Building a E-commerce Data Warehouse Using Hive
By Keyur Rajput

Step 1: Setup
1.1. Log in to your Cloudera environment.
1.2. Start the Hadoop cluster and Hive service:

```
sudo service hadoop-hdfs-datanode start
sudo service hadoop-hdfs-namenode start
sudo service hadoop-yarn-resourcemanager start
sudo service hadoop-yarn-nodemanager start
```

Step 2: Data Ingestion
2.1. Begin by downloading a sample e-commerce dataset from Kaggle. You can find a suitable dataset at - https://www.kaggle.com/datasets/carrie1/ecommerce-data
2.2. Upload the dataset to HDFS using the following commands:

```
hadoop fs -mkdir /user/hive/warehouse/ecommerce
hadoop fs -put /home/cloudera/Desktop/ecomdata.csv
/user/hive/warehouse/ecommerce/
```

Step 3: Start Hive
3.1. Open a terminal and run Hive:

```
hive
```

Step 4: Create a Hive Database
4.1. Inside the Hive shell, create a new database:
CREATE DATABASE ecommerce;

Step 5: Define External Tables:

```
USE ecommerce;
CREATE EXTERNAL TABLE ecommerce_data (
InvoiceNo STRING,
CompanyName STRING,
StockCode STRING,
Quantity INT,
InvoiceDate STRING,
UnitPrice DOUBLE,
CustomerID STRING,
Country STRING
)
ROW FORMAT DELIMITED FIELDS TERMINATED BY ','
LOCATION '/user/hive/warehouse/ecommerce/';
```

Step 6: Data Exploration and Understanding
6.1. Start exploring your data using Hive commands.
For example, to see the first five rows:

```
SELECT * FROM ecommerce_data LIMIT 5;
```

Step 7: Write Basic Queries
7.1. Begin by writing basic Hive queries.
For instance, to find the total number of records in your dataset:

```
SELECT COUNT(*) FROM ecommerce_data;
```

Step 8: Data Cleaning and Transformation
8.1. Data cleaning and transformation steps can be dataset-specific
-- To remove duplicate entries

```
INSERT OVERWRITE TABLE ecommerce_data
SELECT DISTINCT *
FROM ecommerce_data;
```

Step 9: Data Loading
9.1. Load data from the external table into an internal table.

```
CREATE TABLE customer AS
SELECT
customerid AS customer_id,
companyname AS company_name,
stockcode AS stock_code,
quantity AS purchase_quantity,
invoicedate AS invoice_date,
unitprice AS unit_price,
country AS customer_country
FROM ecommerce_data;
```

Step 10: More Advanced Analysis and Partitioning
10.1. Calculate the total revenue or find the top-selling products.

```
USE ecommerce;
-- Calculate total revenue
SELECT SUM(price * quantity) AS total_revenue
FROM ecommerce_data;
```

10.2. An example of partitioning by the `order_date` column:
– Create Table partitioned

```
CREATE TABLE ecommerce_partitioned (
   InvoiceNo STRING,
   CompanyName STRING,
   StockCode STRING,
   Quantity INT,
   UnitPrice INT,
   CustomerID STRING,
   Country STRING
)
PARTITIONED BY (order_date DATE)
STORED AS ORC
LOCATION '/user/hive/warehouse/ecommerce_partitioned/';
```

– Load Data into the partitioned table

```
INSERT OVERWRITE TABLE ecommerce_partitioned PARTITION (order_date)
SELECT
   InvoiceNo,
   CompanyName,
   StockCode,
   Quantity,
   UnitPrice,
   CustomerID,
   Country,
   CAST(InvoiceDate AS DATE) AS order_date
FROM ecommerce_data;
```

10.3. Select All Data:
-- To retrieve all rows from the table, you can use the following query:

```
SELECT * FROM ecommerce_data;
```

10.4. Total Number of Records:
-- To find out how many records are in the table, you can use the COUNT function:

```
SELECT COUNT(*) AS record_count FROM ecommerce_data;
```

10.5. Filter Data by Date Range:
-- If you have a date column (e.g., InvoiceDate) and you want a specific date range:

```
SELECT *
FROM ecommerce_data
```

```sql
WHERE InvoiceDate BETWEEN '2023-09-01' AND '2023-10-01';
```

10.6. Aggregations:
-- You can perform various aggregations on numeric columns.
For example, to find the total revenue:

```sql
SELECT SUM(UnitPrice * Quantity) AS total_revenue
FROM ecommerce_data;
```

10.7. Average Unit Price:
-- To calculate the average unit price of products in the dataset:

```sql
SELECT AVG(UnitPrice) AS avg_unit_price
FROM ecommerce_data;
```

10.8. Filter by Customer ID:
-- To filter data for a specific customer based on their CustomerID:

```sql
SELECT *
FROM ecommerce_data
WHERE CustomerID = '12345';
```