DATABASE DESIGN FOR BANKING ENTERPRISE

Contents

DATABASE DESIGN FOR BANKING ENTERPRISE

IMPORTANT POINTS

We apply the two initial database – design phases, namely

- The gathering of data requirements
- Design of the conceptual schema

DATA REQUIREMENTS FOR THE BANK DATABASE

The initial specification of user requirements may be based on interviews with the database users and on the designer's own analysis of the enterprise. The description that arises from this design phase serves as the basis for specifying the conceptual structure of the database. Here are the major characteristics of the banking enterprise:

- The bank is organized into branches. Each branch is located in a particular city and is identified by a unique name. The bank monitors the assets of each branch.

- Bank customers are identified by their *customer_id* values. The bank stores each customer's name and the street and city where the customer lives. Customers may have accounts and can take out loans. A customer may be associated with a particular banker, who may act as a loan officer or personal banker for thar customer.

- Bank employees are identified by their *employee_id* values. The bank administration stores the name and telephone number of each employee, the names of the employee's dependents, and the employee id number of the employee's manager. The bank also keeps track of the employee's start date and, thus, length of employment.

- The bank offers two types of accounts-savings and checking accounts. Accounts can be held by more than one customer, and a customer can have more than one account. Each account is assigned a unique account number. The bank maintains a record of each account's balance and the most recent date on which the account was accessed by each customer holding the account. In addition, each savings account has an interest rate and overdrafts are recorded for each checking account.

- A loan originates at a particular branch and can be held by one or more customers. A loan is identified by a unique loan number. For each loan, the bank keeps track of the loan amount and the loan payments. Although a loan payment number does not uniquely identify a particular payment among those for all the bank's loans, a payment number does identify a particular payment for a specific loan. The date and amount are recorded for each payment.

In a real banking enterprise, the bank would keep track of deposits and withdrawals from savings and checking accounts, just as it keeps track of payments to loan accounts. Since the modeling requirements for that tracking are similar, and we would like to keep our example application small, we do not keep track of such de posits and withdrawals in our model.

SYSTEM REQUIREMENTS

The BANK database typically models banking operations, such as accounts, customers, loans, and branches. Below are the inferred system requirements based on its design and functionality:

Functional Requirements

- Customer Management:
    - Store and manage customer information such as name, address, and customer ID.
    - Link customers to their accounts and loans.
- Branch Management:
    - Maintain branch-specific data, including branch name, branch ID, and location.
    - Track branch-level operations like deposits and loans.
- Account Management:
    - Maintain account information (account number, type, balance).
    - Link accounts to specific branches and customers.
    - Support account operations such as deposits, withdrawals, and balance queries.
- Loan Management:
    - Store and manage loan details (loan number, amount, and payments).
    - Associate loans with branches and customers.
- Transaction Support:
    - Record transactions on accounts (deposits, withdrawals, transfers).
    - Log payments made against loans.
- Relational Queries:
    - Retrieve information using SQL queries such as:
        - Finding customers with accounts in specific branches.
        - Aggregating account balances or loan amounts.
        - Listing branches with specific types of loans.

Non-Functional Requirements

- Performance:
    - Efficient query handling for large-scale banking operations.
- Reliability:
    - Ensure data integrity during concurrent transactions.
- Scalability:

  - Support for adding new branches, accounts, or customers without significant performance degradation.
- Security:
  - Restrict access to sensitive data, such as account balances or loan amounts, to authorized users.
- Backup and Recovery:
  - Regular backups of the database to prevent data loss.
  - Recovery mechanisms to restore the database after failures.

Schema Requirements

The schema for the BANK database typically includes the following tables:

- Branch:
  - Attributes: branch_name, branch_city, assets.
- Customer:
  - Attributes: customer_name, customer_street, customer_city.
- Account:
  - Attributes: account_number, branch_name, balance.
- Depositor (Relationship between Customer and Account):
  - Attributes: customer_name, account_number.
- Loan:
  - Attributes: loan_number, branch_name, amount.
- Borrower (Relationship between Customer and Loan):
  - Attributes: customer_name, loan_number.

