

**INDIAN INSTITUTE OF TECHNOLOGY
GANDHINAGAR**

**SQL-BASED DATA
ENGINEERING PIPELINE
FOR BANKING ANALYTICS**



Ketki Gamre
B.Tech Mechanical Engineering

Contents

1	Abstract	2
2	Introduction	2
3	Objectives	2
4	Methodology	2
5	Database Schema	3
6	Implementation	3
6.1	Creating Tables	3
6.2	Data Cleaning and Transformation	3
6.3	Analytical View Creation	4
7	Results and Insights	4
8	Dashboard in Power BI	5
9	Conclusion	5
10	Future Scope	5

1 Abstract

This project focuses on designing an end-to-end **Data Engineering Pipeline** using SQL for a simulated banking environment. The aim is to extract, transform, and load (ETL) customer and transaction data to produce business insights through analytics and dashboards. The project demonstrates how data handling and reporting automation can support data-driven decision-making for the **Business Intelligence Unit (BIU)** of Axis Bank.

2 Introduction

Modern banking institutions rely heavily on data engineering and analytics to drive customer engagement, risk management, and profitability. This project replicates a simplified version of such an analytical workflow by building a structured SQL pipeline for customer and transaction data. It also connects the processed data to a business intelligence tool like Power BI for visualization.

3 Objectives

- To build a clean, scalable data engineering pipeline using SQL.
- To automate data extraction, transformation, and loading.
- To analyze customer transactions for business insights.
- To design a Power BI dashboard for visualization and reporting.

4 Methodology

The pipeline follows the standard ETL (Extract, Transform, Load) process:

1. **Extract:** Data is simulated using SQL tables for customers and transactions.
2. **Transform:** Data cleaning, aggregation, and business metric computation (credit, debit, net balance).
3. **Load:** The transformed data is stored in analytical views for visualization in Power BI.

5 Database Schema

Tables Used:

- customers(customer_id, name, age, city, income)
- transactions(txn_id, customer_id, txn_date, amount, txn_type)

Relationships:

- Each customer can have multiple transactions.
- transactions.customer_id is a foreign key referencing customers.customer_id.

6 Implementation

6.1 Creating Tables

```
CREATE TABLE customers (  
    customer_id INT PRIMARY KEY,  
    name VARCHAR(100),  
    age INT,  
    city VARCHAR(100),  
    income DECIMAL(10,2)  
);  
  
CREATE TABLE transactions (  
    txn_id INT PRIMARY KEY,  
    customer_id INT,  
    txn_date DATE,  
    amount DECIMAL(10,2),  
    txn_type VARCHAR(10),  
    FOREIGN KEY (customer_id) REFERENCES customers(customer_id)  
);
```

6.2 Data Cleaning and Transformation

```
SET SQL_SAFE_UPDATES = 0;  
  
CREATE TEMPORARY TABLE txn_keep_ids AS  
SELECT MIN(txn_id) AS keep_id  
FROM transactions  
GROUP BY customer_id, txn_date, amount, txn_type;
```

```
DELETE FROM transactions
WHERE txn_id NOT IN (SELECT keep_id FROM txn_keep_ids);

DROP TEMPORARY TABLE txn_keep_ids;
SET SQL_SAFE_UPDATES = 1;
```

6.3 Analytical View Creation

```
CREATE OR REPLACE VIEW customer_financial_summary AS
SELECT
    c.customer_id,
    c.name,
    c.city,
    c.income,
    SUM(CASE WHEN t.txn_type = 'credit' THEN t.amount ELSE 0 END) AS
        total_credit,
    SUM(CASE WHEN t.txn_type = 'debit' THEN t.amount ELSE 0 END) AS
        total_debit,
    SUM(CASE WHEN t.txn_type = 'credit' THEN t.amount ELSE 0 END) -
    SUM(CASE WHEN t.txn_type = 'debit' THEN t.amount ELSE 0 END) AS
        net_balance
FROM customers c
LEFT JOIN transactions t ON c.customer_id = t.customer_id
GROUP BY c.customer_id, c.name, c.city, c.income;
```

7 Results and Insights

The analytical view `customer_financial_summary` provides:

- Total credits and debits per customer.
- Net balance computation.
- City-wise income and spending insights.

