Kevin Huynh
916307020

Banking System Database Model

March 24, 2020

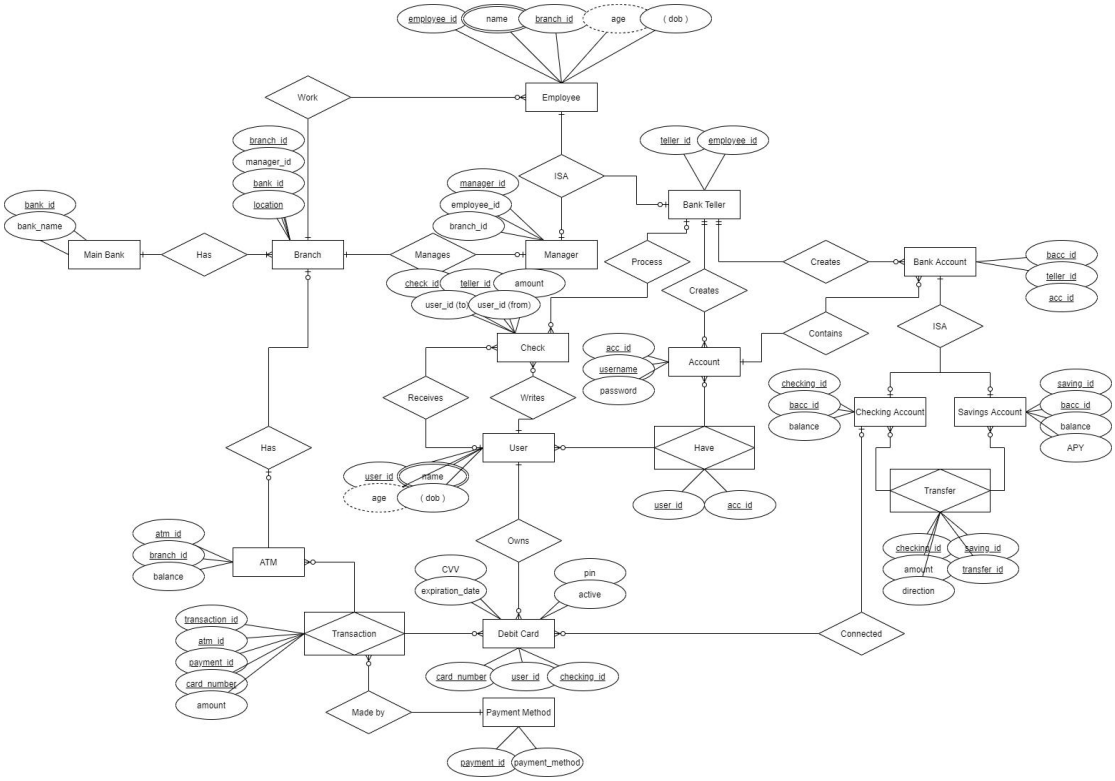
Kevin Huynh

Kevin Huynh
916307020

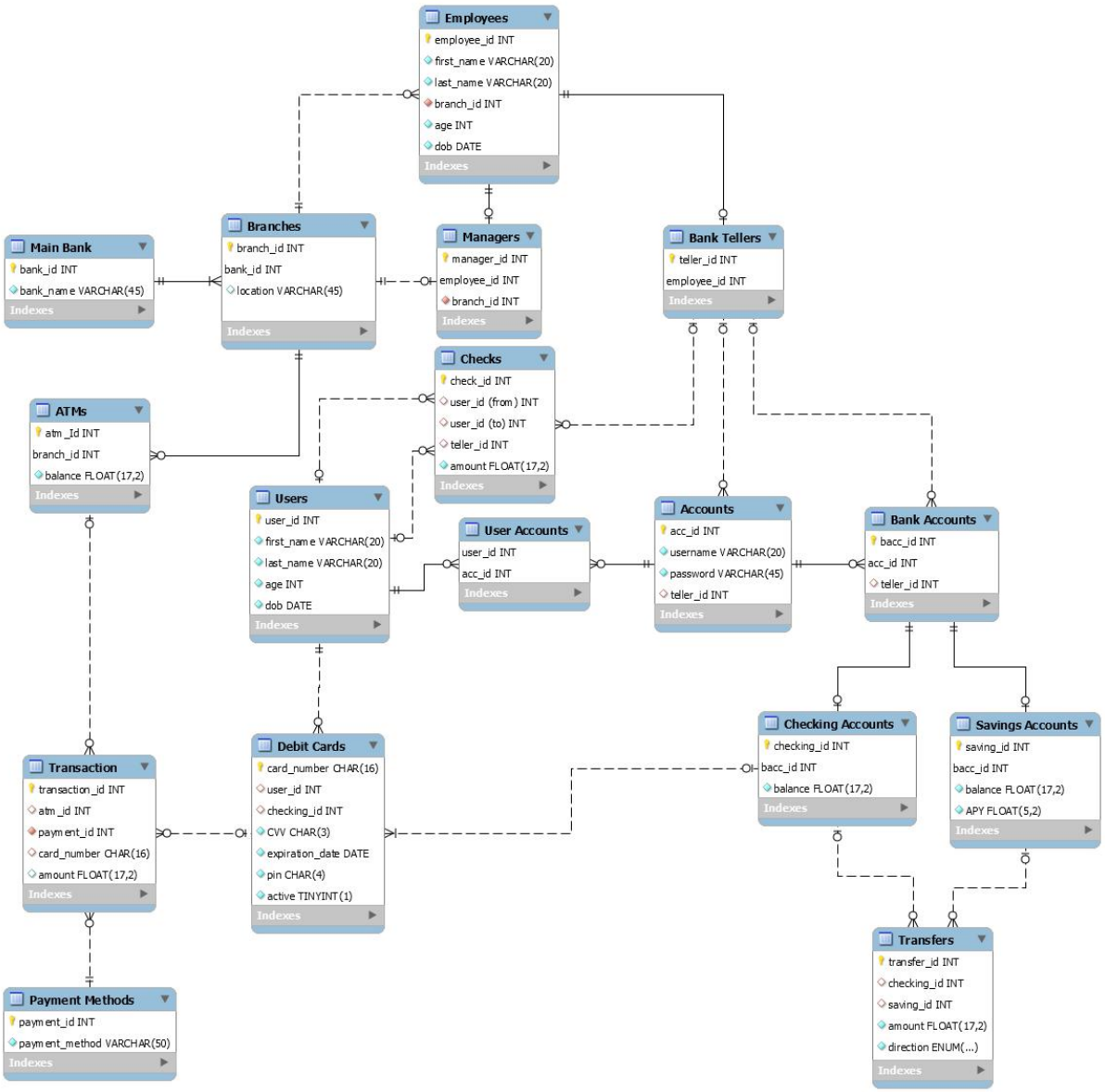
Table of Contents

Kevin Huynh
916307020

Section I: Final ERD Version



Section II: Database Model



Section III: Forward Engineering

```
-- MySQL Workbench Forward Engineering

SET @OLD_UNIQUE_CHECKS=@@UNIQUE_CHECKS, UNIQUE_CHECKS=0;
SET @OLD_FOREIGN_KEY_CHECKS=@@FOREIGN_KEY_CHECKS,
FOREIGN_KEY_CHECKS=0;
SET @OLD_SQL_MODE=@@SQL_MODE,
SQL_MODE='ONLY_FULL_GROUP_BY,STRICT_TRANS_TABLES,NO_ZERO_IN_DA
TE,NO_ZERO_DATE,ERROR_FOR_DIVISION_BY_ZERO,NO_ENGINE_SUBSTITUT
ION';

-- -----------------------------------------------------
-- Schema banking_system
-- -----------------------------------------------------

-- -----------------------------------------------------
-- Schema banking_system
-- -----------------------------------------------------
CREATE SCHEMA IF NOT EXISTS `banking_system` DEFAULT CHARACTER SET
utf8 ;
USE `banking_system` ;

-- -----------------------------------------------------
-- Table `banking_system`.`Main Bank`
-- -----------------------------------------------------
CREATE TABLE IF NOT EXISTS `banking_system`.`Main Bank` (
  `bank_id` INT UNSIGNED NOT NULL AUTO_INCREMENT,
  `bank_name` VARCHAR(45) NOT NULL,
  PRIMARY KEY (`bank_id`))
ENGINE = InnoDB;


-- -----------------------------------------------------
-- Table `banking_system`.`Branches`
-- -----------------------------------------------------
CREATE TABLE IF NOT EXISTS `banking_system`.`Branches` (
  `branch_id` INT UNSIGNED NOT NULL AUTO_INCREMENT,
  `bank_id` INT UNSIGNED NOT NULL,
  `location` VARCHAR(45) NULL,
  PRIMARY KEY (`branch_id`, `bank_id`),
  UNIQUE INDEX `location_UNIQUE` (`location` ASC) VISIBLE,
  INDEX `bank_id_idx` (`bank_id` ASC) VISIBLE,
  CONSTRAINT `BANK_BRANCH_FK`
    FOREIGN KEY (`bank_id`)
    REFERENCES `banking_system`.`Main Bank` (`bank_id`)
    ON DELETE CASCADE
    ON UPDATE CASCADE)
ENGINE = InnoDB;
```

```
-- -----------------------------------------------------
-- Table `banking_system`.`Employees`
-- -----------------------------------------------------
CREATE TABLE IF NOT EXISTS `banking_system`.`Employees` (
  `employee_id` INT UNSIGNED NOT NULL AUTO_INCREMENT,
  `first_name` VARCHAR(20) NOT NULL,
  `last_name` VARCHAR(20) NOT NULL,
  `branch_id` INT UNSIGNED NOT NULL,
  `age` INT NOT NULL,
  `dob` DATE NOT NULL,
  PRIMARY KEY (`employee_id`),
  CONSTRAINT `EMPLOYEE_BRANCH_FK`
    FOREIGN KEY (`branch_id`)
    REFERENCES `banking_system`.`Branches` (`branch_id`)
    ON DELETE CASCADE
    ON UPDATE CASCADE)
