GROUP X

FINAL PROJECT COMPLETE PAPERWORK

HOUSEHOLD ENERGY CONSUMPTION ANALYTICS

ALY6110 DATA MANAGEMENT AND BIG DATA

CRN: 72145

Done By

Anish Nitin Somaiah, NUID: 001894328, somaiah.a@husky.neu.edu

Vignesh Thulasi Dass, NUID: 001385745, thulasidass.v@husky.neu.edu

Aditi Mayurkumar Patel, NUID: 001893349, patel.adit@husky.neu.edu

Shubhi Singal, NUID: 001359286, singal.shu@husky.neu.edu

Instructor: Mr. Valeriy Shevchenko



**INTRODUCTION**

In today’s modern world, every single phase of life has an involvement of energy, starting from our bodies internal kinetics to energy that powers cities. But companies making money out of energy production always considered the amount they produce and the amount they supply as key factor for business growth. After the advent of Big Data Analytics, today’s organizations are able to streamline production and optimize ROI based on **consumption analytics**.

**PROJECT OBJECTIVE**

Our project is similar to the mentioned consumption analytics. We **analyze the amount of energy consumed in a household which is given to us as a timeseries and our objective is to derive patterns from the obtained real time data.**

The application of this project will help any energy providing company to produce and supply based on customer demand during specific time periods to avoid wastage.

This can also be applied by a normal consumer to understand his/her own household’s energy patterns which will eventually help towards energy conservation and bill reduction.

**DATASET DESCRIPTION**

**Dataset Source**:

<http://archive.ics.uci.edu/ml/datasets/Individual+household+electric+power+consumption>

**Dataset Information:**

This archive contains 2075259 measurements gathered in a house located in Sochaux (7km of Paris, France) between December 2006 and November 2010 (47 months) and 9 features as mentioned below.

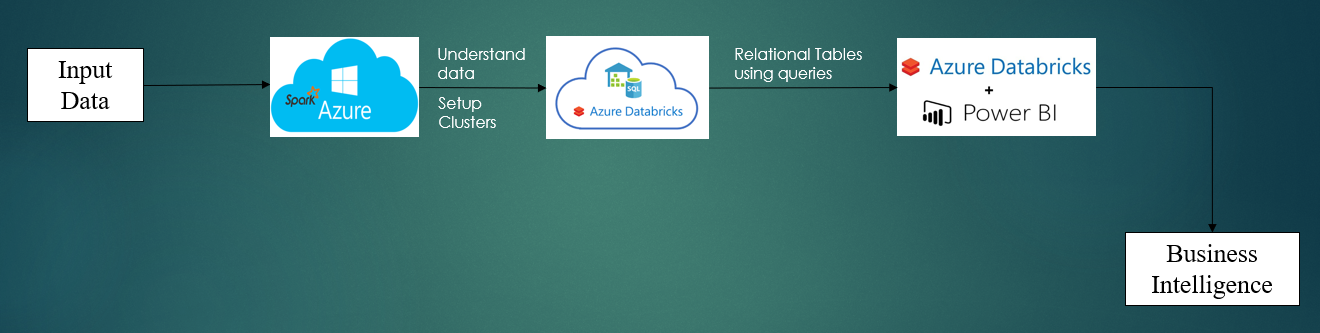
**Note**: **The dataset contains some missing values in the measurements (nearly 0.25% of the rows). All calendar timestamps are present in the dataset but for some timestamps, the measurement values are missing, and a missing value is represented by the absence of value between two consecutive semi-colon attribute separators**.

**Attribute Information:**

1. date: Date in format dd/mm/yyyy
2. time: time in format hh:mm:ss
3. global\_active\_power: household global minute-averaged active power (in kilowatt). Used by low load devices such as bulbs, fans etc.
4. global\_reactive\_power: household global minute-averaged reactive power (in kilowatt). Surplus power not used at the time period.
5. voltage: minute-averaged voltage (in volt)
6. global\_intensity: household global minute-averaged current intensity (in ampere)
7. sub\_metering\_1: energy sub-metering No. 1 (in watt-hour of active energy). It corresponds to the kitchen, containing mainly a dishwasher, an oven and a microwave (hot plates are not electric but gas powered).
8. sub\_metering\_2: energy sub-metering No. 2 (in watt-hour of active energy). It corresponds to the laundry room, containing a washing-machine, a tumble-drier, a refrigerator and a light.
9. sub\_metering\_3: energy sub-metering No. 3 (in watt-hour of active energy). It corresponds to an electric water-heater and an air-conditioner.

**DATA ANALYTICS PROCESS WORKFLOW**

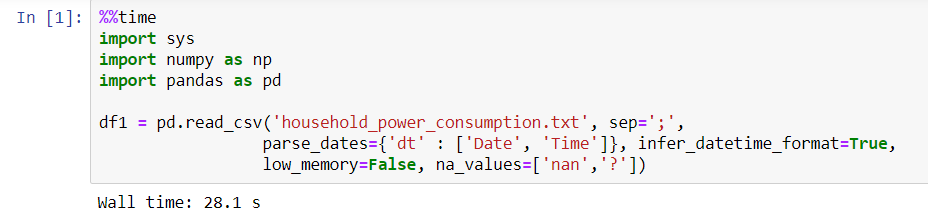
* Import data into Databricks Azure.
* Import data as RDD and clean using PySpark and RDD functions.
* Establish Relational Tables by Querying using Spark SQL.
* Connect Query Tables to Power BI using JDBC line.
* Visualize using Power BI for Business Intelligence.



