BANKING MANAGEMENT SYSTEM

TEAM

Pranay Prasad Pindi
Nitish Ahuja
Aayush Patel
Vamsi Gajja
Rakada Deva Rushi Kamidi

BACKGROUND

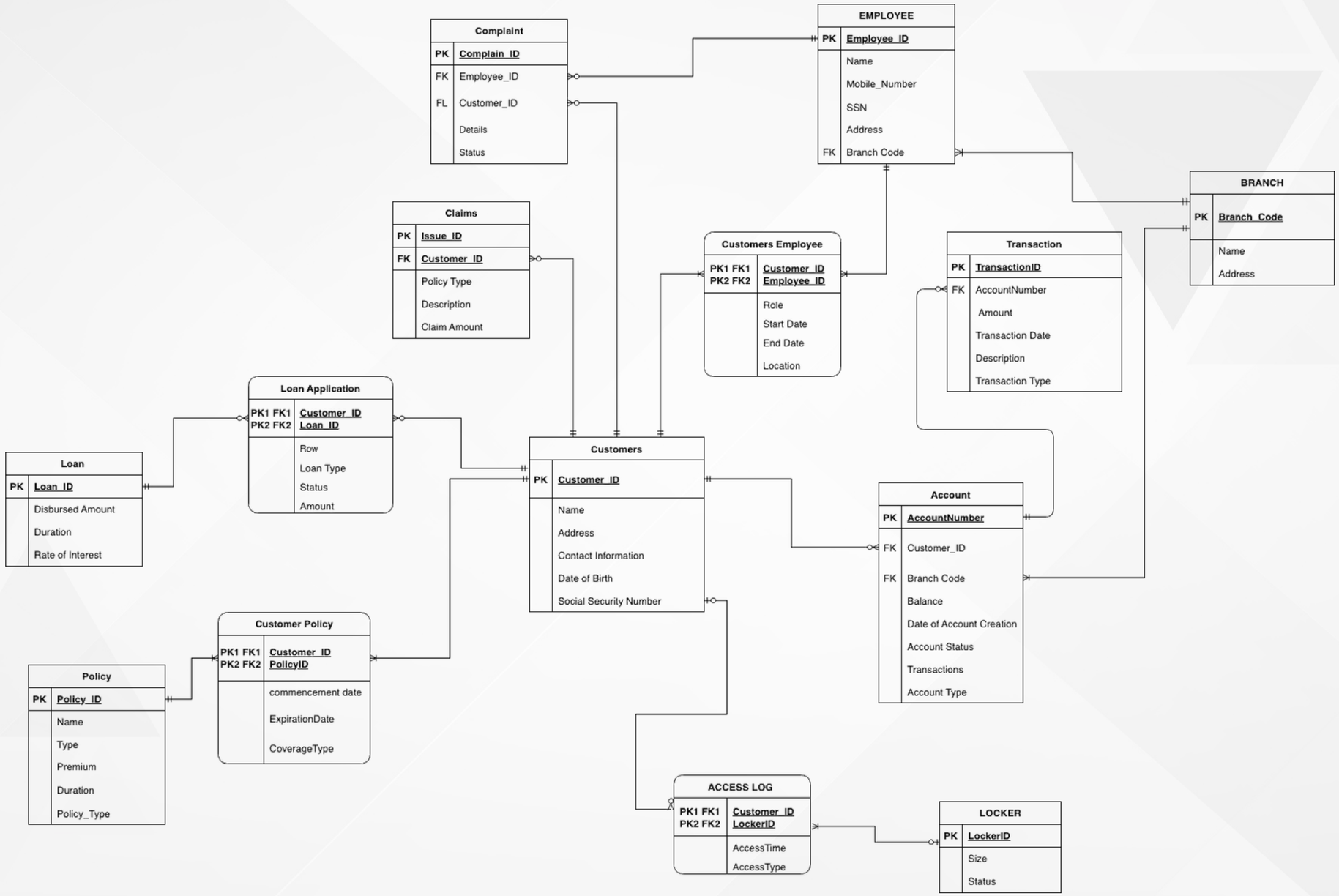
In the modern world, the banking sector plays a crucial role in economic progress and financial stability. Banks and financial institutions manage assets, financial transactions, and contribute to economic growth.

