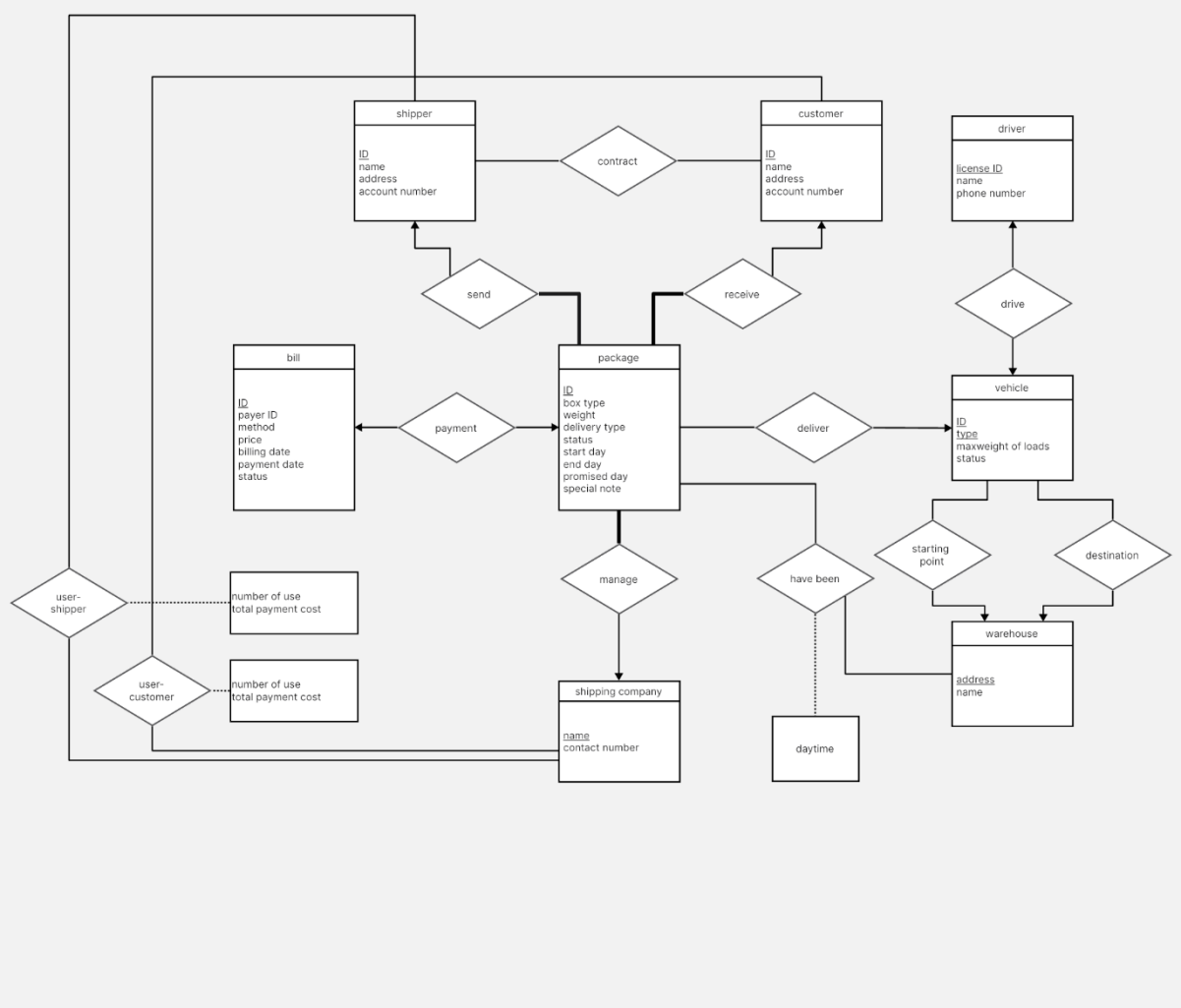
CSE4110 Database System

DB Project 1. E-R design and Relational Schema design

20181294 임승섭

1. E-R diagram



Entity Set

1. shipper

택배의 발송자를 의미한다. primary key로는 ID를 주었다. 발송자의 이름 또는 회사명은 name에 저장할 수 있게 하였고, 택배에 기입하는 발송자의 주소는 address에 저장한다. 환불 등 구매자의 계좌번호를 알아야 하는 경우가 생길 수 있기 때문에 계좌번호를 저장하는 account number가 있다.