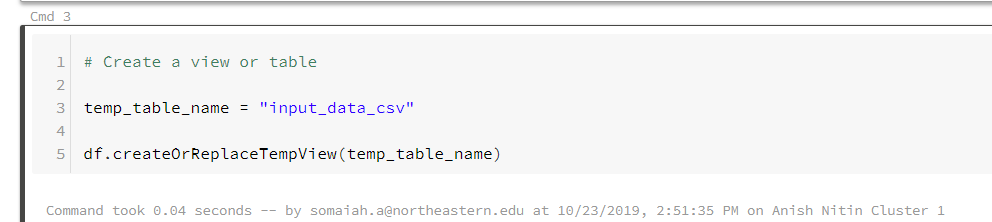
***Figure 1: Data Process workflow***

**STEP 1: WHY DATABRICKS AZURE**

The dataset was imported using Jupyter notebook at the beginning to check the capability of the local system and then to data bricks Azure to understand the most important feature of big data processing system i.e. scalability and a significant **30 second time difference** was observed with Azure being faster than Jupyter Notebook. This is because Spark in Data Bricks Azure uses a **Directed Acyclic Spark Streaming** service which is much faster than the traditional T-SQL DW connectors used in Jupyter Notebooks.



*Figure 2: Data import time in Jupyter*

**

*Figure 3: Data import time in Azure*

**Note that since date and time were given in European format, it was converted to timestamp by merging date and time as shown in figure 2.**

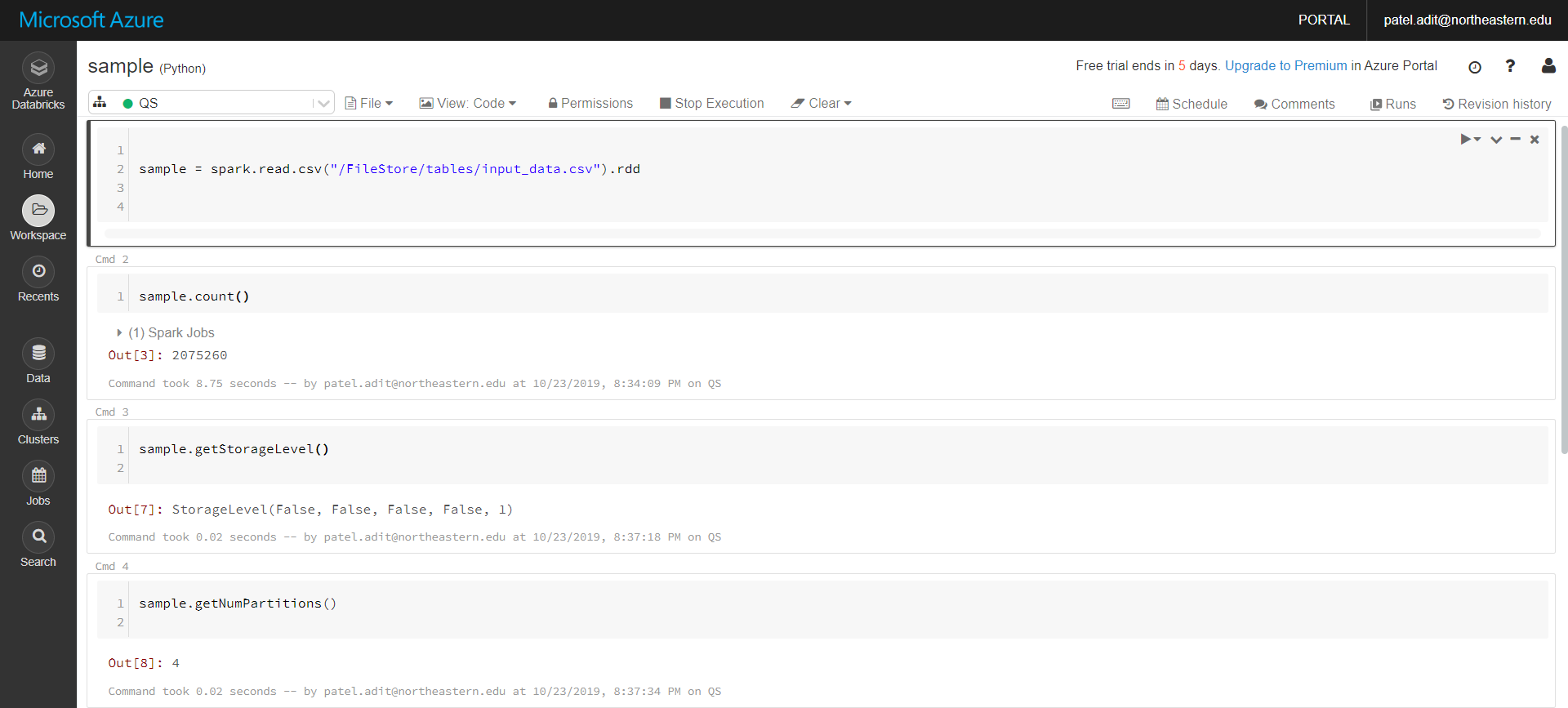
**STEP 2: PySpark and RDD.**

In this step data is imported as an RDD (Resilient Distributed Data) into the Azure system as RDD have several advantages compared to the regular data frames such as **Lazy Evaluation, Fault Tolerance, In Memory Computations and Immutability.**

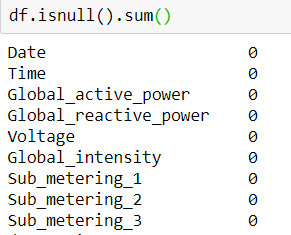
Some of the Lazy evaluation that has been done after importing the data frame as RDD are

1. Check number of rows. (2075260)
2. Check type of storage level.(in memory processing with output False False False False 1)
3. Check number of partitions. (in automated scaling output was 4).

