WASILA REHMAN 12348

PROJECT 1 – KAFKA SPARK STREAMING PIPELINE

WASILA REHMAN - 12348
[Email address]

Industry

Health Care

Problem Statement

According to the World Health Organization, every year 12 million deaths occur worldwide due to heart diseases. Heart disease is one of the major causes of mortality among the population of the world.

The major challenge in heart disease is its detection. An early diagnosis of heart disease can help us make lifestyle changes in high-risk patients and ultimately reduce the chances of future complications.

There are instruments available which can predict heart disease but either are expensive or not fast enough to calculate chance of heart disease in humans.

That's why my goal in this project is to build an analytical dashboard on this big data so doctors can easily access insightful reports on key metrics.

Dataset Information

Originally, the dataset come from the CDC and is a major part of the Behavioral Risk Factor Surveillance System (BRFSS), which conducts annual telephone surveys to gather data on the health status of U.S. residents.

This dataset is already cleaned and has 319795 rows × 18 columns.

Following are the features present in this dataset:

1. **HeartDisease:** Respondents that have ever reported having coronary heart disease (CHD) or myocardial infarction (MI).
2. **BMI:** Body Mass Index (BMI).
3. **Smoking:** Have you smoked at least 100 cigarettes in your entire life?
4. **AlcoholDrinking:** Heavy drinkers (adult men having more than 14 drinks per week and adult women having more than 7 drinks per week
5. **Stroke:** (Ever told) (you had) a stroke?
6. **PhysicalHealth:** Now thinking about your physical health, which includes physical illness and injury, for how many days during the past 30 days was your physical health not good? (0-30 days).
7. **MentalHealth:** Thinking about your mental health, for how many days during the past 30 days was your mental health not good? (0-30 days).
8. **DiffWalking:** Do you have serious difficulty walking or climbing stairs?
9. **Sex:** Are you male or female?
10. **AgeCategory:** Fourteen-level age category.
11. **Race:** Imputed race/ethnicity value.
12. **Diabetic:** (Ever told) (you had) diabetes?
13. **PhysicalActivity:** Adults who reported doing physical activity or exercise during the past 30 days other than their regular job.
14. **GenHealth:** Would you say that in general your health is...
15. **SleepTime:** On average, how many hours of sleep do you get in a 24-hour period?

16. **Asthma:** (Ever told) (you had) asthma?
17. **KidneyDisease:** Not including kidney stones, bladder infection or incontinence, were you ever told you had kidney disease?
18. **SkinCancer:** (Ever told) (you had) skin cancer?

Key Metrics to Track

According to BRFSS Machine Learning researchers, it was decided the key factors that contribute the most in heart diseases are:

- Mental Health
- Physical Health
- Age
- Gender
- BMI
- Sleep Time

For most hospitals the number of available doctors is always limited. Hence a quick diagnosis is needed to figure out whether a patient needs urgent treatment.

Analyzing dataset of the past 2 million records helps in uncovering the correlations between each health factor. Consequently, it saves time for doctor's diagnosis and understanding in what category a new patient will fall in.

The key questions this project answers are:

1. **Are the chances of Heart Disease higher in a certain Age category?**
2. **What is the average BMI of Heart Patients in each Age Category?**
3. **Is there a correlation between Heart Diseases and Gender?**
4. **What is the average number of sleeping hours in heart patients of each age category?**

STAGE 1 – Docker Compose

Building the Docker Compose File with the following services:

- Hive-server
- Hive-metastore
- Hive-metastore-postgresql
- Kafka
- Prometheus
- Grafana (dependency on Prometheus)
- Zeppelin (dependency on hive)

STAGE 2 – Dependencies

Downloading dependencies in Zeppelin:

To install kafka-python in Zeppelin shell

```
%sh
```

```
pip install kafka-python
```

These jar files were the essence of code running inside Zeppelin. Figuring each dependency out from maven is crucial to run the pipeline.

```
%sh

mkdir /zeppelin/dep

cd /zeppelin/dep && wget https://repo1.maven.org/maven2/org/apache/spark/spark-streaming-kafka-0-8-assembly_2.11/2.0.2/spark-streaming-kafka-0-8-assembly_2.11-2.0.2.jar

cd /zeppelin/dep && wget https://repo1.maven.org/maven2/org/apache/spark/spark-streaming-kafka-0-10_2.11/2.0.2/spark-streaming-kafka-0-10_2.11-2.0.2.jar

cd /zeppelin/dep && wget https://repo1.maven.org/maven2/org/apache/hive/hive-jdbc/2.3.6/hive-jdbc-2.3.6.jar

cd /zeppelin/dep && wget https://repo1.maven.org/maven2/org/apache/hive/hive-service/2.3.6/hive-service-2.3.6.jar

cd /zeppelin/dep && wget https://repo1.maven.org/maven2/org/apache/hive/hive-exec/2.3.6/hive-exec-2.3.6.jar
```