ENGINE = InnoDB;


-- -----------------------------------------------------
-- Table `banking_system`.`Managers`
-- -----------------------------------------------------
CREATE TABLE IF NOT EXISTS `banking_system`.`Managers` (
  `manager_id` INT UNSIGNED NOT NULL AUTO_INCREMENT,
  `employee_id` INT UNSIGNED NOT NULL,
  `branch_id` INT UNSIGNED NOT NULL,
  PRIMARY KEY (`manager_id`, `employee_id`),
  INDEX `branch_id_idx` (`branch_id` ASC) VISIBLE,
  INDEX `manager_id_idx` (`employee_id` ASC) VISIBLE,
  UNIQUE INDEX `employee_id_UNIQUE` (`employee_id` ASC) VISIBLE,
  UNIQUE INDEX `branch_id_UNIQUE` (`branch_id` ASC) VISIBLE,
  CONSTRAINT `MANAGER_BRANCH_FK`
    FOREIGN KEY (`branch_id`)
    REFERENCES `banking_system`.`Branches` (`branch_id`)
    ON DELETE CASCADE
    ON UPDATE CASCADE,
  CONSTRAINT `EMPLOYEE_MANAGER_FK`
    FOREIGN KEY (`employee_id`)
    REFERENCES `banking_system`.`Employees` (`employee_id`)
    ON DELETE CASCADE
    ON UPDATE CASCADE)
ENGINE = InnoDB;


-- -----------------------------------------------------
-- Table `banking_system`.`Bank Tellers`
-- -----------------------------------------------------
```

```
CREATE TABLE IF NOT EXISTS `banking_system`.`Bank Tellers` (
  `teller_id` INT UNSIGNED NOT NULL AUTO_INCREMENT,
  `employee_id` INT UNSIGNED NOT NULL,
  PRIMARY KEY (`teller_id`, `employee_id`),
  UNIQUE INDEX `employee_id_UNIQUE` (`employee_id` ASC) VISIBLE,
  CONSTRAINT `EMPLOYEE_TELLER_FK`
    FOREIGN KEY (`employee_id`)
    REFERENCES `banking_system`.`Employees` (`employee_id`)
    ON DELETE CASCADE
    ON UPDATE CASCADE)
ENGINE = InnoDB;


-- -----------------------------------------------------
-- Table `banking_system`.`Accounts`
-- -----------------------------------------------------
CREATE TABLE IF NOT EXISTS `banking_system`.`Accounts` (
  `acc_id` INT UNSIGNED NOT NULL AUTO_INCREMENT,
  `username` VARCHAR(20) NOT NULL,
  `password` VARCHAR(45) NOT NULL,
  `teller_id` INT UNSIGNED NULL,
  PRIMARY KEY (`acc_id`),
  UNIQUE INDEX `username_UNIQUE` (`username` ASC) VISIBLE,
  CONSTRAINT `ACC_BY_TELLER_FK`
    FOREIGN KEY (`teller_id`)
    REFERENCES `banking_system`.`Bank Tellers` (`teller_id`)
    ON DELETE SET NULL
    ON UPDATE CASCADE)
ENGINE = InnoDB;


-- -----------------------------------------------------
-- Table `banking_system`.`Bank Accounts`
-- -----------------------------------------------------
CREATE TABLE IF NOT EXISTS `banking_system`.`Bank Accounts` (
  `bacc_id` INT UNSIGNED NOT NULL AUTO_INCREMENT,
  `acc_id` INT UNSIGNED NOT NULL,
  `teller_id` INT UNSIGNED NULL,
  PRIMARY KEY (`bacc_id`, `acc_id`),
  CONSTRAINT `BACC_ACC_FK`
    FOREIGN KEY (`acc_id`)
