

BIG DATA ANALYSIS ON HEALTHCARE INSURANCE COMPANY

CS6003 – BIG DATA ANALYTICS

Submitted by

I Sheeba Grace (2020103047)

Swathi M (2020103053)

*in partial fulfillment of the requirements for the award of
the degree of*

BACHELOR OF ENGINEERING

in

**COMPUTER SCIENCE AND
ENGINEERING**



**COLLEGE OF ENGINEERING,
GUINDY**

**ANNA UNIVERSITY: CHENNAI 600 025
MAY 2023**

ANNA UIVERSITY: CHENNAI 600025

BONAFIDE CERTIFICATE

Certificate that this project request titled Analysis on Healthcare Insurance Company is the bonafide work of **I Sheeba Grace(2020103047)**, **Swathi M(2020103053)** who carried out the project work under my supervision, for the fulfillment of the requirements as part of the CS6003 – Big Data Analytics

Dr. S. Valli

HEAD OF THE DEPARTMENT

Professor

Department of Computer Science and
Engineering

College of Engineering Guindy
12, Sardar Patel Rd, Guindy
Chennai, Tamil Nadu - 600025

Dr. R. Arockia Xavier Annie

SUPERVISOR

Assistant Professor

Department of Computer Science and
Engineering

College of Engineering Guindy
12, Sardar Patel Rd, Guindy
Chennai, Tamil Nadu - 600025

ABSTRACT

Big data analytics in insurance companies involve the use of advanced technology and tools to analyze vast amounts of data generated within the insurance industry. The application of big data analytics in the insurance industry can help insurers to better understand their customers, provide more personalized insurance products, and improve the overall customer experience. The analysis of big data in insurance can also help to identify fraudulent claims, manage risk more effectively, and improve underwriting decisions. Additionally, the use of big data analytics can help insurers to optimize their operations and drive their growth. In our project, we carry out the analysis using various big data tools like Hadoop, Spark, Sqoop etc. Initially the preprocessed dataset is stored in Relational Database Management System (RDBMS). To do analysis the data in RDBMS is transferred to Hadoop using sqoop. After uploading the data in to HDFS we connect to spark. Here we analyze the data with help of python. Finally we are expected to get our desired result in tabular form and that result is used to visualize our use cases.

ACKNOWLEDGEMENT

Foremost, we would like to express our sincere gratitude to our project guide, **Dr.R.Arockia Xavier Annie**, Assistant Professor, Department of Computer Science and Engineering, College of Engineering Guindy, Chennai for her constant source of inspiration. We thank her for the continuous support and guidance which was instrumental in taking the project to successful completion.

We are grateful to **Dr. S. Valli**, Professor and Head, Department of Computer Science and Engineering, College of Engineering Guindy, Chennai for her support and for providing necessary facilities to carry out for the project.

We would also like to thank our friends and family for their encouragement and continued support. We would also like to thank the Almighty for giving us the moral strength to accomplish our task.

I SHEEBA GRACE

SWATHI M

TABLE OF CONTENTS

CHAPTER NO.	TITLE	PAGE NO.
	ABSTARCT	iii
1.	INTRODUCTION	1
	1.1 OBJECTIVES	1
	1.2 PROBLEM STATEMENT	1
	1.3 NEED FOR THE SYSTEM	2
	1.4 CHALLENGES IN THE SYSTEM	2
	1.5 SCOPE OF THE PROJECT	2
2.	LITERATURE SURVEY	3
	2.1 BIG DATA ANALYSIS	3
	2.2 BIG DATA APPLICATION	4
3.	SYSTEM DESIGN	5
	3.1 SYSTEM ARCHITECTURE	5
	3.2 SYSTEM REQUIREMENTS	5
4.	MODULE DESCRIPTION	8
	4.1 DATA PREPROCESSING	8
	4.2 STORAGE RETRIEVAL AND MANAGEMENT OF DATA	8

4.3 DATA INGESTION SERVICES	8
4.4 DATA STORAGE ,PROCESSING AND ANALYTICS SERVICES	9
5. RESULTS AND IMPLEMENTATION	10
5.1 DATA SOURCES	10
5.2 DATA PREPROCESSING	13
5.3 STORAGE RETRIEVAL AND MANAGEMENT OF DATA	22
5.4 DATA INGESTION SERVICES	27
5.5 DATA STORAGE,PROCESSING AND ANALYTICS SERVICES	39
6. CONCLUSION AND FUTURE WORK	58
6.1 CONCLUSION	58
6.2 FUTURE WORK	58
REFERENCES	59

CHAPTER 1

INTRODUCTION

A healthcare insurance company is seeking to boost its revenue and gain a better understanding of its customers. To achieve this, it plans to leverage the power of the Big Data Ecosystem to analyze competitors' data collected from various sources such as web scraping and third-party sources. The aim of this analysis is to track customer behavior and conditions, enabling the company to tailor insurance policy offers to individual needs and calculate royalties for prior policyholders. By doing so, the company hopes to increase its revenue and gain a better understanding of its customers to enhance its services.

1.1 OBJECTIVES

The goal of the project is to create data pipelines for the Health Care insurance company which will make the company make appropriate business strategies to enhance their revenue by analyzing customers behaviors and send offers and royalties to customers respectively.

1.2 PROBLEM STATEMENT

Healthcare companies face challenges in leveraging big data analysis to improve revenue and gain deeper insights into customer behavior. These organizations aim to harness the power of the Big Data Ecosystem to analyze vast amounts of data collected from various sources, including web scraping and third-party data. However, the current problem lies in the lack of efficient methods to extract meaningful information from this extensive data pool and utilize it effectively. The healthcare industry requires an advanced analytics solution that can process and interpret the collected data to track customer behavior and conditions accurately. By doing so, insurance companies can tailor their insurance policy offers to individual needs and calculate royalties for prior policyholders more accurately. This data-driven approach is expected to enhance revenue generation and improve customer understanding, ultimately leading to better service provision.

1.3 NEED FOR THE SYSTEM

By leveraging the power of the Big Data Ecosystem to analyze competitors' data and track customer behavior, insurance companies can identify patterns and trends that can be used to tailor insurance policy offers to individual needs. This personalized approach can lead to higher customer acquisition, retention, and increased revenue.

Utilizing big data analytics can provide valuable insights into customer behavior and conditions, allowing insurance companies to optimize their services. By understanding customers' needs and expectations better, companies can enhance the overall customer experience and provide proactive support, leading to higher customer satisfaction and loyalty.

Calculating royalties for prior policyholders accurately is a crucial aspect of the insurance business. By leveraging big data analytics, companies can analyze historical data and customer behavior to determine the appropriate royalties, reducing errors and ensuring fair compensation for policyholders.

1.4 CHALLENGES IN THE SYSTEM

Implementing a big data analytics system in healthcare insurance companies is met with challenges such as managing large data volumes, ensuring data security, integrating diverse data sources, analyzing complex datasets, scaling infrastructure, addressing talent shortage, and managing ethical considerations and biases. Overcoming these challenges is crucial to leverage the potential of big data for improved decision-making, personalized offerings, and enhanced customer satisfaction, ultimately leading to increased revenue and a better understanding of customers in the dynamic healthcare insurance industry.

1.5 SCOPE OF THE PROJECT

The scope of the project involves implementing a big data analytics system in a healthcare insurance company. The project encompasses collecting and managing large volumes of data from various sources, such as web scraping and third-party data. It includes developing a secure and scalable infrastructure to store and process the data. The project also involves designing and implementing advanced analytics techniques to analyze the data and extract valuable insights related to customer behavior and conditions.

CHAPTER 2

LITERATURE SURVEY

2.1 Big Data Analytics

The paper [1] proposes the idea of implementing big data technologies in healthcare system. . The volume of data available in digital form pertaining to the health of individuals has already become huge and is rapidly growing. It includes both the traditional clinically collected data, administrative data and increasingly other health and contextual sensor data collected in non-clinical settings. While the importance of analytics to healthcare is increasingly recognized, the healthcare industry is yet to fully harness the emergence of big data technologies specifically for carrying out analytics and to fully recognize the implications for healthcare. In this paper they introduced a framework to help organize and guide the understanding of the application of big data technologies for processing health and healthcare data. The framework includes four layers: Health Data Sources, the Big Data Technology Layer, the Big Data Analytics Layer, and the Applications Layer. Once data is collected and uploaded into the big data technology platform, analytics, data mining or machine learning approaches and technologies can be applied.

2.2 Big Data Applications

The paper[2] proposes the application of big data technologies in insurance companies. Big data in a narrow sense refers to the extraction of valuable, massive, and diversified transaction data, interactive data, sensor data, etc. through its rapid acquisition, processing, and analysis. The data involved are so large that they cannot be collected, processed, or managed in a reasonable time frame by existing technology and tools. Therefore, they require a new processing mode, decision-making ability, insight and discovery ability, and process optimization ability. In traditional insurance product innovation it is easy to ignore the information derived from the behavior of individual consumers, so it is difficult to launch personalized products. Moreover, the data held by insurance companies are mainly internal data, but rarely contain external customer behavior data. Through big data, insurance companies can analyze user needs, reduce product development costs, solve the problem of product development cycle lengths being too long to keep up with consumer needs, and develop products more scientifically according to consumer behavior data. This can be done to achieve

more interaction with customers,to provide products and services closer to customer needs, and to develop innovative, personalized products for consumers.

2.3 Summary:

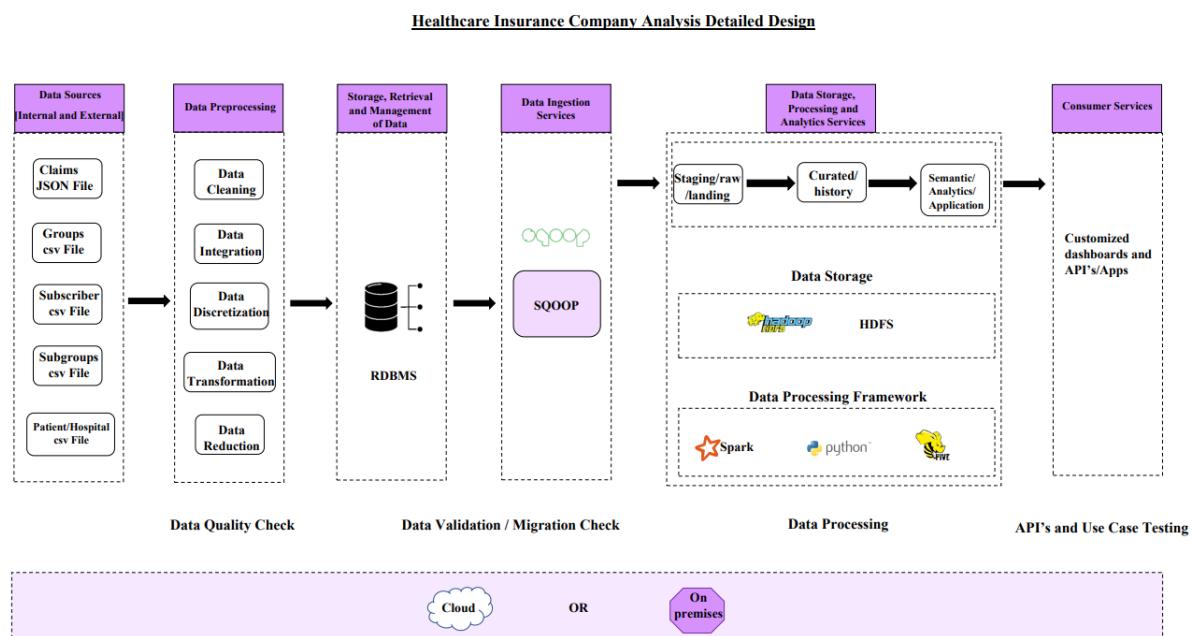
Overall in this research, big data analytics system in a healthcare insurance company offered tremendous potential for revenue growth and enhanced customer understanding. By addressing challenges such as data volume management, security, integration, analysis complexity, scalability, talent shortage, and ethical considerations, the project can pave the way for personalized insurance policy offerings, accurate royalty calculations, and improved service provision. Leveraging the power of big data analytics, the system enables the company to track customer behavior and conditions, leading to tailored insurance solutions and increased revenue. Additionally, the project promotes better decision-making through advanced analytics techniques, while adhering to data privacy regulations and ethical standards. Ultimately, the successful implementation of the project contributes to a better understanding of customers and an improved overall customer experience in the healthcare insurance industry.

CHAPTER 3

SYSTEM DESIGN

3.1 SYSTEM ARCHITECTURE

The system is divided into 6 modules namely data sources, data preprocessing, storage, retrieval and management of data, data ingestion services, data storage and analytics services and consumer services.



3.2 SYSTEM REQUIREMENTS

The system requirements for implementing a big data analytics system in a healthcare insurance company include robust data storage and processing capabilities, support for data integration and ETL processes, strong security and privacy measures, advanced analytics and machine learning capabilities, scalability and performance optimization, user-friendly interfaces for data exploration and visualization, integration with existing systems, backup and disaster recovery mechanisms, compliance with regulatory requirements and data governance practices, and provision of training and ongoing support. These requirements are essential for building a comprehensive and efficient system that can handle large volumes of data, derive meaningful insights, ensure data security and privacy, and meet industry-specific needs.

3.2.1 FUNCTIONAL REQUIREMENTS:

3.2.1.1 Hardware Requirements

- Processor: The hardware should consist of a processor with sufficient processing power to handle the bigdata tools involved in the project. A multi-core processor such as Intel Core i3 or i5 or i7 would be suitable.
- Memory: The hardware should have sufficient memory to handle the large data involved in the project. A minimum of 8GB of RAM is recommended.
- Graphics Card: The hardware should have a graphics card with dedicated GPU memory to accelerate the installation of Hadoop and other tools. A graphics card with a minimum of 2GB VRAM is recommended.
- Storage: The hardware should have sufficient storage capacity to store the big data. A minimum of 256GB of storage is recommended.
- Connectivity: The hardware should have connectivity options such as Wi-Fi, Bluetooth, or Ethernet to transfer data from external sources.
- Dataset: It includes 6 csv files and 1 json file related to healthcare insurance company.

3.2.1.2 Software Requirements

- Programming language: The project should be implemented using a programming language suitable big data analytics such as Python.
- Development environment: The project should be developed using a suitable development environment such as Hadoop 3.3.5,spark1.4.4,hive3.1.2 and sqoop 3.4.0
- Documentation: The project should include detailed documentation on how to install, configure, and run the project, as well as the implementation details of the various big data technologies like Hadoop,spark sql,sqoop and hive.
- Testing framework: The project should use a testing framework such as pytest to automate testing and ensure code quality.
- Deployment: The project should be deployable in a cloud-based or local environment.

3.2.2 NON-FUNCTIONAL REQUIREMENTS:

The following are some potential non-functional requirements in terms of performance for the project “Analysis on Healthcare Insurance Company”.

3.2.2.1 Performance: The project should be optimized for efficient processing and response time. The pre-processing of the dataset should be implemented in a manner such that it reduces the computation time and the response time and does not slow down the analysis of the data. The project should be able to handle data from various sources and provide a reasonably correct output with valuable insights.

CHAPTER 4

MODULE DESCRIPTION

4.1 DATA PRE-PROCESSING:

- After parsing and inferring the schema of the given xml and csv formats, data is ingested.
- We did general data cleaning steps like empty string replacements with actual NULL, data type checks (including date format) and corrections/ rejections, file name checks, empty file checks, malformed record checks and rejection etc.

4.2 STORAGE, RETRIEVAL AND MANAGEMENT OF DATA:

- After the files data is preprocessed, the data model for RDBMS is created so that the files data can be stored into the RDBMS.
- After storing the data into the RDBMS, we now transform according to our business requirements.

Schema Design for SQL Database:

4.3 DATA INGESTION SERVICES:

- Data is then moved to HDFS using the data ingestion tool, Sqoop.
- Sqoop is used for the efficient transfer of data from RDBMS to HDFS and vice versa.

4.4 DATA STORAGE, PROCESSING AND ANALYTICS SERVICES:

- These are carried out using technologies such as HDFS for data storage and Spark, Python and Hive for data processing.
- Data landed to HDFS needs to be analyzed by some analytical queries.

CHAPTER 5

RESULTS & IMPLEMENTATION

5.1 DATA SOURCES:

- Data coming from third-party sources reside in local directory and has csv and json format.
- Fields present in the data files and tables.
- Data files contain the below fields:

Column Name/Field Name Column Description/Field Description:

Json File Fields

- CLAIM_ID
- PATIENT_ID
- DISEASE_NAME
- SUB_ID
- CLAIM_DATE
- CLAIM_TYPE
- CLAIM_AMOUNT
- CLAIMED_OR_REJECTED

CSV File 1 Fields (Patient.csv)

- PATIENT_ID
- PATIENT_NAME

- PATIENT _GENDER
- PATIENT _BIRTHDATE
- PATIENT _PHONE
- HOSPITAL_ID
- DISEASE _ NAME
- CITY

CSV File 2 Fields (Subscriber.csv)

- SUB_ID
- FIRST_NAME
- LAST_NAME
- STREET
- BIRTH_DATE
- GENDER
- PHONE_NO
- COUNTRY
- CITY
- ZIP_CODE
- SUBGRP_ID
- ELIG_IND
- E_DATE
- T_DATE

CSV File 3 Fields (Group.csv)

- GRP_ID
- GRP_NAME
- PREMIUM_WRITTEN
- GRP_TYPE
- PIN_CODE
- CITY
- COUNTRY
- ESTABLISHMENT_YEAR

CSV File 4 Fields (disease.csv)

- SUBGRP_ID
- DISEASE_NAME
- DISEASE_ID

CSV File 5 Fields (subgroup.csv)

- SUBGRP_ID
- SUBGRP_NAME
- GRP_ID

CSV File 6 Fields (hospital.csv)

- HOSPITAL_ID

- HOSPITAL_NAME

- CITY

- STATE

- COUNTRY

CSV File 7 Fields (grpsubgrp.csv)

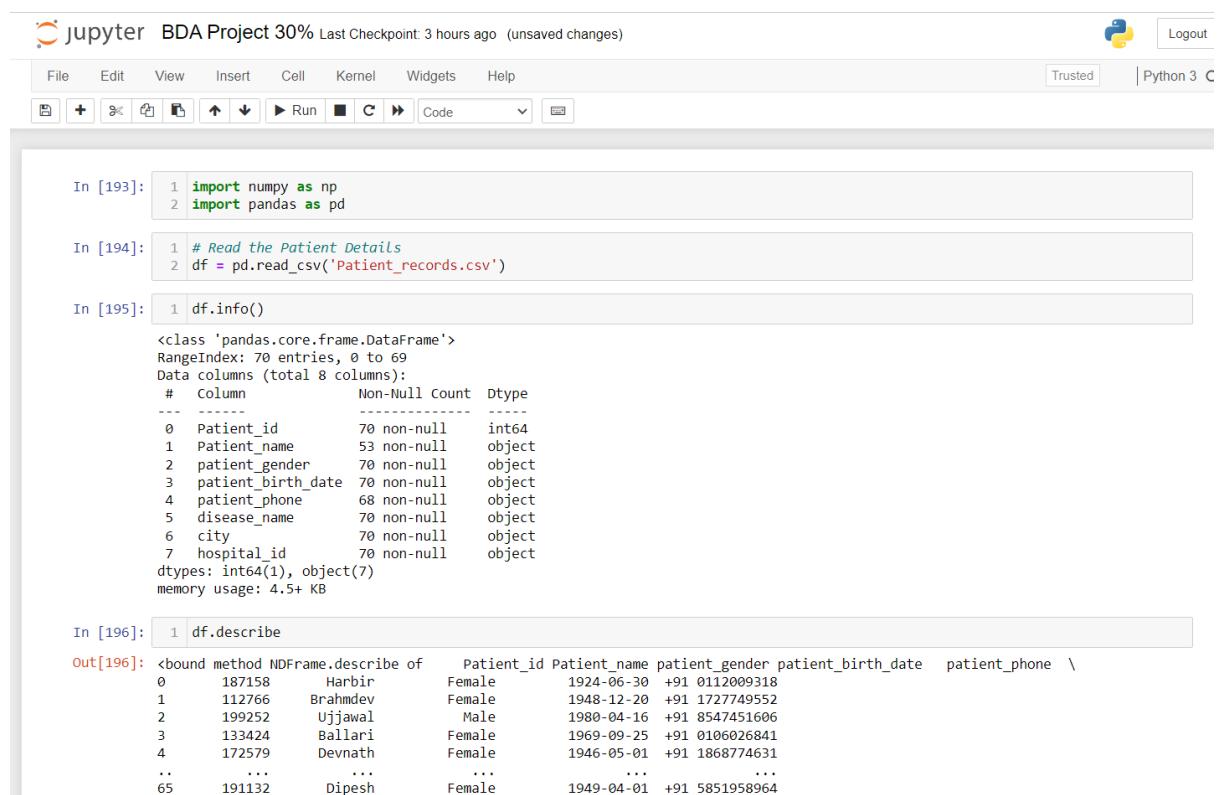
- SUBGRP_ID

- GRP_ID

5.2. DATA PRE-PROCESSING:

The JSON and csv files are preprocessed to make them suitable for further application.

The dataset is loaded and its structure is viewed.



The screenshot shows a Jupyter Notebook interface with the following details:

- Title Bar:** jupyter BDA Project 30% Last Checkpoint: 3 hours ago (unsaved changes)
- Toolbar:** File, Edit, View, Insert, Cell, Kernel, Widgets, Help
- Code Cells:**
 - In [193]:

```
1 import numpy as np
2 import pandas as pd
```
 - In [194]:

```
1 # Read the Patient Details
2 df = pd.read_csv('Patient_records.csv')
```
 - In [195]:

```
1 df.info()
```

Output:

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 70 entries, 0 to 69
Data columns (total 8 columns):
 #   Column           Non-Null Count  Dtype  
--- 
 0   Patient_id      70 non-null    int64  
 1   Patient_name    53 non-null    object  
 2   patient_gender  70 non-null    object  
 3   patient_birth_date 70 non-null    object  
 4   patient_phone   68 non-null    object  
 5   disease_name   70 non-null    object  
 6   city            70 non-null    object  
 7   hospital_id    70 non-null    object  
dtypes: int64(1), object(7)
memory usage: 4.5+ KB
```
 - In [196]:

```
1 df.describe
```

Output:

```
Out[196]: <bound method NDFrame.describe of      Patient_id Patient_name patient_gender patient_birth_date  patient_phone \
0      187158      Harbir     Female    1924-06-30 +91 0112009318
1      112766    Brahmdev    Female    1948-12-20 +91 1727749552
2      199252     Ujjawal     Male     1980-04-16 +91 8547451606
3      133424    Ballari     Female    1969-09-25 +91 0106026841
4      172579    Devnath     Female    1946-05-01 +91 1868774631
..        ...
65     191132     Dipesh     Female    1949-04-01 +91 5851958964
```

```
68     Diabetes    Ambarnath    H1014
69  Pet allergy    Sonipat    H1017
```

```
[ 70 rows x 8 columns]>
```

```
In [197]: 1 df.head(10)
```

```
Out[197]:
```

	Patient_id	Patient_name	patient_gender	patient_birth_date	patient_phone	disease_name	city	hospital_id
0	187158	Harbir	Female	1924-06-30	+91 0112009318	Galactosemia	Rourkela	H1001
1	112766	Brahmdev	Female	1948-12-20	+91 1727749552	Bladder cancer	Tiruvottiyur	H1016
2	199252	Ujjawal	Male	1980-04-16	+91 8547451606	Kidney cancer	Berhampur	H1009
3	133424	Ballari	Female	1969-09-25	+91 0106026841	Suicide	Bihar Sharif	H1017
4	172579	Devnath	Female	1946-05-01	+91 1868774631	Food allergy	Bidhannagar	H1019
5	171320	Atasi	Male	1967-10-02	+91 9747336855	Whiplash	Amravati	H1013
6	107794	Manish	Male	1967-06-06	+91 4354294043	Sunbathing	Panvel	H1004
7	130339	Aakar	Female	1925-03-05	+91 2777633911	Drug consumption	Bihar Sharif	H1000
8	110377	Gurudas	Male	1945-05-06	+91 1232859381	Dengue	Kamarhati	H1001
9	149367	NaN	Male	1925-06-12	+91 1780763280	Head banging	Bangalore	H1013

```
In [198]: 1 df.isnull()
```

```
Out[198]:
```

	Patient_id	Patient_name	patient_gender	patient_birth_date	patient_phone	disease_name	city	hospital_id
0	False	False	False	False	False	False	False	False
1	False	False	False	False	False	False	False	False
2	False	False	False	False	False	False	False	False
3	False	False	False	False	False	False	False	False
4	False	False	False	False	False	False	False	False

The null values in the dataset are counted and replaced with “NA”

```
In [199]: 1 # Count the total Null values
2 df.isnull().sum()
```

```
Out[199]: Patient_id      0
Patient_name     17
patient_gender     0
patient_birth_date   0
patient_phone      2
disease_name      0
city              0
hospital_id       0
dtype: int64
```

```
In [200]: 1 # Replace the null values
2 df["Patient_name"].fillna("NA", inplace = True)
```

```
In [201]: 1 df.head()
```

```
Out[201]:
```

	Patient_id	Patient_name	patient_gender	patient_birth_date	patient_phone	disease_name	city	hospital_id
0	187158	Harbir	Female	1924-06-30	+91 0112009318	Galactosemia	Rourkela	H1001
1	112766	Brahmdev	Female	1948-12-20	+91 1727749552	Bladder cancer	Tiruvottiyur	H1016
2	199252	Ujjawal	Male	1980-04-16	+91 8547451606	Kidney cancer	Berhampur	H1009
3	133424	Ballari	Female	1969-09-25	+91 0106026841	Suicide	Bihar Sharif	H1017
4	172579	Devnath	Female	1946-05-01	+91 1868774631	Food allergy	Bidhannagar	H1019

```
In [202]: 1 # Replace the null values
2 df['patient_phone'].fillna("NA", inplace = True)
```

```
In [203]: 1 df.head()
```

```
Out[203]:
```

Patient_id	Patient_name	patient_gender	patient_birth_date	patient_phone	disease_name	city	hospital_id	
0	187158	Harbir	Female	1924-06-30	+91 0112009318	Galactosemia	Rourkela	H1001
1	112766	Brahmdev	Female	1948-12-20	+91 1727749552	Bladder cancer	Tiruvottiyur	H1016
2	199252	Ujjawal	Male	1980-04-16	+91 8547451606	Kidney cancer	Berhampur	H1009
3	133424	Ballari	Female	1969-09-25	+91 0106026841	Suicide	Bihar Sharif	H1017
4	172579	Devnath	Female	1946-05-01	+91 1868774631	Food allergy	Bidhannagar	H1019

The duplicate values are checked and dropped, if any.