Downloading the same dependencies for spark, consumer.pyspark, and producer.pyspark

```
%spark.dep

z.reset()

z.load("/zeppelin/dep/hive-jdbc-2.3.6.jar")

z.load("/zeppelin/dep/hive-service-2.3.6.jar")

z.load("/zeppelin/dep/hive-exec-2.3.6.jar")


%producer.dep

z.reset()

z.load("/zeppelin/dep/spark-streaming-kafka-0-10_2.11-2.0.2.jar")

z.load("/zeppelin/dep/spark-sql-kafka-0-10_2.11:2.3.0.jar")



%consumer.dep

z.reset()

z.load("/zeppelin/dep/spark-sql-kafka-0-10_2.11:2.2.0.jar")
```

```
z.load("/zeppelin/dep/spark-streaming-kafka-0-10-assembly_2.11-2.0.2.jar")
z.load("/zeppelin/dep/kafka_2.11.jar")
z.load("/zeppelin/dep/kafka-clients.jar")
z.load("/zeppelin/dep/hive-jdbc-2.3.6.jar")
z.load("/zeppelin/dep/hive-service-2.3.6.jar")
z.load("/zeppelin/dep/hive-exec-2.3.6.jar")
```

STAGE 3 – Building Kafka Producer

(CAN KEEP RUNNING FOR 40 MINUTES BEFORE KAFKA PRODUCER TIMEOUTS)

First step is to build the data frame for the Producer by reading a big data csv file.

```
%producer.pyspark


df = (spark.read.format("com.databricks.spark.csv")
      .option("header", "true")
      .option("inferSchema","true")
      .load("/datadrive/heart_2020.csv"))


df_list = df.collect()
df.show()
df.printSchema()
```

Next step is to enable Producer send data to our exposed Kafka_brokers Ports. Kafka_Brokers are the mid-level broker who take data from the Producer and send it to Consumer. It is responsible for all the connections.

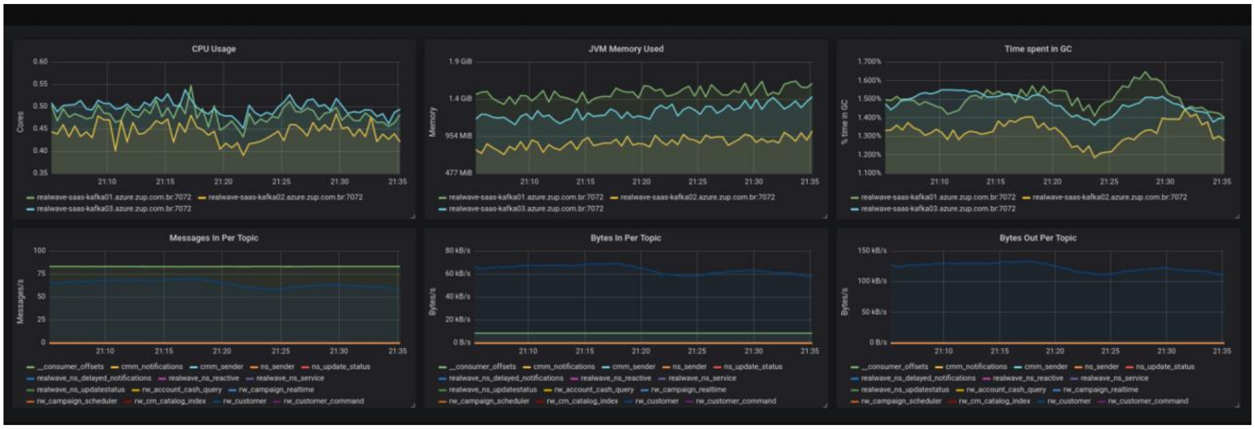
In the Kafka Topic we have specified our key and string. The json dumps it into our row_dictionary.

```
%producer.pyspark
import time
import json
import random
import logging

from kafka import KafkaProducer
from kafka.errors import KafkaError

KAFKA_BROKER = "172.25.0.12:9092"
KAFKA_INPUT_TOPIC = "default_topic"

producer = KafkaProducer(bootstrap_servers=[KAFKA_BROKER])
index = 0

while True:

    row_dict = df_list[index].asDict()

    future = producer.send(
        topic=KAFKA_INPUT_TOPIC,
        key=str(row_dict["PatientID"]).encode("utf-8"),
        value=json.dumps(row_dict).encode("utf-8"))

    try:

        record_metadata = future.get(timeout=10)
    except KafkaError:
        # Decide what to do if produce request failed...
        logging.exception("Error")
        pass

    producer.flush()
    index += 1


    time.sleep(random.uniform(0.1,3.0))
return (row_dict)
```

Stage 4 – Building Kafka Consumer

TIME TAKEN: 8MINUTES (until zeppelin range ends)

