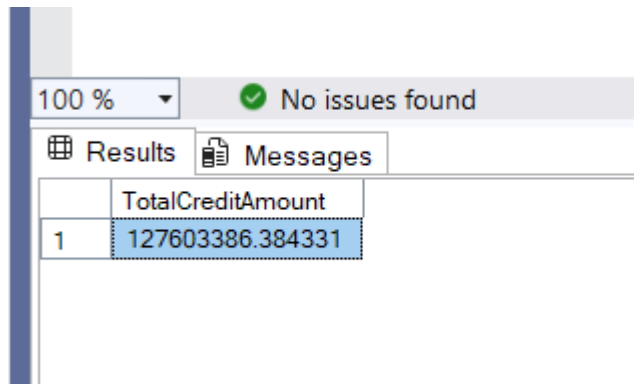


MySQL Query

1. Total Credit Amount

```
SELECT SUM(amount) AS total_credit_amount  
FROM transactions  
WHERE transaction_type = 'Credit';
```

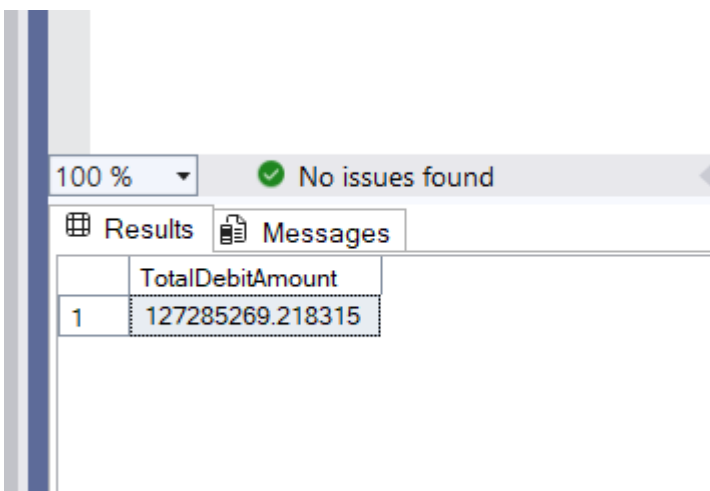


A screenshot of a MySQL query results window. At the top, there is a status bar showing '100 %' and a green checkmark with the text 'No issues found'. Below this are two tabs: 'Results' (active) and 'Messages'. The 'Results' tab displays a table with one column, 'TotalCreditAmount', and one row with the value '127603386.384331'.

	TotalCreditAmount
1	127603386.384331

2. Total Debit Amount

```
SELECT SUM(amount) AS total_debit_amount  
FROM transactions  
WHERE transaction_type = 'Debit';
```

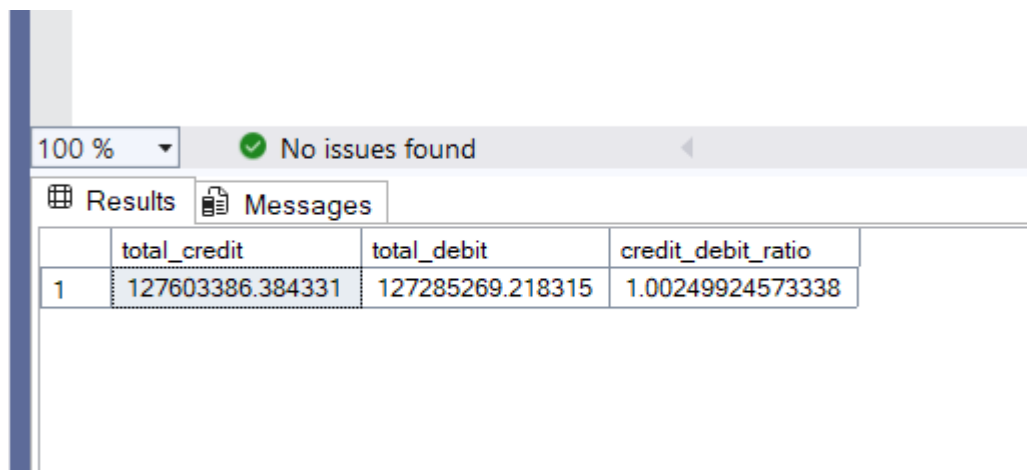


A screenshot of a MySQL query results window. At the top, there is a status bar showing '100 %' and a green checkmark with the text 'No issues found'. Below this are two tabs: 'Results' (active) and 'Messages'. The 'Results' tab displays a table with one column, 'TotalDebitAmount', and one row with the value '127285269.218315'.