Additional Considerations

- Normalization: Ensure the schema is normalized to eliminate redundancy.
- Concurrency Control: Support for ACID properties in transactions.
- Indexing: Use indexes for efficient query execution, such as on primary and foreign keys.

INSTANCES

TABLE: The ***account*** relation with ordered tuples

account_number	***branch_name***	***balance***
A-101	Downtown	500
A-102	Perryridge	400
A-201	Brighton	700
A-215	Mianus	900
A-217	Brighton	700
A-222	Redwood	750
A-305	Round Hill	350

TABLE: The ***branch*** relation

branch_name	***branch_city***	***Assets***
Brighton	Brooklyn	7100000
Downtown	Brooklyn	9000000
Mianus	Horseneck	400000
North Town	Rye	3700000
Perryridge	Horseneck	1700000
Pownal	Bennington	300000
Redwood	Palo Alto	2100000
Round Hill	Horseneck	8000000

TABLE: The ***customer*** relation

customer_name	***customer_street***	***customer_city***
Adams	Spring	Pittsfield
Brooks	Senator	Brooklyn
Curry	North	Rye
Glenn	Sand Hill	Woodside
Green	Walnut	Stamford
Hayes	Main	Harrison
Johnson	Alma	Palo Alto
Jones	Main	Harrison
Lindsay	Park	Pittsfield
Smith	North	Rye
Turner	Putnum	Stamford
Williams	Nassau	Princeton

TABLE: The ***depositor*** relation

customer_name	***account_number***
Hayes	A-102
Johnson	A-101
Johnson	A-201
Jones	A-217
Lindsay	A-222
Smith	A-215
Turner	A-305

TABLE: The ***loan*** relation

loan_number	***branch_name***	***Amount***
L-11	Round Hill	900
L-14	Downtown	1500
L-15	Perryridge	1500
L-16	Perryridge	1300
L-17	Downtown	1000
L-23	Redwood	2000
L-93	Mianus	500

TABLE: The ***borrower*** relation

customer_name	***loan_number***
Adams	L-16
Curry	L-93
Hayes	L-15
Jackson	L-14
Jones	L-17
Smith	L-11
Smith	L-23
Williams	L-17

IDENTIFYING ENTITY SETS FOR THE BANK DATABASE

Our specification of data requirements serves as the starting point for constructing a conceptual schema for the database. From the characteristics listed in data requirements, we begin to identify entity sets and their attributes:

