가상의 택배 회사에 대한 ER-Diagram  
- 데이터베이스 미니프로젝트1 보고서

1. 주제 선정
2. 요구 사항 분석
3. Entity 설계
4. 참여 관계, 순환 관계, 3 Degree
5. Weak entity type 설계
6. Partial Key 설계
7. ER-Diagram

경제통상학과 18012248

신민기

1. 주제 선정

주제는 가상의 택배 회사를 선정했다. 이 회사는 고객에게 택배를 전달하는 것을 주목적으로 한다. 택배에 대한 반송, 교환 등의 업무는 타 회사에서 진행한다고 가정한다.

1. 요구 사항 분석

ER-Diagram을 설계하는데 있어 다음의 조건들을 만족하도록 한다.

1. Entity 설계

엔티티는 5개로 제한하도록 한다.

택배 회사의 엔티티는 customer(고객), package(택배 물품), driver(배달원), car(배달원의 차량), branch(지점)이다.

1. 참여 관계, 순환 관계, 3 Degree

전체 참여, 부분 참여, 순환적 관계, 차수가 3인 관계, 1개 이상을 포함해야 한다.

**Customer**

* Package와 1:N 관계를 가진다. Driver와 1:1 관계를 가진다.

**Package**

* Customer와 N:1 관계를 가진다. Driver와 N:1 관계를 가진다.

**Driver**

* Package와 1:N 관계를 가진다. Car와 1:1 관계를 가진다. Branch와 N:1 관계를 가진다.

**Car**

* Driver와 1:1 관계를 가진다. Branch와 N:1 관계를 가진다.

**Branch**

* Driver와 1:N 관계를 가진다. Car와 1:N 관계를 가진다. Low level Branch와 High level Branch 간의 순환적 관계를 가진다.

1. Weak entity type 설계

Customer-Package / Package-Driver / Driver-Car-Branch / Branch-Branch relation이 있다.

Customer는 Package와의 관계에서 부분 참여, Package는 전체 참여다. Customer의 pk를 Package의 fk에 넣어 관리할 것이다. Package는 weak entity type이 되고, Customer는 owner entity type이 된다.

Package는 Driver와의 관계에서 전체 참여, Driver는 부분 참여다. Driver의 pk를 Package의 fk에 넣어 관리할 것이다. Package는 weak entity type이 되고, Driver는 owner entity type이 된다.

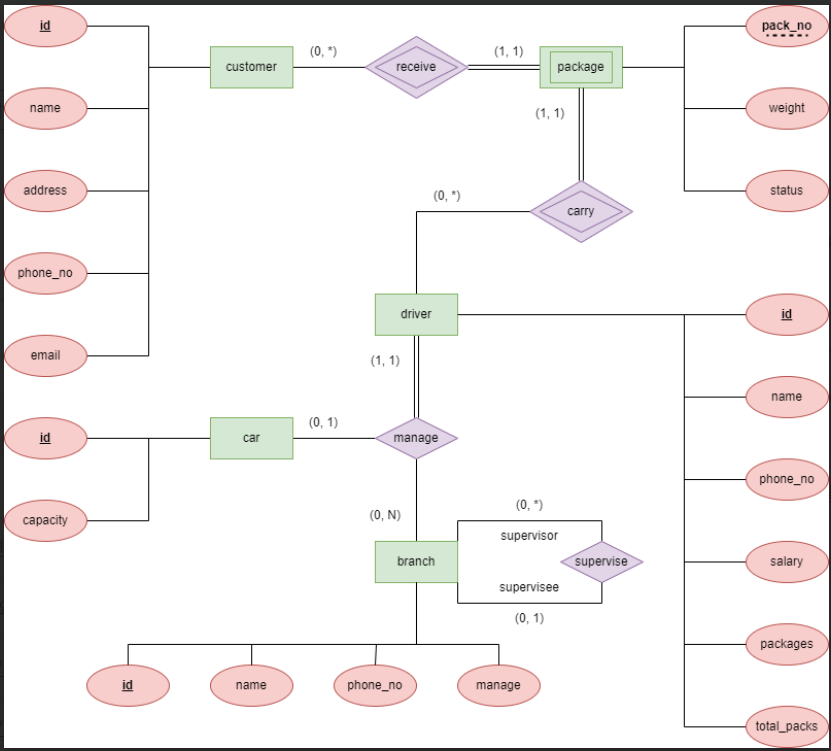
Driver-Car-Branch는 Manage라는 하나의 relation에 속한다. 이 관계에서 Driver와 Car는 1:1 관계를 가지고 Car와 Branch, Driver와 Branch는 N:1 관계를 가진다.

