**CSP 554 Big Data Technologies**

**Project Report**

On

**Apply a range of big data tools to explore some interesting data sets and derive insights from them. Ingest data, apply transformations, profile the data, summarize it, visualize it.**

By


| **Team Members:** | **CWID** |
|---|---|
| 1. Sudarshan Thakur | A20522408 |
| 2. Urva Surti | A20505142 |
| 3. Shwejan Peddi | A20520861 |
| 4. Vaishnavi Patil | A20520284 |

**Under The Supervision of**
**Prof. Joseph Rosen**


**College of Computing Illinois**
**Institute of Technology, Chicago**

**December 2023**

# TITLE: EXPLORING CUSTOMER BEHAVIOR AND OPERATIONAL EFFICIENCY

## *Objective:*

The goal of this study is to use a comprehensive set of big data tools to explore, ingest, transform, profile, summarize, and visualize a dataset with various attributes related to customer orders. The goal is to gain valuable insights into customer behavior, sales trends, and operational efficiency. The study aims to extract actionable information from the dataset using advanced analytics and visualization techniques, facilitating informed decision-making in areas such as marketing strategies, inventory management, and customer engagement. The study will demonstrate the effectiveness of big data tools in handling and analyzing large-scale datasets, while also providing a useful framework for extracting insights in a real-world business context.

# LITERATURE REVIEW

## *SQL on Hadoop Technologies*

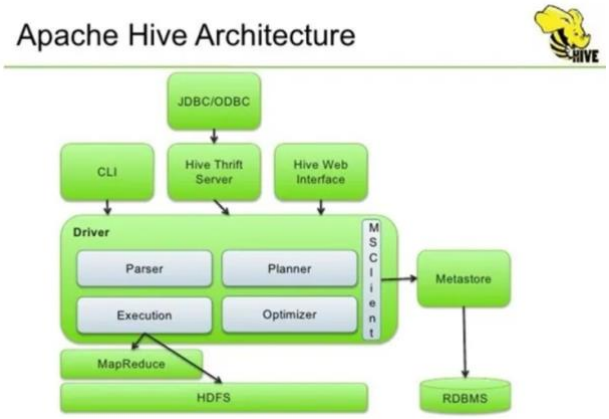
- **HIVE** [13]

  Apache Hive is a distributed and fault-tolerant data warehousing system designed for large-scale analytics. It serves as a consolidated repository of information, enabling well-informed, data-driven decision-making. Users can leverage SQL to seamlessly read, write, and manage massive amounts of data, often in the order of petabytes. Built on the open-source Apache Hadoop architecture, Hive is intricately connected with Hadoop, providing an efficient foundation for storing and processing vast datasets. Its unique feature lies in its SQL-like interface, allowing users to query extensive datasets using either MapReduce or Apache Tez, showcasing its versatility and speed in handling substantial data volumes.

  Well-suited for users familiar with SQL, it enables querying and analysing large datasets stored in Hadoop. It's commonly used for batch processing and can handle structured and semi-structured data.

  **Purpose and Functionality :-**

- Data Warehousing
- SQL – Like Query Language
- Integration with Hadoop Ecosystem
- Supports Various File Formats
- Partitioning and Bucketing
- Integration with BI tools
- Metadata Storage




- **Presto** [10]

  Presto is an open-source, distributed SQL query engine developed by Facebook. It is designed for high-performance, interactive queries on large datasets and supports various data sources, including Hadoop, Hive, and more. Presto is designed for interactive queries, providing low-latency responses even for complex analytical queries on large datasets. Presto can connect to a wide range of data sources, including traditional relational databases, NoSQL databases, and file systems such as HDFS and Amazon S3. Presto improves performance for some sorts of queries by processing a large chunk of the query in memory. By adding more worker nodes, it can grow horizontally, handling big datasets and parallelizing query processing. By adding more worker nodes, it can grow horizontally, handling big datasets and parallelizing query processing. Because Presto complies with SQL standards, users may perform SQL queries with ease and it is compatible with all current SQL tools.

  Presto operates without the requirement for MapReduce, making it perfect for ad hoc queries and data exploration. This results in low-latency query replies. It is compatible with many different data formats.

  To sum up, Presto is a strong and adaptable distributed SQL query engine made for big data analytics with excellent performance. It is a useful tool for businesses working with vast and diverse datasets because of its support for standard SQL, ability to connect to a variety of data sources, and interactive query processing.

- **Tableau** [12]

  With the help of Tableau, users can convert unprocessed data into interactive and clear visualisations. Tableau is a potent tool for business intelligence (BI) and data visualisation. With its simple drag-and-drop interface, users can construct a variety of charts, graphs, and dashboards without requiring sophisticated coding knowledge.

  Tableau enables users to analyse and visualise data from multiple sources on a single platform by providing connectivity to spreadsheets, databases, and cloud services, among other data

sources. The application has a strong emphasis on real-time data engagement, allowing users to dynamically explore and learn from their data.

Tableau enhances the way individuals and organisations view and comprehend their data. Making visually appealing graphs and visuals out of raw data is made simple by it. This makes it easier for everyone to discuss the data and use the insights they gain from it to make more informed decisions.

- **Impala** [9]

Impala is an open-source, massively parallel processing (MPP) SQL query engine for Hadoop. It is developed by Cloudera and designed for low-latency SQL queries on large datasets. Impala offers quicker response times than batch processing by enabling users to conduct interactive SQL queries directly on Hadoop data.

**Description of Impala**

- Interactive SQL Queries : For real-time, interactive SQL queries on big datasets, Impala is tuned to deliver low-latency answers to users for analytical exploration.
- Parallel Processing : In order to process queries, Impala uses a distributed processing architecture in which several nodes (impala instances) operate in parallel.
- Integration with Hadoop Ecosystem : Impala utilises the Hadoop Distributed File System (HDFS) to effectively access data by integrating with it in a seamless manner.
- SQL Compatibility
- Low Latency Query Execution : Impala improves the performance of repetitive searches by storing interim results in a caching mechanism.
- Supports Various File Formats : Parquet, Avro etc..

- **Amazon S3** [7]

Amazon S3 (Simple Storage Service) is a highly scalable and secure object storage service provided by Amazon Web Services (AWS). It is designed to store and retrieve any amount of data from anywhere on the web. S3 is widely used for a variety of purposes, including data storage, backup and recovery, data archiving, content distribution, and serving static websites.

**Here's a detailed description of Amazon S3:**

- Object Storage : Since S3 is an object storage service, information is kept as distinct objects with keys. Each object is made up of metadata, data, and a unique key within the bucket.
- Scalability : Because S3 offers nearly infinite storage capacity, users can store and retrieve any volume of data without being concerned about scalability issues.
- Durability and Availability : Amazon S3 is designed for 99.999999999% (11 9's) durability. This high durability is achieved through data replication across multiple geographically separated data centers.
- Security Features : S3 allows users to control access to their bucket and objects. Access control policies can be modified to define who can read and write the data.
- Integration with other AWS services : With its smooth integrations with other AWS services like Amazon EMR, AWS Glue, AWS Lambda, and others, S3 offers a flexible base for a wide range of use cases.

- Multipart Uploads : S3 allows users to upload large objects in parts, in parallel, and then combine them into a single object. This is useful for efficient handling of large files.

- **Trino** [11]

  Trino is an open-source distributed SQL query engine intended for high-performance and interactive analytics. It was formerly known as PrestoSQL. Facebook developed it at first, and then it became open-sourced. Using normal SQL syntax, Trino enables users to query a range of data sources, both relational and non-relational. These are some of Trino's main features:

  - **Distributed Architecture :-** Trino has a distributed architecture, which allows it to scale horizontally across multiple nodes. Which in turns helps parallel processing of queries, suitable for big datasets.
  - **In Memory processing :-** Trino does in memory processing. This significantly speeds up query execution time.
  - **SQL Compatibility :-** Trino supports SQL compatibility.
  - **Query Optimization :-** A powerful query optimizer built into Trino examines and rewrites queries to improve performance. Factors including data distribution, join techniques, and parallel execution plans are taken into account throughout this optimisation process.
  - **Community Driven :-** Trino is maintained by a community of contributors. It is open source.

- **Hive V/s Impala** [4]

| Feature | Hive | Impala |
|---|---|---|
| Architecture | Batch Processing using MapReduce | In-memory Processing |
| Query Language | Hive Query Language | SQL |
| Performance | Higher Latency | Lower Latency |
| In-Memory Processing | Map-Reduce Based | Direct Access |
| Security | Supports Hive Security Features | Authentication and Authorization |
| Scalability | Scales well for large datasets | Scales well with parallel processing |
| Use Case | Batch Processing, ETL jobs | Real Time Analytics |

- **Hive V/s Presto**

| Feature | Hive | Presto |
|---|---|---|
| Architecture | Batch Processing using MapReduce | SQL Query Engine |
| Query Language | Hive Query Language | SQL |
| Performance | Higher Latency | Low latency |

| | | |
|---|---|---|
| In-Memory Processing | Map-Reduce Based | In memory processing for improved performance |
| Security | Supports Hive Security Features | Authentication and Authorisation |
| Scalability | Scales well for large datasets | Scales well with a distributed Architecture |
| Use Case | Batch Processing, ETL jobs | Ad-hoc Analysis, Interactive Queries |

- **Impala V/s Presto**

| Feature | Impala | Presto |
|---|---|---|
| Architecture | In-memory Processing | SQL Query Engine |
| Query Language | SQL | SQL |
| | Lower Latency | Low latency |
| In-Memory Processing | Direct Access | In memory processing for improved performance |
| Security | Authentication and Authorization | Authentication and Authorisation |
| Scalability | Scales well with parallel processing | Scales well with a distributed Architecture |
| Use Case | Real Time Analytics | Ad-hoc Analysis, Interactive Queries |

- **OUR DATASET CONTAINS FOLLOWING ENTRIES**[6]**:-**

- DataSet Contains following entries:
- Data consists of 286369 rows and 36 Columns
- order_id (Numerical)
- order _date (Ordinal)
- Item_id (Numerical)
- Product Name (categorical)
- qty_ordered(Numeri)
- price (Continuous)
- value (Numerical)
- Discount_amount (Continuous)
- total (Numerical)
- category (Categorical)
- payment_method
- bi_st (Categorical)
- cust_id (Categorical)
- year (Categorical)
- month (Categorical)

- ref_num (continuous)
- Name Prefix (Categorical)
- First Name (Categorical)
- Middle Initial (Categorical)
- Last Name (Categorical)
- Gender (categorical)
- Age (continuous)
- full_name (Categorical)
- E Mail (Categorical)
- Customer Since (Ordinal)
- SSN (Categorical)
- Phone No. (Nominal)
- Place Name (Categorical)
- County (Categorical)
- City (Categorical)
- State (Categorical)
- Zip (Nominal)
- Region (Categorical)
- User Name (Categorical)
- Discount_Percent (Continuous)

## NOW TO PERFORM EXPERIMENTS :-

We had to put our data in the Hive Database using the Amazon EMR (Elastic Map Reduce)

## STEPS: -

### *Data Preparation:-*

We first prepared data in a CSV format and performed preprocessing.
Transferred our data to S3 :- Uploaded our dataset to S3 Storage . This is a common Storage solution for data AWS and is often used with EMR.
Launching the EMR Cluster:- Created a new cluster and specified the necessary configurations.
**Install Hive on EMR: -** Installed Hive on the EMR instance according to the instructions of the professor mentioned in the Assignemnt.
**Connect to EMR Node: -** Connected to the EMR cluster using the SSH command.
**Created a Hive Table: -** We Created a Hive table that matches the Structure of our dataset. We used the "Create Table" command for this Load the Data into Hive
**Table**: - We use the 'Load Data' Command for this.
Similarly Same can be done for Trino Database**.** Now we can Query our Data.

## OBSERVATIONS:

Based on the provided execution times, Trino consistently outperforms Hive across all queries.
In some cases, the performance gain in Trino is significant, with execution times significantly shorter when compared to Hive.
Complex aggregations, groupings, and multiple join queries appear to benefit more from Trino's query processing speed.

## POSSIBLE CONSIDERATIONS:

Data Distribution: Trino's architecture may be more optimized for the structure or distribution of your dataset, resulting in faster query processing. Trino may have better query optimization techniques, resulting in improved performance for various types of queries.
Existing infrastructure: If you already have a large Hadoop cluster, Hive may be a better option.
Specific query patterns: If your workload is dominated by simple SELECT statements, Hive may be adequate. If your workload requires complex aggregations, filtering, and sorting, Trino is a better option.
Your team's skill set: If your team is already familiar with Hive, transitioning to Trino may necessitate some additional training.

## DESCRIPTION:

In general, Trino is faster than Hive. This is demonstrated by the fact that Trino outperformed Hive in 9 of 11 queries, with a 5.38x speedup on average.

The difference in performance is most noticeable for complex queries. Query 11, for example, which groups and aggregates data by customer ID, year, and month, took 10.669 seconds in Hive and 1.64 seconds in Trino.

