Customer Management Project

1.Project Overview & Business Goal

This project demonstrates the design and implementation of a modern data pipeline and data warehouse ecosystem.

- ❖ **Objective:** To integrate multiple data sources (customer data, usage logs, transactions, offers, and support tickets), store them in a centralized warehouse, and enable insight generation through PySpark/Hive and business dashboards in Power BI.

- ❖ **Business Goal:** Improve customer churn prediction and lifetime value (CLV) analysis, while providing a single source of truth for reporting.

2. Architecture & Technologies

The architecture follows a data lakehouse pattern, combining raw ingestion, warehouse modeling, and analytics:

- ❖ **Ingestion Layer**: Apache NiFi (streaming ingestion), Apache Sqoop (RDBMS to HDFS)

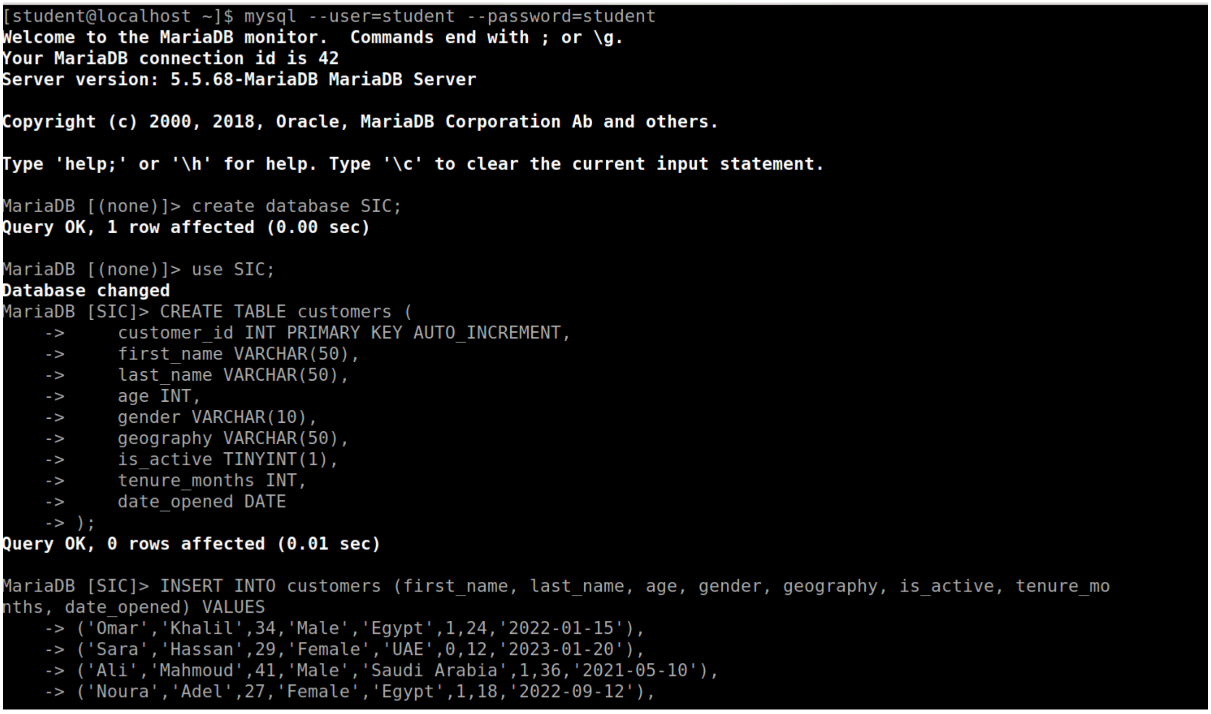
- ❖ **Storage Layer**: HDFS for raw data, Hive for structured warehouse

- ❖ **Processing Layer**: Hue/Hive for ETL, transformations, and analytics

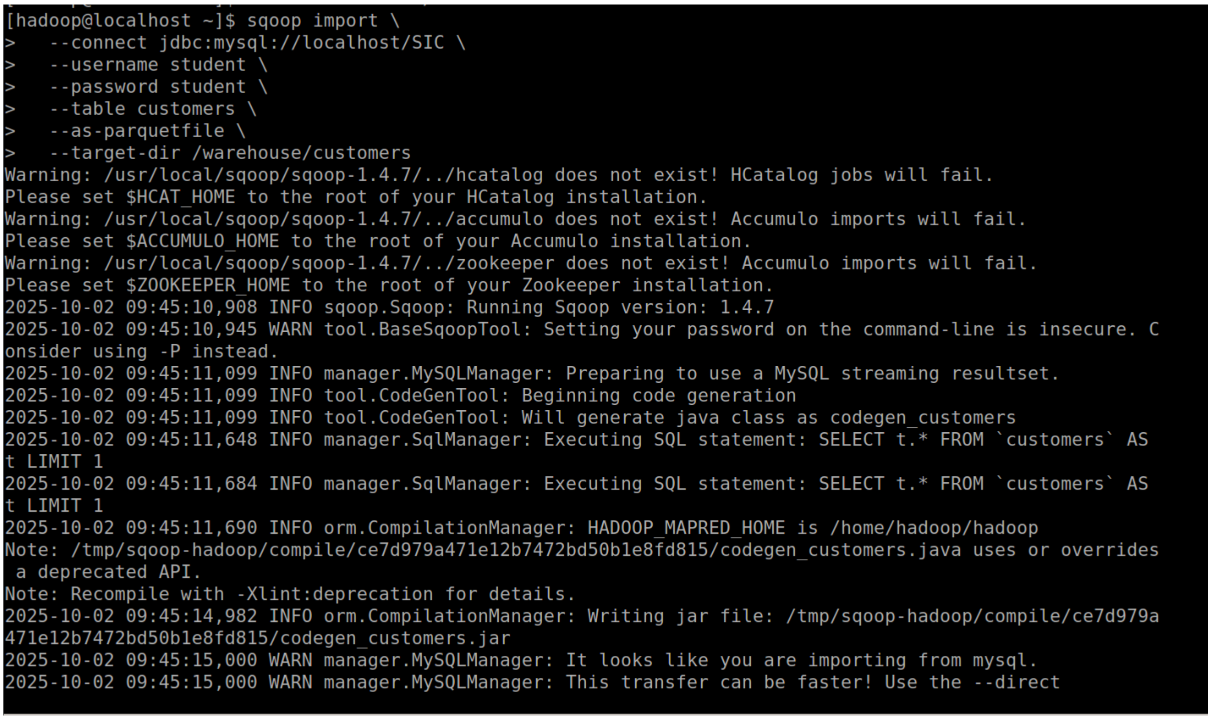
- ❖ **BI Layer**: Power BI for visualization and insights

3. Data Ingestion Layer

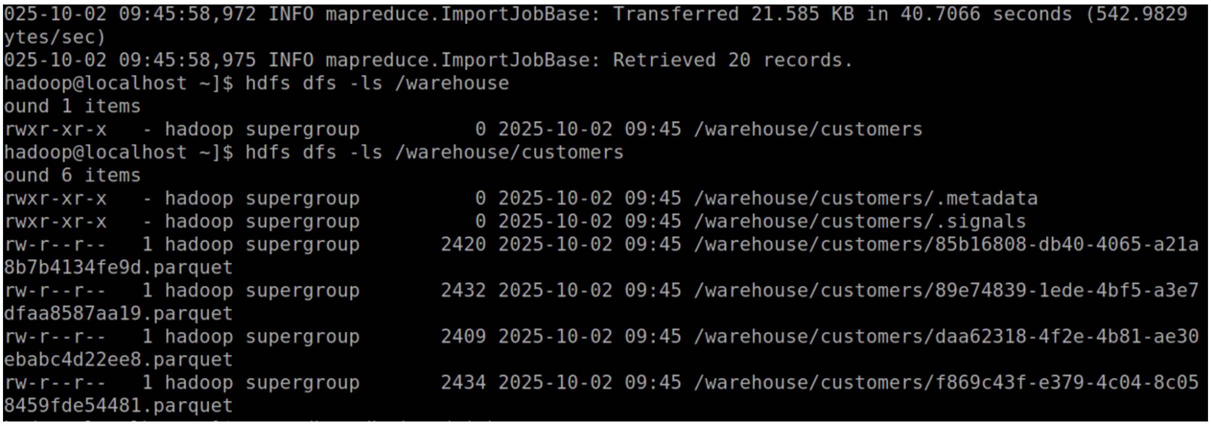
1. **Apache Scoop:** imports relational data (customer) from MariaDB into HDFS and Hive.



```
[student@localhost ~]$ mysql --user=student --password=student
Welcome to the MariaDB monitor.  Commands end with ; or \g.
Your MariaDB connection id is 42
Server version: 5.5.68-MariaDB MariaDB Server

Copyright (c) 2000, 2018, Oracle, MariaDB Corporation Ab and others.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

MariaDB [(none)]> create database SIC;
Query OK, 1 row affected (0.00 sec)

MariaDB [(none)]> use SIC;
Database changed
MariaDB [SIC]> CREATE TABLE customers (
    ->     customer_id INT PRIMARY KEY AUTO_INCREMENT,
    ->     first_name VARCHAR(50),
    ->     last_name VARCHAR(50),
    ->     age INT,
    ->     gender VARCHAR(10),
    ->     geography VARCHAR(50),
    ->     is_active TINYINT(1),
    ->     tenure_months INT,
    ->     date_opened DATE
    -> );
Query OK, 0 rows affected (0.01 sec)

MariaDB [SIC]> INSERT INTO customers (first_name, last_name, age, gender, geography, is_active, tenure_mo
nths, date_opened) VALUES
    -> ('Omar','Khalil',34,'Male','Egypt',1,24,'2022-01-15'),
    -> ('Sara','Hassan',29,'Female','UAE',0,12,'2023-01-20'),
    -> ('Ali','Mahmoud',41,'Male','Saudi Arabia',1,36,'2021-05-10'),
    -> ('Noura','Adel',27,'Female','Egypt',1,18,'2022-09-12'),
```

