ENTERPRISE DATABASE DESIGN

Core Digital Transformation for Retail Banking

Conceptual & Logical ERD | Normalization | SQL DDL | Data Dictionary

Project	Prepared By	Date
Core Digital Transformation for Retail Banking	Business Analyst / Data Architect	February 18, 2026

Task 1: Identify Key Banking Data Entities

The following core and supporting entities form the foundation of the enterprise banking database. Each entity maps directly to a key business domain within the retail banking platform.

Entity Name	Description
Customer	Stores personal information, KYC details, contact data, and identity verification status for each retail banking customer.
Account	Represents checking, savings, or business accounts linked to a customer. Tracks balance, account type, and status.
Loan	Captures loan product details including principal, interest rate, term, repayment schedule, and delinquency status.
Transaction	Records all financial activity including deposits, withdrawals, transfers, card payments, and ATM usage with timestamps.
Compliance Log	Audit trail for KYC events, AML flags, login attempts, GDPR data access requests, and regulatory reporting activities.
Credit Score	Stores credit bureau scores, risk ratings, and assessment dates for each customer — supports loan eligibility decisions.
Branch	Represents physical or virtual bank branches including location, IFSC code, region, and assigned operations manager.

Task 2: Conceptual ERD

The conceptual ERD below illustrates the high-level relationships between core banking entities without implementation-level attributes. It shows entity connections and cardinalities across the retail banking domain.