	TotalDebitAmount
1	127285269.218315

3. Credit to Debit Ratio

```
WITH credits AS (  
    SELECT SUM(amount) AS total_credit FROM transactions WHERE transaction_type =  
    'Credit'  
),  
debits AS (  
    SELECT SUM(amount) AS total_debit FROM transactions WHERE transaction_type =  
    'Debit'  
)  
SELECT credits.total_credit, debits.total_debit,  
    CASE WHEN debits.total_debit = 0 THEN NULL  
    ELSE credits.total_credit / debits.total_debit END AS credit_debit_ratio  
FROM credits CROSS JOIN debits;
```

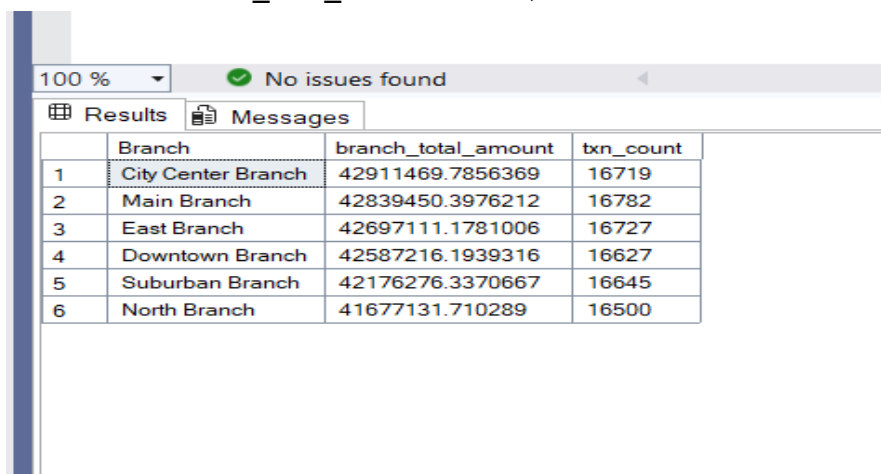


The screenshot shows a SQL query results window with a status bar at the top indicating '100 %' and 'No issues found'. Below the status bar are two tabs: 'Results' and 'Messages'. The 'Results' tab is active, displaying a table with four columns: an index column, 'total_credit', 'total_debit', and 'credit_debit_ratio'. There is one data row with the following values: 1, 127603386.384331, 127285269.218315, and 1.00249924573338.

	total_credit	total_debit	credit_debit_ratio
1	127603386.384331	127285269.218315	1.00249924573338

4. Total Transaction Amount by Branch

```
SELECT branch, SUM(amount) AS branch_total_amount, COUNT(*) AS txn_count  
FROM transactions  
GROUP BY branch  
ORDER BY branch_total_amount DESC;
```

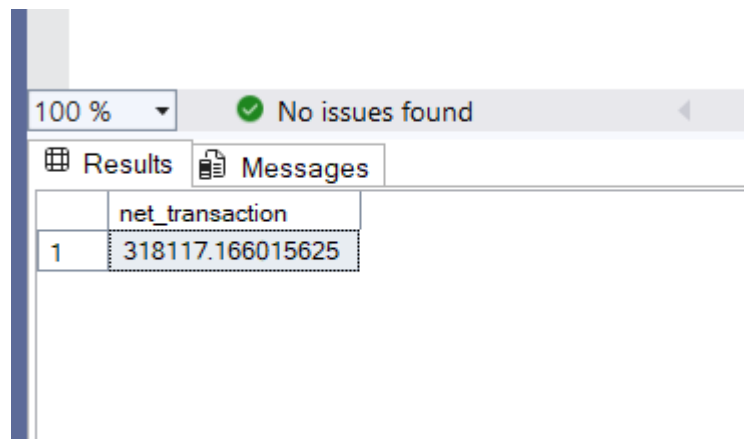


The screenshot shows a SQL query results window with a status bar at the top indicating '100 %' and 'No issues found'. Below the status bar are two tabs: 'Results' and 'Messages'. The 'Results' tab is active, displaying a table with four columns: an index column, 'Branch', 'branch_total_amount', and 'txn_count'. There are six data rows, each representing a different branch, ordered by total amount in descending order.