Kafka Consumer builds a spark session and creates a table in Hive while consumer keeps on appending on each iteration.



```
%consumer.pyspark

import json

from pyspark.sql import SparkSession

from pyspark.sql import HiveContext

from pyspark.sql import Row

from kafka import KafkaConsumer

import pyspark

from ast import literal_eval


spark =
SparkSession.builder.master("local[1]").appName("Heart_Diseases").enableHiveSupport().getOrCreate()
```

```
HiveContext = HiveContext(sc)
HiveContext.sql("CREATE DATABASE IF NOT EXISTS heart_db")

# To consume latest messages and auto-commit offsets
consumer = KafkaConsumer('default_topic',
                         group_id='final-consumer',
                         bootstrap_servers=['172.25.0.12:9092'])

for message in consumer:

    data = literal_eval(message.value)
    studentDf = spark.createDataFrame([
                  Row(PatientID= data['PatientID'], heartDisease=data['HeartDisease'],
PhysicalHealth=data['PhysicalHealth'], MentalHealth=data['MentalHealth'],
AgeCategory=data['AgeCategory'], SleepTime=data['SleepTime'], gender=data['Sex'], BMI=data['BMI'])
    ])

    studentDf.write.mode('append').saveAsTable("heart_db.table_1")
    df1 = spark.sql("select * from heart_db.table_1")
    df1.show()

# consume earliest available messages, don't commit offsets
KafkaConsumer(auto_offset_reset='earliest', enable_auto_commit=False)

# consume json messages
KafkaConsumer(value_deserializer=lambda m: json.loads(m.decode('ascii')))

# StopIteration if no message after 1sec
KafkaConsumer(consumer_timeout_ms=100)
```

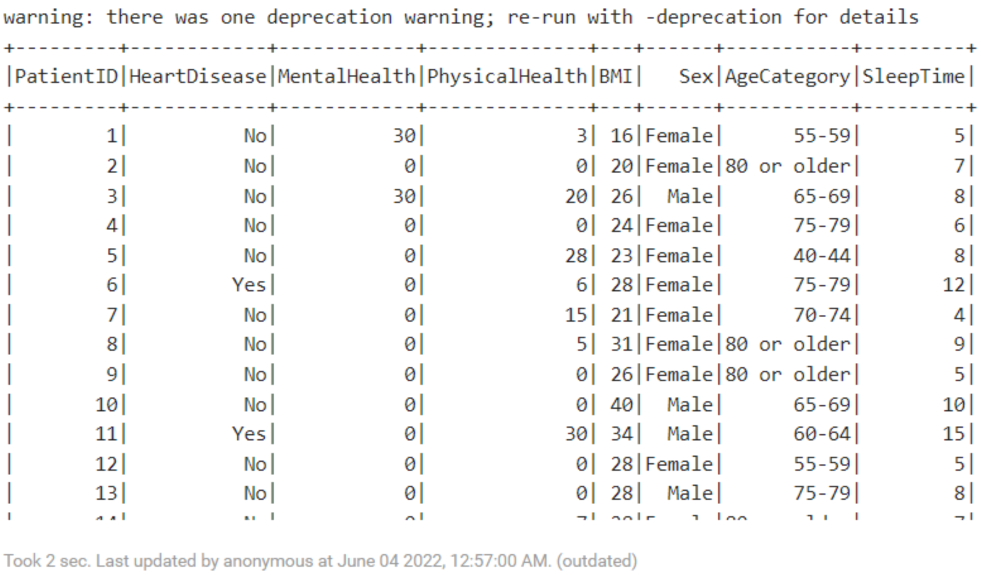
STAGE 5 – Accessing tables from Hive & Hive Query Language (Spark SQL)

First we will access the data from the Hive table.

(TIME TAKEN : 2 SECONDS)

```
%spark

import org.apache.spark.sql.hive.HiveContext

val HiveContext = new org.apache.spark.sql.hive.HiveContext(sc)


HiveContext.sql("CREATE DATABASE IF NOT EXISTS heart_patients")

HiveContext.sql("CREATE TABLE IF NOT EXISTS heart_patients_Dets(PatientID INT, HeartDisease STRING, MentalHealth STRING, PhysicalHealth STRING, BMI INT, Sex string, AgeCategory string, SleepTime INT) ROW FORMAT DELIMITED FIELDS TERMINATED BY ',' LINES TERMINATED BY '\n' ")

HiveContext.sql("LOAD DATA LOCAL INPATH '/datadrive/table_1.csv' INTO TABLE heart_patients_Dets")

val df1 = HiveContext.sql("Select * from heart_patients_Dets").show()
```