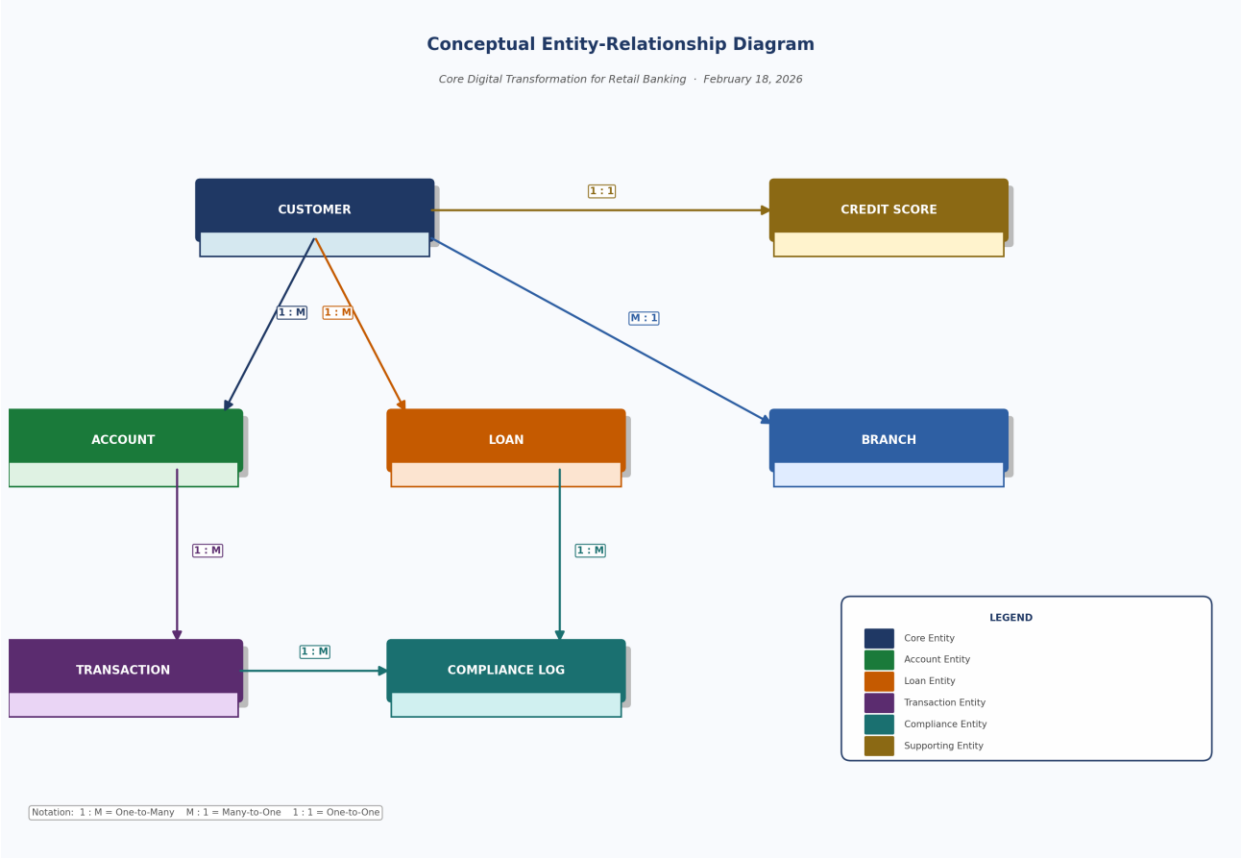

Figure 1: Conceptual Entity-Relationship Diagram — Core Digital Banking Platform

Relationship Summary

From Entity	Cardinality	To Entity
Customer	1 : M (One-to-Many)	Account — one customer can hold multiple accounts
Customer	1 : M (One-to-Many)	Loan — one customer can have multiple loans
Account	1 : M (One-to-Many)	Transaction — one account can have many transactions
Transaction	1 : M (One-to-Many)	Compliance Log — each transaction can trigger compliance events
Loan	1 : M (One-to-Many)	Compliance Log — loan activity generates audit and AML log entries
Customer	1 : 1 (One-to-One)	Credit Score — each customer has one active credit profile
Customer	M : 1 (Many-to-One)	Branch — multiple customers belong to one branch

Task 3: Logical ERD with Attributes and Keys

The logical ERD defines entities with their complete attribute sets, primary keys (PK), foreign keys (FK), data types, and cardinality constraints. This forms the blueprint for physical database implementation.

CUSTOMER Entity

CUSTOMER	Data Type	Constraints
customer_id	UUID (PK)	Primary Key
first_name	VARCHAR(50)	NOT NULL
last_name	VARCHAR(50)	NOT NULL
dob	DATE	NOT NULL
email	VARCHAR(100) UNIQUE	UNIQUE, NOT NULL
phone	VARCHAR(20)	NOT NULL
kyc_status	VARCHAR(20)	CHECK IN ('PENDING','VERIFIED','REJECTED')
address	TEXT	NOT NULL
created_at	TIMESTAMP	DEFAULT NOW()

ACCOUNT Entity

ACCOUNT	Data Type	Constraints
account_id	UUID (PK)	Primary Key
customer_id	UUID (FK)	FK → Customer
account_type	VARCHAR(30)	CHECK IN ('SAVINGS','CHECKING','BUSINESS')
balance	DECIMAL(18,2)	DEFAULT 0.00, NOT NULL
currency	CHAR(3)	DEFAULT 'USD'
status	VARCHAR(20)	CHECK IN ('ACTIVE','FROZEN','CLOSED')
opened_at	TIMESTAMP	DEFAULT NOW()
branch_id	UUID (FK)	FK → Branch

LOAN Entity

LOAN	Data Type	Constraints
loan_id	UUID (PK)	Primary Key
customer_id	UUID (FK)	FK → Customer
loan_type	VARCHAR(30)	CHECK IN ('HOME','PERSONAL','AUTO','BUSINESS')

principal	DECIMAL(18,2)	NOT NULL
interest_rate	DECIMAL(5,4)	NOT NULL
term_months	INT	NOT NULL
status	VARCHAR(20)	CHECK IN ('ACTIVE','CLOSED','DELINQUENT')
disbursed_at	TIMESTAMP	NOT NULL

TRANSACTION Entity

TRANSACTION	Data Type	Constraints
transaction_id	UUID (PK)	Primary Key
account_id	UUID (FK)	FK → Account
transaction_type	VARCHAR(30)	CHECK IN ('DEPOSIT','WITHDRAWAL','TRANSFER','PAYMENT')
amount	DECIMAL(18,2)	NOT NULL
currency	CHAR(3)	DEFAULT 'USD'
status	VARCHAR(20)	CHECK IN ('SUCCESS','PENDING','FAILED')
reference_no	VARCHAR(50)	UNIQUE
created_at	TIMESTAMP	DEFAULT NOW()

