Database System

Package Delivery System

Project 1. E-R design & relational schema design

담당 교수 : 정성원

이름 : 김도현

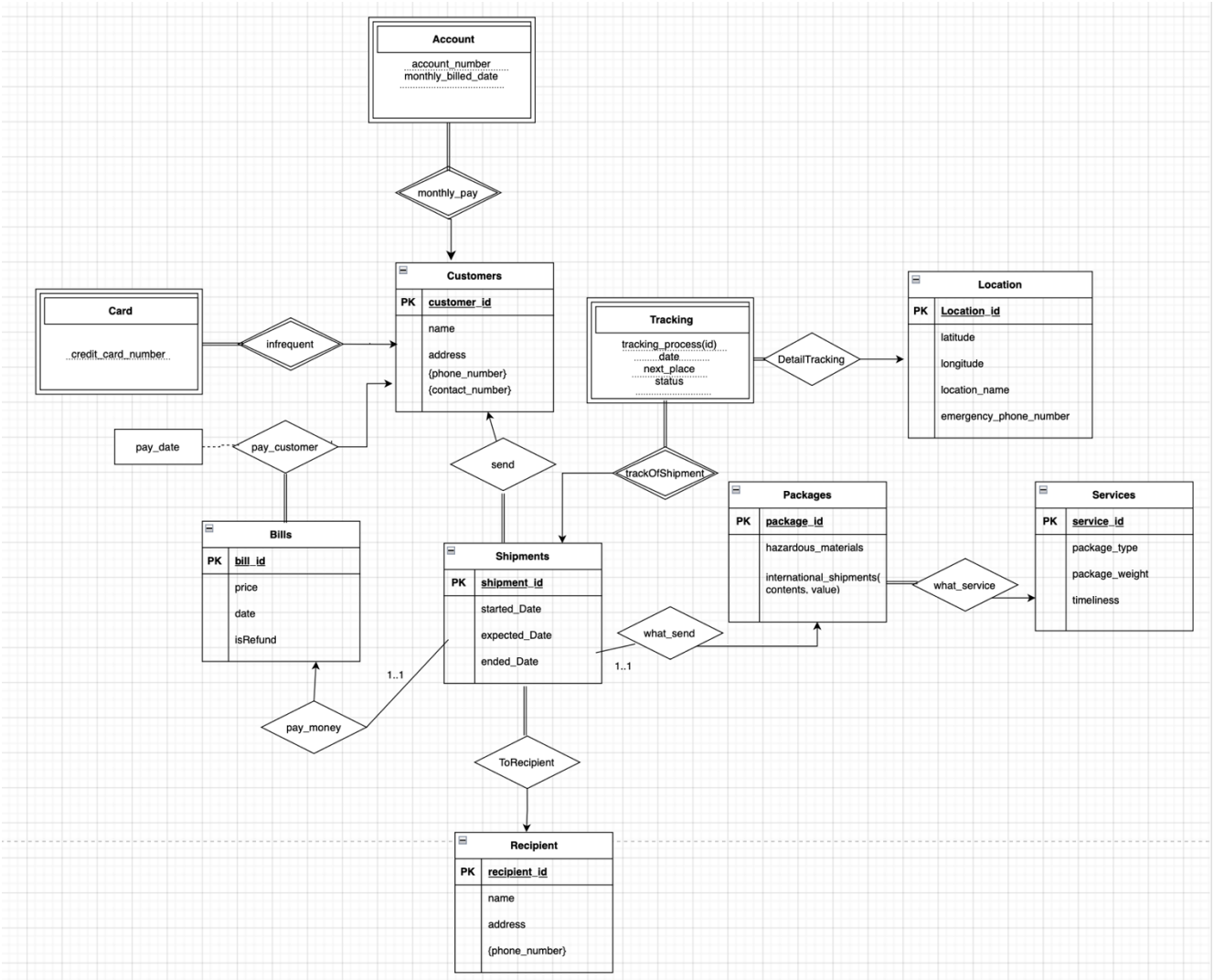
학번 : 20181256

0. 프로젝트 설명

이번 프로젝트는 E-R Model 과 relational schema design 에 대해 공부하고 이해하기 위해 데이터 베이스 시스템의 흔한 문제인 package delivery system 을 직접 구현해보는 것이다. 실제 배달 시스템을 고려한다면 결제, 결제 취소, 수취인, 발송인 고객 정보, 물건 정보, 물건의 서비스 내용, 물건 추적 기능, 수취인에게 잘 전달 되었는지, 등 많은 데이터들이 서로 엉키고 연관되어 있다. 여기서는 package handling, billing aspect 에 대한 데이터베이스에 초점을 맞췄다.