```
In [204]: 1 # Count the total Null values
2 df.isnull().sum()

Out[204]: Patient_id      0
Patient_name      0
patient_gender      0
patient_birth_date      0
patient_phone      0
disease_name      0
city      0
hospital_id      0
dtype: int64

In [205]: 1 # Check the duplicates
2 df.duplicated()

Out[205]: 0    False
1    False
2    False
3    False
4    False
...
65   False
66   False
67   False
68   False
69   False
Length: 70, dtype: bool

In [206]: 1 # Drop duplicates
2 df.drop_duplicates(subset=None, keep='first', inplace=False, ignore_index=False)

Out[206]:
  Patient_id Patient_name patient_gender patient_birth_date patient_phone disease_name city hospital_id
0     187158        Harbir       Female  1924-06-30 +91 0112009318  Galactosemia  Rourkela    H1001
1     112766      Brahmdev       Female  1948-12-20 +91 1727749552  Bladder cancer Tiruvottiyur    H1016
```

The patient ID is validated and the preprocessed data is exported. Similar process is applied to all other JSON and csv files.

66	105686	NA	Male	1930-09-01	+91 7061843400	Hepatitis	Kolhapur	H1008
67	160140	Kishan	Male	1923-05-12	+91 9067652693	Rett Syndrome	Srikakulam	H1002
68	114252	NA	Female	1927-02-26	+91 4984346995	Diabetes	Ambarnath	H1014
69	188365	Bhageeratha	Male	1973-03-21	+91 0590662722	Pet allergy	Sonipat	H1017

70 rows × 8 columns

```
In [207]: 1 # Check Patient Id not less than 6 digit
2 count = 0
3 for i in df['Patient_id'].values:
4
5     if len(str(i))<6:
6         df.drop([count], axis=0, inplace=True)
7     elif len(str(i))>6:
8         df.drop([count], axis=0, inplace=True)
9     count = count+1

In [208]: 1 # Count the patient_gender types
2 df['patient_gender'].unique()

Out[208]: array(['Female', 'Male'], dtype=object)

In [209]: 1 # Export the Data
2 df.to_csv("patient.csv",index=False)

In [210]: 1 # Read the subscriber Details
2 df = pd.read_csv('subscriber.csv')

In [211]: 1 df.head()

Out[211]:
  sub_id first_name last_name Street Birth_date Gender Phone Country City Zip_Code Subgrp_id Elig_ind eff_date term_date
```

Here is another example using subscriber.csv:

```
In [214]: 1 df.isnull()
```

```
Out[214]:
   sub_id first_name last_name Street Birth_date Gender Phone Country City Zip_Code Subgrp_id Elig_ind eff_date term_date
0 False False
1 False False
2 False False
3 False False
4 False False
... ... ... ... ... ... ... ... ... ... ... ... ... ... ... ... ... ... ...
95 False False False False False False True False False
96 False False
97 False True False False
98 False False
99 False False
```

100 rows × 14 columns

```
In [215]: 1 # Count the total Null values
2 df.isnull().sum()
```

```
Out[215]:
sub_id      0
first_name  27
last_name    0
Street       0
Birth_date   0
Gender       0
Phone        3
Country      0
City         0
Zip Code     0
```

```
In [216]: 1 # Replace the null values
2 df["first_name"].fillna("NA", inplace = True)
```

```
In [217]: 1 # Replace the null values
2 df["Phone"].fillna("NA", inplace = True)
```

```
In [218]: 1 # Replace the null values
2 df["Subgrp_id"].fillna("NA", inplace = True)
```

```
In [219]: 1 # Replace the null values
2 df["Elig_ind"].fillna("N", inplace = True)
```

```
In [220]: 1 df.head(60)
```

```
Out[220]:
   sub_id first_name last_name Street Birth_date Gender Phone Country City Zip_Code Subgrp_id Elig_ind eff_date term_date
0 SUBID10000 Harbir Vishwakarma Baria Marg 1924-06-30 Female +91 0112009318 India Rourkela 767058 S107 Y 1944-06-30 1
1 SUBID10001 Brahmdev Sonkar Lala Marg 1948-12-20 Female +91 1727749552 India Tiruvottiyur 34639 S105 Y 1968-12-20 1
2 SUBID10002 Ujjawal Devi Mammen Zila 1980-04-16 Male +91 8547451606 India Berhampur 914455 S106 N 2000-04-16 2
3 SUBID10003 Ballari Mishra Sahni Zila 1969-09-25 Female +91 0106026841 India Bihar Sharif 91481 S104 N 1989-09-25 1
4 SUBID10004 Devnath Srivastav Magar Zila 1946-05-01 Female +91 1868774631 India Bidhannagar 531742 S110 N 1966-05-01 1
5 SUBID10005 Atasi Seth Khatri Nagar 1967-10-02 Male +91 9747336855 India Amravati 229062 S104 Y 1987-10-02 1
6 SUBID1006 Manish Maurya Swaminathan Chunch 1967-06-08 Male +91 1251204042 India Panvel 438733 S109 N 1987-06-08 1
```

```
In [221]: 1 df.isnull().sum()
```

```
In [221]: 1 df.isnull().sum()
```

```
Out[221]: sub_id      0
first_name    0
last_name     0
Street        0
Birth_date    0
Gender        0
Phone         0
Country       0
City          0
Zip Code      0
Subgrp_id     0
Elig_ind      0
eff_date      0
term_date     0
dtype: int64
```

```
In [222]: 1 # Check the duplicates
2 df.duplicated()
```

```
Out[222]: 0    False
1    False
2    False
3    False
4    False
...
95   False
96   False
97   False
98   False
99   False
Length: 100, dtype: bool
```

```
In [223]: 1 # Drop duplicates
2 df.drop_duplicates(subset=None, keep='first', inplace=False, ignore_index=False)
```

```
Out[223]:
```

	sub_id	first_name	last_name	Street	Birth_date	Gender	Phone	Country	City	Zip Code	Subgrp_id	Elig_ind	eff_date	term_date
0	SUBID10000	Harbir	Vishwakarma	Baria Marg	1924-06-30	Female	+91 0112009318	India	Rourkela	767058	S107	Y	1944-06-30	1954-01-14
1	SUBID10001	Brahmdev	Sonkar	Lala Marg	1948-12-20	Female	+91 1727749552	India	Tiruvottiyur	34639	S105	Y	1968-12-20	1970-05-16
2	SUBID10002	Ujjawal	Devi	Mammen Zila	1980-04-16	Male	+91 8547451606	India	Berhampur	914455	S106	N	2000-04-16	2008-05-04
3	SUBID10003	Ballari	Mishra	Sahni Zila	1969-09-25	Female	+91 0106026841	India	Bihar Sharif	91481	S104	N	1989-09-25	1995-06-05
4	SUBID10004	Devnath	Srivastav	Magar Zila	1946-05-01	Female	+91 1868774631	India	Bidhannagar	531742	S110	N	1966-05-01	1970-12-09
...
95	SUBID10095	Ekaaksh	Rai	Bansal Ganj	1933-12-02	Others	NA	India	Pimpri-Chinchwad	158186	S107	N	1953-12-02	1960-07-29
96	SUBID10096	Chanak	Sonkar	Kaur	1959-04-07	Others	+91 7284540687	India	Rourkela Industrial Township	899590	S101	Y	1979-04-07	1986-03-07
97	SUBID10097	NA	Sonkar	Rana Ganj	1940-02-04	Others	+91 8908240160	India	Mira-Bhayandar	896506	S107	Y	1960-02-04	1965-01-12
98	SUBID1098	Pushkar	Kumar	Sodhi Zila	1934-10-05	Others	+91 8956368286	India	Korba	910732	S107	Y	1954-10-05	1961-04-05
99	SUBID1099	Shikha	Srivastav	Ahuja Road	1970-09-06	Others	+91 3042509956	India	Nanded	101500	S109	Y	1990-09-06	1997-11-27

100 rows × 14 columns

```
In [224]: 1 # Check Subscriber Id not less than 9 digit
2 count = 0
3 for i in df['sub_id'].values:
4
5     if len(i)<9:
6         df.drop([count], axis=0, inplace=True)
7     elif len(str(i))>10:
8         df.drop([count], axis=0, inplace=True)
9     count = count+1
```

```
In [225]: 1 # Check the Elig_id types
2 df['Elig_id'].unique()
```

```
Out[225]: array(['Y', 'N'], dtype=object)
```

```
In [226]: 1 # Check always eff_date is greater than term_date
2 count = 0
3 for x,y in df[['eff_date','term_date']].values:
4     dob1 = datetime.strptime(x,"%Y-%m-%d").date()
5     dob2 = datetime.strptime(y,"%Y-%m-%d").date()
6     if dob1 > dob2:
7         df.drop([count], axis=0, inplace=True)
8     count = count + 1
```

```
In [227]: 1 # Export the Data
2 df.to_csv('subscriber.csv',index=False)
```

Exploratory Data Analysis (EDA):

EDA is used to analyze and investigate data sets and summarize their main characteristics.

The screenshot shows a Jupyter Notebook interface with the title "jupyter EDA Last Checkpoint: 2 hours ago (autosaved)". The toolbar includes File, Edit, View, Insert, Cell, Kernel, Widgets, Help, Trusted, Python 3, and Logout. Below the toolbar, there are buttons for New, Run, Stop, Cell, Kernel, Help, and Code.

In [54]:

```
1 import matplotlib.pyplot as plt
2 plt.style.use("seaborn")
3 import pandas as pd
4 import numpy as np
```

In [55]:

```
1 import seaborn as sns
```

In [56]:

```
1 df=pd.read_csv("patient.csv",header=None)
2 #set column names equal to values in row index position 0
3 df.columns = df.iloc[0]
4
5 #remove first row from DataFrame
6 df = df[1:]
7
8 #view updated DataFrame
9 df
```

Out[56]:

	Patient_id	Patient_name	Patient_gender	Patient_birth_date	Patient_phone	Disease_name	City	Hospital_id
1	187158	Harbir	Female	1924-06-30	+91 0112009318	Galactosemia	Rourkela	H1001
2	112766	Brahmdev	Female	1948-12-20	+91 1727749552	Bladder cancer	Tiruvottiyur	H1016
3	199252	Ujjawal	Male	1980-04-16	+91 8547451606	Kidney cancer	Berhampur	H1009
4	133424	Ballari	Female	1969-09-25	+91 0106026841	Suicide	Bihar Sharif	H1017
5	172579	Devnath	Female	1946-05-01	+91 1868774631	Food allergy	Bidhannagar	H1019
...
66	191132	Dipesh	Female	1949-04-01	+91 5851958964	Glaucoma	Kochi	H1016
67	105686	NaN	Male	1930-09-01	+91 7061843400	Hepatitis	Kolhapur	H1008
68	160140	Kishan	Male	1923-05-12	+91 9067652693	Rett Syndrome	Srikakulam	H1002
69	114252	NaN	Female	1927-02-26	+91 4984346995	Diabetes	Ambarnath	H1014
70	188365	Bhageeratha	Male	1973-03-21	+91 0590662722	Pet allergy	Sonipat	H1017

```
In [57]: 1 #reset index values
2 df.reset_index(drop=True, inplace=True)
3
4 #view updated DataFrame
5 df
```

Out[57]:

	Patient_id	Patient_name	patient_gender	patient_birth_date	patient_phone	disease_name	city	hospital_id
0	187158	Harbir	Female	1924-06-30	+91 0112009318	Galactosemia	Rourkela	H1001
1	112766	Brahmdev	Female	1948-12-20	+91 1727749552	Bladder cancer	Tiruvottiyur	H1016
2	199252	Ujjawal	Male	1980-04-16	+91 8547451606	Kidney cancer	Berhampur	H1009
3	133424	Ballari	Female	1969-09-25	+91 0106026841	Suicide	Bihar Sharif	H1017
4	172579	Devnath	Female	1946-05-01	+91 1868774631	Food allergy	Bidhannagar	H1019
...
65	191132	Dipesh	Female	1949-04-01	+91 5851958964	Glaucoma	Kochi	H1016
66	105686	NaN	Male	1930-09-01	+91 7061843400	Hepatitis	Kolhapur	H1008
67	160140	Kishan	Male	1923-05-12	+91 9067652693	Rett Syndrome	Srikakulam	H1002
68	114252	NaN	Female	1927-02-26	+91 4984346995	Diabetes	Ambarnath	H1014
69	188365	Bhageeratha	Male	1973-03-21	+91 0590662722	Pet allergy	Sonipat	H1017

70 rows × 8 columns

```
In [58]: 1 def age(born):
2     born = datetime.strptime(born, "%Y-%m-%d").date()
3     today = date.today()
4     return today.year - (today.month,
5                           today.day) < (born.month,
6                                         born.day))
```

A column called “Age” is calculated and added to the “patient.csv” dataset.

```
In [59]: 1 from datetime import datetime, date
```

```
In [60]: 1 df['Age'] = df['patient_birth_date'].apply(age)
```

```
In [61]: 1 df
```

Out[61]:

	Patient_id	Patient_name	patient_gender	patient_birth_date	patient_phone	disease_name	city	hospital_id	Age
0	187158	Harbir	Female	1924-06-30	+91 0112009318	Galactosemia	Rourkela	H1001	98
1	112766	Brahmdev	Female	1948-12-20	+91 1727749552	Bladder cancer	Tiruvottiyur	H1016	74
2	199252	Ujjawal	Male	1980-04-16	+91 8547451606	Kidney cancer	Berhampur	H1009	42
3	133424	Ballari	Female	1969-09-25	+91 0106026841	Suicide	Bihar Sharif	H1017	53
4	172579	Devnath	Female	1946-05-01	+91 1868774631	Food allergy	Bidhannagar	H1019	76
...
65	191132	Dipesh	Female	1949-04-01	+91 5851958964	Glaucoma	Kochi	H1016	73
66	105686	NaN	Male	1930-09-01	+91 7061843400	Hepatitis	Kolhapur	H1008	92
67	160140	Kishan	Male	1923-05-12	+91 9067652693	Rett Syndrome	Srikakulam	H1002	99
68	114252	NaN	Female	1927-02-26	+91 4984346995	Diabetes	Ambarnath	H1014	96
69	188365	Bhageeratha	Male	1973-03-21	+91 0590662722	Pet allergy	Sonipat	H1017	49

70 rows × 9 columns

```
In [62]: 1 df['Age']
```

```
Out[62]: 0    98
1    74
2    42
3    53
4    76
..
```

A graph of Patient Count vs Age is plotted.

```
In [63]: 1 plt.figure(figsize=(6,6))
2 bins=[0,10,20,30,40,50,60,70,80,90,100,110]
3 a=df['Age']
4 plt.hist(a, bins, facecolor='crimson', alpha=0.8, edgecolor='k', linewidth=1)
5 plt.xlabel("Age", fontsize=12)
6 plt.ylabel("Patient Count", fontsize=12)
7 plt.locator_params('y', nbins=12)
8 plt.ylim(ymax=20)
9 plt.locator_params('x', nbins=10)
10 plt.xlim(xmin=0,xmax=100)
11 plt.title("Patient Count (According to Age)", fontsize=15)
12 plt.savefig('PatientCount(Age).png')
```



Similarly for Patient Count vs Gender

```
In [64]: 1 plt.figure(figsize=(6,6))
2 ax=sns.countplot(x='patient_gender',data=df)
3 ax.set_ylabel("Patient Count", fontsize = 12)
4 ax.set_xlabel("Gender", fontsize = 12)
5 ax.set_title("Gender Count (Patient)", fontsize=15)
6 for p in ax.patches:
7     ax.annotate('{:.1f}'.format(p.get_height()), (p.get_x() + 0.25, p.get_height() + 0.01))
8 plt.savefig('GenderCount(Patient).png')
```



Similarly the above steps are done for other datasets:

Here is another example using the claims.json dataset

```
In [65]: 1 js=pd.read_json('claims.json')

In [66]: 1 js

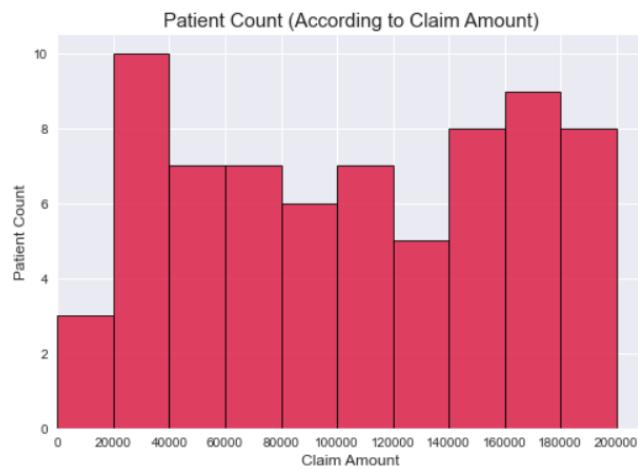
Out[66]:
   claim_id  patient_id  disease_name      SUB_ID Claim_Or_Rejected  claim_type  claim_amount  claim_date
0         0       187158  Galactosemia  SUBID1000                  N  claims of value        79874  1949-03-14
1         1      112766  Bladder cancer  SUBID10001                 N  claims of policy       151142  1970-03-16
2         2      199252  Kidney cancer  SUBID10002                 N  claims of value        59924  2008-02-03
3         3      133424        Suicide  SUBID10003                 N  claims of fact        143120  1995-02-08
4         4      172579  Food allergy  SUBID10004                 Y  claims of value       168634  1967-05-23
...
65        65      191132     Glaucoma  SUBID1065                 Y  claims of policy       81980  1969-05-31
66        66      105686    Hepatitis  SUBID10066                N  claims of fact        13667  1957-09-12
67        67      160140  Rett Syndrome  SUBID1067                 N  claims of value       109433  1944-12-25
68        68      114252     Diabetes  SUBID10068                N  claims of policy       152901  1948-02-13
69        69      188365    Pet allergy  SUBID10069                N  claims of fact        99313  1994-08-25

70 rows × 8 columns
```

```
In [67]: 1 plt.figure(figsize=(5,5))
2 ax=sns.countplot(x='Claim_Or_Rejected',data=js,hue='Claim_Or_Rejected')
3 ax.set_ylabel("Count",fontsize=12)
4 ax.set_xlabel("Status", fontsize=12)
5 ax.set_title("Claim Status",fontsize=15)
6 for p in ax.patches:
7     ax.annotate('{:.1f}'.format(p.get_height()), (p.get_x()+0.25, p.get_height()+0.01))
8 plt.savefig('claimStatus.png')
```



```
In [68]: 1 bins=[0, 20000, 40000, 60000, 80000, 100000, 120000, 140000, 160000, 180000, 200000, 220000, 240000]
2 a=js['claim_amount']
3 plt.hist(a, bins, facecolor='crimson', alpha=0.8, edgecolor='k', linewidth=1)
4 plt.xlabel("Claim Amount", fontsize=12)
5 plt.ylabel("Patient Count", fontsize=12)
6 plt.locator_params('x', nbins=12)
7 plt.xlim(xmin=0, xmax=210000)
8 plt.title("Patient Count (According to Claim Amount)", fontsize=15)
9 plt.savefig('Patient_count(claimAmt).png')
```



5.3 STORAGE, RETRIEVAL AND MANAGEMENT OF DATA:

Connecting to MySQL using Python :

```
In [33]: 1 # Importing data preprocessing lib
2 import pandas as pd
3
In [34]: 1 pip install mysql-connector-python-rf
Requirement already satisfied: mysql-connector-python-rf in c:\users\sheeba\anaconda3\lib\site-packages (2.2.2)
Note: you may need to restart the kernel to use updated packages.
In [35]: 1 # Import mysql connector and establish connection
2 import mysql.connector
3
4 mydb = mysql.connector.connect(
5     host="localhost", user="root", password="password")
6
7 print(mydb)
<mysql.connector.connection.MySQLConnection object at 0x000002220892EE0>
```

Viewing Existing Databases:

```
In [36]: 1 # Print the databases (list of databases)
2 mycursor = mydb.cursor()
3
4 mycursor.execute("show databases")
5 myresult = mycursor.fetchall()
6
7 for x in myresult:
8     print(x)

('a',)
('database1',)
('demo',)
('epes_db',)
('healthcare',)
('information_schema',)
('loginregister',)
('mysql',)
('performance_schema',)
('phpmyadmin',)
('registration',)
('test',)
('testing',)
('userregister',)
('userregistration',)
```

Creating and using new database(health_care):

```
In [38]: 1 # Create new database
2
3 mycursor.execute("CREATE DATABASE HEALTH_CARE")
```

```
In [40]: 1 # Use new database
2
3 mydb = mysql.connector.connect(
4     host="localhost",
5     user="root",
6     password="password",
7     database="HEALTH_CARE"
8 )
9
10 mycursor = mydb.cursor()
```

Creating new tables:

```
In [41]: 1 # Create new table groups
2
3 mycursor.execute("""CREATE TABLE groups_tble (grp_sk int NOT NULL UNIQUE AUTO_INCREMENT,grp_id VARCHAR(6) NOT NULL PRIMARY KEY,
4 grp_name VARCHAR(90),premium_written int NOT NULL,city VARCHAR(20),
5 zip_code int,country VARCHAR(5),grp_type VARCHAR(10))""")
```

```
In [42]: 1 # Create new table subgroup
2 mycursor.execute("""CREATE TABLE subgroup (subgrp_sk int NOT NULL UNIQUE AUTO_INCREMENT,subgrp_id VARCHAR(4) NOT NULL PRIMARY KEY,
3 subgrp_name VARCHAR(90),monthly_premium float(6,2))""")
```

```
In [43]: 1 # Create new table group_subgroup
2 mycursor.execute("""CREATE TABLE group_subgroup (grpsub_sk int NOT NULL UNIQUE AUTO_INCREMENT,
3 g_id VARCHAR(6) NOT NULL,
4 s_id VARCHAR(4) NOT NULL,
5 FOREIGN KEY(g_id) REFERENCES groups_tble(grp_id),
6 FOREIGN KEY(s_id) REFERENCES subgroup(subgrp_id))""")
```

```
In [44]: 1 # Create new table disease
2 mycursor.execute("""CREATE TABLE disease (disease_id int NOT NULL PRIMARY KEY ,disease_name VARCHAR(30) NOT NULL ,
3 subgrp_id VARCHAR(4),FOREIGN KEY(subgrp_id) REFERENCES subgroup(subgrp_id))""")
```

```
In [45]: 1 # Create new table subscriber
2 mycursor.execute("""CREATE TABLE subscriber (sub_id VARCHAR(10) NOT NULL PRIMARY KEY ,first_name VARCHAR(30) ,
3 last_name VARCHAR(20),street VARCHAR(30),birth_date DATE,gender VARCHAR(6),phone VARCHAR(15),city VARCHAR(30),
4 zip_code int,country VARCHAR(10),subgrp_id VARCHAR(4),elig_ind VARCHAR(2) NOT NULL,eff_date DATE NOT NULL,t
5 ...
```

```
In [46]: 1 # create new table hospital_details
2 mycursor.execute("""CREATE TABLE hospital_details (hospital_id VARCHAR(5) NOT NULL PRIMARY KEY ,hospital_name VARCHAR(255),
3 city VARCHAR(20),state VARCHAR(20),country VARCHAR(6))""")
```

```
In [47]: 1 # Create new table patient_details
2 mycursor.execute("""CREATE TABLE patient_details(patient_id int NOT NULL PRIMARY KEY,patient_name VARCHAR(20),patient_gender
3 ...
```

```
In [48]: 1 # Create new table claims
2 mycursor.execute("""CREATE TABLE claims (claim_id int NOT NULL AUTO_INCREMENT PRIMARY KEY ,patient_id int NOT NULL,disease_n
3 ...
```

Inserting values into tables from preprocessed data files:

```
In [51]: 1 #inserting data into table group
2 with open('group.csv','r') as file:
3     mycursor.execute('`USE HEALTH_CARE`')
4     data=file.readlines()
5     for row in data:
6         list= row.split(",")
7         top=tuple([list[0],int(list[1]),int(list[2]),list[3],list[4],list[5],list[6]])
8         print(top)
9         sql = "INSERT INTO HEALTH_CARE.groups_tble(country,premium_written,zip_code,grp_id,grp_name,grp_type,city) VALUE"
10        mycursor.execute(sql, top)
11        mydb.commit()
```

```
('0', 1, 2, '3', '4', '5', '6')
('India', 72000, 482018, 'GRP101', 'Life Insurance Corporation of India', 'Govt.', 'Mumbai')
('India', 45000, 482049, 'GRP102', 'HDFC Standard Life Insurance Co. Ltd.', 'Private', 'Mumbai')
('India', 64000, 482030, 'GRP103', 'Max Life Insurance Co. Ltd.', 'Private', 'Delhi')
('India', 59000, 482028, 'GRP104', 'ICICI Prudential Life Insurance Co. Ltd.', 'Private', 'Mumbai')
('India', 37000, 482014, 'GRP105', 'Kotak Mahindra Life Insurance Co. Ltd.', 'Private', 'Mumbai')
('India', 89000, 482011, 'GRP106', 'Aditya Birla Sun Life Insurance Co. Ltd.', 'Private', 'Mumbai')
('India', 70000, 482006, 'GRP107', 'TATA AIG Life Insurance Co. Ltd.', 'Private', 'Mumbai')
('India', 52000, 482034, 'GRP108', 'SBI Life Insurance Co. Ltd.', 'Private', 'Mumbai')
('India', 78000, 482032, 'GRP109', 'Exide Life Insurance Co. Ltd.', 'Private', 'Bangalore')
('India', 48000, 482015, 'GRP110', 'Bajaj Allianz Life Insurance Co. Ltd.', 'Private', 'Pune')
('India', 57000, 482011, 'GRP111', 'PNB Metlife India Insurance Co. Ltd.', 'Private', 'Mumbai')
('India', 57000, 482022, 'GRP112', 'Reliance Nippon Life Insurance Company', 'Private', 'Mumbai')
('India', 64000, 482009, 'GRP113', 'Aviva Life Insurance Company India Ltd.', 'Private', 'Gurugram')
('India', 33000, 482043, 'GRP114', 'Sahara India Life Insurance Co. Ltd.', 'Private', 'Lucknow')
('India', 79000, 482036, 'GRP115', 'Shriram Life Insurance Co. Ltd.', 'Private', 'Hyderabad')
('India', 32000, 482002, 'GRP116', 'Bharti AXA Life Insurance Co. Ltd.', 'Private', 'Mumbai')
('India', 59000, 482017, 'GRP117', 'Future Generali India Life Insurance Co. Ltd.', 'Private', 'Mumbai')
('India', 97000, 482023, 'GRP118', 'IDBI Federal Life Insurance Co. Ltd.', 'Private', 'Mumbai')
('India', 47000, 482046, 'GRP119', 'Canara HSBC Oriental Bank of Commerce Life Insurance Co. Ltd.', 'Private', 'Gurugram')
```

```
In [52]: 1 #insert data into subgroup
2 with open('subgroup.csv', 'r') as file:
3     mycursor.execute(''use HEALTH_CARE''')
4     data=file.readlines()
5     for row in data:
6         list=row.split(",")
7         top=tuple([list[0],list[1],float(list[2])])
8         print(top)
9         sql = "INSERT INTO HEALTH_CARE.subgroup(subgrp_id,subgrp_name,monthly_premium) VALUES(%s,%s,%s)"
10        mycursor.execute(sql, top)
11        mydb.commit()