    REFERENCES `banking_system`.`Accounts` (`acc_id`)
    ON DELETE CASCADE
    ON UPDATE NO ACTION,
  CONSTRAINT `BACC_BY_TELLER_FK`
    FOREIGN KEY (`teller_id`)
    REFERENCES `banking_system`.`Bank Tellers` (`teller_id`)
    ON DELETE SET NULL
    ON UPDATE CASCADE)
```

```
ENGINE = InnoDB;


-- -----------------------------------------------------
-- Table `banking_system`.`Checking Accounts`
-- -----------------------------------------------------
CREATE TABLE IF NOT EXISTS `banking_system`.`Checking Accounts`
(
  `checking_id` INT UNSIGNED NOT NULL AUTO_INCREMENT,
  `bacc_id` INT UNSIGNED NOT NULL,
  `balance` FLOAT(17,2) NOT NULL DEFAULT 0.00,
  PRIMARY KEY (`checking_id`, `bacc_id`),
  UNIQUE INDEX `bacc_id_UNIQUE` (`bacc_id` ASC) VISIBLE,
  CONSTRAINT `CHECKING_BACC_FK`
    FOREIGN KEY (`bacc_id`)
    REFERENCES `banking_system`.`Bank Accounts` (`bacc_id`)
    ON DELETE CASCADE
    ON UPDATE CASCADE)
ENGINE = InnoDB;


-- -----------------------------------------------------
-- Table `banking_system`.`Savings Accounts`
-- -----------------------------------------------------
CREATE TABLE IF NOT EXISTS `banking_system`.`Savings Accounts` (
  `saving_id` INT UNSIGNED NOT NULL AUTO_INCREMENT,
  `bacc_id` INT UNSIGNED NOT NULL,
  `balance` FLOAT(17,2) NOT NULL DEFAULT 0.00,
  `APY` FLOAT(5,2) NOT NULL DEFAULT 0.06,
  PRIMARY KEY (`saving_id`, `bacc_id`),
  UNIQUE INDEX `bacc_id_UNIQUE` (`bacc_id` ASC) VISIBLE,
  CONSTRAINT `SAVING_BACC_FK`
    FOREIGN KEY (`bacc_id`)
    REFERENCES `banking_system`.`Bank Accounts` (`bacc_id`)
    ON DELETE CASCADE
    ON UPDATE CASCADE)
ENGINE = InnoDB;


-- -----------------------------------------------------
-- Table `banking_system`.`Transfers`
-- -----------------------------------------------------
CREATE TABLE IF NOT EXISTS `banking_system`.`Transfers` (
  `transfer_id` INT UNSIGNED NOT NULL AUTO_INCREMENT,
  `checking_id` INT UNSIGNED NULL,
  `saving_id` INT UNSIGNED NULL,
  `amount` FLOAT(17,2) NOT NULL,
  `direction` ENUM("saving-to-checking", "checking-to-saving")
NOT NULL,
```

```
  PRIMARY KEY (`transfer_id`),
  CONSTRAINT `CHECKING_TRANSFER_FK`
    FOREIGN KEY (`checking_id`)
    REFERENCES `banking_system`.`Checking Accounts`
(`checking_id`)
    ON DELETE SET NULL
    ON UPDATE CASCADE,
  CONSTRAINT `SAVING_TRANSFER_FK`
    FOREIGN KEY (`saving_id`)
    REFERENCES `banking_system`.`Savings Accounts` (`saving_id`)
    ON DELETE SET NULL
    ON UPDATE CASCADE)
ENGINE = InnoDB;


-- -----------------------------------------------------
-- Table `banking_system`.`Users`
-- -----------------------------------------------------
CREATE TABLE IF NOT EXISTS `banking_system`.`Users` (