Branch는 내부적으로 순환 관계를 가질 수 있다. 어떤 한 Branch는 상위 Branch의 관리를 받을 수 있기 때문이다. Branch table내에 Branch의 pk를 참조하는 fk를 애트리뷰트로 하여 순환 관계를 성사 시킨다. High level Branch와 Low level Branch는 1:N 관계를 가진다. 상위 Branch는 여러 하위 Branch를 관리할 수 있다.

1. Partial Key 설계

Weak entity type인 Package의 key를 부분키로 설정한다. Customer-Package의 relation을 예로 들면, Customer와 Package 각각의 엔티티의 Key가 있을 것이다. Customer의 Key는 pk가 될 수 있다. 반면, Package는 Customer에 의존적이기 때문에 Package의 Key 자체로 식별의 역할을 다할 수 없다. Package는 Customer의 pk를 본인 테이블의 외래키로 가져와 식별한다. Package의 Key는 partial Key가 된다.

1. ER-Diagram



해당 ER-Diagram은 ER-Diagram tool인 draw.io 사이트를 사용했다.

**customer**

- attribute는 pk인 id와 이름(name), 주소(address), 전화번호(phone\_no), 이메일(email)로 구성한다. Customer의 pk는 package에 의해 참조된다.

- package와 1:N 관계를 갖는다. Customer 1명을 선택했을 때, 대응되는 package 수는 0~N 개의 범위를 갖는다.

- 한 customer의 package의 status가 최종 단계에 도달하면, package table의 튜플이 삭제된다. Customer에 대응되는 package가 0일 경우, customer가 수령할 택배가 없음을 의미한다.

**Package**

- attribute는 택배 번호(pack\_no), 무게(weight), 배송 상태(status)로 구성한다. Customer의 pk값을 참조하며 partial key인 pack\_no로 식별자를 구성한다.

- customer와 N:1 관계를 갖는다. Package 1개를 선택했을 때, 대응되는 customer 수는 1~1 범위를 가지며 전체 참여다.

- driver와 N:1 관계를 갖는다. Package 1개를 선택했을 때, 대응되는 driver 수는 마찬가지로 1~1 범위를 가지며 전체 참여다.

- 배송 상태는 정수로 표현한다. Table 내에 튜플이 생성되면 status는 0으로 시작한다. 한 택배는 1개의 driver를 배정받는다. Driver가 배정되면 status를 1로 변경한다. Status가 4에 도달하는 순간 customer에게 배송이 완료되었다 판단하고 튜플을 삭제한다.

**Driver**

- attribute는 pk인 id와 이름(name), 전화번호(phone\_no), 월급(salary), 배송할 택배 개수(packages), 월 누적 택배 개수(total\_packs)로 구성한다.

- package와 1:N 관계를 갖는다. Driver 1명을 선택했을 때, 대응되는 package 수는 0~N 범위를 갖는다.

- Car, Branch와 degree가 3인 relation을 갖는다. Driver 1명을 선택했을 때, 대응되는 Car의 수는 1~1 범위를 갖는다. 대응되는 Branch 수는 1~1 범위를 갖는다. Driver는 소속 지점과 배송 수단이 무조건 1개가 mapping 되어야 하는 전체 참여 관계다.