('0', '1', 2.0)
('S101', 'Deficiency Diseases', 3000.0)
('S102', 'Accident', 1000.0)
('S103', 'Physiology', 2000.0)
('S104', 'Therapy', 1500.0)
('S105', 'Allergies', 2300.0)
('S106', 'Self inflicted', 1200.0)
('S107', 'Cancer', 3200.0)
('S108', 'Infectious disease', 1500.0)
('S109', 'Hereditary', 2000.0)
('S110', 'Viral', 1000.0)
```

```
In [54]: 1 #inserting data into hospital_details
2 with open('hospital.csv','r') as file:
3     mycursor.execute(''use HEALTH_CARE''')
4     data=file.readlines()
5     for row in data:
6         list=row.split(",")
7         top=tuple(list)
8         print(top)
9         sql = "INSERT INTO hospital_details(hospital_id,hospital_name,city,state,country) VALUES(%s,%s,%s,%s,%s)"
10        mycursor.execute(sql,top)
11        mydb.commit()

('Hospital_id', 'Hospital_name', 'city', 'state', 'country\n')
('H1000', 'All India Institute of Medical Sciences', 'New Delhi', 'NaN', 'India\n')
('H1001', 'Medanta The Medicity', 'Gurgaon', 'Haryana', 'India\n')
('H1002', 'The Christian Medical College', 'Vellore', 'Tamil Nadu', 'India\n')
('H1003', 'PGIMER - Postgraduate Institute of Medical Education and Research', 'Chandigarh', 'Haryana', 'India\n')
('H1004', 'Apollo Hospital - Chennai', 'Chennai', 'Tamil Nadu', 'India\n')
('H1005', 'P. D. Hinduja National Hospital & Medical Research Centre', 'Mumbai', 'Maharashtra', 'India\n')
('H1006', 'Breach Candy Hospital', 'Mumbai', 'Maharashtra', 'India\n')
('H1007', 'Fortis Flt. Lt. Rajan Dhall Hospital', 'New Delhi', 'NaN', 'India\n')
('H1008', 'King Edward Memorial Hospital', 'Mumbai', 'Maharashtra', 'India\n')
('H1009', 'Indraprastha Apollo Hospital', 'Delhi', 'NaN', 'India\n')
('H1010', 'Lilavati Hospital And Research Centre', 'Mumbai', 'Maharashtra', 'India\n')
('H1011', 'Sir Ganga Ram Hospital', 'Delhi', 'NaN', 'India\n')
('H1012', 'Bombay Hospital & Medical Research Centre', 'Mumbai', 'Maharashtra', 'India\n')
('H1013', 'Apollo Health City - Jubilee Hills', 'Hyderabad', 'Telangana', 'India\n')
('H1014', 'Fortis Hiranandani Hospital', 'Mumbai', 'Maharashtra', 'India\n')
('H1015', 'Fortis Hospital Mulund', 'Mumbai', 'Maharashtra', 'India\n')
('H1016', 'Jaslok Hospital and Research Centre', 'Mumbai', 'Maharashtra', 'India\n')
('H1017', 'Manipal Hospitals', 'Bengaluru', 'Karnataka', 'India\n')
('H1018', 'Yashoda Hospital Secunderabad', 'Hyderabad', 'Telangana', 'India\n')
('H1019', 'Apollo Hospitals - Bannerghatta Road', 'Bengaluru', 'Karnataka', 'India\n')
```

Viewing the database and tables in phpMyAdmin:

phpMyAdmin interface showing the 'groups_ibl' table in the 'health_care' database.

Table Structure:

```
CREATE TABLE `groups_ibl` (
  `grp_sk` int(11) NOT NULL,
  `grp_id` int(11) NOT NULL,
  `grp_name` varchar(255) NOT NULL,
  `premium_written` int(11) NOT NULL,
  `city` int(11) NOT NULL,
  `zip_code` int(11) NOT NULL,
  `country` int(11) NOT NULL,
  `grp_type` int(11) NOT NULL
)
```

Table Data:

grp_sk	grp_id	grp_name	premium_written	city	zip_code	country	grp_type
1	3	4	1	6	2	0	5
2	2	GRP101	72000	Mumbai	482018	India	Govt
3	3	GRP102	45000	Mumbai	482049	India	Private
4	4	GRP103	64000	Delhi	482030	India	Private
5	5	GRP104	59000	Mumbai	482028	India	Private
6	6	GRP105	37000	Mumbai	482014	India	Private
7	7	GRP106	89000	Mumbai	482011	India	Private
8	8	GRP107	70000	Mumbai	482006	India	Private
9	9	GRP108	52000	Mumbai	482034	India	Private
10	10	GRP109	78000	Bangalore	482032	India	Private
11	11	GRP110	48000	Pune	482015	India	Private
12	12	GRP111	57000	Mumbai	482011	India	Private
13	13	GRP112	57000	Mumbai	482022	India	Private
14	14	GRP113	64000	Gurugram	482009	India	Private
15	15	GRP114	33000	Lucknow	482043	India	Private
16	16	GRP115	79000	Hyderabad	482036	India	Private

phpMyAdmin interface showing the 'hospital_details' table in the 'health_care' database.

Table Structure:

```
CREATE TABLE `hospital_details` (
  `hospital_id` int(11) NOT NULL,
  `hospital_name` varchar(255) NOT NULL,
  `city` int(11) NOT NULL,
  `state` int(11) NOT NULL,
  `country` int(11) NOT NULL
)
```

Table Data:

hospital_id	hospital_name	city	state	country
H1000	All India Institute of Medical Sciences	New Delhi	Nan	India
H1001	Medanta The Medicity	Gurgaon	Haryana	India
H1002	The Christian Medical College	Vellore	Tamil Nadu	India
H1003	PGIMER - Postgraduate Institute of Medical Education...	Chandigarh	Haryana	India
H1004	Apollo Hospital - Chennai	Chennai	Tamil Nadu	India
H1005	P. D. Hinduja National Hospital & Medical Research...	Mumbai	Maharashtra	India
H1006	Breach Candy Hospital	Mumbai	Maharashtra	India
H1007	Fortis Fit Lt Rajan Dhall Hospital	New Delhi	Nan	India
H1008	King Edward Memorial Hospital	Mumbai	Maharashtra	India
H1009	Indraprastha Apollo Hospital	Delhi	Nan	India
H1010	Lilavati Hospital And Research Centre	Mumbai	Maharashtra	India
H1011	Sir Ganga Ram Hospital	Delhi	Nan	India
H1012	Bombay Hospital & Medical Research Centre	Mumbai	Maharashtra	India
H1013	Apollo Health City - Jubilee Hills	Hyderabad	Telangana	India
H1014	Fortis Hiranandani Hospital	Mumbai	Maharashtra	India
H1015	Fortis Hospital Mulund	Mumbai	Maharashtra	India
H1016	IeLink Hospital and Research Centre	Mumbai	Maharashtra	India

		grpsub_sk	g_id	s_id
<input type="checkbox"/>	Edit	Copy	Delete	1 GRP101 S101
<input type="checkbox"/>	Edit	Copy	Delete	2 GRP105 S101
<input type="checkbox"/>	Edit	Copy	Delete	3 GRP110 S102
<input type="checkbox"/>	Edit	Copy	Delete	4 GRP150 S102
<input type="checkbox"/>	Edit	Copy	Delete	5 GRP136 S102
<input type="checkbox"/>	Edit	Copy	Delete	6 GRP122 S103
<input type="checkbox"/>	Edit	Copy	Delete	7 GRP108 S103
<input type="checkbox"/>	Edit	Copy	Delete	8 GRP138 S103
<input type="checkbox"/>	Edit	Copy	Delete	9 GRP148 S103
<input type="checkbox"/>	Edit	Copy	Delete	10 GRP103 S104
<input type="checkbox"/>	Edit	Copy	Delete	11 GRP113 S104
<input type="checkbox"/>	Edit	Copy	Delete	12 GRP123 S104
<input type="checkbox"/>	Edit	Copy	Delete	13 GRP133 S104
<input type="checkbox"/>	Edit	Copy	Delete	14 GRP143 S104
<input type="checkbox"/>	Edit	Copy	Delete	15 GRP153 S105
<input type="checkbox"/>	Edit	Copy	Delete	16 GRP104 S105

5.4 DATA INGESTION SERVICES:

Sqoop version 1.4.4 is installed

```
C:\WINDOWS\system32>sqoop version
Warning: HBASE_HOME and HBASE_VERSION not set.
Warning: HCAT_HOME not set
Warning: HCATALOG_HOME does not exist HCatalog imports will fail.
Please set HCATALOG_HOME to the root of your HCatalog installation.
Warning: ACCUMULO_HOME not set.
Warning: ZOOKEEPER_HOME not set.
Warning: HBASE_HOME does not exist HBase imports will fail.
Please set HBASE_HOME to the root of your HBase installation.
Warning: ACCUMULO_HOME does not exist Accumulo imports will fail.
Please set ACCUMULO_HOME to the root of your Accumulo installation.
Warning: ZOOKEEPER_HOME does not exist Accumulo imports will fail.
Please set ZOOKEEPER_HOME to the root of your Zookeeper installation.
Sqoop 1.4.4
git commit id 050a2015514533bc25f3134a33401470ee9353ad
Compiled by jarcec on Wed Jul 31 08:37:07 PDT 2013

C:\WINDOWS\system32>
```

Hadoop and Hive are also Installed and configured.

start-all.cmd is given in sbin directory of hadoop

```
C:\hadoop\hadoop-3.3.5\sbin>start-all.cmd
This script is Deprecated. Instead use start-dfs.cmd and start-yarn.cmd
starting yarn daemons

C:\hadoop\hadoop-3.3.5\sbin>
```

The Hadoop namenode starts

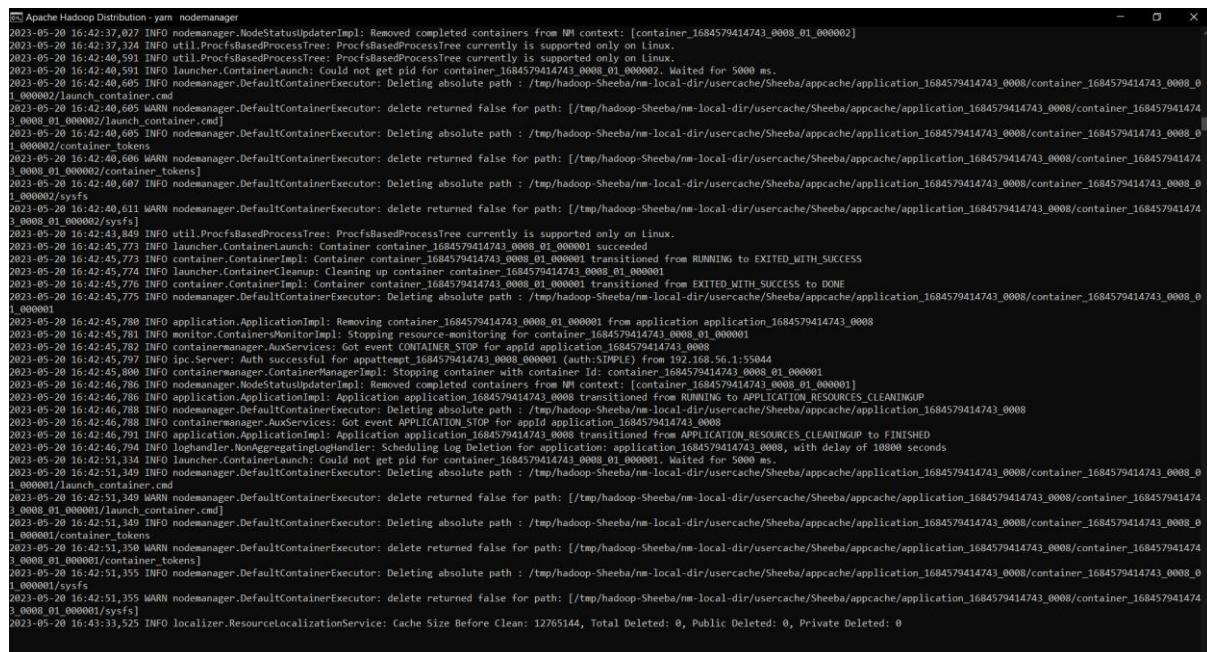
The Hadoop datanode starts

```
Apache Hadoop Distribution - hadoop datanode  
-168457613801//current/_finalized/subdir0/subdir0/blk_1073741937  
2023-05-20 16:42:43,554 INFO impl.FsDatasetAsyncDiskService: Scheduling blk_1073741931_1107 replica FinalizedReplica, blk_1073741931_1107, FINALIZED  
getHmabytes() = 36455  
getBytesOnDisk() = 36455  
getVisibleLength() = 36455  
getVolume() = C:/hadoop/hadoop-3.3.5/data/datanode  
getBlockURI() = file:///C:/hadoop/hadoop-3.3.5/data/datanode/current/BP-1673887656-192.168.56.1-1684576138017/current/_finalized/subdir0/subdir0/blk_1073741931 for deletion  
2023-05-20 16:42:43,554 INFO impl.FsDatasetAsyncDiskService: Deleted BP-1673887656-192.168.56.1-1684576138017 blk_1073741938_1114 URI file:///C:/hadoop/hadoop-3.3.5/data/datanode/current/BP-1673887656-192.168.56.1-1684576138017/current/_finalized/subdir0/subdir0/blk_1073741938 for deletion  
168457613801//current/_finalized/subdir0/blk_1073741938  
2023-05-20 16:42:43,554 INFO impl.FsDatasetAsyncDiskService: Scheduling blk_1073741932_1108 replica FinalizedReplica, blk_1073741932_1108, FINALIZED  
getHmabytes() = 755876  
getBytesOnDisk() = 755876  
getVisibleLength() = 755876  
getVolume() = C:/hadoop/hadoop-3.3.5/data/datanode  
getBlockURI() = file:///C:/hadoop/hadoop-3.3.5/data/datanode/current/BP-1673887656-192.168.56.1-1684576138017/current/_finalized/subdir0/subdir0/blk_1073741932 for deletion  
2023-05-20 16:42:43,554 INFO impl.FsDatasetAsyncDiskService: Deleted BP-1673887656-192.168.56.1-1684576138017 blk_1073741939_1115 URI file:///C:/hadoop/hadoop-3.3.5/data/datanode/current/BP-1673887656-192.168.56.1-1684576138017/current/_finalized/subdir0/subdir0/blk_1073741939 for deletion  
168457613801//current/_finalized/subdir0/blk_1073741939  
2023-05-20 16:42:43,556 INFO impl.FsDatasetAsyncDiskService: Scheduling blk_1073741933_1109 replica FinalizedReplica, blk_1073741933_1109, FINALIZED  
getHmabytes() = 224277  
getBytesOnDisk() = 224277  
getVisibleLength() = 224277  
getVolume() = C:/hadoop/hadoop-3.3.5/data/datanode  
getBlockURI() = file:///C:/hadoop/hadoop-3.3.5/data/datanode/current/BP-1673887656-192.168.56.1-1684576138017/current/_finalized/subdir0/subdir0/blk_1073741933 for deletion  
2023-05-20 16:42:43,557 INFO impl.FsDatasetAsyncDiskService: Deleted BP-1673887656-192.168.56.1-1684576138017 blk_1073741941_1117 URI file:///C:/hadoop/hadoop-3.3.5/data/datanode/current/BP-1673887656-192.168.56.1-1684576138017/current/_finalized/subdir0/subdir0/blk_1073741941 for deletion  
168457613801//current/_finalized/subdir0/blk_1073741941  
2023-05-20 16:42:43,557 INFO impl.FsDatasetAsyncDiskService: Scheduling blk_1073741934_1110 replica FinalizedReplica, blk_1073741934_1110, FINALIZED  
getHmabytes() = 566623  
getBytesOnDisk() = 566623  
getVisibleLength() = 566623  
getVolume() = C:/hadoop/hadoop-3.3.5/data/datanode  
getBlockURI() = file:///C:/hadoop/hadoop-3.3.5/data/datanode/current/BP-1673887656-192.168.56.1-1684576138017/current/_finalized/subdir0/subdir0/blk_1073741934 for deletion  
2023-05-20 16:42:43,558 INFO impl.FsDatasetAsyncDiskService: Deleted BP-1673887656-192.168.56.1-1684576138017 blk_1073741935_1110 URI file:///C:/hadoop/hadoop-3.3.5/data/datanode/current/BP-1673887656-192.168.56.1-1684576138017/current/_finalized/subdir0/subdir0/blk_1073741935 for deletion  
168457613801//current/_finalized/subdir0/blk_1073741935  
2023-05-20 16:42:43,560 INFO impl.FsDatasetAsyncDiskService: Scheduling blk_1073741935_1111 replica FinalizedReplica, blk_1073741935_1111, FINALIZED  
getHmabytes() = 4558  
getBytesOnDisk() = 4558  
getVisibleLength() = 4558  
getVolume() = C:/hadoop/hadoop-3.3.5/data/datanode  
getBlockURI() = file:///C:/hadoop/hadoop-3.3.5/data/datanode/current/BP-1673887656-192.168.56.1-1684576138017/current/_finalized/subdir0/subdir0/blk_1073741935 for deletion  
2023-05-20 16:42:43,561 INFO impl.FsDatasetAsyncDiskService: Deleted BP-1673887656-192.168.56.1-1684576138017 blk_1073741932_1108 URI file:///C:/hadoop/hadoop-3.3.5/data/datanode/current/BP-1673887656-192.168.56.1-1684576138017/current/_finalized/subdir0/subdir0/blk_1073741932 for deletion  
168457613801//current/_finalized/subdir0/blk_1073741932  
2023-05-20 16:42:43,562 INFO impl.FsDatasetAsyncDiskService: Deleted BP-1673887656-192.168.56.1-1684576138017 blk_1073741934_1109 URI file:///C:/hadoop/hadoop-3.3.5/data/datanode/current/BP-1673887656-192.168.56.1-1684576138017/current/_finalized/subdir0/subdir0/blk_1073741934 for deletion  
168457613801//current/_finalized/subdir0/blk_1073741934  
2023-05-20 16:42:43,563 INFO impl.FsDatasetAsyncDiskService: Deleted BP-1673887656-192.168.56.1-1684576138017 blk_1073741935_1111 URI file:///C:/hadoop/hadoop-3.3.5/data/datanode/current/BP-1673887656-192.168.56.1-1684576138017/current/_finalized/subdir0/subdir0/blk_1073741935 for deletion  
168457613801//current/_finalized/subdir0/blk_1073741935
```

The Hadoop Resourcemanager starts

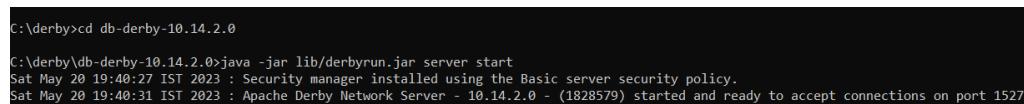
```
Apache Hadoop Distribution - yarn resourcemanager
2023-05-20 16:42:25,611 INFO attempt.RMAppAttemptImpl: appattempt_1684579414743_0008 State change from LAUNCHED to RUNNING on event = REGISTERED
2023-05-20 16:42:27,093 INFO rmapp.RMAppImpl: application_1684579414743_0008 State change from ACCEPTED to RUNNING on event = ATTEMPT_REGISTERED
2023-05-20 16:42:27,093 INFO container.ContainerAllocator: assigned container application attempt=appattempt_1684579414743_0008_000001 container=null queue=default clusterResource=<memory:8192, vCores:8>
type=DF, SNITCH=nodemanagedPartition
2023-05-20 16:42:27,094 INFO rmcontainer.RMContainerImpl: container_1684579414743_0008_01_000002 Container transitioned from NEW to ALLOCATED
2023-05-20 16:42:27,095 INFO fisca.FiCaSchedulerNode: Assigned container container_1684579414743_0008_01_000002 of capacity <memory:1024, vCores:1> on host LAPTOP-51Q5CHA:54092, which has 2 containers, <memory:32768, vCores:2> available after allocation
2023-05-20 16:42:27,095 INFO capacity.ParentQueue: assignedContainer queue=0.375 absoluteUsedCapacity=0.375 used=<memory:3072, vCores:2> cluster=<memory:8192, vCores:8>
2023-05-20 16:42:27,095 INFO container.ContainerImpl: container_1684579414743_0008_01_000002 Container transitioned from ALLOCATED to ACQUIRED
2023-05-20 16:42:28,197 INFO rmcontainer.RMContainerImpl: container_1684579414743_0008_01_000002 Container transitioned from ACQUIRED to RUNNING
2023-05-20 16:42:28,692 INFO scheduler.AppschedulingInfo: checking for deactivation of application: application_1684579414743_0008
2023-05-20 16:42:35,015 INFO rmcontainer.RMContainerImpl: container_1684579414743_0008_01_000002 Container transitioned from RUNNING to COMPLETED
2023-05-20 16:42:39,669 INFO attempt.RMAppAttemptImpl: updating application attempt=appattempt_1684579414743_0008_000001 with final state: FINISHING, and exit status: -1000
2023-05-20 16:42:39,669 INFO attempt.RMAppAttemptImpl: application_1684579414743_0008_000001 State change from RUNNING to FINAL_SAVING on event = UNREGISTERED
2023-05-20 16:42:39,670 INFO rmapp.RMAppImpl: updating application application_1684579414743_0008_000001 with final state: FINISHING
2023-05-20 16:42:39,670 INFO recovery.RMStateStore: Updating info for application_1684579414743_0008_000001
2023-05-20 16:42:39,670 INFO recovery.RMStateStore: Application application_1684579414743_0008_000001 transitioned to FINAL_SAVING on event = ATTEMPT_UNREGISTERED
2023-05-20 16:42:39,674 INFO attempt.RMAppAttemptImpl: appattempt_1684579414743_0008_000001 State change from FINAL_SAVING to FINISHING on event = ATTEMPT_UPDATE_SAVED
2023-05-20 16:42:39,674 INFO rmapp.RMAppImpl: application_1684579414743_0008 State change from FINAL_SAVING to FINISHING on event = APP_UPDATE_SAVED
2023-05-20 16:42:46,686 INFO resourcemanager.ApplicationMasterService: application_1684579414743_0008 unregistered successfully.
2023-05-20 16:42:45,731 INFO ipc.Server: Socket Reader #0 for port 8080: readAndProcess from client 192.168.56.1:55923 threw exception [java.io.IOException: An existing connection was forcibly closed by the remote host]
java.io.IOException: An existing connection was forcibly closed by the remote host
    at sun.nio.ch.SocketDispatcher.read0(Native Method)
    at sun.nio.ch.SocketDispatcher.read(SocketDispatcher.java:43)
    at sun.nio.ch.IOUtil.readIntoNativeBuffer(IOUtil.java:223)
    at sun.nio.ch.IOUtil.read(IOUtil.java:197)
    at sun.nio.ch.SocketChannelImpl.read(SocketChannelImpl.java:378)
    at org.apache.hadoop.ipc.Server$channelRead(Server.java:170)
    at org.apache.hadoop.ipc.Client$Connection$Reader$1.run(Client.java:118)
    at org.apache.hadoop.ipc.$Connection.readAndRunProcess(Server.java:2312)
    at org.apache.hadoop.ipc.$Connection$Listener.doRead(Server.java:1495)
    at org.apache.hadoop.ipc.$Server$ListenerReader.run(Server.java:1350)
    at org.apache.hadoop.ipc.$Server$ListenerReader.run(Server.java:1321)
2023-05-20 16:42:45,782 INFO resourcemanager.ApplicationMasterService: Unregistering app attempt : appattempt_1684579414743_0008_000001
2023-05-20 16:42:45,782 INFO attempt.RMAppAttemptImpl: application_1684579414743_0008_000001 State change from FINISHING to FINISHED on event = COMPLETED
2023-05-20 16:42:45,782 INFO rmapp.RMAppDescriptor: Application finished, removing record for appattempt_1684579414743_0008_000001
2023-05-20 16:42:45,786 INFO attempt.RMAppAttemptImpl: appattempt_1684579414743_0008_000001 State change from FINISHING to FINISHED on event = CONTAINER_FINISHED
2023-05-20 16:42:45,786 INFO rmapp.RMAppImpl: application_1684579414743_0008_000001 State change from FINISHING to FINISHED on event = ATTEMPT_FINISHED
2023-05-20 16:42:45,788 INFO capacity.CapacityScheduler: Application Attempt appattempt_1684579414743_0008_000001 is done. FinalState=FINISHED
2023-05-20 16:42:45,789 INFO resourcemanager.RMAppManager:$ApplicationSummary: appid=application_1684579414743_0008, name=claims_jar, user=Sheeba, queue=default, state=FINISHED, trackingUrl=http://LAPTOP-51Q5CHA:8088/proxy/application_1684579414743_0008/appMasterHost=LAPTOP-51Q5CHA, submitTime=1684581135943, startTime=1684581135946, launchTime=1684581136016, finishTime=1684581135967, finalStatus=SUCCEEDED, memorySeconds=69133, vCoreSeconds=36, processSeconds=1684581135943, mapredAppId=appattempt_1684579414743_0008, mapredAppMemorySeconds=<memory:0>, vCores=0, applicationType=MAPREDUCE, resourceSeconds=69133 MB-seco
2023-05-20 16:42:45,789 INFO scheduler.AppschedulingInfo: Application application_1684579414743_0008 requests cleared
2023-05-20 16:42:45,789 INFO amlauncher.AMLauncher: Cleaning master appattempt_1684579414743_0008_000001
2023-05-20 16:42:45,792 INFO capacity.LeafQueue: Application removed - appId: application_1684579414743_0008 user: Sheeba queue: root.default #user-pending-applications: 0 #user-active-applications: 0
2023-05-20 16:42:45,794 INFO capacity.ParentQueue: Application removed - appId: application_1684579414743_0008 user: Sheeba leaf-queue of parent: root #applications: 0
```

The Hadoop Nodemanager starts