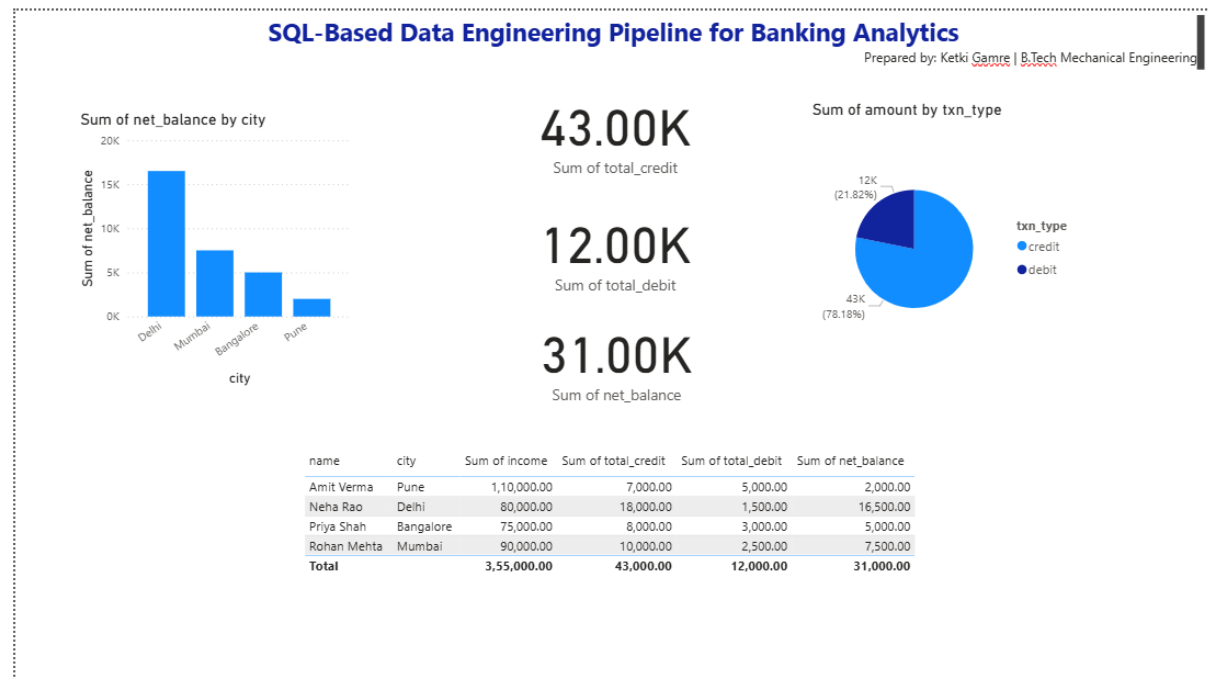
Sample Output:

Customer	City	Total Credit	Net Balance
Rohan Mehta	Mumbai	10000	7500
Priya Shah	Bangalore	8000	5000
Neha Rao	Delhi	9000	7500

8 Dashboard in Power BI

The Power BI dashboard includes:

- Bar chart: City vs Average Net Balance
- KPI cards: Total Credit, Total Debit
- Pie chart: Transaction type distribution
- Table: Customer-wise balance and income



9 Conclusion

This project demonstrates a simple yet powerful approach to automate data engineering and business reporting using SQL and Power BI. Such pipelines are critical in modern banking for risk management, marketing analytics, and operational efficiency.

10 Future Scope

- Integration with live APIs for real-time data ingestion.
- Use of Python for automated ETL scheduling.
- Cloud database deployment (AWS RDS, Azure SQL).