Cheatography

Cleaning with PySpark Cheat Sheet

by datamansam via cheatography.com/139410/cs/32141/

Defining Schema

```
from pyspark.sql.types import *
Schema =St ruc tType([
    Str uct Fie ld( 'St ore ',S tri ngT ype (), nul lab le= True),
    Str uct Fie ld( 'St ore Typ e', Str ing Typ e() ,nu lla ble =True),
    Str uct Fie ld( 'As sor tme nt' ,St rin gTy pe( ),n ull abl e=T rue),
    Str uct Fie ld( 'Co mpe tit ion Dis tan ce' ,Fl oat Typ e() ,nu lla ble =True),
    Str uct Fie ld( 'Co mpe tit ion Ope nSi nce Mon th' ,In teg erT ype (), nul lab le= True), Struct Fie -
ld( 'Co mpe tit ion Ope nSi nce Yea r', Int ege rTy pe( ),n ull abl e=T rue), Struct Fie ld( 'Pr omo 2', -
Int ege rTy pe( ),n ull abl e=T rue),
    Str uct Fie ld( 'Pr omo 2Si nce Wee k', Int ege rTy pe( ),n ull abl e=T rue),
    Str uct Fie ld( 'Pr omo 2Si nce Yea r', Int ege rTy pe( ),n ull abl e=T rue),
    Str uct Fie ld( 'Pr omo Int erv al' ,St rin gTy pe( ),n ull abl e=True)
])
df = spark.r ea d.o pti on( "he ade r", Tru e).s ch ema (Sc hem a).c sv ('s tor e.csv')
# We can drop invalid rows while reading the dataset by setting the read mode as "DROPM ALF ORMED"
df_1=s par k.r ead.op tio n(" hea der ",T rue ).o pti on( "mo de" ,'D ROP MAL FOR MED ').c sv ('s tor -
e.csv')
df.show()
```

Spark does not detect schema itself properly, so we need to define the schema as well for the data set.

PySpark DataTypes

Type	Size (Byte)	Default	Range (Digits)	
byte	1	0	3	Ints
short	2	0	5	
int	4	0	10	
long	8	0	Lots	
floats	4	0.0f	Lots	floats
double	8	0.0d	Lots	
DecimalType	32	0.0	Lots	

Filtering Data

String Data Types

StringType

VarcharType(length)	A variant of StringType which has a length limitation. Data writing will fail if the input string exceeds the length limitation
CharType(length)	Reading column of type CharType(n) always returns string values of length n. Char type column comparison will pad the short one to the longer length.

Adding, renaming and removing columns

Complex Data Types

ArrayType(elementType, containsNull)	nts values comprising a sequence of elements
MapType(keyType, valueType, valueContainsNull)	Represents values comprising a set of key-value pairs. The data type of keys is described by keyType and the data type of values is described by valueType. For a MapType value, keys are not allowed to have null values. valueContainsNull is used to indicate if values of a MapType value can have null values.
StructType(fields)	Represents values with the structure described by a sequence of StructFields (fields)

If, elif, else equivalent

```
voter_df.filter(voter_df['name'].isNoaNull ())
   OR
voter_ df.w here(~ voter_ df._ c1.is Nul l())
voter_ df.f il ter (vo ter _df.da te.year >
1800)
voter_ df.w he re( vot er_ df[ '_c 0'].c ~
nta ins ('V OTE'))
#Multiple Conditions
whereDF = flatte nDF.wh ere ((c ol( "voter_
stN ame ") == " xia ngr ui") | (col("Lum
tNa me") == " mic hae l")).so rt( ascf("lap
stN ame "))
whereD F.s how (tr unc ate =False)
#Unique Values
voter_df = df.sel ect (df ["VOTER
NAME"]).di sti nct()
# Show the rows with 10 highest IDs in the
set
voter_ df.o rd erB y(v ote r_d f.R OW_ID.d -
es c() ).s how(10)
```

```
.when(<if condition>, <then x>)
df.sel ect (df.Name, df.Age,
.when( df.Age >= 18, " Adu lt")
.when( df.Age < 18, " Min or"))
.other wise() is like else
df.sel ect (df.Name, df.Age,
.when( df.Age >= 18, " Adu lt")
.other wis e("M ino r")
```

Remove duplicate rows & replace values

```
dropDuplicates()
test_d f_n o_dup =
  tes t_d f.s ele ct( 'Us er_ -
ID' ,'G ender', 'Age').dr opD -
upl ica tes()
fillna()
used to replace null value with
any other value
df.fil lna (va lue =-9 9,s -
ubset=
```

User Defined Functions

```
1. Define a Python method
def revers eSt rin g(m ystr):
return mystr[ ::-1]
2. Wrap the function and store
as a variable
udfRev ers eString = udf(re -
ver seS tring, String Type())
3. Use with Spark
user_df = user_d f.w ith Col -
umn ('R eve rse Name',
udfRev ers eSt rin g(u ser -
_df.Name))
```

Using SQL to clean script

```
df.createOrReplaceTempView("table1")
df2 = spark.sql("SELECT field1, field2
FROM table1")
```

```
Add - withColumn
voter_ df.w it hCo lum n(' -
Year')voter_ df.d at e.year)
year > ng - withCo lum nRe named
test_d f_sex = test_d f.w ith -
Co lum n Ren ame d(' Gen der',
'Sex')
drop
voter_ df.d ro p(' unu sed _co -
lum ns)-
from pyspar k.sql import
functions as F
add_n = udf(lambda x, y: x + y,
Intege rTy pe())
# We register a UDF that adds a
column to the DataFrame,
and we cast the id column to an
Integer type.
df.id.c - wit hCo lum n(' id_ -
off set', add_n( F.l it( 1000),
df.id.c as t(I nte ger Typ -
e())))
```

Validating with Joins

```
parsed_df =
spark.read.parquet('parsed_data.parquet')
company_df = spark.r ea d.p arq uet ('c -
omp ani es.p ar quet')
verifi ed_df = parsed _df.jo in( com -
pan y_df, parsed _df.co mpany ==
compan y_d f.c ompany)
# This automa tically removes any rows
with a company not in the valid_df !
```

View data/actions:

printSchema(), head(), show(), count(), columns and describe()

show() - Displays/Prints a number of rows in a tabular format. By default it displays 20 rows and to change the default number, you can pass a value to show(n).

where as take(n) returns first n rows as Array of row objects. It is an alias for first().

count() - total rows

Sponsored by **Readable.com**
Measure your website readability!
https://readable.com

Remove duplicate rows & replace values (cont)

> [“Promo2SinceWeek”,”Promo2SinceYe-
ar”]).show()
.withColumn() ,when()
creating a new column, with value equal to 1 if
Promo2SinceYear > 2000 otherwise 0
df.withColumn(“greater_than_2000”,
when(df.CompetitionDistance==2000,1).ot-
herwise(0)
.alias(‘value_desc’)).show()

By **datamansam**

cheatography.com/datamansam/

Published 3rd September, 2022.
Last updated 12th September, 2022.

Sponsored by **Readable.com**
Measure your website readability!
https://readable.com