

Read / Write .csv

```
df =
(sqlContext.read.format("com.databricks.spark.csv")\
  .option("header", "true")\
  .option("inferSchema", "true")\
  .option("mode", "DROPPED")\
  .load("hdfs://file.csv"))
df.write.mode('overwrite').
  option("header", "true").
  csv("file://filename.csv")
```

Meta Data

```
df.printSchema()           df.count()
len(df.columns)           df.columns
df.dtypes
```

Arrange

```
df.withColumnRenamed("col", "newcol")
df.orderBy(['var1', 'var2'], ascending = [True,
False])
df.orderBy(df.var1, df.var2.desc())
```

Filter

```
df.filter(df.var > 10000)
df.select('col1', 'col2')
df[col list] # collist = ['var1', ...]
df.head(5) / df.show(5, truncate= True) /
df.take(5)
df.drop(' var')
df.distinct()
df.dropDuplicates()
df.dropna(subset=' var') / df.na.drop()
df.isNotNull()
df.var.isin("levell", "level2")
df.var.like("string")
df.var.startswith("m") / df.var.endswith("m")
df.sample(False, with replacement, 0.5 fraction,
12345 seed)
```

Useful Functions

Write Functions / UDF

```
from pyspark.sql.functions import udf
F1 = udf(lambda x: '-1' if condition else x,
String Type()) # NB return type
df = df.withColumn('new var', F1(df['in -
var']))
```

Applying Functions

```
df.select('val').map(lambda x: x*2)
```

Summarise

```
df.crosstab('col1', 'col2') # pair-wise count
df.groupby(' var ').function()
df.groupby(' var ').agg({'val' : 'mean'})
```

Join

```
df3 = df1.join(df2, [df1.var1 == df2.var1,
df1.var2 == df2.var2], 'left')
extra = df1.select(' var ').subtract(df2.select(' var')) # anti-join
```

Method Chaining

```
df
  .select("col1", "col2", " col 3",
  ...)
  .filter(df.col1 > 30)
  .show()
```

New Variable / Column

```
df.withColumn('varnew', df.var / 2.0)
```

To SQL / Pandas

```
df.registerAsTable('df_tbl')
sqlContext.sql('select var from df_tbl').show(5)
df.toPandas()
```

```
.describe('optional_var'.count()
r')

.show() .fillna(value)

.min() / .max() .mean()

.stdev() / .variance() .subset(1,3)

F.when(df.var > 30, "Y").otherwise("N")

df.var.alias('newvar') # used to rename something

df.cache() / df.unpersist .cast('Double')
()

.replace(10, 20) # ????.na.fill(0, subset = 'var'
)

time()

from pyspark.sql import functions as F
```



By **mitcht**
cheatography.com/mitcht/

Not published yet.
Last updated 29th December, 2017.
Page 1 of 2.

Sponsored by **ApolloPad.com**
Everyone has a novel in them. Finish
Yours!
<https://apollopad.com>