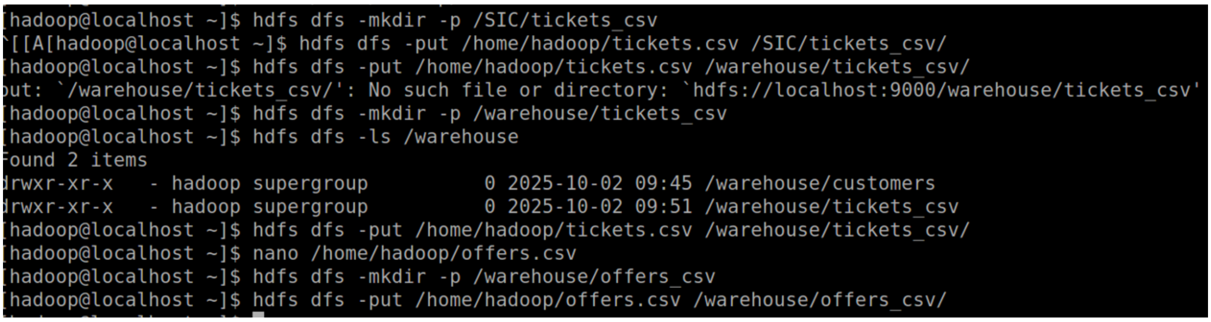


```
[hadoop@localhost ~]$ sqoop import \
>   --connect jdbc:mysql://localhost/SIC \
>   --username student \
>   --password student \
>   --table customers \
>   --as-parquetfile \
>   --target-dir /warehouse/customers
Warning: /usr/local/sqoop/sqoop-1.4.7/../hcatalog does not exist! HCatalog jobs will fail.
Please set $HCAT_HOME to the root of your HCatalog installation.
Warning: /usr/local/sqoop/sqoop-1.4.7/../accumulo does not exist! Accumulo imports will fail.
Please set $ACCUMULO_HOME to the root of your Accumulo installation.
Warning: /usr/local/sqoop/sqoop-1.4.7/../zookeeper does not exist! Accumulo imports will fail.
Please set $ZOOKEEPER_HOME to the root of your Zookeeper installation.
2025-10-02 09:45:10,908 INFO sqoop.Sqoop: Running Sqoop version: 1.4.7
2025-10-02 09:45:10,945 WARN tool.BaseSqoopTool: Setting your password on the command-line is insecure. C
onsider using -P instead.
2025-10-02 09:45:11,099 INFO manager.MySQLManager: Preparing to use a MySQL streaming resultset.
2025-10-02 09:45:11,099 INFO tool.CodeGenTool: Beginning code generation
2025-10-02 09:45:11,099 INFO tool.CodeGenTool: Will generate java class as codegen_customers
2025-10-02 09:45:11,648 INFO manager.SqlManager: Executing SQL statement: SELECT t.* FROM `customers` AS
t LIMIT 1
2025-10-02 09:45:11,684 INFO manager.SqlManager: Executing SQL statement: SELECT t.* FROM `customers` AS
t LIMIT 1
2025-10-02 09:45:11,690 INFO orm.CompilationManager: HADOOP_MAPRED_HOME is /home/hadoop/hadoop
Note: /tmp/sqoop-hadoop/compile/ce7d979a471e12b7472bd50b1e8fd815/codegen_customers.java uses or overrides
 a deprecated API.
Note: Recompile with -Xlint:deprecation for details.
2025-10-02 09:45:14,982 INFO orm.CompilationManager: Writing jar file: /tmp/sqoop-hadoop/compile/ce7d979a
471e12b7472bd50b1e8fd815/codegen_customers.jar
2025-10-02 09:45:15,000 WARN manager.MySQLManager: It looks like you are importing from mysql.
2025-10-02 09:45:15,000 WARN manager.MySQLManager: This transfer can be faster! Use the --direct
```



```
025-10-02 09:45:58,972 INFO mapreduce.ImportJobBase: Transferred 21.585 KB in 40.7066 seconds (542.9829
ytes/sec)
025-10-02 09:45:58,975 INFO mapreduce.ImportJobBase: Retrieved 20 records.
hadoop@localhost ~]$ hdfs dfs -ls /warehouse
ound 1 items
rwxr-xr-x   - hadoop supergroup          0 2025-10-02 09:45 /warehouse/customers
hadoop@localhost ~]$ hdfs dfs -ls /warehouse/customers
ound 6 items
rwxr-xr-x   - hadoop supergroup          0 2025-10-02 09:45 /warehouse/customers/.metadata
rwxr-xr-x   - hadoop supergroup          0 2025-10-02 09:45 /warehouse/customers/.signals
rw-r--r--   1 hadoop supergroup       2420 2025-10-02 09:45 /warehouse/customers/85b16808-db40-4065-a21a
8b7b4134fe9d.parquet
rw-r--r--   1 hadoop supergroup       2432 2025-10-02 09:45 /warehouse/customers/89e74839-1ede-4bf5-a3e7
dfaa8587aa19.parquet
rw-r--r--   1 hadoop supergroup       2409 2025-10-02 09:45 /warehouse/customers/daa62318-4f2e-4b81-ae30
ebabc4d22ee8.parquet
rw-r--r--   1 hadoop supergroup       2434 2025-10-02 09:45 /warehouse/customers/f869c43f-e379-4c04-8c05
8459fde54481.parquet
```

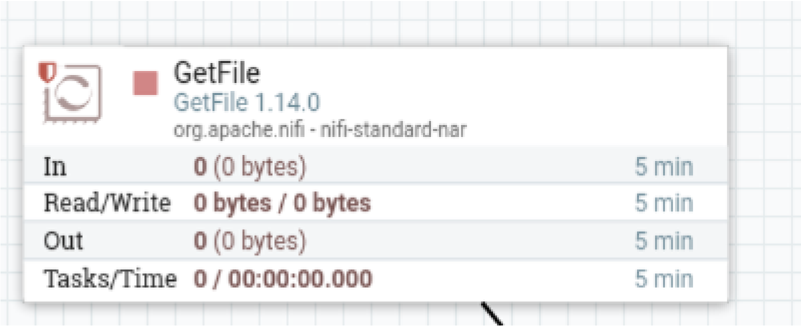
-loading csv files to HDFS:



```
[hadoop@localhost ~]$ hdfs dfs -mkdir -p /SIC/tickets_csv
`[[A[hadoop@localhost ~]$ hdfs dfs -put /home/hadoop/tickets.csv /SIC/tickets_csv/
[hadoop@localhost ~]$ hdfs dfs -put /home/hadoop/tickets.csv /warehouse/tickets_csv/
put: `/warehouse/tickets_csv/': No such file or directory: `hdfs://localhost:9000/warehouse/tickets_csv'
[hadoop@localhost ~]$ hdfs dfs -mkdir -p /warehouse/tickets_csv
[hadoop@localhost ~]$ hdfs dfs -ls /warehouse
Found 2 items
drwxr-xr-x   - hadoop supergroup          0 2025-10-02 09:45 /warehouse/customers
drwxr-xr-x   - hadoop supergroup          0 2025-10-02 09:51 /warehouse/tickets_csv
[hadoop@localhost ~]$ hdfs dfs -put /home/hadoop/tickets.csv /warehouse/tickets_csv/
[hadoop@localhost ~]$ nano /home/hadoop/offers.csv
[hadoop@localhost ~]$ hdfs dfs -mkdir -p /warehouse/offers_csv
[hadoop@localhost ~]$ hdfs dfs -put /home/hadoop/offers.csv /warehouse/offers_csv/
```

2. **Apache NiFi:** ingests semi-structured / unstructured data (JSON) and pushes into HDFS.

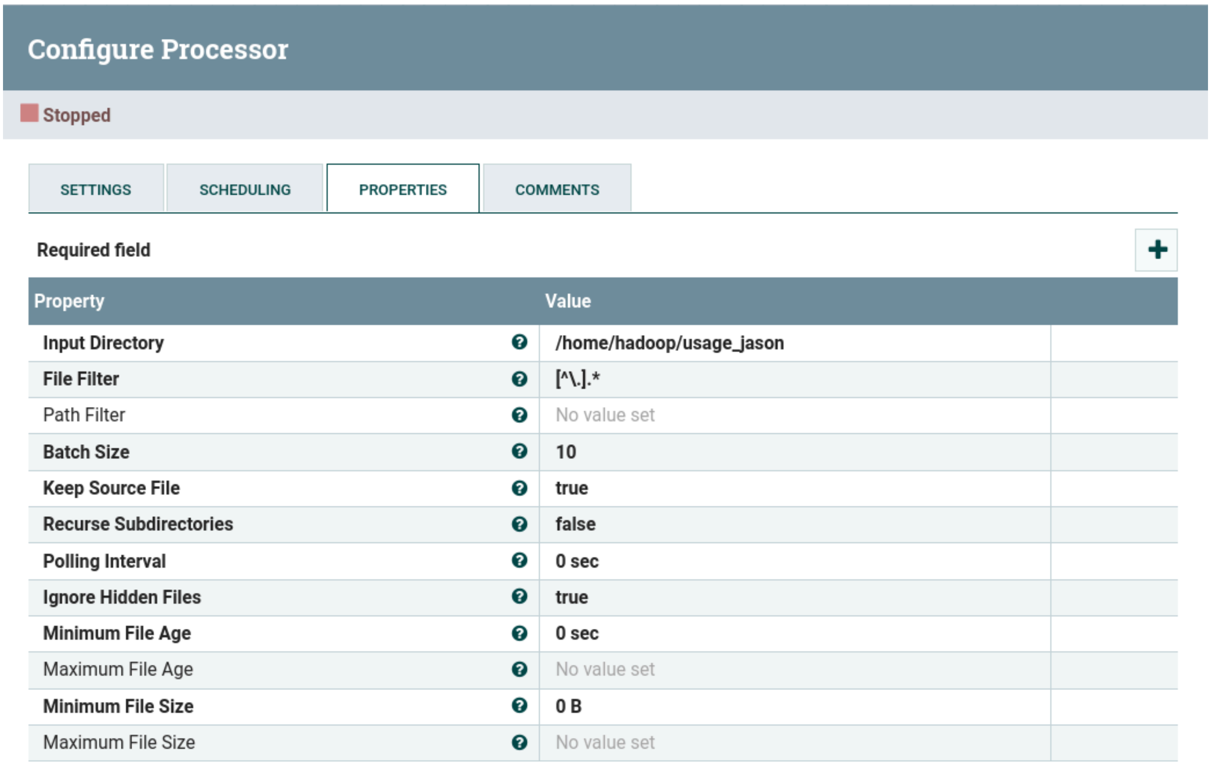
-GetFile Processor:



GetFile
GetFile 1.14.0
org.apache.nifi - nifi-standard-nar

In	0 (0 bytes)	5 min
Read/Write	0 bytes / 0 bytes	5 min
Out	0 (0 bytes)	5 min
Tasks/Time	0 / 00:00:00.000	5 min

-Configuration:



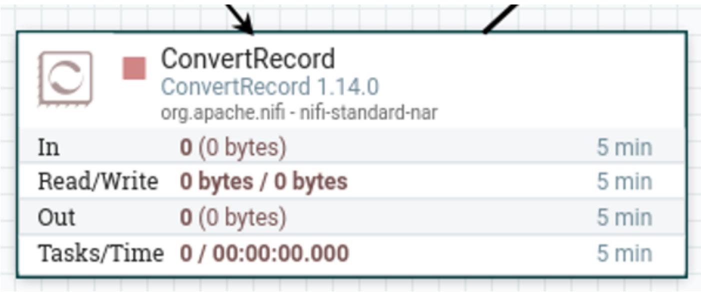
Configure Processor

Stopped

Required field

Property	Value
Input Directory	/home/hadoop/usage_jason
File Filter	[^\.].*
Path Filter	No value set
Batch Size	10
Keep Source File	true
Recurse Subdirectories	false
Polling Interval	0 sec
Ignore Hidden Files	true
Minimum File Age	0 sec
Maximum File Age	No value set
Minimum File Size	0 B
Maximum File Size	No value set