OUTPUT:



```
warning: there was one deprecation warning; re-run with -deprecation for details
+---------+------------+------------+--------------+---+------+-----------+---------+
|PatientID|HeartDisease|MentalHealth|PhysicalHealth|BMI|   Sex|AgeCategory|SleepTime|
+---------+------------+------------+--------------+---+------+-----------+---------+
|        1|          No|          30|             3| 16|Female|      55-59|        5|
|        2|          No|           0|             0| 20|Female|80 or older|        7|
|        3|          No|          30|            20| 26|  Male|      65-69|        8|
|        4|          No|           0|             0| 24|Female|      75-79|        6|
|        5|          No|           0|            28| 23|Female|      40-44|        8|
|        6|         Yes|           0|             6| 28|Female|      75-79|       12|
|        7|          No|           0|            15| 21|Female|      70-74|        4|
|        8|          No|           0|             5| 31|Female|80 or older|        9|
|        9|          No|           0|             0| 26|Female|80 or older|        5|
|       10|          No|           0|             0| 40|  Male|      65-69|       10|
|       11|         Yes|           0|            30| 34|  Male|      60-64|       15|
|       12|          No|           0|             0| 28|Female|      55-59|        5|
|       13|          No|           0|             0| 28|  Male|      75-79|        8|
```

Took 2 sec. Last updated by anonymous at June 04 2022, 12:57:00 AM. (outdated)

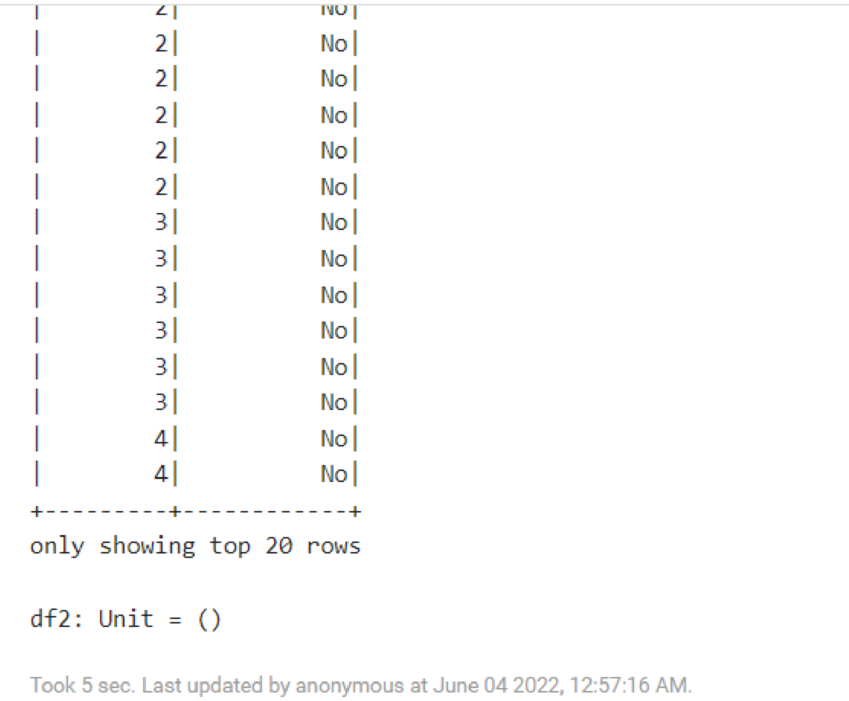
Next, we start querying the data to figure out dataframes for our Key Metrics.

Query 1 – to return patients with SleepTime < 10

(TIME TAKEN 5 SECS)

%spark

val df2 = HiveContext.sql("SELECT PatientID, HeartDisease FROM heart_patients_Details WHERE SleepTime < 10 ORDER BY PatientID").show()



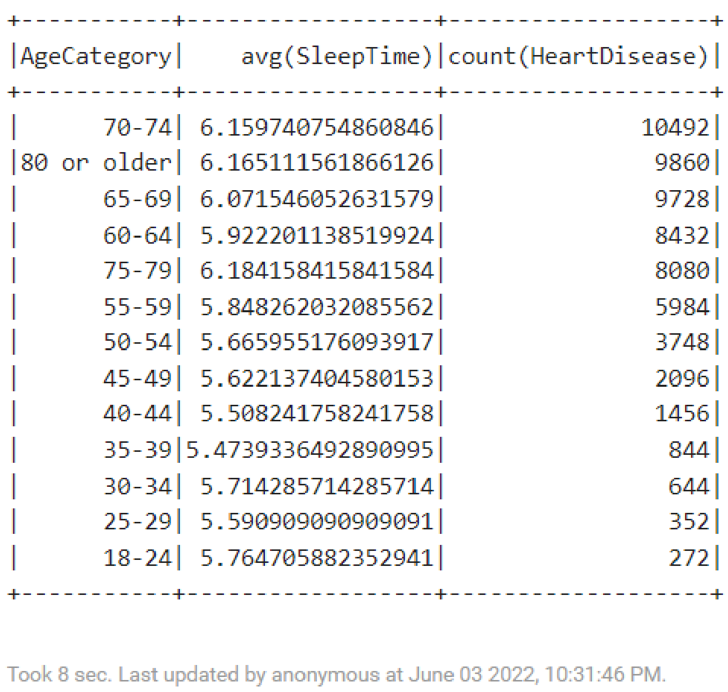
```
|        2|          No|
|        2|          No|
|        2|          No|
|        2|          No|
|        2|          No|
|        2|          No|
|        3|          No|
|        3|          No|
|        3|          No|
|        3|          No|
|        3|          No|
|        3|          No|
|        4|          No|
|        4|          No|
+---------+------------+
only showing top 20 rows

df2: Unit = ()
```

Took 5 sec. Last updated by anonymous at June 04 2022, 12:57:16 AM.