COMPLIANCE LOG Entity

COMPLIANCE LOG	Data Type	Constraints
log_id	UUID (PK)	Primary Key
entity_type	VARCHAR(30)	CHECK IN ('TRANSACTION','LOAN','CUSTOMER')
entity_id	UUID	NOT NULL
event_type	VARCHAR(50)	NOT NULL
severity	VARCHAR(20)	CHECK IN ('LOW','MEDIUM','HIGH','CRITICAL')
description	TEXT	NOT NULL
created_by	VARCHAR(100)	NOT NULL
created_at	TIMESTAMP	DEFAULT NOW()

CREDIT SCORE Entity

CREDIT SCORE	Data Type	Constraints
score_id	UUID (PK)	Primary Key

customer_id	UUID (FK)	FK → Customer, UNIQUE
bureau_name	VARCHAR(50)	NOT NULL
score	INT	CHECK (score BETWEEN 300 AND 900)
risk_rating	VARCHAR(20)	CHECK IN ('LOW','MEDIUM','HIGH')
assessed_at	TIMESTAMP	DEFAULT NOW()
valid_until	DATE	NOT NULL

BRANCH Entity

BRANCH	Data Type	Constraints
branch_id	UUID (PK)	Primary Key
branch_name	VARCHAR(100)	NOT NULL
ifsc_code	VARCHAR(20)	UNIQUE, NOT NULL
city	VARCHAR(50)	NOT NULL
region	VARCHAR(50)	NOT NULL
country	VARCHAR(50)	NOT NULL
manager_name	VARCHAR(100)	
is_active	BOOLEAN	DEFAULT TRUE

Task 4: Apply Normalization (1NF → 2NF → 3NF)

Each normalization step below documents the transformation applied to the banking schema to eliminate redundancy and ensure data integrity.

Step	Action Taken	Example
1NF	Ensured all columns contain atomic (indivisible) values. Removed multi-valued fields — e.g., a customer's multiple phone numbers stored in a single column were split into a separate PhoneNumbers table. Ensured each table has a defined primary key.	customer.phone split from a comma-separated list into separate rows with customer_id as FK. Every table assigned a UUID primary key.
2NF	Eliminated partial dependencies. Ensured all non-key attributes depend on the full primary key, not just part of it. Applied to composite-key tables by separating attributes that depend on only one part of the composite key.	In a hypothetical OrderItem table with (order_id, product_id) as PK, product_name depended only on product_id — moved to a Products table. In Transactions, account_type was moved to Accounts.
3NF	Removed transitive dependencies. Ensured non-key attributes depend only on the primary key, not on other non-key attributes. Extracted any derived or indirectly dependent data into separate reference tables.	In Customer, branch_city and branch_region transitively depended on branch_id — extracted to a dedicated Branch table. Credit Score data separated from Customer to its own table since it depends on bureau assessment, not just customer identity.

Task 5: SQL DDL Scripts

The following SQL DDL scripts create the core banking tables with primary keys, foreign keys, indexes, and compliance-ready constraints. All scripts use PostgreSQL syntax.

Customers Table

SQL DDL — customers

```
CREATE TABLE customers (
    customer_id   UUID PRIMARY KEY DEFAULT gen_random_uuid(),
    first_name    VARCHAR(50) NOT NULL,
    last_name     VARCHAR(50) NOT NULL,
    dob           DATE NOT NULL,
    email         VARCHAR(100) UNIQUE NOT NULL,
    phone         VARCHAR(20) NOT NULL,
    kyc_status    VARCHAR(20) NOT NULL
                  CHECK (kyc_status IN ('PENDING','VERIFIED','REJECTED')),
    address       TEXT NOT NULL,
    created_at    TIMESTAMP NOT NULL DEFAULT NOW()
);
CREATE INDEX idx_customers_email ON customers(email);
CREATE INDEX idx_customers_kyc   ON customers(kyc_status);
```