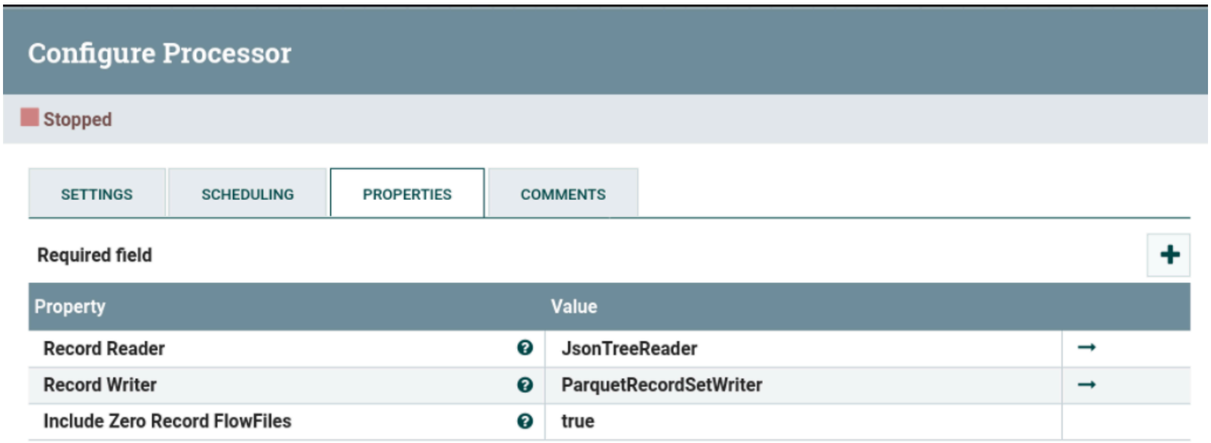
-ConvertRecord Processor:



ConvertRecord
ConvertRecord 1.14.0
org.apache.nifi - nifi-standard-nar

In	0 (0 bytes)	5 min
Read/Write	0 bytes / 0 bytes	5 min
Out	0 (0 bytes)	5 min
Tasks/Time	0 / 00:00:00.000	5 min

-Configuration:



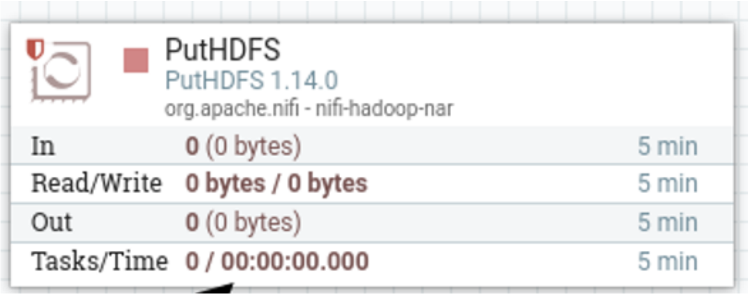
Configure Processor

🟥 Stopped

Required field

Property	Value
Record Reader	JsonTreeReader
Record Writer	ParquetRecordSetWriter
Include Zero Record FlowFiles	true

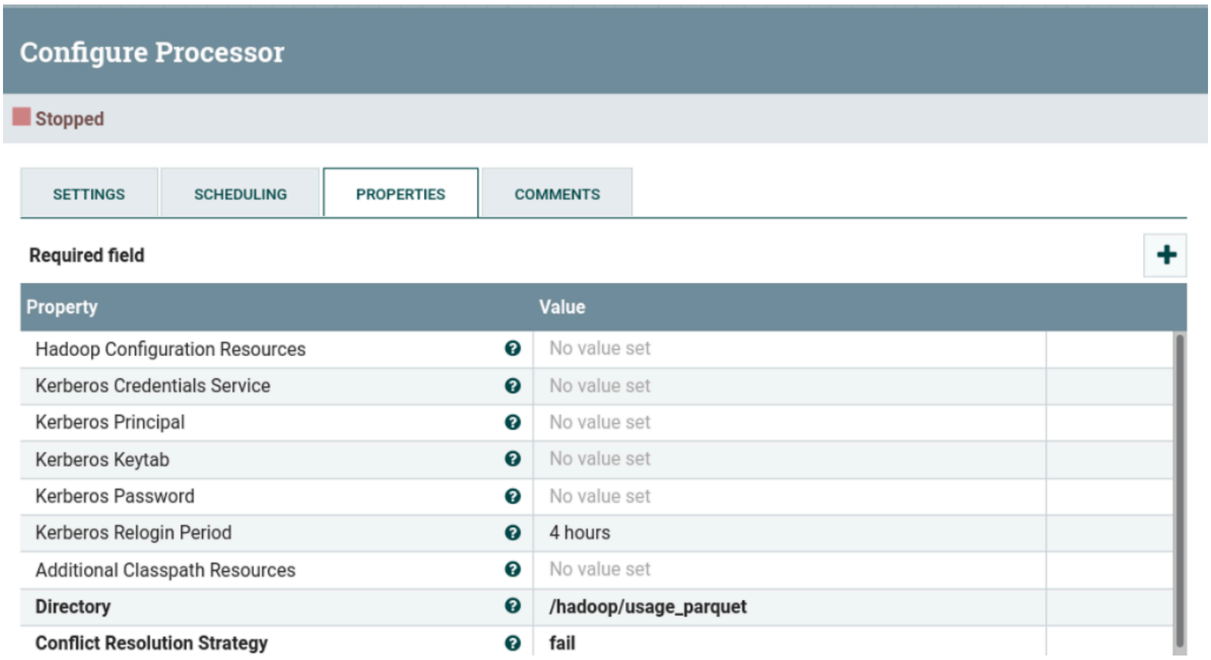
-PutHDFS Processor:



PutHDFS
PutHDFS 1.14.0
org.apache.nifi - nifi-hadoop-nar

In	0 (0 bytes)	5 min
Read/Write	0 bytes / 0 bytes	5 min
Out	0 (0 bytes)	5 min
Tasks/Time	0 / 00:00:00.000	5 min

-Configuration



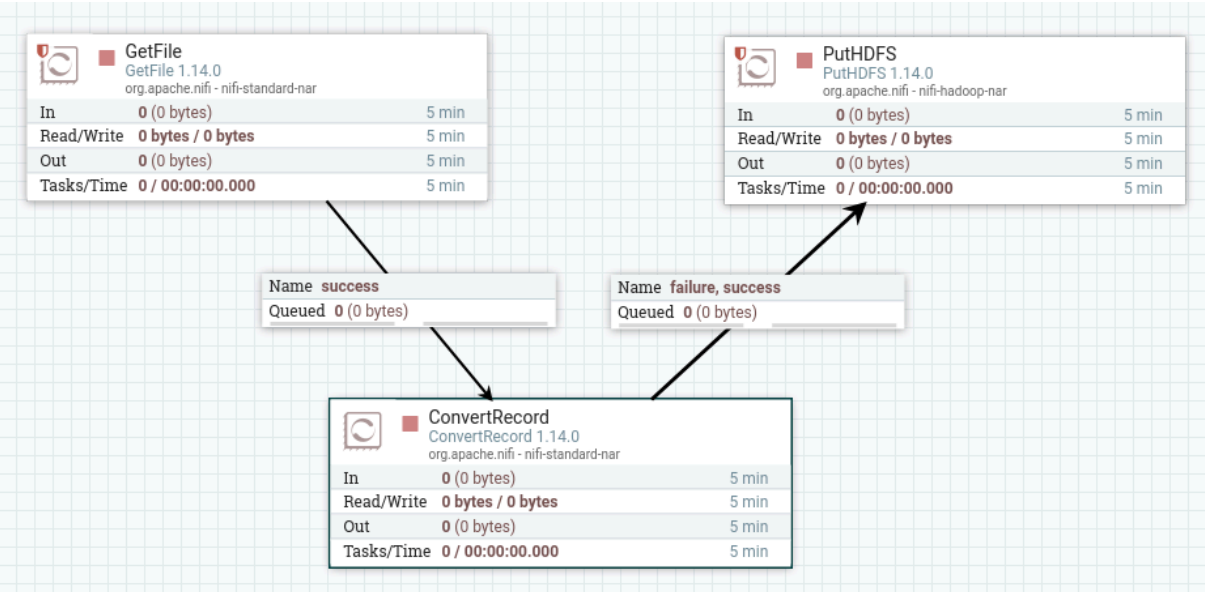
Configure Processor

Stopped

Required field

Property	Value
Hadoop Configuration Resources	No value set
Kerberos Credentials Service	No value set
Kerberos Principal	No value set
Kerberos Keytab	No value set
Kerberos Password	No value set
Kerberos Relogin Period	4 hours
Additional Classpath Resources	No value set
Directory	**/hadoop/usage_parquet**
Conflict Resolution Strategy	**fail**

-Full Connection:



GetFile
GetFile 1.14.0
org.apache.nifi - nifi-standard-nar

In	0 (0 bytes)	5 min
Read/Write	0 bytes / 0 bytes	5 min
Out	0 (0 bytes)	5 min
Tasks/Time	0 / 00:00:00.000	5 min

PutHDFS
PutHDFS 1.14.0
org.apache.nifi - nifi-hadoop-nar

In	0 (0 bytes)	5 min
Read/Write	0 bytes / 0 bytes	5 min
Out	0 (0 bytes)	5 min
Tasks/Time	0 / 00:00:00.000	5 min

Name success
Queued 0 (0 bytes)

Name failure, success
Queued 0 (0 bytes)

ConvertRecord
ConvertRecord 1.14.0
org.apache.nifi - nifi-standard-nar

In	0 (0 bytes)	5 min
Read/Write	0 bytes / 0 bytes	5 min
Out	0 (0 bytes)	5 min
Tasks/Time	0 / 00:00:00.000	5 min

4. Mapping External Hive tables to HDFS files:

1. Customers Table:

```
CREATE EXTERNAL TABLE customers (
    customer_id INT,
    first_name STRING,
    last_name STRING,
    age INT,
    gender STRING,
    geography STRING,
    is_active BOOLEAN,
    tenure_months INT,
    date_opened DATE
)
STORED AS PARQUET
LOCATION '/SIC/customers/';
```

2. Customer_support_ticket Table:

```
CREATE EXTERNAL TABLE customer_support_tickets (
    ticket_id INT,
    customer_id INT,
    issue_type STRING,
    severity STRING,
    resolution_time_hrs INT
)
ROW FORMAT DELIMITED
FIELDS TERMINATED BY ','
STORED AS TEXTFILE
LOCATION '/SIC/customer_support_tickets/';
```

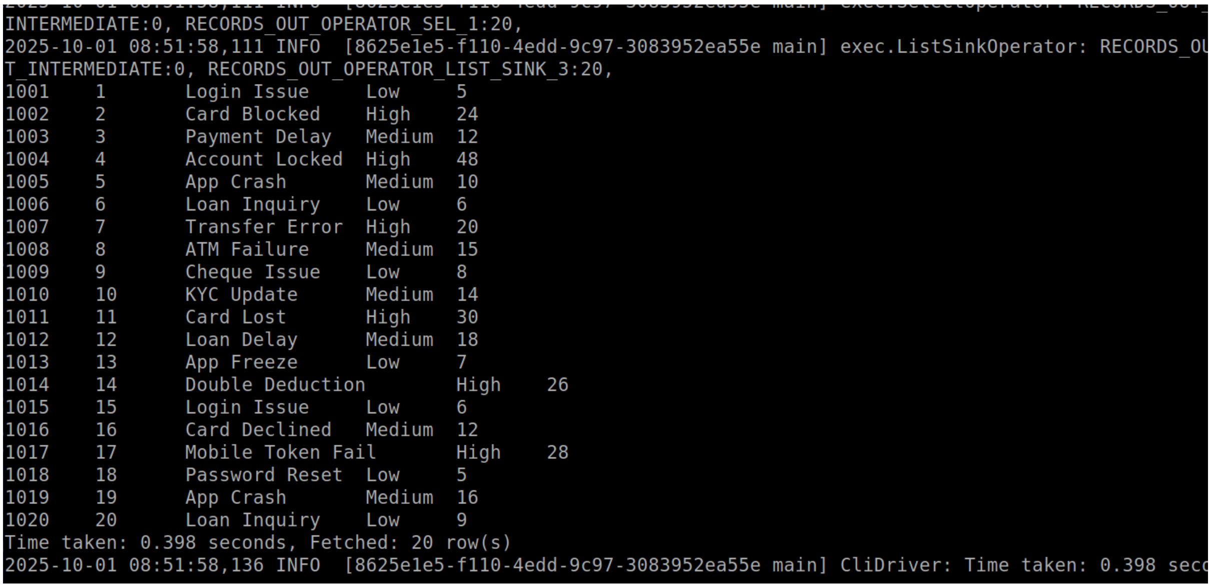
3. **Offers Table:**

```sql
CREATE EXTERNAL TABLE offers (
    offer_id INT,
    customer_id INT,
    offer_type STRING,
    accepted INT,
    date_offered DATE
)
ROW FORMAT DELIMITED
FIELDS TERMINATED BY ','
STORED AS TEXTFILE
LOCATION '/SIC/offers/';
```

