Analyzing Superstore Data with Apache Hive

This project uses **Apache Hive** to explore and analyze a dataset containing sales information for a superstore. Apache Hive is a data warehousing tool that allows us to query large datasets stored in distributed storage like Hadoop using a SQL-like language. Here, I created a Hive table for the superstore data and ran a series of complex queries to gain insights into sales performance.

What This Project Does

1. **Creating a Table for Sales Data**: The superstore data includes information like order date, quantity, discount, cost, and shipping information. I used Hive to create a table that organizes this data for efficient querying.
2. **Running Complex Queries**: I ran six complex queries to uncover various insights, including:
    - Total sales by customer type, grouping and ordering by specific conditions.
    - Summing orders based on customer type, order frequency, and specific conditions for dates or discounts.
    - Analyzing sales by location (province) and grouping results by region.
    - Filtering orders by date, location, and customer type to identify high-value transactions or trends.

Key Insights from the Queries

- **Sales by Customer Type**: The queries provide total sales figures for different customer types (e.g., "Consumer" or "Corporate") and sort them to highlight the top contributors.
- **Region-Based Analysis**: By grouping data by provinces like Nunavut and Ontario, we can see which regions have the most orders and highest sales.
- **Time-Based Trends**: Several queries analyze sales activity during specific months or years, identifying peak sales periods or changes over time.
- **High-Value Orders**: By filtering on criteria like high unit prices or large order quantities, we focus on the most significant transactions.

Why Apache Hive?

Hive is excellent for querying large datasets in a structured way, much like a SQL database. It’s particularly helpful for big data projects because:

- It enables SQL-like querying over large, distributed datasets.
- It’s scalable, allowing it to handle massive datasets effectively.
- It integrates well with Hadoop, a popular platform for big data storage and processing.

Creating a Hive table – Superstore Data:

CREATE TABLE sales(OrderDate TIMESTAMP, OrderQuantity INT, OrderDiscount FLOAT, UnitPrice FLOAT, UnitCost FLOAT, ShippingCost FLOAT, Province VARCHAR(20), CustomerType VARCHAR(20), ShipDate TIMESTAMP) ROW FORMAT DELIMITED FIELDS TERMINATED BY ', STORED AS TEXTFILE;

Run six complex queries on the superstore data. You should have:

- ***at least two predicates in in the WHERE clause of all queries***
- ***at least three queries should have a GROUP by clause***
- ***at least two queries should have an ORDER BY clause***
- ***at least one should have a UNION clause***

Query 1: Displays total sales of each customer that are either a consumer or home office type in descending order, grouped by customer type.

SELECT customertype, SUM((orderquantity*(unitprice-unitcost))-orderdiscount+shippingcost) AS TotalSales FROM week12.sales WHERE customertype='Consumer' OR customertype='Home Office' GROUP BY customertype ORDER BY TotalSales DESC;

Output:

Query 2: Displays total sales of each customer that have more than one total order and customers that have shipping cost greater than 15 dollars in descending order, grouped by customer type.

SELECT customertype, SUM((orderquantity*(unitprice-unitcost))-orderdiscount+shippingcost) AS TotalSales FROM week12.sales WHERE orderquantity>1 AND shippingcost>15 GROUP BY customertype ORDER BY TotalSales DESC;

Output:

```
0: jdbc:hive2://localhost:10000> SELECT customertype, SUM((orderquantity*(unitprice-unitcost))-orderdiscount+shippingcost) AS TotalSales FROM week12.sales WHERE orderquantity>1 AN
D shippingcost>15 GROUP BY customertype ORDER BY TotalSales DESC;
+----------------+--------------------+
|  customertype  |     totalsales     |
+----------------+--------------------+
| Corporate      | 279068.3529587444  |
| Home Office    | 188405.92953750677 |
| Consumer       | 172039.13548143767 |
| Small Business | 153832.85458520614 |
+----------------+--------------------+
4 rows selected (35.261 seconds)
```

Query 3: Displays total number of orders by provinces who are named either Nunavut or Ontario and orders placed by only corporate customers. All orders are grouped by province.

SELECT province, SUM(TotalOrders) as TotalOrders FROM (SELECT province, SUM(orderquantity) AS TotalOrders FROM week12.sales WHERE province='Nunavut' AND customertype='Corporate' GROUP BY province UNION SELECT province, SUM(orderquantity) AS TotalOrders FROM week12.sales WHERE province='Ontario' AND customertype='Corporate' GROUP BY province) sub GROUP BY province;

Output:

```
0: jdbc:hive2://localhost:10000> SELECT province, SUM(TotalOrders) as TotalOrders FROM (SELECT province, SUM(orderquantity) AS TotalOrders FROM week12.sales WHERE province='Nunavu
t' AND customertype='Corporate' GROUP BY province UNION SELECT province, SUM(orderquantity) AS TotalOrders FROM week12.sales WHERE province='Ontario' AND customertype='Corporate'
GROUP BY province) sub GROUP BY province;
+-----------+-------------+
| province  | totalorders |
+-----------+-------------+
| Nunavut   | 805         |
| Ontario   | 16243       |
+-----------+-------------+
2 rows selected (49.135 seconds)
```

Query 4: Displays total sales of each customer that have a total number of orders greater than 1 and total sales greater than zero. Of the total number of orders there needs to be no discount and the unit price of each needs to be greater than 150. Total sales grouped by customer type.