1. E-R Model Design

먼저 필자가 구상한 ER Diagram 도표를 사진 첨부하였다.



1-1. 구상 단계

우리는 여기서 초점을 두 군데에 두었다. 바로 Package handling, billing aspect 이다. 하지만 결국 근본적인 package delivery system 을 만들기 위해서는 가장 먼저 고객(customers) 이 필요하다. 그리고 배달에 대한 물건 (packages), 배달에 대한 정보(shipments), 배달에 관련 영수증 (Bills), 물건을 받는 사람, 수취인(recipient)이 필요하다.

Proj 명세서를 읽어보면 지불에 대한 자세한 내용이 나온다. 만약 customer 가 정기적인 고객이면 계좌를 등록하여 계좌에서 정기적으로 돈이 빠져 나가도록 한다. 그렇지 않다면 비 정기적인 고객이기 때문에 계좌가 아니라 따로 카드 결제를 통해 돈이 빠져 나가야 한다. 이 문장을 읽고 weak entity 관계가 생각이 났다. 결국 card, account 가 weak entity 로 있어서 identifying strong entity set 인 customer 를 구독(dependent) 하고 있으면 되지 않을까? 그리고 card, account entity 안에서 discriminator 를 두어서 한 customer 가 여러 카드나 계좌를 가지고 있어도 되도록 구상해도 될 것 같았다.

그리고 relationship 을 결정해야 하는데 relationship set 에 추가적인 attribute 을 기입 할 수도 있지만 이는 생각보다 직관적이지 못한 것 같아 최대한 entity set 으로 빼려고 노력하였다.

마지막으로 고려해야 할 부분은 tracking system 이였다. 이 경우 날짜에 대한 정보를 가지고 있어서 어떤 물건이 어떤 시간에 어디서 어떤 상태로 보관 중이고 곧 어느 지역을 지날 것인지 등 모든 세부적인 정보를 다 가지고 있어야 했다. 이런 것들 역시 따로 entity set 으로 가지고 있어야 한다고 생각을 하였다.

1-2. ER Model 설명

ER Model 이기 때문에 Entity 와 Relationship 에 대해 설명을 하도록 하겠다.

우선 Entity set 에 대한 설명이다.

- shipments

Shipment entity는 primary key로 shipment_id를 가진다. 이는 실제 세상의 배송 시스템과 비교를 하자면 택배 송장번호의 느낌이다. 그리고 attribute 로 택배의 출발 날짜(여기서는 DateTime 으로 볼 수 도 있다.) 그리고 예상된 날짜 즉 배송이 언제 이내로 도착할 것인지에 대한 것, 마지막으로 실제 도착 날짜를 적을 수 있도록 하였다.

- recipient

여기서는 수취인에 대한 정보를 기록하는 entity이다. 일단 수취인에 대한 recipient_id 를 primary key 로 가진다. 그리고 기본 정보인 수취인의 이름, 주소, 그리고 휴대폰 번호는 경우 여러 개 일 수 도 있으니 multi value 로 구성 하면 좋을 것 이다. Name, address 도 composite 하게 구성하면 더 세부 정보들을 접근할 수 있다.

- Packages

물건에 대한 정보를 기록하는 entity 이다. 물건의 id, package_id를 primary key로 잡았다. Proj1 명세서에 적힌 내용인 위험한 물건에 대한 정보와 국제 배송일 경우 필요한 해당 가치를 명시한 세관 신고서 같은 내용들이 필요할 것 같아서

attribute 으로 추가해주었다.

- Services

배달 서비스에 대한 정보를 기록하는 entity 이다. 물건을 보내기 위해서는 먼저 어떤 종류의 배달 서비스를 이용할 것인지, 해당 물건의 무게는 어느정도 일지, 도착 예정 시간은 어느 정도로 해야 할지 (쿠팡의 로켓배송, 로켓와우 같은 서비스) 등이 적혀 있다. 해당 종류별로 service_id를 다르게 하여 이를 primary key로 설정하였다.

- Customers

고객에 대한 정보로 어떤 고객이 물건을 배송업체에 맡기고 보내는 지에 대한 entity 이다. 고유한 고객 번호인 customer_id 를 primary key로 둔다. 그리고 해당 고객의 이름, 주소, 휴대폰 번호 와 연락 가능한 번호까지 attribute으로 둔다. 이 때 위의 recipient entity 에서와 비슷하게 이름, 주소 데이터 같은 경우는 composite attribute, 여러 개를 사용할 수 있는 번호 데이터 같은 경우는 multi value attribute 을 사용해서 더 자세한 정보를 기입하고 나중에 query 에서 더 쉽게 탐색 할 수 있다.