  `user_id` INT UNSIGNED NOT NULL AUTO_INCREMENT,
  `first_name` VARCHAR(20) NOT NULL,
  `last_name` VARCHAR(20) NOT NULL,
  `age` INT NOT NULL,
  `dob` DATE NOT NULL,
  PRIMARY KEY (`user_id`))
ENGINE = InnoDB;


-- -----------------------------------------------------
-- Table `banking_system`.`User Accounts`
-- -----------------------------------------------------
CREATE TABLE IF NOT EXISTS `banking_system`.`User Accounts` (
  `user_id` INT UNSIGNED NOT NULL,
  `acc_id` INT UNSIGNED NOT NULL,
  PRIMARY KEY (`user_id`, `acc_id`),
  INDEX `acc_id_idx` (`acc_id` ASC) VISIBLE,
  CONSTRAINT `OWNER_FK`
    FOREIGN KEY (`user_id`)
    REFERENCES `banking_system`.`Users` (`user_id`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION,
  CONSTRAINT `ACC_OWNED_FK`
    FOREIGN KEY (`acc_id`)
    REFERENCES `banking_system`.`Accounts` (`acc_id`)
    ON DELETE CASCADE
    ON UPDATE NO ACTION)
ENGINE = InnoDB;
```

```
-- -----------------------------------------------------
-- Table `banking_system`.`Debit Cards`
-- -----------------------------------------------------
CREATE TABLE IF NOT EXISTS `banking_system`.`Debit Cards` (
  `card_number` CHAR(16) NOT NULL,
  `user_id` INT UNSIGNED NULL,
  `checking_id` INT UNSIGNED NULL,
  `CVV` CHAR(3) NOT NULL,
  `expiration_date` DATE NOT NULL,
  `pin` CHAR(4) NOT NULL,
  `active` TINYINT(1) NOT NULL,
  PRIMARY KEY (`card_number`),
  UNIQUE INDEX `checking_id_UNIQUE` (`checking_id` ASC) VISIBLE,
  UNIQUE INDEX `user_id_UNIQUE` (`user_id` ASC) VISIBLE,
  CONSTRAINT `CARD_OWNER_FK`
    FOREIGN KEY (`user_id`)
    REFERENCES `banking_system`.`Users` (`user_id`)
    ON DELETE SET NULL
    ON UPDATE CASCADE,
  CONSTRAINT `CHECKING_CONNECTED_FK`
    FOREIGN KEY (`checking_id`)
    REFERENCES `banking_system`.`Checking Accounts`
(`checking_id`)
    ON DELETE SET NULL
    ON UPDATE CASCADE)
ENGINE = InnoDB;


-- -----------------------------------------------------
-- Table `banking_system`.`ATMs`
-- -----------------------------------------------------
CREATE TABLE IF NOT EXISTS `banking_system`.`ATMs` (
  `atm_Id` INT UNSIGNED NOT NULL AUTO_INCREMENT,
  `branch_id` INT UNSIGNED NOT NULL,
  `balance` FLOAT(17,2) NOT NULL DEFAULT 0.00,
  PRIMARY KEY (`atm_Id`, `branch_id`),
  CONSTRAINT `BRANCH_LOCATED_FK`
    FOREIGN KEY (`branch_id`)
    REFERENCES `banking_system`.`Branches` (`branch_id`)
    ON DELETE CASCADE
    ON UPDATE CASCADE)
ENGINE = InnoDB;


-- -----------------------------------------------------
-- Table `banking_system`.`Payment Methods`
-- -----------------------------------------------------
CREATE TABLE IF NOT EXISTS `banking_system`.`Payment Methods` (