- Driver는 package와 1:N 관계를 맺는다. 다수의 package를 책임질 수 있다. Package와 매핑되면 packages라는 attribute를 1씩 올린다. Packages attribute는 Car 엔티티의 capacity의 영향을 받는다. Driver는 사전에 매핑된 Car 인스턴스가 있다. Car 인스턴스의 capacity가 20이라면, Driver는 20개 이상의 택배를 배정받을 수 없다.

- Driver는 Branch와 1:N 관계를 맺는다. Driver는 무조건 소속 branch가 있어야 한다. Driver는 Car와 1:1 관계를 맺는다. Driver는 무조건 매핑된 car 인스턴스가 있어야만 택배 운송을 할 수 있다. 따라서 해당 relation에서 Driver는 전체 참여다.

- Driver의 Salary는 total\_packs의 영향을 받는다. Total\_packs는 0과 양의 정수 범위를 가진다. 0일 경우, 월급은 없다. 0~4일 경우, 기본급을 수령할 수 있다. 5 이상일 경우, 기본급 + (누적 택배 개수 X 10000)원의 월급을 수령한다. Total\_packs는 package와 매핑되는 시점에 1씩 증가하고 배송이 완료되는 경우에도 total\_packs는 줄어들지 않는다. 다만, 매월 초에 0으로 초기화된다.

**Car**

- attribute는 pk인 id와 차에 실을 수 있는 택배의 개수(capability)로 구성한다. 해당 엔티티의 인스턴스와 매핑된 drive는 배송할 택배 개수(packages)를 capability를 초과하지 못한다는 제약이 생긴다.

- Driver, Branch와 degree가 3인 relation을 갖는다. Car 1개를 선택했을 때, 대응되는 driver 수는 0~1 범위를 갖는다. Car의 입장에서 driver를 지정받지 못한 car가 존재할 수 있다는 것이다. 대응되는 branch 수는 1~1 범위를 갖는 전체 참여 관계다. 어느 한 car는 소속 branch가 항상 있어야 한다.

- Car는 Driver와 1:1 관계에서 부분 참여다. Driver는 무조건 Car가 필요하지만, Car는 배정받은 Driver가 없을 수 있다.

- Car는 Branch와 N:1 관계에서 전체 참여다. Car는 무조건 Branch에 소속되어야 한다.

**Branch**

- attrubute는 pk인 id와 지점 이름(name), 지점 번호(phone\_no), 상위 지점(manage)으로 구성한다.

- Driver, Car와 degree가 3인 relation을 갖는다. Branch 1개를 선택했을 때, 대응되는 driver의 수는 0~N 범위를 갖는다. 대응되는 Car의 수는 0~N 범위를 갖는다.

- Branch는 자기 자신의 엔티티와 순환적 관계를 갖는다. 하위 지점을 Low branch, 하위 지점을 관리하는 상위 지점을 High branch라 가정하면, Low branch는 High branch와 N:1 관계를 가진다. Low branch 1개를 선택했을 때, 대응되는 High branch 수는 0~1의 범위를 갖는다. 대응되는 High branch가 0이라면 해당 지점은 최상위 지점 혹은 본사가 될 것이다. High branch는 Low branch와 1:N 관계를 가진다. High branch 1개를 선택했을 때, 대응되는 Low branch 수는 0~N 범위를 가진다. 대응되는 Low branch가 N이라면 해당 지점은 많은 하위 지점을 관리하는 것을 의미한다. 0의 경우, 해당 지점은 관리하는 지점이 없는 최하위 지점을 의미한다.

- Branch는 Driver와 1:N 관계에서 부분 참여다. 해당 branch에 소속된 driver는 0일 수도 있다.

- Branch는 Car와 1:N 관계에서 부분 참여다. 해당 branch에서 관리하는 배달 운송 수단은 0일 수도 있다.