Coming to data cleaning, as 0.25% of rows had null values, the rows were dropped as it would not have any adverse effects over the analysis and after removal, the dataset was rechecked for null values using the df.isnull().sum() command under PySpark.



*Figure 4: Lazy evaluation using RDD functions*

**

*Figure 5: Data cleaned and rechecked for null values.*

**STEP 3: Relational Tables by Querying using SparkSQL**

In order to answer the business objective, SparkSQL was used to make different tables using queries. Some of the queries are as below

1. **Yearly average total power consumption trend**.

%sql

select Year(dt),AVG(Global\_active\_power) from `input\_data1\_csv` group by Year(dt) order by Year(dt)

1. **Distribution of different power consumption meters**.

%sql

select Month(dt),AVG(Global\_active\_power),AVG(Sub\_metering\_1),AVG(Sub\_metering\_2),Avg(Sub\_metering\_3) from `input\_data1\_csv` group by Month(dt) order by Month(dt)

1. **Monthly consumption pattern and comparison Year-wise**

%sql

select Month(dt),Avg(Global\_active\_power) from `input\_data1\_csv` where Year(dt)="2007" group by Month(dt) order by Month(dt)

Union

select Month(dt),Avg(Global\_active\_power) from `input\_data1\_csv` where Year(dt)="2010" group by Month(dt) order by Month(dt)

Union

select Month(dt),Avg(Global\_active\_power) from `input\_data1\_csv` where Year(dt)="2009" group by Month(dt) order by Month(dt)

1. **Daily consumption analysis for one month**

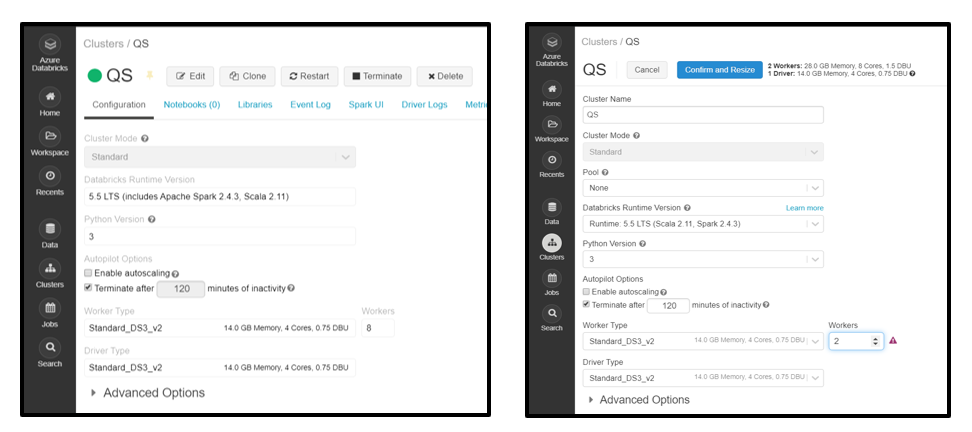
%sql

select day(dt),Avg(Global\_active\_power) from `input\_data1\_csv` where Year(dt)="2008" and Month(dt)= 12 group by day(dt) order by day(dt)

