**Health Insurance Marketplace & Credit Card System Documentation**

Following Files are included in the project

1. Python file (Main program) – HealthInsuranceMarketPlace.py
2. Jupyter file(Visualization file) – HMI\_Visual.ipynb
3. Sql file – cdw\_sapp.sql
4. Requirement document – System Requirements Document pdf
5. Data source and description document – Dataset Description and information Document.pdf
6. Mapping Document – Description of CreditCard table.

**Part 1 - Extracting & Transforming data from SQL & loading to MongoDB**

*\*\*\*To prep the data for Part 1*

**Step 1**: Load the attached ‘cdw\_sapp.sql’ in HeidiSQL 9.5.0.5196 file.

* Upon loading the file, a database by name of ‘cdw\_sapp’ should be created with the following three tables:
  + cdw\_sapp\_branch
  + cdw\_sapp\_creditcard
  + cdw\_sapp\_customer

*\*\*\*To extract the data**to mongoDB*

**Step 2:**

* Run the python file, HealthInsuranceMarketplace.py
* Following options are displayed
* Choose option 1

Welcome!

1- Credit Card data from MariaDB

2- Health Insurance Marketplace

3- Quit

Enter your option:

**Note:** Option 1 will run 9 programs in linear sequence as shown below:

ETL process for each table shown above takes place in 3 steps/functions:

* cdw\_branch(), cdw\_branch\_t(), cdw\_branch\_mongo()
* cdw\_sapp\_creditcard(), cdw\_sapp\_creditcard\_t() , creditcard\_mongo()
* cdw\_sapp\_customer(), cdw\_sapp\_customer()\_t, cdw\_sapp\_mongo()

1st function- cdw\_branch() – The first function establishes a JDBC connection to import data from ‘cdw\_sapp\_branch’ table in MariaDB to Spark. The function returns a DataFrame.

2nd function- cdw\_branch\_t() – The second function is used for transforming the data and assigning column to the data. This is accomplished by creating a temporary view using ‘createOrReplaceTempView’ which allow SQL transformation function and queries to be executed.

3rd function- cdw\_branch\_mongo () – In the final step of the extraction process, a spark session is created that connects to MongoDB, a collection is created in MongoDB database and the transformed data is written into this collection.

The above 3 steps are repeated for each of the 3 tables in MariaDB database, i.e. cdw\_sapp\_branch, cdw\_sapp\_creditcard, cdw\_sapp\_customer.

Once the transfer is complete, the main menu with 2 option is shown again

**Part 2 - Extracting & Transforming data from website and storing in MongoDB using Spark and Kafka**

Welcome!

1- Credit Card data from MariaDB

2- Health Insurance Marketplace

3- Quit

Enter your option:

**Note:** Option 2, provide the following sub options:

Select from the following options:

1. Benefit Cost,

2. Network,

3. Service Area,

4. Insurance,

5. Plan Attribute,

6. Quit

Option 2, provide 5 topics that the user can extract and broadcast as a Kafka topic, and transform and save in MongoDB using spark.

Following are the url of the data for each topic saved in a raw format:

1. BenefitCostSharing.txt - **Url** :

https://raw.githubusercontent.com/platformps/Healthcare-Insurance-Data/master/BenefitsCostSharing

\_partOne.txt

https://raw.githubusercontent.com/platformps/Healthcare-Insurance-Data/master/BenefitsCostSharing

\_partTwo.txt

https://raw.githubusercontent.com/platformps/Healthcare-Insurance-Data/master/BenefitsCostSharing

\_partThree.txt

https://raw.githubusercontent.com/platformps/Healthcare-Insurance-Data/master/BenefitsCostSharing

\_partFour.txt

1. Network.csv - Url –

<https://github.com/platformps/Healthcare-Insurance--Data/blob/master/Network.csv>

1. ServiceArea.csv - **Url** :

https://raw.githubusercontent.com/platformps/Healthcare-Insurance-Data/master/ServiceAre

a.csv

1. Insurance.txt - **Url:**

[**https://raw.githubusercontent.com/platformps/Healthcare-Insurance-Data/master/insurance.txt**](https://raw.githubusercontent.com/platformps/Healthcare-Insurance-Data/master/insurance.txt)

1. PlanAttributes.txt - **Url** :

https://raw.githubusercontent.com/platformps/Healthcare-Insurance-Data/master/PlanAttributes.csv

**Part 2 - Extracting & Transforming data from website and storing in MongoDB using Spark and Kafka**

**Step 1**: Choose any among the 5 topics shown in the sub menu to install.

Note: The sub menu was created due to limited processing capacity of the single laptop(node).

For each topic shown above there are 2 functions executed. The 1st function, i.e. kafka\_*(topic name)* extracts data from the GitHub website using Kafka’s request library and saves it in a list. A topic is created by the producer and each element in the list is streamed in ‘utf-8’ encode format.

The 2nd function, spark\_(topic name) creates a spark session that connects as a consumer to the kafka topic being broadcasted. A writestream in spark session is used to iterate through each row and assign a column to each element in a row and finally append the data to a MongoDB database’s collection.

The above step is repeated until all the topic are saved in MongoDB.

Once all the topic are saved in MongoDB, choose option 6 to exit out of the sub menu and then option 3 to exit of the program.

**Part 3 –** Following queries are plotted using pymongo in ‘HMI\_Visual.ipynb’ for data analysis.

a) Use “Service Area Dataset” from MongoDB. Find and plot the count

of ServiceAreaName, SourceName , and BusinessYear across the

country each state?

b) Use “Service Area Dataset” from MongoDB. Find and plot the count

of “ sources ” across the country.

c) Use the “Benefit-Cost Sharing” dataset from MongoDB. Display a

table of the names of the plans with the most customers by state, the

number of customers for that plan and the total number of customers.

( Hint: use Statecode, benefitName)

d) Use the “Benefit Cost Sharing” dataset from MongoDB. Find and

plot the number of benefit plans in each state.

e) Use the “Insurance” dataset from MongoDB and find the number of

mothers who smoke and also have children.

f) Use the “Insurance” dataset from MongoDB. Find out which region

has the highest rate of smokers. Plot the results for each region.