```
Apache Hadoop Distribution - yarn nodemanager
2023-05-20 16:42:37.927 INFO nodemanager.NodeStatusUpdaterImpl: Removed completed containers from NM context: [container_1684579414743_0008_01_000002]
2023-05-20 16:42:37.324 INFO util.ProcfsBasedProcessTree: ProcfsBasedProcessTree currently is supported only on Linux.
2023-05-20 16:42:40.591 INFO util.ProcfsBasedProcessTree: ProcfsBasedProcessTree currently is supported only on Linux.
2023-05-20 16:42:40.591 INFO launcher.ContainerLaunch: Could not get pid for container_1684579414743_0008_01_000002. Waited for 5000 ms.
2023-05-20 16:42:40.605 INFO nodemanager.DefaultContainerExecutor: Deleting absolute path : /tmp/hadoop-Sheeba/nm-local-dir/usercache/Sheeba/appcache/application_1684579414743_0008/container_1684579414743_0008_01_000002/launch_container.cmd
2023-05-20 16:42:40.605 INFO nodemanager.DefaultContainerExecutor: delete returned false for path: [/tmp/hadoop-Sheeba/nm-local-dir/usercache/Sheeba/appcache/application_1684579414743_0008/container_1684579414743_0008_01_000002/launch_container.cmd]
2023-05-20 16:42:40.605 INFO nodemanager.DefaultContainerExecutor: Deleting absolute path : /tmp/hadoop-Sheeba/nm-local-dir/usercache/Sheeba/appcache/application_1684579414743_0008/container_1684579414743_0008_01_000002/container_tokens
2023-05-20 16:42:40.606 WARN nodemanager.DefaultContainerExecutor: delete returned false for path: [/tmp/hadoop-Sheeba/nm-local-dir/usercache/Sheeba/appcache/application_1684579414743_0008/container_1684579414743_0008_01_000002/container_tokens]
2023-05-20 16:42:40.607 INFO nodemanager.DefaultContainerExecutor: Deleting absolute path : /tmp/hadoop-Sheeba/nm-local-dir/usercache/Sheeba/appcache/application_1684579414743_0008/container_1684579414743_0008_01_000002/sysfs
2023-05-20 16:42:40.611 WARN nodemanager.DefaultContainerExecutor: delete returned false for path: [/tmp/hadoop-Sheeba/nm-local-dir/usercache/Sheeba/appcache/application_1684579414743_0008/container_1684579414743_0008_01_000002/sysfs]
2023-05-20 16:42:43.849 INFO util.ProcfsBasedProcessTree: ProcfsBasedProcessTree currently is supported only on Linux.
2023-05-20 16:42:43.849 INFO launcher.ContainerLaunch: Container container_1684579414743_0008_01_000001 transitioned from RUNNING to EXITED_WITH_SUCCESS
2023-05-20 16:42:45.773 INFO container.ContainerImpl: Container container_1684579414743_0008_01_000001 transitioned from EXITED_WITH_SUCCESS to DONE
2023-05-20 16:42:45.776 INFO container.ContainerImpl: Container container_1684579414743_0008_01_000001 transitioned from RUNNING to APPPLICATION_RESOURCES_CLEANINGUP
2023-05-20 16:42:45.776 INFO nodemanager.DefaultContainerExecutor: Deleting absolute path : /tmp/hadoop-Sheeba/nm-local-dir/usercache/Sheeba/appcache/application_1684579414743_0008/container_1684579414743_0008_01_000001
2023-05-20 16:42:45.780 INFO application.ApplicationImpl: Removing container_1684579414743_0008_01_000001 from application application_1684579414743_0008
2023-05-20 16:42:45.781 INFO monitor.ContainerMonitorImpl: Stopping resource monitoring for container_1684579414743_0008_01_000001
2023-05-20 16:42:45.782 INFO container.ContainerImpl: Got event CONTAINER_STOP for application application_1684579414743_0008_01_000001
2023-05-20 16:42:45.782 INFO ipc.Server: Auth successful for appattempt_1684579414743_0008_000001 (auth:SIMPLIFIED) from 192.168.50.1:55844
2023-05-20 16:42:45.786 INFO ipc.Server: Stopped running with application application_1684579414743_0008_01_000001
2023-05-20 16:42:45.773 INFO nodemanager.NodeStatusUpdaterImpl: Removed deleted containers from NM context: [container_1684579414743_0008_01_000001]
2023-05-20 16:42:45.786 INFO application.ApplicationImpl: Application application_1684579414743_0008 transitioned from RUNNING to APPLICATION_RESOURCES_CLEANINGUP
2023-05-20 16:42:46.788 INFO nodemanager.DefaultContainerExecutor: Deleting absolute path : /tmp/hadoop-Sheeba/nm-local-dir/usercache/Sheeba/appcache/application_1684579414743_0008
2023-05-20 16:42:46.788 INFO containerManager.AuxServices: Got event APPLICATION_STOP for apid application_1684579414743_0008
2023-05-20 16:42:46.794 INFO loghandler.NonAggregatingLogHandler: Scheduling Log Deletion for application: application_1684579414743_0008, with delay of 10800 seconds
2023-05-20 16:42:51.334 INFO launcher.ContainerLaunch: Could not get pid for container_1684579414743_0008_01_000001. Waited for 5000 ms.
2023-05-20 16:42:51.349 INFO nodemanager.DefaultContainerExecutor: Deleting absolute path : /tmp/hadoop-Sheeba/nm-local-dir/usercache/Sheeba/appcache/application_1684579414743_0008/container_1684579414743_0008_01_000001/launch_container.cmd
2023-05-20 16:42:51.349 INFO nodemanager.DefaultContainerExecutor: Deleting absolute path : /tmp/hadoop-Sheeba/nm-local-dir/usercache/Sheeba/appcache/application_1684579414743_0008/container_1684579414743_0008_01_000001/container_tokens
2023-05-20 16:42:51.349 INFO nodemanager.DefaultContainerExecutor: delete returned false for path: [/tmp/hadoop-Sheeba/nm-local-dir/usercache/Sheeba/appcache/application_1684579414743_0008/container_1684579414743_0008_01_000001/container_tokens]
2023-05-20 16:42:51.355 WARN nodemanager.DefaultContainerExecutor: Deleting absolute path : /tmp/hadoop-Sheeba/nm-local-dir/usercache/Sheeba/appcache/application_1684579414743_0008
2023-05-20 16:42:51.355 WARN nodemanager.DefaultContainerExecutor: delete returned false for path: [/tmp/hadoop-Sheeba/nm-local-dir/usercache/Sheeba/appcache/application_1684579414743_0008/container_1684579414743_0008_01_000001/sysfs]
2023-05-20 16:43:33.525 INFO localizer.ResourceLocalizationService: Cache Size Before Clean: 12765144, Total Deleted: 0, Public Deleted: 0, Private Deleted: 0
```

Derby is installed and started



```
C:\derby>cd db-derby-10.14.2.0
C:\derby\db-derby-10.14.2.0>java -jar lib/derbyrun.jar server start
Sat May 20 19:40:27 IST 2023 : Security manager installed using the Basic server security policy.
Sat May 20 19:40:31 IST 2023 : Apache Derby Network Server - 10.14.2.0 - (1828579) started and ready to accept connections on port 1527
```

```
[Administrator: Command Prompt - startNetworkServer -h 0.0.0.0]
Microsoft Windows [Version 10.0.19045.2965]
(c) Microsoft Corporation. All rights reserved.

C:\WINDOWS\system32>cd ..

C:\Windows>cd ..\

C:>cd derby\db-derby-10.14.2.0\bin\startNetworkServer -h 0.0.0.0
The system cannot find the path specified.

C:>cd derby\db-derby-10.14.2.0\bin

C:\derby\db-derby-10.14.2.0\bin>startNetworkServer -h 0.0.0.0
Sun May 21 15:34:53 IST 2023 : Security manager installed using the Basic server security policy.
Sun May 21 15:34:53 IST 2023 : Apache Derby Network Server - 10.14.2.0 - (1828579) started and ready to accept connections on port 1527
```

Hive is started

```
C:\hive\apache-hive-3.1.2-bin\bin>hive
2023-05-21 15:35:55,800 INFO conf.HiveConf: Found configuration file file:/C:/hive/apache-hive-3.1.2-bin/conf/hive-site.xml
2023-05-21 15:35:56,087 WARN conf.HiveConf: HiveConf of name hive.server2.enable.impersonation does not exist
2023-05-21 15:35:57,760 WARN conf.HiveConf: HiveConf of name hive.server2.enable.impersonation does not exist
Hive Session ID = 4a0eefb1-9567-4b2e-ad8a-aabf77b4f9fc
2023-05-21 15:35:57,779 INFO SessionState: Hive Session ID = 4a0eefb1-9567-4b2e-ad8a-aabf77b4f9fc

Logging initialized using configuration in jar:file:/C:/hive/apache-hive-3.1.2-bin/lib/hive-common-3.1.2.jar!/hive-log4j2.properties Async: true
2023-05-21 15:35:57,843 INFO SessionState:
Logging initialized using configuration in jar:file:/C:/hive/apache-hive-3.1.2-bin/lib/hive-common-3.1.2.jar!/hive-log4j2.properties Async: true
2023-05-21 15:35:58,777 INFO session.SessionState: Created HDFS directory: /tmp/hive/Sheeba/4a0eefb1-9567-4b2e-ad8a-aabf77b4f9fc
2023-05-21 15:35:58,782 INFO session.SessionState: Created local directory: C:/Users/Sheeba/AppData/Local/Temp/Sheeba/4a0eefb1-9567-4b2e-ad8a-aabf77b4f9fc/_tmp_space.db
2023-05-21 15:35:58,785 INFO session.SessionState: Created HDFS directory: /tmp/hive/Sheeba/4a0eefb1-9567-4b2e-ad8a-aabf77b4f9fc/_tmp_space.db
2023-05-21 15:35:58,798 INFO conf.HiveConf: Using the default value passed in for log id: 4a0eefb1-9567-4b2e-ad8a-aabf77b4f9fc
2023-05-21 15:35:58,798 INFO session.SessionState: Updating thread name to 4a0eefb1-9567-4b2e-ad8a-aabf77b4f9fc main
2023-05-21 15:35:58,830 WARN conf.HiveConf: HiveConf of name hive.server2.enable.impersonation does not exist
2023-05-21 15:35:59,790 INFO metastore.HiveMetaStore: 0: Opening raw store with implementation class:org.apache.hadoop.hive.metastore.ObjectStore
2023-05-21 15:35:59,815 WARN metastore.ObjectStore: datanucleus.autoStartMechanismMode is set to unsupported value null . Setting it to value: ignored
2023-05-21 15:35:59,820 INFO metastore.ObjectStore: ObjectStore, initialize called
2023-05-21 15:35:59,822 INFO conf.MetastoreConf: Found configuration file file:/C:/hive/apache-hive-3.1.2-bin/conf/hive-site.xml
2023-05-21 15:35:59,823 INFO conf.MetastoreConf: Unable to find config file hivemetastore-site.xml
2023-05-21 15:35:59,823 INFO conf.MetastoreConf: Found configuration file null
2023-05-21 15:35:59,824 INFO conf.MetastoreConf: Unable to find config file metastore-site.xml
2023-05-21 15:35:59,824 INFO conf.MetastoreConf: Found configuration file null
2023-05-21 15:36:00,007 INFO DataNucleus.Persistence: Property datanucleus.cache.level2 unknown - will be ignored
2023-05-21 15:36:00,265 INFO hikari.HikariDataSource: HikariPool-1 - Starting...
```

Schematool is initialized

```
Sheeba@LAPTOP-51QJSCHA /cygdrive/c/hive/apache-hive-3.1.2-bin/bin
$ schematool -dbType derby -initSchema
WARNING: DEFAULT_LIBEXEC_DIR ignored. It has been replaced by HADOOP_DEFAULT_LIBEXEC_DIR.
Metastore connection URL:      jdbc:derby://localhost:1527/metastore_db;create=true
Metastore Connection Driver :  org.apache.derby.jdbc.ClientDriver
Metastore connection User:    APP
Starting metastore schema initialization to 3.1.0
Initialization script hive-schema-3.1.0.derby.sql
```

```
Initialization script completed
schemaTool completed

Sheeba@LAPTOP-51QJSCHA /cygdrive/c/hive/apache-hive-3.1.2-bin/bin
```

hive server2 starts

```
C:\Administrator:Command Prompt - hive --service hiveserver2 start
Microsoft Windows [Version 10.0.19045.2965]
(c) Microsoft Corporation. All rights reserved.

C:\WINDOWS\system32>hive --service hiveserver2 start
2023-05-21 16:55:28,909 INFO conf.HiveConf: Found configuration file file:/C:/hive/apache-hive-3.1.2-bin/conf/hive-site.xml
2023-05-21 16:55:29,287 WARN conf.HiveConf: HiveConf of name hive.server2.enable.impersonation does not exist
2023-05-21 16:55:30,802 INFO server.HiveServer2: STARTUP_MSG:
/************************************************************
STARTUP_MSG: Starting HiveServer2
STARTUP_MSG:   host = LAPTOP-51QJSCHA/192.168.56.1
STARTUP_MSG:   args = [start]
STARTUP_MSG:   version = 3.1.2
```

```

2023-05-21 16:55:30,834 INFO server.HiveServer2: Starting HiveServer2
2023-05-21 16:55:30,834 WARN conf.HiveConf: hiveConf of name hive.server2.enable.impersonation does not exist
HiveServer2 started at: 26c20e5f-6b85-4dbd-acab-d449892ddfe7
2023-05-21 16:55:30,970 INFO session.SessionState: Hive Session ID = 26c20e5f-6b85-4dbd-acab-d449892ddfe7
2023-05-21 16:55:31,903 INFO session.SessionState: Created local directory: C:/Users/Sheeba/AppData/Local/Temp/Sheeba/26c20e5f-6b85-4dbd-acab-d449892ddfe7/tmp_space.db
2023-05-21 16:55:31,916 INFO session.SessionState: Created HDFS directory: /tmp/hive/sheeba/26c20e5f-6b85-4dbd-acab-d449892ddfe7/tmp_space.db
2023-05-21 16:55:31,940 INFO sqlst.SQLStdHiveAccessController: Created SQLStdHiveAccessController for session context : HiveAuthzSessionContext [sessionString=26c20e5f-6b85-4dbd-acab-d449892ddfe7, clientType=HIVESERVER2]
2023-05-21 16:55:31,945 WARN session.SessionState: METASTORE_FILTER_HOOK will be ignored, since hive.security.authorization.manager is set to instance of HiveAuthorizerFactory.
2023-05-21 16:55:32,869 INFO metastore.HiveMetaStore: 0: Opening raw store with implementation class:org.apache.hadoop.hive.metastore.ObjectStore
2023-05-21 16:55:32,895 WARN metastore.ObjectStore: datanucleus.autoStartMechanismMode is set to unsupported value null . Setting it to value: ignored
2023-05-21 16:55:32,902 INFO metastore.ObjectStore: initialize called
2023-05-21 16:55:32,902 INFO metastore.ObjectStore: Found configuration file:C:/hive/apache-hive-3.1.2-bin/conf/hive-site.xml
2023-05-21 16:55:32,905 INFO conf.MetastoreConf: Unable to find config file hivemetastore-site.xml
2023-05-21 16:55:32,905 INFO MetastoreConf: Found configuration file null
2023-05-21 16:55:32,906 INFO conf.MetastoreConf: Found configuration file metastore-site.xml
2023-05-21 16:55:32,906 INFO DataNucleus.Persistence: Property datanucleus.cache.level2 unknown - will be ignored
2023-05-21 16:55:33,382 INFO hikari.HikariDataSource: HikariPool-1 - Starting...
2023-05-21 16:55:33,516 INFO pool.PoolBase: HikariPool-1 - Driver does not support get/set network timeout for connections. (Feature not implemented: getNetworkTimeout.)
2023-05-21 16:55:33,539 INFO hikari.HikariDataSource: HikariPool-1 - Start completed.
2023-05-21 16:55:33,552 INFO hikari.HikariDataSource: HikariPool-2 - Starting...
2023-05-21 16:55:33,552 INFO pool.PoolBase: HikariPool-2 - Driver does not support get/set network timeout for connections. (Feature not implemented: getNetworkTimeout.)
2023-05-21 16:55:33,552 INFO pool.PoolBase: HikariPool-2 - Start completed.
2023-05-21 16:55:33,746 INFO metastore.ObjectStore: Setting MetaStore object pin classes with hive.metastore.cache.pinobjtypes="Table,StorageDescriptor,SerDeInfo,Partition,Database,Type,FieldSchema,Order"
2023-05-21 16:55:33,870 INFO metastore.MetaStoreDirectSql: Using direct SQL, underlying DB is DERBY
2023-05-21 16:55:33,871 INFO metastore.ObjectStore: Initialized ObjectStore

```

```

Administrator: Command Prompt - hive --service hiveserver2 start
2023-05-21 17:19:38,920 INFO metastore.HiveMetaStore: 2: get_config_value: name=metastore.batch.retrieve.max.defaultValue=50
2023-05-21 17:19:38,920 INFO HiveMetaStore.audit: ugi=Sheeba ip=unknown-ip-addr cmd=get_config_value: name=metastore.batch.retrieve.max.defaultValue=50
2023-05-21 17:20:38,917 INFO metastore.HiveMetaStore: 2: get_config_value: name=metastore.batch.retrieve.max.defaultValue=50
2023-05-21 17:20:38,917 INFO HiveMetaStore.audit: ugi=Sheeba ip=unknown-ip-addr cmd=get_config_value: name=metastore.batch.retrieve.max.defaultValue=50
2023-05-21 17:21:38,908 INFO metastore.HiveMetaStore: 2: get_config_value: name=metastore.batch.retrieve.max.defaultValue=50
2023-05-21 17:21:38,908 INFO HiveMetaStore.audit: ugi=Sheeba ip=unknown-ip-addr cmd=get_config_value: name=metastore.batch.retrieve.max.defaultValue=50
2023-05-21 17:22:38,909 INFO metastore.HiveMetaStore: 2: get_config_value: name=metastore.batch.retrieve.max.defaultValue=50
2023-05-21 17:22:38,909 INFO HiveMetaStore.audit: ugi=Sheeba ip=unknown-ip-addr cmd=get_config_value: name=metastore.batch.retrieve.max.defaultValue=50
2023-05-21 17:23:38,925 INFO metastore.HiveMetaStore: 2: get_config_value: name=metastore.batch.retrieve.max.defaultValue=50
2023-05-21 17:23:38,925 INFO HiveMetaStore.audit: ugi=Sheeba ip=unknown-ip-addr cmd=get_config_value: name=metastore.batch.retrieve.max.defaultValue=50
2023-05-21 17:24:38,912 INFO metastore.HiveMetaStore: 2: get_config_value: name=metastore.batch.retrieve.max.defaultValue=50
2023-05-21 17:24:38,912 INFO HiveMetaStore.audit: ugi=Sheeba ip=unknown-ip-addr cmd=get_config_value: name=metastore.batch.retrieve.max.defaultValue=50
2023-05-21 17:25:38,921 INFO metastore.HiveMetaStore: 2: get_config_value: name=metastore.batch.retrieve.max.defaultValue=50
2023-05-21 17:25:38,921 INFO HiveMetaStore.audit: ugi=Sheeba ip=unknown-ip-addr cmd=get_config_value: name=metastore.batch.retrieve.max.defaultValue=50
2023-05-21 17:26:38,921 INFO metastore.HiveMetaStore: 2: get_config_value: name=metastore.batch.retrieve.max.defaultValue=50
2023-05-21 17:26:38,921 INFO HiveMetaStore.audit: ugi=Sheeba ip=unknown-ip-addr cmd=get_config_value: name=metastore.batch.retrieve.max.defaultValue=50
2023-05-21 17:27:38,917 INFO metastore.HiveMetaStore: 2: get_config_value: name=metastore.batch.retrieve.max.defaultValue=50
2023-05-21 17:27:38,917 INFO HiveMetaStore.audit: ugi=Sheeba ip=unknown-ip-addr cmd=get_config_value: name=metastore.batch.retrieve.max.defaultValue=50
2023-05-21 17:28:38,918 INFO metastore.HiveMetaStore: 2: get_config_value: name=metastore.batch.retrieve.max.defaultValue=50
2023-05-21 17:28:38,918 INFO HiveMetaStore.audit: ugi=Sheeba ip=unknown-ip-addr cmd=get_config_value: name=metastore.batch.retrieve.max.defaultValue=50
2023-05-21 17:28:38,913 INFO metastore.HiveMetaStore: 2: get_config_value: name=metastore.batch.retrieve.max.defaultValue=50
2023-05-21 17:28:38,913 INFO HiveMetaStore.audit: ugi=Sheeba ip=unknown-ip-addr cmd=get_config_value: name=metastore.batch.retrieve.max.defaultValue=50
2023-05-21 17:29:38,915 INFO metastore.HiveMetaStore: 2: get_config_value: name=metastore.batch.retrieve.max.defaultValue=50
2023-05-21 17:29:38,915 INFO HiveMetaStore.audit: ugi=Sheeba ip=unknown-ip-addr cmd=get_config_value: name=metastore.batch.retrieve.max.defaultValue=50
2023-05-21 17:30:38,913 INFO metastore.HiveMetaStore: 2: get_config_value: name=metastore.batch.retrieve.max.defaultValue=50
2023-05-21 17:30:38,913 INFO HiveMetaStore.audit: ugi=Sheeba ip=unknown-ip-addr cmd=get_config_value: name=metastore.batch.retrieve.max.defaultValue=50
2023-05-21 17:31:38,916 INFO metastore.HiveMetaStore: 2: get_config_value: name=metastore.batch.retrieve.max.defaultValue=50
2023-05-21 17:31:38,916 INFO HiveMetaStore.audit: ugi=Sheeba ip=unknown-ip-addr cmd=get_config_value: name=metastore.batch.retrieve.max.defaultValue=50
2023-05-21 17:32:38,933 INFO metastore.HiveMetaStore: 2: get_config_value: name=metastore.batch.retrieve.max.defaultValue=50
2023-05-21 17:32:38,933 INFO HiveMetaStore.audit: ugi=Sheeba ip=unknown-ip-addr cmd=get_config_value: name=metastore.batch.retrieve.max.defaultValue=50
2023-05-21 17:33:38,917 INFO metastore.HiveMetaStore: 2: get_config_value: name=metastore.batch.retrieve.max.defaultValue=50
2023-05-21 17:33:38,917 INFO HiveMetaStore.audit: ugi=Sheeba ip=unknown-ip-addr cmd=get_config_value: name=metastore.batch.retrieve.max.defaultValue=50
2023-05-21 17:34:38,911 INFO metastore.HiveMetaStore: 2: get_config_value: name=metastore.batch.retrieve.max.defaultValue=50
2023-05-21 17:34:38,911 INFO HiveMetaStore.audit: ugi=Sheeba ip=unknown-ip-addr cmd=get_config_value: name=metastore.batch.retrieve.max.defaultValue=50
2023-05-21 17:35:38,911 INFO metastore.HiveMetaStore: 2: get_config_value: name=metastore.batch.retrieve.max.defaultValue=50
2023-05-21 17:35:38,911 INFO HiveMetaStore.audit: ugi=Sheeba ip=unknown-ip-addr cmd=get_config_value: name=metastore.batch.retrieve.max.defaultValue=50

```

Database healthcare_System is created in hive:

```
hive> create database healthcare_System;
2023-05-21 17:06:18.846 INFO conf.HiveConf: Using the default value passed in for log id: fe20bda1-led1-49eb-b699-a55e04cafe74
2023-05-21 17:06:18.847 INFO session.SessionState: Updating thread name to fe20bda1-led1-49eb-b699-a55e04cafe74 main
2023-05-21 17:06:18.848 INFO ql.Driver: Compiling command(queryId=Sheeba_20230521170618_a000c7e-e162-45bc-93b5-b5daf81f4d16): create database healthcare_System
2023-05-21 17:06:18.859 INFO ql.Driver: Concurrency mode is disabled, not creating a lock manager
2023-05-21 17:06:18.859 INFO ql.Driver: Semantic Analysis Completed (retryal = false)
2023-05-21 17:06:18.860 INFO ql.Driver: Returning Hive schema: Schema(fieldSchemas:null, properties:null)
2023-05-21 17:06:18.860 INFO ql.Driver: completed executing command(queryId=Sheeba_20230521170618_a000c7e-e162-45bc-93b5-b5daf81f4d16); Time taken: 0.012 seconds
2023-05-21 17:06:18.869 INFO reexec.ReExecDriver: Execution #1 of query
2023-05-21 17:06:18.869 INFO ql.Driver: Concurrency mode is disabled, not creating a lock manager
2023-05-21 17:06:18.869 INFO ql.Driver: Executing command(queryId=Sheeba_20230521170618_a000c7e-e162-45bc-93b5-b5daf81f4d16): create database healthcare_System
2023-05-21 17:06:18.863 INFO metastore.HiveMetaStore: 0: create database: Database(name:healthcare_System, description:null, locationUri:hdfs://localhost:9000/user/hive/warehouse/healthcare_system.db, parameters: null, ownerName:Sheeba, ownerType:USER, catalogName:hive)
2023-05-21 17:06:18.863 INFO HiveMetaStore.audit: ugi=Sheeba ip=unknown-ip-addr cmd=create_database: Database(name:healthcare_System, description:null, locationUri:hdfs://localhost:9000/user/hive/warehouse/healthcare_system.db, parameters: null, ownerName:Sheeba, ownerType:USER, catalogName:hive)
2023-05-21 17:06:18.865 WARN metastore.ObjectStore: Failed to get database hive.healthcare_System, returning NoSuchObjectException
2023-05-21 17:06:18.874 INFO util.FileUtils: Creating directory if it doesn't exist: hdfs://localhost:9000/user/hive/warehouse/healthcare_system.db
2023-05-21 17:06:18.880 INFO ql.Driver: Completed executing command(queryId=Sheeba_20230521170618_a000c7e-e162-45bc-93b5-b5daf81f4d16); Time taken: 0.019 seconds
OK
2023-05-21 17:06:18.861 INFO ql.Driver: Starting task [Stage-0:DDL] in serial mode
Time taken: 0.035 seconds
2023-05-21 17:06:18.883 INFO CliDriver: Time taken: 0.035 seconds
2023-05-21 17:06:18.884 INFO conf.HiveConf: Using the default value passed in for log id: fe20bda1-led1-49eb-b699-a55e04cafe74
2023-05-21 17:06:18.885 INFO session.SessionState: Resetting thread name to  main
```

```
hive> use healthcare_System;
2023-05-21 17:10:48.881 INFO conf.HiveConf: Using the default value passed in for log id: fe20bda1-led1-49eb-b699-a55e04cafe74
2023-05-21 17:10:48.881 INFO session.SessionState: Updating thread name to fe20bda1-led1-49eb-b699-a55e04cafe74 main
2023-05-21 17:10:48.882 INFO ql.Driver: Compiling command(queryId=Sheeba_20230521171048_d56cb87a-c947-410f-8625-95fd8493f3d): use healthcare_System
2023-05-21 17:10:48.897 INFO ql.Driver: Concurrency mode is disabled, not creating a lock manager
2023-05-21 17:10:48.898 INFO metastore.HiveMetaStore: 0: get_database: @hive@healthcare_System
2023-05-21 17:10:48.898 INFO HiveMetaStore.audit: ugi=Sheeba ip=unknown-ip-addr cmd=get_database: @hive@healthcare_System
2023-05-21 17:10:48.902 INFO ql.Driver: Semantic Analysis Completed (retryal = false)
2023-05-21 17:10:48.902 INFO ql.Driver: Returning Hive schema: Schema(fieldSchemas:null, properties:null)
2023-05-21 17:10:48.902 INFO ql.Driver: Completed compiling command(queryId=Sheeba_20230521171048_d56cb87a-c947-410f-8625-95fd8493f3d); Time taken: 0.02 seconds
2023-05-21 17:10:48.902 INFO reexec.ReExecDriver: Execution #1 of query
2023-05-21 17:10:48.902 INFO ql.Driver: Concurrency mode is disabled, not creating a lock manager
2023-05-21 17:10:48.902 INFO ql.Driver: Executing command(queryId=Sheeba_20230521171048_d56cb87a-c947-410f-8625-95fd8493f3d): use healthcare_System
2023-05-21 17:10:48.902 INFO ql.Driver: Starting task [Stage-0:DML] in serial mode
2023-05-21 17:10:48.903 INFO metastore.HiveMetaStore: 0: get_database: @hive@healthcare_System
2023-05-21 17:10:48.903 INFO HiveMetaStore.audit: ugi=Sheeba ip=unknown-ip-addr cmd=get_database: @hive@healthcare_System
2023-05-21 17:10:48.905 INFO metastore.HiveMetaStore: 0: get_database: @hive@healthcare_System
2023-05-21 17:10:48.905 INFO HiveMetaStore.audit: ugi=Sheeba ip=unknown-ip-addr cmd=get_database: @hive@healthcare_System
2023-05-21 17:10:48.917 INFO ql.Driver: Completed executing command(queryId=Sheeba_20230521171048_d56cb87a-c947-410f-8625-95fd8493f3d); Time taken: 0.015 seconds
OK
2023-05-21 17:10:48.918 INFO ql.Driver: OK
2023-05-21 17:10:48.918 INFO ql.Driver: Concurrency mode is disabled, not creating a lock manager
Time taken: 0.036 seconds
2023-05-21 17:10:48.918 INFO CliDriver: Time taken: 0.036 seconds
2023-05-21 17:10:48.918 INFO conf.HiveConf: Using the default value passed in for log id: fe20bda1-led1-49eb-b699-a55e04cafe74
2023-05-21 17:10:48.918 INFO session.SessionState: Resetting thread name to  main
```

Importing the mysql tables using sqoop into hive tables:

```
C:\Administrator: Command Prompt
Microsoft Windows [Version 10.0.19045.2965]
(c) Microsoft Corporation. All rights reserved.

