisfied: py4j==0.10.9.7 in /usr/local/lib/python3.10/dist-packages (from pyspark) (0.10.9.7)
```

```
from pyspark.sql import SparkSession

spark = SparkSession.builder.getOrCreate()
```

```
from datetime import datetime, date
import pandas as pd
from pyspark.sql import Row

df = spark.createDataFrame([
    Row(a=1, b=2., c='string1', d=date(2000, 1, 1), e=datetime(2000, 1, 1, 12, 0)),
    Row(a=2, b=3., c='string2', d=date(2000, 2, 1), e=datetime(2000, 1, 2, 12, 0)),
    Row(a=4, b=5., c='string3', d=date(2000, 3, 1), e=datetime(2000, 1, 3, 12, 0))
])
df
```

```
DataFrame[a: bigint, b: double, c: string, d: date, e: timestamp]
```

```
df = spark.createDataFrame([
    (1, 2., 'string1', date(2000, 1, 1), datetime(2000, 1, 1, 12, 0)),
    (2, 3., 'string2', date(2000, 2, 1), datetime(2000, 1, 2, 12, 0)),
    (3, 4., 'string3', date(2000, 3, 1), datetime(2000, 1, 3, 12, 0))
], schema='a long, b double, c string, d date, e timestamp')
df
```

```
DataFrame[a: bigint, b: double, c: string, d: date, e: timestamp]
```

```
pandas_df = pd.DataFrame({
    'a': [1, 2, 3],
    'b': [2., 3., 4.],
    'c': ['string1', 'string2', 'string3'],
    'd': [date(2000, 1, 1), date(2000, 2, 1), date(2000, 3, 1)],
    'e': [datetime(2000, 1, 1, 12, 0), datetime(2000, 1, 2, 12, 0), datetime(2000, 1, 3
})
df = spark.createDataFrame(pandas_df)
df
```

```
DataFrame[a: bigint, b: double, c: string, d: date, e: timestamp]
```

```
# All DataFrames above result same.
df.show()
df.printSchema()
```

```
+---+---+-------+----------+-------------------+
|  a|  b|      c|         d|                  e|
+---+---+-------+----------+-------------------+
|  1|2.0|string1|2000-01-01|2000-01-01 12:00:00|
|  2|3.0|string2|2000-02-01|2000-01-02 12:00:00|
|  3|4.0|string3|2000-03-01|2000-01-03 12:00:00|
+---+---+-------+----------+-------------------+

root
 |-- a: long (nullable = true)
 |-- b: double (nullable = true)
 |-- c: string (nullable = true)
 |-- d: date (nullable = true)
 |-- e: timestamp (nullable = true)
```

```
df.show(1)
```

```
+---+---+-------+----------+-------------------+
|  a|  b|      c|         d|                  e|
+---+---+-------+----------+-------------------+
|  1|2.0|string1|2000-01-01|2000-01-01 12:00:00|
```

```
+---+---+-------+----------+-------------------+
only showing top 1 row
```

```
spark.conf.set('spark.sql.repl.eagerEval.enabled', True)
df
```

| a | b | c | d | e |
|---|---|---|---|---|
| 1 | 2.0 | string1 | 2000-01-01 | 2000-01-01 12:00:00 |
| 2 | 3.0 | string2 | 2000-02-01 | 2000-01-02 12:00:00 |
| 3 | 4.0 | string3 | 2000-03-01 | 2000-01-03 12:00:00 |

```
df.show(1, vertical=True)
```

```
-RECORD 0------------------
 a   | 1
 b   | 2.0
 c   | string1
 d   | 2000-01-01
 e   | 2000-01-01 12:00:00
only showing top 1 row
```

```
df.columns
```

```
['a', 'b', 'c', 'd', 'e']
```

```
df.printSchema()
```

```
root
 |-- a: long (nullable = true)
 |-- b: double (nullable = true)
 |-- c: string (nullable = true)
 |-- d: date (nullable = true)
 |-- e: timestamp (nullable = true)
```

```
df.select("a", "b", "c").describe().show()
```

```
+-------+---+---+-------+
|summary|  a|  b|      c|
+-------+---+---+-------+
|  count|  3|  3|      3|
|   mean|2.0|3.0|   NULL|
| stddev|1.0|1.0|   NULL|
|    min|  1|2.0|string1|
|    max|  3|4.0|string3|
+-------+---+---+-------+
```

```
df.collect()
```

```
[Row(a=1, b=2.0, c='string1', d=datetime.date(2000, 1, 1), e=datetime.datetime(2000, 1, 1, 12, 0)),
 Row(a=2, b=3.0, c='string2', d=datetime.date(2000, 2, 1), e=datetime.datetime(2000, 1, 2, 12, 0)),
 Row(a=3, b=4.0, c='string3', d=datetime.date(2000, 3, 1), e=datetime.datetime(2000, 1, 3, 12, 0))]
```

```
df.take(1)
```

```
[Row(a=1, b=2.0, c='string1', d=datetime.date(2000, 1, 1), e=datetime.datetime(2000, 1, 1, 12, 0))]
```

```
df.toPandas()
```

| | a | b | c | d | e |
|---|---|---|---|---|---|
| **0** | 1 | 2.0 | string1 | 2000-01-01 | 2000-01-01 12:00:00 |
| **1** | 2 | 3.0 | string2 | 2000-02-01 | 2000-01-02 12:00:00 |
| **2** | 3 | 4.0 | string3 | 2000-03-01 | 2000-01-03 12:00:00 |

```
df.a
```

```
Column<'a'>
```

```
from pyspark.sql import Column
from pyspark.sql.functions import upper

type(df.c) == type(upper(df.c)) == type(df.c.isNull())
```

```
True
```

```
df.select(df.c).show()
```

```
+-------+
|      c|
+-------+
|string1|
|string2|
|string3|
+-------+
```

Assign new `Column` instance.