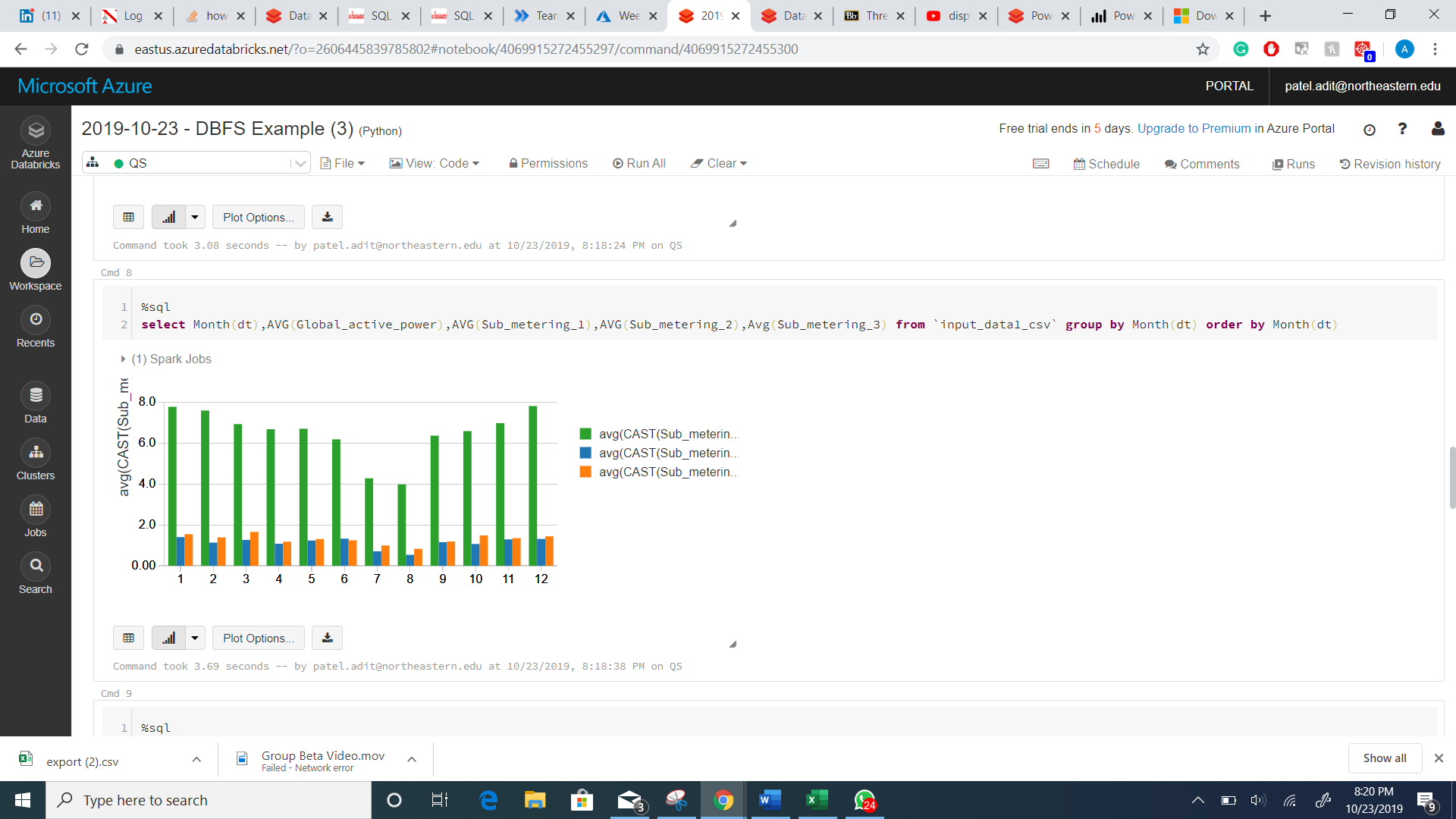
**NOTE: The %sql line is used in every query to declare Azure that the language used is SQL as the notebook was initially generated with Python as API’s compiler. This is how multiple languages are supported in Data Bricks Azure API.**

**STEP 4: Cluster Management**

As mentioned earlier, scalability is one of the most important features of Azure or any big data system. Hence here, different cluster numbers were manually assigned to check the performance of the big data system. It has been observed that **queries took the same time** for both minimum and maximum number of nodes. Hence for our case number of nodes can be tuned to **ensure fault tolerance** purposes.



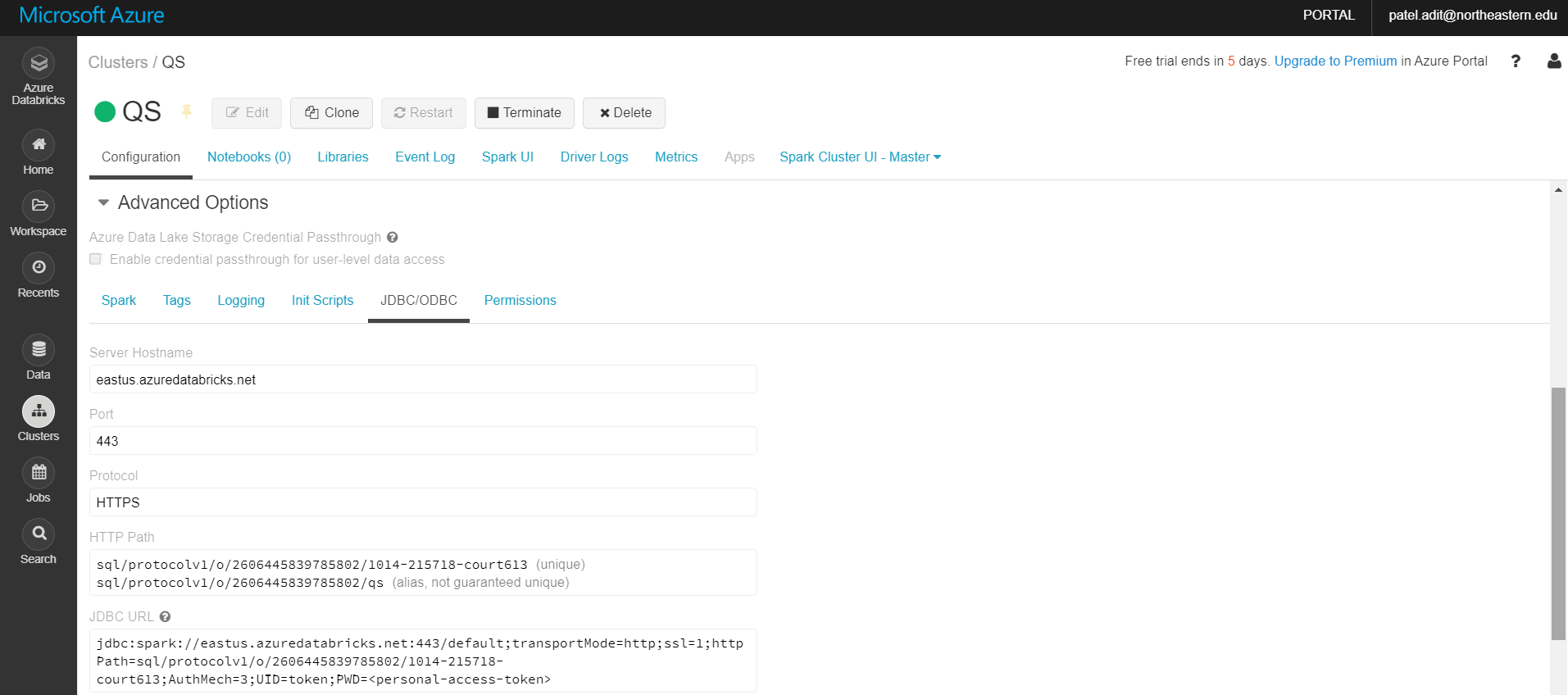
*Figure 6: Cluster management under Data Bricks Azure*