  `payment_id` INT UNSIGNED NOT NULL AUTO_INCREMENT,
```

```
  `payment_method` VARCHAR(50) NOT NULL,
  PRIMARY KEY (`payment_id`))
ENGINE = InnoDB;


-- -----------------------------------------------------
-- Table `banking_system`.`Transaction`
-- -----------------------------------------------------
CREATE TABLE IF NOT EXISTS `banking_system`.`Transaction` (
  `transaction_id` INT UNSIGNED NOT NULL AUTO_INCREMENT,
  `atm_id` INT UNSIGNED NULL,
  `payment_id` INT UNSIGNED NOT NULL,
  `card_number` CHAR(16) NULL,
  `amount` FLOAT(17,2) NULL DEFAULT 0.00,
  PRIMARY KEY (`transaction_id`),
  INDEX `atm_id_idx` (`atm_id` ASC) VISIBLE,
  INDEX `card_number_idx` (`card_number` ASC) VISIBLE,
  INDEX `payment_id_idx` (`payment_id` ASC) VISIBLE,
  UNIQUE INDEX `COMP_TRANS_UNIQUE` (`transaction_id` ASC,
`atm_id` ASC, `card_number` ASC) VISIBLE,
  CONSTRAINT `ATM_FK`
    FOREIGN KEY (`atm_id`)
    REFERENCES `banking_system`.`ATMs` (`atm_Id`)
    ON DELETE SET NULL
    ON UPDATE CASCADE,
  CONSTRAINT `DEBIT_CARD_FK`
    FOREIGN KEY (`card_number`)
    REFERENCES `banking_system`.`Debit Cards` (`card_number`)
    ON DELETE SET NULL
    ON UPDATE CASCADE,
  CONSTRAINT `PAYMENT_METHODS_FK`
    FOREIGN KEY (`payment_id`)
    REFERENCES `banking_system`.`Payment Methods` (`payment_id`)
    ON DELETE NO ACTION
    ON UPDATE CASCADE)
ENGINE = InnoDB;


-- -----------------------------------------------------
-- Table `banking_system`.`Checks`
-- -----------------------------------------------------
CREATE TABLE IF NOT EXISTS `banking_system`.`Checks` (
  `check_id` INT UNSIGNED NOT NULL AUTO_INCREMENT,
  `user_id (from)` INT UNSIGNED NULL,
  `user_id (to)` INT UNSIGNED NULL,
  `teller_id` INT UNSIGNED NULL,
  `amount` FLOAT(17,2) NOT NULL DEFAULT 0.00,
  PRIMARY KEY (`check_id`),
  INDEX `user_id (to)_idx` (`user_id (to)` ASC) VISIBLE,
```

```
  INDEX `user_id (from)_idx` (`user_id (from)` ASC) VISIBLE,
  UNIQUE INDEX `COMP_UNIQUE` (`check_id` ASC, `user_id (from)` ASC,
`user_id (to)` ASC) VISIBLE,
  CONSTRAINT `USER_RECEIVER_FK`
    FOREIGN KEY (`user_id (to)`)
    REFERENCES `banking_system`.`Users` (`user_id`)
    ON DELETE SET NULL
    ON UPDATE CASCADE,
  CONSTRAINT `USER_SENDER_FK`
    FOREIGN KEY (`user_id (from)`)
    REFERENCES `banking_system`.`Users` (`user_id`)
    ON DELETE SET NULL
    ON UPDATE CASCADE,
  CONSTRAINT `TELLER_PROCESSED_FK`
    FOREIGN KEY (`teller_id`)
    REFERENCES `banking_system`.`Bank Tellers` (`teller_id`)
    ON DELETE SET NULL
    ON UPDATE CASCADE)
ENGINE = InnoDB;


SET SQL_MODE=@OLD_SQL_MODE;
SET FOREIGN_KEY_CHECKS=@OLD_FOREIGN_KEY_CHECKS;
SET UNIQUE_CHECKS=@OLD_UNIQUE_CHECKS;
```