- Bills

고객이 택배를 보낸 영수증에 대한 정보를 기록한다. 고유한 번호인 bill_id 를 primary key로 두고 해당 물건을 보낼 때의 가격, 영수증이 발행된 날짜, 그리고 환불 여부를 통해 이 고객이 물건을 처음 발송한 입장인지 혹은 물건을 수령하고 어떠한 이유로(우리는 이 이유에 대해서까지 알 필요는 없다.) 환불 했을 때 환불 여부 정도만 알 수 있도록 하였다.

- Card

1-1 에서 잠깐 설명한 것 처럼 proj1 명세서에서는 정기 고객인지 아닌 지에 따라 결제 방식이 다르다. 여기서는 불규칙적인, 혹은 비 정기 고객들이 결제 할 수 있는 수단인 카드 정보를 저장하는 customer에 의존적인 weak entity 이다. Discriminator 로 credit_card_number를 둬서 고유한 카드번호를 통해 같은 고객이더라도 여러 개의 카드를 가질 수 있도록 하였다.

- Account

여기서는 card entity 와 달리 규칙적으로 거래를 하는 customer 에 대한 계좌 정보를 저장하고 있다. Attribute으로 account_number, monthly-billed_Date 를 둬서 계좌 번호, 그리고 언제 돈이 인출되는지에 대한 날짜 정보를 가진다. 이들을 discriminator 로 사용해서 primary key인 customer_id와 함께 고유성을 가지도록 한다.

- Tracking

핵심 기능 중 하나인 배송 추적 시스템 정보를 기록하는 entity 이다. 이 역시 shipment 즉 배송에 대한 정보 (택배 송장 번호) 가 있어야만 조회가 가능하다. 즉 weak entity 로 구성을 해서 shipment 에 의존적으로 연결이 되어 있어야 한다. Tracking process 는 shipment_id 와 동일 시 봐도 될 것이다. 하지만 여기서는 date, next_place, status 라는 attribute 을 따로 가지고 있다. 해당 택배를 추적하고자 하는 날짜(이를 통해 과거에 대한 추적, 현재의 추적까지 가능하다), 그리고 상태 (예를 들어 이 택배가 움직이고 있는지, 멈춰 있는지, 간선 상차, 하차 등 실제 택배 조회 시 나오는 상태 값들을 의미한다.) 마지막으로 다음 장소는 어디로 갈 것 인지 까지 적혀있다. Primary key로 shipment_id를 받고 diagram 처럼

discriminator로 고유성을 가지도록 하였다.

- Location

여기서는 tracking 에 대한 detail 한 정보를 적고자 추가한 entity 이다. 우선 primary key로 location 에 대한 고유 id를 가지도록 한다. 위치에 대한 attribute이라 할 수 있는 latitude, longitude, 위도 와 경도를 적어주고 그리고 해당 하는 지역 이름도 적는다. 마지막을 해당위치로 연락 할 수 있는 비상 연락망을 적으면 될 것이다. Query 예처럼 만약 택배 hub 가 아니라 현재 운송중인 택배 트럭 이나 배, 비행기 등이라면 latitude, longitude를 0 이나 Null 값을 주고 location_name 을 해당 운송수단 이름, location_id를 고유한 운송수단 id, 그리고 비상연락망에는 대표 운전기사의 번호를 적으면 될 것 이다.

다음은 relationship 에 대한 설명이다.

- Send

Send는 customer과 shipment 사이의 관계이다. 한 명의 customer가 여러 상품을 배송할 수 있기 때문에 one to many 관계로 설정했다. 이때 배송에 대한 정보인 shipment는 무조건 고객이 있어야 가능하기 때문에 total participation 이고 customer는 partial 이다.

- ToRecipient

Recipient와 shipment 사이의 관계이다. 한 명의 recipient 가 여러 shipment를 받을 수 있기 때문에 one to many 관계로 설정했다. Shipment에 대해 무조건 한 명의 수취인은 필요하기 때문에 total participation 이고 recipient는 partial 이다.

- What_send

Shipment 와 package 간의 관계로 어떤 물건을 보냈는지에 대한 relationship 이다. 여기는 한 물건당 하나의 송장번호가 발송되기 때문에 1 to 1 관계 이다. 이때 아직 송장번호가 안 나온 package 들도 있으니 partial 로 하고 shipment 의 경우 무조건 package가 있어야 해서 최소 1개 최대 1개 의미인 1..1 으로 cardinality 작성했다.

