databricks notebook

```
from pyspark.sql import SparkSession
from pyspark.sql.functions import lit
from pyspark.sql import functions as F


  spark =
SparkSession.builder.appName('Script').master('local[*]').getOrCreate()
```

Extraction

```
# Extraction

def extract(file_path, file_type):
    return spark.read.format(file_type).option('header', True).load(file_path)
```

Cleaning

```
# Cleaning

def cleanse_USA(df_USA):
    return df_USA.drop('ID', 'Name',
'VaccinationDate').dropna().withColumn('CountryName', lit('USA'))

def cleanse_IND(df_IND):
    return df_IND.drop('ID', 'Name', 'DOB', 'VaccinationDate', 'Free or
Paid').dropna().withColumn('CountryName', lit('IND'))

def cleanse_AUS(df_AUS):
    return df_AUS.drop('Unique ID', 'Patient Name', 'Date of Birth', 'Date of
Vaccination').dropna().withColumn('CountryName',
lit('AUS')).withColumnRenamed('Vaccine Type', 'VaccinationType')
```

Merging

```
# Merging

def merge_all(df_USA, df_IND, df_AUS):
    cleansed_df_USA = cleanse_USA(df_USA)
    cleansed_df_IND = cleanse_IND(df_IND)
    cleansed_df_AUS = cleanse_AUS(df_AUS)
    return
cleansed_df_USA.unionByName(cleansed_df_IND).unionByName(cleansed_df_AUS)
```

Transforming & Aggregating

```
# Transforming & Aggregating

def metric(df_USA, df_IND, df_AUS):

    countries_data = [
        ('USA', 3295),
        ('IND', 13800),
        ('AUS', 257)]
    coutries_columns = ['CountryName', 'TotalPopulation']
    countries = spark.createDataFrame(countries_data, coutries_columns)

    df = merge_all(df_USA, df_IND, df_AUS)

    metric1 = df.groupby('CountryName',
'VaccinationType').agg(F.count('VaccinationType').alias('No. of vaccinations'))

    temp =
df.groupby('CountryName').agg(F.count('VaccinationType').alias('count'))
    metric2 = temp.withColumn('% Contribution',
temp['count']/df.count()*100).drop('count')
    temp =
df.groupby('CountryName').agg(F.count('VaccinationType').alias('count'))
    temp = temp.join(countries, on='CountryName')
    metric3 = temp.withColumn('% Vaccinated',
temp['count']/temp['TotalPopulation']*100).drop('count', 'TotalPopulation')

    return metric1, metric2, metric3
```