Accounts Table

SQL DDL — accounts

```
CREATE TABLE accounts (
    account_id    UUID PRIMARY KEY DEFAULT gen_random_uuid(),
    customer_id   UUID NOT NULL REFERENCES customers(customer_id) ON DELETE
RESTRICT,
    branch_id     UUID NOT NULL REFERENCES branches(branch_id),
    account_type  VARCHAR(30) NOT NULL
                  CHECK (account_type IN ('SAVINGS','CHECKING','BUSINESS')),
    balance       DECIMAL(18,2) NOT NULL DEFAULT 0.00,
    currency      CHAR(3) NOT NULL DEFAULT 'USD',
    status        VARCHAR(20) NOT NULL
                  CHECK (status IN ('ACTIVE','FROZEN','CLOSED')),
    opened_at     TIMESTAMP NOT NULL DEFAULT NOW()
);
CREATE INDEX idx_accounts_customer ON accounts(customer_id);
CREATE INDEX idx_accounts_status   ON accounts(status);
```

Loans Table

SQL DDL — loans

```
CREATE TABLE loans (
    loan_id       UUID PRIMARY KEY DEFAULT gen_random_uuid(),
    customer_id   UUID NOT NULL REFERENCES customers(customer_id),
    loan_type     VARCHAR(30) NOT NULL
                  CHECK (loan_type IN ('HOME','PERSONAL','AUTO','BUSINESS')),
    principal     DECIMAL(18,2) NOT NULL,
    interest_rate DECIMAL(5,4) NOT NULL,
    term_months   INT NOT NULL,
    status        VARCHAR(20) NOT NULL
```

```
                    CHECK (status IN ('ACTIVE','CLOSED','DELINQUENT')),
    disbursed_at    TIMESTAMP NOT NULL
);
CREATE INDEX idx_loans_customer ON loans(customer_id);
CREATE INDEX idx_loans_status   ON loans(status);
```

Transactions Table

SQL DDL — transactions

```
CREATE TABLE transactions (
    transaction_id   UUID PRIMARY KEY DEFAULT gen_random_uuid(),
    account_id       UUID NOT NULL REFERENCES accounts(account_id),
    transaction_type VARCHAR(30) NOT NULL
                     CHECK (transaction_type IN
                     ('DEPOSIT','WITHDRAWAL','TRANSFER','PAYMENT')),
    amount           DECIMAL(18,2) NOT NULL,
    currency         CHAR(3) NOT NULL DEFAULT 'USD',
    status           VARCHAR(20) NOT NULL
                     CHECK (status IN ('SUCCESS','PENDING','FAILED')),
    reference_no     VARCHAR(50) UNIQUE NOT NULL,
    created_at       TIMESTAMP NOT NULL DEFAULT NOW()
);
CREATE INDEX idx_transactions_account ON transactions(account_id);
CREATE INDEX idx_transactions_date    ON transactions(created_at);
```

Compliance Logs Table

SQL DDL — compliance_logs

```
CREATE TABLE compliance_logs (
    log_id        UUID PRIMARY KEY DEFAULT gen_random_uuid(),
    entity_type   VARCHAR(30) NOT NULL
                  CHECK (entity_type IN ('TRANSACTION','LOAN','CUSTOMER')),
    entity_id     UUID NOT NULL,
    event_type    VARCHAR(50) NOT NULL,
    severity      VARCHAR(20) NOT NULL
                  CHECK (severity IN ('LOW','MEDIUM','HIGH','CRITICAL')),
    description   TEXT NOT NULL,
    created_by    VARCHAR(100) NOT NULL,
    created_at    TIMESTAMP NOT NULL DEFAULT NOW()
);
CREATE INDEX idx_compliance_entity   ON compliance_logs(entity_id);
CREATE INDEX idx_compliance_severity ON compliance_logs(severity);
CREATE INDEX idx_compliance_date     ON compliance_logs(created_at);
```

Task 6: Data Dictionary

The data dictionary provides a comprehensive reference for all fields across core banking tables, documenting field purpose, data types, and constraints for governance and developer onboarding.