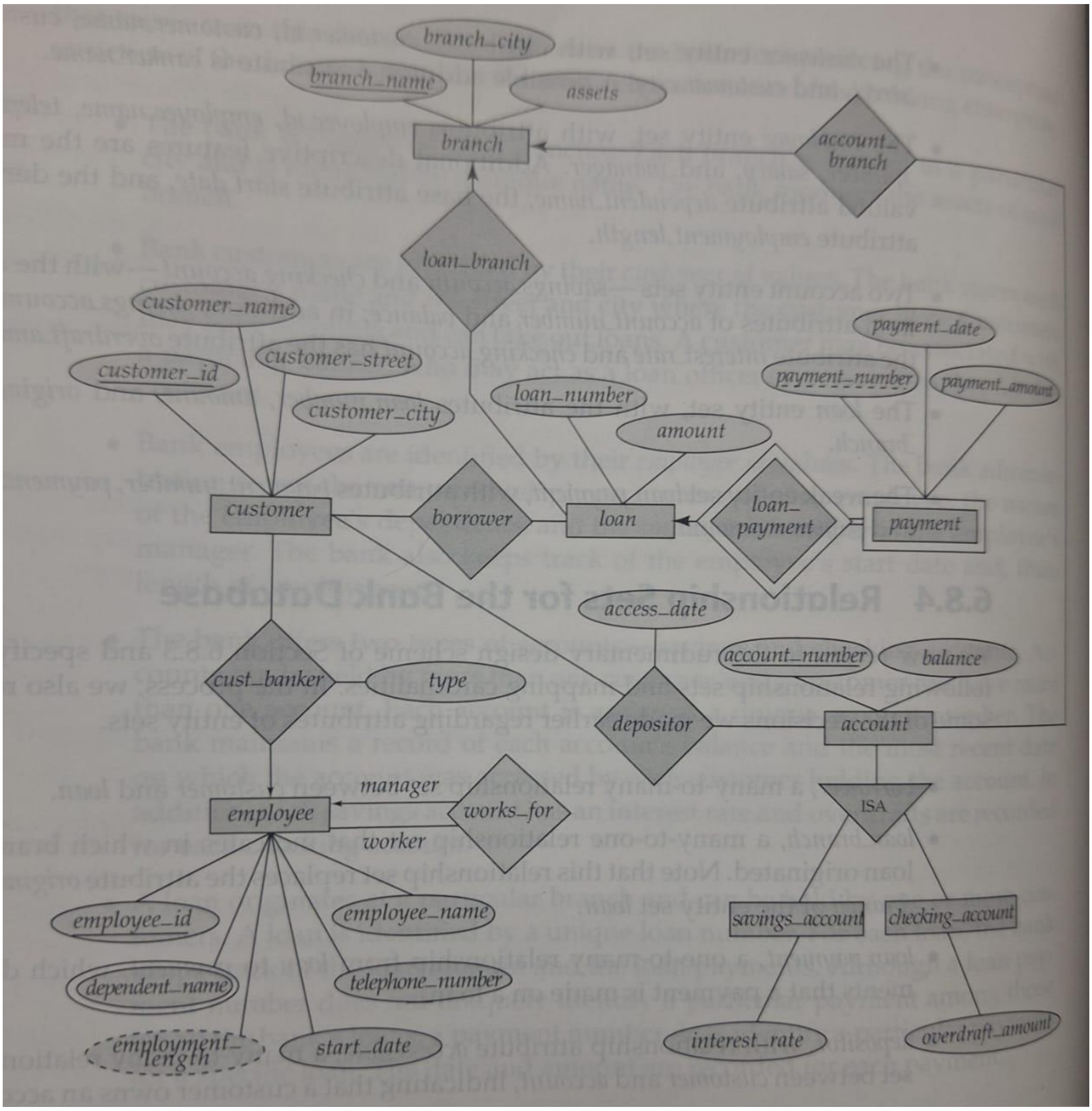
- The branch entity set, with attributes *branch name, branch_city,* and *assets*.
- The customer entity set, with attributes *customer_id, customer_name, customer_street,* and *customer_city*. A possible additional attribute is banker name.
- The employee entity set, with attributes *employee_id, employee_name, telephone_number, salary,* and *manager*. Additional descriptive features are the multi- valued attribute dependent name, the base attribute start date, and the derived attribute *employment_length*.
- Two account entity sets-*savings_account and checking_account* with the common attributes of *account_number* and *balance*, in addition, *savings_account* has the attribute interest rate and checking account has the attribute *overdraft_amount*.
- The loan entity set, with the attributes *loan_number*, amount, and originating branch.
- The weak entity set loan payment, with attributes payment, number, payment date. and *payment_amount*.

RELATIONSHIP SETS FOR THE BANK DATABASE

We now return to the rudimentary design scheme of Entity sets and specify the following relationship sets and mapping cardinalities. In the process, we also refine some of the decisions we made earlier regarding attributes of entity sets.