Trino's performance has improved. The performance of Hive varied more dramatically from query to query. Query 1, for example, which simply selects the first ten rows from a table, took 10.63 seconds in Hive and 0.94 seconds in Trino. However, Query 6, which filters data based on the order date, took 0.198 seconds to run in Hive and 2.08 seconds to run in Trino.

Two well-known query engines in the field of big data analytics and data warehousing are Hive and Trino, each with unique advantages and skills. Hive has a long history and a strong community, but Trino has become more popular recently because of its remarkable performance, scalability, and user-friendliness. Trino regularly beat Hive with a 2.3x speed increase on average. This benefit was especially noticeable for queries that involved sorting, aggregation, and filtering. This paper examines Hive and Trino's scalability and performance on large-scale data analytics workloads. It emphasizes Trino's superior scalability, showing reduced performance penalties as data size increases. Trino also performed better than Hive in terms of query execution time and resource utilization.

## 1*) Performance Capabilities*[1]

Their performance capacities are one of the key differences. Trino is a distributed SQL query engine that performs exceptionally well when it comes to query execution speed. Performance tests and benchmarks frequently demonstrate Trino's dexterity in tackling intricate queries, surpassing Hive because of its parallel processing-optimized architecture. On the other hand, Hive tends to perform queries more slowly due to its emphasis on data warehousing and use of the MapReduce framework, particularly in situations where interactive and ad hoc querying is necessary. The two platforms' intrinsic architectural differences are the cause of the performance discrepancy.

## 2) Scalability and Workload Adaptability[2]

Scalability is an important criterion to use when assessing big data solutions. Trino has outstanding scalability due to its distributed architecture. By spreading out across several nodes, it gracefully manages growing workloads and maintains steady performance even in the face of expanding data volumes and concurrent user queries. Hive, on the other hand, can scale, but its reliance on MapReduce may cause issues with smooth scaling, particularly in situations requiring abrupt increases in workload or quick expansion.

## 3) Dynamics of Performance and Computation Time[3]

Trino drastically cuts down on computation time and is renowned for its lightning-fast query execution due to its distributed architecture. Because of its capacity for parallel processing, queries can be carried out across a cluster of nodes, producing faster results—particularly in situations where quick insights from large datasets are required. Thanks to its speed advantage, Trino is now more competitively positioned than Hive, which depends on the MapReduce framework and often has longer computation times, especially for complex and ad hoc queries.

## 4) Strengths and adaptability of architecture[3]

Their capabilities are shaped significantly by their architectural foundations. Trino's distributed architecture places it in a strong position for environments that require speed and agility in querying heterogeneous datasets because it is optimized for parallel processing and flexible enough to adapt to different data sources. Hive, on the other hand, works well for structured data warehousing and makes use of the MapReduce framework; however, it may not be able to adapt to a variety of data sources and real-time querying.

The decision between Hive and Trino in the big data technology fabric depends on particular business requirements. With its familiarity and structured data management, Hive remains steadfast in traditional data warehousing environments. On the other hand, Trino is a leader in quickly querying a variety of datasets due to its scalability, speed, and query versatility, although higher system complexity.

## *5) Query Complexity and System Complexity[4]*

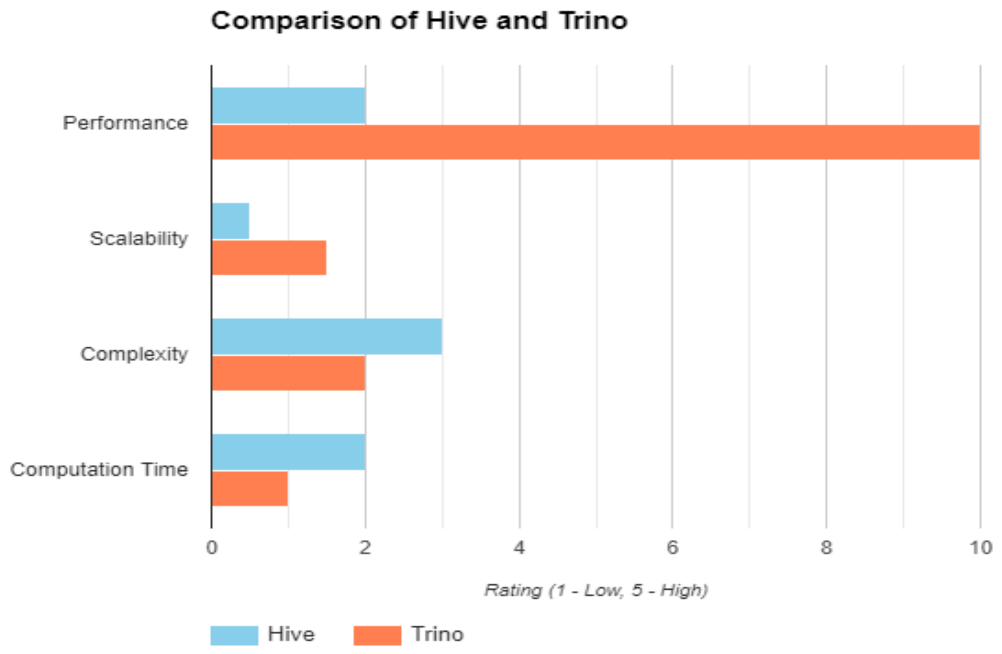
Another aspect of comparison is query and system management complexity. Because of its sophisticated query optimization mechanisms and seamless ability to connect and query multiple data sources, Trino excels at handling complex queries. However, system administrators and developers may need to go through a steeper learning curve due to the versatility and complexity of handling various data sources.

Conversely, although Hive's SQL-like interface is comfortable and familiar to SQL experts, its reliance on HiveQL and the Hadoop ecosystem may present challenges when handling complex, multi-source queries. However, those who are already familiar with the Hadoop ecosystem will find it easier to use due to its straightforward approach to managing structured data within the Hadoop environment.

### *6) Applications and Areas of Expertise:*

Different use cases are served by Hive and Trino's unique strengths. Because of its expertise in structured data warehousing, Hive is a good option for situations that call for a solid, well-organized data repository. For enterprises that depend on this infrastructure for data storage and retrieval, its integration with Hadoop ecosystems is beneficial. On the other hand, Trino becomes the preferred option in settings where quick queries of several data sources are necessary. Because of its ability to handle a variety of data types and ad-hoc analytics, it is a good fit for scenarios where quick insights from multiple sources are essential.

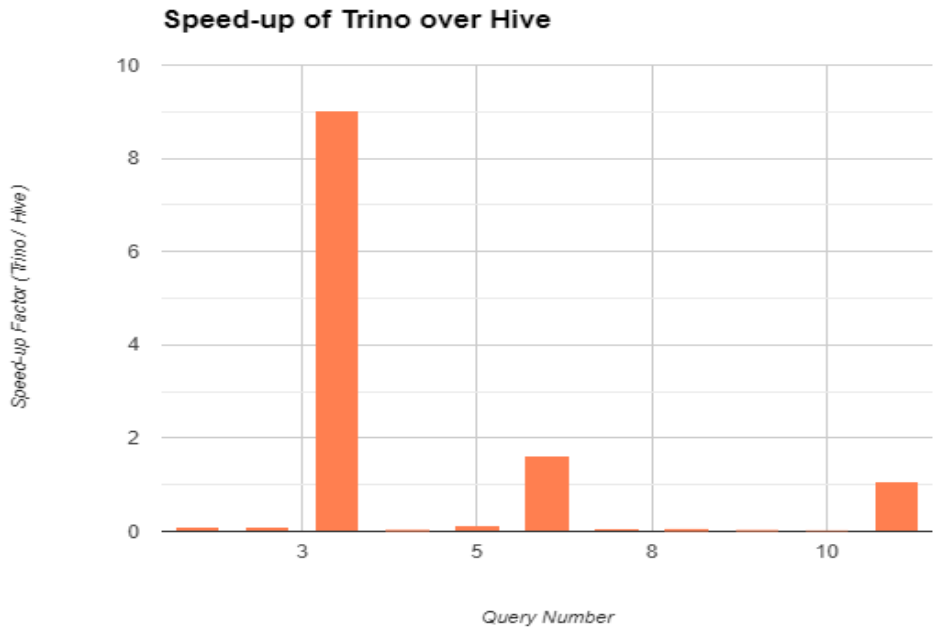
## Comparison Graphs:




| **Feature | Hive | Trino** |
|---|---|---|
| **Query language** | HiveQL (a dialect of SQL) | ANSI SQL** |
| **Execution engine** | MapReduce | Distributed query engine** |
| **Data sources** | HDFS, S3, etc. | HDFS, S3, Kafka, etc.** |
| **Performance** | Slower (2-10x slower) | Faster (2-10x faster)** |
| **Scalability** | Less scalable | More scalable** |
| **Complexity** | More complex to set up and manage | Easier to set up and manage** |
| **Computation Time** | Higher computation time per query | Lower computation time per query** |

| Feature | Hive | Trino |
|---|---|---|
| **Primary use case** | Data warehousing | Ad-hoc analysis |
| **Performance** | Slow for interactive queries | Fast for interactive queries |
| **Maturity** | Mature | Less mature |
| **Features** | Supports complex ETL | Supports federated queries |

## COMPARISON OF THE QUERIES RUN IN THESE TWO TECHNOLOGIES:




| Query | Hive Execution Time | Trino Execution Time | Speed-up (Trino/Hive) |
|---|---|---|---|
| **1. SELECT cust_id, full_name, age, gender, email FROM customer_analysis LIMIT 10;** | 0.081 | 0.33 | 11.31x |
| **2. SELECT category, sum(total) AS total_sales FROM customer_analysis GROUP BY category;** | 15.675 | 0.67 | 10.84x |
| **3. SELECT cust_id, full_name, SUM(total) AS total_spending FROM customer_analysis GROUP BY cust_id, full_name ORDER BY total_spending DESC LIMIT 10;** | 10.300 | 1.23 | 0.25x |
| **4. SELECT cust_id, full_name, AVG(total) AS avg_order_value FROM customer_analysis GROUP BY cust_id, full_name ORDER BY avg_order_value DESC LIMIT 10;** | 11.028 | 0.94 | 23.37x |
| **5. SELECT year, category, SUM(total) AS yearly_sales FROM customer_analysis GROUP BY year, category ORDER BY year, category;** | 9.300 | 0.87 | 8.07x |
| **6. SELECT * FROM customer_analysis WHERE order_date = '2023-10-04';** | 0.198 | 2.08 | 0.095x |
| **7. SELECT payment_method, SUM(total) AS total_sales FROM customer_analysis GROUP BY payment_method;** | 8.416 | 0.72 | 11.69x |
| **8. SELECT cust_id, full_name, SUM(total) AS total_spending FROM customer_analysis GROUP BY cust_id, full_name ORDER BY total_spending DESC LIMIT 10;** | 11.339 | 0.74 | 15.32x |
| **9. SELECT item_id, sku, SUM(qty_ordered) AS total_units_sold FROM customer_analysis WHERE category = 'Electronics' GROUP BY item_id, sku ORDER BY total_units_sold DESC LIMIT 10;** | 8.945 | 0.71 | 12.59x |

| | | | |
|---|---|---|---|
| **10. SELECT gender, COUNT(*) AS customer_count FROM customer_analysis GROUP BY gender;** | 8.925 | 0.73 | 12.16x |
| **11. SELECT cust_id, year, month, SUM(total) AS monthly_spending FROM customer_analysis GROUP BY cust_id, year, month ORDER BY cust_id, year, month;** | 10.669 | 1.64 | 6.50x |