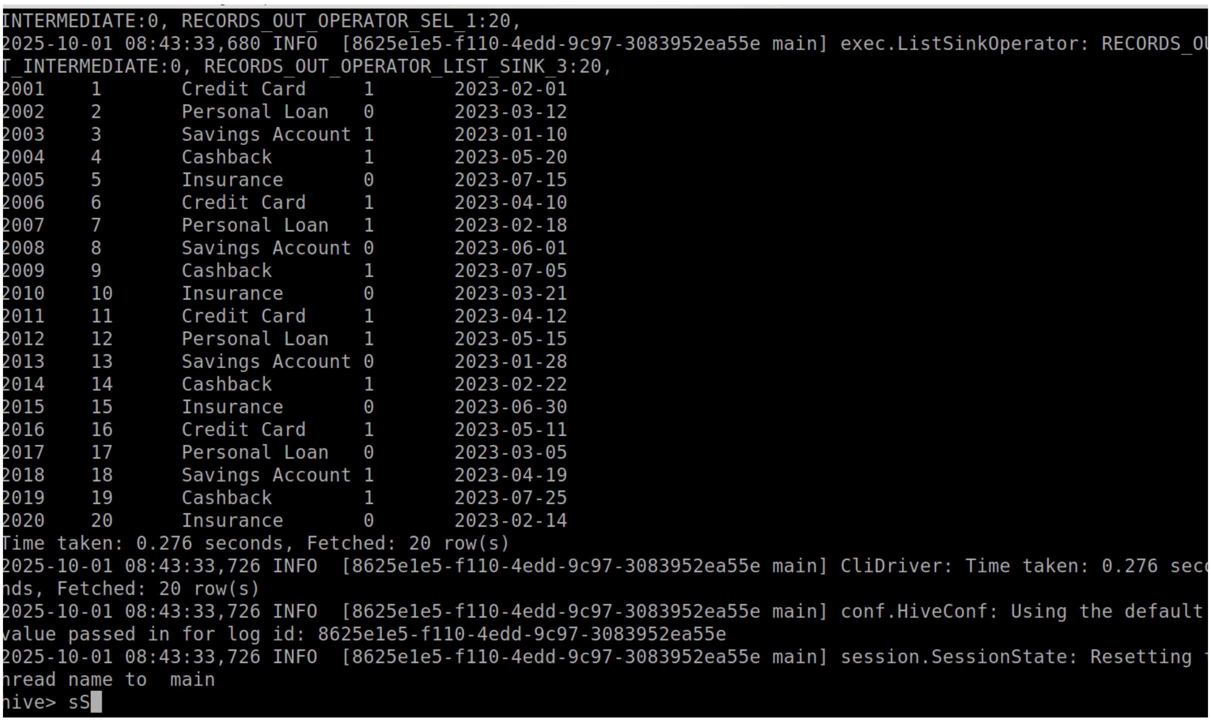
4. **Usage Table:**

```sql
 CREATE EXTERNAL TABLE usage (
     usage_log_id INT,
     customer_id INT,
     product_type STRING,
     monthly_balance INT,
     num_products INT
 )
 STORED AS PARQUET
 LOCATION '/SIC/usage/';
```

-Example of how data looks inside hive after mapping tables to HDFS files:



```
2025-10-01 08:51:58,111 INFO  [8625e1e5-f110-4edd-9c97-3083952ea55e main] exec.SelectOperator: RECORDS_OUT_
INTERMEDIATE:0, RECORDS_OUT_OPERATOR_SEL_1:20,
2025-10-01 08:51:58,111 INFO  [8625e1e5-f110-4edd-9c97-3083952ea55e main] exec.ListSinkOperator: RECORDS_OU
T_INTERMEDIATE:0, RECORDS_OUT_OPERATOR_LIST_SINK_3:20,
1001    1       Login Issue     Low     5
1002    2       Card Blocked    High    24
1003    3       Payment Delay   Medium  12
1004    4       Account Locked  High    48
1005    5       App Crash       Medium  10
1006    6       Loan Inquiry    Low     6
1007    7       Transfer Error  High    20
1008    8       ATM Failure     Medium  15
1009    9       Cheque Issue    Low     8
1010    10      KYC Update      Medium  14
1011    11      Card Lost       High    30
1012    12      Loan Delay      Medium  18
1013    13      App Freeze      Low     7
1014    14      Double Deduction        High    26
1015    15      Login Issue     Low     6
1016    16      Card Declined   Medium  12
1017    17      Mobile Token Fail       High    28
1018    18      Password Reset  Low     5
1019    19      App Crash       Medium  16
1020    20      Loan Inquiry    Low     9
Time taken: 0.398 seconds, Fetched: 20 row(s)
2025-10-01 08:51:58,136 INFO  [8625e1e5-f110-4edd-9c97-3083952ea55e main] CliDriver: Time taken: 0.398 seco
```



```
INTERMEDIATE:0, RECORDS_OUT_OPERATOR_SEL_1:20,
2025-10-01 08:43:33,680 INFO  [8625e1e5-f110-4edd-9c97-3083952ea55e main] exec.ListSinkOperator: RECORDS_OU
T_INTERMEDIATE:0, RECORDS_OUT_OPERATOR_LIST_SINK_3:20,
2001    1       Credit Card     1       2023-02-01
2002    2       Personal Loan   0       2023-03-12
2003    3       Savings Account 1       2023-01-10
2004    4       Cashback        1       2023-05-20
2005    5       Insurance       0       2023-07-15
2006    6       Credit Card     1       2023-04-10
2007    7       Personal Loan   1       2023-02-18
2008    8       Savings Account 0       2023-06-01
2009    9       Cashback        1       2023-07-05
2010    10      Insurance       0       2023-03-21
2011    11      Credit Card     1       2023-04-12
2012    12      Personal Loan   1       2023-05-15
2013    13      Savings Account 0       2023-01-28
2014    14      Cashback        1       2023-02-22
2015    15      Insurance       0       2023-06-30
2016    16      Credit Card     1       2023-05-11
2017    17      Personal Loan   0       2023-03-05
2018    18      Savings Account 1       2023-04-19
2019    19      Cashback        1       2023-07-25
2020    20      Insurance       0       2023-02-14
Time taken: 0.276 seconds, Fetched: 20 row(s)
2025-10-01 08:43:33,726 INFO  [8625e1e5-f110-4edd-9c97-3083952ea55e main] CliDriver: Time taken: 0.276 seco
nds, Fetched: 20 row(s)
2025-10-01 08:43:33,726 INFO  [8625e1e5-f110-4edd-9c97-3083952ea55e main] conf.HiveConf: Using the default
value passed in for log id: 8625e1e5-f110-4edd-9c97-3083952ea55e
2025-10-01 08:43:33,726 INFO  [8625e1e5-f110-4edd-9c97-3083952ea55e main] session.SessionState: Resetting t
hread name to  main
hive> sS
```

4. Data Warehouse and Modeling in Hive

The data warehouse is modeled using a star schema.

❖ **Fact Tables:**

➢ `fact_customer_activity` → contains key customer activity metrics (transactions, usage, churn indicators).

➢ `fact_customer_metrics_date` → captures customer-level performance metrics such as churn status, customer lifetime value (CLV), average ticket resolution time, number of products, and tracking dates (`dw_start_date`, `dw_end_date`) for historical changes.

❖ **Dimension Tables:**

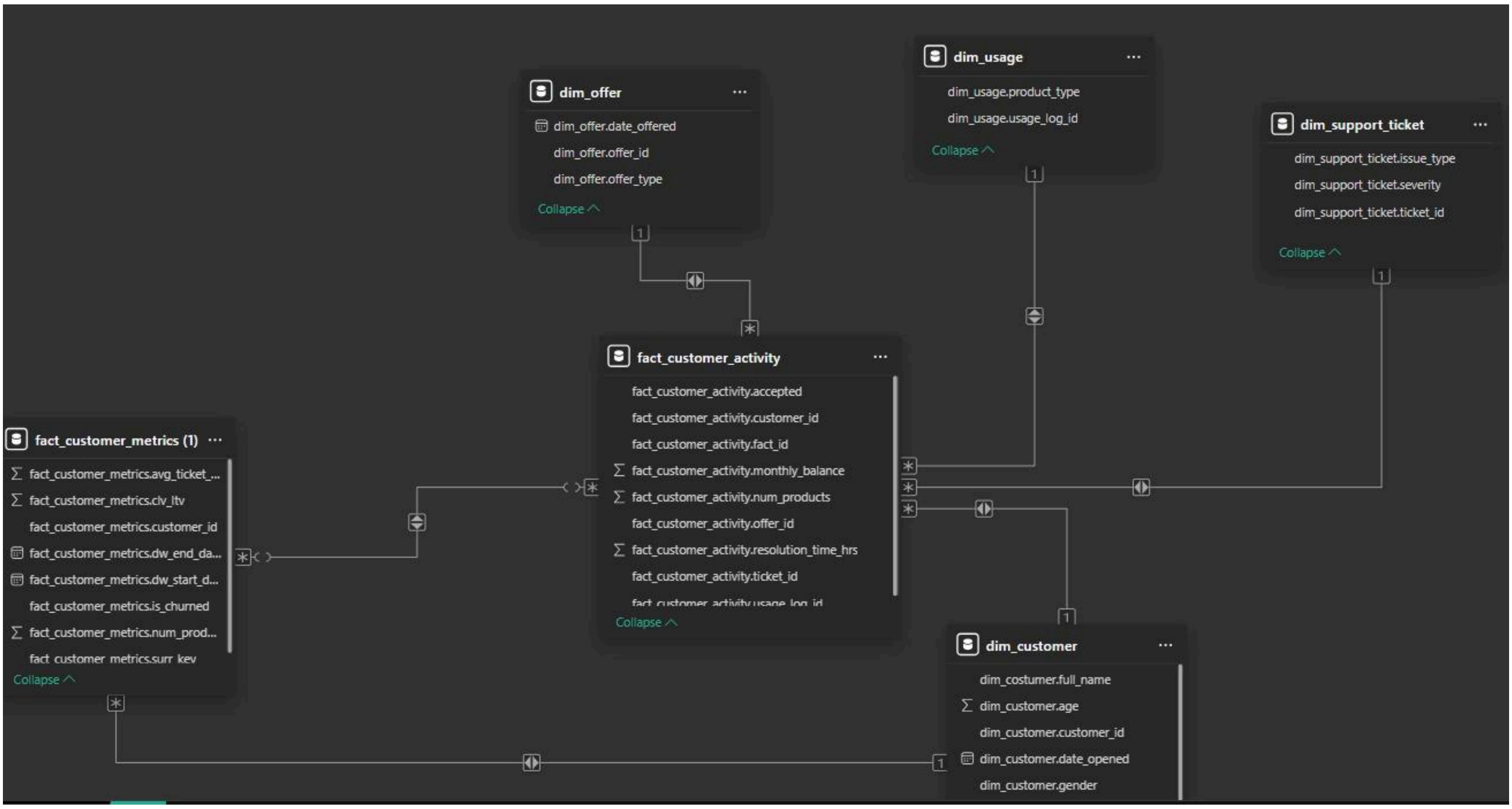
➢ `dim_customer` → customer details, demographics, churn flag.