- *borrower*, a many-to-many relationship set between *customer* and *loan*.
- *loan_branch*, a many-to-one relationship set that indicates in which branch a Joan originated. Note that this relationship set replaces the attribute *originating_branch* of the entity set *loan*.
- *loan_payment*, a one-to-many relationship from *loan* to *payment*, which documents that a payment is made on a loan.
- depositor, with relationship attribute *access_date*, a many-to-many relationship set between *customer* and *account*, indicating that a customer owns an account.
- *cust_banker*, with relationship attribute *type*, a many-to-one relationship set ex- pressing that a customer can be advised by a bank employee, and that a bank employee can advise one or more customers. Note that this relationship set has replaced the attribute *banker_name* of the entity set *customer*.
- *works_for*, a relationship set between *employee* entities with role indicators *manager* and *worker*, the mapping cardinalities express that an employee works for only one manager and that a manager supervises one or more employees. Note that this relationship set has replaced the *manager* attribute of *employee*

E-R DIAGRAM


branch_city
branch_name
assets
branch
account_branch
loan_branch
customer_name
customer_street
customer_id
customer_city
loan_number
amount
payment_date
payment_number
payment_amount
customer
borrower
loan
loan_payment
payment
access_date
account_number
balance
cust_banker
type
depositor
account
manager
works_for
employee
worker
ISA
employee_id
employee_name
savings_account
checking_account
dependent_name
telephone_number
employment_length
start_date
interest_rate
overdraft_amount

RELATIONAL SCHEMAS FOR BANKING ENTERPRISE

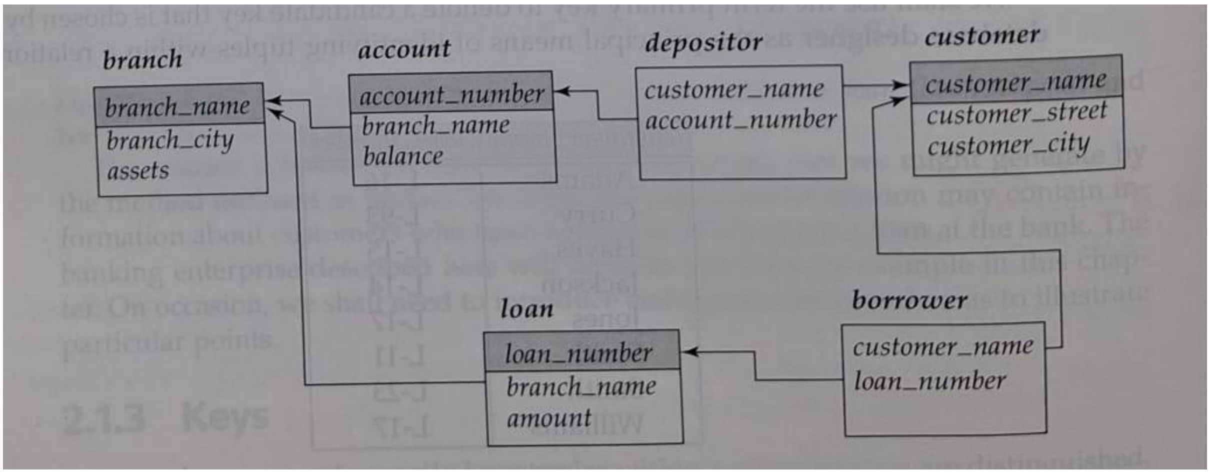
Previously, we showed an E-R diagram for a banking enterprise. The corresponding set of relation schemas, generated using the techniques described earlier in this section, is shown below. We denote the primary key for each relation schema by an underscore

- Schemas derived from a strong entity:
    - *branch = (branch_name, branch city, assets)*
    - *customer = (customer_id, customer name, customer street, customer_city)*
    - *loan = (loan_number, amount)*
    - *account = (account_number, balance)*
    - *employee = (employee_id, employee_name, telephone_number, start_date)*

- Schemas derived from a multivalued attribute: (We do not represent derived attributes.) They are defined in a view or specially defined function.
    - *dependent name = (employee_id, d_name)*