- **Evolution: From conventional to advanced banking systems.**
- **Challenges: Security and efficiency in managing physical records.**
- **Solution: Implementation of centralized databases for efficiency and security.**

HIGHLIGHTS

- **Enhanced Data Management:**
  - **Robust system for efficient storage and retrieval of customer data.**
  - **Emphasis on accuracy and security for handling sensitive information.**
- **Optimized Banking Operations:**
  - **Streamlined processes for improved efficiency.**
  - **Reduction in processing times and minimized transaction errors.**
- **Improved Customer Service Experiences:**
  - **User-friendly interfaces for a seamless experience.**
  - **Quick query resolution and personalized services for enhanced satisfaction.**
- **Operational Efficiency:**
  - **Significant improvements in processing times and error reduction.**
  - **Overall enhancement of operational efficiency.**
- **Security Measures:**
  - **Implementation of advanced security features.**
  - **Safeguarding sensitive financial information and transactions for both the bank and customers.**

ENTITY RELATIONSHIP DIAGRAM



DDL STATEMENTS

STORED PROCEDURES

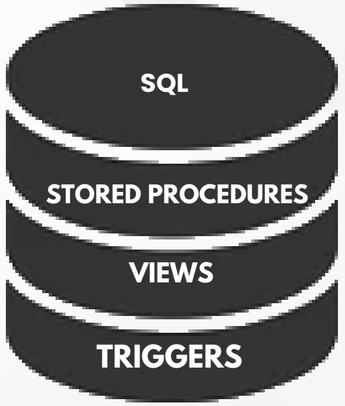
GetCustomerInfo Stored Procedure:
Retrieve customer information.
Parameters: @CustomerID, @CustomerName OUTPUT, @CustomerAddress OUTPUT, @TotalBalance OUTPUT.
Example: EXEC GetCustomerInfo @CustomerID = 201, @CustomerName OUTPUT, @CustomerAddress OUTPUT, @TotalBalance OUTPUT;
GetLoanStatus Stored Procedure:
Retrieve loan status and remaining amount.
Parameters: @LoanID, @LoanStatus OUTPUT, @RemainingAmount OUTPUT.
Example: EXEC GetLoanStatus @LoanID = 701, @LoanStatus OUTPUT, @RemainingAmount OUTPUT;
GetEmployeeDetails Stored Procedure:
Retrieve employee details.
Parameters: @EmployeeID, @EmployeeName OUTPUT, @EmployeeAddress OUTPUT, @EmployeeBranch OUTPUT.
Example: EXEC GetEmployeeDetails @EmployeeID = 501, @EmployeeName OUTPUT,

VIEWS

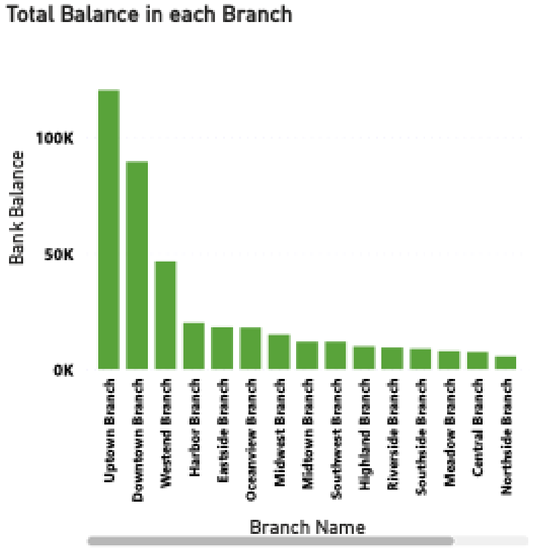
ActiveAccountsView: Account details for active accounts.
HighValueCustomersView: High-value customer details.
PolicySummaryView: Summary of customer policies.