➢ `dim_usage` → aggregated usage per period.

➢ `dim_offer` → marketing campaigns and offers.

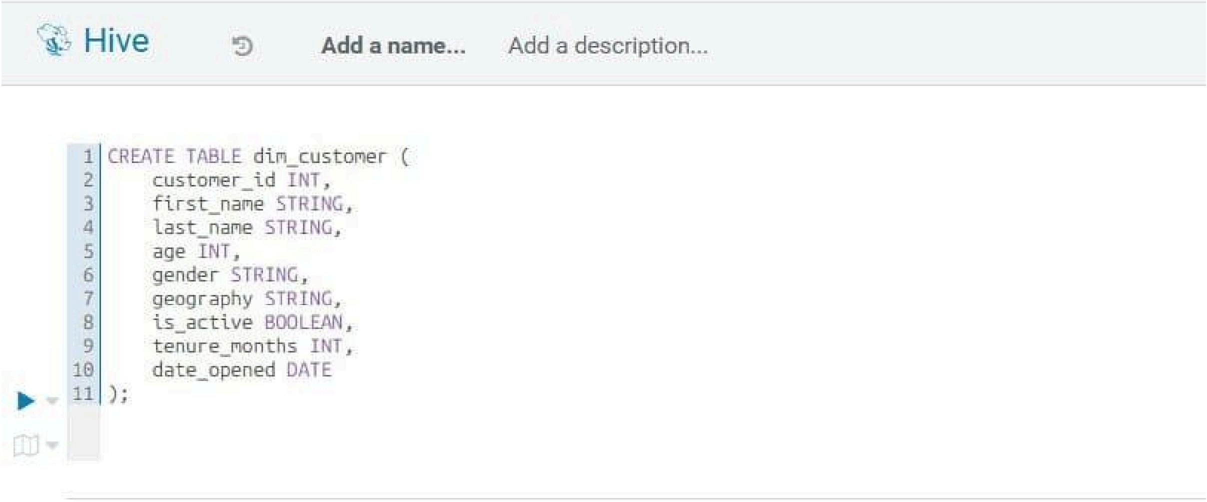
➢ `dim_support_ticket` → customer support interactions.

-Star Schema Diagram:


dim_usage
dim_usage.product_type
dim_usage.usage_log_id
Collapse
dim_offer
dim_offer.date_offered
dim_offer.offer_id
dim_offer.offer_type
Collapse
dim_support_ticket
dim_support_ticket.issue_type
dim_support_ticket.severity
dim_support_ticket.ticket_id
Collapse
fact_customer_activity
fact_customer_activity.accepted
fact_customer_activity.customer_id
fact_customer_activity.fact_id
fact_customer_activity.monthly_balance
fact_customer_activity.num_products
fact_customer_activity.offer_id
fact_customer_activity.resolution_time_hrs
fact_customer_activity.ticket_id
fact_customer_activity.usage_log_id
Collapse
fact_customer_metrics (1)
fact_customer_metrics.avg_ticket_...
fact_customer_metrics.clv_ltv
fact_customer_metrics.customer_id
fact_customer_metrics.dw_end_da...
fact_customer_metrics.dw_start_d...
fact_customer_metrics.is_churned
fact_customer_metrics.num_prod...
fact_customer_metrics.surr_key
Collapse
dim_customer
dim_costumer.full_name
dim_customer.age
dim_customer.customer_id
dim_customer.date_opened
dim_customer.gender
1
*

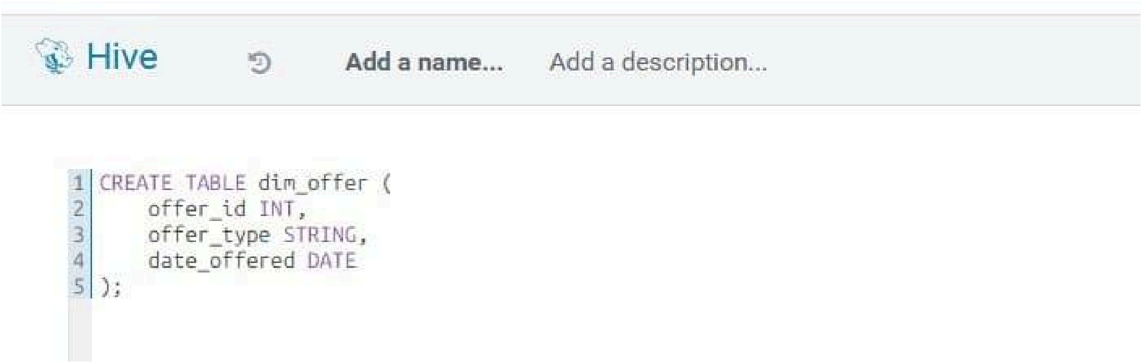
-Hive Queries:

1. **Table dim_customer:**



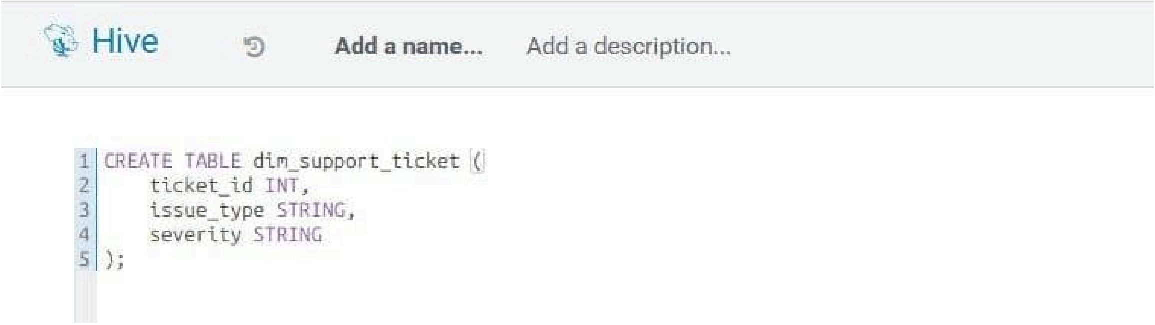
```
CREATE TABLE dim_customer (
    customer_id INT,
    first_name STRING,
    last_name STRING,
    age INT,
    gender STRING,
    geography STRING,
    is_active BOOLEAN,
    tenure_months INT,
    date_opened DATE
);
```

2. **Table dim_offer:**



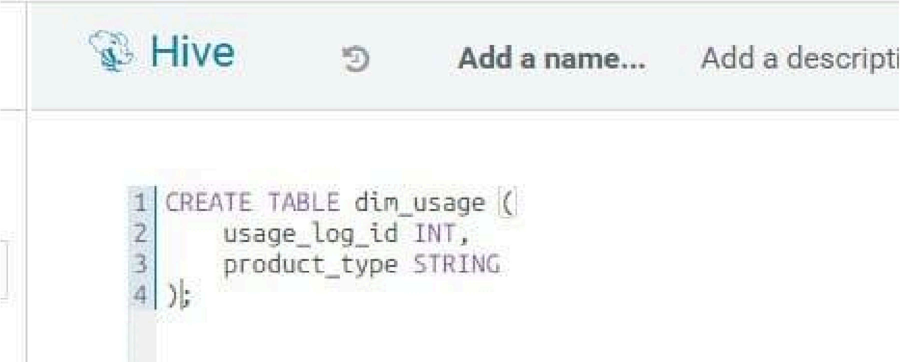
```
CREATE TABLE dim_offer (
    offer_id INT,
    offer_type STRING,
    date_offered DATE
);
```

3. **Table dim_support_ticket:**



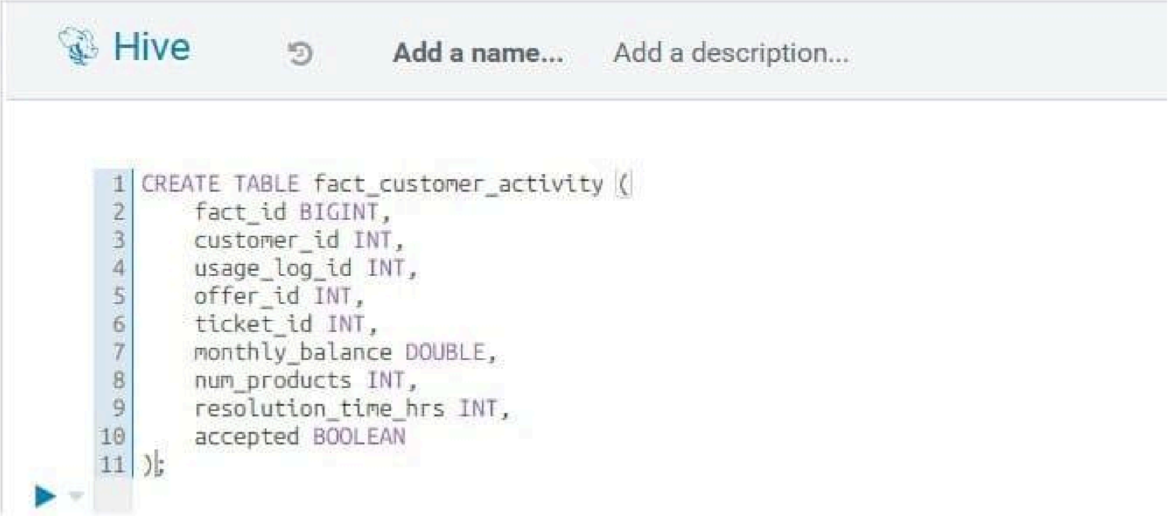
```
CREATE TABLE dim_support_ticket (
    ticket_id INT,
    issue_type STRING,
    severity STRING
);
```

4. Table dim_usage:



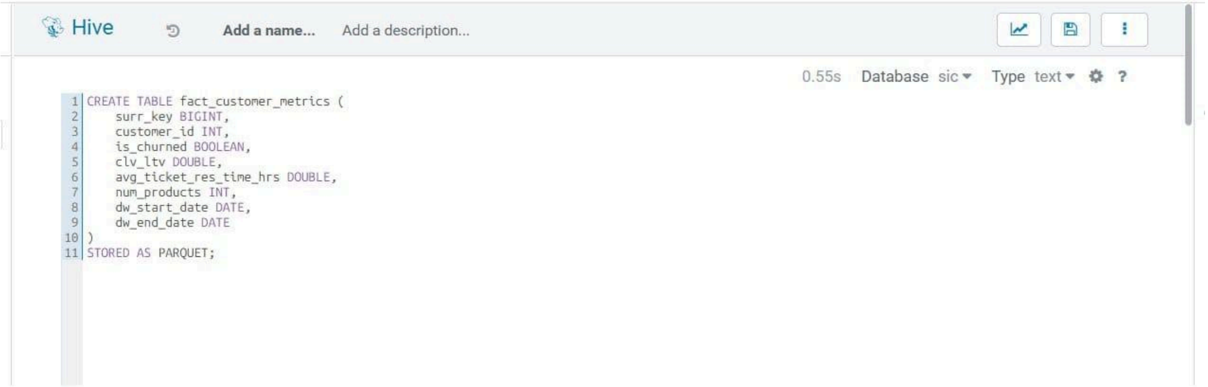
```
CREATE TABLE dim_usage (
    usage_log_id INT,
    product_type STRING
);
```