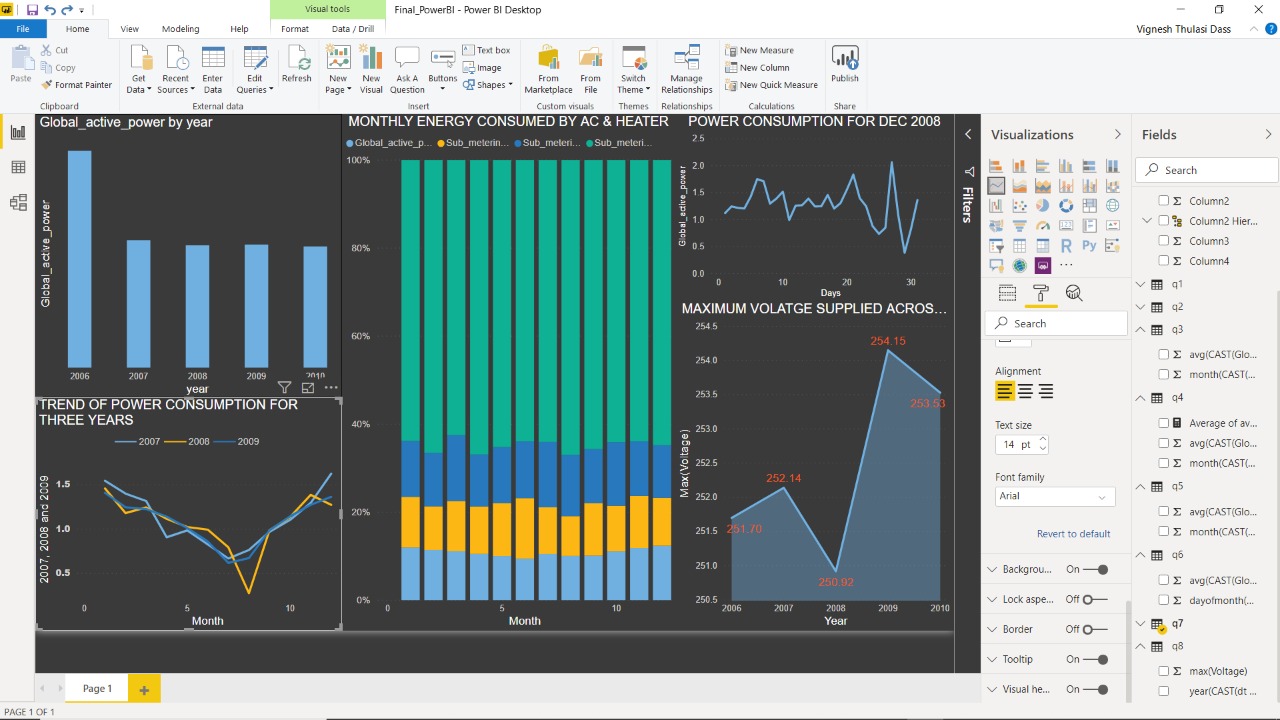
*Figure 7: Sample SparkSQL Query and implementation.*

**STEP 5: PowerBI Integration**

As shown in figure 7, the query part was easy but the visualization that came out of it is difficult to interpret. Hence PowerBI integrated under Databricks Azure is used to visualize the relational tables into comprehensive visualizations to derive meaningful insights from the data. So for connecting Power BI to Azure Databricks clusters, the in-built Spark connector that uses the **JDBC link** of the cluster generated is utilized.