Section IV: Inserting Data

```
USE banking_system;

INSERT INTO `main bank`(bank_name) VALUES("XYZ Bank");
INSERT INTO `main bank`(bank_name) VALUES("ABC Bank");
INSERT INTO `main bank`(bank_name) VALUES("BIG Bank");

INSERT INTO branches(bank_id, location) VALUES(1, "XYZ Street,
CA");
INSERT INTO branches(bank_id, location) VALUES(2, "ABC Street,
CA");
INSERT INTO branches(bank_id, location) VALUES(3, "Large Street,
CA");
INSERT INTO branches(bank_id, location) VALUES (1, "2134 Nowhere
Ville");

INSERT INTO employees(first_name, last_name, branch_id, age, dob)
VALUES("John", "Smith", 1, 25, "1995-07-23");
INSERT INTO employees(first_name, last_name, branch_id, age, dob)
VALUES("Emma", "Johnson", 3, 27, "1993-05-13");
INSERT INTO employees(first_name, last_name, branch_id, age, dob)
VALUES("Robert", "Brown", 2, 33, "1987-11-06");
INSERT INTO employees(first_name, last_name, branch_id, age, dob)
VALUES("David", "Miller", 2, 20, "2020-01-02");
INSERT INTO employees(first_name, last_name, branch_id, age, dob)
VALUES("Thomas", "Jones", 2, 30, "1990-12-25");
INSERT INTO employees(first_name, last_name, branch_id, age, dob)
VALUES("Nancy", "Anderson", 3, 35, "1985-09-16");

INSERT INTO managers(employee_id, branch_id) VALUES (
   (SELECT (employee_id) FROM employees WHERE employee_id = 1),
    (SELECT (b.branch_id) FROM branches b, employees e WHERE
e.employee_id = 1 AND e.branch_id = b.branch_id));
INSERT INTO managers(employee_id, branch_id) VALUES (
   (SELECT (employee_id) FROM employees WHERE employee_id = 2),
    (SELECT (b.branch_id) FROM branches b, employees e WHERE
e.employee_id = 2 AND e.branch_id = b.branch_id));
INSERT INTO managers(employee_id, branch_id) VALUES (
   (SELECT (employee_id) FROM employees WHERE employee_id = 3),
    (SELECT (b.branch_id) FROM branches b, employees e WHERE
e.employee_id = 3 AND e.branch_id = b.branch_id));

INSERT INTO `bank tellers`(employee_id) VALUES (
   (SELECT (employee_id) FROM employees WHERE employee_id = 5 AND
employee_id NOT IN
    (SELECT employee_id FROM managers)));
INSERT INTO `bank tellers`(employee_id) VALUES (
```

```
   (SELECT (employee_id) FROM employees WHERE employee_id = 6 AND
employee_id NOT IN
    (SELECT employee_id FROM managers)));
INSERT INTO `bank tellers`(employee_id) VALUES (
   (SELECT (employee_id) FROM employees WHERE employee_id = 4 AND
employee_id NOT IN
    (SELECT employee_id FROM managers)));

INSERT INTO users(first_name, last_name, age, dob) VALUES ("Kevin",
"Huynh", 23, "1997-03-19");
INSERT INTO users(first_name, last_name, age, dob) VALUES ("Sarah",
"Davis", 18, "2003-10-15");
INSERT INTO users(first_name, last_name, age, dob) VALUES
("William", "Miller", 60, "1960-02-09");

INSERT INTO accounts(username, `password`, teller_id) VALUES
("kevin123", "something",
   (SELECT teller_id FROM `bank tellers` WHERE teller_id = 1));
INSERT INTO accounts(username, `password`, teller_id) VALUES
("sarahqwerty", "somepassword",
   (SELECT teller_id FROM `bank tellers` WHERE teller_id = 2));
INSERT INTO accounts(username, `password`, teller_id) VALUES
("willmill", "newpassword",
   (SELECT teller_id FROM `bank tellers` WHERE teller_id = 3));