SELECT customertype, SUM((orderquantity * (unitprice-unitcost))-orderdiscount+shippingcost) AS TotalSales FROM week12.sales WHERE unitprice>150 AND orderdiscount=0 GROUP BY customertype HAVING TotalSales > 0 AND SUM(orderquantity) > 1;

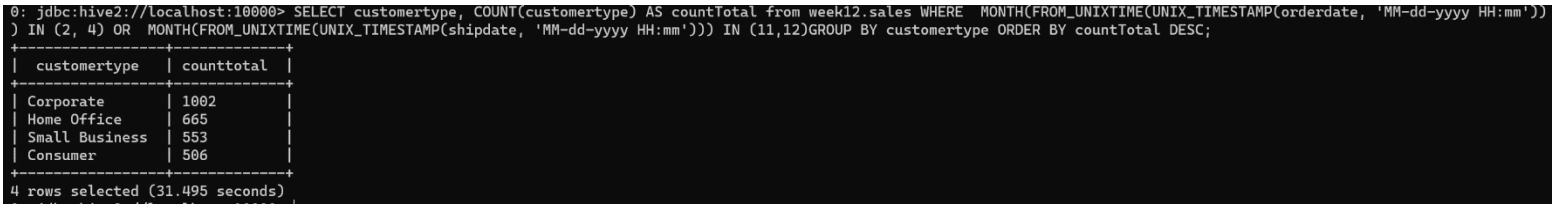
Output:

```
0: jdbc:hive2://localhost:10000> SELECT customertype, SUM((orderquantity * (unitprice-unitcost))-orderdiscount+shippingcost) AS TotalSales FROM week12.sales WHERE unitprice>150 AN
D orderdiscount=0 GROUP BY customertype HAVING TotalSales > 0 AND SUM(orderquantity) > 1;
+----------------+--------------------+
|  customertype  |     totalsales     |
+----------------+--------------------+
| Consumer       | 19782.161160469055 |
| Corporate      | 19988.849451303482 |
| Home Office    | 16145.127407073975 |
| Small Business | 14015.399411201477 |
+----------------+--------------------+
4 rows selected (18.702 seconds)
```

Query 5: Displays the summation of orders that were placed in either February or April and whose shipping date was in either November or December, aggregated by customer type and ordered in descending order by total orders.

```
SELECT customertype, COUNT(customertype) AS countTotal from week12.sales WHERE
MONTH(FROM_UNIXTIME(UNIX_TIMESTAMP(orderdate, 'MM-dd-yyyy HH:mm'))) IN (2, 4) OR
MONTH(FROM_UNIXTIME(UNIX_TIMESTAMP(shipdate, 'MM-dd-yyyy HH:mm'))) IN (11,12)GROUP BY
customertype ORDER BY countTotal DESC;
```

Output:

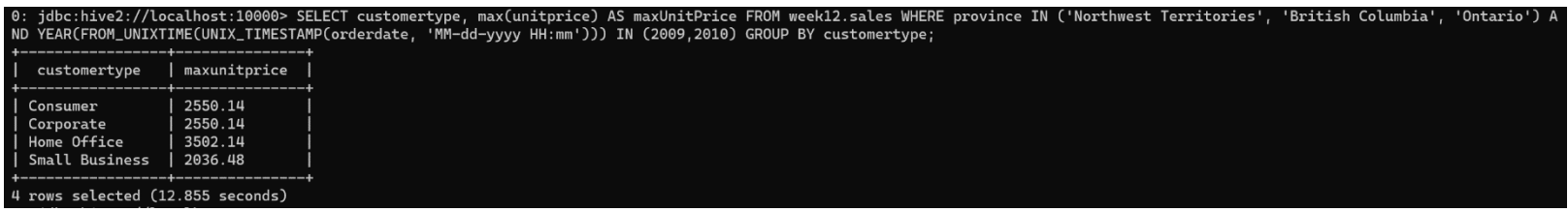


```
0: jdbc:hive2://localhost:10000> SELECT customertype, COUNT(customertype) AS countTotal from week12.sales WHERE  MONTH(FROM_UNIXTIME(UNIX_TIMESTAMP(orderdate, 'MM-dd-yyyy HH:mm'))
) IN (2, 4) OR  MONTH(FROM_UNIXTIME(UNIX_TIMESTAMP(shipdate, 'MM-dd-yyyy HH:mm'))) IN (11,12)GROUP BY customertype ORDER BY countTotal DESC;
+-----------------+-------------+
|  customertype   | counttotal  |
+-----------------+-------------+
| Corporate       | 1002        |
| Home Office     | 665         |
| Small Business  | 553         |
| Consumer        | 506         |
+-----------------+-------------+
4 rows selected (31.495 seconds)
```

Query 6: Displays the highest unit price placed by each customer whose province was in either the Northwest Territories, British Columbia, or Ontario and the order was placed in the year 2009 or 2010. Unit prices are grouped by customer type.

```
SELECT customertype, max(unitprice) AS maxUnitPrice FROM week12.sales WHERE province IN
('Northwest Territories', 'British Columbia', 'Ontario') AND
YEAR(FROM_UNIXTIME(UNIX_TIMESTAMP(orderdate, 'MM-dd-yyyy HH:mm'))) IN (2009,2010) GROUP
BY customertype;
```

Output:



```
0: jdbc:hive2://localhost:10000> SELECT customertype, max(unitprice) AS maxUnitPrice FROM week12.sales WHERE province IN ('Northwest Territories', 'British Columbia', 'Ontario') A
ND YEAR(FROM_UNIXTIME(UNIX_TIMESTAMP(orderdate, 'MM-dd-yyyy HH:mm'))) IN (2009,2010) GROUP BY customertype;
+-----------------+--------------+
|  customertype   | maxunitprice |
+-----------------+--------------+
| Consumer        | 2550.14      |
| Corporate       | 2550.14      |
| Home Office     | 3502.14      |
| Small Business  | 2036.48      |
+-----------------+--------------+
4 rows selected (12.855 seconds)
```