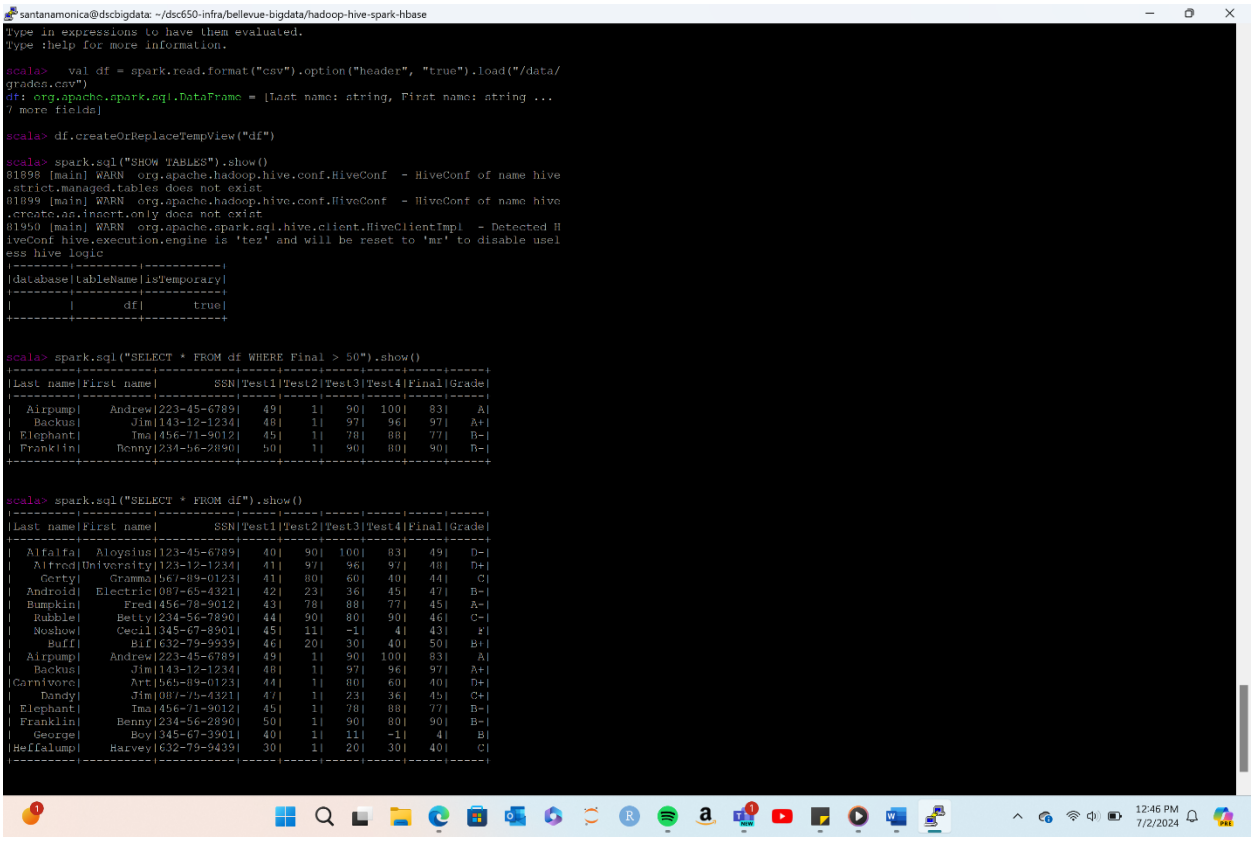
Monica Santana – Exercise 5

Scala Commands



SparkSQL Scala Command's 1, 2 & 3

Query #1: Max quantity of grades from Test 1 results, order by from least to greatest grade result

CODE:

```
spark.sql("SELECT Test1 AS max FROM df GROUP BY Test1 ORDER BY Test1").show()
```