2. customer

편의를 위해 shipper와 customer entity를 따로 생성하였다. primary key를 비롯한 attribute들은 shipper와 동일하다. 같은 사람이 shipper와 customer로 택배를 이용할 수도 있다. 이러한 경우는

shipper entity의 ID와 customer entity의 ID가 같은지를 확인해서 한 사람이 shipper와 customer로 이용한 것을 확인한다.

3. package

배송이 되는 택배를 의미한다. primary key로는 ID를 주었다. ID는 택배의 송장번호를 의미한다. 택배 박스에 대한 정보(flat, small, big, etc)는 box type에 저장된다. 택배의 무게는 weight에 저장된다. 배송 정보에 대한 내용을 저장하는 delivery type이 있다. 배송 정보는 일반 배송, 정기 배송, 환불 등 택배의 배송에 대한 종류를 의미한다. 택배의 상태를 저장하는 status가 있다. 택배의 상태는 배송 전, 배송 중, 배송 완료 등의 값이 저장된다. 배송 시작일, 배송 완료일, 약속된 배송 완료일은 각각 start day, end day, promised day에 저장된다. 택배의 특이사항은 special note에 저장한다. 특이사항은 내용물이 위험한지, 국제 배송인지 등의 정보를 담고 있다.

4. bill

택배의 결제 정보를 저장하고 있다. primary key로는 ID를 주었다. 모든 청구서마다 unique하게 있는 결제 번호를 의미한다. 결제자의 ID를 payer ID에 저장한다. 경우에 따라 shipper가 결제할 수도 있고, customer가 결제할 수도 있다. 결제 방식을 method에 저장한다. 결제 방식은 무통장 입금, 카드결제, 정기결제 등이 있다. 결제 가격은 price에 저장한다. 청구 날짜와 결제 완료 날짜는 각각 billing date와 payment date에 저장한다. 결제 상태는 status에 저장한다. 결제 전, 결제 완료, 결제 취소 등의 정보를 의미한다.

5. shipping company

택배를 보내주는 회사를 의미한다. primary key로는 회사명인 name을 주었다. 고객이 회사에 문의할 때 필요한 연락 번호를 contact number에 저장한다.

6. vehicle

택배를 배송하는 운송 수단을 의미한다. primary key로는 ID와 type을 주었다. ID는 번호를 의미하고 type은 종류를 의미한다. 예를 들면 트럭과 비행기가 같은 번호를 가질 수 있게 하기 위해 두 값을 primary key로 설정하였다. 운송 수단이 최대로 적재할 수 있는 택배 무게를 maxweight of loads에 저장한다. 이 값을 넘게 택배를 실을 수는 없다. 운송 수단의 현재 상태는 status에 저장한다. 현재 배송중인지, 정차중인지, 아니면 사고가 나서 수리중인지 등의 정보를 담고 있다.

7. warehouse

택배를 보관하는 물류창고를 의미한다. primary key로는 창고의 주소인 address를 주었고, 물류창고의 이름을 저장하는 name이 있다.

8. driver

운송 수단의 대표 운전자 한 명을 의미한다. 경우에 따라 누군지 확인할 필요가 있기 때문에 만들었다. primary key로는 운전자의 면허 번호인 license ID를 주었다. 여기서는 운송 수단이 다르면 license ID도 달라야 한다. 즉, 같은 사람이라도 여러 종류의 운송 수단을 운전할 줄 알면, 각각이 다른 tuple로 저장된다. 이번 프로젝트에서는 굳이 이걸 같은 사람으로 인식할 필요가 없고, 운송 수단에 더 중점을 두었기 때문에 이렇게 구현하였다. 운전자에게 연락하기 위해 필요한 운전자의 휴대폰 번호를 phone number에 저장한다.

