**DATA WAREHOUSING BY IBM CLOUD COMPUTING**

**PHASE 4 : DEVELOPMENT PART 2**

**PROJECT OBJECTIVES:**

In this part you will continue building your project. Continue building the data warehouse by implementing ETL processes and enabling data exploration.

Implement ETL processes to extract, transform, and load data into the data warehouse.

Enable data architects to explore and analyze data within Db2 Warehouse using SQL queries and analysis techniques.

**PROJECT TASKS:**

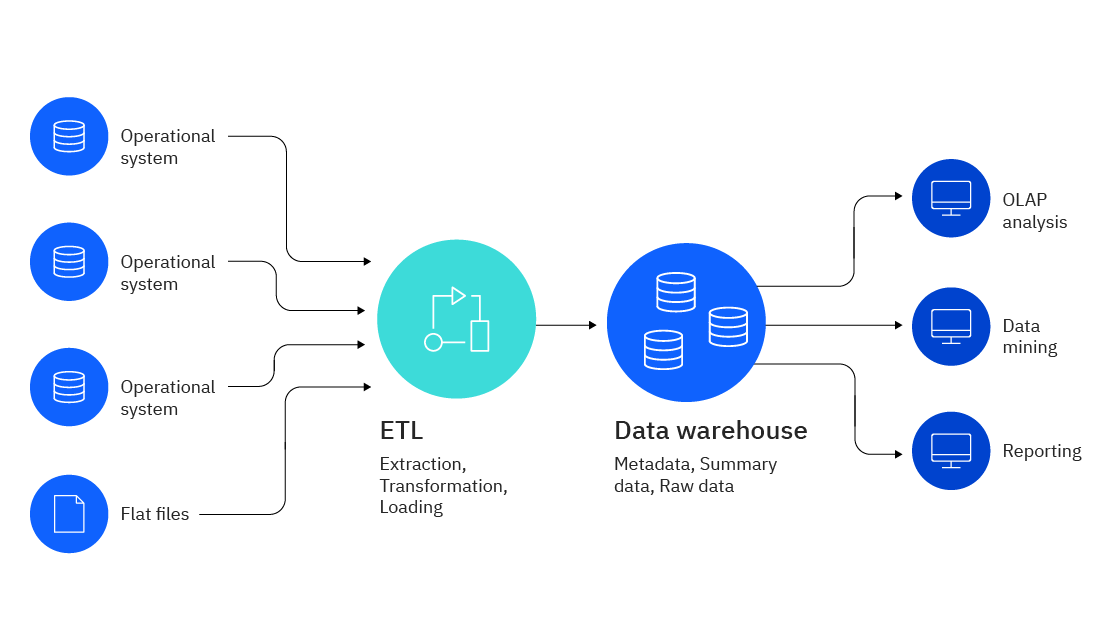
**INTRODUCTION:**

A data warehouse, or enterprise data warehouse (EDW), is a system that aggregates data from different sources into a single, central, consistent data store to support data analysis, data mining, artificial intelligence (AI), and machine learning. A data warehouse system enables an organization to run powerful analytics on huge volumes (petabytes and petabytes) of historical data in ways that a standard database cannot. Data warehousing systems have been a part of business intelligence (BI) solutions for over three decades, but they have evolved recently with the emergence of new data types and data hosting methods.

**DATA WAREHOUSING ARCHITECTURE:**

Generally speaking, data warehouses have a three-tier architecture, which consists of a:

* **Bottom tier:**The bottom tier consists of a data warehouse server, usually a relational database system, which collects, cleanses, and transforms data from multiple data sources through a process known as Extract, Transform, and Load (ETL) or a process known as Extract, Load, and Transform (ELT).
* **Middle tier:**The middle tier consists of an OLAP (i.e. online analytical processing) server which enables fast query speeds. Three types of OLAP models can be used in this tier, which are known as ROLAP, MOLAP and HOLAP. The type of OLAP model used is dependent on the type of database system that exists.
* **Top tier:**The top tier is represented by some kind of front-end user interface or reporting tool, which enables end users to conduct ad-hoc data analysis on their business data.



**COMPONENTS OF DATAWAREHOUSE ARCHITECTURE:**

**1.ETL:**

When database analysts want to move data from a data source into their data warehouse, this is the process they use. In short, ETL converts data into a usable format so that once it’s in the data warehouse, it can be analyzed/queried/etc.

**2.Metadata:**

Metadata is data about data. Basically, it describes all of the data that’s stored in a system to make it searchable. Some examples of metadata include authors, dates or locations of an article, create date of a file, the size of a file, etc. Think of it like the titles of a column in a spreadsheet. Metadata allows you to organize your data to make it usable, so you can analyze it to create dashboards and reports.