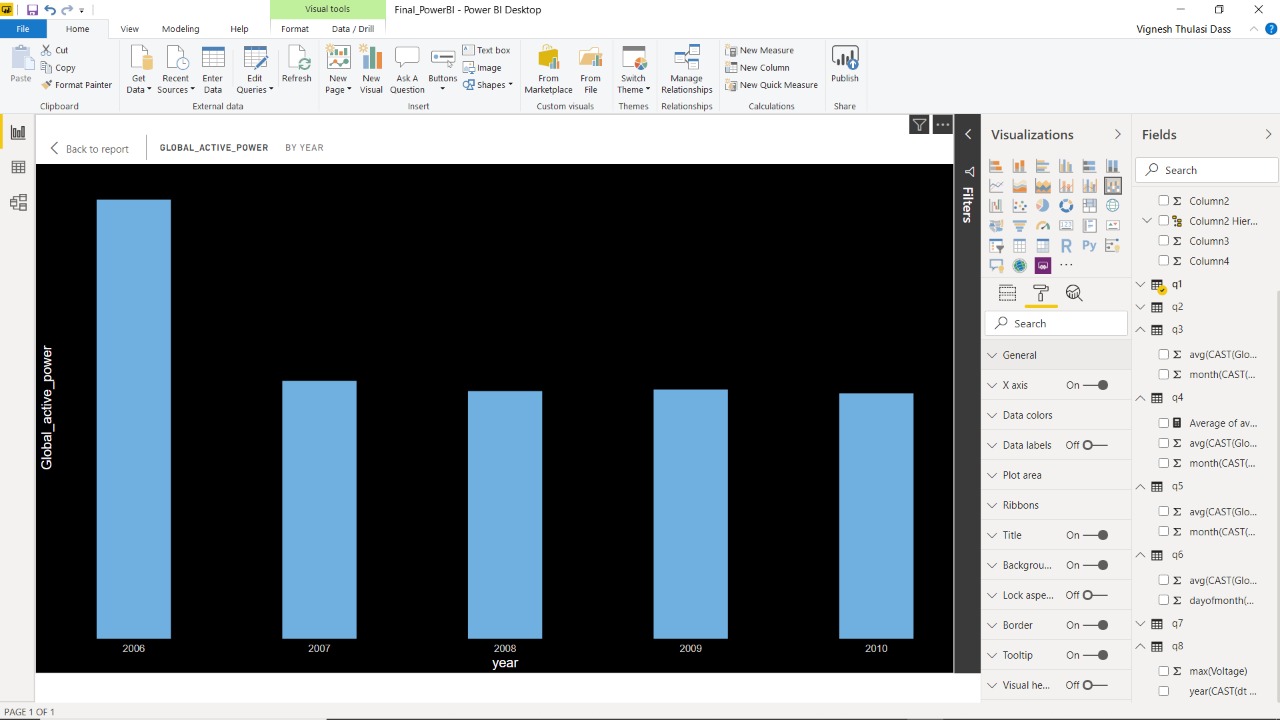


*Figure 8: JDBC link extraction for cluster to integrate PowerBI*

**

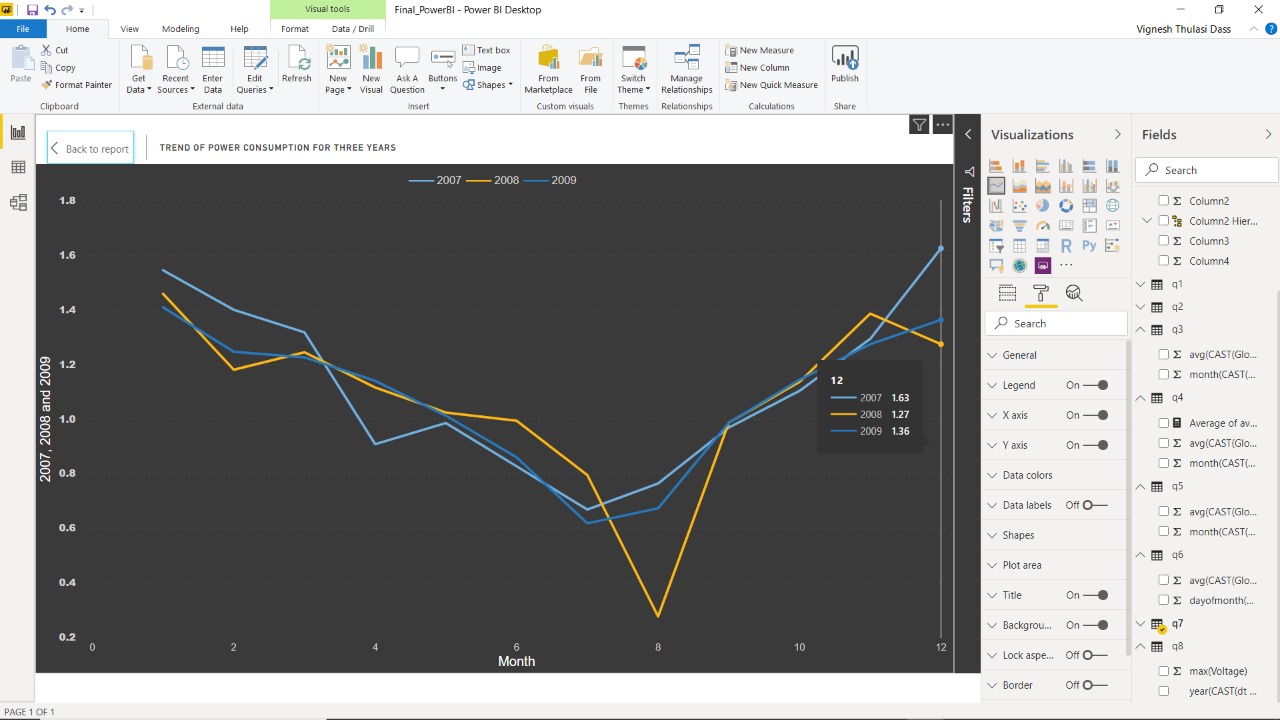
*Figure 9: PowerBI Dashboard*

**BUSINESS CONCLUSIONS**

****

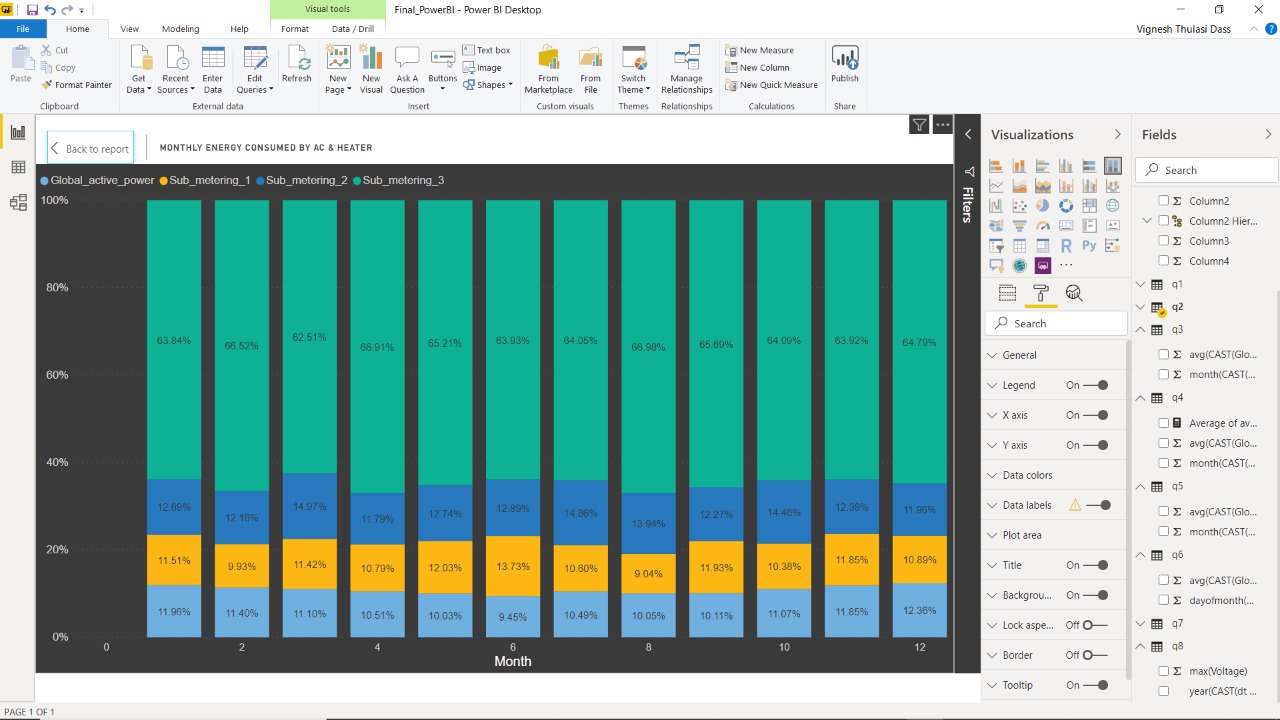
*Figure 10: Average Global Active Power Consumption Trend Yearly*