Query #2: Count of grades from Test 2 column results

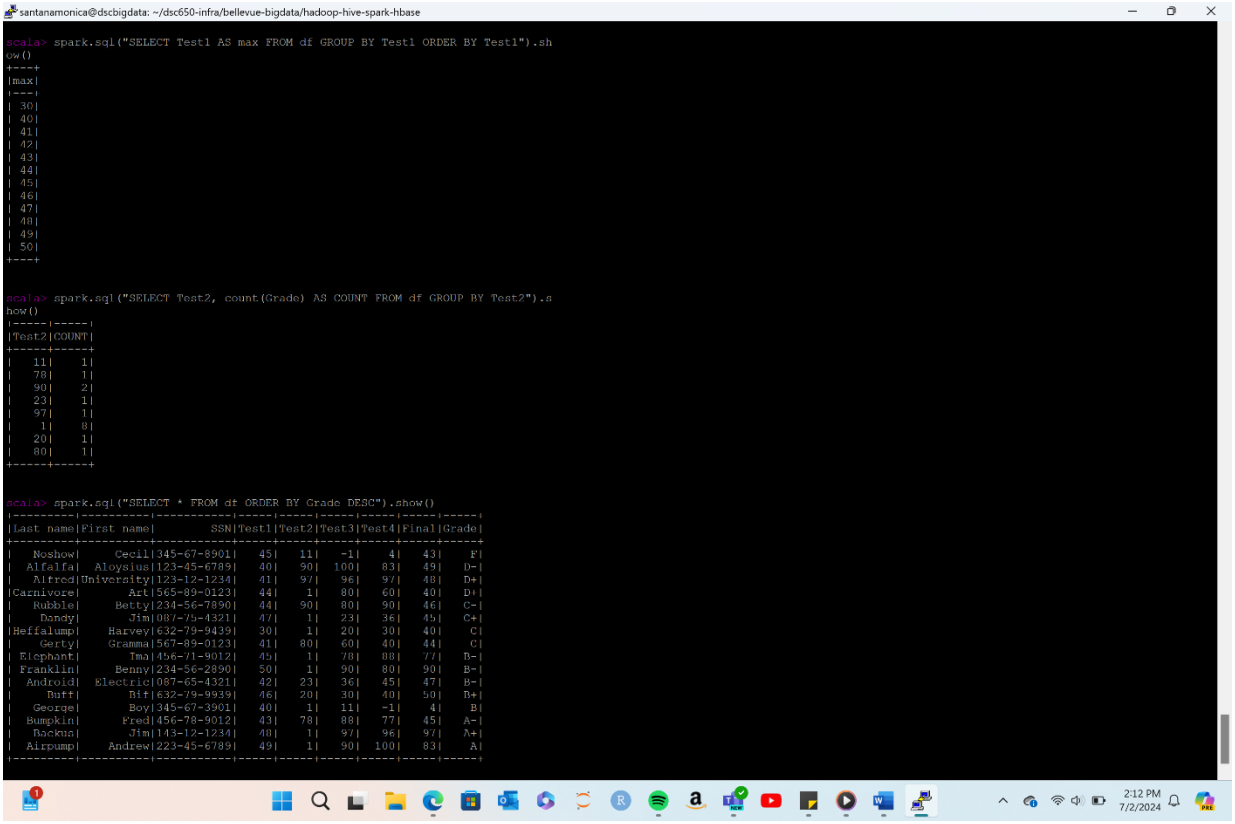
CODE:

```
spark.sql("SELECT Test2, count(Grade) AS COUNT FROM df GROUP BY Test2").show()
```

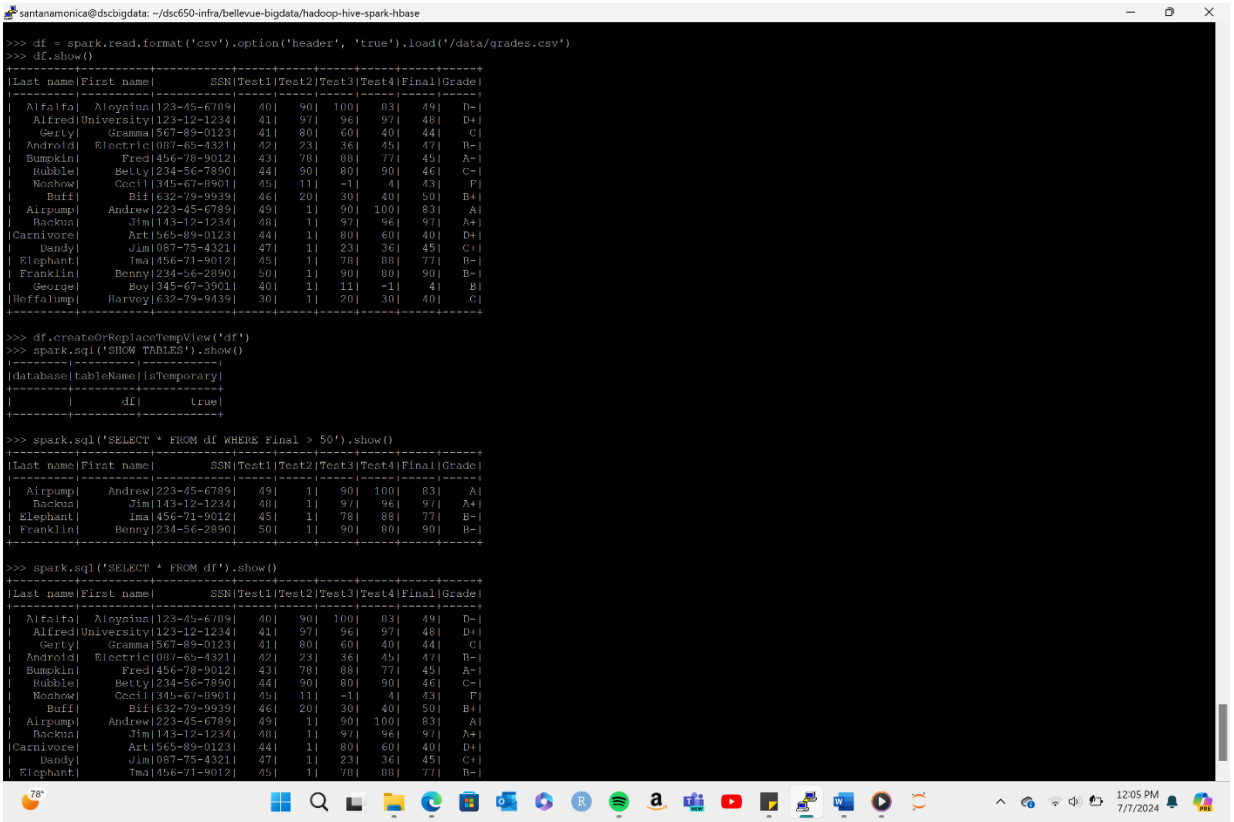
Query #3: Order the grade column in descending order so F grade will appear at the top