**3.SQL query processing:**

SQL is the de facto standard language for querying your data. This is the language that analysts use to pull out insights from their data stored in the data warehouse. Typically data warehouses have proprietary SQL query processing technologies tightly coupled with the compute. This allows for very high performance when it comes to your analytics. One thing to note, however, is that the cost of a data warehouse can start getting expensive the more data and SQL compute resources you have.

**EXAMPLE:**

SELECT c.customer\_name, SUM(o.total\_amount) AS total\_spent

FROM customers c JOIN orders o ON c.customer\_id = o.customer\_id

GROUP BY c.customer\_name

ORDER BY total\_spent DESC;

**4.Data layer:**

The data layer is the access layer that allows users to actually get to the data. This is typically where you’d find a data mart. This layer partitions segments of your data out depending on who you want to give access to, so you can get very granular across your organization. For instance, you may not want to give your sales team access to your HR team’s data, and vice versa.

**IMPLEMENTATION OF ETL PROCESS:**

**1. Extract Data:**

To extract data, you can use various libraries in Python. In this example, we'll use the pandas library to read data from a CSV file.

CODE:

import pandas as pd

# Extract data from a CSV file

source\_data = pd.read\_csv('source\_data.csv')

**2. Transform Data:**

Perform data transformations as needed. For this example, let's convert a column to uppercase:

CODE:

# Transform data

Source\_data['column\_name'] = source\_data['column\_name'].str.upper()

**3. Load Data into Db2 Warehouse:**

To load data into Db2 Warehouse, you can use Python libraries like ibm\_db or SQLAlchemy. Make sure to replace the placeholders with your actual database connection details.

CODE:

import ibm\_db

# Database connection details

conn\_str = "DATABASE=your\_db;HOSTNAME=your\_hostname;PORT=your\_port;PROTOCOL=TCPIP;UID=your\_username;PWD=your\_password;"# Create a database connection

conn ibm\_db.connect(conn\_str, "", "")

# Create a cursor

stmt = ibm\_db.exec\_immediate(conn, "")

# Load data into a table

for index, row in source\_data.iterrows():

query = f"INSERT INTO your\_table (column1, column2) VALUES ('{row['column1']}', '{row['column2']}')"ibm\_db.exec\_immediate(conn, query)

ibm\_db.commit(conn)

**DATA EXPLORATION USING SQL QUERIES:**

Now, let's enable data architects to explore and analyze data within Db2 Warehouse using SQL queries and analysis techniques.

**1. Connect to Db2 Warehouse:**

Data architects can use Python and libraries like ibm\_db to connect to Db2 Warehouse.

CODE:

import ibm\_db

# Database connection details

conn\_str="DATABASE=your\_db;HOSTNAME=your\_hostname;PORT=your\_port;PROTOCOL=TCPIP;UID=your\_username;PWD=your\_password;"

# Create a database connection

conn = ibm\_db.connect(conn\_str, "", "")

**2. Run SQL Queries:**

Data architects can execute SQL queries to analyze the data. For example, to retrieve data from a table and calculate the average of a column:

CODE:

import ibm\_db\_dbi

# Create a database connection (using ibm\_db\_dbi for DataFrames)

conn = ibm\_db\_dbi.Connection(conn)

# Run SQL query

query = "SELECT column1, AVG(column2) FROM your\_table GROUP BY column1"

result = pd.read\_sql\_query(query, conn)

print(result)

**3. Generate Analysis Outputs:**

Data architects can use Python libraries like pandas, matplotlib, or seaborn to generate visualizations and insights from the data. For example, creating a bar chart of the results:

CODE:

import matplotlib.pyplot as plt

# Create a bar chart

plt.bar(result['column1'], result['AVG(column2)'])

plt.xlabel('Category')

plt.ylabel('Average Value')

plt.title('Average Value by Category')