```
df.withColumn('upper_c', upper(df.c)).show()
```

```
+---+---+-------+----------+-------------------+-------+
|  a|  b|      c|         d|                  e|upper_c|
+---+---+-------+----------+-------------------+-------+
|  1|2.0|string1|2000-01-01|2000-01-01 12:00:00|STRING1|
|  2|3.0|string2|2000-02-01|2000-01-02 12:00:00|STRING2|
|  3|4.0|string3|2000-03-01|2000-01-03 12:00:00|STRING3|
+---+---+-------+----------+-------------------+-------+
```

To select a subset of rows, use `DataFrame.filter()` .

```
df.filter(df.a == 1).show()
```

```
+---+---+-------+----------+-------------------+
|  a|  b|      c|         d|                  e|
+---+---+-------+----------+-------------------+
|  1|2.0|string1|2000-01-01|2000-01-01 12:00:00|
+---+---+-------+----------+-------------------+
```

```
import pandas as pd
from pyspark.sql.functions import pandas_udf

@pandas_udf('long')
def pandas_plus_one(series: pd.Series) -> pd.Series:
    return series + 1

df.select(pandas_plus_one(df.a)).show()
```

```
+------------------+
|pandas_plus_one(a)|
+------------------+
|                 2|
|                 3|
|                 4|
+------------------+
```

```
def pandas_filter_func(iterator):
    for pandas_df in iterator:
        yield pandas_df[pandas_df.a == 1]

df.mapInPandas(pandas_filter_func, schema=df.schema).show()

df = spark.createDataFrame([
    ['red', 'banana', 1, 10], ['blue', 'banana', 2, 20], ['red', 'carrot', 3, 30],
    ['blue', 'grape', 4, 40], ['red', 'carrot', 5, 50], ['black', 'carrot', 6, 60],
    ['red', 'banana', 7, 70], ['red', 'grape', 8, 80]], schema=['color', 'fruit', 'v1'
df.show()
```

```
df.groupby('color').avg().show()

def plus_mean(pandas_df):
    return pandas_df.assign(v1=pandas_df.v1 - pandas_df.v1.mean())

df.groupby('color').applyInPandas(plus_mean, schema=df.schema).show()
```

```
+-----+------+---+---+
|color| fruit| v1| v2|
+-----+------+---+---+
|black|carrot|  0| 60|
| blue|banana| -1| 20|
| blue| grape|  1| 40|
|  red|banana| -3| 10|
|  red|carrot| -1| 30|
|  red|carrot|  0| 50|
|  red|banana|  2| 70|
|  red| grape|  3| 80|
+-----+------+---+---+
```

```
df1 = spark.createDataFrame(
    [(20000101, 1, 1.0), (20000101, 2, 2.0), (20000102, 1, 3.0), (20000102, 2, 4.0)],
    ('time', 'id', 'v1'))

df2 = spark.createDataFrame(
    [(20000101, 1, 'x'), (20000101, 2, 'y')],
    ('time', 'id', 'v2'))

def merge_ordered(l, r):
    return pd.merge_ordered(l, r)

df1.groupby('id').cogroup(df2.groupby('id')).applyInPandas(
    merge_ordered, schema='time int, id int, v1 double, v2 string').show()

df.write.csv('foo.csv', header=True)
spark.read.csv('foo.csv', header=True).show()

df.write.parquet('bar.parquet')
spark.read.parquet('bar.parquet').show()
```

```
+-----+------+---+---+
|color| fruit| v1| v2|
+-----+------+---+---+
|  red|carrot|  5| 50|
|black|carrot|  6| 60|
|  red|banana|  7| 70|
|  red| grape|  8| 80|
|  red|banana|  1| 10|
| blue|banana|  2| 20|
|  red|carrot|  3| 30|
| blue| grape|  4| 40|
+-----+------+---+---+
```

```
df.write.orc('zoo.orc')
spark.read.orc('zoo.orc').show()

df.createOrReplaceTempView("tableA")
spark.sql("SELECT count(*) from tableA").show()

@pandas_udf("integer")
def add_one(s: pd.Series) -> pd.Series:
    return s + 1

spark.udf.register("add_one", add_one)
spark.sql("SELECT add_one(v1) FROM tableA").show()
```

```
from pyspark.sql.functions import expr

df.selectExpr('add_one(v1)').show()
df.select(expr('count(*)') > 0).show()
```