```
# Replace with your actual storage account info
storage_account_name = "pimamedalliondata"
storage_account_key =
"4tettVyFg9h7Neu4Zse9mn4R/1cjqbA3R5/gNuJCzOPK4h1uT0aMYU41D36snXkoaH6hb
qYVZ8nE+AStFT+GRQ=="

configs = {
  f"fs.azure.account.key.
{storage_account_name}.blob.core.windows.net": storage_account_key
}

# Mount Bronze container
dbutils.fs.mount(
  source =
f"wasbs://bronze@{storage_account_name}.blob.core.windows.net/",
  mount_point = "/mnt/bronze",
  extra_configs = configs
)
```

```
True
```

```
# unmount Bronze container
mount_point_bronze = "/mnt/bronze"

try:
    dbutils.fs.unmount(mount_point_bronze)
except:
    pass

dbutils.fs.mount(
  source =
f"wasbs://bronze@{storage_account_name}.blob.core.windows.net/",
  mount_point = mount_point_bronze,
  extra_configs = configs)
```

```
/mnt/bronze has been unmounted.
```

```
True
```

```
# Define schema for the PIMA dataset columns
from pyspark.sql.types import StructType, StructField, IntegerType,
DoubleType

schema = StructType([
    StructField("Pregnancies", IntegerType(), True),
    StructField("Glucose", IntegerType(), True),
    StructField("BloodPressure", IntegerType(), True),
    StructField("SkinThickness", IntegerType(), True),
    StructField("Insulin", IntegerType(), True),
    StructField("BMI", DoubleType(), True),
```

```
    StructField("DiabetesPedigreeFunction", DoubleType(), True),
    StructField("Age", IntegerType(), True),
    StructField("Outcome", IntegerType(), True)
])

# Load raw CSV from bronze mount point
raw_df = spark.read.schema(schema).option("header",
False).csv("/mnt/bronze/pima-indians-diabetes.data.csv")

# Save raw data as Delta table in bronze layer (for performance &
versioning)
raw_df.write.format("delta").mode("overwrite").save("/mnt/bronze/delta
/pima_raw")

bronze_df =
spark.read.format("delta").load("/mnt/bronze/delta/pima_raw")

from pyspark.sql.functions import when, col

# Columns where zero is invalid
zero_invalid_cols = ["Glucose", "BloodPressure", "SkinThickness",
"Insulin", "BMI"]

for c in zero_invalid_cols:
    bronze_df = bronze_df.withColumn(c, when(col(c) == 0,
None).otherwise(col(c)))

from pyspark.sql.functions import expr

for c in zero_invalid_cols:
    median_val = bronze_df.approxQuantile(c, [0.5], 0.25)[0]
    bronze_df = bronze_df.na.fill({c: median_val})

from pyspark.sql.functions import expr

bronze_df = bronze_df.withColumn("BMICategory",
    expr("""
    CASE
      WHEN BMI < 18.5 THEN 'Underweight'
      WHEN BMI >= 18.5 AND BMI < 25 THEN 'Normal'
      WHEN BMI >= 25 AND BMI < 30 THEN 'Overweight'
      ELSE 'Obese'
    END
    """)
)

#save clean data into silver delta table
bronze_df.write.format("delta").mode("overwrite").save("/mnt/silver/
delta/pima_cleaned")
```

```
#load silver data
silver_df =
spark.read.format("delta").load("/mnt/silver/delta/pima_cleaned")

#calculate summary stats
gold_df = silver_df.groupBy("Outcome", "BMICategory").agg(
    {"BMI": "avg", "Age": "avg"}
).withColumnRenamed("avg(BMI)",
"Avg_BMI").withColumnRenamed("avg(Age)", "Avg_Age")

#Save gold data
gold_df.write.format("delta").mode("overwrite").save("/mnt/gold/delta/
pima_summary")

#gold layer results
display(spark.read.format("delta").load("/mnt/gold/delta/pima_summary"
))
```