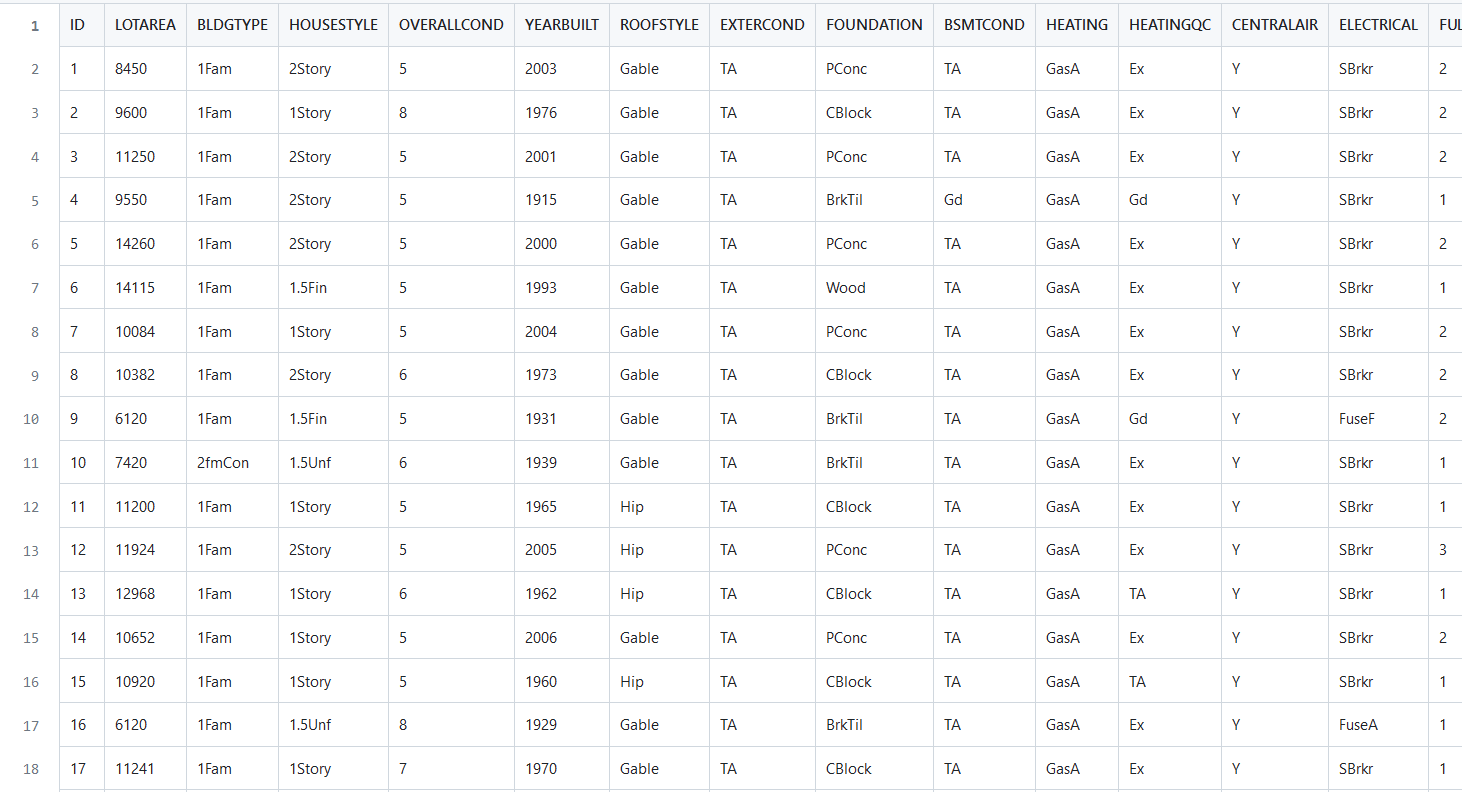
plt.show()

This code provides a basic framework for ETL processes in Python and data exploration in Db2 Warehouse. Please replace placeholders with your actual database and data details. Additionally, ensure that you have the necessary Python libraries installed and configured according to your environment.

**SCHEMAS IN DATA WAREHOUSE:**

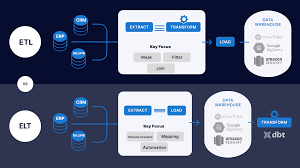


**EXAMPLE DATASET:**

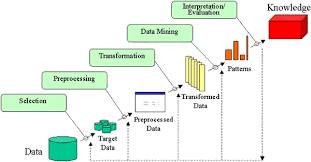


**OUTPUT IMAGE:**

**1.ETL process:**



**2.Data Exploration:**



**BENEFITS OF DATA WAREHOUSE:**

 **Better data quality:** A data warehouse centralizes data from a variety of data sources, such as transactional systems, operational databases, and flat files. It then cleanses it, eliminates duplicates, and standardizes it to create a single source of the truth.

 **Faster, business insights:**Data from disparate sources limit the ability of decision makers to set business strategies with confidence. Data warehouses enable data integration, allowing business users to leverage all of a company’s data into each business decision.

 **Smarter decision-making:**  A data warehouse supports large-scale BI functions such as data mining (finding unseen patterns and relationships in data), artificial intelligence, and machine learning—tools data professionals and business leaders can use to get hard evidence for making smarter decisions in virtually every area of the organization, from business processes to financial management and inventory management.

 **Gaining and growing competitive advantage:**All of the above combine to help an organization finding more opportunities in data, more quickly than is possible from disparate data stores.

**CONCLUSION:**

The open data lakehouse allows you to run warehouse workloads on all kinds of data in an open and flexible architecture. Instead of a tightly coupled system, the data lakehouse is much more flexible and also can manage unstructured and semi-structured data like photos, videos, IoT data and more.The data lakehouse can also support your data science, ML and AI workloads in addition to your reporting and dashboarding workloads. If you are looking to upgrade from data warehouse architecture, then developing an open data lakehouse is the way to go.

**PREPARED BY:**

**1.RAJESHWARI R**

**2.PAVITHRA A**

**3.SHAMSHUNNAFIYA N**

**4.BHUVANESHWARI P**