CODE:

```
spark.sql("SELECT * FROM df ORDER BY Grade DESC").show()
```



PySpark Commands



SparkSQL PySpark Commands 1, 2 & 3

Query #1: Max quantity of grades from Test 1 results, order by from least to greatest grade result

CODE:

spark.sql('SELECT Test1 AS max FROM df GROUP BY Test1 ORDER BY Test1').show()

Query #2: Count of grades from Test 2 column results

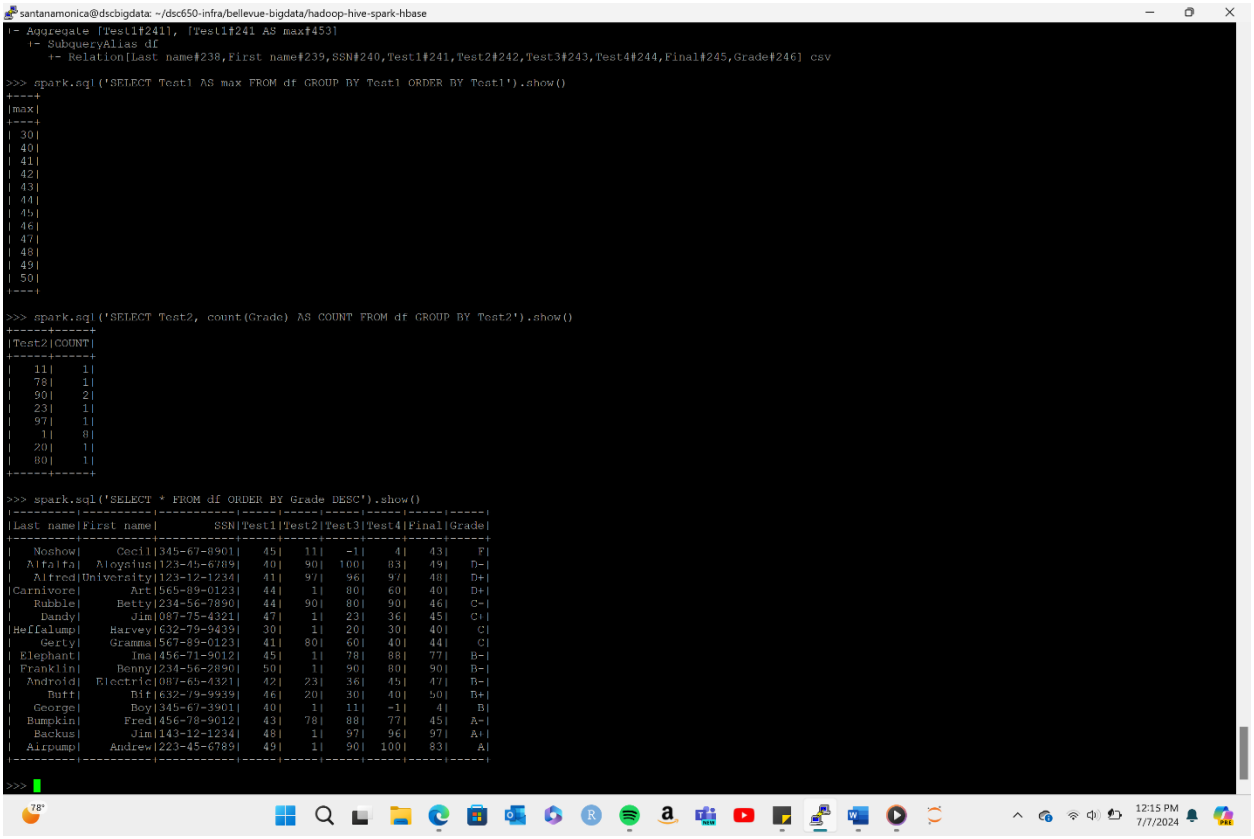
CODE:

spark.sql('SELECT Test2, count(Grade) AS COUNT FROM df GROUP BY Test2').show()

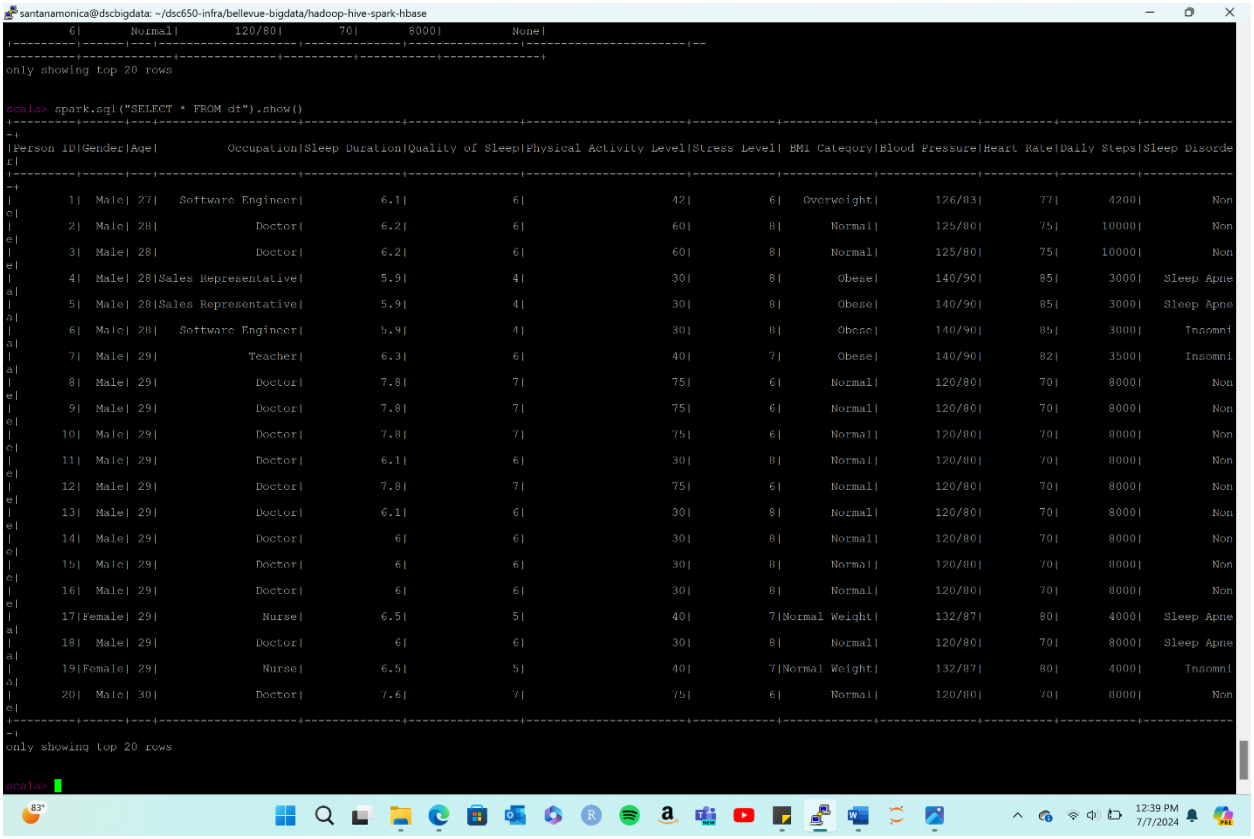
Query #3: Order the grade column in descending order so F grade will appear at the top

CODE:

spark.sql('SELECT * FROM df ORDER BY Grade DESC').show()