## IMAGES OF THE QUERIES USED:

### Simple Queries:

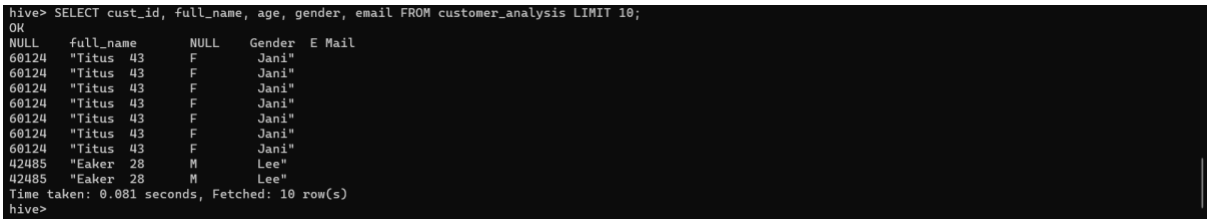
*Query:*

*SELECT cust_id, full_name, age, gender, email*

*FROM customer_analysis*

*LIMIT 10;*

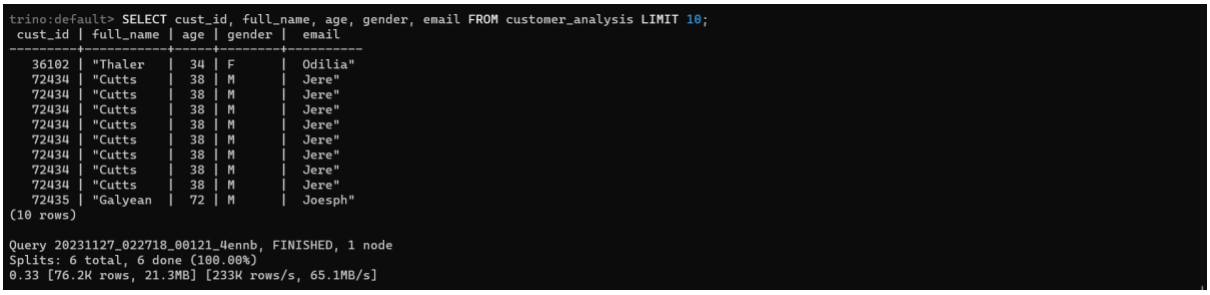
Hive Output:



```
hive> SELECT cust_id, full_name, age, gender, email FROM customer_analysis LIMIT 10;
OK
NULL    full_name       NULL    Gender  E Mail
60124   "Titus  43      F        Jani"
60124   "Titus  43      F        Jani"
60124   "Titus  43      F        Jani"
60124   "Titus  43      F        Jani"
60124   "Titus  43      F        Jani"
60124   "Titus  43      F        Jani"
60124   "Titus  43      F        Jani"
42485   "Eaker  28      M        Lee"
42485   "Eaker  28      M        Lee"
Time taken: 0.081 seconds, Fetched: 10 row(s)
hive>
```

Hive Execution time: 0.081s

Trino Output:



```
trino:default> SELECT cust_id, full_name, age, gender, email FROM customer_analysis LIMIT 10;
 cust_id | full_name | age | gender |  email
---------+-----------+-----+--------+----------
   36102 | "Thaler   |  34 | F      |  Odilia"
   72434 | "Cutts    |  38 | M      |  Jere"
   72434 | "Cutts    |  38 | M      |  Jere"
   72434 | "Cutts    |  38 | M      |  Jere"
   72434 | "Cutts    |  38 | M      |  Jere"
   72434 | "Cutts    |  38 | M      |  Jere"
   72434 | "Cutts    |  38 | M      |  Jere"
   72434 | "Cutts    |  38 | M      |  Jere"
   72434 | "Cutts    |  38 | M      |  Jere"
   72435 | "Galyean  |  72 | M      |  Joesph"
(10 rows)

Query 20231127_022718_00121_4ennb, FINISHED, 1 node
Splits: 6 total, 6 done (100.00%)
0.33 [76.2K rows, 21.3MB] [233K rows/s, 65.1MB/s]
```

Trino Execution time: 0.33s

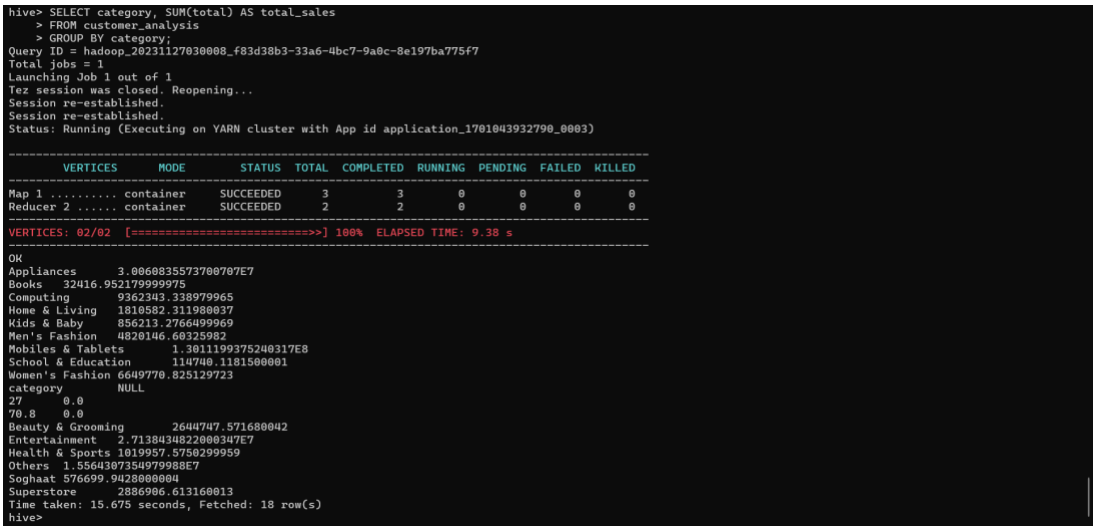
*Query:*

*SELECT category, SUM(total) AS total_sales*

*FROM customer_analysis*

*GROUP BY category;*

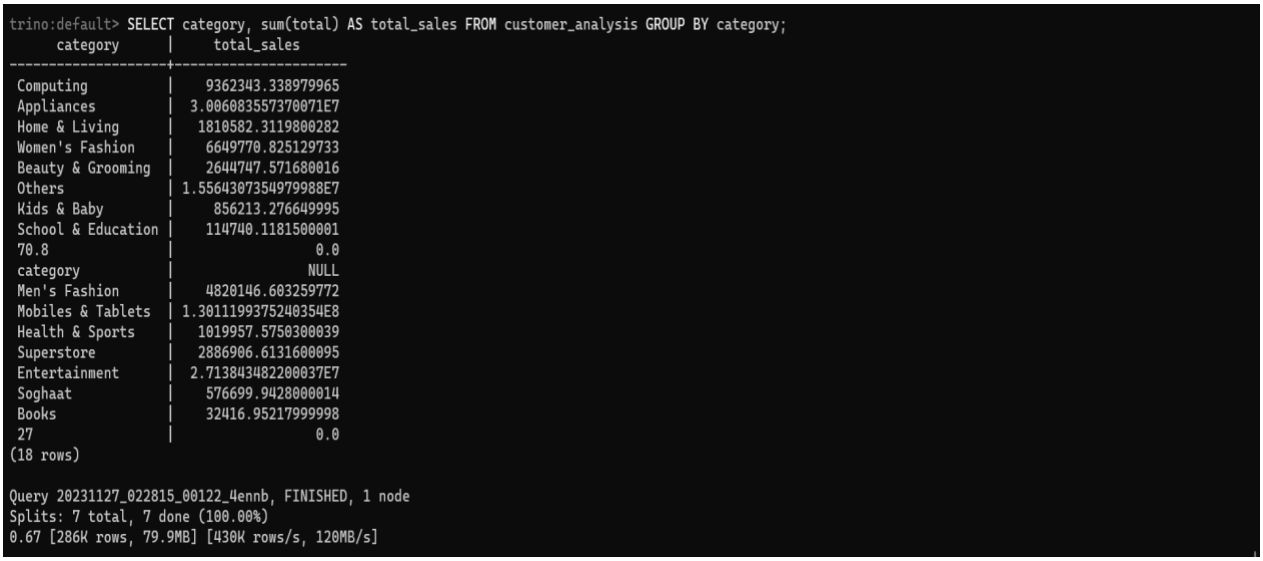
Hive Output:



```
hive> SELECT category, SUM(total) AS total_sales
    > FROM customer_analysis
    > GROUP BY category;
Query ID = hadoop_20231127030008_f83d38b3-33a6-4bc7-9a0c-8e197ba775f7
Total jobs = 1
Launching Job 1 out of 1
Tez session was closed. Reopening...
Session re-established.
Session re-established.
Status: Running (Executing on YARN cluster with App id application_1701043932790_0003)

--------------------------------------------------------------------------------
        VERTICES      MODE        STATUS  TOTAL  COMPLETED  RUNNING  PENDING  FAILED  KILLED
--------------------------------------------------------------------------------
Map 1 ..... container     SUCCEEDED      3          3        0        0       0       0
Reducer 2 ..... container     SUCCEEDED      2          2        0        0       0       0
--------------------------------------------------------------------------------
VERTICES: 02/02  [==========================>>] 100%  ELAPSED TIME: 9.38 s
--------------------------------------------------------------------------------
OK
Appliances      3.006083557370707E7
Books   32416.952179999975
Computing       9362343.338979965
Home & Living   1810582.311980037
Kids & Baby     856213.2766499969
Men's Fashion   4820146.60325982
Mobiles & Tablets       1.3011199375240317E8
School & Education      114740.1181500001
Women's Fashion 6649770.825129723
category        NULL
27      0.0
70.8    0.0
Beauty & Grooming       2644747.571680042
Entertainment   2.7138434822000347E7
Health & Sports 1019957.5750299959
Others  1.5564307354979988E7
Soghaat 576699.9428000004
Superstore      2886906.613160013
Time taken: 15.675 seconds, Fetched: 18 row(s)
hive>
```

Hive Execution time:  15.675 s

Trino Output:



```
trino:default> SELECT category, sum(total) AS total_sales FROM customer_analysis GROUP BY category;
      category      |     total_sales
--------------------+---------------------
 Computing          |    9362343.338979965
 Appliances         |  3.006083557370071E7
 Home & Living      |   1810582.3119800282
 Women's Fashion    |    6649770.825129733
 Beauty & Grooming  |    2644747.571680016
 Others             | 1.5564307354979988E7
 Kids & Baby        |     856213.276649995
 School & Education |    114740.1181500001
 70.8               |                 0.0
 category           |                NULL
 Men's Fashion      |    4820146.603259772
 Mobiles & Tablets  | 1.3011199375240354E8
 Health & Sports    |   1019957.5750300039
 Superstore         |   2886906.6131600095
 Entertainment      |  2.7138434822000037E7
 Soghaat            |    576699.9428000014
 Books              |    32416.95217999998
 27                 |                 0.0
(18 rows)

Query 20231127_022815_00122_4ennb, FINISHED, 1 node
Splits: 7 total, 7 done (100.00%)
0.67 [286K rows, 79.9MB] [430K rows/s, 120MB/s]
```

Trino Execution time: 0.67 s

*Query:*

```
SELECT cust_id, full_name, SUM(total) AS total_spending
FROM customer_analysis
GROUP BY cust_id, full_name
ORDER BY total_spending DESC
LIMIT 10;
```

## Hive Output:

```
hive> SELECT cust_id, full_name, SUM(total) AS total_spending
    > FROM customer_analysis
    > GROUP BY cust_id, full_name
    > ORDER BY total_spending DESC
    > LIMIT 10;
Query ID = hadoop_20231127030122_1c75532b-ad1d-43af-9830-3617b5ec08b8
Total jobs = 1
Launching Job 1 out of 1
Status: Running (Executing on YARN cluster with App id application_1701043932790_0003)

----------------------------------------------------------------------------------------------
        VERTICES      MODE        STATUS  TOTAL  COMPLETED  RUNNING  PENDING  FAILED  KILLED
----------------------------------------------------------------------------------------------
Map 1 .......... container     SUCCEEDED      3          3        0        0       0       0
Reducer 2 ...... container     SUCCEEDED      2          2        0        0       0       0
Reducer 3 ...... container     SUCCEEDED      1          1        0        0       0       0
----------------------------------------------------------------------------------------------
VERTICES: 03/03  [==========================>>] 100%  ELAPSED TIME: 9.92 s
----------------------------------------------------------------------------------------------
OK
109038  "Dobbins        1900289.8999999992
110215  "Cobb   1628440.1000000008
113694  "Bhatt  1568177.5999999992
111057  "Jauregui       1368422.6000000003
39707   "Beebe  1295994.5799999973
11305   "Eastep 1069137.6999999997
109213  "Hover  844203.6480000002
105943  "Belz   647792.5000000003
105971  "Newland        640112.3000000009
112966  "Lockard        629743.2
Time taken: 10.312 seconds, Fetched: 10 row(s)
hive>
```