5. Table fact_customer_activity:



```
CREATE TABLE fact_customer_activity (
    fact_id BIGINT,
    customer_id INT,
    usage_log_id INT,
    offer_id INT,
    ticket_id INT,
    monthly_balance DOUBLE,
    num_products INT,
    resolution_time_hrs INT,
    accepted BOOLEAN
);
```

6. Table fact_customer_metrics:



0.55s Database sic Type text

```
CREATE TABLE fact_customer_metrics (
    surr_key BIGINT,
    customer_id INT,
    is_churned BOOLEAN,
    clv_ltv DOUBLE,
    avg_ticket_res_time_hrs DOUBLE,
    num_products INT,
    dw_start_date DATE,
    dw_end_date DATE
)
STORED AS PARQUET;
```

-We designed two different fact tables to serve distinct analytical purposes:

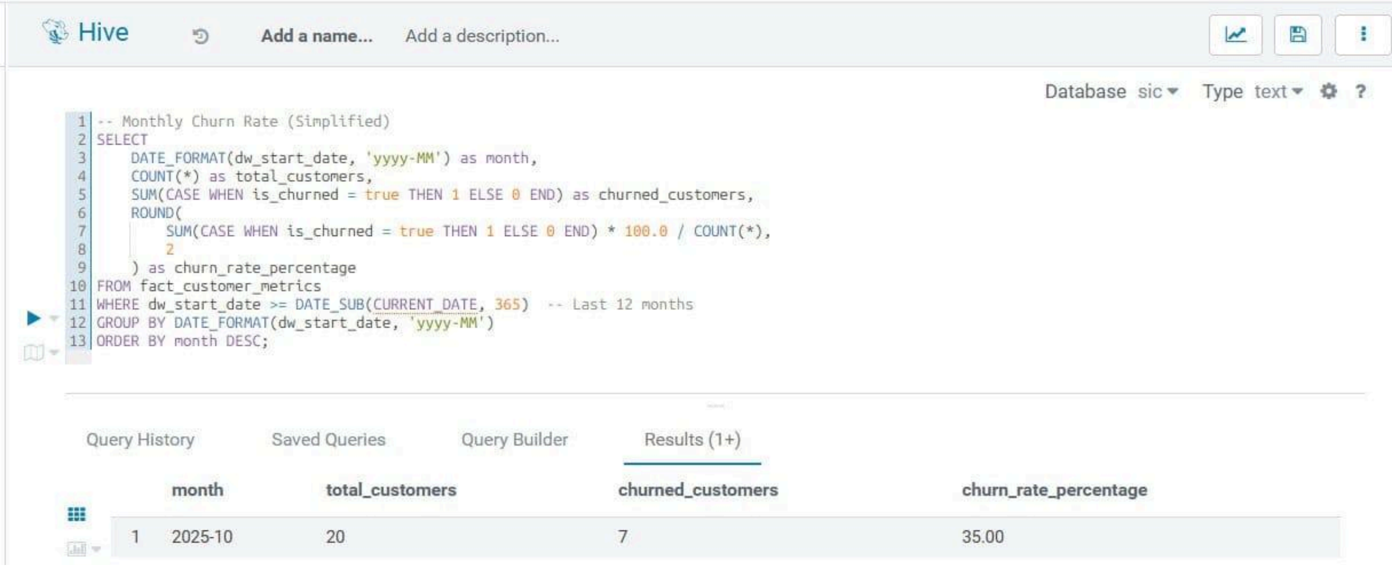
- **Fact Customer Activity Table**
  This table captures detailed records of individual customer actions. It includes information such as monthly balances, product usage, support tickets, and offers. The purpose of this table is to provide a granular, event-level view of customer interactions, similar to a detailed log or diary of all activities.

- **Fact Customer Metrics Table**
  This table focuses on customer-level summaries and insights derived from the raw activities. It contains calculated metrics such as churn status, customer lifetime value (CLV), and average ticket resolution time. Its role is to provide a consolidated, high-level view of customer performance and risk, similar to a report card.

5. Analysis with Hive (Hue)

Originally, the plan was to use PySpark for advanced analysis, but due to setup issues we performed all the analysis directly in Hive using Hue.

1.Overall customer churn rate for the latest month:

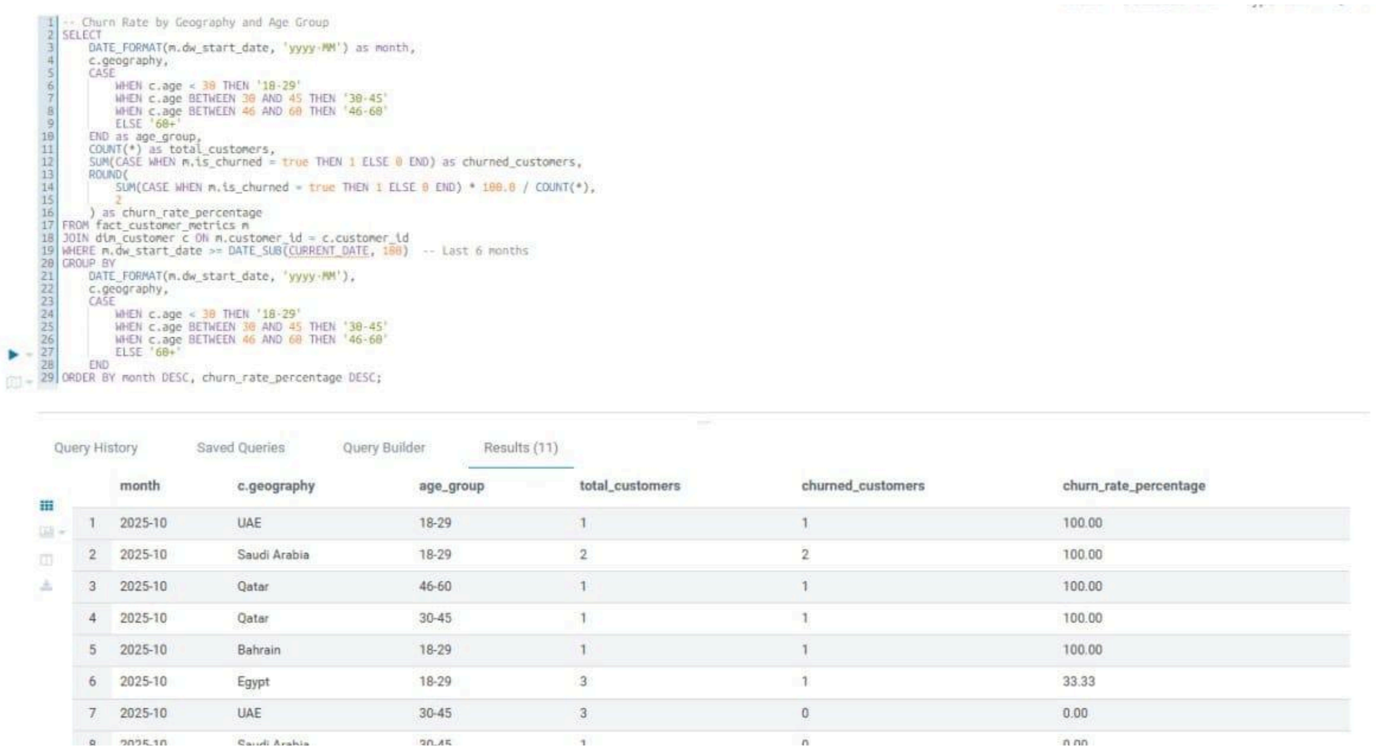


```sql
-- Monthly Churn Rate (Simplified)
SELECT
    DATE_FORMAT(dw_start_date, 'yyyy-MM') as month,
    COUNT(*) as total_customers,
    SUM(CASE WHEN is_churned = true THEN 1 ELSE 0 END) as churned_customers,
    ROUND(
        SUM(CASE WHEN is_churned = true THEN 1 ELSE 0 END) * 100.0 / COUNT(*),
        2
    ) as churn_rate_percentage
FROM fact_customer_metrics
WHERE dw_start_date >= DATE_SUB(CURRENT_DATE, 365)  -- Last 12 months
GROUP BY DATE_FORMAT(dw_start_date, 'yyyy-MM')
ORDER BY month DESC;
```

	month	total_customers	churned_customers	churn_rate_percentage
1	2025-10	20	7	35.00

- **Key Insight:** Our churn rate is critically high. At 35%, we are losing more than one-third of our customers each month. This is an unsustainable business emergency that requires immediate, company-wide action.

2.Churn rate by geography and age group:

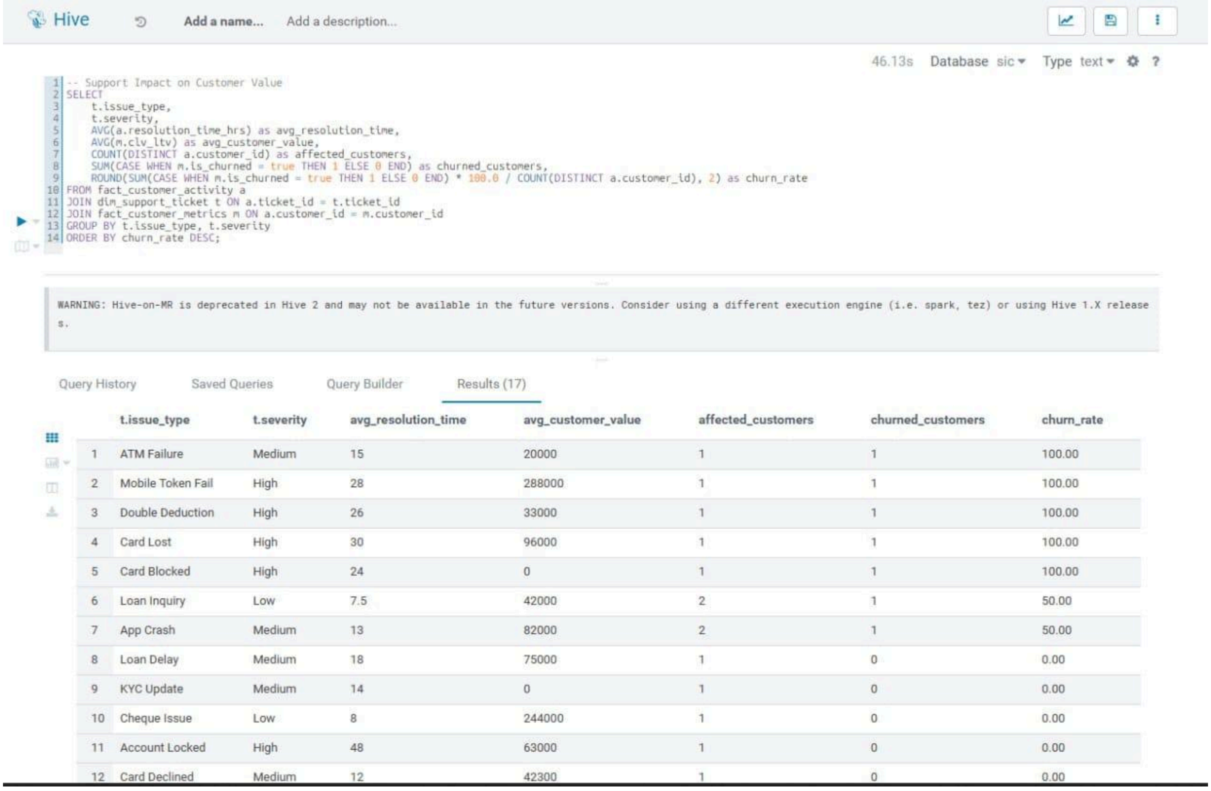


```
-- Churn Rate by Geography and Age Group
SELECT
    DATE_FORMAT(m.dw_start_date, 'yyyy-MM') as month,
    c.geography,
    CASE
        WHEN c.age < 30 THEN '18-29'
        WHEN c.age BETWEEN 30 AND 45 THEN '30-45'
        WHEN c.age BETWEEN 46 AND 60 THEN '46-60'
        ELSE '60+'
    END as age_group,
    COUNT(*) as total_customers,
    SUM(CASE WHEN m.is_churned = true THEN 1 ELSE 0 END) as churned_customers,
    ROUND(
        SUM(CASE WHEN m.is_churned = true THEN 1 ELSE 0 END) * 100.0 / COUNT(*),
        2
    ) as churn_rate_percentage
FROM fact_customer_metrics m
JOIN dim_customer c ON m.customer_id = c.customer_id
WHERE m.dw_start_date >= DATE_SUB(CURRENT_DATE, 180)  -- Last 6 months
GROUP BY
    DATE_FORMAT(m.dw_start_date, 'yyyy-MM'),
    c.geography,
    CASE
        WHEN c.age < 30 THEN '18-29'
        WHEN c.age BETWEEN 30 AND 45 THEN '30-45'
        WHEN c.age BETWEEN 46 AND 60 THEN '46-60'
        ELSE '60+'
    END
ORDER BY month DESC, churn_rate_percentage DESC;
```

