به نام خدا

گزارش‌کار فاز اول پروژه

درس اصول و طراحی پایگاه داده

استاد درس: دکتر پوربهمن

نیم‌سال اول ۰۴ - ۰۳

سیدعلی محمد داستان

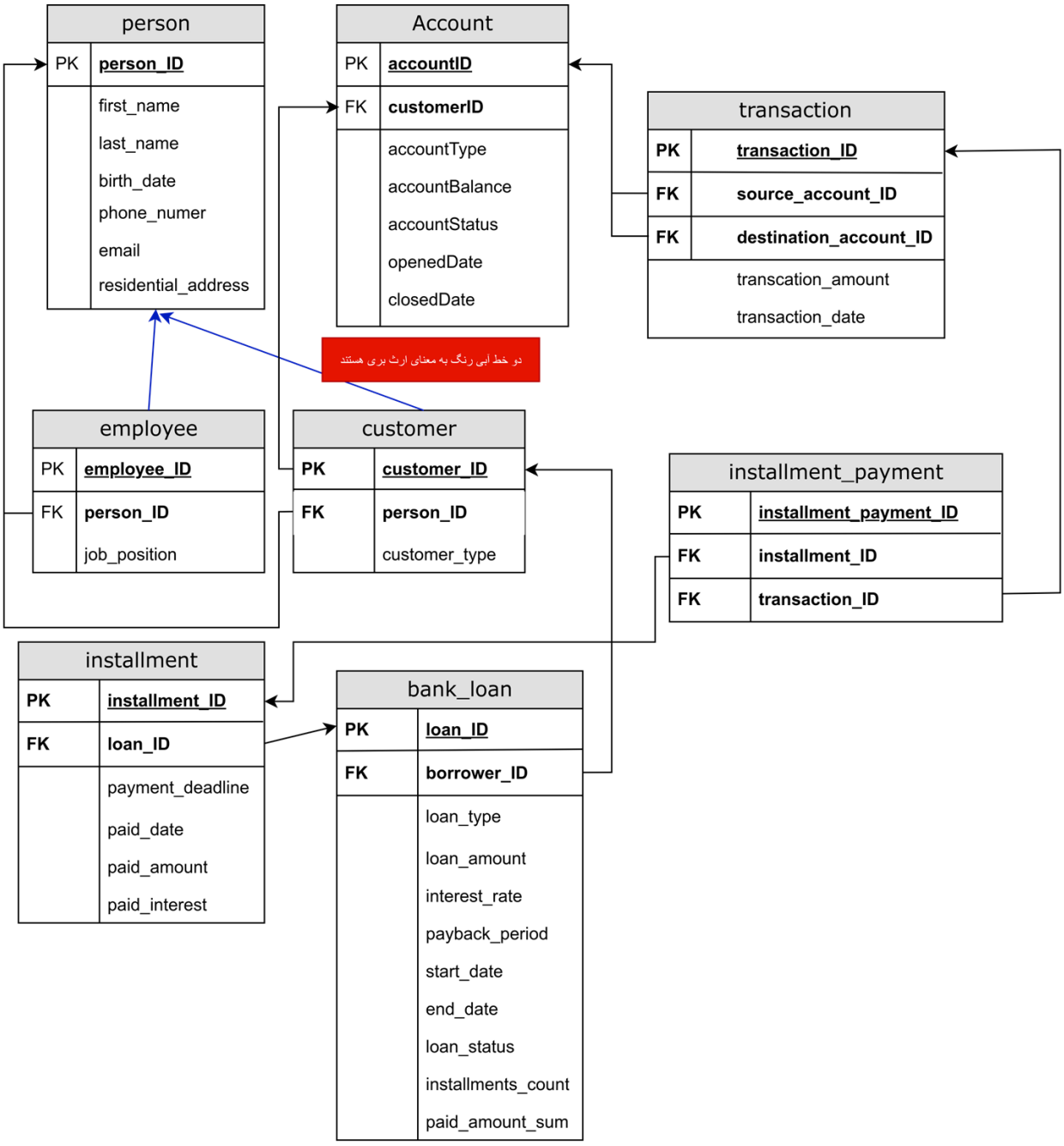