Query 2 – To gauge what is an average sleep time of HeartDisease Patients

(TIME TAKEN 8 SECS)

val df3 = HiveContext.sql("SELECT AgeCategory, AVG(SleepTime), count(HeartDisease) FROM heart_patients_Details WHERE HeartDisease = 'Yes' AND SleepTime < 8 GROUP BY AgeCategory Order BY count(HeartDisease) desc").show()



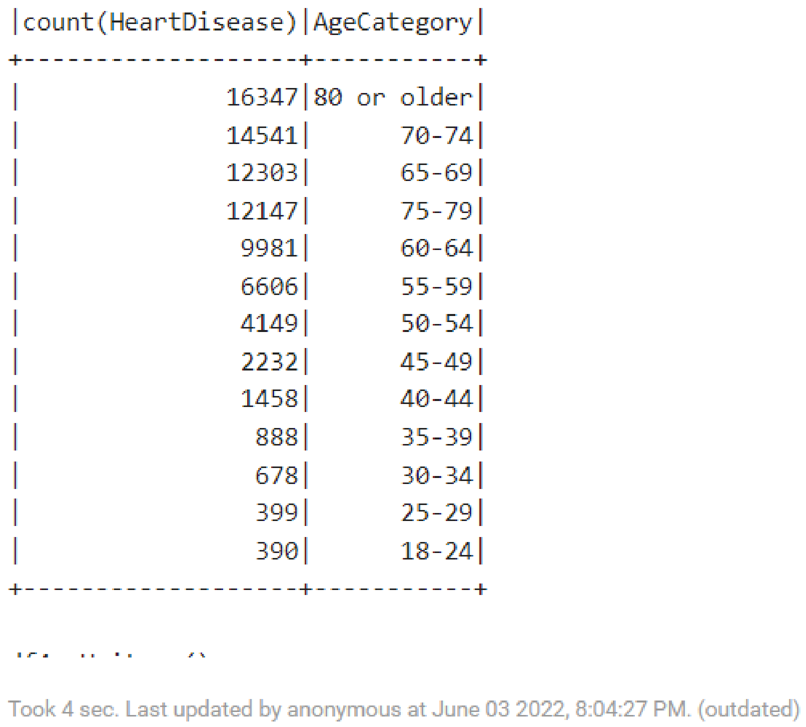
```
+-----------+-----------------+------------------+
|AgeCategory|   avg(SleepTime)|count(HeartDisease)|
+-----------+-----------------+------------------+
|      70-74| 6.159740754860846|             10492|
|80 or older| 6.16511561866126|              9860|
|      65-69| 6.071546052631579|              9728|
|      60-64| 5.922201138519924|              8432|
|      75-79| 6.184158415841584|              8080|
|      55-59| 5.848262032085562|              5984|
|      50-54| 5.665955176093917|              3748|
|      45-49| 5.622137404580153|              2096|
|      40-44| 5.508241758241758|              1456|
|      35-39|5.4739336492890995|               844|
|      30-34| 5.714285714285714|               644|
|      25-29| 5.590909090909091|               352|
|      18-24| 5.764705882352941|               272|
+-----------+-----------------+------------------+
```

Took 8 sec. Last updated by anonymous at June 03 2022, 10:31:46 PM.

Query 3 – To see number of Heart Disease Patients based on Age

(TIME TAKEN 4 SECONDS)

val df4 = HiveContext.sql("Select count(HeartDisease) , AgeCategory from heart_patients_Details where HeartDisease = 'Yes' group by AgeCategory Order by count(HeartDisease) desc").show()