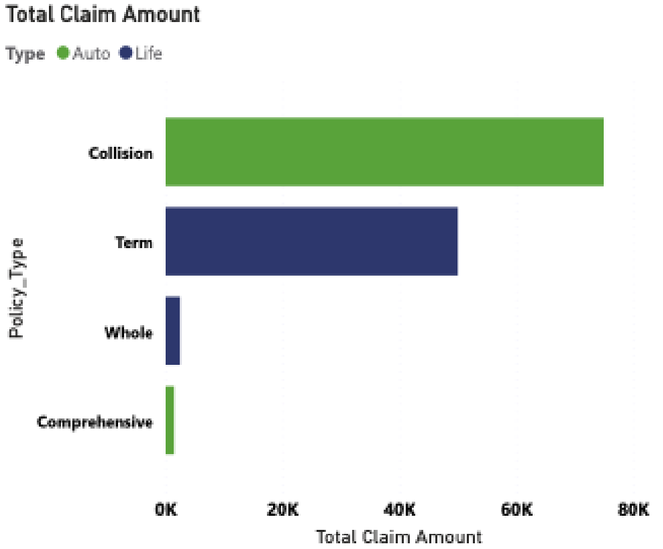
TRIGGERS

UpdateAccountBalanceTrigger
Updates account balance after each transaction.
Triggered AFTER INSERT on [Transaction].
Example transactions: Deposit and Withdrawal.