شماره دانشجویی: ۴۰۱۳۱۰۱۳

دی ۱۴۰۳

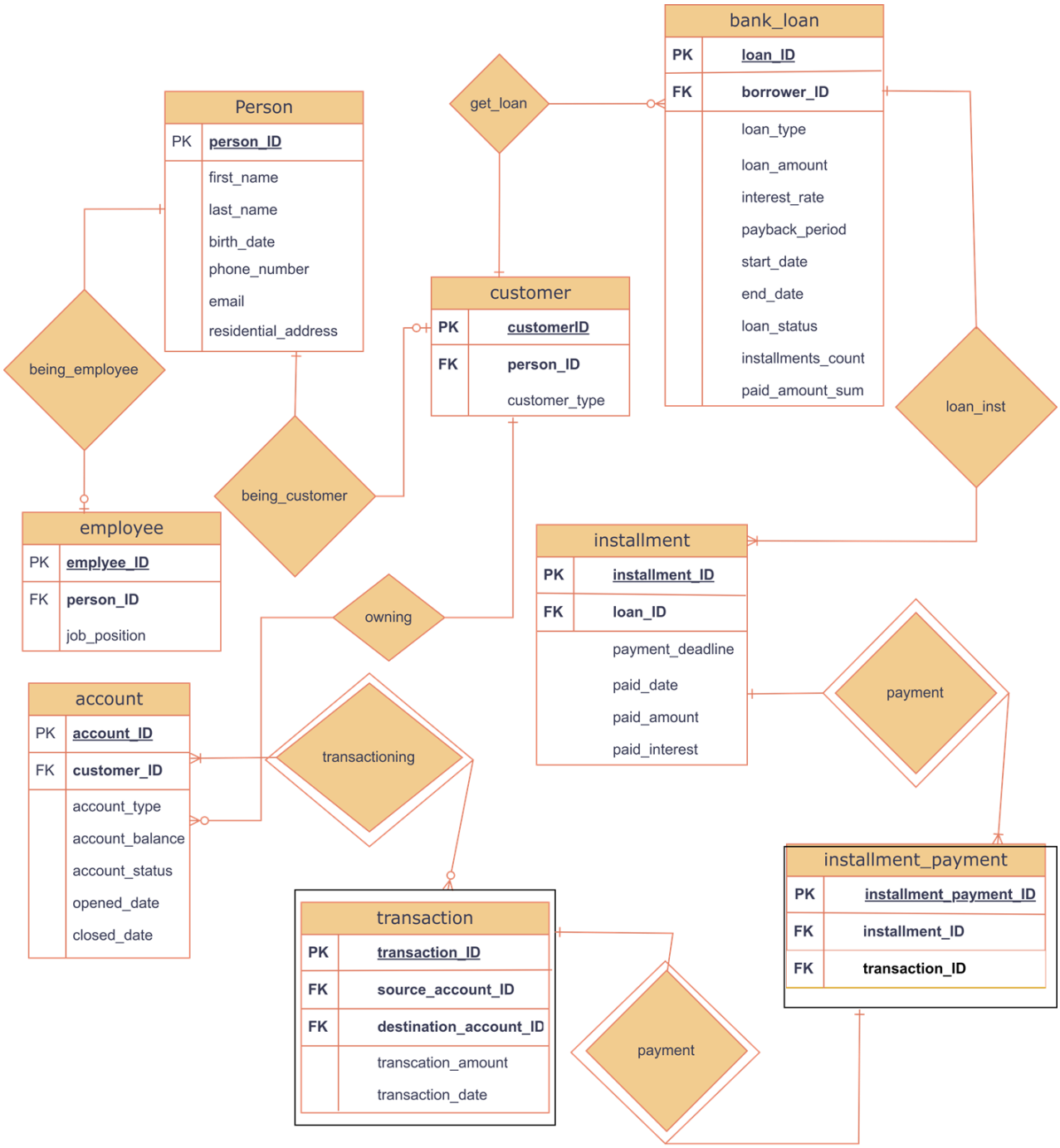
اسکیما

با توجه به توضیحات مربوط به موجودیت‌های مختلف، اسکیما بدین شکل طراحی شد:

(فایل png. و drawio. ضمیمه شده‌اند)



مدل E-R:


bank_loan
PK loan_ID
FK borrower_ID
loan_type
loan_amount
interest_rate
payback_period
start_date
end_date
loan_status
installments_count
paid_amount_sum
get_loan
Person
PK person_ID
first_name
last_name
birth_date
phone_number
email
residential_address
being_employee
customer
PK customerID
FK person_ID
customer_type
being_customer
loan_inst
employee
PK emplyee_ID
FK person_ID
job_position
owning
installment
PK installment_ID
FK loan_ID
payment_deadline
paid_date
paid_amount
paid_interest
payment
account
PK account_ID
FK customer_ID
account_type
account_balance
account_status
opened_date
closed_date
transactioning
installment_payment
PK installment_payment_ID
FK installment_ID
FK transaction_ID
transaction
PK transaction_ID
FK source_account_ID
FK destination_account_ID
transcation_amount
transaction_date
payment

توضیحات نرمال‌سازی و فرم BCNF:

person:

با توجه به این که person_ID کلید اصلی می‌باشد و شماره تلفن و ایمیل یکتا هستند و می‌توانند بقیه رفتارهای جدول را شناسایی کنند؛ پس این ۳ رفتار می‌توانند کلیدهای کاندید باشند.
همچنین وابستگی تابعی‌های زیر را داریم:

person_ID → first_name, last_name, birth_date, phone_number, email, residential_address

phone_number → person_ID, first_name, last_name, birth_date, email, residential_address

email → person_ID, first_name, last_name, birth_date, email, residential_address

سمت چپ هر ۳ وابستگی تابعی بالا، یک کلید کاندید و در نتیجه بخشی از یک سوپرکلید است و در نتیجه هیچکدام BCNF را نقض نمی‌کنند.

در نتیجه این جدول به فرم BCNF قرار دارد و نیازی به نرمال‌سازی دیگری نیست.

customer:

در این جدول، customer_ID کلید اصلی و در نتیجه در هر ردیف یکتا است و از آنجایی که رابطه جدول‌های person و customer یک به یک است، در نتیجه person_ID هرکدام از مشتری‌ها نیز یکتا است.

وابستگی‌های تابعی:

customer_ID → person_ID, customer_type

person_ID → person_ID, customer_type

سمت چپ وابستگی تابعی اول، یک کلید اصلی و در نتیجه یک سوپر کلید است و در نتیجه این وابستگی BCNF را نقض نمی‌کند.
person_ID در وابستگی دوم نیز یک سوپر کلید و در نتیجه یک کلید کاندید است و BCNF را رعایت می‌کند.

در نتیجه موجودیت customer هم به فرم BCNF است و نیازی به نرمال‌سازی دیگری ندارد.

بدین صورت به صورت خلاصه برای بقیه جداول هم داریم:

employee:

کلید اصلی:

employee_ID

کلیدهای کاندید:

employee_ID

person_ID

وابستگی‌های تابعی:

employee_ID → person_ID, job_position

person_ID → employee_ID, job_position

account:

کلید اصلی:

account_ID

کلیدهای کاندید:

account_ID

customer_ID, account_type

وابستگی‌های تابعی:

account_ID → customer_ID, account_type, account_balance, account_status, opened_date, closed_date

customer_ID, account_type → account_ID, account_balance, account_status, opened_date, closed_date

:bank_loan

کلید اصلی:

loan_ID

کلیدهای کاندید:

loan_ID

borrower_ID, loan_type, start_date

وابستگی‌های تابعی:

loan_ID → borrower_ID, loan_type, loan_amount, interest_rate, payback_period, start_date, end_date, loan_status, installments_count, paid_amount_sum

borrower_ID, loan_type, start_date → loan_ID, loan_amount, interest_rate, payback_period, end_date, loan_status, installments_count, paid_amount_sum