* **Yearly Trend of Active power consumption shows that the average power consumed remained the same throughout the years**
* **Hence a distributor has no necessity to increase supply as demand remained the same. Note that 2016 has a spike as December month data alone was present for 2016**



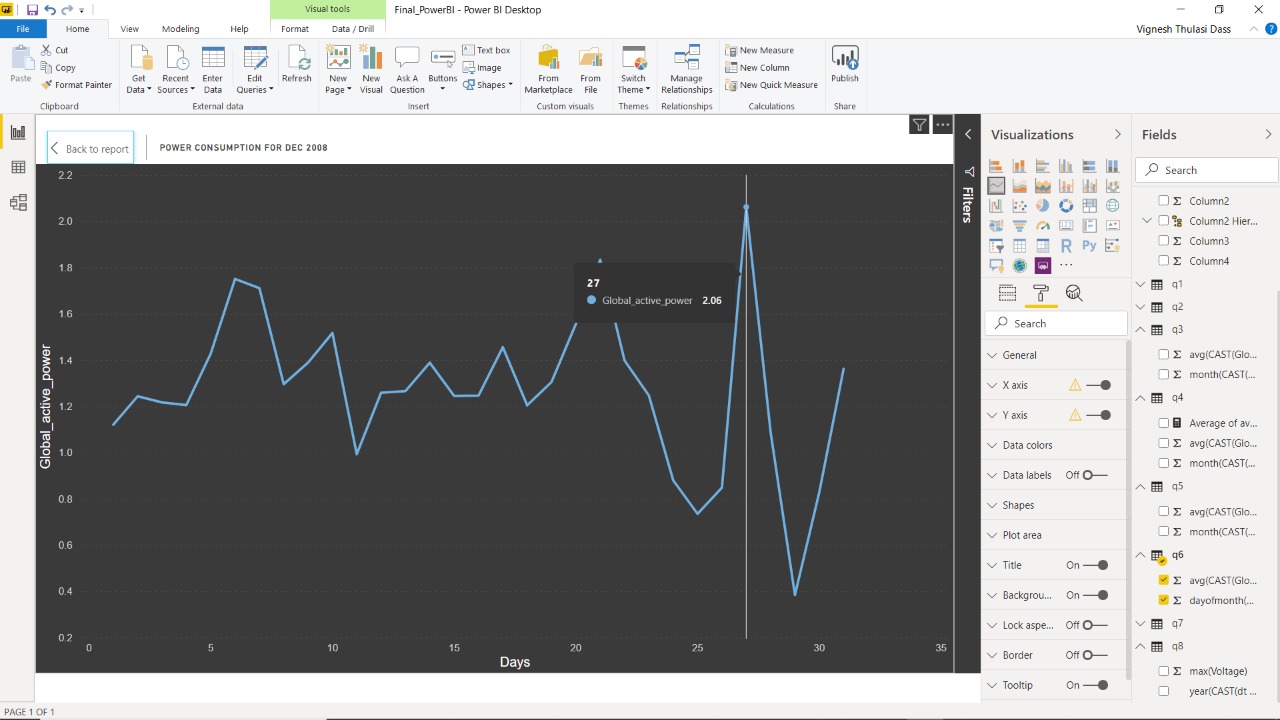
*Figure 11: Average Global Active Power Consumption Monthly Trend comparison for 3 years*

