**Day18**

**DATA ENGINEERING**

# Databricks notebook source

from pyspark.sql.functions import\*

from pyspark.sql.types import StringType,IntegerType

# COMMAND ----------

df\_001=spark.read.option("header",True).option("mode","permisive").option("inferschema",True).format("csv").load("/mnt/databricks/employee.csv")

# COMMAND ----------

df\_001.display()

# COMMAND ----------

df\_001.write.format("delta").save("/mnt/databricks/storage2/")

# COMMAND ----------

# MAGIC %sql

# MAGIC CREATE TABLE ext\_store\_trans1

# MAGIC USING Delta

# MAGIC OPTIONS (path "abfss://databricks@adlsgmde1710.dfs.core.windows.net/storage2")

# COMMAND ----------

dbutils.widgets.text("param1","department")

# COMMAND ----------

dbutils.widgets.get("param1")

# COMMAND ----------

# MAGIC %sql

# MAGIC select \*from ext\_store\_trans1 where Department="$param1"

# COMMAND ----------

selected\_department = dbutils.widgets.get("param1")

filtered\_df = df\_001.filter(df\_001["Department"] == selected\_department)

# COMMAND ----------

filtered\_df.display()

# COMMAND ----------

dbutils.widgets.dropdown("param2\_dropdown","Engineering",["Engineering","Marketing","Sales"])

# COMMAND ----------

selected\_department2 = dbutils.widgets.get("param2\_dropdown")

filtered\_df2 = df\_001.filter(df\_001["Department"] == selected\_department2)

# COMMAND ----------

filtered\_df2.display()

# COMMAND ----------

# MAGIC %sql

# MAGIC select \*from ext\_store\_trans1 where Department="$param2\_dropdown"

# COMMAND ----------

dbutils.widgets.combobox("param3\_combobox","Engineering",["Engineering","Marketing","Sales"])

# COMMAND ----------

# MAGIC %sql

# MAGIC select \*from ext\_store\_trans1 where Department="$param3\_combobox"

# COMMAND ----------

selected\_department3 = dbutils.widgets.get("param3\_combobox")

filtered\_df3 = df\_001.filter(df\_001["Department"] == selected\_department3)

# COMMAND ----------

filtered\_df3.display()

# COMMAND ----------

dbutils.widgets.multiselect("param4\_multiselect","Engineering",["Engineering","Marketing","Sales"])

# COMMAND ----------

# MAGIC %sql

# MAGIC select \*from ext\_store\_trans1 where Department in ($param4\_multiselect)

# COMMAND ----------

def addSal(a):

a+=1000

return a

# COMMAND ----------

add\_fun=udf(addSal,IntegerType())

# COMMAND ----------

df\_add=df\_001.withColumn("updatedSalary",add\_fun("Salary"))

# COMMAND ----------

df\_add.display()

# COMMAND ----------

df.createOrReplaceTempView("tran\_table")

# COMMAND ----------

spark.udf.register("add\_fun\_sql",addSal,IntegerType())

# COMMAND ----------

# MAGIC %sql

# MAGIC select add\_fun\_sql(Salary) from ext\_store\_trans

# COMMAND ----------

# MAGIC %run /Workspace/Users/rigigowdaazure@gmail.com/GrayMatter\_Databricks/utitlity

# COMMAND ----------

df\_add2=df\_001.withColumn("subedSalary",add\_Sub(col("Salary")))

# COMMAND ----------

df\_add2.display()

# COMMAND ----------

#scheduling

dbutils.notebook.run("/Workspace/Users/rigigowdaazure@gmail.com/GrayMatter\_Databricks/utitlity",1)

Utility Notebook Where the functions are stored to be imported

# Databricks notebook source

from pyspark.sql.functions import\*

from pyspark.sql.types import StringType,IntegerType

# COMMAND ----------

def addSal(a):

a+=1000

return a

# COMMAND ----------

def SubSal(a):

a-=1000

return a

# COMMAND ----------

def mulSal(a):

a\*=2

return a

# COMMAND ----------

def RedSal(a):

a/=2

return a

# COMMAND ----------

add\_Sal=udf(addSal,IntegerType())

# COMMAND ----------

add\_Sub=udf(SubSal,IntegerType())

# COMMAND ----------

add\_mul=udf(mulSal,IntegerType())

# COMMAND ----------

add\_div=udf(RedSal,IntegerType())

# COMMAND ----------

Cache & Persist

# Databricks notebook source

# DBTITLE 1,base

jdbc\_url = "jdbc:sqlserver://gmde1710.database.windows.net:1433;database=gmde"

jdbc\_properties = {

"user": "nanuninujodi",

"password": "ninuNanujodi69",

"driver": "com.microsoft.sqlserver.jdbc.SQLServerDriver"

}

# COMMAND ----------

table\_name = "saleslt.Customer"

df\_001 = spark.read.jdbc(url=jdbc\_url, table = table\_name,properties=jdbc\_properties)

# COMMAND ----------

df\_001.display()

# COMMAND ----------

df\_001.rdd.getNumPartitions()

# COMMAND ----------

spark.conf.get("spark.sql.files.maxPartitionBytes")

# COMMAND ----------

df\_001=df\_001.coalesce(5)

# COMMAND ----------

df\_001.rdd.getNumPartitions()

# COMMAND ----------

df\_001=df\_001.repartition(5)

# COMMAND ----------

df\_001.rdd.getNumPartitions()

# COMMAND ----------

df\_001.cache().count()

# COMMAND ----------

df\_001.display()

# COMMAND ----------

from pyspark.storagelevel import StorageLevel

df\_001.persist(StorageLevel.MEMORY\_AND\_DISK)