C:\WINDOWS\system32>sqoop import --connect jdbc:mysql://localhost/_health_care__ --username root --password '' --table groups_tble --hive-import --hive-table healthcare_System.groups -m 1;
Warning: HBASE_HOME and HBASE_VERSION not set.
Warning: HCAT_HOME not set
Warning: HCATALOG_HOME does not exist HCatalog imports will fail.
Please set HCATALOG_HOME to the root of your HCatalog installation.
Warning: ACCUMULO_HOME not set.
Warning: ZOOKEEPER_HOME not set.
Warning: HBASE_HOME does not exist HBase imports will fail.
Please set HBASE_HOME to the root of your HBase installation.
Warning: ACCUMULO_HOME does not exist Accumulo imports will fail.
Please set ACCUMULO_HOME to the root of your Accumulo installation.
Warning: ZOOKEEPER_HOME does not exist Accumulo imports will fail.
Please set ZOOKEEPER_HOME to the root of your Zookeeper installation.
2023-05-21 17:07:44,242 WARN tool.BaseSqoopTool: Setting your password on the command-line is insecure. Consider using -P instead.
2023-05-21 17:07:44,243 INFO tool.BaseSqoopTool: Using Hive-specific delimiters for output. You can override
2023-05-21 17:07:44,243 INFO tool.BaseSqoopTool: delimiters with --fields-terminated-by, etc.
2023-05-21 17:07:44,243 WARN tool.BaseSqoopTool: It seems that you've specified at least one of following:
2023-05-21 17:07:44,243 INFO tool.BaseSqoopTool:   --hive-home
2023-05-21 17:07:44,243 INFO tool.BaseSqoopTool:   --hive-overwrite
2023-05-21 17:07:44,243 INFO tool.BaseSqoopTool:   --create-hive-table
2023-05-21 17:07:44,243 INFO tool.BaseSqoopTool:   --hive-table
2023-05-21 17:07:44,243 INFO tool.BaseSqoopTool:   --hive-partition-key
2023-05-21 17:07:44,243 INFO tool.BaseSqoopTool:   --hive-partition-value
2023-05-21 17:07:44,243 INFO tool.BaseSqoopTool:   --map-column-hive
2023-05-21 17:07:44,244 WARN tool.BaseSqoopTool: Without specifying parameter --hive-import. Please note that
2023-05-21 17:07:44,244 WARN tool.BaseSqoopTool: those arguments will not be used in this session. Either
2023-05-21 17:07:44,244 WARN tool.BaseSqoopTool: specify --hive-import to apply them correctly or remove them
2023-05-21 17:07:44,245 WARN tool.BaseSqoopTool: from command line to remove this warning.
2023-05-21 17:07:44,245 INFO tool.BaseSqoopTool: Please note that --hive-home, --hive-partition-key,
2023-05-21 17:07:44,245 INFO tool.BaseSqoopTool:   --hive-partition-value and --map-column-hive options are
2023-05-21 17:07:44,246 INFO tool.BaseSqoopTool:   are also valid for HCatalog imports and exports
2023-05-21 17:07:44,312 INFO manager.MySQLManager: Preparing to use a MySQL streaming resultset.
2023-05-21 17:07:44,315 INFO tool.CodeGenTool: Beginning code generation
2023-05-21 17:07:44,558 INFO manager.SqlManager: Executing SQL statement: SELECT t.* FROM `groups_tble` AS t LIMIT 1
2023-05-21 17:07:44,580 INFO manager.SqlManager: Executing SQL statement: SELECT t.* FROM `groups_tble` AS t LIMIT 1
2023-05-21 17:07:44,584 INFO orm.CompilationManager: HADOOP_MAPRED_HOME is C:\hadoop\hadoop-3.3.5
Note: /tmp/sqoop-Sheeba/compile\0d6bf94fce5fbcf14a9baecd96a44fa9\groups_tble.java uses or overrides a deprecated API.
Note: Recompiling with -Xlint:deprecation for details.
2023-05-21 17:07:46,215 INFO orm.CompilationManager: Writing jar file: /tmp/sqoop-Sheeba/compile\0d6bf94fce5fbcf14a9baecd96a44fa9\groups_tble.jar
2023-05-21 17:07:46,352 WARN manager.MySQLManager: It looks like you are importing from mysql.
2023-05-21 17:07:46,352 WARN manager.MySQLManager: This transfer can be faster! Use the --direct
2023-05-21 17:07:46,354 WARN manager.MySQLManager: option to exercise a MySQL-specific fast path.
2023-05-21 17:07:46,354 INFO manager.MySQLManager: Setting zero DATETIME behavior to convertToNull (mysql)
2023-05-21 17:07:46,378 INFO mapreduce.ImportJobBase: Beginning import of groups_tble.
2023-05-21 17:07:46,371 INFO Configuration.deprecation: mapred.job.tracker is deprecated. Instead, use mapreduce.jobtracker.address
2023-05-21 17:07:46,478 INFO Configuration.deprecation: mapred.map.tasks is deprecated. Instead, use mapreduce.job.maps
2023-05-21 17:07:47,166 INFO client.DefaultNoHARMailoverProxyProvider: Connecting to ResourceManager at /0.0.0.0:8032
2023-05-21 17:07:47,804 INFO mapreduce.JobResourceUploader: Disabling Erasure Coding for path: /tmp/hadoop-yarn/staging/Sheeba/.staging/job_1684663344794_0010
```

```
C:\WINDOWS\system32>sqoop import --connect jdbc:mysql://localhost/_health_care__ --username root --password '' --table subgroup --hive-import --hive-table healthcare_System.subgroup -m 1;
Warning: HBASE_HOME and HBASE_VERSION not set.
Warning: HCAT_HOME not set
Warning: HCATALOG_HOME does not exist HCatalog imports will fail.
Please set HCATALOG_HOME to the root of your HCatalog installation.
Warning: ACCUMULO_HOME not set.
Warning: ZOOKEEPER_HOME not set.
Warning: HBASE_HOME does not exist HBase imports will fail.
Please set HBASE_HOME to the root of your HBase installation.
Warning: ACCUMULO_HOME does not exist Accumulo imports will fail.
Please set ACCUMULO_HOME to the root of your Accumulo installation.
Warning: ZOOKEEPER_HOME does not exist Accumulo imports will fail.
Please set ZOOKEEPER_HOME to the root of your Zookeeper installation.
2023-05-21 17:14:15,278 WARN tool.BaseSqoopTool: Setting your password on the command-line is insecure. Consider using -P instead.
2023-05-21 17:14:15,279 INFO tool.BaseSqoopTool: Using Hive-specific delimiters for output. You can override
2023-05-21 17:14:15,281 INFO tool.BaseSqoopTool: delimiters with --fields-terminated-by, etc.
2023-05-21 17:14:15,285 WARN tool.BaseSqoopTool: It seems that you've specified at least one of following:
2023-05-21 17:14:15,285 INFO tool.BaseSqoopTool:   --hive-home
2023-05-21 17:14:15,286 INFO tool.BaseSqoopTool:   --hive-overwrite
2023-05-21 17:14:15,287 WARN tool.BaseSqoopTool:   --create-hive-table
2023-05-21 17:14:15,288 WARN tool.BaseSqoopTool:   --hive-table
2023-05-21 17:14:15,288 WARN tool.BaseSqoopTool:   --hive-partition-key
2023-05-21 17:14:15,289 WARN tool.BaseSqoopTool:   --hive-partition-value
2023-05-21 17:14:15,289 WARN tool.BaseSqoopTool:   --map-column-hive
2023-05-21 17:14:15,295 WARN tool.BaseSqoopTool: Without specifying parameter --hive-import. Please note that
2023-05-21 17:14:15,295 WARN tool.BaseSqoopTool: those arguments will not be used in this session. Either
2023-05-21 17:14:15,296 WARN tool.BaseSqoopTool: specify --hive-import to apply them correctly or remove them
2023-05-21 17:14:15,296 WARN tool.BaseSqoopTool: from command line to remove this warning.
2023-05-21 17:14:15,301 INFO tool.BaseSqoopTool: Please note that --hive-home, --hive-partition-key,
2023-05-21 17:14:15,312 INFO tool.BaseSqoopTool:   --hive-partition-value and --map-column-hive options are
2023-05-21 17:14:15,312 INFO tool.BaseSqoopTool:   are also valid for HCatalog imports and exports
2023-05-21 17:14:15,378 INFO manager.MySQLManager: Preparing to use a MySQL streaming resultset.
2023-05-21 17:14:15,374 INFO tool.CodeGenTool: Beginning code generation
2023-05-21 17:14:15,393 INFO manager.SqlManager: Executing SQL statement: SELECT t.* FROM `subgroup` AS t LIMIT 1
2023-05-21 17:14:15,628 INFO manager.SqlManager: Executing SQL statement: SELECT t.* FROM `subgroup` AS t LIMIT 1
2023-05-21 17:14:15,633 INFO orm.CompilationManager: HADOOP_MAPRED_HOME is C:\hadoop\hadoop-3.3.5
Note: /tmp/sqoop-Sheeba/compile\df0f58e1e72bf51ddec12864ea435e9\subgroup.java uses or overrides a deprecated API.
Note: Recompiling with -Xlint:deprecation for details.
2023-05-21 17:14:17,030 INFO orm.CompilationManager: Writing jar file: /tmp/sqoop-Sheeba/compile\df0f58e1e72bf51ddec12864ea435e9\subgroup.jar
2023-05-21 17:14:17,052 WARN manager.MySQLManager: It looks like you are importing from mysql.
2023-05-21 17:14:17,052 WARN manager.MySQLManager: This transfer can be faster! Use the --direct
2023-05-21 17:14:17,053 WARN manager.MySQLManager: option to exercise a MySQL-specific fast path.
2023-05-21 17:14:17,054 INFO manager.MySQLManager: Setting zero DATETIME behavior to convertToNull (mysql)
2023-05-21 17:14:17,071 INFO mapreduce.ImportJobBase: Beginning import of subgroup.
2023-05-21 17:14:17,072 INFO Configuration.deprecation: mapred.job.tracker is deprecated. Instead, use mapreduce.jobtracker.address
2023-05-21 17:14:17,182 INFO Configuration.deprecation: mapred.map.tasks is deprecated. Instead, use mapreduce.job.maps
2023-05-21 17:14:17,781 INFO Configuration.deprecation: mapred.job.maps is deprecated. Instead, use mapreduce.job.maps
2023-05-21 17:14:17,865 INFO client.DefaultNoHARMailoverProxyProvider: Connecting to ResourceManager at /0.0.0.0:8032
2023-05-21 17:14:18,513 INFO mapreduce.JobResourceUploader: Disabling Erasure Coding for path: /tmp/hadoop-yarn/staging/Sheeba/.staging/job_1684663344794_0011
2023-05-21 17:14:18,988 INFO mapreduce.JobSubmitter: number of splits:1
2023-05-21 17:14:19,113 INFO mapreduce.JobSubmitter: Submitting tokens for job: job_1684663344794_0011
```

```

Administrator: Command Prompt
C:\Windows\system32>sqoop import --connect jdbc:mysql://localhost/_health_care__ --username root --password '' --table group_subgroup --hive-import --hive-table healthcare_System.grp_subgrp -m 1;
Warning: HBASE_HOME and HBASE_VERSION not set.
Warning: HCAT_HOME not set
Warning: HCATALOG_HOME does not exist HCatalog imports will fail.
Please set HCATALOG_HOME to the root of your HCatalog installation.
Warning: ACCUMULO_HOME not set
Warning: ZOOKEEPER_HOME not set.
Warning: HBASE_HOME does not exist Hbase imports will fail.
Please set HBASE_HOME to the root of your Hbase installation.
Warning: ACCUMULO_HOME does not exist Accumulo imports will fail.
Please set ACCUMULO_HOME to the root of your Accumulo installation.
Warning: ZOOKEEPER_HOME does not exist Accumulo imports will fail.
Please set ZOOKEEPER_HOME to the root of your Zookeeper installation.
2023-05-21 17:15:37,132 WARN tool.BaseSqoopTool: Setting your password on the command-line is insecure. Consider using -P instead.
2023-05-21 17:15:37,132 INFO tool.BaseSqoopTool: Using Hive-specific delimiters for output. You can override
2023-05-21 17:15:37,132 INFO tool.BaseSqoopTool: delimiters with --fields-terminated-by, etc.
2023-05-21 17:15:37,138 WARN tool.BaseSqoopTool: It seems that you've specified at least one of following:
2023-05-21 17:15:37,139 WARN tool.BaseSqoopTool:   --hive-home
2023-05-21 17:15:37,140 WARN tool.BaseSqoopTool:   --hive-overwrite
2023-05-21 17:15:37,140 INFO tool.BaseSqoopTool:   --create-hive-table
2023-05-21 17:15:37,141 WARN tool.BaseSqoopTool:   --hive-table
2023-05-21 17:15:37,141 WARN tool.BaseSqoopTool:   --hive-partition-key
2023-05-21 17:15:37,142 WARN tool.BaseSqoopTool:   --hive-partition-value
2023-05-21 17:15:37,143 WARN tool.BaseSqoopTool:   --map-column-hive
2023-05-21 17:15:37,144 WARN tool.BaseSqoopTool: Without specifying parameter --hive-import. Please note that
2023-05-21 17:15:37,144 WARN tool.BaseSqoopTool: those arguments will not be used in this session. Either
2023-05-21 17:15:37,145 WARN tool.BaseSqoopTool: specify --hive-import to apply them correctly or remove them
2023-05-21 17:15:37,146 WARN tool.BaseSqoopTool: from command line to remove this warning.
2023-05-21 17:15:37,146 INFO tool.BaseSqoopTool: Please note that --hive-home, --hive-partition-key,
2023-05-21 17:15:37,146 INFO tool.BaseSqoopTool:   --hive-partition-value and --map-column-hive options are
2023-05-21 17:15:37,147 INFO tool.BaseSqoopTool:   are also valid for HCatalog imports and exports
2023-05-21 17:15:37,214 INFO manager.MySQLManager: Preparing to use a MySQL streaming resultset.
2023-05-21 17:15:37,218 INFO tool.CodeGenTool: Beginning code generation
2023-05-21 17:15:37,312 INFO manager.SQLManager: Executing SQL statement: SELECT t.* FROM `group_subgroup` AS t LIMIT 1
2023-05-21 17:15:37,496 INFO manager.SQLManager: Executing SQL statement: SELECT t.* FROM `group_subgroup` AS t LIMIT 1
2023-05-21 17:15:37,500 INFO manager.SQLManager: HADOOP_MAPRED_HOME is C:\hadoop\hadoop-3.3.5
Note: 'lmplsqoop-Sheeba\compile\v1\c4d45119dd8bd0554021692887d75\group_subgroup.java' uses or overrides a deprecated API.
Note: Recompile with -Xlint:deprecation for details.
2023-05-21 17:15:38,825 INFO orm.CompilationManager: Writing jar file: /tmp\sqoop-Sheeba\compile\221c445119d4bad0354021692887d75\group_subgroup.jar
2023-05-21 17:15:38,847 WARN manager.MySQLManager: It looks like you are importing from mysql.
2023-05-21 17:15:38,847 INFO manager.MySQLManager: This transfer can be faster! Use the --direct
2023-05-21 17:15:38,848 WARN manager.MySQLManager: option to exercise a MySQL-specific fast path.
2023-05-21 17:15:38,848 INFO manager.MySQLManager: Setting zero DATETIME behavior to convertToNull (mysql)
2023-05-21 17:15:38,865 INFO mapreduce.ImportJobBase: Beginning import of group_subgroup
2023-05-21 17:15:38,866 INFO Configuration.deprecation: mapred.job.tracker is deprecated. Instead, use mapreduce.jobtracker.address
2023-05-21 17:15:38,866 INFO Configuration.deprecation: mapred.map.tasks is deprecated. Instead, use mapreduce.job.maps
2023-05-21 17:15:39,606 INFO Configuration.deprecation: mapred.map.tasks is deprecated. Instead, use mapreduce.job.maps
2023-05-21 17:15:39,698 INFO client.DefaultNoHDFSFailoverProxyProvider: Connecting to ResourceManager at /0.0.0.0:8083
2023-05-21 17:15:40,301 INFO mapreduce.JobResourceUploader: Disabling Erasure Coding for path: /tmp/hadoop-yarn/staging/sheeba/.staging/job_1684663344794_0012
2023-05-21 17:15:40,766 INFO mapreduce.JobSubmitter: number of splits:1
2023-05-21 17:15:40,866 INFO mapreduce.JobSubmitter: Submitting tokens for job: job_1684663344794_0012

```

```

Administrator: Command Prompt
C:\Windows\system32>sqoop import --connect jdbc:mysql://localhost/_health_care__ --username root --password '' --table hospital_details --hive-import --hive-table healthcare_System.hospital -m 1;
Warning: HBASE_HOME and HBASE_VERSION not set.
Warning: HCAT_HOME not set
Warning: HCATALOG_HOME does not exist HCatalog imports will fail.
Please set HCATALOG_HOME to the root of your HCatalog installation.
Warning: ACCUMULO_HOME not set.
Warning: ZOOKEEPER_HOME not set.
Warning: HBASE_HOME does not exist Hbase imports will fail.
Please set HBASE_HOME to the root of your Hbase installation.
Warning: ACCUMULO_HOME does not exist Accumulo imports will fail.
Please set ACCUMULO_HOME to the root of your Accumulo installation.
Warning: ZOOKEEPER_HOME does not exist Accumulo imports will fail.
Please set ZOOKEEPER_HOME to the root of your Zookeeper installation.
2023-05-21 17:16:37,710 WARN tool.BaseSqoopTool: Setting your password on the command-line is insecure. Consider using -P instead.
2023-05-21 17:16:37,711 INFO tool.BaseSqoopTool: Using Hive-specific delimiters for output. You can override
2023-05-21 17:16:37,711 INFO tool.BaseSqoopTool: delimiters with --fields-terminated-by, etc.
2023-05-21 17:16:37,712 WARN tool.BaseSqoopTool: It seems that you've specified at least one of following:
2023-05-21 17:16:37,717 WARN tool.BaseSqoopTool:   --hive-home
2023-05-21 17:16:37,717 INFO tool.BaseSqoopTool:   --hive-overwrite
2023-05-21 17:16:37,718 WARN tool.BaseSqoopTool:   --create-hive-table
2023-05-21 17:16:37,718 INFO tool.BaseSqoopTool:   --hive-table
2023-05-21 17:16:37,719 WARN tool.BaseSqoopTool:   --hive-partition-key
2023-05-21 17:16:37,720 WARN tool.BaseSqoopTool:   --hive-partition-value
2023-05-21 17:16:37,721 WARN tool.BaseSqoopTool:   --map-column-hive
2023-05-21 17:16:37,721 WARN tool.BaseSqoopTool: Without specifying parameter --hive-import. Please note that
2023-05-21 17:16:37,721 WARN tool.BaseSqoopTool: those arguments will not be used in this session. Either
2023-05-21 17:16:37,722 WARN tool.BaseSqoopTool: specify --hive-import to apply them correctly or remove them
2023-05-21 17:16:37,727 WARN tool.BaseSqoopTool: from command line to remove this warning.
2023-05-21 17:16:37,728 INFO tool.BaseSqoopTool: Please note that --hive-home, --hive-partition-key,
2023-05-21 17:16:37,728 INFO tool.BaseSqoopTool:   --hive-partition-value and --map-column-hive options are
2023-05-21 17:16:37,734 INFO tool.BaseSqoopTool:   are also valid for HCatalog imports and exports
2023-05-21 17:16:37,788 INFO manager.MySQLManager: Preparing to use a MySQL streaming resultset.
2023-05-21 17:16:37,791 INFO tool.CodeGenTool: Beginning code generation
2023-05-21 17:16:38,008 INFO manager.SQLManager: Executing SQL statement: SELECT t.* FROM `hospital_details` AS t LIMIT 1
2023-05-21 17:16:38,848 INFO manager.MySQLManager: Executing SQL statement: SELECT t.* FROM `hospital_details` AS t LIMIT 1
2023-05-21 17:16:39,284 INFO manager.SQLManager: HADOOP_MAPRED_HOME is C:\hadoop\hadoop-3.3.5
Note: 'lmplsqoop-Sheeba\compile\v1\c2ae1a152a29\hospital_details.java' uses or overrides a deprecated API.
Note: Recompile with -Xlint:deprecation for details.
2023-05-21 17:16:39,284 INFO orm.CompilationManager: Writing jar file: /tmp\sqoop-Sheeba\compile\9766f5aa9ec515094cc3f2ae1a152a29\hospital_details.jar
2023-05-21 17:16:39,316 WARN manager.MySQLManager: It looks like you are importing from mysql.
2023-05-21 17:16:39,316 INFO manager.MySQLManager: This transfer can be faster! Use the --direct
2023-05-21 17:16:39,316 INFO manager.MySQLManager: option to exercise a MySQL-specific fast path.
2023-05-21 17:16:39,317 INFO manager.MySQLManager: Setting zero DATETIME behavior to convertToNull (mysql)
2023-05-21 17:16:39,333 INFO mapreduce.ImportJobBase: Beginning import of hospital_details
2023-05-21 17:16:39,333 INFO Configuration.deprecation: mapred.job.tracker is deprecated. Instead, use mapreduce.jobtracker.address
2023-05-21 17:16:39,434 INFO Configuration.deprecation: mapred.map.tasks is deprecated. Instead, use mapreduce.job.maps
2023-05-21 17:16:40,047 INFO Configuration.deprecation: mapred.map.tasks is deprecated. Instead, use mapreduce.job.maps
2023-05-21 17:16:40,133 INFO client.DefaultNoHDFSFailoverProxyProvider: Connecting to ResourceManager at /0.0.0.0:8083
2023-05-21 17:16:40,706 INFO mapreduce.JobResourceUploader: Disabling Erasure Coding for path: /tmp/hadoop-yarn/staging/sheeba/.staging/job_1684663344794_0013
2023-05-21 17:16:41,127 INFO mapreduce.JobSubmitter: number of splits:1
2023-05-21 17:16:41,651 INFO mapreduce.JobSubmitter: Submitting tokens for job: job_1684663344794_0013

```

```

Administrator: Command Prompt
2023-05-21 17:17:07,800 INFO hive.HiveImport: Export directory is not empty, keeping it.

C:\WINDOWS\system32>sqoop import --connect jdbc:mysql://localhost/_health_care__ --username root --password '' --table patient_details --hive-import --hive-table healthcare_System.patient -m 1;
Warning: HBASE_HOME and HBASE_VERSION not set.
Warning: HCATALOG_HOME not set.
Warning: HCATALOG_HOME does not exist HCatalog imports will fail.
Please set HCATALOG_HOME to the root of your HCatalog installation.
Warning: ACCUMULO_HOME not set.
Warning: ZOOKEEPER_HOME not set.
Warning: HBASE_HOME does not exist HBase imports will fail.
Please set HBASE_HOME to the root of your HBase installation.
Warning: ACCUMULO_HOME does not exist Accumulo imports will fail.
Please set ACCUMULO_HOME to the root of your Accumulo installation.
Warning: ZOOKEEPER_HOME does not exist Zookeeper imports will fail.
Please set ZOOKEEPER_HOME to the root of your Zookeeper installation.
2023-05-21 17:17:38,949 WARN tool.BaseSqoopTool: Setting your password on the command-line is insecure. Consider using -P instead.
2023-05-21 17:17:38,949 INFO tool.BaseSqoopTool: Using Hive-specific delimiters for output. You can override
2023-05-21 17:17:38,951 INFO tool.BaseSqoopTool: delimiters with --fields-terminated-by, etc.
2023-05-21 17:17:38,951 WARN tool.BaseSqoopTool: It seems that you've specified at least one of following:
2023-05-21 17:17:38,956 WARN tool.BaseSqoopTool:    --hive-home
2023-05-21 17:17:38,956 WARN tool.BaseSqoopTool:    --hive-overwrite
2023-05-21 17:17:38,957 WARN tool.BaseSqoopTool:    --create-hive-table
2023-05-21 17:17:38,957 WARN tool.BaseSqoopTool:    --hive-table
2023-05-21 17:17:38,958 WARN tool.BaseSqoopTool:    --hive-partition-key
2023-05-21 17:17:38,958 WARN tool.BaseSqoopTool:    --hive-partition-value
2023-05-21 17:17:38,959 WARN tool.BaseSqoopTool:    --map-column-hive
2023-05-21 17:17:38,960 WARN tool.BaseSqoopTool: Without specifying parameter --hive-import. Please note that
2023-05-21 17:17:38,960 WARN tool.BaseSqoopTool: those arguments will not be used in this session. Either
2023-05-21 17:17:38,961 WARN tool.BaseSqoopTool: specify --hive-import to apply them correctly or remove them
2023-05-21 17:17:38,961 WARN tool.BaseSqoopTool: from command line to remove this warning.
2023-05-21 17:17:38,961 INFO tool.BaseSqoopTool: Please note that --hive-home, --hive-partition-key,
2023-05-21 17:17:38,967 INFO tool.BaseSqoopTool:    --hive-partition-value and --map-column-hive options are
2023-05-21 17:17:38,968 INFO tool.BaseSqoopTool:    also valid for HCatalog imports and exports
2023-05-21 17:17:39,024 INFO manager.MySQLManager: Preparing to use a MySQL streaming resultset.
2023-05-21 17:17:39,027 INFO tool.CodeGenTool: Beginning code generation
2023-05-21 17:17:39,242 INFO manager.SqlManager: Executing SQL statement: SELECT t.* FROM `patient_details` AS t LIMIT 1
2023-05-21 17:17:39,278 INFO manager.SqlManager: Executing SQL statement: SELECT t.* FROM `patient_details` AS t LIMIT 1
2023-05-21 17:17:39,281 INFO orm.CompilationManager: HADOOP_MAPRED_HOME is C:\hadoop\hadoop-3.3.5
Note: /tmp/sqoop-Sheeba/compile\333ea08f5a0d82f584aa02ebe453dd84\patient_details.java uses or overrides a deprecated API.
Note: Recompile with -Xlint:deprecation for details.
2023-05-21 17:17:40,547 INFO orm.CompilationManager: Writing jar file: /tmp/sqoop-Sheeba/compile\333ea08f5a0d82f584aa02ebe453dd84\patient_details.jar
2023-05-21 17:17:40,563 WARN manager.MySQLManager: It looks like you are importing from mysql.
2023-05-21 17:17:40,563 INFO manager.MySQLManager: This transfer can be faster. Use the --direct
2023-05-21 17:17:40,564 WARN manager.MySQLManager: option to exercise a MySQL-specific fast path.
2023-05-21 17:17:40,564 INFO manager.MySQLManager: Setting zero DATETIME behavior to convertToNull (mysql)
2023-05-21 17:17:40,580 INFO mapreduce.ImportJobBase: Beginning import of patient_detail.
2023-05-21 17:17:40,582 INFO Configuration.deprecation: mapred.job.tracker is deprecated. Instead, use mapreduce.jobtracker.address
2023-05-21 17:17:40,686 INFO Configuration.deprecation: mapred.jar is deprecated. Instead, use mapreduce.job.jar
2023-05-21 17:17:41,365 INFO Configuration.deprecation: mapred.map.tasks is deprecated. Instead, use mapreduce.job.maps
2023-05-21 17:17:41,440 INFO client.DefaultNoHDFSFailoverProxyProvider: Connecting to ResourceManager at /0.0.0.0:8032
2023-05-21 17:17:42,051 INFO mapreduce.JobResourceUploader: Disabling Erasure Coding for path: /tmp/hadoop-yarn/staging/Sheeba/.staging/job_1684663344794_0014
2023-05-21 17:17:42,488 INFO mapreduce.JobSubmitter: number of splits:1

```

```

Administrator: Command Prompt
2023-05-21 17:18:09,424 INFO hive.HiveImport: Export directory is not empty, keeping it.