	month	c.geography	age_group	total_customers	churned_customers	churn_rate_percentage
1	2025-10	UAE	18-29	1	1	100.00
2	2025-10	Saudi Arabia	18-29	2	2	100.00
3	2025-10	Qatar	46-60	1	1	100.00
4	2025-10	Qatar	30-45	1	1	100.00
5	2025-10	Bahrain	18-29	1	1	100.00
6	2025-10	Egypt	18-29	3	1	33.33
7	2025-10	UAE	30-45	3	0	0.00
8	2025-10	Saudi Arabia	30-45	1	0	0.00

- **Key Insight:** Churn is concentrated in younger customers (18-29) across all geographies. In several regions, 100% of our young customers churned last month, indicating a critical failure to retain the next generation of clients.

3.Support Impact on Customer Value:



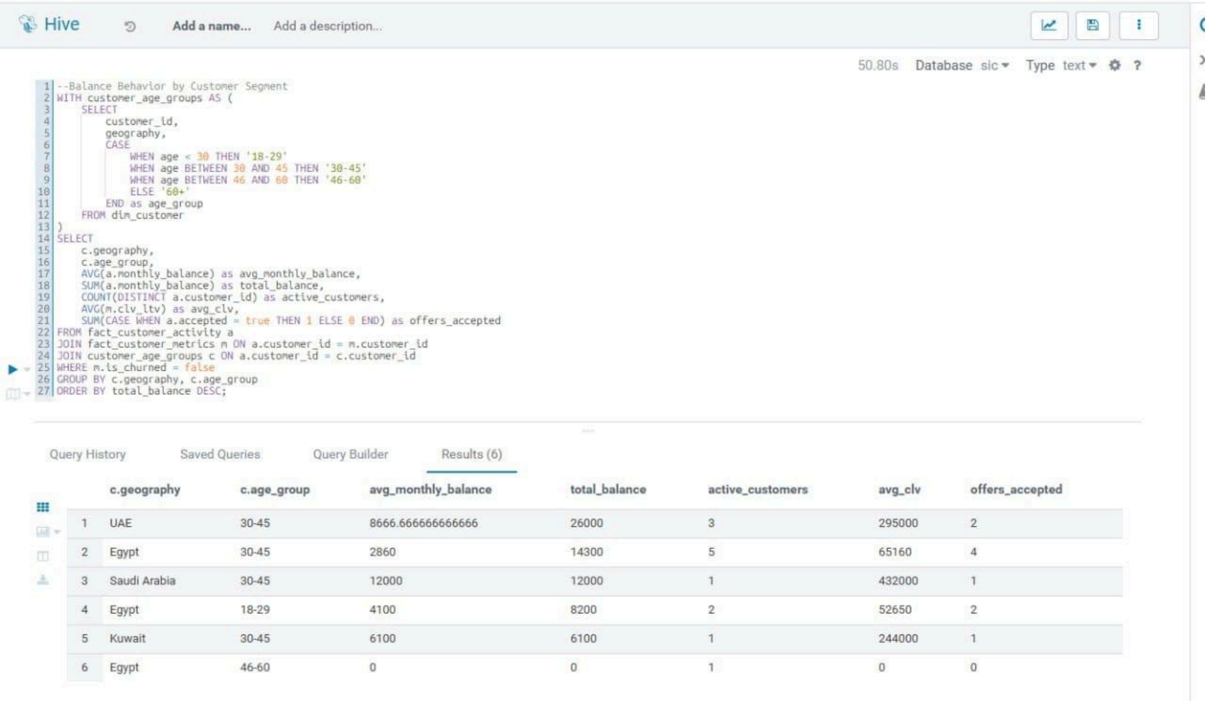
```sql
-- Support Impact on Customer Value
SELECT
    t.issue_type,
    t.severity,
    AVG(a.resolution_time_hrs) as avg_resolution_time,
    AVG(m.clv_ltv) as avg_customer_value,
    COUNT(DISTINCT a.customer_id) as affected_customers,
    SUM(CASE WHEN m.is_churned = true THEN 1 ELSE 0 END) as churned_customers,
    ROUND(SUM(CASE WHEN m.is_churned = true THEN 1 ELSE 0 END) * 100.0 / COUNT(DISTINCT a.customer_id), 2) as churn_rate
FROM fact_customer_activity a
JOIN dim_support_ticket t ON a.ticket_id = t.ticket_id
JOIN fact_customer_metrics m ON a.customer_id = m.customer_id
GROUP BY t.issue_type, t.severity
ORDER BY churn_rate DESC;
```

```
WARNING: Hive-on-MR is deprecated in Hive 2 and may not be available in the future versions. Consider using a different execution engine (i.e. spark, tez) or using Hive 1.X release
s.
```

	t.issue_type	t.severity	avg_resolution_time	avg_customer_value	affected_customers	churned_customers	churn_rate
1	ATM Failure	Medium	15	20000	1	1	100.00
2	Mobile Token Fail	High	28	288000	1	1	100.00
3	Double Deduction	High	26	33000	1	1	100.00
4	Card Lost	High	30	96000	1	1	100.00
5	Card Blocked	High	24	0	1	1	100.00
6	Loan Inquiry	Low	7.5	42000	2	1	50.00
7	App Crash	Medium	13	82000	2	1	50.00
8	Loan Delay	Medium	18	75000	1	0	0.00
9	KYC Update	Medium	14	0	1	0	0.00
10	Cheque Issue	Low	8	244000	1	0	0.00
11	Account Locked	High	48	63000	1	0	0.00
12	Card Declined	Medium	12	42300	1	0	0.00

- **Key Insight:** High-severity technical and financial issues are a primary driver of churn. Problems like "ATM Failure," "Mobile Token Fail," and "Double Deduction" have a 100% churn rate, directly destroying customer value.

4.Balance behaviour by customer segment

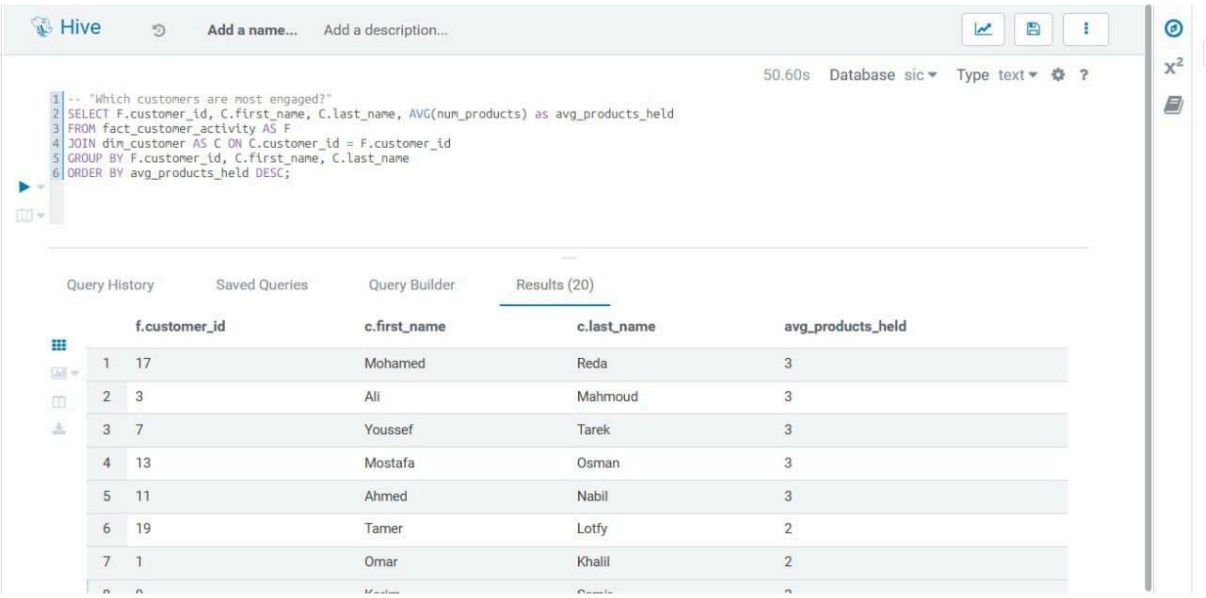


```sql
--Balance Behavior by Customer Segment
WITH customer_age_groups AS (
    SELECT
        customer_id,
        geography,
        CASE
            WHEN age < 30 THEN '18-29'
            WHEN age BETWEEN 30 AND 45 THEN '30-45'
            WHEN age BETWEEN 46 AND 60 THEN '46-60'
            ELSE '60+'
        END as age_group
    FROM dim_customer
)
SELECT
    c.geography,
    c.age_group,
    AVG(a.monthly_balance) as avg_monthly_balance,
    SUM(a.monthly_balance) as total_balance,
    COUNT(DISTINCT a.customer_id) as active_customers,
    AVG(m.clv_ltv) as avg_clv,
    SUM(CASE WHEN a.accepted = true THEN 1 ELSE 0 END) as offers_accepted
FROM fact_customer_activity a
JOIN fact_customer_metrics m ON a.customer_id = m.customer_id
JOIN customer_age_groups c ON a.customer_id = c.customer_id
WHERE m.is_churned = false
GROUP BY c.geography, c.age_group
ORDER BY total_balance DESC;
```

	c.geography	c.age_group	avg_monthly_balance	total_balance	active_customers	avg_clv	offers_accepted
1	UAE	30-45	8666.666666666666	26000	3	295000	2
2	Egypt	30-45	2860	14300	5	65160	4
3	Saudi Arabia	30-45	12000	12000	1	432000	1
4	Egypt	18-29	4100	8200	2	52650	2
5	Kuwait	30-45	6100	6100	1	244000	1
6	Egypt	46-60	0	0	1	0	0