- Schemas derived from a relationship set involving strong entity sets:
    - *account_branch = (account_number, branch_name)*
    - *loan_branch = (loan_number, branch_name)*
    - *borrower = (customer_id, loan_number)*
    - *depositor = (customer_id, account_number)*
    - *cust_banker = (customer_id, employee_id, type)*
    - *works_for = (worker_employee_id, manager_employee_id)*
- Schemas derived from a weak entity:
    - *payment = (loan_number, payment_number, payment_date, payment_amount)*
- Schemas derived from an ISA relationship:
    - *savings_account = (account_number, interest_rate)*
    - *checking_account = (account_number, overdraft_amount)*

RELATIONSHIP SCHEMA DIAGRAM



SQL

```
CREATE TABLE CUSTOMER(
  CUSTOMER_NAME VARCHAR2(20) CONSTRAINT CCN PRIMARY KEY,
  CUSTOMER_STREET VARCHAR2(20) NOT NULL,
  CUSTOMER_CITY VARCHAR2(20) NOT NULL);

CREATE TABLE BRANCH(
  BRANCH_NAME VARCHAR2(20) CONSTRAINT CBN PRIMARY KEY,
  BRANCH_CITY VARCHAR2(20) NOT NULL,
  ASSETS NUMBER(*,2));

INSERT INTO CUSTOMER (CUSTOMER_NAME, CUSTOMER_STREET, CUSTOMER_CITY)
  SELECT 'Adams', 'Spring', 'Pittsfield' FROM DUAL UNION ALL
  SELECT 'Brooks', 'Senator', 'Brooklyn' FROM DUAL UNION ALL
  SELECT 'Curry', 'North', 'Rye' FROM DUAL UNION ALL
  SELECT 'Glenn', 'Sand Hill', 'Woodside' FROM DUAL UNION ALL
  SELECT 'Green', 'Walnut', 'Stamford' FROM DUAL UNION ALL
  SELECT 'Hayes', 'Main', 'Harrison' FROM DUAL UNION ALL
  SELECT 'Johnson', 'Alma', 'Palo Alto' FROM DUAL UNION ALL
  SELECT 'Jones', 'Main', 'Harrison' FROM DUAL UNION ALL
 SELECT 'Lindsay', 'Park', 'Pittsfield' FROM DUAL UNION ALL
 SELECT 'Smith', 'North', 'Rye' FROM DUAL UNION ALL
 SELECT 'Turner', 'Putnam', 'Stamford' FROM DUAL UNION ALL
 SELECT 'Williams', 'Nassau', 'Princeton' FROM DUAL;

INSERT INTO BRANCH(BRANCH_NAME,BRANCH_CITY,ASSETS)
  SELECT 'Brighton', 'Brooklyn', 7100000 FROM DUAL UNION ALL
  SELECT 'Downtown', 'Brooklyn', 9000000 FROM DUAL UNION ALL
  SELECT 'Mianus', 'Horseneck', 400000 FROM DUAL UNION ALL
  SELECT 'North Town', 'Rye', 3700000 FROM DUAL UNION ALL
  SELECT 'Perryridge', 'Horseneck', 1700000 FROM DUAL UNION ALL
  SELECT 'Pownal', 'Bennington', 300000 FROM DUAL UNION ALL
  SELECT 'Redwood', 'Palo Alto', 2100000 FROM DUAL UNION ALL
  SELECT 'Round Hill', 'Horseneck', 8000000 FROM DUAL;

CREATE TABLE ACCOUNT(
  ACCOUNT_NUMBER VARCHAR2(10) CONSTRAINT CAN PRIMARY KEY,
  BRANCH_NAME VARCHAR2(20) NOT NULL,
  BALANCE NUMBER(*,2),
  CONSTRAINT AFK FOREIGN KEY(BRANCH_NAME) REFERENCES BRANCH(BRANCH_NAME) ON DELETE CASCADE);

INSERT INTO ACCOUNT(ACCOUNT_NUMBER,BRANCH_NAME,BALANCE)
  SELECT 'A-101','Downtown',500 FROM DUAL UNION ALL
  SELECT 'A-102','Perryridge',400 FROM DUAL UNION ALL
  SELECT 'A-201','Brighton',900 FROM DUAL UNION ALL
  SELECT 'A-215','Mianus',700 FROM DUAL UNION ALL
  SELECT 'A-217','Brighton',750 FROM DUAL UNION ALL
  SELECT 'A-222','Redwood',700 FROM DUAL UNION ALL