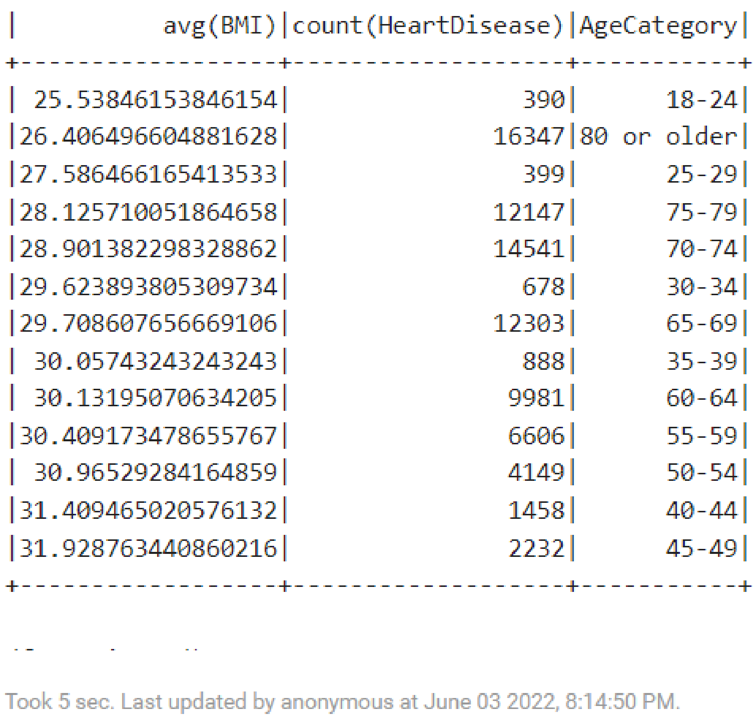


```
|count(HeartDisease)|AgeCategory|
+------------------+-----------+
|             16347|80 or older|
|             14541|      70-74|
|             12303|      65-69|
|             12147|      75-79|
|              9981|      60-64|
|              6606|      55-59|
|              4149|      50-54|
|              2232|      45-49|
|              1458|      40-44|
|               888|      35-39|
|               678|      30-34|
|               399|      25-29|
|               390|      18-24|
+------------------+-----------+
```

Took 4 sec. Last updated by anonymous at June 03 2022, 8:04:27 PM. (outdated)

Query 4 – To see average BMI of Heart Patients based on AgeCateogry
(TIME TAKEN 5 SECONDS)

val df5 = HiveContext.sql("Select AVG(BMI) , count(HeartDisease), AgeCategory from heart_patients_Details where HeartDisease = 'Yes' group by AgeCategory Order by AVG(BMI), count(HeartDisease) desc").show()

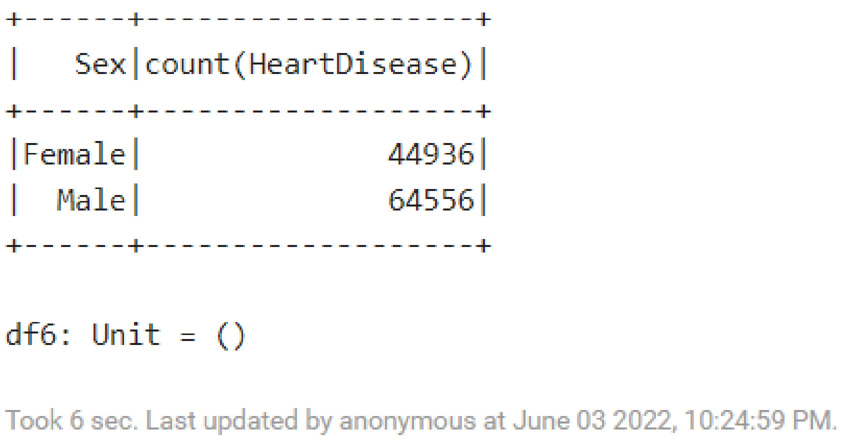


```
|          avg(BMI)|count(HeartDisease)|AgeCategory|
+------------------+-------------------+-----------+
| 25.53846153846154|                390|      18-24|
|26.406496604881628|              16347|80 or older|
|27.586466165413533|                399|      25-29|
|28.125710051864658|              12147|      75-79|
|28.901382298328862|              14541|      70-74|
|29.623893805309734|                678|      30-34|
|29.708607656669106|              12303|      65-69|
| 30.05743243243243|                888|      35-39|
| 30.13195070634205|               9981|      60-64|
|30.409173478655767|               6606|      55-59|
| 30.96529284164859|               4149|      50-54|
|31.409465020576132|               1458|      40-44|
|31.928763440860216|               2232|      45-49|
+------------------+-------------------+-----------+
```

Took 5 sec. Last updated by anonymous at June 03 2022, 8:14:50 PM.

Query 5 – To gauge Heart Patients based on gender
(TIME TAKEN 6 SECONDS)

val df6 = HiveContext.sql("Select Sex, count(HeartDisease) from heart_patients_Details where HeartDisease = 'Yes' group by Sex").show()



```
+------+-------------------+
|   Sex|count(HeartDisease)|
+------+-------------------+
|Female|              44936|
|  Male|              64556|
+------+-------------------+

df6: Unit = ()
```