- What_service

Package와 service 관계로 여러 물건들이 한 택배 서비스에 대해 선택할 수 있기 때문에 many to one 관계로 설정하였다. 이때 물건은 무조건 하나의 택배 서비스를 가지고 있어야 한다고 판단해서 total로 두었다.

- Trackofshipment

Tracking 과 shipment 관계로 weak entity 관계이기 때문에 한 shipment가 여러 tracking을 가질 수 있어 one to many 관계로 설정했다. 그리고 이중 다이아몬드로 relationship을 그렸다.

- Detailtracking

Tracking 과 location 관계로 여러 tracking이 한 location을 가질 수 있어 many to one 관계로 설정했다. Tracking은 무조건 하나의 location을 가져야 하기 때문에 total로 설정했다.

- Pay_money

Bill 과 shipment 관계로 하나의 shipment 에는 무조건 하나의 bill이 필요하기 때문에 one to one 관계로 설정했는데 shipment에는 최소 1개, 최대 1개의 조건을 추가 했다.

- Pay_customer

Bill과 customer 의 관계로 한 customer가 여러 bill을 가질 수 있어 one to many 로 설정했다. 이때 이 relationship set은 pay_date라는 속성을 가져 실제 결제가 이루어진 날짜를 가진다. 그리고 bill은 무조건 하나의 고객이 있어야 해서 total로 설정했다.

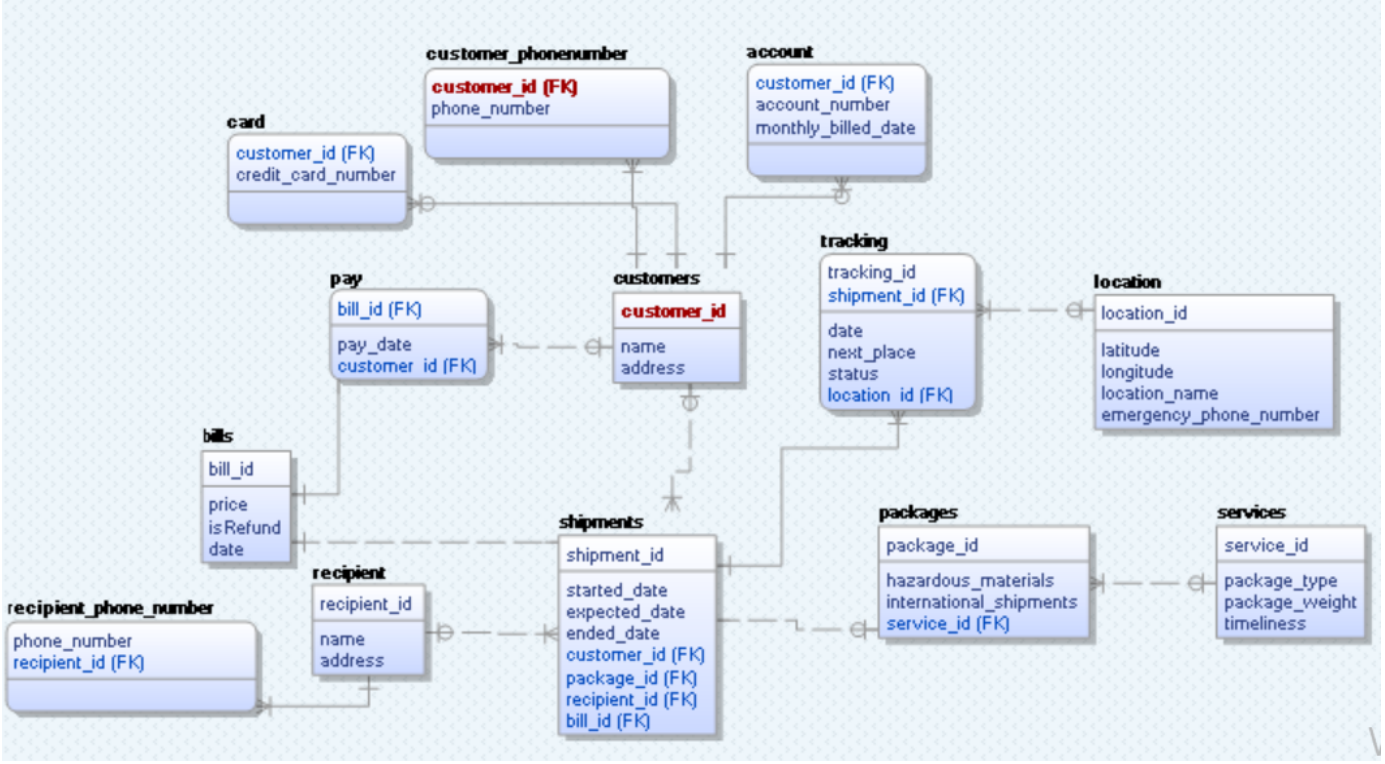
- Infrequent

Customer 과 card 사이의 weak entity 를 이어주는 관계로 한 고객이 여러 카드를 가질 수 있어 one to many 관계로 설정했다. Weak entity 이기 때문에 relation을 이중 다이아몬드로 그렸다.