SparkSQL with custom dataset: Sleep dataset, diagnoses of sleep apnea, insomnia or no disorder and features used to diagnose such as demographics and medical information / history



SparkSQL Scala Command's 1, 2 & 3 with custom dataset

Query #1: Max quantity of occupation titles from occupation column results, and order by to make it alphabetical order

CODE:

spark.sql("SELECT Occupation AS max FROM df GROUP BY Occupation ORDER BY Occupation").show()

Query #2: Count of how many people in the dataset per occupation from occupation column results

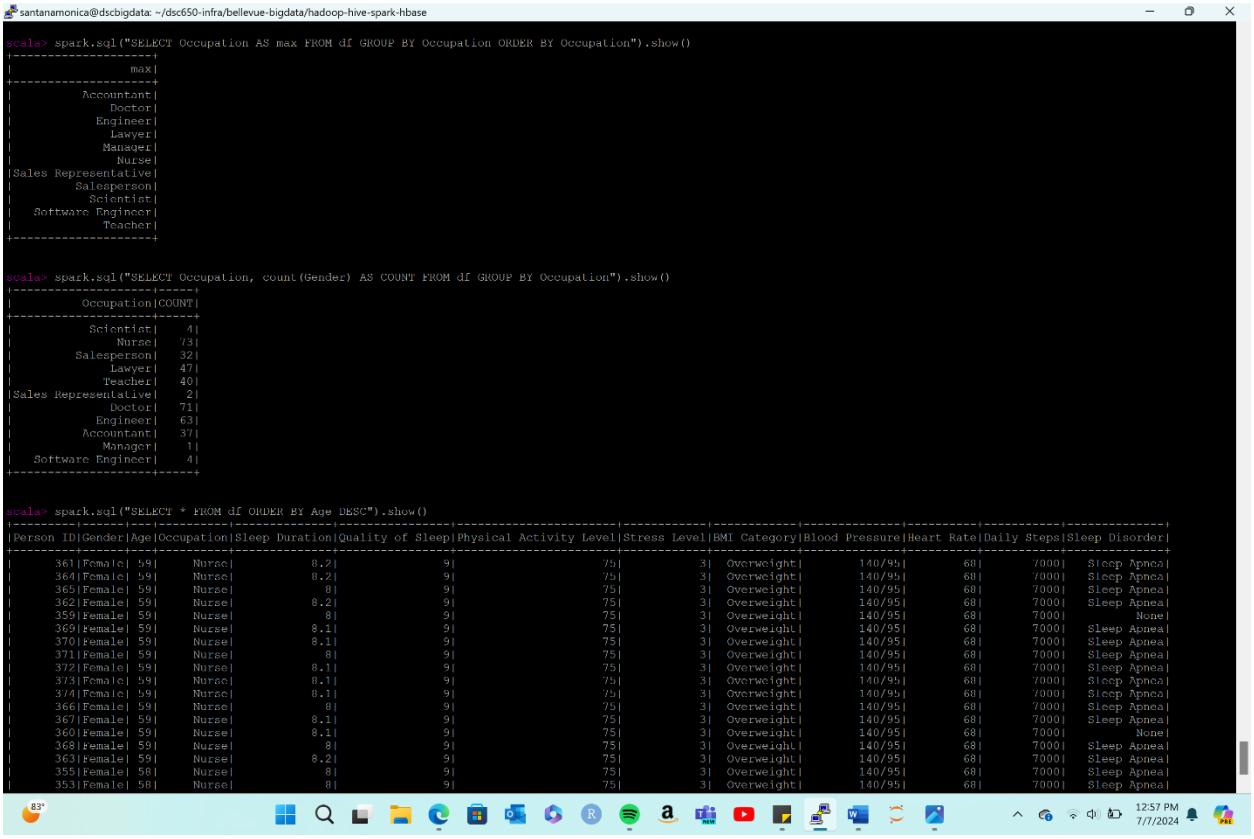
CODE:

spark.sql("SELECT Occupation, count(Gender) AS COUNT FROM df GROUP BY Occupation").show()

Query #3: Order the age column in descending order so the oldest people in the dataset (59) will appear at the top and descend from there

CODE:

spark.sql("SELECT * FROM df ORDER BY Age DESC").show()



The main difference I see using SparkSQL compared to DataFrames is the efficiency. The dataset I chose is not a small one, and creating a dataframe with it took longer than uploading it in SparkSQL. Even the commands outputs where the entire dataset would be reordered, appeared in rapid speed. Granted it was just the first 20 rows, but I know it still would appear faster than running the dataframe transformations in jupyter notebook for example with the same head(20) rows.