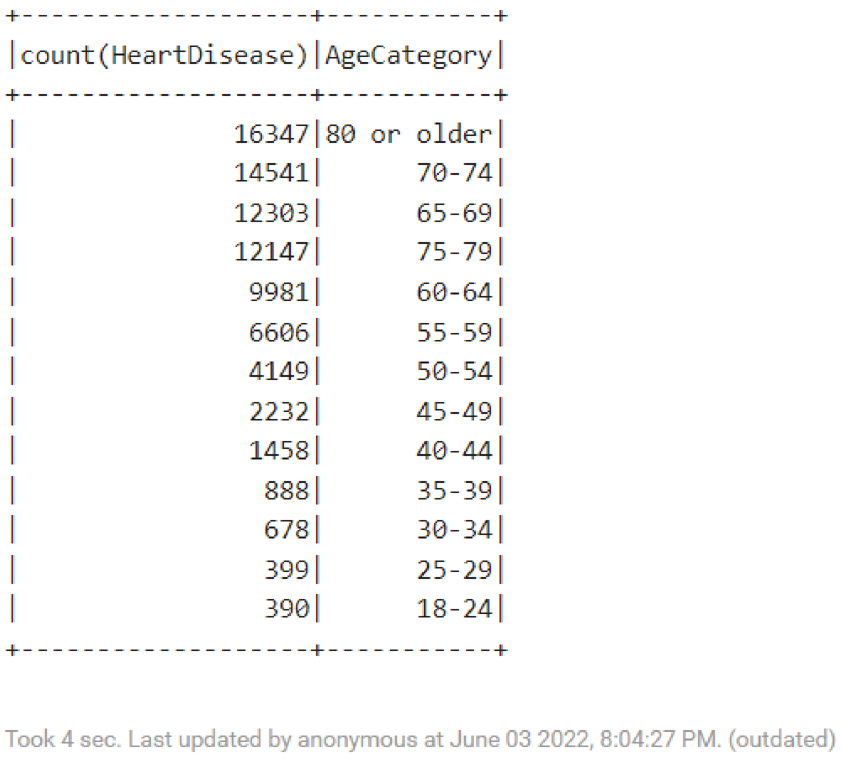
Took 6 sec. Last updated by anonymous at June 03 2022, 10:24:59 PM.

Analysis & Conclusion

Let’s answer the questions we posed in the beginning of the report.

1. **Are the chances of Heart Disease higher in a certain Age category?**

As can be seen from this output, Age Categories over 55+ have higher number of heart disease patients it means AgeCategory has an impact on number of heart disease patients.

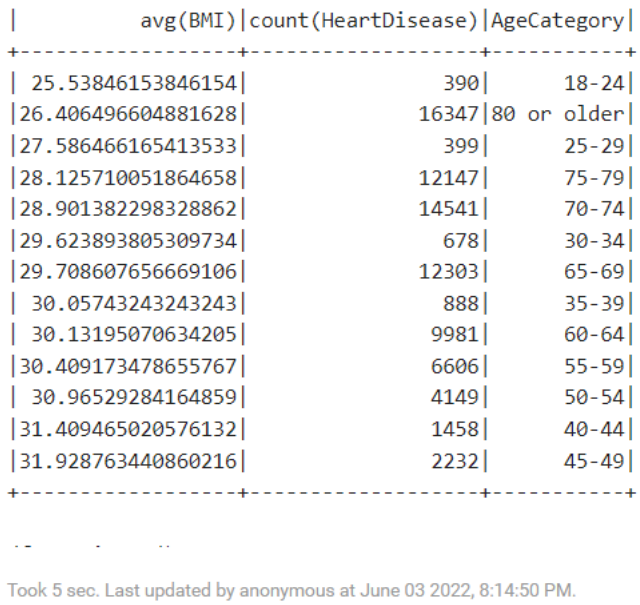


```
+-------------------+-----------+
|count(HeartDisease)|AgeCategory|
+-------------------+-----------+
|              16347|80 or older|
|              14541|      70-74|
|              12303|      65-69|
|              12147|      75-79|
|               9981|      60-64|
|               6606|      55-59|
|               4149|      50-54|
|               2232|      45-49|
|               1458|      40-44|
|                888|      35-39|
|                678|      30-34|
|                399|      25-29|
|                390|      18-24|
+-------------------+-----------+
```

Took 4 sec. Last updated by anonymous at June 03 2022, 8:04:27 PM. (outdated)

2. **What is the average BMI of Heart Patients in each Age Category?**

As can be seen from this output, the highest BMI is 32 for a heart disease patient who is 45-49 years old.