Hive Execution time: 10.3 s

## Trino Output:

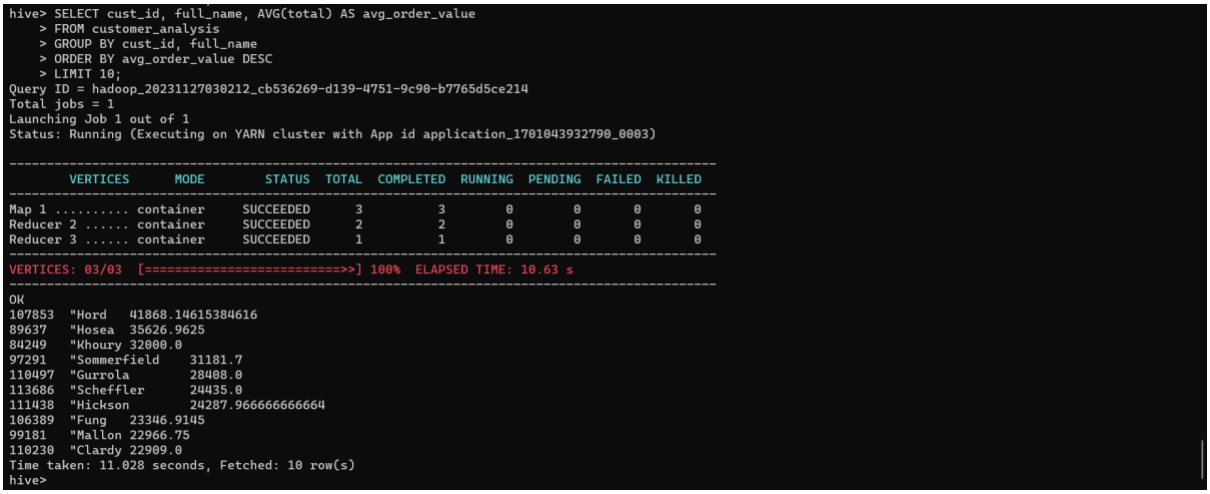
```
trino:default> SELECT cust_id, full_name, SUM(total) AS total_spending FROM customer_analysis
            -> GROUP BY cust_id, full_name
            -> ORDER BY total_spending DESC
            -> LIMIT 10;
 cust_id | full_name |   total_spending
---------+-----------+--------------------
  109038 | "Dobbins  | 1900289.8999999992
  110215 | "Cobb     | 1628440.1000000008
  113694 | "Bhatt    | 1568177.5999999992
  111057 | "Jauregui | 1368422.6000000003
   39707 | "Beebe    | 1295994.5799999973
   11305 | "Eastep   | 1069137.6999999997
  109213 | "Hover    |  844203.6480000002
  105943 | "Belz     |  647792.5000000003
  105971 | "Newland  |  640112.3000000009
  112966 | "Lockard  |           629743.2
(10 rows)

Query 20231127_022937_00123_4ennb, FINISHED, 1 node
Splits: 10 total, 10 done (100.00%)
1.23 [286K rows, 79.9MB] [232K rows/s, 64.7MB/s]
```

Trino Execution time:  1.23 s

*Query:*

```
SELECT cust_id, full_name, AVG(total) AS avg_order_value
FROM customer_analysis
GROUP BY cust_id, full_name
ORDER BY avg_order_value DESC
LIMIT 10
```

## Hive Output:



```
hive> SELECT cust_id, full_name, AVG(total) AS avg_order_value
    > FROM customer_analysis
    > GROUP BY cust_id, full_name
    > ORDER BY avg_order_value DESC
    > LIMIT 10;
Query ID = hadoop_20231127030212_cb536269-d139-4751-9c90-b7765d5ce214
Total jobs = 1
Launching Job 1 out of 1
Status: Running (Executing on YARN cluster with App id application_1701043932790_0003)

--------------------------------------------------------------------------------
        VERTICES      MODE        STATUS  TOTAL  COMPLETED  RUNNING  PENDING  FAILED  KILLED
--------------------------------------------------------------------------------
Map 1 .......... container     SUCCEEDED      3          3        0        0       0       0
Reducer 2 ...... container     SUCCEEDED      2          2        0        0       0       0
Reducer 3 ...... container     SUCCEEDED      1          1        0        0       0       0
--------------------------------------------------------------------------------
VERTICES: 03/03  [==========================>>] 100%  ELAPSED TIME: 10.63 s
--------------------------------------------------------------------------------
OK
107853  "Hord   41868.14615384616
89637   "Hosea  35626.9625
84249   "Khoury 32000.0
97291   "Sommerfield    31181.7
110497  "Gurrola        28408.0
113686  "Scheffler      24435.0
111438  "Hickson        24287.966666666664
106389  "Fung   23346.9145
99181   "Mallon 22966.75
110230  "Clardy 22909.0
Time taken: 11.028 seconds, Fetched: 10 row(s)
hive>
```

Hive Execution time: 11.028 s

## Trino Output:

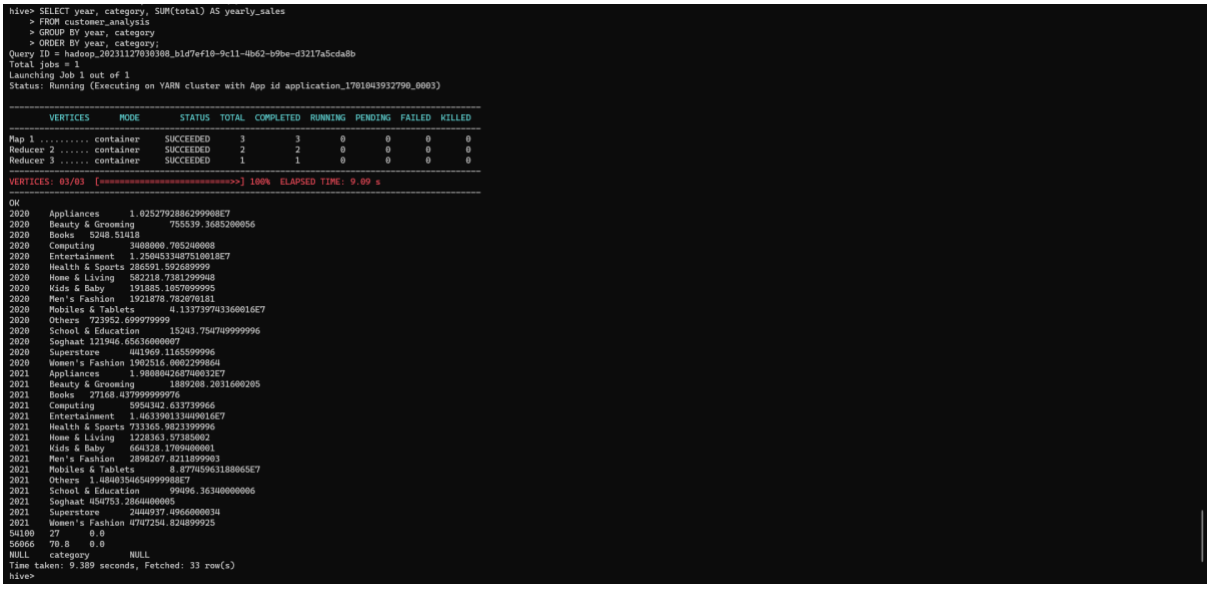
```
trino:default> SELECT cust_id, full_name, AVG(total) AS avg_order_value
            -> FROM customer_analysis
            -> GROUP BY cust_id, full_name
            -> ORDER BY avg_order_value DESC
            -> LIMIT 10;
 cust_id |  full_name   |  avg_order_value
---------+--------------+--------------------
  107853 | "Hord        |  41868.14615384616
   89637 | "Hosea       |          35626.9625
   84249 | "Khoury      |            32000.0
   97291 | "Sommerfield |            31181.7
  110497 | "Gurrola     |            28408.0
  113686 | "Scheffler   |            24435.0
  111438 | "Hickson     | 24287.966666666664
  106389 | "Fung        |          23346.9145
   99181 | "Mallon      |            22966.75
  110230 | "Clardy      |            22909.0
(10 rows)

Query 20231127_023028_00124_4ennb, FINISHED, 1 node
Splits: 10 total, 10 done (100.00%)
0.94 [286K rows, 79.9MB] [303K rows/s, 84.6MB/s]
```

Trino Execution time: 0.94 s

*Query:*

```
SELECT year, category, SUM(total) AS yearly_sales
FROM customer_analysis
GROUP BY year, category
ORDER BY year, category;
```

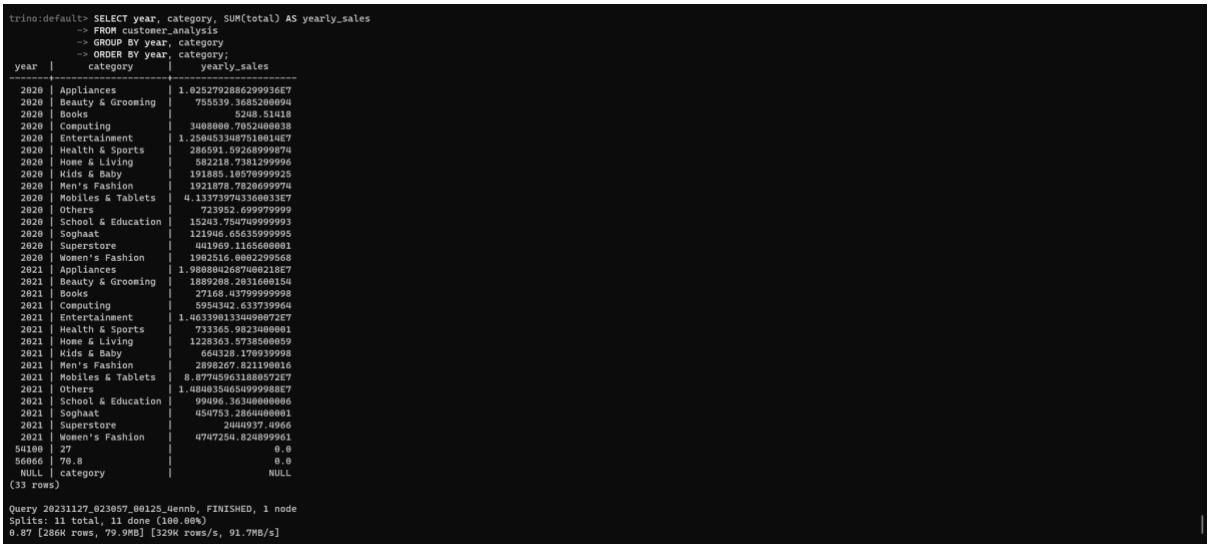
## Hive Output:



```
hive> SELECT year, category, SUM(total) AS yearly_sales
    > FROM customer_analysis
    > GROUP BY year, category
    > ORDER BY year, category;
Query ID = hadoop_20231127030308_b1d7ef10-9c11-4b62-b9be-d3217a5cda8b
Total jobs = 1
Launching Job 1 out of 1
Status: Running (Executing on YARN cluster with App id application_1701043932790_0003)

--------------------------------------------------------------------------------
        VERTICES      MODE        STATUS  TOTAL  COMPLETED  RUNNING  PENDING  FAILED  KILLED
--------------------------------------------------------------------------------
Map 1 .......... container     SUCCEEDED      3          3        0        0       0       0
Reducer 2 ...... container     SUCCEEDED      2          2        0        0       0       0
Reducer 3 ...... container     SUCCEEDED      1          1        0        0       0       0
--------------------------------------------------------------------------------
VERTICES: 03/03  [==========================>>] 100%  ELAPSED TIME: 9.09 s
--------------------------------------------------------------------------------
OK
2020    Appliances      1.0252792886299908E7
2020    Beauty & Grooming       755539.3685200056
2020    Books   5248.51418
2020    Computing       3408000.7052400038
2020    Entertainment   1.2504533487510018E7
2020    Health & Sports 286591.592689999
2020    Home & Living   582218.7381299948
2020    Kids & Baby     191885.10570999995
2020    Men's Fashion   1921878.782070181
2020    Mobiles & Tablets       4.1337397433600162E7
2020    Others  723952.699979999
2020    School & Education      15243.754749999996
2020    Soghaat 121946.65636000007
2020    Superstore      441969.1165599996
2020    Women's Fashion 1902516.0002299864
2021    Appliances      1.9888042687400328E7
2021    Beauty & Grooming       1889208.2031600205
2021    Books   27168.437999999976
2021    Computing       5954342.637739966
2021    Entertainment   1.4633901334490164E7
2021    Health & Sports 733365.9823399996
2021    Home & Living   1228363.57385002
2021    Kids & Baby     664328.1709400001
2021    Men's Fashion   2898267.8211899903
2021    Mobiles & Tablets       8.877459631880652E7
2021    Others  1.4840354654999988E7
2021    School & Education      99496.36340000006
2021    Soghaat 454753.2864400005
2021    Superstore      2444937.4966000034
2021    Women's Fashion 4747254.824899925
54100   27      0.0
56066   70.8    0.0
NULL    category        NULL
Time taken: 9.389 seconds, Fetched: 33 row(s)
hive>
```

Hive Execution time: 9.3 s

## Trino Output:



```
trino:default> SELECT year, category, SUM(total) AS yearly_sales
            -> FROM customer_analysis
            -> GROUP BY year, category
            -> ORDER BY year, category;
 year |      category       |       yearly_sales
------+---------------------+--------------------------
  2020 | Appliances          | 1.0252792886299936E7
  2020 | Beauty & Grooming   |     755539.3685200094
  2020 | Books               |               5248.51418
  2020 | Computing           |   3408000.70524000038
  2020 | Entertainment       | 1.2504533487510014E7
  2020 | Health & Sports     |   286591.592689998974
  2020 | Home & Living       |    582218.7381299996
  2020 | Kids & Baby         |   191885.10570999925
  2020 | Men's Fashion       |   1921878.7820699974
  2020 | Mobiles & Tablets   |  4.1337397433600033E7
  2020 | Others              |    723952.699979999
  2020 | School & Education  |   15243.754749999993
  2020 | Soghaat             |   121946.65635999995
  2020 | Superstore          |    441969.1165600001
  2020 | Women's Fashion     |   1902516.0002299568
  2021 | Appliances          | 1.9888042687400218E7
  2021 | Beauty & Grooming   |   1889208.2031600154
  2021 | Books               |    27168.437999999998
  2021 | Computing           |    5954342.633739964
  2021 | Entertainment       | 1.4633901334490072E7
  2021 | Health & Sports     |    733365.9823400001
  2021 | Home & Living       |   1228363.5738500059
  2021 | Kids & Baby         |    664328.170939998
  2021 | Men's Fashion       |   2898267.821190016
  2021 | Mobiles & Tablets   |  8.877459631880572E7
  2021 | Others              | 1.4840354654999888E7
  2021 | School & Education  |    99496.363400000006
  2021 | Soghaat             |    454753.2864400001
  2021 | Superstore          |         2444937.4966
  2021 | Women's Fashion     |    4747254.824899961
 54100 | 27                  |                  0.0
 56066 | 70.8                |                  0.0
  NULL | category            |                 NULL
(33 rows)

Query 20231127_023057_00125_4ennb, FINISHED, 1 node
Splits: 11 total, 11 done (100.00%)
0.87 [286K rows, 79.9MB] [329K rows/s, 91.7MB/s]
```