C:\WINDOWS\system32>sqoop import --connect jdbc:mysql://localhost/_health_care__ --username root --password '' --table disease --hive-import --hive-table healthcare_System.disease -m 1;
Warning: HBASE_HOME and HBASE_VERSION not set.
Warning: HCATALOG_HOME not set.
Warning: HCATALOG_HOME does not exist HCatalog imports will fail.
Please set HCATALOG_HOME to the root of your HCatalog installation.
Warning: ACCUMULO_HOME not set.
Warning: ZOOKEEPER_HOME not set.
Warning: HBASE_HOME does not exist HBase imports will fail.
Please set HBASE_HOME to the root of your HBase installation.
Warning: ACCUMULO_HOME does not exist Accumulo imports will fail.
Please set ACCUMULO_HOME to the root of your Accumulo installation.
Warning: ZOOKEEPER_HOME does not exist Zookeeper imports will fail.
Please set ZOOKEEPER_HOME to the root of your Zookeeper installation.
2023-05-21 17:20:07,799 WARN tool.BaseSqoopTool: Setting your password on the command-line is insecure. Consider using -P instead.
2023-05-21 17:20:07,800 INFO tool.BaseSqoopTool: Using Hive-specific delimiters for output. You can override
2023-05-21 17:20:07,802 INFO tool.BaseSqoopTool: delimiters with --fields-terminated-by, etc.
2023-05-21 17:20:07,805 WARN tool.BaseSqoopTool: It seems that you've specified at least one of following:
2023-05-21 17:20:07,806 WARN tool.BaseSqoopTool:    --hive-home
2023-05-21 17:20:07,807 WARN tool.BaseSqoopTool:    --hive-overwrite
2023-05-21 17:20:07,807 WARN tool.BaseSqoopTool:    --create-hive-table
2023-05-21 17:20:07,808 WARN tool.BaseSqoopTool:    --hive-table
2023-05-21 17:20:07,809 WARN tool.BaseSqoopTool:    --hive-partition-key
2023-05-21 17:20:07,810 WARN tool.BaseSqoopTool:    --hive-partition-value
2023-05-21 17:20:07,810 WARN tool.BaseSqoopTool:    --map-column-hive
2023-05-21 17:20:07,815 WARN tool.BaseSqoopTool: Without specifying parameter --hive-import. Please note that
2023-05-21 17:20:07,816 WARN tool.BaseSqoopTool: those arguments will not be used in this session. Either
2023-05-21 17:20:07,817 WARN tool.BaseSqoopTool: specify --hive-import to apply them correctly or remove them
2023-05-21 17:20:07,817 INFO tool.BaseSqoopTool: from command line to remove this warning.
2023-05-21 17:20:07,818 INFO tool.BaseSqoopTool: Please note that --hive-home, --hive-partition-key,
2023-05-21 17:20:07,828 INFO tool.BaseSqoopTool:    --hive-partition-value and --map-column-hive options are
2023-05-21 17:20:07,828 INFO tool.BaseSqoopTool:    also valid for HCatalog imports and exports
2023-05-21 17:20:07,831 INFO manager.MySQLManager: Preparing to use a MySQL streaming resultset.
2023-05-21 17:20:07,831 INFO tool.CodeGenTool: Beginning code generation
2023-05-21 17:20:09,181 INFO manager.SqlManager: Executing SQL statement: SELECT t.* FROM `disease` AS t LIMIT 1
2023-05-21 17:20:09,181 INFO manager.SqlManager: Executing SQL statement: SELECT t.* FROM `disease` AS t LIMIT 1
2023-05-21 17:20:08,187 INFO orm.CompilationManager: HADOOP_MAPRED_HOME is C:\hadoop\hadoop-3.3.5
Note: /tmp/sqoop-Sheeba/compile\842cf92a19a84b19c77334bdd7e15b\disease.java uses or overrides a deprecated API.
Note: Recompile with -Xlint:deprecation for details.
2023-05-21 17:20:09,474 INFO orm.CompilationManager: Writing jar file: /tmp/sqoop-Sheeba/compile\842cf92a19a84b19c77334bdd7e15b\disease.jar
2023-05-21 17:20:09,498 WARN manager.MySQLManager: It looks like you are importing from mysql.
2023-05-21 17:20:09,498 INFO manager.MySQLManager: This transfer can be faster. Use the --direct
2023-05-21 17:20:09,498 WARN manager.MySQLManager: option to exercise a MySQL-specific fast path.
2023-05-21 17:20:09,498 INFO manager.MySQLManager: Setting zero DATETIME behavior to convertToNull (mysql)
2023-05-21 17:20:09,515 INFO mapreduce.ImportJobBase: Beginning import of disease
2023-05-21 17:20:09,517 INFO Configuration.deprecation: mapred.job.tracker is deprecated. Instead, use mapreduce.jobtracker.address
2023-05-21 17:20:09,619 INFO Configuration.deprecation: mapred.jar is deprecated. Instead, use mapreduce.job.jar
2023-05-21 17:20:10,255 INFO Configuration.deprecation: mapred.map.tasks is deprecated. Instead, use mapreduce.job.maps
2023-05-21 17:20:10,341 INFO client.DefaultNoHDFSFailoverProxyProvider: Connecting to ResourceManager at /0.0.0.0:8032
2023-05-21 17:20:10,475 INFO mapreduce.JobResourceUploader: Disabling Erasure Coding for path: /tmp/hadoop-yarn/staging/Sheeba/.staging/job_1684663344794_0015
2023-05-21 17:20:11,499 INFO mapreduce.JobSubmitter: number of splits:1

```

```

Administrator: Command Prompt

C:\WINDOWS\system32>sqoop import --connect jdbc:mysql://localhost/_health_care__ --username root --password '' --table subscriber --hive-import --hive-table healthcare_System.subscriber -m 1;
Warning: HBASE_HOME and HBASE_VERSION not set.
Warning: HCAT_HOME not set
Warning: HCATALOG_HOME does not exist HCatalog imports will fail.
Please set HCATALOG_HOME to the root of your HCatalog installation.
Warning: ACCUMULO_HOME not set.
Warning: ZOOKEEPER_HOME not set.
Warning: HBASE_HOME does not exist HBase imports will fail.
Please set HBASE_HOME to the root of your HBase installation.
Warning: ACCUMULO_HOME does not exist Accumulo imports will fail.
Please set ACCUMULO_HOME to the root of your Accumulo installation.
Warning: ZOOKEEPER_HOME does not exist Accumulo imports will fail.
Please set ZOOKEEPER_HOME to the root of your Zookeeper installation.
2023-05-21 17:22:23,125 WARN tool.BaseSqoopTool: Setting your password on the command-line is insecure. Consider using -P instead.
2023-05-21 17:22:23,125 INFO tool.BaseSqoopTool: Using Hive-specific delimiters for output. You can override
2023-05-21 17:22:23,127 INFO tool.BaseSqoopTool: delimiters with --fields-terminated-by, etc.
2023-05-21 17:22:23,130 WARN tool.BaseSqoopTool: It seems that you've specified at least one of following:
2023-05-21 17:22:23,132 WARN tool.BaseSqoopTool:   --hive-home
2023-05-21 17:22:23,132 WARN tool.BaseSqoopTool:   --hive-overwrite
2023-05-21 17:22:23,132 WARN tool.BaseSqoopTool:   --create-hive-table
2023-05-21 17:22:23,133 WARN tool.BaseSqoopTool:   --hive-table
2023-05-21 17:22:23,133 WARN tool.BaseSqoopTool:   --hive-partition-key
2023-05-21 17:22:23,134 WARN tool.BaseSqoopTool:   --hive-partition-value
2023-05-21 17:22:23,139 WARN tool.BaseSqoopTool: Without specifying parameter --hive-import. Please note that
2023-05-21 17:22:23,140 WARN tool.BaseSqoopTool: those arguments will not be used in this session. Either
2023-05-21 17:22:23,140 WARN tool.BaseSqoopTool: specify --hive-import to apply them correctly or remove them
2023-05-21 17:22:23,142 INFO tool.BaseSqoopTool: from command line to remove this warning.
2023-05-21 17:22:23,142 INFO tool.BaseSqoopTool: Please note that --hive-home, --hive-partition-key,
2023-05-21 17:22:23,142 INFO tool.BaseSqoopTool:   --hive-partition-value and --map-column-hive options are
2023-05-21 17:22:23,142 INFO tool.BaseSqoopTool: also valid for HCatalog imports and exports
2023-05-21 17:22:23,224 INFO tool.CodeGenTool: Beginning code generation
2023-05-21 17:22:23,508 INFO manager.SqlManager: Executing SQL statement: SELECT t.* FROM `subscriber` AS t LIMIT 1
2023-05-21 17:22:23,516 INFO manager.SqlManager: Executing SQL statement: SELECT t.* FROM `subscriber` AS t LIMIT 1
2023-05-21 17:22:23,520 INFO orm.CompilationManager: HADOOP_MAPRED_HOME is C:\hadoop\hadoop-3.3.5
Note: \tmp\sqoop-Sheeba\compile49a8d8957b1fcf72e1a408e1871b1d0a\subscriber.java uses or overrides a deprecated API.
Note: Recompiling with -Xlint:deprecation for details.
2023-05-21 17:22:24,872 INFO orm.CompilationManager: Writing jar file: \tmp\sqoop-Sheeba\compile49a8d8957b1fcf72e1a408e1871b1d0a\subscriber.jar
2023-05-21 17:22:24,938 WARN manager.SqlManager: It looks like you are importing from mysql.
2023-05-21 17:22:24,938 WARN manager.SqlManager: This transfer can be faster! Use the --direct
2023-05-21 17:22:24,941 WARN manager.SqlManager: option to exercise a MySQL-specific fast path.
2023-05-21 17:22:24,942 INFO manager.SqlManager: Setting zero DATETIME behavior to convertToNull (mysql)
2023-05-21 17:22:24,956 INFO mapreduce.ImportJobBase: Beginning import of subscriber
2023-05-21 17:22:25,070 INFO Configuration.deprecation: mapred.job.tracker is deprecated. Instead, use mapreduce.job.tracker
2023-05-21 17:22:25,070 INFO Configuration.deprecation: mapred.jar is deprecated. Instead, use mapreduce.job.jar
2023-05-21 17:22:25,707 INFO Configuration.deprecation: mapred.map.tasks is deprecated. Instead, use mapreduce.job.maps
2023-05-21 17:22:25,793 INFO client.DefaultNoHARMFailoverProxyProvider: Connecting to ResourceManager at /0.0.0.0:8032
2023-05-21 17:22:26,396 INFO mapreduce.JobResourceUploader: Disabling Erasure Coding for path: /tmp/hadoop-yarn/staging/Sheeba/.staging/job_1684663344794_0016
2023-05-21 17:22:26,856 INFO mapreduce.JobSubmitter: number of splits:1
2023-05-21 17:22:26,953 INFO mapreduce.JobSubmitter: Submitting tokens for job: job_1684663344794_0016

```

```

Administrator: Command Prompt

C:\WINDOWS\system32>sqoop import --connect jdbc:mysql://localhost/_health_care__ --username root --password '' --table claims --hive-import --hive-table healthcare_System.claims -m 1;
Warning: HBASE_HOME and HBASE_VERSION not set.
Warning: HCAT_HOME not set
Warning: HCATALOG_HOME does not exist HCatalog imports will fail.
Please set HCATALOG_HOME to the root of your HCatalog installation.
Warning: ACCUMULO_HOME not set.
Warning: ZOOKEEPER_HOME not set.
Warning: HBASE_HOME does not exist HBase imports will fail.
Please set HBASE_HOME to the root of your HBase installation.
Warning: ACCUMULO_HOME does not exist Accumulo imports will fail.
Please set ACCUMULO_HOME to the root of your Accumulo installation.
Warning: ZOOKEEPER_HOME does not exist Accumulo imports will fail.
Please set ZOOKEEPER_HOME to the root of your Zookeeper installation.
2023-05-21 17:23:25,567 WARN tool.BaseSqoopTool: Setting your password on the command-line is insecure. Consider using -P instead.
2023-05-21 17:23:25,568 INFO tool.BaseSqoopTool: Using Hive-specific delimiters for output. You can override
2023-05-21 17:23:25,572 WARN tool.BaseSqoopTool: It seems that you've specified at least one of following:
2023-05-21 17:23:25,574 WARN tool.BaseSqoopTool:   --hive-home
2023-05-21 17:23:25,574 WARN tool.BaseSqoopTool:   --hive-overwrite
2023-05-21 17:23:25,574 WARN tool.BaseSqoopTool:   --create-hive-table
2023-05-21 17:23:25,575 WARN tool.BaseSqoopTool:   --hive-table
2023-05-21 17:23:25,575 WARN tool.BaseSqoopTool:   --hive-partition-key
2023-05-21 17:23:25,580 WARN tool.BaseSqoopTool:   --hive-partition-value
2023-05-21 17:23:25,581 WARN tool.BaseSqoopTool:   --map-column-hive
2023-05-21 17:23:25,582 WARN tool.BaseSqoopTool: Without specifying parameter --hive-import. Please note that
2023-05-21 17:23:25,582 WARN tool.BaseSqoopTool: those arguments will not be used in this session. Either
2023-05-21 17:23:25,583 WARN tool.BaseSqoopTool: specify --hive-import to apply them correctly or remove them
2023-05-21 17:23:25,583 WARN tool.BaseSqoopTool: from command line to remove this warning.
2023-05-21 17:23:25,587 INFO tool.BaseSqoopTool: Please note that --hive-home, --hive-partition-key,
2023-05-21 17:23:25,588 INFO tool.BaseSqoopTool:   --hive-partition-value and --map-column-hive options are
2023-05-21 17:23:25,593 INFO tool.BaseSqoopTool: also valid for HCatalog imports and exports
2023-05-21 17:23:25,649 INFO manager.SqlManager: Preparing to use a MySQL streaming resultset.
2023-05-21 17:23:25,653 INFO tool.CodeGenTool: Beginning code generation
2023-05-21 17:23:25,883 INFO manager.SqlManager: Executing SQL statement: SELECT t.* FROM `claims` AS t LIMIT 1
2023-05-21 17:23:25,916 INFO manager.SqlManager: Executing SQL statement: SELECT t.* FROM `claims` AS t LIMIT 1
2023-05-21 17:23:25,926 INFO orm.CompilationManager: HADOOP_MAPRED_HOME is C:\hadoop\hadoop-3.3.5
Note: \tmp\sqoop-Sheeba\compile12cfa62529fd4a6feedb5c8cab60c98\claims.java uses or overrides a deprecated API.
Note: Recompiling with -Xlint:deprecation for details.
2023-05-21 17:23:27,265 INFO orm.CompilationManager: Writing jar file: \tmp\sqoop-Sheeba\compile12cfa62529fd4a6feedb5c8cab60c98\claims.jar
2023-05-21 17:23:27,295 WARN manager.SqlManager: It looks like you are importing from mysql.
2023-05-21 17:23:27,295 WARN manager.SqlManager: This transfer can be faster! Use the --direct
2023-05-21 17:23:27,296 WARN manager.SqlManager: option to exercise a MySQL-specific fast path.
2023-05-21 17:23:27,296 INFO manager.SqlManager: Setting zero DATETIME behavior to convertToNull (mysql)
2023-05-21 17:23:27,320 INFO mapreduce.ImportJobBase: Beginning import of claims
2023-05-21 17:23:27,321 INFO Configuration.deprecation: mapred.job.tracker is deprecated. Instead, use mapreduce.jobtracker.address
2023-05-21 17:23:27,430 INFO Configuration.deprecation: mapred.jar is deprecated. Instead, use mapreduce.job.jar
2023-05-21 17:23:28,058 INFO Configuration.deprecation: mapred.map.tasks is deprecated. Instead, use mapreduce.job.maps
2023-05-21 17:23:28,739 INFO client.DefaultNoHARMFailoverProxyProvider: Connecting to ResourceManager at /0.0.0.0:8032
2023-05-21 17:23:29,177 INFO mapreduce.JobResourceUploader: Disabling Erasure Coding for path: /tmp/hadoop-yarn/staging/Sheeba/.staging/job_1684663344794_0017
2023-05-21 17:23:29,276 INFO mapreduce.JobSubmitter: number of splits:1
2023-05-21 17:23:29,276 INFO mapreduce.JobSubmitter: Submitting tokens for job: job_1684663344794_0017

```

Hive import complete: Successfully Imported tables from mysql to hive using sqoop

```
2023-05-21 17:23:55,100 INFO hive.HiveImport: OK
2023-05-21 17:23:55,100 INFO hive.HiveImport: 2023-05-21 17:23:55,103 INFO ql.Driver: OK
2023-05-21 17:23:55,108 INFO hive.HiveImport: 2023-05-21 17:23:55,103 INFO ql.Driver: Concurrency mode is disabled, not creating a lock manager
2023-05-21 17:23:55,111 INFO hive.HiveImport: Time taken: 0.5 seconds
2023-05-21 17:23:55,111 INFO hive.HiveImport: 2023-05-21 17:23:55,103 INFO CliDriver: Time taken: 0.5 seconds
2023-05-21 17:23:55,111 INFO hive.HiveImport: 2023-05-21 17:23:55,104 INFO conf.HiveConf: Using the default value passed in for log id: ee540734-11a5-48c1-9e96-10605a91eb47
2023-05-21 17:23:55,111 INFO hive.HiveImport: 2023-05-21 17:23:55,104 INFO session.SessionState: Resetting thread name to _main
2023-05-21 17:23:55,111 INFO hive.HiveImport: 2023-05-21 17:23:55,103 INFO conf.HiveConf: Using the default value passed in for log id: ee540734-11a5-48c1-9e96-10605a91eb47
2023-05-21 17:23:55,121 INFO hive.HiveImport: 2023-05-21 17:23:55,121 INFO session.SessionState: Deleted directory: /tmp/hive/Sheeba/e540734-11a5-48c1-9e96-10605a91eb47 on fs with scheme hdfs
2023-05-21 17:23:55,129 INFO hive.HiveImport: 2023-05-21 17:23:55,128 INFO session.SessionState: Deleted directory: C:/Users/Sheeba/AppData/Local/Tmp/Sheeba/e540734-11a5-48c1-9e96-10605a91eb47 on fs with scheme hdfs
2023-05-21 17:23:55,138 INFO hive.HiveImport: 2023-05-21 17:23:55,138 INFO metastore.HiveMetaStore: 0: Cleaning up thread local RawStore...
2023-05-21 17:23:55,138 INFO hive.HiveImport: 2023-05-21 17:23:55,138 INFO HiveMetaStore.audit: ugi=Sheeba ip=unknown-ip-addr cmd=cleaning up thread local RawStore...
2023-05-21 17:23:55,138 INFO hive.HiveImport: 2023-05-21 17:23:55,138 INFO metastore.HiveMetaStore: 0: Done cleaning up thread local RawStore
2023-05-21 17:23:55,146 INFO hive.HiveImport: 2023-05-21 17:23:55,138 INFO HiveMetaStore.audit: ugi=Sheeba ip=unknown-ip-addr cmd=Done cleaning up thread local RawStore
2023-05-21 17:23:55,214 INFO hive.HiveImport: Hive import complete.
2023-05-21 17:23:55,223 INFO hive.HiveImport: Export directory is not empty, keeping it.
C:\WINDOWS\system32>
```

5.5 DATA STORAGE, PROCESSING AND ANALYTICS SERVICES:

Showing imported tables in hive

```
hive> show tables;
2023-05-21 17:24:08,798 INFO conf.HiveConf: Using the default value passed in for log id: fe20bdal-1ed1-49eb-b699-a55e04cafe74
2023-05-21 17:24:08,800 INFO session.SessionState: Updating thread name to fe20bdal-1ed1-49eb-b699-a55e04cafe74 main
2023-05-21 17:24:08,809 INFO ql.Driver: Compiling command(queryId=Sheeba_20230521172408_a518f4ef-ad4c-419f-b1ce-6d8c56b7a0d9): show tables
2023-05-21 17:24:08,837 INFO ql.Driver: Concurrency mode is disabled, not creating a lock manager
2023-05-21 17:24:08,837 INFO metastore.HiveMetaStore: 0: get_database: @hive#healthcare_system
2023-05-21 17:24:08,838 INFO HiveMetaStore.audit: ugi=Sheeba ip=unknown-ip-addr cmd=get_database: @hive#healthcare_system
2023-05-21 17:24:08,844 INFO ql.Driver: Semantic Analysis Completed (retryial = false)
2023-05-21 17:24:08,845 INFO ql.Driver: Returning Hive schema: Schema[fieldSchemas:[FieldSchema(name:tab_name, type:string, comment:from deserializer)], properties:null]
2023-05-21 17:24:08,845 INFO exec.ListSinkOperator: Initializing operator LIST_SINK[0]
2023-05-21 17:24:08,846 INFO ql.Driver: Completed compiling command(queryId=Sheeba_20230521172408_a518f4ef-ad4c-419f-b1ce-6d8c56b7a0d9); Time taken: 0.036 seconds
2023-05-21 17:24:08,846 INFO reexec.ReExecDriver: Execution #1 of query
2023-05-21 17:24:08,846 INFO ql.Driver: Concurrency mode is disabled, not creating a lock manager
2023-05-21 17:24:08,846 INFO ql.Driver: Executing command(queryId=Sheeba_20230521172408_a518f4ef-ad4c-419f-b1ce-6d8c56b7a0d9): show tables
2023-05-21 17:24:08,847 INFO ql.Driver: Starting task [Stage-0:DDL] in serial mode
2023-05-21 17:24:08,847 INFO metastore.HiveMetaStore: 0: get_database: @hive#healthcare_system
2023-05-21 17:24:08,847 INFO HiveMetaStore.audit: ugi=Sheeba ip=unknown-ip-addr cmd=get_database: @hive#healthcare_system
2023-05-21 17:24:08,849 INFO metastore.HiveMetaStore: 0: get_tables: db=@hive#healthcare_system pat=.*
2023-05-21 17:24:08,850 INFO HiveMetaStore.audit: ugi=Sheeba ip=unknown-ip-addr cmd=get_tables: db=@hive#healthcare_system pat=.*
2023-05-21 17:24:08,854 INFO ql.Driver: Completed executing command(queryId=Sheeba_20230521172408_a518f4ef-ad4c-419f-b1ce-6d8c56b7a0d9); Time taken: 0.008 seconds
OK
2023-05-21 17:24:08,855 INFO ql.Driver: OK
2023-05-21 17:24:08,857 INFO ql.Driver: Concurrency mode is disabled, not creating a lock manager
2023-05-21 17:24:08,859 INFO mapred.FileInputFormat: Total input files to process : 1
2023-05-21 17:24:08,875 INFO exec.ListSinkOperator: RECORDS_OUT_INTERMEDIATE:0, RECORDS_OUT_OPERATOR_LIST_SINK_0:8,
claims
disease
groups
grp_subgrp
hospital
patient
subgroup
subscriber
Time taken: 0.049 seconds, Fetched: 8 row(s)
2023-05-21 17:24:08,885 INFO CliDriver: Time taken: 0.049 seconds, Fetched: 8 row(s)
2023-05-21 17:24:08,897 INFO conf.HiveConf: Using the default value passed in for log id: fe20bdal-1ed1-49eb-b699-a55e04cafe74
2023-05-21 17:24:08,902 INFO session.SessionState: Resetting thread name to _main
hive>
```

Hive In web ui

Name	Size	Last Modified	Block Size
claims	0 B	May 21 17:23	0 B
disease	0 B	May 21 17:20	0 B
groups	0 B	May 21 17:08	0 B
grp_subgrp	0 B	May 21 17:16	0 B
hospital	0 B	May 21 17:17	0 B
patient	0 B	May 21 17:18	0 B
subgroup	0 B	May 21 17:14	0 B
subscriber	0 B	May 21 17:22	0 B

Queries in hive