  SELECT 'A-305','Round Hill',350 FROM DUAL;

CREATE TABLE DEPOSITOR(
  CUSTOMER_NAME VARCHAR2(20),
  ACCOUNT_NUMBER VARCHAR2(10),
  CONSTRAINT CDN PRIMARY KEY(CUSTOMER_NAME,ACCOUNT_NUMBER),
  CONSTRAINT DFK FOREIGN KEY(CUSTOMER_NAME) REFERENCES CUSTOMER(CUSTOMER_NAME) ON DELETE CASCADE,
  CONSTRAINT DFK2 FOREIGN KEY(ACCOUNT_NUMBER) REFERENCES ACCOUNT(ACCOUNT_NUMBER) ON DELETE CASCADE);

INSERT INTO DEPOSITOR(CUSTOMER_NAME,ACCOUNT_NUMBER)
  SELECT 'Hayes','A-102' FROM DUAL UNION ALL
  SELECT 'Johnson','A-101' FROM DUAL UNION ALL
  SELECT 'Johnson','A-201' FROM DUAL UNION ALL
  SELECT 'Jones','A-217' FROM DUAL UNION ALL
  SELECT 'Lindsay','A-222' FROM DUAL UNION ALL
  SELECT 'Smith','A-215' FROM DUAL UNION ALL
  SELECT 'Turner','A-305' FROM DUAL;


CREATE TABLE LOAN(
  LOAN_NUMBER VARCHAR2(10) CONSTRAINT CLN PRIMARY KEY,
  BRANCH_NAME VARCHAR2(20) NOT NULL,
  AMOUNT NUMBER(*,2) NOT NULL,
  CONSTRAINT LFK FOREIGN KEY(BRANCH_NAME) REFERENCES BRANCH(BRANCH_NAME) ON DELETE CASCADE);


INSERT INTO LOAN(LOAN_NUMBER,BRANCH_NAME,AMOUNT)
  SELECT 'L-11','Round Hill',900 FROM DUAL UNION ALL
  SELECT 'L-14','Downtown',1500 FROM DUAL UNION ALL
  SELECT 'L-15','Perryridge',1500 FROM DUAL UNION ALL
  SELECT 'L-16','Perryridge',1300 FROM DUAL UNION ALL
  SELECT 'L-17','Downtown',1000 FROM DUAL UNION ALL
  SELECT 'L-23','Redwood',2000 FROM DUAL UNION ALL
  SELECT 'L-93','Mianus',500 FROM DUAL;

CREATE TABLE BORROWER(
  LOAN_NUMBER VARCHAR2(10),
  CUSTOMER_NAME VARCHAR2(20),
  CONSTRAINT CBRN PRIMARY KEY(LOAN_NUMBER,CUSTOMER_NAME),
  CONSTRAINT BFK2 FOREIGN KEY(CUSTOMER_NAME) REFERENCES CUSTOMER(CUSTOMER_NAME) ON DELETE CASCADE,
  CONSTRAINT BFK3 FOREIGN KEY(LOAN_NUMBER) REFERENCES LOAN(LOAN_NUMBER) ON DELETE
CASCADE);

INSERT INTO BORROWER (CUSTOMER_NAME, LOAN_NUMBER)
  SELECT 'Adams', 'L-16' FROM DUAL UNION ALL
  SELECT 'Curry', 'L-93' FROM DUAL UNION ALL
  SELECT 'Hayes', 'L-15' FROM DUAL UNION ALL
  SELECT 'Johnson', 'L-14' FROM DUAL UNION ALL
  SELECT 'Jones', 'L-17' FROM DUAL UNION ALL
  SELECT 'Smith', 'L-11' FROM DUAL UNION ALL
  SELECT 'Smith', 'L-23' FROM DUAL UNION ALL
  SELECT 'Williams', 'L-17' FROM DUAL;


CREATE TABLE TRANSACTION (
  TRANSACTION_ID VARCHAR2(10) CONSTRAINT CTN PRIMARY KEY,
  ACCOUNT_NUMBER VARCHAR2(10),
  DAY DATE,
  AMOUNT NUMBER(15, 2),
  TYPE VARCHAR2(10),
  FOREIGN KEY (ACCOUNT_NUMBER) REFERENCES ACCOUNT(ACCOUNT_NUMBER));


INSERT INTO Transaction (Transaction_ID, Account_Number, Day, Amount, Type)
  SELECT 'C1', 'A-101', '2025-01-01', 1000.00, 'Credit' FROM DUAL UNION ALL
  SELECT 'D1', 'A-102', '2025-01-02', 500.00, 'Debit' FROM DUAL UNION ALL
  SELECT 'C2', 'A-201', '2025-01-03', 700.00, 'Credit' FROM DUAL;


SQL> SELECT * FROM CUSTOMER;
```

```
CUSTOMER_NAME        CUSTOMER_STREET      CUSTOMER_CITY
-------------------- -------------------- --------------------
Adams                Spring               Pittsfield
Brooks               Senator              Brooklyn