Relationship Set

1. user-shipper

shipper와 shipping company 사이의 relationship이다. 회사 측에서 배송 서비스를 이용한 유저들을 저장하고, 이 때 누적 이용 횟수와 누적 결제 비용을 추가 속성으로 저장한다. 각각 number of use, total payment cost에 저장한다. 이는 나중에 가장 많이 이용한 고객 또는 가장 많은 돈을 사용한 고객을 찾을 때 유용하게 사용할 수 있다. 고객 입장에서는 다양한 회사를 이용할 수 있고, 회사 입장에서도 다양한 고객들이 있을 수 있기 때문에 many to many 관계이다.

2. user-customer

마찬가지로 customer와 shipping company 사이의 relationship이다. 내용은 user-shipper와 동일하다. 굳이 이렇게 두 개로 나눈 이유는 shipper와 customer를 따로 만든 이유와 동일하다. 같은 사람인지는 ID로 확인하고, 대부분의 정보는 이 사람이 shipper로 서비스를 이용한 경우와 customer로 서비스를 이용한 경우를 따로 저장해야 한다고 생각했다. 즉, 회사 측에서는 한 사람의 배송자 데이터와 수신자 데이터를 각각 따로 확인할 수 있다. 만약 통합해서 보고 싶다면 두 값을 합치기만 하면 된다.

3. contract

shipper와 customer 사이에 계약이 있을 경우, contract relationship으로 확인할 수 있다. 계약이

란, 따로 결제를 하지 않고, shipper와 customer 간 정기 결제가 존재하는 경우를 의미한다. 이러한 경우는 분명히 shipper와 customer 사이의 관계가 존재하기 때문에 이를 표시하기 위해 새로운 relationship을 구현하였다. shipper는 다양한 customer와 계약이 있을 수 있고, customer도 마찬가지로 다양한 shipper와 계약이 있을 수 있기 때문에 many to many 관계이다.

4. send

shipper와 package 사이의 관계이다. 즉, 발송자가 보낸 택배이다. 발송자는 여러 택배를 보낼 수 있고, 택배는 하나의 발송자밖에 존재할 수 없으므로 one to many 관계이다. (shipper : one, package : many). 택배는 무조건 발송자가 있어야 하므로 total이기도 하다.

5. receive

customer와 package 사이의 관계이다. 즉, 수신자가 받아야 하는 택배이다. 수신자는 여러 택배를 받을 수 있고, 택배는 하나의 수신자밖에 존재할 수 없으므로, 마찬가지로 one to many 관계이다. (customer : one, package : many). 또한 택배는 무조건 수신자가 있어야 하므로 total이다.

6. payment

bill과 package 사이의 관계이다. 현재 프로젝트에서 다루는 모든 택배는 청구서가 존재해야 한다. 각각은 서로 여러 개와 관계가 있을 수 없기 때문에 one to one 관계이다. 현재 택배의 결제 정보는 하나만 있을 수 있고, 어떠한 결제 정보는 하나의 택배에 대한 내용일 수 밖에 없기 때문이다.

7. manage

package와 shipping company 사이의 관계이다. 현재 택배 회사가 택배를 관리하는 것을 의미한다. 이를 통해 이용하는 택배 회사가 무엇인지 확인할 수 있다. 하나의 택배는 단 하나의 택배 회사만 있을 수 있고, 무조건 존재해야 한다. 택배 회사는 다양한 택배를 관리하기 때문에 one to many(package : many, shipping company : one)이고, package 입장에서 total이다.

8. deliver

package과 vehicle 사이의 관계이다. 택배가 지금 어떤 운송 수단에 있는지 확인한다. 택배는 하나의 운송 수단에 있을 수 있고, 운송 수단은 다양한 택배를 싣고 있기 때문에 many to

one(many : package, one : vehicle)이다. 주의해야 할 점은, 택배가 지금 물류 창고에 있을 수도 있다. 즉, vehicle과의 관계가 없는 택배가 존재할 수 있기 때문에 택배 입장에서 partial이다.