:installment

کلید اصلی:

installment_ID

کلیدهای کاندید:

installment_ID

loan_ID, payment_deadline

وابستگی‌های تابعی:

installment_ID → loan_ID, payment_deadline, paid_date, paid_amount, paid_interest

loan_ID, payment_deadline → installment_ID, paid_date, paid_amount, paid_interest

installment_payment:

کلید اصلی:

installment_payment_ID

کلیدهای کاندید:

installment_payment_ID

installment_ID, transaction_ID

وابستگی‌های تابعی:

installment_payment_ID → installment_ID, transaction_ID

installment_ID, transaction_ID → installment_payment_ID

پس در نتیجه این جدول هم مانند تمام جداولی که بالاتر بررسی شدند، به فرم BCNF است و نیازی به نرمال‌سازی ندارد.

کدهای SQL برای ساخت موجودیت‌ها:

```sql
CREATE TABLE person (
    person_ID INT AUTO_INCREMENT PRIMARY KEY,
    first_name VARCHAR(15),
    last_name VARCHAR(15),
    birth_date DATE,
    phone_number VARCHAR(15) UNIQUE,
    email VARCHAR(63) UNIQUE,
    residential_address VARCHAR(255)
);
CREATE TABLE customer (
    customer_ID INT AUTO_INCREMENT PRIMARY KEY,
    person_ID INT NOT NULL,
    customer_type ENUM(
        'individual',
        'corporate'
    ) NOT NULL,
    FOREIGN KEY (person_ID) REFERENCES person (person_ID) ON DELETE CAS-
CADE
);
CREATE TABLE employee (
    employee_ID INT PRIMARY KEY,
    person_ID INT NOT NULL,
    job_position VARCHAR(15) NOT NULL,
    FOREIGN KEY (person_ID) REFERENCES person (person_ID) ON DELETE CASCADE
);
```

```sql
CREATE TABLE account (
    account_ID INT AUTO_INCREMENT PRIMARY KEY,
    customer_ID INT NOT NULL,
    account_type VARCHAR(15) NOT NULL,
    account_balance DECIMAL(15, 2) DEFAULT 0,
    account_status ENUM(
        'active',
        'suspended',
        'closed'
    ) NOT NULL DEFAULT 'active',
    opened_date DATE,
    closed_date DATE,
    FOREIGN KEY (customer_ID) REFERENCES customer (customer_ID) ON DELETE
CASCADE
);
CREATE TABLE bank_loan (
    loan_ID INT PRIMARY KEY,
    borrower_ID INT NOT NULL,
    loan_type VARCHAR(15) NOT NULL,
    loan_amount DECIMAL(15, 2) NOT NULL,
    interest_rate DECIMAL(5, 2) NOT NULL,
    payback_period INT,
    start_date DATE NOT NULL,
    end_date DATE,
    loan_status ENUM(
        'pending',
        'approved',
        'rejected',
        'completed'
    ) DEFAULT 'pending',
```

```
    installments_count INT,
    paid_amount_sum DECIMAL(15, 2) DEFAULT 0,
    FOREIGN KEY (borrower_ID) REFERENCES customer (customer_ID) ON DELETE
CASCADE
);
CREATE TABLE installment (
    installment_ID INT PRIMARY KEY,
    loan_ID INT NOT NULL,
    payment_deadline DATE NOT NULL,
    paid_date DATE,
    paid_amount DECIMAL(15, 2) DEFAULT 0,
    paid_interest DECIMAL(15, 2) DEFAULT 0,
    FOREIGN KEY (loan_ID) REFERENCES bank_loan (loan_ID) ON DELETE CASCADE
);
CREATE TABLE installment_payment (
    installment_payment_ID INT PRIMARY KEY,
    installment_ID INT NOT NULL,
    transaction_ID INT NOT NULL,
    FOREIGN KEY (installment_ID) REFERENCES installment (installment_ID),
    FOREIGN KEY (transaction_ID) REFERENCES transaction (transaction_ID),
);
```