Curry                North                Rye
Glenn                Sand Hill            Woodside
Green                Walnut               Stamford
Hayes                Main                 Harrison
Johnson              Alma                 Palo Alto
Jones                Main                 Harrison
Lindsay              Park                 Pittsfield
Smith                North                Rye
Turner               Putnam               Stamford

CUSTOMER_NAME        CUSTOMER_STREET      CUSTOMER_CITY
-------------------- -------------------- --------------------
Williams             Nassau               Princeton

12 rows selected.

SQL> SELECT * FROM BRANCH;

BRANCH_NAME          BRANCH_CITY              ASSETS
-------------------- -------------------- ----------
Brighton             Brooklyn                7100000
Downtown             Brooklyn                9000000
Mianus               Horseneck                400000
North Town           Rye                     3700000
Perryridge           Horseneck               1700000
Pownal               Bennington               300000
Redwood              Palo Alto               2100000
```

```
Round Hill          Horseneck             8000000

8 rows selected.

SQL> SELECT * FROM ACCOUNT;

ACCOUNT_NU BRANCH_NAME             BALANCE
---------- -------------------- ----------
A-101      Downtown                    500
A-102      Perryridge                  400
A-201      Brighton                    900
A-215      Mianus                      700
A-217      Brighton                    750
A-222      Redwood                     700
A-305      Round Hill                  350

7 rows selected.

SQL> SELECT * FROM DEPOSITOR;

CUSTOMER_NAME        ACCOUNT_NU
-------------------- ----------
Hayes                A-102
Johnson              A-101
Johnson              A-201
Jones                A-217
Lindsay              A-222
Smith                A-215
Turner               A-305

7 rows selected.
```

```
SQL> SELECT * FROM LOAN;

LOAN_NUMBE BRANCH_NAME              AMOUNT
---------- -------------------- ----------
L-11       Round Hill              900
L-14       Downtown                1500
L-15       Perryridge             1500
L-16       Perryridge             1300
L-17       Downtown                1000
L-23       Redwood                 2000
L-93       Mianus                 500

7 rows selected.

SQL> SELECT * FROM BORROWER;

LOAN_NUMBE CUSTOMER_NAME
---------- --------------------
L-16       Adams
L-93       Curry
L-15       Hayes
L-14       Johnson
L-17       Jones
L-11       Smith
L-23       Smith
L-17       Williams

8 rows selected.
```

```
SELECT * FROM TRANSACTION;

TRANSACTIO ACCOUNT_NU DAY            AMOUNT TYPE
---------- ---------- ---------- ---------- ----------
C1         A-101      2025-01-01       1000 Credit
D1         A-102      2025-01-02        500 Debit
C2         A-201      2025-01-03        700 Credit
```