	Branch	branch_total_amount	txn_count
1	City Center Branch	42911469.7856369	16719
2	Main Branch	42839450.3976212	16782
3	East Branch	42697111.1781006	16727
4	Downtown Branch	42587216.1939316	16627
5	Suburban Branch	42176276.3370667	16645
6	North Branch	41677131.710289	16500

5. Net Transaction Amount

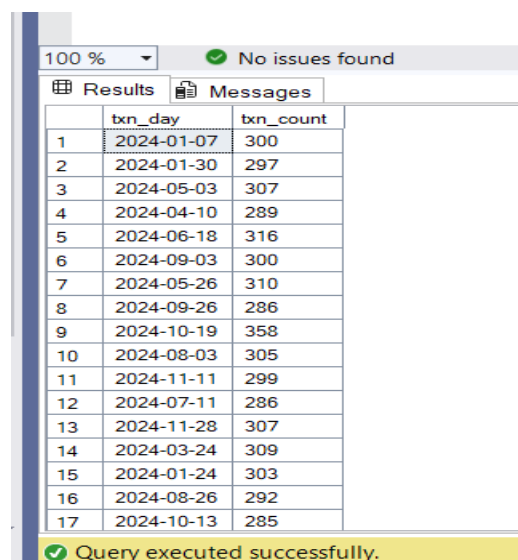
```
SELECT  
    SUM(CASE WHEN transaction_type = 'Credit' THEN amount ELSE 0 END) -  
    SUM(CASE WHEN transaction_type = 'Debit' THEN amount ELSE 0 END)  
    AS net_transaction  
FROM debit_and_credit;
```



	net_transaction
1	318117.166015625

6. Transactions per Day/Week/Month

```
SELECT DATE(transaction_date) AS txn_day, COUNT(*) AS txn_count  
FROM debit_and_credit  
GROUP BY DATE(transaction_date);
```

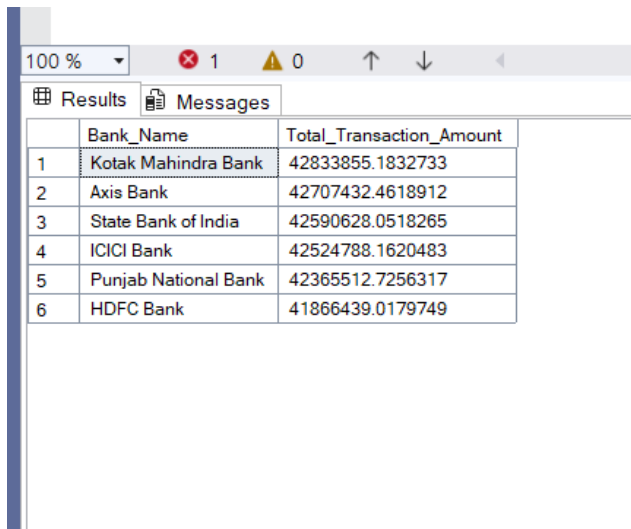


	txn_day	txn_count
1	2024-01-07	300
2	2024-01-30	297
3	2024-05-03	307
4	2024-04-10	289
5	2024-06-18	316
6	2024-09-03	300
7	2024-05-26	310
8	2024-09-26	286
9	2024-10-19	358
10	2024-08-03	305
11	2024-11-11	299
12	2024-07-11	286
13	2024-11-28	307
14	2024-03-24	309
15	2024-01-24	303
16	2024-08-26	292
17	2024-10-13	285

Query executed successfully.