9. have been

package와 warehouse 사이의 관계이다. 추가 속성으로는 daytime이 있다. 택배가 이전에 있었던 물류창고들을 나타낸다. 언제까지 있었는지 확인하기 위해 날짜와 시간 정보를 저장하는 daytime을 추가해주었다. 위에 deliver에서 설명했듯이, 현재 택배가 물류창고에 있을 수도 있다. 이럴 때는, daytime값을 "present" 등으로 현재 있다는 것을 표현하는 값으로 저장해준다. 즉, 만약 택배가 deliver관계가 없다면, 무조건 have been 관계에서 현재 어디 있는지 확인할 수 있어야 한다. 택배는 여러 곳의 물류창고를 이동할 수 있고, 물류창고에 있었던 택배가 많을 수 있기 때문에 many to many 관계이다.

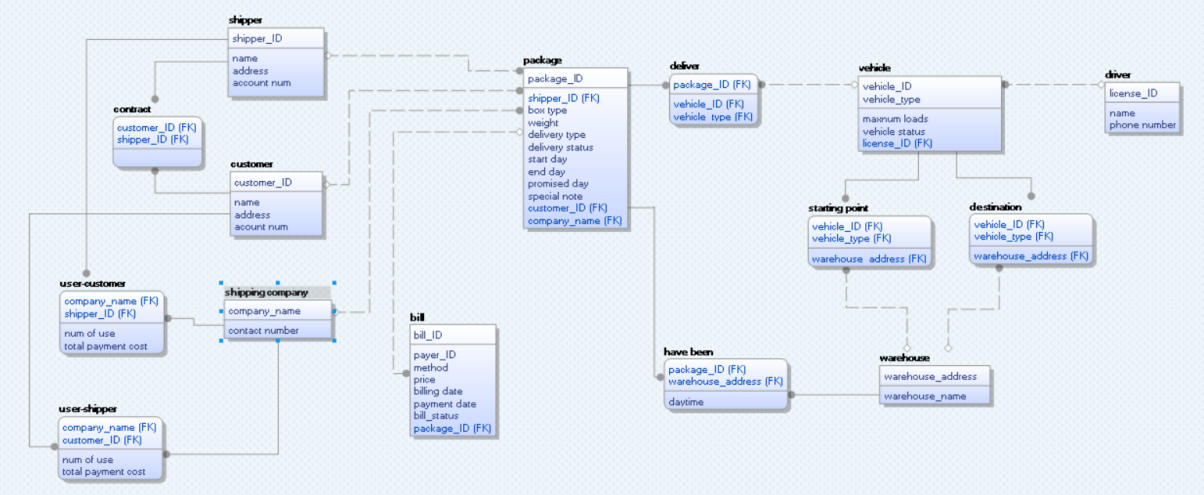
10. drive

vehicle과 driver 사이의 관계이다. 하나의 운송 수단에는 반드시 하나의 대표 운전자가 등록되어야 한다. 따라서 one to one 관계이다.

11. starting point / destination

운송 수단의 출발지와 목적지를 나타낸다. 각 운송 수단은 최대 하나의 출발지 또는 목적지를 가질 수 있고, 특정 물류창고를 출발한 운송 수단과 특정 물류창고로 오고 있는 운송 수단은 많을 수 있기 때문에 모두 many to one(many : vehicle, one : warehouse) 관계이다. 여기서 주의해야 할 점은 특정 운송 수단이 이 관계가 없을 수 있다. 먼저 출발지가 없는 경우는, 이제 배송을 출발하여, 출발지가 shipper의 address인 경우를 의미한다. 이 때는 물류창고에서 출발한 경우가 아니기 때문에 starting point 관계가 있을 수 없다. 또 목적지가 없는 경우는, 이제 배송 막바지여서 현재 목적지가 customer의 address인 경우를 의미한다. 이 때는 목적지가 특정 물류창고가 아니기 때문에 destination 관계가 있을 수 없다. 따라서 두 관계 모두 vehicle 입장에서 partial이다.