```

hive> select * from hospital;
2023-05-21 17:56:58.439 INFO conf.HiveConf: Using the default value passed in for log id: fe20bdal-1ed1-49eb-b699-a55e04cafe74
2023-05-21 17:56:58.441 INFO session.SessionState: Updating thread name to fe20bdal-1ed1-49eb-b699-a55e04cafe74 main
2023-05-21 17:56:58.470 INFO ql.Driver: Compiling command(queryId=sheeba_20230521175658_cda3112f-91b9-4dc0-a2c1-4ab36b8718e4): select * from hospital
2023-05-21 17:56:58.520 INFO ql.Driver: Concurrency mode is disabled, not creating a lock manager
2023-05-21 17:56:58.521 INFO ql.Driver: Starting Semantic Analysis
2023-05-21 17:56:58.527 INFO parse.CalcitePlanner: Completed phase 1 of Semantic Analysis
2023-05-21 17:56:58.528 INFO parse.CalcitePlanner: Get metadata for source tables
2023-05-21 17:56:58.529 INFO metastore.HiveMetaStore: 0: get_table : tbl=hive.healthcare_System.hospital
2023-05-21 17:56:58.531 INFO metastore.HiveMetaStore: audit: ugisSheeba ip=unknown-ip-addr cmd=get_table :tbl=hive.healthcare_System.hospital
2023-05-21 17:56:58.606 INFO parse.CalcitePlanner: Get metadata for subqueries
2023-05-21 17:56:58.607 INFO parse.CalcitePlanner: Get metadata for destination tables
2023-05-21 17:56:58.639 INFO ql.Context: New scratch dir is hdfs://localhost:9000/tmp/hive/Sheeba/fe20bdal-1ed1-49eb-b699-a55e04cafe74/hive_2023-05-21_17-56-58_510_2741165843716222131-1
2023-05-21 17:56:58.640 INFO parse.CalcitePlanner: Completed getting MetaData in Semantic Analysis
2023-05-21 17:56:58.666 INFO metastore.HiveMetaStore: 0: get_not_null_constraints :tbl=hive.healthcare_system.hospital
2023-05-21 17:56:58.670 INFO metastore.HiveMetaStore: 0: get_primary_keys :tbl=hive.healthcare_system.hospital
2023-05-21 17:56:58.672 INFO metastore.HiveMetaStore: 0: get_primary_keys :tbl=hive.healthcare_system.hospital
2023-05-21 17:56:58.674 INFO metastore.HiveMetaStore: 0: get_primary_keys :tbl=hive.healthcare_system.hospital
2023-05-21 17:56:58.676 INFO metastore.HiveMetaStore: 0: get_primary_keys :tbl=hive.healthcare_system.hospital
2023-05-21 17:56:58.678 INFO metastore.HiveMetaStore: 0: get_primary_keys :tbl=hive.healthcare_system.hospital
2023-05-21 17:56:58.681 INFO metastore.HiveMetaStore: 0: get_foreign_keys :parentdb=null parenttbl=null foreigndb=healthcare_system foreigntbl=hospital
2023-05-21 17:56:58.686 INFO metastore.HiveMetaStore: 0: get_foreign_keys :parentdb=null parenttbl=null foreigndb=healthcare_system foreigntbl=hospital
2023-05-21 17:56:58.690 INFO metastore.HiveMetaStore: 0: get_databases :@hive#
2023-05-21 17:56:58.700 INFO HiveMetaStore.audit: ugisSheeba ip=unknown-ip-addr cmd=get_databases:@hive#
2023-05-21 17:56:58.706 INFO metastore.HiveMetaStore: 0: get_materialized_views_for_rewriting: db=@hive#abc
2023-05-21 17:56:58.706 INFO HiveMetaStore.audit: ugisSheeba ip=unknown-ip-addr cmd=get_materialized_views_for_rewriting: db=@hive#abc
2023-05-21 17:56:58.707 INFO metastore.HiveMetaStore: 0: get_materialized_views_for_rewriting: db=@hive#default
2023-05-21 17:56:58.709 INFO HiveMetaStore.audit: ugisSheeba ip=unknown-ip-addr cmd=get_materialized_views_for_rewriting: db=@hive#default
2023-05-21 17:56:58.713 INFO metastore.HiveMetaStore: 0: get_auto_indexed_views_for_rewriting: db=@hiveHealthcare_system
2023-05-21 17:56:58.777 INFO parse.CalcitePlanner: Get metadata for source tables
2023-05-21 17:56:58.777 INFO metastore.HiveMetaStore: 0: get_table : thl=hive.healthcare_system.hospital
2023-05-21 17:56:58.778 INFO HiveMetaStore.audit: ugisSheeba ip=unknown-ip-addr cmd=get_table :tbl=hive.healthcare_system.hospital
2023-05-21 17:56:58.802 INFO parse.CalcitePlanner: Get metadata for subqueries
2023-05-21 17:56:58.802 INFO parse.CalcitePlanner: Get metadata for destination tables
2023-05-21 17:56:58.806 INFO ql.Context: New scratch dir is hdfs://localhost:9000/tmp/hive/Sheeba/fe20bdal-1ed1-49eb-b699-a55e04cafe74/hive_2023-05-21_17-56-58_510_2741165843716222131-1
2023-05-21 17:56:58.810 INFO common.FileUtils: Creating directory if it doesn't exist: hdfs://localhost:9000/tmp/hive/Sheeba/fe20bdal-1ed1-49eb-b699-a55e04cafe74/hive_2023-05-21_17-56-58_510_2741165843716222131-1/_mr_10001/hive-staging_hive_2023-05-21_17-56-58_510_2741165843716222131-1
2023-05-21 17:56:58.811 INFO pdp.OpProcessor: Processing for TSQL
2023-05-21 17:56:58.821 INFO pdp.OpProcFactory: Processing for SEL(1)
2023-05-21 17:56:58.821 INFO pdp.OpProcFactory: Processing for TS(0)
2023-05-21 17:56:58.828 INFO parse.CalcitePlanner: Completed plan generation
2023-05-21 17:56:58.828 INFO parse.CalcitePlanner: Not eligible for results caching - no mr/tez/spark jobs
2023-05-21 17:56:58.829 INFO ql.Driver: Semantic Analysis Completed (retry= false)
2023-05-21 17:56:58.829 INFO ql.Driver: Returning Hive schema: Schema{FieldSchemas:[FieldSchema{name:hospital.hospital_id, type:string, comment:null}, FieldSchema{name:hospital.hospital_name, type:string, comment:null}, FieldSchema{name:hospital.city, type:string, comment:null}, FieldSchema{name:hospital.state, type:string, comment:null}, FieldSchema{name:hospital.country, type:string, comment:null}], properties:null}

```

```

OK
2023-05-21 17:56:58,880 INFO ql.Driver: OK
2023-05-21 17:56:58,880 INFO ql.Driver: Concurrency mode is disabled, not creating a lock manager
2023-05-21 17:56:58,900 INFO mapred.FileInputFormat: Total input files to process : 1
2023-05-21 17:56:58,916 INFO lazy.LazyStruct: Missing fields! Expected 5 fields but only got 1! Ignoring similar problems.
2023-05-21 17:56:58,918 INFO exec.TableScanOperator: RECORDS_OUT_INTERMEDIATE:0, RECORDS_OUT_OPERATOR_TS_0:A1,
2023-05-21 17:56:58,918 INFO exec.SelectOperator: RECORDS_OUT_INTERMEDIATE:0, RECORDS_OUT_OPERATOR_SEL_1:A1,
2023-05-21 17:56:58,919 INFO exec.ListSinkOperator: RECORDS_OUT_INTERMEDIATE:0, RECORDS_OUT_OPERATOR_LIST_SINK_3:A1,
H1000 All India Institute of Medical Sciences New Delhi      NaN      India
NULL NULL NULL NULL
H1001 Medanta The Medicity Gurgaon Haryana India
NULL NULL NULL NULL
H1002 The Christian Medical College Vellore Tamil Nadu India
NULL NULL NULL NULL
H1003 PGIMER - Postgraduate Institute of Medical Education and Research Chandigarh Haryana India
NULL NULL NULL NULL
H1004 Apollo Hospital - Chennai Chennai Tamil Nadu India
NULL NULL NULL NULL
H1005 P. D. Hindujia National Hospital & Medical Research Centre Mumbai Maharashtra India
NULL NULL NULL NULL
H1006 Breach Candy Hospital Mumbai Maharashtra India
NULL NULL NULL NULL
H1007 Fortis Ft. Lt. Rajan Dhall Hospital New Delhi NaN India
NULL NULL NULL NULL
H1008 King Edward Memorial Hospital Mumbai Maharashtra India
NULL NULL NULL NULL
H1009 Indraprastha Apollo Hospital Delhi NaN India
NULL NULL NULL NULL
H1010 Lilavati Hospital And Research Centre Mumbai Maharashtra India
NULL NULL NULL NULL
H1011 Sir Ganga Ram Hospital Delhi NaN India
NULL NULL NULL NULL
H1012 Bombay Hospital & Medical Research Centre Mumbai Maharashtra India
NULL NULL NULL NULL
H1013 Apollo Health City - Jubilee Hills Hyderabad Telangana India
NULL NULL NULL NULL
H1014 Fortis Hiranandani Hospital Mumbai Maharashtra India
NULL NULL NULL NULL
H1015 Fortis Hospital Mulund Mumbai Maharashtra India
NULL NULL NULL NULL
H1016 Jaslok Hospital and Research Centre Mumbai Maharashtra India
NULL NULL NULL NULL
H1017 Manipal Hospitals Bengaluru Karnataka India
NULL NULL NULL NULL
H1018 Yashoda Hospital Secunderabad Hyderabad Telangana India
NULL NULL NULL NULL
H1019 Apollo Hospitals - Bannerghatta Road Bengaluru Karnataka India
NULL NULL NULL NULL
Hospit Hospital_name city state countr
Time taken: 0.427 seconds, Fetched: 41 row(s)
2023-05-21 17:56:58,991 INFO CliDriver: Time taken: 0.427 seconds, Fetched: 41 row(s)
2023-05-21 17:56:58,991 INFO conf.HiveConf: Using the default value passed in for log id: fe20bd1-1ed1-49eb-b699-a55e04cafe74

```

```

hive> select * from hospital where hospital_id='H1013'
2023-05-21 17:59:31,773 INFO conf.HiveConf: Using the default value passed in for log id: fe20bd1-1ed1-49eb-b699-a55e04cafe74
2023-05-21 17:59:31,777 INFO session.SessionState: Updating thread name to fe20bd1-1ed1-49eb-b699-a55e04cafe74 main
2023-05-21 17:59:31,783 INFO ql.Driver: Compiling command(queryId=sheeba_20230521175931_40f8255f-6901-4304-82f8-9163f5172aa8): select * from hospital where hospital_id="H1013"
2023-05-21 17:59:31,800 INFO ql.Driver: Concurrency mode is disabled, not creating a lock manager
2023-05-21 17:59:31,808 INFO parse.CalcitePlanner: Starting Semantic Analysis
2023-05-21 17:59:31,803 INFO parse.CalcitePlanner: Get metadata for source tables
2023-05-21 17:59:31,804 INFO metastore.HiveMetaStore: 0: get_table : tbl:hive.healthcare_System.hospital
2023-05-21 17:59:31,820 INFO parse.CalcitePlanner: Get metadata for destination tables
2023-05-21 17:59:31,829 INFO ql.Context: New scratch dir is hdfs://localhost:9000/tmp/hive/Sheeba/fe20bd1-1ed1-49eb-b699-a55e04cafe74/hive_2023-05-21_17-59-31_800_6168806748029619938-1
2023-05-21 17:59:31,839 INFO parse.CalcitePlanner: Completed getting Metadata in Semantic Analysis
2023-05-21 17:59:31,841 INFO metastore.HiveMetaStore: audit: ugi:sheeba ipunknown:ip-addr cmd:get_not_null_constraints :tbl:hive.healthcare_system.hospital
2023-05-21 17:59:31,842 INFO metastore.HiveMetaStore: 0: get_primary_keys :tbl:hive.healthcare_system.hospital
2023-05-21 17:59:31,843 INFO metastore.HiveMetaStore: audit: ugi:sheeba ipunknown:ip-addr cmd:get_primary_keys :tbl:hive.healthcare_system.hospital
2023-05-21 17:59:31,845 INFO metastore.HiveMetaStore: 0: get_primary_keys :tbl:hive.healthcare_system.hospital
2023-05-21 17:59:31,850 INFO metastore.HiveMetaStore: audit: ugi:sheeba ipunknown:ip-addr cmd:get_primary_keys :tbl:hive.healthcare_system.hospital
2023-05-21 17:59:31,859 INFO metastore.HiveMetaStore: audit: ugi:sheeba ipunknown:ip-addr cmd:get_uniq_constraints :tbl:hive.healthcare_system.hospital
2023-05-21 17:59:31,862 INFO metastore.HiveMetaStore: 0: get_foreign_keys :parentdb=null parenttbl=null foreigndb=healthcare_system foreigntbl=foreigntbl=hospital
2023-05-21 17:59:31,863 INFO metastore.HiveMetaStore: audit: ugi:sheeba ipunknown:ip-addr cmd:get_foreign_keys :parentdb=null parenttbl=null foreigndb=healthcare_system foreigntbl=hospital
2023-05-21 17:59:32,036 INFO metastore.HiveMetaStore: 0: get_databases: @hive#
2023-05-21 17:59:32,038 INFO HiveMetaStore.audit: ugi:sheeba ipunknown:ip-addr cmd:get_table :tbl:hive.healthcare_system.hospital
2023-05-21 17:59:32,032 INFO metastore.HiveMetaStore: 0: get_materialized_views_for_rewriting: db@hiveabc
2023-05-21 17:59:32,033 INFO HiveMetaStore.audit: ugi:sheeba ipunknown:ip-addr cmd:get_materialized_views_for_rewriting: db@hiveabc
2023-05-21 17:59:32,034 INFO metastore.HiveMetaStore: 0: get_materialized_views_for_rewriting: db@hivedefault
2023-05-21 17:59:32,036 INFO metastore.HiveMetaStore: 0: get_materialized_views_for_rewriting: db@hivedefault
2023-05-21 17:59:32,039 INFO metastore.HiveMetaStore: 0: get_materialized_views_for_rewriting: db@hivehealthcare_system
2023-05-21 17:59:32,041 INFO HiveMetaStore.audit: ugi:sheeba ipunknown:ip-addr cmd:get_materialized_views_for_rewriting: db@hiveHealthcare_system
2023-05-21 17:59:32,049 INFO parse.CalcitePlanner: Get metadata for source tables
2023-05-21 17:59:32,050 INFO metastore.HiveMetaStore: 0: get_table :tbl:hive.healthcare_System.hospital
2023-05-21 17:59:32,058 INFO HiveMetaStore.audit: ugi:sheeba ipunknown:ip-addr cmd:get_table :tbl:hive.healthcare_System.hospital
2023-05-21 17:59:32,075 INFO parse.CalcitePlanner: Get metadata for subqueries
2023-05-21 17:59:32,075 INFO parse.CalcitePlanner: Get metadata for destination tables
2023-05-21 17:59:32,077 INFO ql.Context: New scratch dir is hdfs://localhost:9000/tmp/hive/Sheeba/fe20bd1-1ed1-49eb-b699-a55e04cafe74/hive_2023-05-21_17-59-31_800_6168806748029619938-1
2023-05-21 17:59:32,077 INFO parse.CalcitePlanner: Creating directory if it doesn't exist: hdfs://localhost:9000/tmp/hive/Sheeba/fe20bd1-1ed1-49eb-b699-a55e04cafe74/hive_2023-05-21_17-59-31_800_6168806748029619938-1/_mr_10001/_hive_qe_engineering_hive_2023-05-21_17-59-31_800_6168806748029619938-1
2023-05-21 17:59:32,083 INFO parse.CalcitePlanner: CBO Succeeded: optimized logical plan.
2023-05-21 17:59:32,102 INFO ppd.OpProcFactory: Processing for FS(3)
2023-05-21 17:59:32,102 INFO ppd.OpProcFactory: Processing for SEL(2)
2023-05-21 17:59:32,102 INFO ppd.OpProcFactory: Processing for FIL(1)
2023-05-21 17:59:32,114 INFO ppd.OpProcFactory: Processing for TS(0)
2023-05-21 17:59:32,196 INFO parse.CalcitePlanner: Completed plan generation
2023-05-21 17:59:32,196 INFO parse.CalcitePlanner: Not eligible for results caching - no mr/tez/spark jobs
2023-05-21 17:59:32,197 INFO ql.Driver: Semantic Analysis Completed (retryal = false)
2023-05-21 17:59:32,197 INFO ql.Driver: Returning Hive schema: Schema(fieldschemas:[FieldSchema(name:hospital.hospital_id, type:string, comment:null), FieldSchema(name:hospital.hospital_name, type:string, comment:null), FieldSchema(name:hospital.state, type:string, comment:null), FieldSchema(name:hospital.country, type:string, comment:null)]), properties:null
2023-05-21 17:59:32,201 INFO exec.TableScanOperator: Initializing operator TS(0)

```

```
2023-05-21 17:59:32,217 INFO ql.Driver: Executing command(queryId=sheeba_20230521175931_40f8255f-6901-4364-82f8-9163f5172aa8): select * from hospital where hospital_id='H1013'
2023-05-21 17:59:32,221 INFO ql.Driver: Completed executing command(queryId=sheeba_20230521175931_40f8255f-6901-4364-82f8-9163f5172aa8); Time taken: 0.004 seconds
OK
2023-05-21 17:59:32,228 INFO ql.Driver: OK
2023-05-21 17:59:32,228 INFO ql.Driver: Concurrency mode is disabled, not creating a lock manager
2023-05-21 17:59:32,234 INFO mapred.FileInputFormat: Total input files to process : 1
2023-05-21 17:59:32,258 INFO lazy.LazyStruct: Missing fields! Expected 5 fields but only got 1! Ignoring similar problems.
2023-05-21 17:59:32,259 INFO exec.TableScanOperator: RECORDS_OUT_INTERMEDIATE:0, RECORDS_OUT_OPERATOR_TS:0:41,
2023-05-21 17:59:32,261 INFO exec.FilterOperator: RECORDS_OUT_INTERMEDIATE:0, RECORDS_OUT_OPERATOR_FILTER:4:1,
2023-05-21 17:59:32,264 INFO exec.SelectOperator: RECORDS_OUT_OPERATOR_SEL:2:1, RECORDS_OUT_INTERMEDIATE:0,
2023-05-21 17:59:32,265 INFO exec.ListsSinkOperator: RECORDS_OUT_OPERATOR_LIST_SINK:5:1, RECORDS_OUT_INTERMEDIATE:0,
H1013 Apollo Health City - Jubilee Hills Hyderabad Telangana India
Time taken: 0.449 seconds, Fetched: 1 row(s)
2023-05-21 17:59:32,271 INFO ClipDriver: Time taken: 0.449 seconds, Fetched: 1 row(s)
2023-05-21 17:59:32,274 INFO conf.HiveConf: Using the default value passed in for log id: fe20bda1-1ed1-49eb-b699-a55e04cafe74
2023-05-21 17:59:32,272 2023 INFO session.SessionState: Resetting thread name to main
hive: 2023-05-21 17:59:34,293 INFO metastore.HiveMetaStore: 2: Opening raw store with implementation class:org.apache.hadoop.hive.metastore.ObjectStore
2023-05-21 17:59:34,295 INFO metastore.ObjectStore: ObjectStore, initialized called
2023-05-21 17:59:34,300 INFO metastore.MetaStoreDirectSql: Using direct SQL, underlying DB is DERBY
2023-05-21 17:59:34,300 INFO metastore.ObjectStore: Initialized ObjectStore
```

Spark is installed

```
C:\Administrator: Command Prompt - spark-shell
04/07/2023 08:13 AM      1,935 find-spark-home
04/07/2023 08:13 AM      2,685 find-spark-home.cmd
04/07/2023 08:13 AM      2,337 load-spark-env.cmd
04/07/2023 08:13 AM      2,678 load-spark-env.sh
04/07/2023 08:13 AM      2,636 pyspark
04/07/2023 08:13 AM      1,170 pyspark.cmd
04/07/2023 08:13 AM      1,542 pyspark2.cmd
04/07/2023 08:13 AM      1,030 run-example
04/07/2023 08:13 AM      1,223 run-example.cmd
04/07/2023 08:13 AM      3,561 spark-class
04/07/2023 08:13 AM      1,180 spark-class.cmd
04/07/2023 08:13 AM      2,891 spark-class2.cmd
04/07/2023 08:13 AM      1,154 spark-connect-shell
04/07/2023 08:13 AM      3,122 spark-shell
04/07/2023 08:13 AM      1,178 spark-shell.cmd
04/07/2023 08:13 AM      1,818 spark-shell2.cmd
04/07/2023 08:13 AM      1,065 spark-sql
04/07/2023 08:13 AM      1,173 spark-sql.cmd
04/07/2023 08:13 AM      1,118 spark-sql2.cmd
04/07/2023 08:13 AM      1,040 spark-submit
04/07/2023 08:13 AM      1,180 spark-submit.cmd
04/07/2023 08:13 AM      1,155 spark-submit2.cmd
04/07/2023 08:13 AM      1,047 sparkR
04/07/2023 08:13 AM      1,168 sparkR.cmd
04/07/2023 08:13 AM      1,097 sparkR2.cmd
28 File(s)      55,323 bytes
2 Dir(s) 227,609,923,584 bytes free

C:\spark\spark-3.4.0-bin-hadoop3\bin>spark-shell
Setting default log level to "WARN".
To adjust logging level use sc.setLogLevel(newLevel). For SparkR, use setLogLevel(newLevel).
Spark context Web UI available at http://LAPTOP-51QJSCHA:4040
Spark context available as 'sc' (master = local[*], app id = local-1684683448971).
Spark session available as 'spark'.
Welcome to


$$\begin{array}{c} \diagup \diagdown \diagup \diagdown \\ \diagup \diagdown \diagup \diagdown \end{array}$$

version 3.4.0

Using Scala version 2.12.17 (Java HotSpot(TM) 64-Bit Server VM, Java 1.8.0_351)
Type in expressions to have them evaluated.
```

The screenshot shows the Apache Spark 3.4.0 Executors application UI. The top navigation bar includes links for Jobs, Stages, Storage, Environment, and Executors, with Executors being the active tab. Below the navigation is a summary table for executors, followed by a detailed table for the single executor 'driver'.

	RDD Blocks	Storage Memory	Disk Used	Cores	Active Tasks	Failed Tasks	Complete Tasks	Total Tasks	Task Time (GC Time)	Input	Shuffle Read	Shuffle Write	Excluded
Active(1)	0	0.0 B / 366.3 MiB	0.0 B	8	0	0	0	0	17 min (0.3 s)	0.0 B	0.0 B	0.0 B	0
Dead(0)	0	0.0 B / 0.0 B	0.0 B	0	0	0	0	0	0.0 ms (0.0 ms)	0.0 B	0.0 B	0.0 B	0
Total(1)	0	0.0 B / 366.3 MiB	0.0 B	8	0	0	0	0	17 min (0.3 s)	0.0 B	0.0 B	0.0 B	0

Executor ID	Address	Status	RDD Blocks	Storage Memory	Disk Used	Cores	Active Tasks	Failed Tasks	Complete Tasks	Total Tasks	Task Time (GC Time)	Input	Shuffle Read	Shuffle Write	Thread Dump
driver	LAPTOP-51QJSCHA57959	Active	0	0.0 B / 366.3 MiB	0.0 B	8	0	0	0	0	17 min (0.3 s)	0.0 B	0.0 B	0.0 B	Thread Dump

Data Analysis using Spark

```
1 pip install pyspark
```

Requirement already satisfied: pyspark in c:\users\sheeba\anaconda3new\lib\site-packages (3.4.0)
Requirement already satisfied: py4j==0.10.9.7 in c:\users\sheeba\anaconda3new\lib\site-packages (from pyspark) (0.10.9.7)
Note: you may need to restart the kernel to use updated packages.

```
1 from pyspark.sql import SparkSession
```

```
1 spark = SparkSession.builder \
2     .appName("Spark Hive Example") \
3     .config("spark.sql.warehouse.dir", "hdfs://localhost:9870/user/hive/warehouse") \
4     .config("spark.sql.catalogImplementation", "hive") \
5     .enableHiveSupport() \
6     .getOrCreate()
```

```
: # USE healthcare_system Database
spark.sql("use healthcare_system").show()
```

```
++  
||  
++  
++
```

```
: # Print all the tables which are present in the healthcare_system database.
sparkdf = spark.sql("show tables")
sparkdf.toPandas().to_csv('Spark Outputs for Visualization/Database.csv')
sparkdf.show()
```

```
+-----+-----+
| database| tableName|isTemporary|
+-----+-----+-----+
|healthcare_system| claims| false|
|healthcare_system| disease| false|
|healthcare_system| groups| false|
|healthcare_system| grp_subgrp| false|
|healthcare_system| hospital| false|
|healthcare_system| patient| false|
|healthcare_system| subgroup| false|
|healthcare_system| subscriber| false|
+-----+-----+-----+
```

```
: # Describe the claims table
sparkdf = spark.sql("desc claims")
sparkdf.toPandas().to_csv('Spark Outputs for Visualization/Claims_tble.csv')
sparkdf.show()
```

```
+-----+-----+-----+
| col_name|data_type|comment|
+-----+-----+-----+
| claim_id| int| null|
| patient_id| int| null|
| disease_name| string| null|
| sub_id| string| null|
| claim_or_rejected| string| null|
| claim_type| string| null|
| claim_amount| double| null|
| claim_date| string| null|
+-----+-----+-----+
```

```
# Describe the disease table
sparkdf = spark.sql("desc disease")
sparkdf.toPandas().to_csv('Spark Outputs for Visualization/disease_tble.csv')
sparkdf.show()
```

col_name	data_type	comment
disease_id	int	null
disease_name	string	null
subgrp_id	string	null

```
# Describe the groups table
sparkdf = spark.sql("desc groups")
sparkdf.toPandas().to_csv('Spark Outputs for Visualization/groups_tble.csv')
sparkdf.show()
```

col_name	data_type	comment
grp_sk	int	null
grp_id	string	null
grp_name	string	null
premium_written	int	null
city	string	null
zip_code	int	null
country	string	null
grp_type	string	null

```
# Describe the grp_subgrp table
sparkdf = spark.sql("desc grp_subgrp")
sparkdf.toPandas().to_csv('Spark Outputs for Visualization/grp_subgrp_tble.csv')
sparkdf.show()
```

col_name	data_type	comment
grpsub_sk	int	null
g_id	string	null
s_id	string	null

```
# Describe the hospital table
sparkdf = spark.sql("desc hospital")
sparkdf.toPandas().to_csv('Spark Outputs for Visualization/hospital_tble.csv')
sparkdf.show()
```

col_name	data_type	comment
hospital_id	string	null
hospital_name	string	null
city	string	null
state	string	null
country	string	null

```
# Describe the patient table
sparkdf = spark.sql("desc patient")
sparkdf.toPandas().to_csv('Spark Outputs for Visualization/patient_tble.csv')
sparkdf.show()
```

col_name	data_type	comment
patient_id	int	null
patient_name	string	null
patient_gender	string	null
patient_birth_date	string	null
patient_phone	string	null
disease_name	string	null
city	string	null
hospital_id	string	null

```
# Describe the subgroup table
sparkdf = spark.sql("desc subgroup")
sparkdf.toPandas().to_csv('Spark Outputs for Visualization/subgroup_tble.csv')
sparkdf.show()
```

col_name	data_type	comment
subgrp_sk	int	null
subgrp_id	string	null
subgrp_name	string	null
monthly_premium	double	null

```

: # Describe the subscriber table
sparkdf = spark.sql("desc subscriber")
sparkdf.toPandas().to_csv('Spark Outputs for Visualization/subscriber_tble.csv')
sparkdf.show()

+-----+-----+-----+
| col_name|data_type|comment|
+-----+-----+-----+
| sub_id| string| null|
| first_name| string| null|
| last_name| string| null|
| street| string| null|
| birth_date| string| null|
| gender| string| null|
| phone| string| null|
| city| string| null|
| zip_code| int| null|
| country| string| null|
| subgrp_id| string| null|
| elig_ind| string| null|
| eff_date| string| null|
| term_date| string| null|
+-----+-----+-----+
: # which disease having maximum number of claims.
results = spark.sql("select disease_name,count(claim_id) as \
max from claims group by disease_name order by max desc")