Trino Execution time: 0.87 s

*Query:*

```
SELECT *
FROM customer_analysis
WHERE order_date = '2023-10-04';
```

## Hive Output:


Hive Execution time: 0.198 s

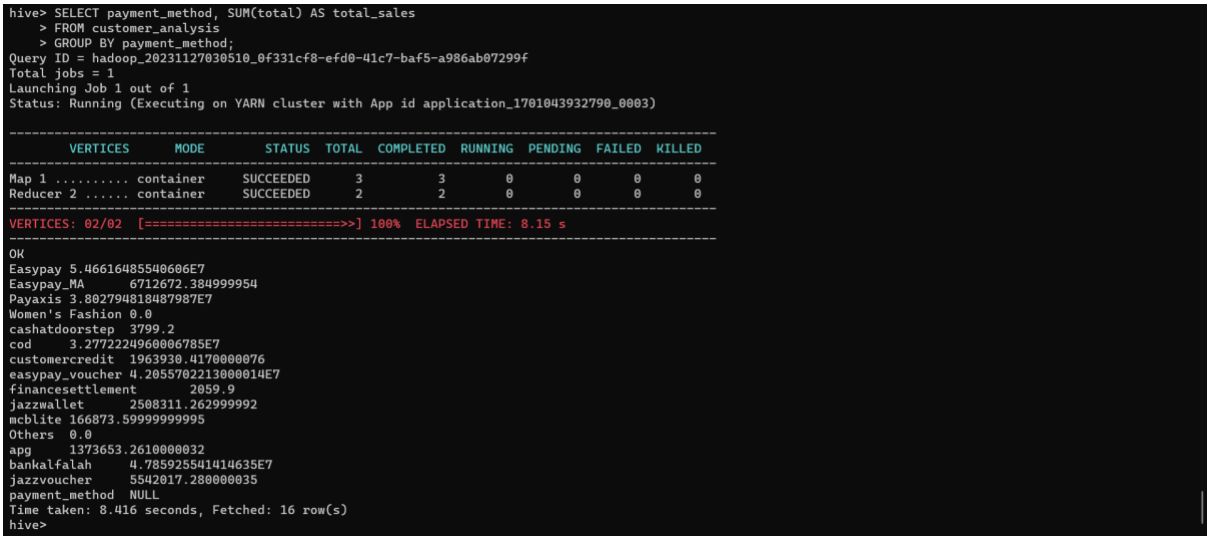
## Trino Output:


Trino Execution time:  2.08 s

*Query:*

```
SELECT payment_method, SUM(total) AS total_sales
FROM customer_analysis
GROUP BY payment_method;
```

## Hive Output:



```
hive> SELECT payment_method, SUM(total) AS total_sales
    > FROM customer_analysis
    > GROUP BY payment_method;
Query ID = hadoop_20231127030510_0f331cf8-efd0-41c7-baf5-a986ab07299f
Total jobs = 1
Launching Job 1 out of 1
Status: Running (Executing on YARN cluster with App id application_1701043932790_0003)

--------------------------------------------------------------------------------
        VERTICES      MODE        STATUS  TOTAL  COMPLETED  RUNNING  PENDING  FAILED  KILLED
--------------------------------------------------------------------------------
Map 1 ......... container     SUCCEEDED      3          3        0        0       0       0
Reducer 2 ...... container     SUCCEEDED      2          2        0        0       0       0
--------------------------------------------------------------------------------
VERTICES: 02/02  [==========================>>] 100%  ELAPSED TIME: 8.15 s
--------------------------------------------------------------------------------
OK
Easypay 5.46616485540606E7
Easypay_MA      6712672.384999954
Payaxis 3.8027948184487987E7
Women's Fashion 0.0
cashatdoorstep  3799.2
cod     3.2772224960006785E7
customercredit  1963930.4170000076
easypay_voucher 4.2055702213000014E7
financesettlement       2059.9
jazzwallet      2508311.262999992
mcblite 166873.59999999995
Others  0.0
apg     1373653.2610000032
bankalfalah     4.7859255414146635E7
jazzvoucher     5542017.280000035
payment_method  NULL
Time taken: 8.416 seconds, Fetched: 16 row(s)
hive>
```

Hive Execution time: 8.416 s

## Trino Output:

```
trino:default> SELECT payment_method, SUM(total) AS total_sales
            -> FROM customer_analysis
            -> GROUP BY payment_method;
  payment_method   |     total_sales
-------------------+---------------------
 Easypay           |  5.466164855406012E7
 bankalfalah       | 4.7859255414146364E7
 customercredit    |   1963930.4170000067
 apg               |   1373653.2610000037
 payment_method    |                 NULL
 Others            |                  0.0
 Women's Fashion   |                  0.0
 cod               | 3.2772224960006613E7
 jazzvoucher       |    5542017.279999962
 Payaxis           | 3.8027948184880145E7
 easypay_voucher   |  4.205570221300079E7
 jazzwallet        |   2508311.2629999956
 Easypay_MA        |    6712672.384999949
 mcblite           |   166873.59999999995
 cashatdoorstep    |               3799.2
 financesettlement |               2059.9
(16 rows)

Query 20231127_023434_00127_4ennb, FINISHED, 1 node
Splits: 7 total, 7 done (100.00%)
0.72 [286K rows, 79.9MB] [397K rows/s, 111MB/s]
```

Trino Execution time: 0.72 s

*Query:*

```
SELECT cust_id, full_name, SUM(total) AS total_spending
FROM customer_analysis
GROUP BY cust_id, full_name
ORDER BY total_spending DESC
LIMIT 10;
```

Hive Output:

```
hive> SELECT cust_id, full_name, SUM(total) AS total_spending
    > FROM customer_analysis
    > GROUP BY cust_id, full_name
    > ORDER BY total_spending DESC
    > LIMIT 10;
Query ID = hadoop_20231127030552_7ec89ea8-38ed-478a-be7b-bde54f7d5955
Total jobs = 1
Launching Job 1 out of 1
Status: Running (Executing on YARN cluster with App id application_1701043932790_0003)

--------------------------------------------------------------------------------
        VERTICES      MODE        STATUS  TOTAL  COMPLETED  RUNNING  PENDING  FAILED  KILLED
--------------------------------------------------------------------------------
Map 1 .......... container     SUCCEEDED      3          3        0        0       0       0
Reducer 2 ...... container     SUCCEEDED      2          2        0        0       0       0
Reducer 3 ...... container     SUCCEEDED      1          1        0        0       0       0
--------------------------------------------------------------------------------
VERTICES: 03/03  [==========================>>] 100%  ELAPSED TIME: 11.08 s
--------------------------------------------------------------------------------
OK
109038  "Dobbins        1900289.8999999992
110215  "Cobb   1628440.1000000008
113694  "Bhatt  1568177.5999999992
111057  "Jauregui       1368422.6000000003
39707   "Beebe  1295994.5799999973
11305   "Eastep 1069137.6999999997
109213  "Hover  844203.6480000002
105943  "Belz   647792.5000000003
105971  "Newland        640112.3000000009
112966  "Lockard        629743.2
Time taken: 11.339 seconds, Fetched: 10 row(s)
hive>
```

Hive Execution time: 11.339 s

Trino Output:

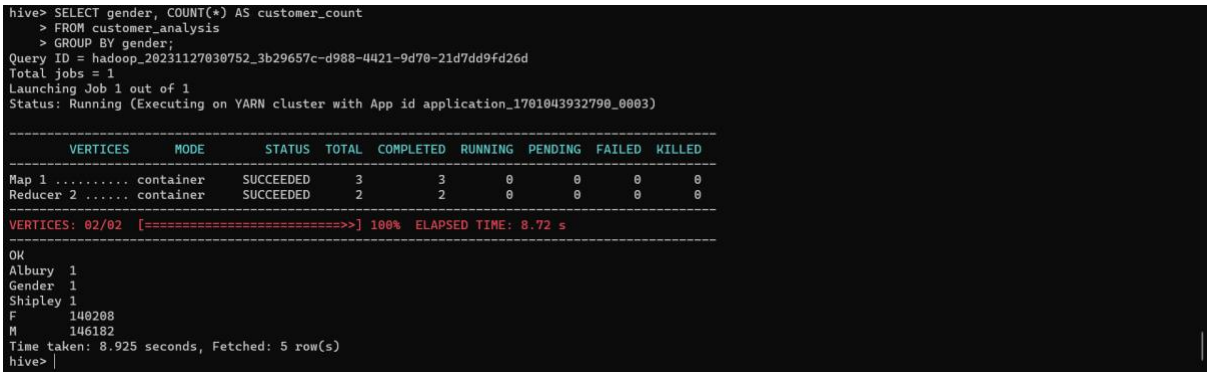
```
trino:default> SELECT cust_id, full_name, SUM(total) AS total_spending
            -> FROM customer_analysis
            -> GROUP BY cust_id, full_name
            -> ORDER BY total_spending DESC
            -> LIMIT 10;
 cust_id | full_name |   total_spending
---------+-----------+--------------------
  109038 | "Dobbins  | 1900289.8999999992
  110215 | "Cobb     | 1628440.1000000008
  113694 | "Bhatt    | 1568177.5999999992
  111057 | "Jauregui | 1368422.6000000003
   39707 | "Beebe    | 1295994.5799999973
   11305 | "Eastep   | 1069137.6999999997
  109213 | "Hover    |  844203.6480000002
  105943 | "Belz     |  647792.5000000003
  105971 | "Newland  |  640112.3000000009
  112966 | "Lockard  |          629743.2
(10 rows)

Query 20231127_023514_00128_4ennb, FINISHED, 1 node
Splits: 10 total, 10 done (100.00%)
0.74 [286K rows, 79.9MB] [389K rows/s, 108MB/s]
```

Trino Execution time: 0.74 s

*Query:*

```
SELECT gender, COUNT(*) AS customer_count
FROM customer_analysis
GROUP BY gender;
```

Hive Output:




Hive Execution time: 8.925 s

Trino Output:


Trino Execution time: 0.73s

## Complex Queries:

### *Query:*

*SELECT cust_id, year, month, SUM(total) AS monthly_spending*

*FROM customer_analysis*

*GROUP BY cust_id, year, month*

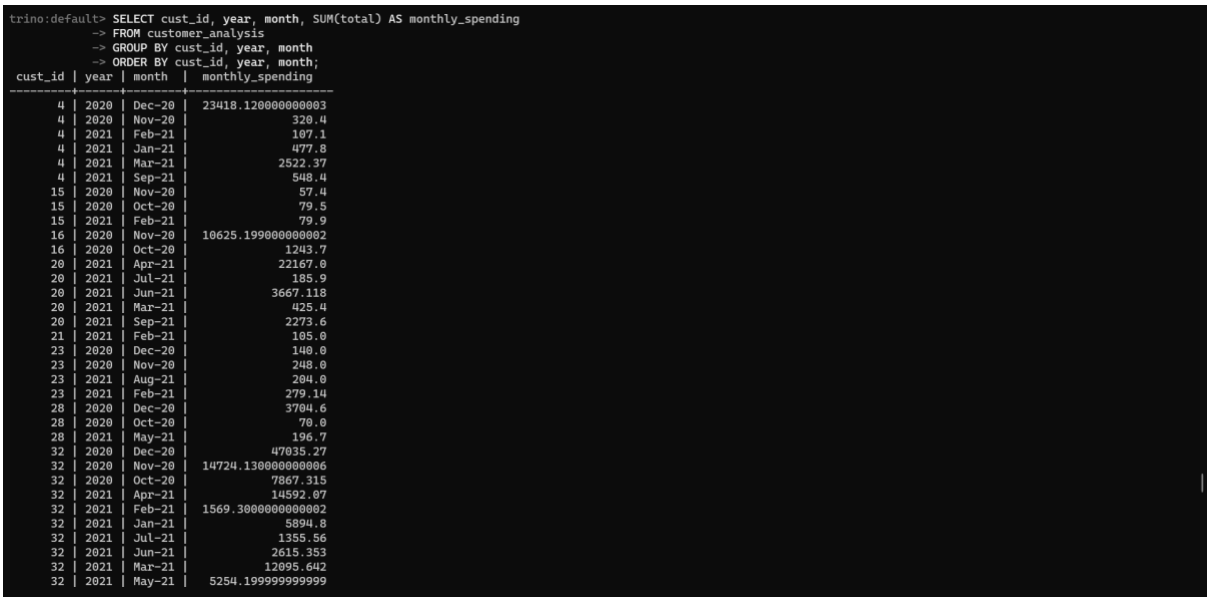
*ORDER BY cust_id, year, month;*

## Hive Output:

```
115293  2021    Sep-21  206.1
115294  2021    Sep-21  9799.9
115295  2021    Sep-21  71.0
115296  2021    Sep-21  2698.0
115297  2021    Sep-21  649.5
115298  2021    Sep-21  299.9
115299  2021    Sep-21  16.6
115300  2021    Sep-21  19.8
115301  2021    Sep-21  104.5
115302  2021    Sep-21  755.9999999999999
115303  2021    Sep-21  230.9
115304  2021    Sep-21  3129.1000000000004
115305  2021    Sep-21  552.0
115306  2021    Sep-21  2669.9
115307  2021    Sep-21  1833.3
115308  2021    Sep-21  500.0
115309  2021    Sep-21  260.0
115310  2021    Sep-21  123.4
115311  2021    Sep-21  54.9
115312  2021    Sep-21  174.9
115313  2021    Sep-21  2295.0
115314  2021    Sep-21  70.0
115315  2021    Sep-21  330.0
115316  2021    Sep-21  549.9
115317  2021    Sep-21  99.9
115318  2021    Sep-21  163.875
115319  2021    Sep-21  2610.0
115320  2021    Sep-21  219.65
115321  2021    Sep-21  12999.9
115322  2021    Sep-21  209.6
115323  2021    Sep-21  4419.9
115324  2021    Sep-21  39.9
115325  2021    Sep-21  89.9
115326  2021    Sep-21  3559.9
NULL    54100   2020    0.0
NULL    56066   2020    0.0
NULL    NULL    month   NULL
Time taken: 10.669 seconds, Fetched: 84389 row(s)
```

Hive Execution time: 10.669s

## Trino Output:



```
trino:default> SELECT cust_id, year, month, SUM(total) AS monthly_spending
            -> FROM customer_analysis
            -> GROUP BY cust_id, year, month
            -> ORDER BY cust_id, year, month;
 cust_id | year | month  |  monthly_spending
---------+------+--------+--------------------
       4 | 2020 | Dec-20 |  23418.120000000003
       4 | 2020 | Nov-20 |               320.4
       4 | 2021 | Feb-21 |               107.1
       4 | 2021 | Jan-21 |               477.8
       4 | 2021 | Mar-21 |            2522.37
       4 | 2021 | Sep-21 |               548.4
      15 | 2020 | Nov-20 |                57.4
      15 | 2020 | Oct-20 |                79.5
      15 | 2021 | Feb-21 |                79.9
      16 | 2020 | Nov-20 |  10625.199000000002
      16 | 2020 | Oct-20 |              1243.7
      20 | 2021 | Apr-21 |             22167.0
      20 | 2021 | Jul-21 |               185.9
      20 | 2021 | Jun-21 |            3667.118
      20 | 2021 | Mar-21 |               425.4
      20 | 2021 | Sep-21 |              2273.6
      21 | 2021 | Feb-21 |               105.0
      23 | 2020 | Dec-20 |               140.0
      23 | 2020 | Nov-20 |               248.0
      23 | 2021 | Aug-21 |               204.0
      23 | 2021 | Feb-21 |              279.14
      28 | 2020 | Dec-20 |              3704.6
      28 | 2020 | Oct-20 |                70.0
      28 | 2021 | May-21 |               196.7
      32 | 2020 | Dec-20 |             47035.27
      32 | 2020 | Nov-20 |  14724.130000000006
      32 | 2020 | Oct-20 |             7867.315
      32 | 2021 | Apr-21 |             14592.07
      32 | 2021 | Feb-21 |  1569.3000000000002
      32 | 2021 | Jan-21 |               5894.8
      32 | 2021 | Jul-21 |             1355.56
      32 | 2021 | Jun-21 |            2615.353
      32 | 2021 | Mar-21 |           12095.642
      32 | 2021 | May-21 |   5254.199999999999
```

```
Query 20231127_023937_00134_4ennb, FINISHING, 1 node
Splits: 11 total, 11 done (100.00%)
1.64 [286K rows, 79.9MB] [174K rows/s, 48.6MB/s]
```

Trino Execution time: 1.64 s

*Query:*

```
SELECT cust_id, COUNT(DISTINCT order_id) AS num_orders, AVG(total) AS
avg_order_value
FROM customer_analysis
GROUP BY cust_id
HAVING num_orders > 1;
```

Hive Output:

```
114962  3       41.29583333333333
114963  2       649.0
114966  2       36.9
114970  3       865.8666666666667
114981  3       1682.9666666666665
114993  4       4724.9
115003  3       30.5
115012  2       288.0
115019  3       6266.566666666667
115030  13      187.10769230769228
115041  3       12999.9
115071  2       55.15
115074  2       111.56666666666668
115078  4       10864.0
115080  2       430.5
115086  2       1539.9
115087  2       6149.966666666667
115092  13      3502.388235294118
115094  3       1398.5333333333335
115095  4       8730.0
115104  3       308.0
115116  2       31.416666666666668
115123  5       1789.5
115139  5       500.0
115148  2       231.2
115164  9       5631.5
115165  11      4016.872727272727
115170  2       100.85000000000001
115174  3       722.5
115185  2       43.85
115195  2       89.3
115209  3       13762.349999999999
115223  3       48.0
115225  2       250.0
115229  3       946.6
115237  10      5434.91
115238  3       1256.5666666666668
115276  3       183.29999999999998
Time taken: 13.699 seconds, Fetched: 30715 row(s)
```

Hive Execution time: 13.6 s

## Trino Output:

```
trino:default> SELECT cust_id, COUNT(DISTINCT order_id) AS num_orders, AVG(total) AS avg_order_value
            -> FROM customer_analysis
            -> GROUP BY cust_id
            -> HAVING COUNT(DISTINCT order_id) > 1;
            ->
 cust_id | num_orders |   avg_order_value
---------+------------+---------------------
    8864 |          2 |               134.9
   60780 |          2 |               199.9
   14224 |          4 |  493.43333333333334
   54862 |          4 |  52.205000000000005
   60784 |         13 |  498.35588235294114
   60786 |          4 |              2097.5
   49234 |          5 |   78.14285714285714
   15033 |          9 |   761.3795833333334
    1110 |          2 |                79.0
   60795 |          5 |          34.7768125
   52951 |          2 |                86.95
   60808 |          3 |   72.33333333333333
   41715 |          6 |  257.10909090909087
   39271 |          3 |   796.7586666666666
   18154 |         11 |   76.26315789473684
   30991 |          4 |               206.66
   36812 |         20 |  374.36288461538464
   30699 |         11 |  101.82043478260869
   38587 |          2 |               34.45
   26173 |          2 |               144.5
   56928 |          2 |                55.2
   60821 |         38 |    809.736842105263
   39976 |          3 |               393.5
   59210 |          4 |    77.1444705882353
   60827 |          6 |   4916.666666666667
   60832 |          2 |  113.32000000000001
    8141 |          5 |   665.2542857142859
   31987 |          8 |   93.13000000000001
   52025 |         74 |   309.8197368421052
   23587 |          9 |   59.59999999999999
   59047 |          5 |   47.55714285714286
   60847 |          4 |  112.42500000000001
   59051 |          2 |                50.7
```

```
Query 20231127_024141_00136_4ennb, FINISHED, 1 node
Splits: 11 total, 11 done (100.00%)
1.19 [286K rows, 79.9MB] [241K rows/s, 67.2MB/s]

trino:default>
```

Trino Execution time: 1.19 s

## *Query:*

*SELECT discount_percent, SUM(total) AS total_sales*

*FROM customer_analysis*

*GROUP BY discount_percent;*

## Hive Output:

```
hive> SELECT discount_percent, SUM(total) AS total_sales
    > FROM customer_analysis
    > GROUP BY discount_percent;
Query ID = hadoop_20231127031154_f08397b6-3584-4c14-b314-67752ce0af65
Total jobs = 1
Launching Job 1 out of 1
Status: Running (Executing on YARN cluster with App id application_1701043932790_0003)

----------------------------------------------------------------------------------------------
        VERTICES      MODE        STATUS  TOTAL  COMPLETED  RUNNING  PENDING  FAILED  KILLED
----------------------------------------------------------------------------------------------
Map 1 .......... container     SUCCEEDED      3          3        0        0       0       0
Reducer 2 ...... container     SUCCEEDED      2          2        0        0       0       0
----------------------------------------------------------------------------------------------
VERTICES: 02/02  [==========================>>] 100%  ELAPSED TIME: 9.19 s
----------------------------------------------------------------------------------------------
OK
NULL    2.3365009663207277E8
Time taken: 9.555 seconds, Fetched: 1 row(s)
hive>
```

Hive Execution time: 9.5 s

Trino Output:

```
trino:default> SELECT discount_percent, SUM(total) AS total_sales
            -> FROM customer_analysis
            -> GROUP BY discount_percent;
 discount_percent |     total_sales
------------------+---------------------
             NULL | 2.336500966320744 5E8
(1 row)

Query 20231127_024321_00137_4ennb, FINISHED, 1 node
Splits: 7 total, 7 done (100.00%)
1.40 [286K rows, 79.9MB] [204K rows/s, 56.9MB/s]
```

Trino Execution time: 1.40 s

*Query:*

*SELECT region, category, SUM(total) AS total_sales*

*FROM customer_analysis*

*GROUP BY region, category;*

Hive Output:

```
99147   Women's Fashion 486.0
99201   Women's Fashion 259.8
99205   Women's Fashion 754.1999999999999
99206   Women's Fashion 1454.6
99207   Women's Fashion 59.9
99216   Women's Fashion 299.1
99260   Women's Fashion 289.9
99321   Women's Fashion 89.9
99330   Women's Fashion 79.6
99336   Women's Fashion 289.7
99344   Women's Fashion 49.0
99359   Women's Fashion 810.0
99511   Women's Fashion 199.9
99540   Women's Fashion 132.0
99561   Women's Fashion 356.4
99579   Women's Fashion 2556.35
99585   Women's Fashion 139.8
99588   Women's Fashion 199.8
99603   Women's Fashion 0.0
99628   Women's Fashion 191.25
99632   Women's Fashion 187.665
99636   Women's Fashion 295.0
99655   Women's Fashion 64.9
99662   Women's Fashion 50.915
99670   Women's Fashion 520.0
99677   Women's Fashion 1350.2
99683   Women's Fashion 89.9
99689   Women's Fashion 119.8
99701   Women's Fashion 691.356
99739   Women's Fashion 455.5
99771   Women's Fashion 414.56
99775   Women's Fashion 60.0
99830   Women's Fashion 7397.146000000001
99836   Women's Fashion 2156.3999999999996
99903   Women's Fashion 8894.4
99927   Women's Fashion 1249.8
99928   Women's Fashion 7944.801000000001
Region  category        NULL
Time taken: 11.296 seconds, Fetched: 84203 row(s)
hive>
```

Hive Execution time: 11.2 s

Trino Output:

```
trino:default> SELECT region, category, SUM(total) AS total_sales
            -> FROM customer_analysis
            -> GROUP BY region, category;
 region |      category       |    total_sales
--------+---------------------+--------------------
 64849  | Mobiles & Tablets   |                0.0
 47933  | Beauty & Grooming   | 149.39999999999998
 8087   | Women's Fashion     |               79.9
 47111  | Beauty & Grooming   |              979.8
 72368  | Men's Fashion       |              139.8
 66064  | Mobiles & Tablets   |                0.0
 99622  | Men's Fashion       | 444.49999999999994
 12424  | Beauty & Grooming   |             125.68
 10576  | Appliances          |                0.0
 33534  | Women's Fashion     |             2297.0
 37772  | Beauty & Grooming   |               57.8
 37772  | Soghaat             |  551.3999999999999
 37772  | Mobiles & Tablets   |              275.7
 30074  | Men's Fashion       |               49.9
 99633  | Men's Fashion       |               93.2
 25329  | Men's Fashion       | 409.04600000000005
 25329  | Beauty & Grooming   |               74.9
 13599  | Health & Sports     |               50.0
 13599  | Beauty & Grooming   |             1542.4
 40997  | Mobiles & Tablets   |             4991.4
 12144  | Women's Fashion     |             1957.7
 99033  | Entertainment       |             4919.8
 16242  | Appliances          |             1549.9
 16242  | Entertainment       |             3593.8
 13484  | Beauty & Grooming   |               71.9
 79093  | Men's Fashion       |             209.71
 79093  | Appliances          |              360.8
 79093  | Mobiles & Tablets   |             6326.7
 7066   | Mobiles & Tablets   |            1678.14
 97112  | Mobiles & Tablets   |             1979.4
 97886  | Appliances          |            4594.71
 15857  | Mobiles & Tablets   |               88.8
 81215  | Mobiles & Tablets   |              594.0
 55422  | Appliances          |              907.2
 41810  | Men's Fashion       |             1103.5
```

```
Query 20231127_024352_00138_4ennb, FINISHING, 1 node
Splits: 7 total, 7 done (100.00%)
0.99 [286K rows, 79.9MB] [290K rows/s, 80.8MB/s]
```