7. Transaction Volume by Bank

```
SELECT
    `Bank Name`,
    SUM(Amount) AS Total_Transaction_Amount
FROM Debit_and_credit_master
GROUP BY `Bank Name`
ORDER BY Total_Transaction_Amount DESC;
```

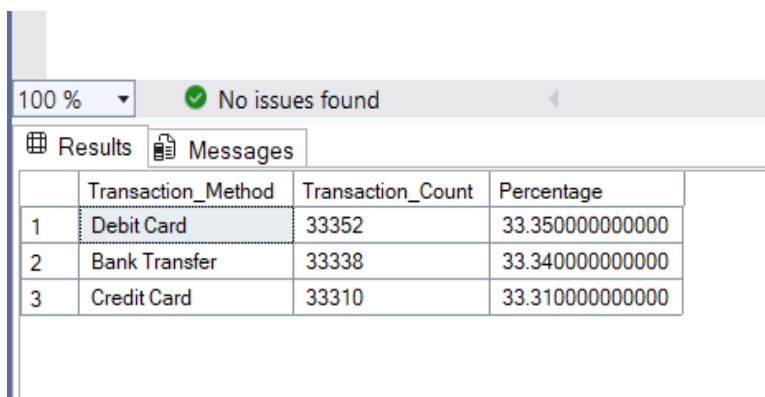


The screenshot shows a SQL query results window with a toolbar at the top indicating 100% zoom, 1 error, and 0 warnings. The 'Results' tab is active, displaying a table with two columns: 'Bank_Name' and 'Total_Transaction_Amount'. The table contains six rows of data, sorted in descending order of transaction amount.

	Bank_Name	Total_Transaction_Amount
1	Kotak Mahindra Bank	42833855.1832733
2	Axis Bank	42707432.4618912
3	State Bank of India	42590628.0518265
4	ICICI Bank	42524788.1620483
5	Punjab National Bank	42365512.7256317
6	HDFC Bank	41866439.0179749

8. Transaction Method Distribution

```
SELECT
    Transaction_Method,
    COUNT(*) AS Transaction_Count,
    ROUND(COUNT(*) * 100.0 / (SELECT COUNT(*) FROM Debit_and_credit_master), 2)
AS Percentage
FROM Debit_and_credit_master
GROUP BY Transaction_Method
ORDER BY Transaction_Count DESC;
```



The screenshot shows a SQL query results window with a toolbar at the top indicating 100% zoom and 'No issues found'. The 'Results' tab is active, displaying a table with four columns: 'Transaction_Method', 'Transaction_Count', and 'Percentage'. The table contains three rows of data, sorted in descending order of transaction count.

	Transaction_Method	Transaction_Count	Percentage
1	Debit Card	33352	33.3500000000000
2	Bank Transfer	33338	33.3400000000000
3	Credit Card	33310	33.3100000000000

9. Branch Transaction Growth

```

WITH MonthlyTotals AS (
    SELECT
        Branch,
        FORMAT(Transaction_Date, 'yyyy-MM') AS Month,
        SUM(Amount) AS Total_Amount
    FROM Debit_and_credit_master
    GROUP BY Branch, FORMAT(Transaction_Date, 'yyyy-MM')
)
SELECT
    Branch,
    Month,
    Total_Amount,
    ROUND(
        (Total_Amount - LAG(Total_Amount) OVER (
            PARTITION BY Branch
            ORDER BY Month
        )) * 100.0 / LAG(Total_Amount) OVER (
            PARTITION BY Branch
            ORDER BY Month
        ),
        2
    ) AS Growth_Percentage
FROM MonthlyTotals
ORDER BY Branch, Month;

```

100 % 2 0				
Results Messages				
	Branch	Month	Total_Amount	Growth_Percentage
1	City Center Branch	2024-01	4072785.45555115	NULL
2	City Center Branch	2024-02	3666383.87414551	-9.98
3	City Center Branch	2024-03	3977565.27716064	8.49
4	City Center Branch	2024-04	3703127.89434814	-6.9
5	City Center Branch	2024-05	3976262.71849823	7.38
6	City Center Branch	2024-06	3928220.92256927	-1.21
7	City Center Branch	2024-07	4025928.68029022	2.49
8	City Center Branch	2024-08	3791764.89493561	-5.82
9	City Center Branch	2024-09	3803875.92218781	0.32
10	City Center Branch	2024-10	3942428.66162872	3.64
11	City Center Branch	2024-11	3853654.91462708	-2.25
12	City Center Branch	2024-12	169470.569694519	-95.6
13	Downtown Branch	2024-01	3888779.08330536	NULL
14	Downtown Branch	2024-02	3571618.11341095	-8.16
15	Downtown Branch	2024-03	3984116.83317566	11.55
16	Downtown Branch	2024-04	3719658.97860718	-6.64
17	Downtown Branch	2024-05	3961467.33034515	6.5