- **Key Insight:** Prime customers are in their 30s-45s living in the UAE and Saudi Arabia. This segment holds the highest average and total balances, has a high CLV, and is most receptive to offers, making them the ideal target for growth initiatives

5.Most engaged customers:

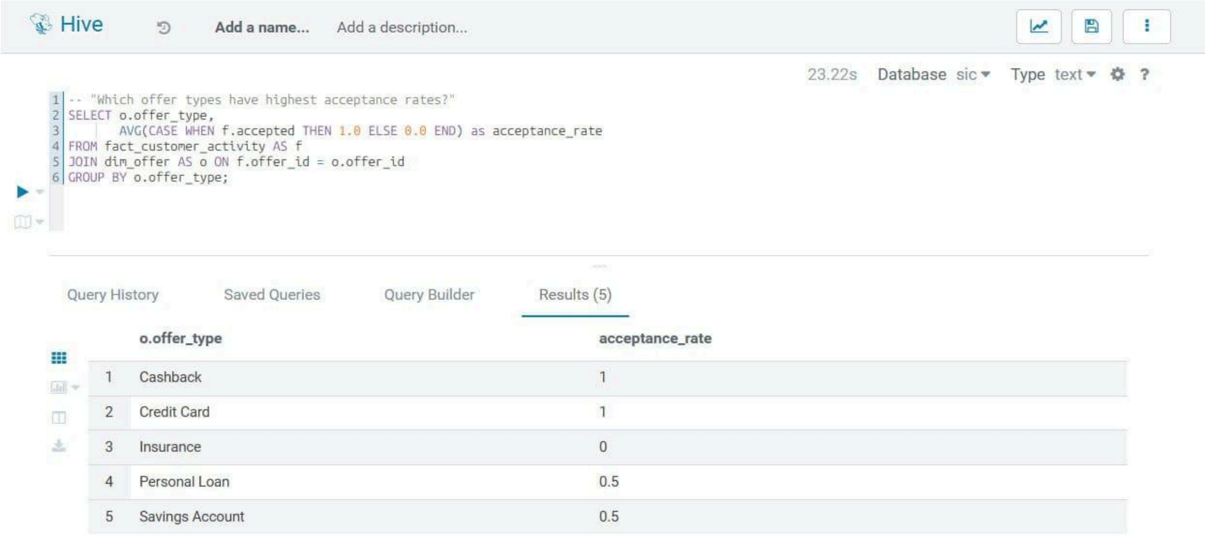


```sql
-- "Which customers are most engaged?"
SELECT F.customer_id, C.first_name, C.last_name, AVG(num_products) as avg_products_held
FROM fact_customer_activity AS F
JOIN dim_customer AS C ON C.customer_id = F.customer_id
GROUP BY F.customer_id, C.first_name, C.last_name
ORDER BY avg_products_held DESC;
```

	f.customer_id	c.first_name	c.last_name	avg_products_held
1	17	Mohamed	Reda	3
2	3	Ali	Mahmoud	3
3	7	Youssef	Tarek	3
4	13	Mostafa	Osman	3
5	11	Ahmed	Nabil	3
6	19	Tamer	Lotfy	2
7	1	Omar	Khalil	2

- **Key Insight:** These are our most loyal and valuable customers, candidates for loyalty rewards and premium product cross-selling.

6.Offers with highest acceptance rate:

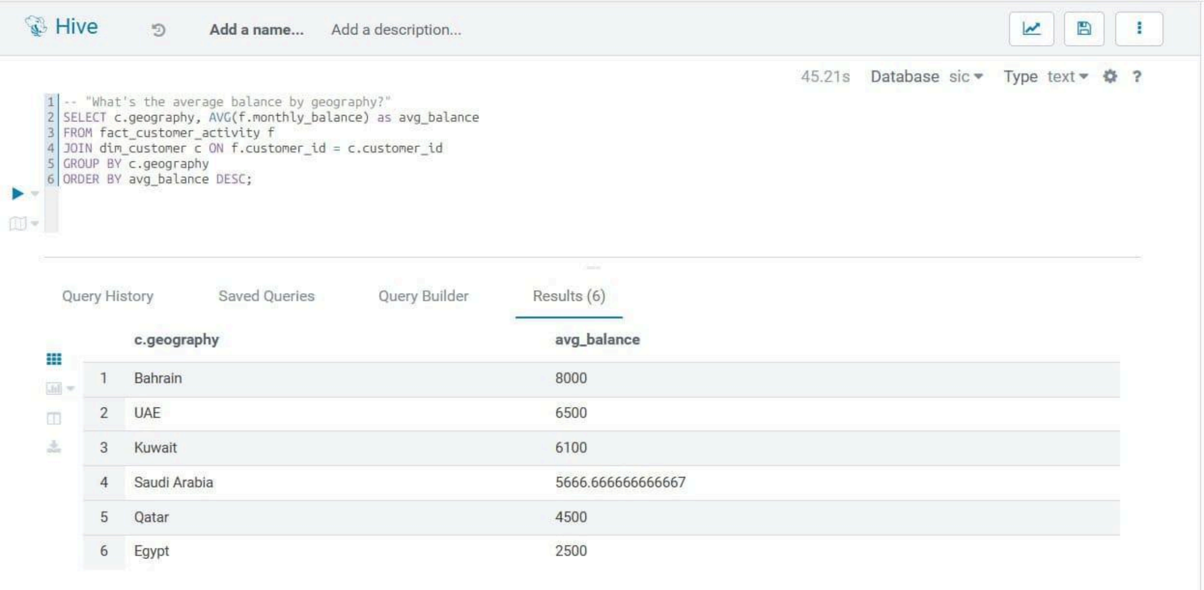


```sql
-- "Which offer types have highest acceptance rates?"
SELECT o.offer_type,
       AVG(CASE WHEN f.accepted THEN 1.0 ELSE 0.0 END) as acceptance_rate
FROM fact_customer_activity AS f
JOIN dim_offer AS o ON f.offer_id = o.offer_id
GROUP BY o.offer_type;
```

	o.offer_type	acceptance_rate
1	Cashback	1
2	Credit Card	1
3	Insurance	0
4	Personal Loan	0.5
5	Savings Account	0.5

- **Key Insight:** Customers prefer immediate, low-risk value. Cashback and Credit Card offers have 100% acceptance, while products like Insurance and Loans see much lower uptake, suggesting customers are wary of long-term commitments.

7.Average balance by geography:

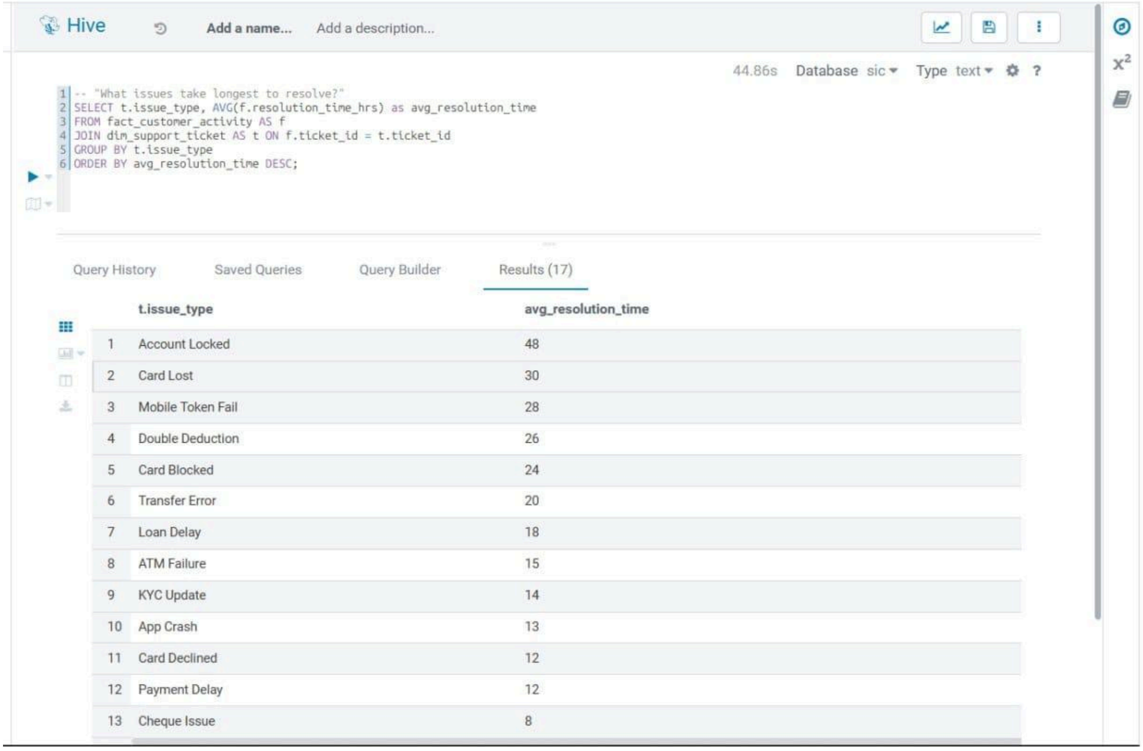


```
-- "What's the average balance by geography?"
SELECT c.geography, AVG(f.monthly_balance) as avg_balance
FROM fact_customer_activity f
JOIN dim_customer c ON f.customer_id = c.customer_id
GROUP BY c.geography
ORDER BY avg_balance DESC;
```

	c.geography	avg_balance
1	Bahrain	8000
2	UAE	6500
3	Kuwait	6100
4	Saudi Arabia	5666.666666666667
5	Qatar	4500
6	Egypt	2500

- **Key Insight:**Marketing and sales teams can leverage this by tailoring offers to high-balance regions (premium products, cross-selling opportunities), while designing retention-focused initiatives for lower-balance regions (cashback, discounts, loyalty rewards).

8.Issues that take the longest to resolve:



```
-- "What issues take longest to resolve?"
SELECT t.issue_type, AVG(f.resolution_time_hrs) as avg_resolution_time
FROM fact_customer_activity AS f
JOIN dim_support_ticket AS t ON f.ticket_id = t.ticket_id
GROUP BY t.issue_type
ORDER BY avg_resolution_time DESC;
```

	t.issue_type	avg_resolution_time
1	Account Locked	48
2	Card Lost	30
3	Mobile Token Fail	28
4	Double Deduction	26
5	Card Blocked	24
6	Transfer Error	20
7	Loan Delay	18
8	ATM Failure	15
9	KYC Update	14
10	App Crash	13
11	Card Declined	12
12	Payment Delay	12
13	Cheque Issue	8

8.Customer Management Dashboard .

Main Objectives

The main objectives of the dashboard are:

To track customer churn and identify high-risk segments.

To measure Customer Lifetime Value (CLV) and highlight high-value customers.

To analyze product usage patterns across customers.

To evaluate support ticket resolution times by issue type.

To provide actionable insights for improving customer retention and satisfaction.

. Data Model Design

The dashboard uses a star schema consisting of:

Fact Tables:

fact_customer_activity: stores customer activity, usage logs, tickets, and balances.

fact_customer_metrics: contains key metrics such as churn, CLV, number of products.

Dimension Tables:

dim_customer: customer demographics and attributes.

dim_offer: details of offers provided.

dim_usage: product categories used.

dim_support_ticket: issue types and severity.

This design improves query performance and makes analysis flexible.