Cheatography

Comparing Core Pyspark and Pandas Code Cheat Sheet

by datamansam via cheatography.com/139410/cs/31975/

Importing Dataset

#SPARK
```
titanic_sp = spark.table("titanic_train")
```

#PANDAS
```
titanic_pd = titanic_sp.select("*").toPandas()
```

#FromPdBackToSPARK
```
pysparkDF2 = spark.createDataFrame(pandasDF)
```

View data in DataFrame

```
titanic_sp.show()
```

```
display(titanic_pd)
```

Display DataFrame schema

```
titanic_sp.printSchema()
```

```
titanic_pd.info()
```

The column names, column data type, non-null values and Pandas memory use

Renaming a column in a DataFrame

```
column_renamed=titanic_sp.withColumnRenamed("Name","PassengerName").columns
```

```
titanic_pd.rename(columns={'Name': 'PassengerName'}).columns
```

View number of columns and rows

#SPARK
```
print((titanic_sp.count(), len(titanic_sp.columns)))
```

#PANDAS
```
titanic_pd.shape
```

Dropping Columns

```
flight_data = flight_data.drop(columns_to_drop, axis = 1)
```

```
flight_data = flight_data.drop(*columns_to_drop)
```

Unique values of a column

```
titanic_sp.select('Survived').distinct().show()
```

```
titanic_pd['Survived'].unique()
```

View column names

```
titanic_sp.columns
```

```
titanic_pd.columns
```

Display column datatypes

```
titanic_sp.dtypes
```

```
titanic_pd.dtypes
```

Convert tyoes

```
flight_data = flight_data.withColumn('dt_departure', f.to_timestamp(flight_data.departure_datetime, 'yyyy-MM-dd HHmm'))
```

```
flight_data['dt_departure'] = pd.to_datetime(flight_data['dt_departure_datetime'], format='%Y-%m-%d %H%M')
```

Summary Stats

```
df.describe()
```

```
df.describe().show()
```

Aggregation

```
df.groupBy("Company").agg({'Sales':'sum'}).show()
```

```
pysparkDF.groupBy("gender") \
.agg(mean("age"),mean("salary"),max("salary")) \
.show()
```

Filter comparisons

```
df[df['species'].isin(['Chinstrap', 'Gentoo'])].show(5)
```

```
df[df['species'].isin(['Chinstrap', 'Gentoo'])].head()
```

```
df[df['species'].rlike('-G.')].show(5)
```

```
df[df['species'].str.match('G.')].head()
```

```
df[df['flipper'].between(225,229)].show(5)
```

```
df[df['flipper'].between(225,229)].head()
```

```
df[df['mass'].isNull()].show(5)
```

```
df[df['mass'].isnull()].head()
```

```
df[(df['mass']<3400) & (df['sex']=='Male')].head()
```

```
df[(df['mass']<3400) & (df['sex']=='Male')].show(5)
```

```
df[~df['flipper'].between(225,229)].show(5)
```

```
df[~df['flipper'].between(225,229)].head()
```

Conditional Transformations

```
flight_data['departure_delay_status'] = np.where(flight_data['departure_delay'] > 90, 'Heavy', 'Moderate')
```

```
flight_data = flight_data.withColumn('departure_delay_status', f.when(flight_data.departure_delay > 90, 'Heavy').otherwise('Moderate')
```