2. Relational Schema



E-R diagram을 기반으로 relational schema를 생성하였다. Reduction하는 과정에서 entity를 고려해야 하는 경우는 없었고, 주로 relationship에 대한 reduction을 하는 과정에서 새로운 스키마를 만들거나 entity의 속성을 추가하였다.

1. user-shipper, user-customer

many to many 관계이기 때문에 각 entity의 primary key를 primary key로 갖는 schema를 생성하였다. 즉, primary key는 shipper/customer의 primary key인 ID와 shipping company의 primary key인 company_name을 갖고, 기존에 있었던 추가 속성인 number of use와 total payment cost를 attribute으로 넣어주었다.

2. contract

many to many 관계이기 때문에 각 entity의 primary key를 primary key로 갖는 contract schema를 생성하였다. primary key로는 발송자의 ID와 수신자의 ID를 주었다.

3. send, receive

many to one관계인데, many side가 total이기 때문에 따로 추가 schema를 생성하지 않고, one side의 primary key를 many side의 속성으로 추가해주었다. 결국, package의 속성에 shipper ID와 customer ID가 추가되었다.

4. manage

위와 마찬가지로 many to one 관계인데, many side가 total인 경우이다. 마찬가지로 one side의 primary key를 many side의 속성으로 추가해주었다. package의 속성에 company name이 추가된 것을 확인할 수 있다

5. have been

many to many 관계이기 때문에, 각 entity의 primary key를 primary key로 갖는 have been schema를 생성하였다. 원래 있었던 'daytime'속성도 추가해주었다. 즉, primary key로는 package ID와 warehouse address를 갖고, 물류창고에 언제까지 있었는지의 정보를 담고 있는 daytime 속성이 있다.

6. payment

one to one 관계이기 때문에 양 쪽이 모두 many 역할을 할 수 있다. 즉, 한 쪽의 primary key를 다른 쪽의 속성으로 추가해준다. 여기서는 package의 primary key인 package_ID를 bill의 속성으로 넣어주었다.

7. deliver

many to one 관계이고 many side가 partial이다. 그래서 deliver라는 새로운 스키마를 생성하여 many side의 primary key를 primary key로 갖고, one side의 primary key는 속성을 갖게 하였다.

8. drive

one to one 관계이기 때문에 양 쪽이 모두 many 역할을 할 수 있다. 여기서는 driver의 primary key인 lincense ID를 vehicle의 속성으로 추가해 주었다.

9. starting point, destination

many to one 관계이고 many side가 partial이다. 그래서 각각 새로운 schema를 생성하고, many side의 primary key를 그대로 primary key로 주었고, one side의 primary key는 속성으로 주었다.

3. Conclusion

수업시간에 배운 E-R diagram과 Reduction to relational schema 내용을 기반으로 택배 시스템을 구현하였다. 처음 시작할 때는 막막하기만 했지만, 시간을 오래 잡고 하나씩 추가하고 수정해가다 보니 결국 마무리를 지을 수 있었다. 아쉬운 점은 weak entity나 multivalued 같은 특이 케이스를 어쩌다보니 전혀 사용하지 않았다는 것이다. 구현하기 전에는 어떤 점을 multivalued로 설정해야겠다 생각했는데 막상 하다보니 새로운 entity를 생성해서 one to many 등으로 완성시켜버렸다. 실제 내 diagram에 있다면 공부하는 데 더 도움이 되었을 것 같아서 그 점이 아쉽다.

결과적으로는 만족스러운 그림이 나왔다. 현재로서는 최선의 결과를 도출하였다. 아마 실제 Query를 돌려보면 오류가 생기는 점을 확인할 수 있을 것 같다. 이번 결과를 기반으로 다음 프로젝트도 진행하며 어떤 문제가 생길 수 있는지 확인하며, 기존에 생각했던 점들을 고쳐갈 예정이다.