- Monthly_pay

Account와 customer 사이의 관계로 한 고객이 여러 account를 가질 수 있기 때문에 one to many 관계로 설정했다. . Weak entity 이기 때문에 relation을 이중 다이아몬드로 그렸다.

2. schema diagram 설명



*Erwin 의 logical notation 을 information Engineering 으로 변경하고 진행하였다.

관계에 대한 정리는 위에서 상세하게 모두 설명하였다. 그래서 이번 schema diagram 에서는 어떤 점이 바뀌었는지, 어떻게 relationship set을 제거하였는지에 초점을 맞추겠다.

우선 대부분의 attribute은 null 값을 허용하지 않도록 하였다. 하지만 package entity, pay entity의 pay_date는 존재 하지 않을 수도 있어 이들 값은 Null이 들어와도 상관 없도록 했다. 날짜 date attribute 에 대한 데이터 타입은 datetime 을 주려고 했으나 해당 타입은 존재하지 않아 통일성을 유지하기 위해 Date 로 주었다. 그리고 tracking의 date attribute 은 자세한 날짜까지 모두 알고 있어야 하기 때문에 timestamp 로 주었다. Bills의 isRefund는 환불 여부를 묻는 거기 때문에 bool type, location entity 의 latitude, longitude 는 소수점이 필요하기 때문에 float 타입, 그리고 id 는 integer 값, 나머지는 모두 char[18] 로 주었다. 해당 ER model

에서는 대부분이 one to many 관계이다. 그래서 one 쪽의 primary key 를 many 쪽에 가져와 Foreign key 로 만들어 주었다.

기존의 pay_customer 라는 relationship 은 추가적인 attribute를 가지고 있었다. 이 relationship set을 지우기 위해 새로운 pay라는 entity를 추가했다. 여기서는 bill entity에 의존적인 (weak entity) entity 이기 때문에 bill의 primary key 인 bill_id 를 foreign key 이자 primary key로 설정해서 zero 를 허용하지 않는 1 to 1 의 관계로 두었다. 이 덕분에 customer와의 관계는 many to 1 의 관계가 되어 one 쪽인 customer의 primary_key 인 customer_id 가 pay의 foreign key 로 들어감을 알 수 있다.

Weak entity는 하나의 예시만 설명하겠다. 다른 예로 tracking entity는 원래 shipments 에 weak entity 관계이기에 shipments 의 primary key 인 shipment_id 를 foreign key 이자 primary key 로 받아 고유성을 유지하고 있다. 이들의 관계는 한 배송에 대해 여러 tracking 이 발생할 수 있으므로 one to many 이다. 이때 shipment는 무조건 하나의 tracking 이상이 필요하다.

또 주의해야 할 점은 multi valued attribute 이다. 이건 phone_number에 해당하는데 일반적으로 schema diagram 을 그릴때는 해당 정보를 따로 Entity 를 만들어 준다고 한다. 그래서 원래의 entity 의 primary key를 foreign key 이자 primary key로 설정하고 해당 휴대폰 번호는 모두 다르기 때문에 이 역시 primary key로 두어 고유성을 유지한다. 이때 customer, recipient 모두 필수적으로 휴대폰 번호는 필요하므로 zero를 허용하지 않았고 여러 개의 번호를 저장할 수 있기 때문에 one to many 관계로 설정하였다.

- Customers(<u>customer_id,</u> name, address)
- Shipments(<u>shipment_id</u>, started_date, expected_date, ended_date, customer_id(FK),

package_id(FK), recipient_id(FK), bill_id(FK))

- Packages(package_id, hazardous_materials, international_shipments, service_id(FK))
- Services(service_id, package_type, package_weight, timeliness)
- Tracking(tracking_id, shipment_id(FK), date, next_place, status, location_id(FK))
- Location(location_id, latitude, longitutde, location_name, emergency_phone_number)
- Recipient(recipient_id, name, address)
- Recipient_phone_number(phone_number, recipient_id(FK))
- Customer_phone_number(customer_id(FK), phone_number)
- Card(customer_id(FK), credit_card_number)
- Account(customer_id(FK), account_number,monthly_billed_date)
- Pay(bill_id(FK), pay_date, customer_id(FK))
- Bills(bill_id, price, isRefund, date)