**Spark**

**import psycopg2 as pg**

**from pyspark.sql import SparkSession**

**from pyspark.sql import Window**

**import sys**

**from pyspark.sql.functions import sum, bround**

**from pyspark.sql import DataFrameReader**

**from pyspark.sql import SparkSession**

**from pyspark.sql.functions import col,count**

**import pandas as pd**

**from pyspark.sql import functions as F**

spark = SparkSession \

.builder \

.appName("Python Spark SQL for DW Project") \

.config("spark.some.config.option", "some-value") \

.getOrCreate()

connection = pg.connect("host='localhost' dbname=DataWarehouse1 user=postgres password='Ankur@1234'")

donation=pd.read\_sql\_query("Select \* from ankur limit 100000",con=connection)

team=pd.read\_sql\_query("Select \* from bike\_teams limit 100000", con=connection)

events=pd.read\_sql\_query("select \* from events limit 100000", con=connection)

particpants=pd.read\_sql\_query("select \* from participants limit 100000", con=connection)

avgyear=pd.read\_sql\_query("select \* from avgyear limit 100000", con=connection)

city=pd.read\_sql\_query("select \* from city limit 100000", con=connection)

gender=pd.read\_sql\_query("select \* from gender limit 100000", con=connection)

team\_df = spark.createDataFrame(team)

donation\_df = spark.createDataFrame(donation)

particpants\_df = spark.createDataFrame(particpants)

avgyear\_df = spark.createDataFrame(avgyear)

city\_df = spark.createDataFrame(city)

gender\_df = spark.createDataFrame(gender)

#-----------------------------------------------------1st new

events\_df.select("average\_team\_size","total\_from\_participant","state").where(col("state").isNotNull()).groupBy("average\_team\_size","state").sum().orderBy("state").show();

#-----------------------------------------------------2nd new

particpants\_df.select("occupation","total\_from\_participant","participant\_gender").where(col("occupation").isNotNull()).groupBy("occupation","total\_from\_participant","participant\_gender").sum().orderBy("occupation").show();

#-----------------------------------------------------3rd new

team\_df.filter(col("fiscal\_year").startswith("2013.0")).rollup("fiscal\_year","event\_type", team\_df.total\_offline\_confirmed\_gifts).count().where(col("event\_type").isNotNull()).orderBy("fiscal\_year","event\_type").show()

#-----------------------------------------------------4th new

donation\_df.rollup("fiscal\_year","donor\_connection\_to\_ms", donation\_df.transaction\_amount).sum("transaction\_amount").orderBy("fiscal\_year","donor\_connection\_to\_ms").limit(100).show()

#-----------------------------------------------------5th new

win\_spec = Window.partitionBy('occupation').orderBy('occupation').rowsBetween(-sys.maxsize, 0)

cum\_sum = particpants\_df.select("occupation","total\_from\_participant").withColumn('cumsum', F.sum(particpants\_df.total\_from\_participant).over(win\_spec))

cum\_sum.show();

#------------------------------------------------------OLAP QUERIES

#------------------------------------------------------OLAP 1st

donation\_df.select("gift\_amount","is\_prior\_participant","fiscal\_year").groupBy("gift\_amount","is\_prior\_participant","fiscal\_year").show();

#------------------------------------------------------OLAP 2nd

city\_df.select("sum","donor\_state","donor\_city").groupBy("donor\_state","donor\_city","sum").max("sum").show();

#------------------------------------------------------OLAP 3rd

avgyear\_df.select("\*").groupBy("event\_id","fiscal\_year","total\_amount").avg("total\_amount").show();

#------------------------------------------------------OLAP 4th

gender\_df.select("average","donor\_gender",col("sum").cast("double")).groupBy("donor\_gender",).max("sum","average").show();

#------------------------------------------------------OLAP 5th

gender\_df.select("average","donor\_gender",col("sum").cast("double"),"fiscal\_year").groupBy("donor\_gender","fiscal\_year").max("sum","average").show();