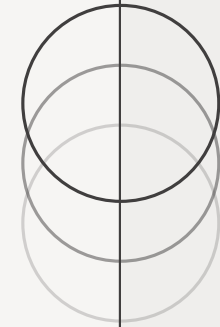


AKTAN DAIROV



BANK ACCOUNTING SYSTEM

DATABASE PROJECT

ESENALIEVA GULZAT

PAGE 1



Project Overview

PURPOSE

Create a relational database for banking operations

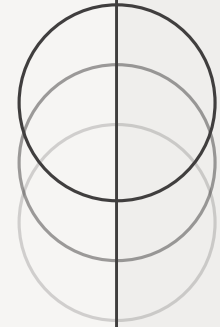
FEATURES IMPLEMENTED

CUSTOMER MANAGEMENT
ACCOUNT MANAGEMENT
DEPOSITS/WITHDRAWALS /TRANSFERS
TRANSACTION HISTORY
VIEW, TRIGGERS, PROCEDURES
DATA INTEGRITY & CONSTRAINTS

Problem Statement

BANKING SYSTEMS REQUIRE

- RELIABLE STORAGE OF CUSTOMER AND ACCOUNT DATA
 - ACCURATE TRACKING OF FINANCIAL TRANSACTIONS
 - PREVENTION OF INVALID OPERATIONS (E.G., NEGATIVE BALANCE)
 - SECURE AND AUDITABLE HISTORY OF ALL ACTIONS
-



System Requirements

FUNCTIONAL REQUIREMENTS

- CREATE CUSTOMER
- CREATE BANK ACCOUNT
- PERFORM DEPOSIT
- PERFORM WITHDRAWAL (WITH BALANCE CHECK)
- TRANSFER FUNDS BETWEEN ACCOUNTS
- SHOW TRANSACTION HISTORY

NON-FUNCTIONAL

- DATA INTEGRITY
- CONSISTENCY
- TRANSACTION SAFETY
- SECURE PERMISSIONS

Database Architecture

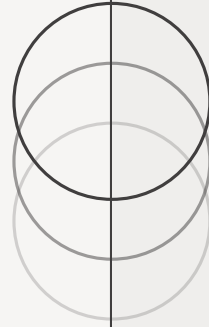


MAIN ENTITIES

- CUSTOMERS
- ACCOUNTS
- TRANSACTIONS

RELATIONSHIPS

- 1 CUSTOMER → MANY ACCOUNTS
- 1 ACCOUNT → MANY TRANSACTIONS

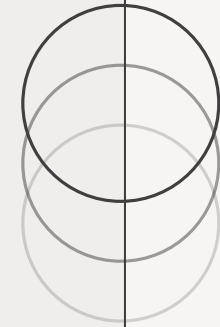


MAY 2030

Performance Overview

PAID ADVERTISING

PAGE 6



Database Schema

TABLES CREATED

- CUSTOMERS
 - ACCOUNTS
 - TRANSACTIONS
-

MENTION PURPOSE OF EACH TABLE:

- CUSTOMERS → PERSONAL INFO
 - ACCOUNTS → FINANCIAL STATE
 - TRANSACTIONS → EVERY OPERATION STORED FOREVER
-

SQL File Structure



01_CREATE_DATABASE.SQL

02_TABLES.SQL

03_CONSTRAINTS.SQL

04_INSERT_SAMPLE_DATA.SQL

05_STORED_PROCEDURES.SQL

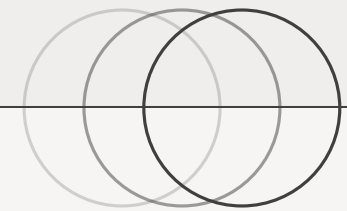
06_TRIGGERS.SQL

07_VIEWS.SQL

08_EXAMPLE_QUERIES.SQL

09_PERMISSIONS.SQL

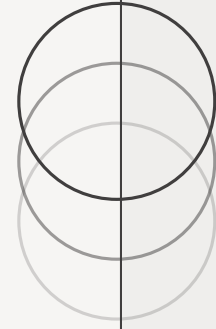
Constraints Implemented



- UNIQUE(PASSPORT_ID)
- CHECK(BALANCE >= 0)
- FOREIGN KEY(CUSTOMER_ID → CUSTOMERS)
- FOREIGN KEY(ACCOUNT_ID → ACCOUNTS)

Why needed?

- PREVENT DUPLICATES
- PREVENT INVALID DATA
- ENSURE REFERENTIAL INTEGRITY



THANK YOU

Q&A