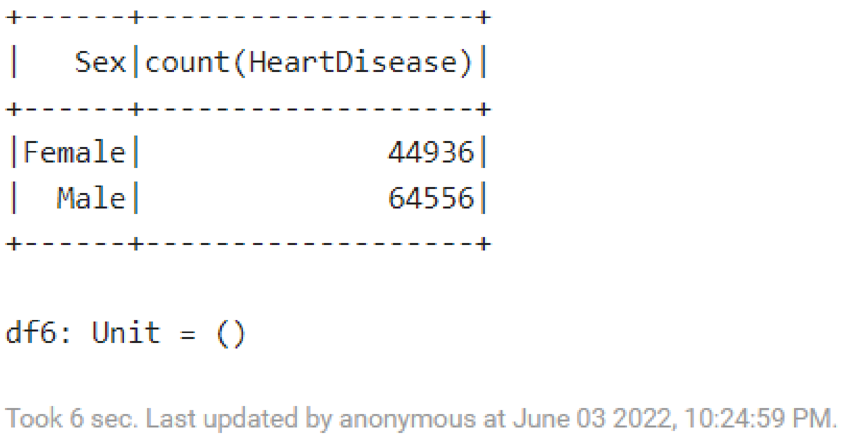


```
|          avg(BMI)|count(HeartDisease)|AgeCategory|
+------------------+-------------------+-----------+
| 25.53846153846154|                390|      18-24|
|26.406496604881628|              16347|80 or older|
|27.586466165413533|                399|      25-29|
|28.125710051864658|              12147|      75-79|
|28.901382298328862|              14541|      70-74|
|29.623893805309734|                678|      30-34|
|29.70860765669106|              12303|      65-69|
| 30.05743243243243|                888|      35-39|
| 30.13195070634205|               9981|      60-64|
|30.409173478655767|               6606|      55-59|
| 30.96529284164859|               4149|      50-54|
|31.409465020576132|               1458|      40-44|
|31.928763440860216|               2232|      45-49|
+------------------+-------------------+-----------+
```

Took 5 sec. Last updated by anonymous at June 03 2022, 8:14:50 PM.

3. Is there a correlation between Heart Diseases and Gender?

There is no such huge correlation as the % of which males and females present in the dataset is almost same.

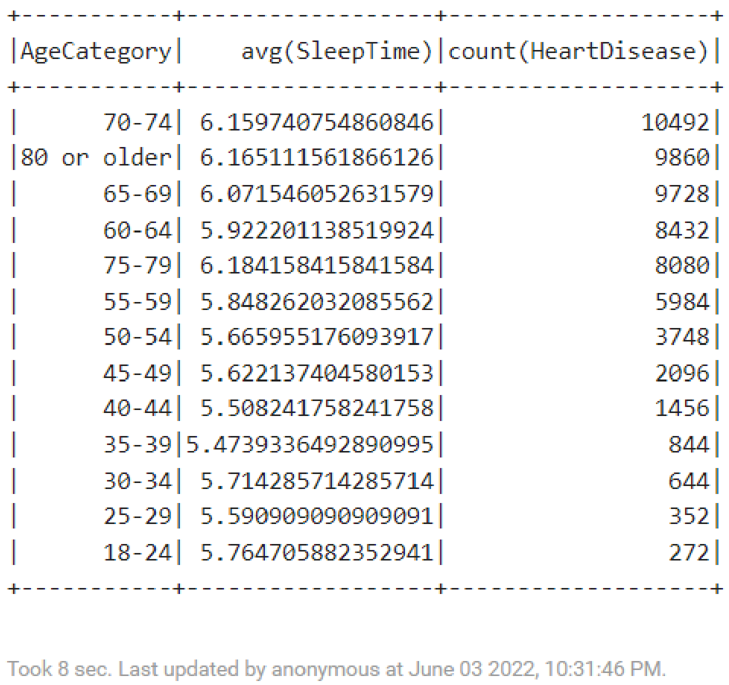


```
+------+-------------------+
|   Sex|count(HeartDisease)|
+------+-------------------+
|Female|              44936|
|  Male|              64556|
+------+-------------------+

df6: Unit = ()
```

Took 6 sec. Last updated by anonymous at June 03 2022, 10:24:59 PM.

4. What is the average number of sleeping hours less than 8 in heart patients of each age category?

As the output displays, for heart patients who sleep less than 8 hours count of heart diseases are higher.



```
+-----------+------------------+-------------------+
|AgeCategory|    avg(SleepTime)|count(HeartDisease)|
+-----------+------------------+-------------------+
|      70-74| 6.159740754860846|              10492|
|80 or older| 6.165111561866126|               9860|
|      65-69| 6.071546052631579|               9728|
|      60-64| 5.922201138519924|               8432|
|      75-79| 6.184158415841584|               8080|
|      55-59| 5.848262032085562|               5984|
|      50-54| 5.665955176093917|               3748|
|      45-49| 5.622137404580153|               2096|
|      40-44| 5.508241758241758|               1456|
|      35-39|5.4739336492890995|                844|
|      30-34| 5.714285714285714|                644|
|      25-29| 5.590909090909091|                352|
|      18-24| 5.764705882352941|                272|
+-----------+------------------+-------------------+
```

Took 8 sec. Last updated by anonymous at June 03 2022, 10:31:46 PM.

To conclude, all of these exploratory analysis can help doctors in providing a diagnosis within a few minutes which will save cost and prevent death cases through Heart Disease risk.