VISUALIZATION





Transaction Details

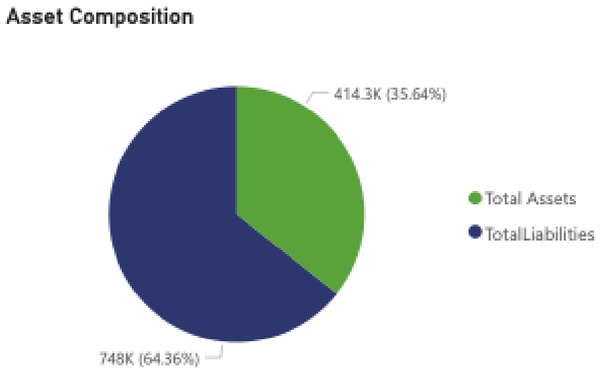
AverageTransactionAmount
115.36

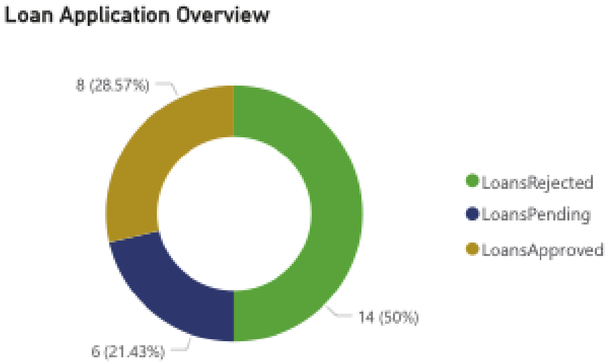
MaximumTransactionAmount
300.00

MinimumTransactionAmount
25.00

TotalTransactionAmount
4.85K

TotalTransactions
42





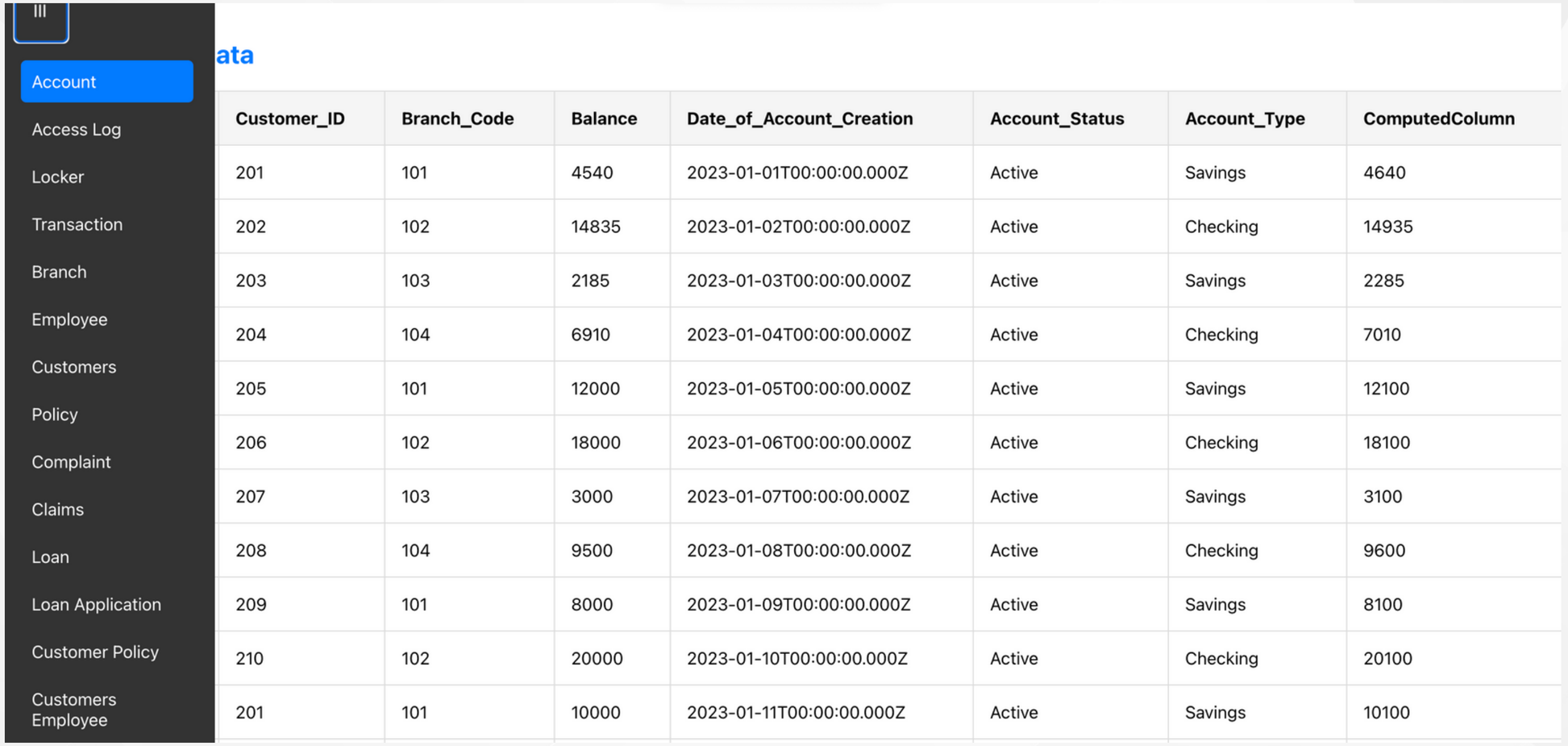
Locker Details

AvailableLockers
16

RentedLockers
12

DEMO

GUI



Customer_ID	Branch_Code	Balance	Date_of_Account_Creation	Account_Status	Account_Type	ComputedColumn
201	101	4540	2023-01-01T00:00:00.000Z	Active	Savings	4640
202	102	14835	2023-01-02T00:00:00.000Z	Active	Checking	14935
203	103	2185	2023-01-03T00:00:00.000Z	Active	Savings	2285
204	104	6910	2023-01-04T00:00:00.000Z	Active	Checking	7010
205	101	12000	2023-01-05T00:00:00.000Z	Active	Savings	12100
206	102	18000	2023-01-06T00:00:00.000Z	Active	Checking	18100
207	103	3000	2023-01-07T00:00:00.000Z	Active	Savings	3100
208	104	9500	2023-01-08T00:00:00.000Z	Active	Checking	9600
209	101	8000	2023-01-09T00:00:00.000Z	Active	Savings	8100
210	102	20000	2023-01-10T00:00:00.000Z	Active	Checking	20100
201	101	10000	2023-01-11T00:00:00.000Z	Active	Savings	10100

THANK YOU

ANY QUESTIONS?

MADE WITH ❤ AND SQL