* **As from the trend, active power consumption mildly decreases in summer months and sharply increases in winter months.**
* **This is because during summer days low load consumption is less as daylight is more and later when winter starts, low load power consumption increases.**

****

*Figure 12: Distribution Analytics of Power Consumption Channels*

* **This Distribution analysis clearly tells that throughout in the year, highest electricity reading is observed in the meter designated to ac and water heaters.**
* **This is a default case, but the graph would suggest a supplier to channelize more energy towards Ac and heater channels and also make necessary hardware improvements for regulatory and damage control purposes.**

****

*Figure 13: Daily Global Active Power Consumption Trend per month.*

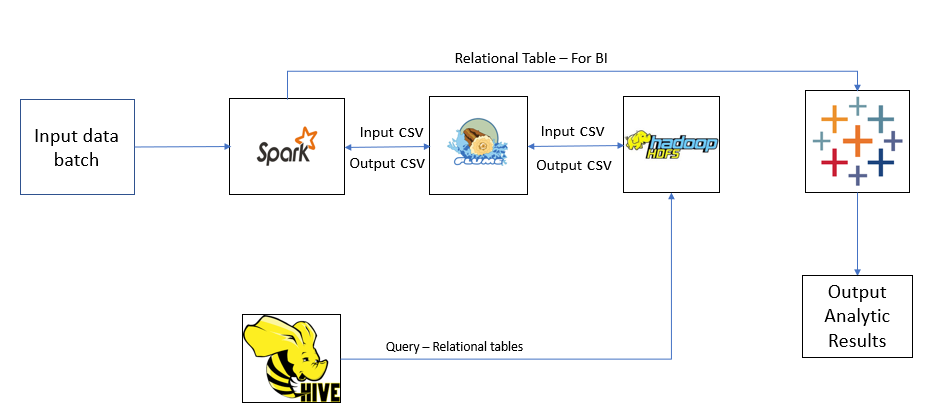
* **A specific month’s (December) consumption data has been selected using SQL query for a consumer to understand usage pattern.**
* **It was observed that there are regular spikes in consumptions during weekends***.* **A huge spike is also observed during Christmas Days.**

**CONCLUSIONS ON PROJECT**

* **Databricks Azure was utilized for the project as it provides multi language API’s for data analytics.**
* **Databricks Azure was also incorporated for the project as it provides integration to tools such as Jupyter Notebook, PowerBI and so on.**
* **Databricks Azure facilitated in cluster management for scalable data processing.**
* **Apache Spark was used for data analytics over MapReduce as Apache Spark is faster than MapReduce and it performs in memory computations.**

**FUTURE SCOPE**

* Tune a big data system to obtain real time data and perform time series forecasting based on trend and seasonality.
* Perform Machine Learning Clustering Model for the given data set using SPARKML to identify clusters based on usage patterns which can be used to segregate supply based on the clusters.
* The scope of this project can be extended not only to electricity generators but also to power plants which produce energy through different sources such as crude oil, wind, water and renewable resources. This can help in sustainable usage of non-renewable energy sources which can be made everlasting and accessible for a longer period of time for any energy production and consumption unit.



*Figure 14: Future Scope Big Data System Architecture*

**REFERENCE:**

*Azure Databricks Quickstart (October 2019)***,** Documentation Retrieved from**,** [**https://docs.azuredatabricks.net/getting-started/quick-start.html**](https://docs.azuredatabricks.net/getting-started/quick-start.html)

*Azure Blob Storage (October 2019),*Documentation Retrieved from**,** [**https://docs.databricks.com/data/data-sources/azure/azure-storage.html**](https://docs.databricks.com/data/data-sources/azure/azure-storage.html)

**Epydoc(November 2014),** *RDD Functions in Spark***,** Documentation Retrieved from**,** [**https://spark.apache.org/docs/1.1.1/api/python/pyspark.rdd.RDD-class.html**](https://spark.apache.org/docs/1.1.1/api/python/pyspark.rdd.RDD-class.html)

**Joshua Daniel Rhodes (May 2014),** *Optimal residential energy consumption, prediction, and analysis***,** Retrieved from**,** [**https://repositories.lib.utexas.edu/handle/2152/33342**](https://repositories.lib.utexas.edu/handle/2152/33342)