10. High-Risk Transaction Flag

(Example: flagging transactions over ₹1,00,000)

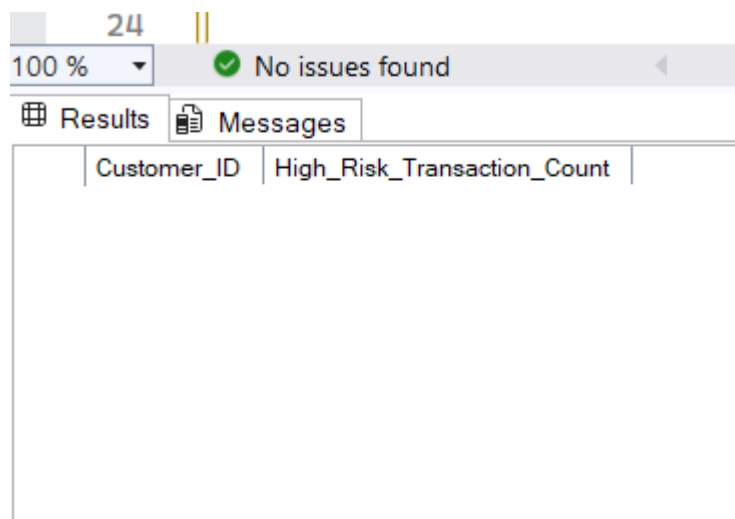
```
SELECT *,  
  
CASE  
  
    WHEN Amount > 100000 THEN 'High-Risk'  
  
    ELSE 'Normal'  
  
END AS Risk_Flag  
  
FROM Debit_and_credit_master;
```

Ln: 14 Ch: 30 SPC CRI				
	Transaction_Method	Currency	Bank_Name	Risk_Flag
	Credit Card	INR	Axis Bank	Normal
anch	Bank Transfer	INR	Kotak Mahindra Bank	Normal
nch	Bank Transfer	INR	Kotak Mahindra Bank	Normal
	Credit Card	INR	ICICI Bank	Normal
	Debit Card	INR	ICICI Bank	Normal
anch	Credit Card	INR	Kotak Mahindra Bank	Normal
nch	Debit Card	INR	Punjab National Bank	Normal
	Bank Transfer	INR	HDFC Bank	Normal
anch	Bank Transfer	INR	ICICI Bank	Normal
	Debit Card	INR	State Bank of India	Normal
nch	Debit Card	INR	Axis Bank	Normal
anch	Bank Transfer	INR	Axis Bank	Normal
	Bank Transfer	INR	ICICI Bank	Normal
anch	Credit Card	INR	HDFC Bank	Normal
nch	Debit Card	INR	Axis Bank	Normal
nch	Debit Card	INR	HDFC Bank	Normal

11. Suspicious Transaction Frequency

(Based on the high-risk flag above)

```
WITH flagged AS (  
    SELECT  
        *,  
        CASE  
            WHEN Amount > 100000 THEN 1  
            ELSE 0  
        END AS High_Risk  
    FROM Debit_and_credit_master  
)  
SELECT  
    Customer_ID,  
    COUNT(*) AS High_Risk_Transaction_Count  
FROM flagged  
WHERE High_Risk = 1  
GROUP BY Customer_ID  
ORDER BY High_Risk_Transaction_Count DESC;
```



The screenshot shows a SQL query execution interface. At the top, there is a status bar with a green checkmark and the text "No issues found". Below this, there are two tabs: "Results" and "Messages". The "Results" tab is active, and it displays a table with two columns: "Customer_ID" and "High_Risk_Transaction_Count". The table is currently empty, showing only the column headers.

Customer_ID	High_Risk_Transaction_Count
-------------	-----------------------------