INSERT INTO `user accounts`(user_id, acc_id) VALUES (1, 1);
INSERT INTO `user accounts`(user_id, acc_id) VALUES (2, 2);
INSERT INTO `user accounts`(user_id, acc_id) VALUES (3, 3);

INSERT INTO `bank accounts`(acc_id, teller_id) VALUES (
   (SELECT acc_id FROM accounts WHERE username = "kevin123"),
    (SELECT teller_id FROM accounts WHERE username = "kevin123"));
INSERT INTO `bank accounts`(acc_id, teller_id) VALUES (
   (SELECT acc_id FROM accounts WHERE username = "kevin123"), 2);
INSERT INTO `bank accounts`(acc_id, teller_id) VALUES (
   (SELECT acc_id FROM accounts WHERE username = "sarahqwerty"),
    (SELECT teller_id FROM accounts WHERE username =
"sarahqwerty"));
INSERT INTO `bank accounts`(acc_id, teller_id) VALUES (
   (SELECT acc_id FROM accounts WHERE username = "willmill"),
    (SELECT teller_id FROM accounts WHERE username = "willmill"));

INSERT INTO `checking accounts`(bacc_id, balance) VALUES (
   (SELECT bacc_id FROM `bank accounts` WHERE bacc_id = 1),
500.00);
INSERT INTO `checking accounts`(bacc_id, balance) VALUES (
   (SELECT bacc_id FROM `bank accounts` WHERE bacc_id = 3),
500.00);
INSERT INTO `checking accounts`(bacc_id, balance) VALUES (
```

```
   (SELECT bacc_id FROM `bank accounts` WHERE bacc_id = 4),
2500.00);

INSERT INTO `savings accounts`(bacc_id, balance) VALUES (
   (SELECT bacc_id FROM `bank accounts` WHERE bacc_id = 2),
1000.00);

INSERT INTO transfers(checking_id, saving_id, amount, direction)
VALUES (1, 1, 0.00, "checking-to-saving");
INSERT INTO transfers(checking_id, saving_id, amount, direction)
VALUES (2, 1, 0.00, "saving-to-checking");
INSERT INTO transfers(checking_id, saving_id, amount, direction)
VALUES (2, 1, 0.00, "checking-to-saving");

INSERT INTO atms(branch_id, balance) VALUES (1, 100000.00);
INSERT INTO atms(branch_id, balance) VALUES (1, 50000.00);
INSERT INTO atms(branch_id, balance) VALUES (3, 100000.00);
INSERT INTO atms(branch_id, balance) VALUES (2, 123000.00);


INSERT INTO `debit cards`(card_number, user_id, checking_id, CVV,
expiration_date, pin, `active`) VALUES (
   "1251351584321659", 1, 1, "513", "2025-06-00", "3899", true);
INSERT INTO `debit cards`(card_number, user_id, checking_id, CVV,
expiration_date, pin, `active`) VALUES (
   "5618651651861568", 2, 2, "221", "2022-09-00", "5252", true);

INSERT INTO `payment methods`(payment_method) VALUE ("Deposit");
INSERT    INTO    `payment    methods`(payment_method)    VALUE
("Withdrawal");
INSERT INTO `payment methods`(payment_method) VALUE ("Cash");

INSERT INTO `transaction`(atm_id, payment_id, card_number, amount)
VALUES (1, 2, "1251351584321659", 50.00);
INSERT INTO `transaction`(atm_id, payment_id, card_number, amount)
VALUES (1, 3, "1251351584321659", 20.00);
INSERT INTO `transaction`(atm_id, payment_id, card_number, amount)
VALUES (2, 3, "5618651651861568", 40.00);

INSERT INTO checks(`user_id (from)`, `user_id (to)`, teller_id,
amount) VALUES (1, 2, 3, 122.53);
INSERT INTO checks(`user_id (from)`, `user_id (to)`, teller_id,
amount) VALUES (1, 3, 1, 150.50);
INSERT INTO checks(`user_id (from)`, `user_id (to)`, teller_id,
amount) VALUES (3, 2, 2, 5.53);
```