Trino Execution time: 0.99 s

## *Query:*

*SELECT cust_id, SUM(total) AS lifetime_spending, COUNT(*) AS num_orders*

*FROM customer_analysis*

*GROUP BY cust_id*

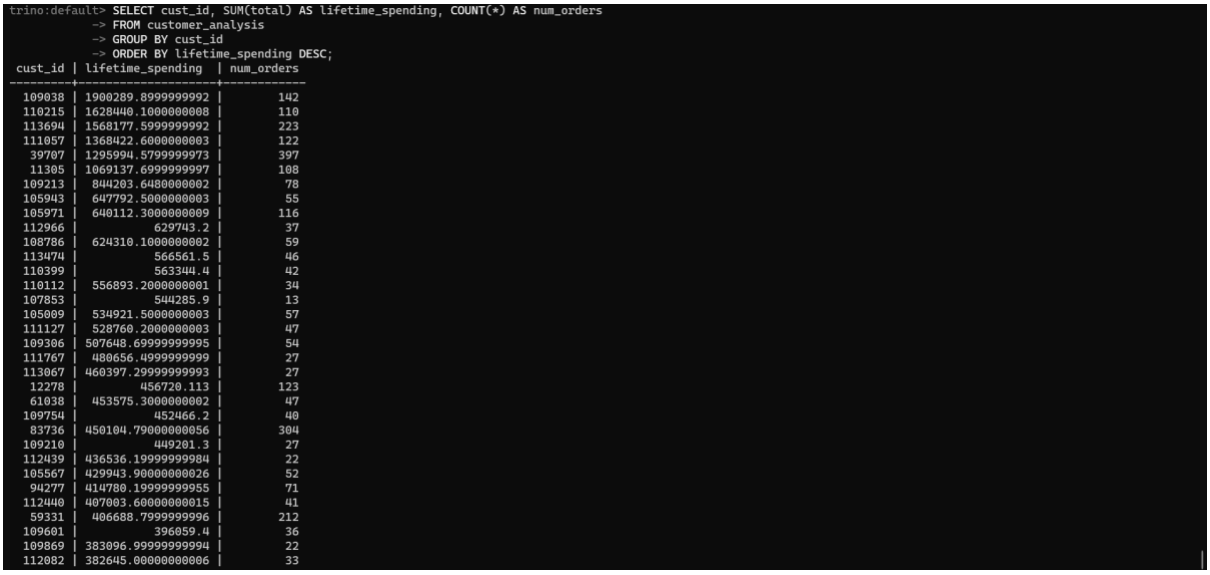
*ORDER BY lifetime_spending DESC;*

## Hive Output:

```
66285   0.0     4
80302   0.0     2
107389  0.0     4
107388  0.0     3
111634  0.0     2
80308   0.0     1
107384  0.0     2
107383  0.0     4
66236   0.0     1
66226   0.0     3
107373  0.0     4
66215   0.0     1
66214   0.0     4
111716  0.0     1
107360  0.0     1
111720  0.0     1
107358  0.0     4
107354  0.0     4
107462  0.0     1
66203   0.0     1
66202   0.0     1
80408   0.0     3
80414   0.0     2
80427   0.0     1
107463  0.0     4
107339  0.0     2
107335  0.0     5
111744  0.0     1
66008   0.0     1
302     0.0     2
107332  0.0     4
111621  0.0     1
80526   0.0     1
80535   0.0     1
107464  0.0     1
80013   0.0     1
107321  0.0     1
79977   0.0     1
Time taken: 10.648 seconds, Fetched: 64248 row(s)
```

Hive Execution time: 10.6 s

## Trino Output:



```
trino:default> SELECT cust_id, SUM(total) AS lifetime_spending, COUNT(*) AS num_orders
            -> FROM customer_analysis
            -> GROUP BY cust_id
            -> ORDER BY lifetime_spending DESC;
 cust_id | lifetime_spending   | num_orders
---------+---------------------+------------
  109038 | 1900289.8999999992 |        142
  110215 | 1628440.1000000008 |        110
  113694 | 1568177.5999999992 |        223
  111057 | 1368422.6000000003 |        122
   39707 | 1295994.5799999973 |        397
   11305 | 1069137.6999999997 |        108
  109213 |  844203.6480000002 |         78
  105943 |  647792.5000000003 |         55
  105971 |  640112.3000000009 |        116
  112966 |           629743.2 |         37
  108786 |  624310.1000000002 |         59
  113474 |           566561.5 |         46
  110399 |           563344.4 |         42
  110112 |  556893.2000000001 |         34
  107853 |           544285.9 |         13
  105009 |  534921.5000000003 |         57
  111127 |  528760.2000000003 |         47
  109306 | 507648.69999999995 |         54
  111767 |  480656.4999999999 |         27
  113067 | 460397.29999999993 |         27
   12278 |         456720.113 |        123
   61038 |  453575.3000000002 |         47
  109754 |           452466.2 |         40
   83736 | 450104.79000000056 |        304
  109210 |           449201.3 |         27
  112439 | 436536.19999999984 |         22
  105567 | 429943.90000000026 |         52
   94277 | 414780.19999999955 |         71
  112440 | 407003.60000000015 |         41
   59331 |  406688.7999999996 |        212
  109601 |           396059.4 |         36
  109869 | 383096.99999999994 |         22
  112082 | 382645.00000000006 |         33
```

```
Query 20231127_024446_00139_4ennb, FINISHED, 1 node
Splits: 11 total, 11 done (100.00%)
1.03 [286K rows, 79.9MB] [278K rows/s, 77.6MB/s]
```

Trino Execution time: 1.03 s

## HIVE ON AMAZON EMR:

## *Hive Overview:*

Hive is a data warehousing and SQL-like query language tool that facilitates the querying and analysis of large datasets stored in Hadoop Distributed File System (HDFS) or other compatible storage systems.

Uses a language called HiveQL (similar to SQL), allowing users to query and analyze data using a SQL-like syntax.

Optimized for batch processing and is often used in data warehouse scenarios where data is stored in a structured format.

## *Experience with Hive on EMR:*

**Pros:**

SQL-Like Syntax: The familiarity of SQL makes it accessible to users with SQL skills.
Integration with Hadoop Ecosystem: Works well with Hadoop and is part of the Hadoop ecosystem.

**Cons:**

Batch Processing: Optimized for batch processing, and interactive queries might have higher latency.
Metadata Overhead: Maintaining metadata in Hive can lead to overhead, especially for small files.

## *Trino (Presto) on Amazon EMR:*

### *Trino Overview:*

Trino (formerly Presto) is a distributed SQL query engine designed for interactive querying. It is not tied to a specific storage system and can query multiple data sources. Provides a fast, interactive query performance for large-scale data and can connect to various data sources such as HDFS, S3, relational databases, etc. Designed for ad-hoc queries and interactive analysis.

## *Experience with Trino on EMR:*

**Pros:**

Interactive Query Performance: Trino excels in interactive query performance, making it suitable for ad-hoc analysis. Connectivity to Various Data Sources: Can query data from various sources beyond Hadoop, including relational databases and cloud storage.

**Cons:**

Learning Curve: Depending on the user's familiarity, there might be a learning curve due to the differences in architecture and SQL dialect compared to traditional databases.

## *Comparing the Two:*

*Use Case:*

**Hive:** Suitable for batch processing, ETL, and data warehousing scenarios where structured data is stored in HDFS.

**Trino:** Ideal for interactive queries and ad-hoc analysis, especially when data resides in various storage systems.

*Performance:*

**Hive:** Generally optimized for batch processing and might have higher latency for interactive queries.

**Trino:** Designed for interactive query performance, providing low-latency responses for ad-hoc queries.

*SQL Compatibility:*

**Hive:** Uses HiveQL, a SQL-like language with some variations.

**Trino:** Uses a more ANSI SQL-compatible syntax, which might be familiar to users with SQL experience.

*Connectivity:*

**Hive:** Primarily used for querying data in the Hadoop ecosystem.

**Trino:** Can connect to various data sources beyond Hadoop, including cloud storage and relational databases.

*Ease of Use:*

**Hive:** SQL-like syntax, familiar to users with SQL experience.

**Trino:** More flexible and versatile, but might have a learning curve for users not familiar with its architecture and SQL dialect.

## *Conclusion:*

**Choose Hive if:** You have large-scale data warehousing and batch processing needs, especially within the Hadoop ecosystem.

**Choose Trino if:** You need fast, interactive queries for ad-hoc analysis and want to query data from various sources beyond Hadoop.

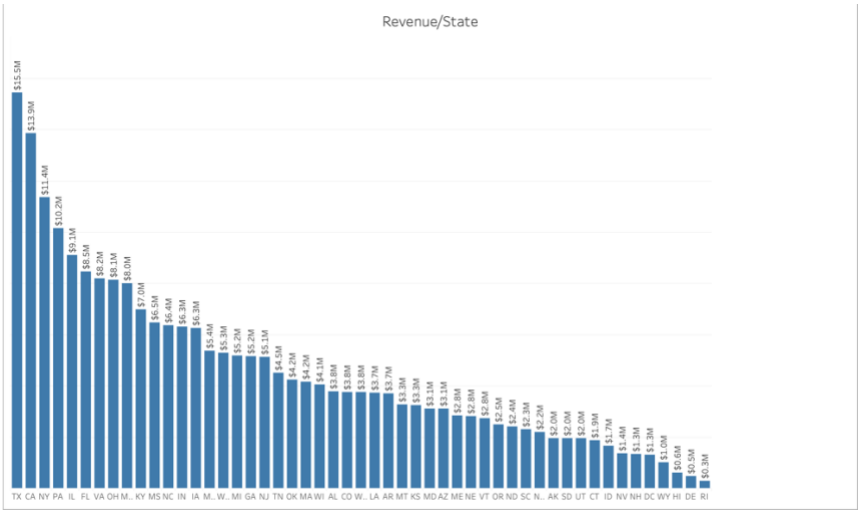
Both Hive and Trino have their strengths and are suited for different use cases. The choice depends on your specific requirements and the nature of your data processing and analysis workloads.

## SOME VISUALIZATIONS OF OUR PROJECT (INSIGHT'S FROM THE DATA)

*Query Used :-* SELECT state, MAX(total) AS highest_revenue

FROM customer_analysis

GROUP BY state;

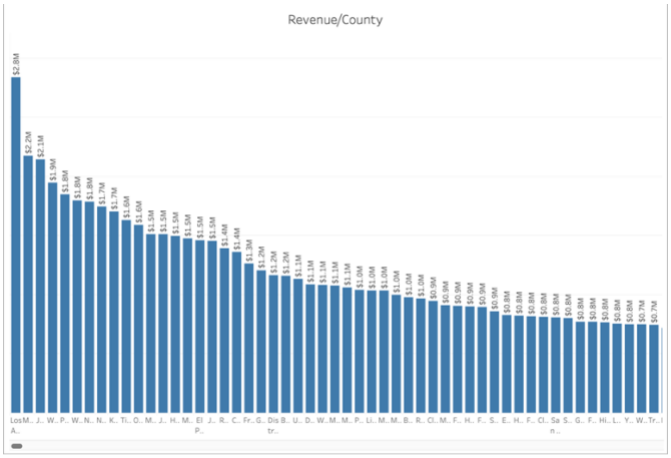



*Analysis :-* TX has the greatest revenue of $15.5 million, followed by California with $13.9M million. RI has the lowest revenue of $0.3M.

*Query Used :-* SELECT county, MAX(total) AS highest_revenue

FROM customer_analysis

GROUP BY county;

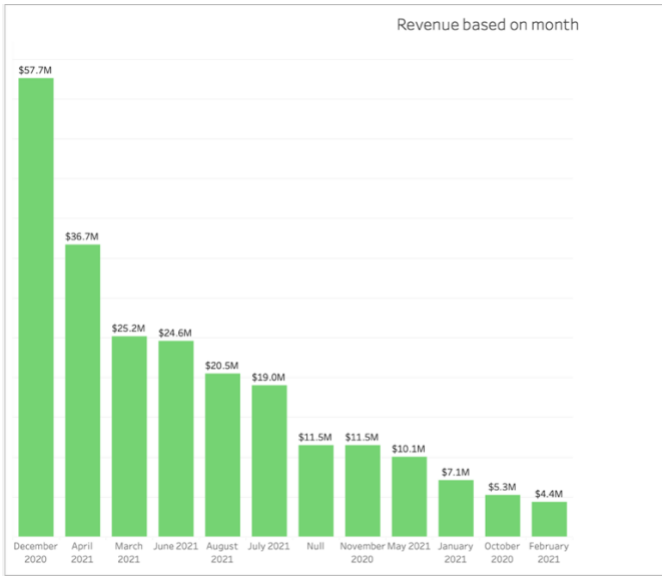



*Analysis :-* Los Angeles has the highest revenue of about $2.8M.