Unit Testing

```
# Unit Testing

import unittest

class TestNotebook(unittest.TestCase):

    def test_cleanse_USA(self):

        test_data = [(1,'Sam','EFG',6152022),
                     (2,'John','XYZ',1052022),
                     (3,'Mike','ABC',12282021)]

        data = ['ID', 'Name', 'VaccinationType', 'VaccinationDate']

        test_df = spark.createDataFrame(test_data,data)

        res = cleanse_USA(test_df)

        self.assertEqual(res.count(), 3)
        self.assertEqual(len(res.columns), 2)

    def test_cleanse_IND(self):

        test_data = [(1,'Sam','EFG','1998-12-01', '1998-12-01','F'),
                     (2,'John','XYZ', '1998-12-01', '1998-12-01','F'),
                     (3,'Mike','ABC', '1998-12-01', '1998-12-01','F')]

        data = ['ID', 'Name', 'DOB', 'VaccinationType', 'VaccinationDate',
'Free or Paid']

        test_df = spark.createDataFrame(test_data,data)

        res = cleanse_IND(test_df)

        self.assertEqual(res.count(), 3)
        self.assertEqual(len(res.columns), 2)
        self.assertListEqual(res.columns, ['VaccinationType', 'CountryName'])

    def test_cleanse_AUS(self):

        test_data = [(1,'Sam', 'EFG', 'NULL', 6152022),
                     (2,'John', 'XYZ', '1998-12-01', 105202),
                     (3,'Mike', 'ABC', '1998-12-03', 122821)]

        data = ['Unique ID', 'Patient Name', 'Vaccine Type', 'Date of Birth',
'Date of Vaccination']
```

```
        test_df = spark.createDataFrame(test_data,data)

        res = cleanse_AUS(test_df)

        self.assertEqual(res.count(), 3)
        self.assertEqual(len(res.columns), 2)
        self.assertListEqual(res.columns, ['VaccinationType', 'CountryName'])

    def test_merge_all(self):

        data = ['VaccinationType', 'CountryName']

        test_df1 = spark.createDataFrame([('EFG',6152022)], data)
        test_df2 = spark.createDataFrame([('XYZ',1052022)], data)
        test_df3 = spark.createDataFrame([('ABC',1228219)], data)

        res = merge_all(test_df1, test_df2, test_df3)

        self.assertEqual(res.count(), 3)
        self.assertEqual(len(res.columns), 2)
        self.assertListEqual(res.columns, data)

    def test_metric(self):

        test_df1 = spark.createDataFrame([('EFG','AUS'),
                                          ('XYZ','AUS'),
                                          ('ABC','AUS')],
                                         ['VaccinationType', 'CountryName'])
        test_df2 = spark.createDataFrame([('XYZ','AUS'),
                                          ('ABC','AUS'),
                                          ('ABC','AUS')],
                                         ['VaccinationType', 'CountryName'])
        test_df3 = spark.createDataFrame([('LMN','AUS'),
                                          ('XYZ','AUS'),
                                          ('ABC','AUS')],
                                         ['VaccinationType', 'CountryName'])

        res1, res2, res3 = metric(test_df1, test_df2, test_df3)

        self.assertEqual(res1.count(), 8)
        self.assertEqual(res2.count(), 3)
        self.assertEqual(res3.count(), 3)
        self.assertEqual(round(res2.select('% Contribution').agg(F.sum('%
Contribution')).collect()[0][0]),100)
```

```
unittest.main(argv=[''], verbosity=3, exit=False)

test_cleanse_AUS (__main__.TestNotebook) ... ok
test_cleanse_IND (__main__.TestNotebook) ... ok
test_cleanse_USA (__main__.TestNotebook) ... ok
test_merge_all (__main__.TestNotebook) ... ok
test_metric (__main__.TestNotebook) ... /usr/lib/python3.8/socket.py:740: Resou
rceWarning: unclosed <socket.socket fd=57, family=AddressFamily.AF_INET, type=S
ocketKind.SOCK_STREAM, proto=6, laddr=('127.0.0.1', 47130), raddr=('127.0.0.1',
41793)>
  self._sock = None
ResourceWarning: Enable tracemalloc to get the object allocation traceback
ok

----------------------------------------------------------------------
Ran 5 tests in 10.990s

OK
Out[16]: <unittest.main.TestProgram at 0x7f8a3b10f2b0>
```

Config

```
# Config

file_path_USA = '/FileStore/tables/USA-1.csv'  #sample_file
file_path_IND = '/FileStore/tables/IND.csv'    #sample_file
file_path_AUS = '/FileStore/tables/AUS.xlsx'   #sample_file

csv = 'csv'
excel = 'com.crealytics.spark.excel'

df_USA = extract(file_path_USA, csv)
df_IND = extract(file_path_IND, csv)
df_AUS = extract(file_path_AUS, excel)
```

Metrics based on Sample File

```
# Metrics based on Sample File

metric1, metric2, metric3 = metric(df_USA, df_IND, df_AUS)


metric1.show()

+-----------+---------------+-------------------+
|CountryName|VaccinationType|No. of vaccinations|
```

```
+-----------+---------------+-------------------+
|        USA|            XYZ|                  1|
|        USA|            EFG|                  1|
|        USA|            ABC|                  1|
|        IND|            ABC|                  2|
|        IND|            XYZ|                  1|
|        AUS|            LMN|                  1|
|        AUS|            XYZ|                  1|
|        AUS|            ABC|                  1|
+-----------+---------------+-------------------+
```

```
metric2.show()
```

```
+-----------+-----------------+
|CountryName|   % Contribution|
+-----------+-----------------+
|        USA|33.33333333333333|
|        IND|33.33333333333333|
|        AUS|33.33333333333333|
+-----------+-----------------+
```

```
metric3.show()
```

```
+-----------+--------------------+
|CountryName|        % Vaccinated|
+-----------+--------------------+
|        USA| 0.09104704097116845|
|        IND|0.021739130434782608|
|        AUS|  1.1673151750972763|
+-----------+--------------------+
```