Table: customers

Field	Data Type	Description	Constraints
customer_id	UUID	Unique customer identifier (PK)	PRIMARY KEY, NOT NULL
first_name	VARCHAR(50)	Customer's legal first name	NOT NULL
last_name	VARCHAR(50)	Customer's legal last name	NOT NULL
dob	DATE	Date of birth for KYC verification	NOT NULL
email	VARCHAR(100)	Primary contact email address	UNIQUE, NOT NULL
kyc_status	VARCHAR(20)	KYC verification state	CHECK IN ('PENDING','VERIFIED','REJECTED')
created_at	TIMESTAMP	Record creation timestamp	DEFAULT NOW()

Table: accounts

Field	Data Type	Description	Constraints
account_id	UUID	Unique account identifier (PK)	PRIMARY KEY, NOT NULL
customer_id	UUID	Reference to owning customer (FK)	FK → customers, NOT NULL
account_type	VARCHAR(30)	Type of bank account	CHECK IN ('SAVINGS','CHECKING','BUSINESS')
balance	DECIMAL(18,2)	Current account balance	DEFAULT 0.00, NOT NULL
status	VARCHAR(20)	Account lifecycle state	CHECK IN ('ACTIVE','FROZEN','CLOSED')
opened_at	TIMESTAMP	Account opening timestamp	DEFAULT NOW()

Table: loans

Field	Data Type	Description	Constraints
loan_id	UUID	Unique loan identifier (PK)	PRIMARY KEY, NOT NULL

customer_id	UUID	Reference to borrower (FK)	FK → customers, NOT NULL
principal	DECIMAL(18,2)	Original loan amount disbursed	NOT NULL
interest_rate	DECIMAL(5,4)	Annual interest rate as decimal	NOT NULL
term_months	INT	Loan duration in months	NOT NULL
status	VARCHAR(20)	Loan repayment state	CHECK IN ('ACTIVE','CLOSED','DELINQUENT')

Table: transactions

Field	Data Type	Description	Constraints
transaction_id	UUID	Unique transaction identifier (PK)	PRIMARY KEY, NOT NULL
account_id	UUID	Reference to associated account (FK)	FK → accounts, NOT NULL
transaction_type	VARCHAR(30)	Nature of the financial operation	CHECK IN ('DEPOSIT','WITHDRAWAL','TRANSFER','PAYMENT')
amount	DECIMAL(18,2)	Transaction monetary value	NOT NULL
status	VARCHAR(20)	Transaction processing state	CHECK IN ('SUCCESS','PENDING','FAILED')
reference_no	VARCHAR(50)	Unique external reference number	UNIQUE, NOT NULL
created_at	TIMESTAMP	Transaction timestamp	DEFAULT NOW()

Table: compliance_logs

Field	Data Type	Description	Constraints
log_id	UUID	Unique audit log entry identifier (PK)	PRIMARY KEY, NOT NULL
entity_type	VARCHAR(30)	Type of entity being audited	CHECK IN ('TRANSACTION','LOAN','CUSTOMER')

entity_id	UUID	ID of the audited entity	NOT NULL
event_type	VARCHAR(50)	Category of compliance event (e.g., AML_FLAG)	NOT NULL
severity	VARCHAR(20)	Risk severity of the compliance event	CHECK IN ('LOW','MEDIUM','HIGH','CRITICAL')
created_by	VARCHAR(100)	System or user that triggered the log	NOT NULL
created_at	TIMESTAMP	Log creation timestamp	DEFAULT NOW()

Task 7: Database Design Validation Checklist

Validation Item	Yes / No
All core entities identified (Customer, Account, Loan, Transaction, Compliance Log + 2 additional)	**YES**
Conceptual ERD complete — entities and relationships shown without attributes	**YES**
Logical ERD includes cardinalities, primary keys (PK), and foreign keys (FK)	**YES**
Normalization clearly shown — 1NF, 2NF, and 3NF steps documented with examples	**YES**
SQL includes PK, FK, indexes, and CHECK constraints for all tables	**YES**
Data dictionary complete with field descriptions, data types, and constraints	**YES**
Compliance fields included (KYC status on Customer, AML flags in Compliance Log, severity levels)	**YES**