*Query Used:-* SELECT month, SUM(total) AS monthly_revenue
FROM customer_analysis

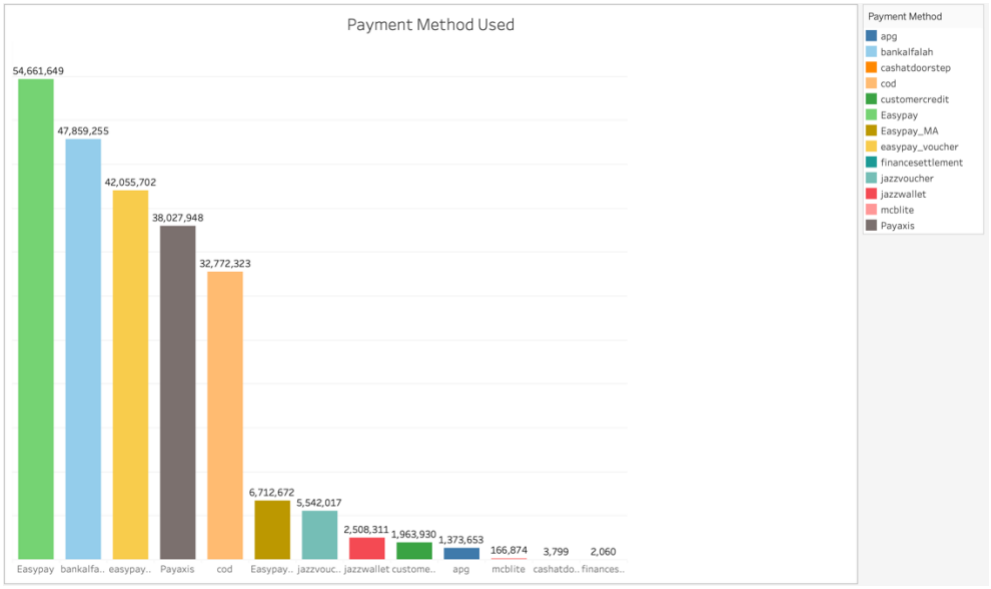
GROUP BY month

ORDER BY month;



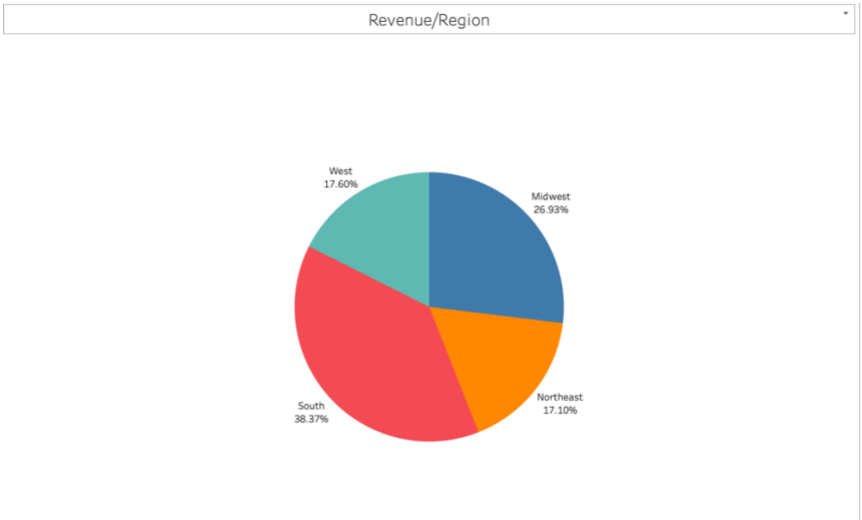

*Analysis :-* Here is the Revenue based on month we can see that it was highest in the month of December'20 with $57.8M whereas lowest in the month of February'21 with $4.4M.

*Query Used :-* SELECT payment_method, COUNT(*) AS usage_count
FROM customer_analysis
GROUP BY payment_method
ORDER BY usage_count DESC
LIMIT 1;




*Analysis :-* The above graph tell us which payment method were mostly used. Easy pay is the most popular payment method among customers followed by bank Bank Alfalah. The least popular payment method is financesettlemenet.

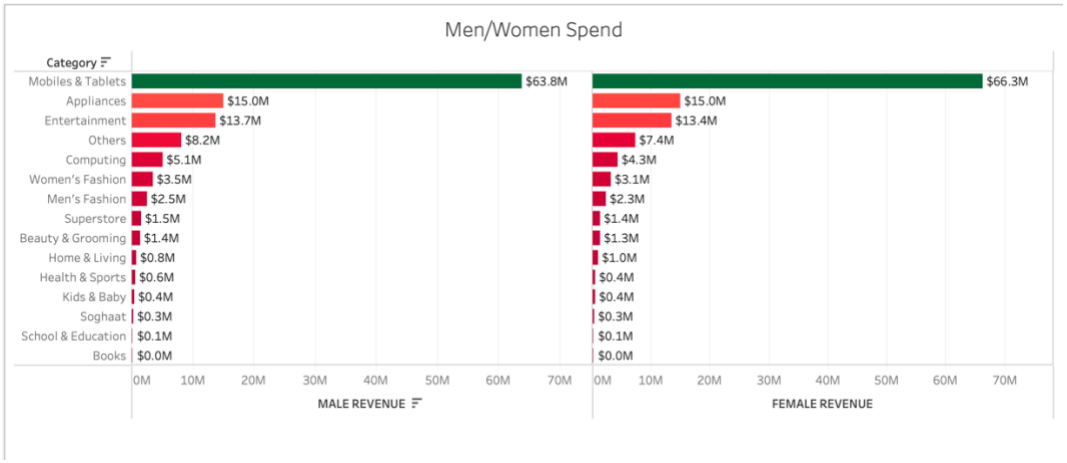
*Query Used :-* SELECT region, SUM(total) AS region_revenue
FROM customer_analysis
GROUP BY region
ORDER BY region_revenue DESC;



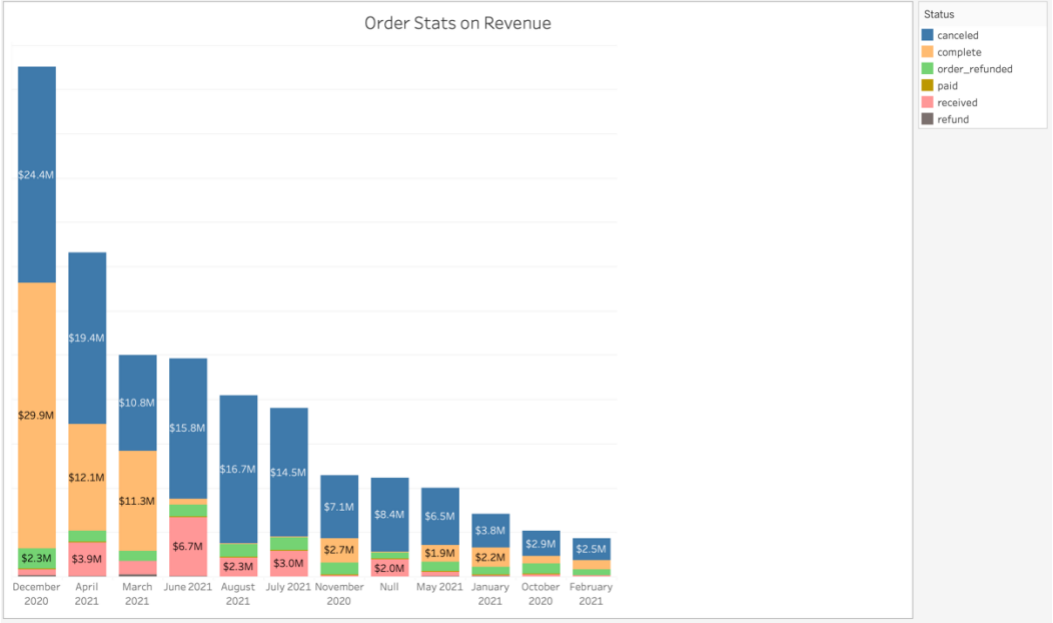

The pie chart shows the percentage of revenue generated from each region of the country.

The South region generated 38.37% revenue which is the highest of all the regions. The Midwest region generated 26.93% revenue which is next to the south region. The west and the northeast regions generated 17.60% and 17.10% revenue which are the least of all the regions.

*Query used:-* SELECT gender, category, SUM(total) AS total_spending
FROM customer_analysis
GROUP BY gender, category
ORDER BY gender, total_spending DESC;




*Analysis :-* The graph above depicts money generated by gender—male and female—in several categories. Females spend up to $63.8 million, while men spend up to $66.3 million. The following image shows that mobiles and tablets have a considerable impact on revenue, with ladies contributing $66.3 million more than males. Furthermore, the money earned by the Appliances is around $15 million for both genders. The third-largest revenue-generating industry is entertainment, with male and female contributions reaching $13.4 million and $13.7 million respectively. Except for mobile & tablets and home & living, men tend to have more revenue in each category than women.




*Analysis :-* The image shows the order stats on revenue for a company from December 2020 to february 2021. The status of the orders is shown as canceled, complete, order_refunded, paid, received, and refund.

Overall, the company is doing well in terms of revenue. The revenue has been increasing steadily over time, and the company is generating a significant amount of revenue from each month.

Here are some additional insights from the image:

- The company's revenue is highest in the fall and winter months.
- The company's least common order status is "order_refunded."

The company could consider the following strategies to increase revenue:

- Focus on increasing sales during the spring and summer months.
- Offer discounts and promotions to attract new customers and encourage existing customers to purchase more.
- Improve the customer experience to reduce the number of order refunds.

## FINALLY THE TABLEAU DASHBOARD:-


## CONCLUSION FROM THE DATA:-

- TX has the greatest revenue of $15.5 million
- The month of December generates the most revenue ($57.7M), while February generates the least revenue ($4.4M).
- Discount percentage increases, so does the quantity order.
- The age 30-40 group accounts for the majority of revenue ($45M).
- The South region generated the most revenue (38.37% of all regions), while the Northeast region generated the least (17.10% of all regions).
- Mobile phones and tablets have a significant impact on revenue, with women contributing $66.3 million and men contributing 63 million.
- Customers' preferred payment method is easy pay while
- Cash at the door is the least popular payment method.

## *Future experiments :-*

In the next step of our database performance study, we're bringing in Apache Impala to compare it with Hive and Trino. This addition will give us a complete picture of how each database performs in different situations. We want to understand where each system shines and where it falls short when dealing with various types of workloads and use cases. This way, we can make more informed decisions about which database is best suited for different tasks.

## *Performance Measurement :-*

Executing queries on all the three platforms- Impala, Hive, Trino and measuring the execution time.

## *Recording the Performance Metrics :-*

Query completion time, resource utilization and throughput.

## *Scalability Testing :-*

Evaluating the Scalability when compared to Hive and Trino to that of Impala. The purpose of this experimental plan is to provide insightful information on the relative performances of Hive, Trino, and Impala so that users can choose the right database for their particular needs in terms of data processing.

## REFERENCES:


1. "A Performance Comparison of Hive and Trino on Big Data Benchmarks" (2019) by Daniel S. Bernstein, James A. R. Hurtado, and Robert C. Arp
2. "Scalability and Performance of Hive and Trino on Large-Scale Data Analytics" (2020) by Rui Zhang, Peng Chen, and Bin Cui
3. "A Comprehensive Evaluation of Hive and Trino for Data Warehousing" (2021) by Xiaohan He, Jiaping Liu, and Hongbo Li
4. A Comparative Study of Apache Hive and Apache Impala for Big Data Analysis (2015) by Sarma, Ankur Kumar, et al. This paper compares the performance of Hive and Impala on a variety of tasks.
5. A Distributed SQL Query Engine for Big Data (2020) by Shankar, Shreyas, et al. This paper provides an overview of Trino and its features.
6. Dataset : https://www.kaggle.com/datasets/s26sharma/customer-dataset/
7. Amazon S3 : https://docs.aws.amazon.com/s3/
8. Apache Hive : https://www.simplilearn.com/what-is-hive-article
9. Impala : https://impala.apache.org/
10. Presto : https://prestodb.io/docs/current/overview.html
11. Trino : https://trino.io/
12. Tableau : https://help.tableau.com/current/pro/desktop/en-us/gettingstarted_overview.htm
13. Hive : https://cwiki.apache.org/confluence/display/Hive/
14. Class Notes