: results.toPandas().to_csv('Spark Outputs for Visualization/query1.csv')
results.show()

```

```

: # which disease having maximum number of claims.
results = spark.sql("select disease_name,count(claim_id) as \
max from claims group by disease_name order by max desc")

: results.toPandas().to_csv('Spark Outputs for Visualization/query1.csv')
results.show()

+-----+-----+
| disease_name|max|
+-----+-----+
| Glaucoma| 3|
| Phenylketonuria| 3|
| Galactosemia| 3|
| Pet allergy| 3|
| Anthrax| 3|
| Head banging| 3|
| Asthma| 2|
| Scurvy| 2|
| Drug consumption| 2|
| Bladder cancer| 2|
| Rett Syndrome| 2|
| Choking| 2|
| Flu| 2|
| Lymphedema| 2|
| Mold allergy| 2|
| Fanconi anaemia| 2|
| Measles| 2|
| Stroke| 2|
| Cholera| 2|
| Malaria| 2|
+-----+-----+
only showing top 20 rows

```

```

# Find those Subscribers having age less than 30 and they subscribe any subgroup
subscriber_ages = spark.sql("select birth_date,current_date() as CurrentDate,\n    year(current_date())-year(birth_date) as age from subscriber")

res = subscriber_ages.filter(subscriber_ages.age < 30).count()

print("Subscribers having age less than 30 --> ",res)

Subscribers having age less than 30 --> 4

# Find out which group has maximum subgroups.
sparkdf = spark.sql("select g_id,count(s_id) as max from grp_subgrp group by g_id order by max desc")
sparkdf.toPandas().to_csv('Spark Outputs for Visualization/query3.csv')
sparkdf.show()

+-----+---+
| g_id|max|
+-----+---+
|GRP104| 2|
|GRP147| 2|
|GRP143| 2|
|GRP108| 1|
|GRP105| 1|
|GRP101| 1|
|GRP148| 1|
|GRP102| 1|
|GRP103| 1|
|GRP114| 1|
|GRP142| 1|
|GRP133| 1|
|GRP112| 1|
|GRP122| 1|
|GRP138| 1|
|GRP157| 1|
|GRP127| 1|
|GRP123| 1|
|GRP126| 1|
|GRP110| 1|
+-----+---+
only showing top 20 rows

```

```

: # Find out hospital which serve most number of patients
sparkdf = spark.sql("select hospital_name,count(patient_id) as total_patient \
    from hospital join patient on hospital.hospital_id=patient.hospital_id \
    group by hospital_name order by total_patient desc")

sparkdf.toPandas().to_csv('Spark Outputs for Visualization/query4.csv')
sparkdf.show()

+-----+-----+
| hospital_name|total_patient|
+-----+-----+
| Manipal Hospitals|      9|
| Apollo Hospitals ...|     8|
| Medanta The Medicity|     7|
| Jaslok Hospital a...|     6|
| Indraprastha Apol...|     5|
| PGIMER - Postgrad...|     4|
| Apollo Hospital ....|     4|
| Fortis Hospital M...|     4|
| King Edward Memor...|     3|
| Apollo Health Cit...|     3|
| Yashoda Hospital ...|     3|
| Bombay Hospital &...|     3|
| Fortis Hiranandani...|     2|
| Lilavati Hospital...|     2|
| The Christian Med...|     2|
| Fortis Flt. Lt. R...|     1|
| P. D. Hinduja Nat...|     1|
| Breach Candy Hosp...|     1|
| All India Institu...|     1|
| Sir Ganga Ram Hos...|     1|
+-----+-----+

```

```
# Find out which subgroups subscribe most number of times
sparkdf = spark.sql("select subgroup_name, \
    count(sub_id) as cunt from subgroup join \
    subscriber on subgroup.subgrp_id = subscriber.subgrp_id group by subgroup_name order by cunt desc")

sparkdf.toPandas().to_csv('Spark Outputs for Visualization/query5.csv')
sparkdf.show()
```

subgrp_name cunt
Therapy 14
Hereditary 11
Viral 11
Deficiency Diseases 11
Physiology 10
Allergies 10
Accident 10
Cancer 9
Self inflicted 7
Infectious disease 7

```
# Find out total number of claims which were rejected
sparkdf = spark.sql("select claim_or_rejected,count(claim_id) from claims group by claim_or_rejected")

sparkdf.toPandas().to_csv('Spark Outputs for Visualization/query6.csv')
sparkdf.show()
print("The above Result shows the total 52 claims were rejected")
```

claim_or_rejected count(claim_id)
Y 18
N 52

The above Result shows the total 52 claims were rejected

```
# From where most claims are comming (city)
sparkdf = spark.sql("select patient.city,count(claim_id) as maxclaim from patient join claims on \
    claims.patient_id = patient.patient_id group by patient.city order by maxclaim desc limit 10")
sparkdf.toPandas().to_csv('Spark Outputs for Visualization/query7.csv')
sparkdf.show()
```

city maxclaim
Mysore 2
Amravati 2
Kamarhati 2
Jabalpur 2
Bihar Sharif 2
Ghaziabad 2
Morbi 2
Karimnagar 2
Bangalore 1
Udaipur 1

```
# Which groups of policies subscriber subscribe mostly Goverment or private
sparkdf = spark.sql("select grp_type,count(grp_id) from groups group by grp_type")
sparkdf.toPandas().to_csv('Spark Outputs for Visualization/query8.csv')
sparkdf.show()
```

grp_type count(grp_id)
Govt. 7
Private 51

```
res = spark.sql("select grp_subgrp.g_id from subscriber \
    join grp_subgrp on grp_subgrp.s_id = subscriber.subgrp_id")

res.registerTempTable("grp_tble")

sparkdf = spark.sql("select grp_type,count(grp_tble.g_id) from groups \
    join grp_tble on groups.grp_id = grp_tble.g_id group by grp_type")

sparkdf.toPandas().to_csv('Spark Outputs for Visualization/query9.csv')
sparkdf.show()
```

```
+-----+-----+
|grp_type|count(g_id)|
+-----+-----+
|  Govt.|      35|
| Private|    347|
+-----+-----+
: print("The Above Result shows that most of the subscribers subscribe private groups")
The Above Result shows that most of the subscribers subscribe private groups
: # Average monthly premium subscriber pay to insurance company subgroup
sparkdf = spark.sql("select subgroup.subgrp_id,avg(monthly_premium) from subscriber right join \
    subgroup on subscriber.subgrp_id = subgroup.subgrp_id group by \
    subgroup.subgrp_id order by subgroup.subgrp_id")
sparkdf.toPandas().to_csv('Spark Outputs for Visualization/query10.csv')
sparkdf.show()
+-----+-----+
|subgrp_id|avg(monthly_premium)|
+-----+-----+
| S101|      3000.0|
| S102|      1800.0|
| S103|      2000.0|
| S104|      1500.0|
| S105|      2300.0|
| S106|      1200.0|
| S107|      3200.0|
| S108|      1500.0|
| S109|      2000.0|
| S110|      1000.0|
+-----+-----+
: # Find out which group is most profitable
sparkdf = spark.sql("select grp_id,premium_written from groups order by premium_written desc")
sparkdf.toPandas().to_csv('Spark Outputs for Visualization/query11.csv')
sparkdf.show()
```

```
+-----+-----+
|grp_id|premium_written|
+-----+-----+
|GRP147|      99000|
|GRP131|      99000|
|GRP123|      99000|
|GRP118|      97000|
|GRP154|      95000|
|GRP133|      93000|
|GRP157|      92000|
|GRP134|      90000|
|GRP143|      90000|
|GRP106|      89000|
|GRP129|      88000|
|GRP152|      87000|
|GRP153|      86000|
|GRP150|      84000|
|GRP124|      81000|
|GRP122|      79000|
|GRP115|      79000|
|GRP121|      78000|
|GRP109|      78000|
|GRP101|      72000|
+-----+-----+
only showing top 20 rows
```

```
: # List all the patients below age of 18 who admit for cancer
sparkdf = spark.sql("select patient_id, patient_name,\n    year(current_date())-year(patient_birth_date) as age from patient \
    where disease_name like '%cancer' order by age limit 2")
sparkdf.toPandas().to_csv('Spark Outputs for Visualization/query12.csv')
sparkdf.show()
+-----+-----+-----+
|patient_id|patient_name|age|
+-----+-----+-----+
|     194166|        NA|   9|
|     197441| Deependu| 18|
+-----+-----+
```

```

: # List patients who have cashless insurance and have total charges greater than or equal for Rs. 50,000.

sparkdf = spark.sql("select patient_name,patient_gender,patient_birth_date \
    from patient join claims on patient.patient_id = claims.patient_id \
    where claim_amount >= 50000 and claim_type = 'claims of value'")
sparkdf.toPandas().to_csv('Spark Outputs for Visualization/query13.csv')
sparkdf.show()

+-----+-----+-----+
|patient_name|patient_gender|patient_birth_date|
+-----+-----+-----+
| Anjushree|     Male| 1982-06-28|
| Chitrangan|   Female| 2020-10-27|
| Gensho|     Male| 1991-07-27|
| Vaijayanti|     Male| 1989-04-06|
| Aakar|   Female| 1990-04-24|
| NA|   Female| 1959-01-06|
| Saroj|   Female| 1953-07-21|
| Bhagyan|   Female| 2011-02-26|
| Dharmadas|     Male| 1964-10-25|
| Umang|   Female| 2017-02-26|
| NA|     Male| 1955-06-03|
| Kishan|     Male| 1955-06-30|
| NA|   Female| 1953-04-04|
| Devnath|   Female| 1982-02-22|
| Harbir|   Female| 1960-02-24|
| Ekant|     Male| 1969-11-01|
| NA|     Male| 2013-10-30|
| NA|     Male| 1956-04-04|
| Lalit|   Female| 1978-04-30|
| Ujjawal|     Male| 1965-12-31|
+-----+-----+-----+


: # List female patients over the age of 40 that have undergone knee surgery in the past year

sparkdf = spark.sql("select patient_name from patient where patient_gender = 'Female' and disease_name = 'Heart Attack'")
sparkdf.toPandas().to_csv('Spark Outputs for Visualization/query14.csv')
sparkdf.show()

+-----+
|patient_name|
+-----+
|      Upasana|
+-----+

```

Spark outputs:

Claims_tbl.csv

Database.csv

disease_tbl.csv

groups_tbl.csv

grp_subgrp_tble.csv

hospital_tbl.csv

patient_tbl.csv

query1.csv

query2.csv

query3.csv

query4.csv

query5.csv

query6.csv

query7.csv

query8.csv

query9.csv

query10.csv

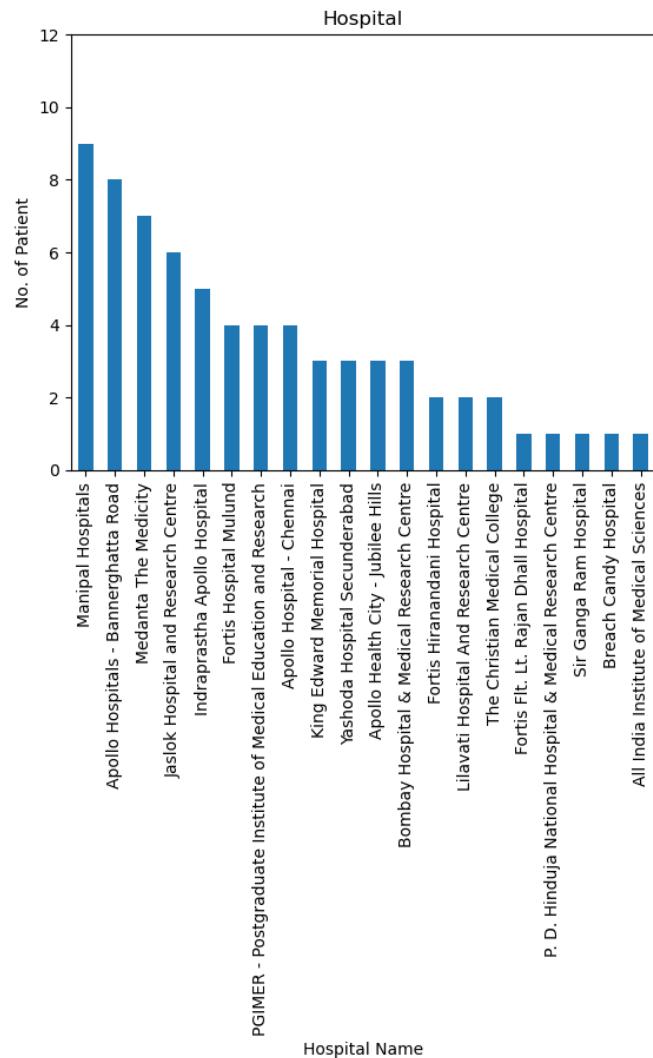
query11.csv

query12.csv

query13.csv

Visualization:

```
In [1]: 1 import pandas as pd  
2 import seaborn as sns  
3 import matplotlib.pyplot as plt  
  
In [2]: 1 # Find out hospital which serve most number of patients  
2  
3 df=pd.read_csv('query4.csv')  
  
In [3]: 1 df.plot(x='hospital_name',y='total_patient',kind='bar',legend=None)  
2 plt.title("Hospital")  
3 plt.ylabel('No. of Patient')  
4 plt.xlabel("Hospital Name")  
5 plt.ylim(ymin=0,ymax=12)  
6 plt.savefig('Hospital with most patient.png',bbox_inches='tight')
```



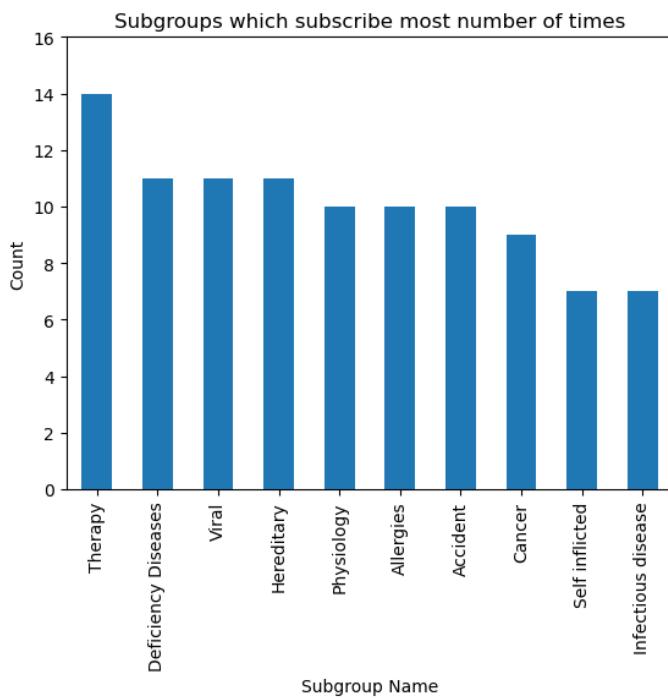
```
In [4]:  
1 ## Find out which subgroups subscribe most number of times  
2  
3 df1=pd.read_csv('query5.csv')
```

```
In [5]: 1 df1
```

Out[5]:

Unnamed: 0	subgrp_name	cunt	
0	0	Therapy	14
1	1	Deficiency Diseases	11
2	2	Viral	11
3	3	Hereditary	11
4	4	Physiology	10
5	5	Allergies	10
6	6	Accident	10
7	7	Cancer	9
8	8	Self inflicted	7
9	9	Infectious disease	7

```
In [6]:  
1 df1.plot(x='subgrp_name',y='cunt',kind='bar',legend=None)  
2 plt.title("Subgroups which subscribe most number of times")  
3 plt.ylabel('Count')  
4 plt.xlabel("Subgroup Name")  
5 plt.ylim(ymin=0,ymax=16)  
6 plt.savefig('Subgroups which subscribe most number of times',bbox_inches='tight')
```



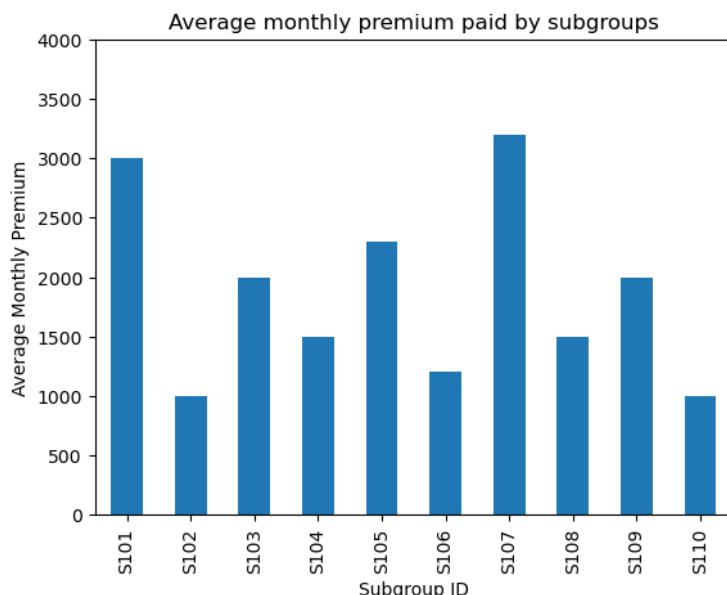
```
In [7]:  
1 # Average monthly premium subscriber pay to insurance company subgroup.  
2  
3 df2=pd.read_csv('query10.csv')
```

```
In [8]:  
1 df2
```

Out[8]:

	Unnamed: 0	subgrp_id	avg(monthly_premium)
0	0	S101	3000.0
1	1	S102	1000.0
2	2	S103	2000.0
3	3	S104	1500.0
4	4	S105	2300.0
5	5	S106	1200.0
6	6	S107	3200.0
7	7	S108	1500.0
8	8	S109	2000.0
9	9	S110	1000.0

```
In [9]:  
1 df2.plot(x='subgrp_id',y='avg(monthly_premium)',kind='bar',legend=None)  
2 plt.title("Average monthly premium paid by subgroups")  
3 plt.ylabel('Average Monthly Premium')  
4 plt.xlabel("Subgroup ID")  
5 plt.ylim(ymin=0,ymax=4000)  
6 plt.savefig('Average Monthly Premium Paid by Subgroups',bbox_inches='tight')
```



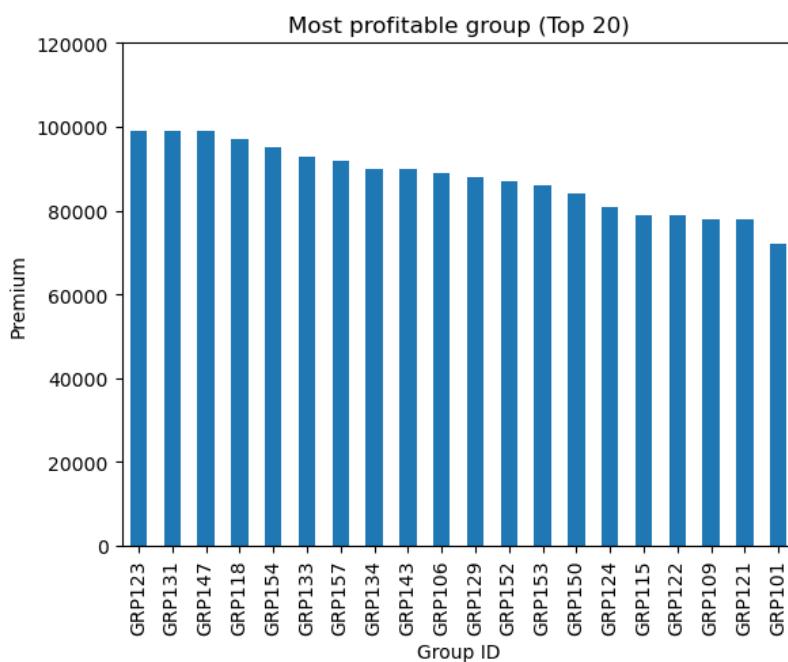
```
In [10]: 1 # Find out Which group is most profitable  
2  
3 df3=pd.read_csv('query11.csv')
```

```
In [11]: 1 df3.head()
```

Out[11]:

	Unnamed: 0	grp_id	premium_written
0	0	GRP123	99000
1	1	GRP131	99000
2	2	GRP147	99000
3	3	GRP118	97000
4	4	GRP154	95000

```
In [12]: 1 df3.head(n=20).plot(x='grp_id',y='premium_written',kind='bar',legend=None)  
2 plt.title("Most profitable group (Top 20)")  
3 plt.xlabel("Group ID")  
4 plt.ylabel("Premium")  
5 plt.ylim(ymin=0,ymax=120000)  
6 plt.savefig('Most Profitable Groups',bbox_inches='tight')
```



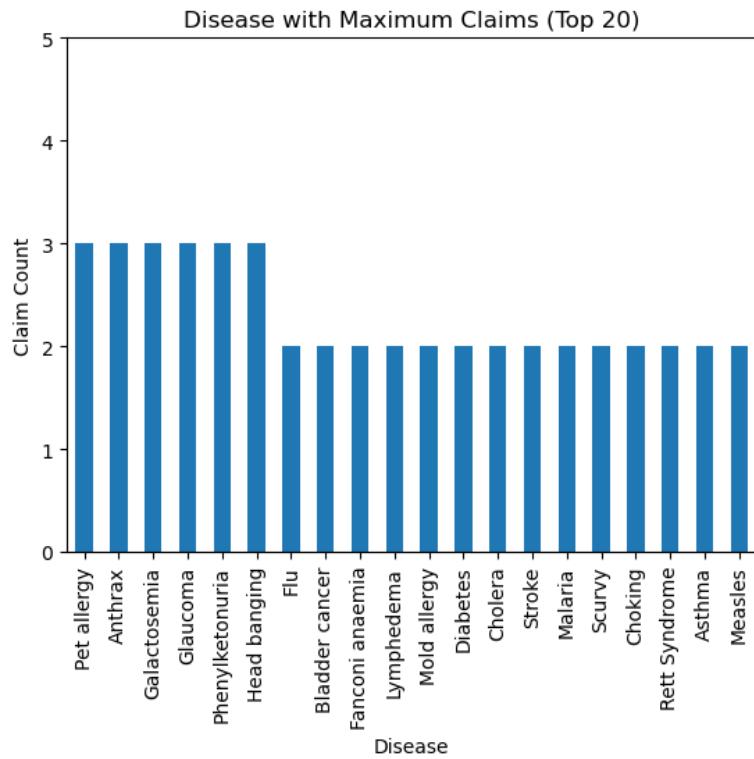
```
In [13]: 1 import matplotlib.pyplot as plt
2 from importlib import reload
3 plt=reload=plt

In [14]: 1 # which disease having maximum number of claims.
2
3 df4=pd.read_csv('query1.csv')

In [15]: 1 df4.head()

Out[15]:
      Unnamed: 0  disease_name  max
0              0    Pet allergy     3
1              1        Anthrax     3
2              2   Galactosemia     3
3              3       Glaucoma     3
4              4  Phenylketonuria     3
```

```
In [16]: 1 df4.head(n=20).plot(x='disease_name',y='max',kind='bar',legend=None)
2 plt.title("Disease with Maximum Claims (Top 20)")
3 plt.xlabel("Disease")
4 plt.ylabel("Claim Count")
5 plt.ylim(ymin=0,ymax=5)
6 plt.savefig('Disease with max Claim',bbox_inches='tight')
```



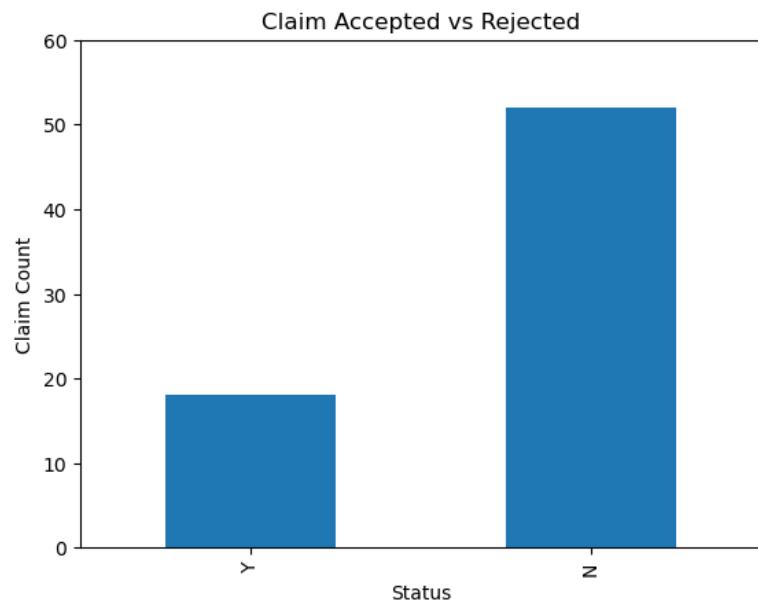
```
In [17]: 1 # Find out total number of claims which were rejected
          2
          3 df5=pd.read_csv('query6.csv')
```

```
In [18]: 1 df5
```

Out[18]:

Unnamed: 0	claim_or_rejected	count(claim_id)
0	Y	18
1	N	52

```
In [19]: 1 df5.head(n=20).plot(x='claim_or_rejected',y='count(claim_id)',kind='bar',legend=None)
          2 plt.title("Claim Accepted vs Rejected")
          3 plt.xlabel("Status")
          4 plt.ylabel("Claim Count")
          5 plt.ylim(ymin=0,ymax=60)
          6 plt.savefig('Claim Accepted vs Rejected',bbox_inches='tight')
```



CHAPTER 6

CONCLUSION AND FUTURE WORKS

6.1 CONCLUSION

Implementing a big data analytics system in a healthcare insurance company offers tremendous potential for revenue growth and enhanced customer understanding. By addressing challenges such as data volume management, security, integration, analysis complexity, scalability, talent shortage, and ethical considerations, the project can pave the way for personalized insurance policy offerings, accurate royalty calculations, and improved service provision. Leveraging the power of big data analytics, the system enables the company to track customer behavior and conditions, leading to tailored insurance solutions and increased revenue. Additionally, the project promotes better decision-making through advanced analytics techniques, while adhering to data privacy regulations and ethical standards. Ultimately, the successful implementation of the project contributes to a better understanding of customers and an improved overall customer experience in the healthcare insurance industry. Some of our analysis includes average monthly premium for each subgroup, which disease have maximum number of claims, which group is most profitable etc.

6.2 FUTURE WORK

In this part of the implementation, the data files of JSON and csv formats are prepared and preprocessed. An RDBMS model is created for storage of contents of these data files which will be used for further processing and analysis in subsequent steps using technologies such as Sqoop, HDFS and Hive. This will enable us to analyze the data from a healthcare insurance company to obtain useful results and test it to increase the revenue of our company.

REFERENCES

[1]Sheeran, M., & Steele, R. (2017, October). A framework for big data technology in health and healthcare. In 2017 IEEE 8th Annual Ubiquitous Computing, Electronics and Mobile Communication Conference (UEMCON) (pp. 401-407). IEEE.

[2]Zheng, L., & Guo, L. (2020, April). Application of big data technology in insurance innovation. In International conference on education, economics and information management (ICEEIM 2019) (pp. 285-294). Atlantis Press.