Section V: Testing

1. Main Bank can have multiple branches.
   a) UPDATE `main bank` SET bank_name="LARGE Bank" WHERE bank_name = "BIG Bank";
   b) DELETE FROM `main bank` WHERE bank_id = 3;
   c) SELECT m.bank_name, b.location FROM `main bank` m, branches b WHERE m.bank_id = b.bank_id;
2. Many employees work at one branch.
   a) UPDATE employees SET last_name = "Wall" WHERE employee_id = 3;
   b) DELETE FROM employees WHERE employee_id = 5;
   c) SELECT e.first_name, e.last_name, b.location FROM employees e, branches b WHERE e.branch_id = b.branch_id;
3. One manager manages one branch.
   a) UPDATE managers SET branch_id = 4 WHERE manager_id = 3;
   b) DELETE FROM managers WHERE branch_id = 4;
   c) SELECT e.first_name, e.last_name, b.branch_id
      FROM employees e, managers m, branches b
      WHERE b.branch_id = m.branch_id AND m.employee_id = e.employee_id;
4. Many ATMs are located at one branch.
   a) DELETE FROM atms WHERE atm_id = 4;
   b) UPDATE atms SET balance = 1000000 WHERE atm_id = 2;
   c) SELECT a.atm_id, b.location FROM atms a LEFT JOIN branches b ON a.branch_id = b.branch_id;
5. One account can contain many bank accounts.
   a) UPDATE `bank accounts` SET acc_id = 2 WHERE bacc_id = 2;
   b) DELETE FROM `bank accounts` WHERE bacc_id = 1;
   c) SELECT * FROM `bank accounts` ba LEFT JOIN accounts a ON ba.acc_id = a.acc_id;
6. One bank teller can create many accounts.
   a) UPDATE accounts SET `password` = "newerpassword" WHERE `password` = "newpassword";
   b) DELETE FROM `accounts` WHERE acc_id = 1;
   c) SELECT t.teller_id, a.acc_id
      FROM `bank tellers` t LEFT JOIN accounts a
      ON t.teller_id = a.teller_id;
7. One bank teller can create many bank accounts.
   a) UPDATE `bank accounts` SET teller_id = 3 WHERE bacc_id = 2;
   b) DELETE FROM `bank accounts` WHERE bacc_id = 3;
   c) SELECT t.teller_id, ba.bacc_id
      FROM `bank tellers` t LEFT JOIN `bank accounts` ba
      ON t.teller_id = ba.teller_id;
8. One account can be owned by many users.
   a) UPDATE `user accounts` SET acc_id = 3 WHERE user_id = 2;
   b) DELETE FROM users WHERE user_id = 1;
   c) SELECT * from `user accounts`;
9. One bank teller can process many checks.

   a) UPDATE checks SET teller_id = 3 WHERE check_id = 3;
   b) DELETE FROM checks WHERE check_id = 2;
   c) SELECT t.teller_id, c.check_id FROM `bank tellers` t, checks c WHERE t.teller_id = 3 AND t.teller_id = c.teller_id;

10. One user can own many debit cards.
   a) UPDATE `checking accounts` SET balance = 15000.00 WHERE checking_id = 3;
   b) DELETE FROM `checking accounts` WHERE checking_id = 3;
   c) SELECT * FROM `checking accounts`;
11. Many debit cards can be connected to one checking account.
   a) UPDATE `debit cards` SET checking_id = 4 WHERE card_number = "5618651651861568";
   b) DELETE FROM `debit cards` WHERE card_number = "1251351584321659";
   c) SELECT * FROM `debit cards`;
12. Many debit cards can perform transactions on many ATMs.
   a) UPDATE `transaction` SET card_number = "5618651651861568" WHERE transaction_id = 2;
   b) DELETE FROM `transaction` WHERE transaction_id = 1;
   c) SELECT * FROM `transaction`;
13. Many checking accounts can transfer money to many savings accounts, if both accounts are owned by the same user.
   a) UPDATE transfers SET amount = 15.00 WHERE transfer_id = 1;
   b) DELETE FROM transfers WHERE transfer_id = 2;
   c) SELECT * FROM transfers;
14. Many savings accounts can transfer money to many checking accounts, if both accounts are owned by the same user.
   a) See 14.

Section VI: Testing Table

Test	Statement	Entity	Pass/Fail	Error Description
1	Update	Main bank	Pass	N/A
2	Delete	Main bank	Fail	Cannot delete or update a parent row
3	Select	Main bank	Pass	N/A
4	Update	Employees	Pass	N/A
5	Delete	Employees	Pass	N/A
6	Select	Employees	Pass	N/A
7	Update	Manager	Pass	N/A
8	Delete	Manager	Pass	N/A
9	Select	Manager	Pass	N/A
10	Update	ATMs	Pass	N/A
11	Delete	ATMs	Pass	N/A
12	Select	ATMs	Pass	N/A
13	Update	Bank Accounts	Pass	N/A
14	Delete	Bank Accounts	Fail	Cannot delete or update a parent row
15	Select	Bank Accounts	Pass	N/A
16	Update	Accounts	Pass	N/A
17	Delete	Bank tellers	Pass	N/A
18	Select	Bank tellers	Pass	N/A
19	Update	User Accounts	Pass	N/A
20	Delete	Users	Fail	Cannot delete or update a parent row
21	Select	User Accounts	Pass	N/A
22	Update	Checks	Pass	N/A
23	Delete	Checks	Pass	N/A
24	Select	Checks	Pass	N/A
25	Update	Checking Accounts	Pass	N/A
26	Delete	Checking Accounts	Pass	N/A
27	Select	Checking Accounts	Pass	N/A
28	Update	Debit cards	Pass	N/A
29	Delete	Debit cards	Fail	Cannot delete or update a parent row
30	Select	Debit cards	Pass	N/A
31	Update	Transaction	Pass	N/A
32	Delete	Transaction	Pass	N/A
33	Select	Transaction	Pass	N/A
34	Update	Transfers	Pass	N/A
35	Delete	Transfers	Pass	N/A
36	Select	Transfers	Pass	N/A