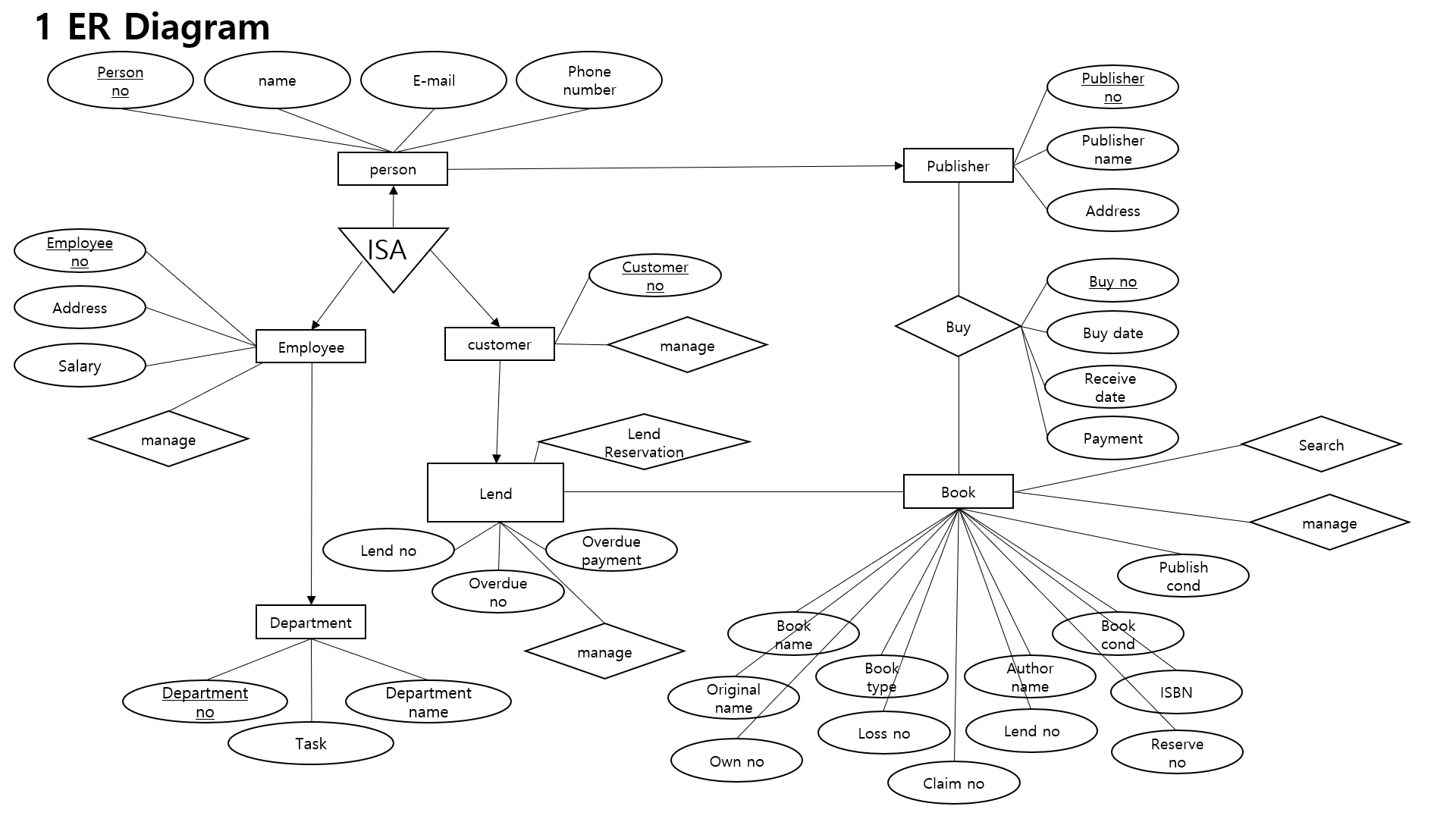
**데이터베이스시스템 프로젝트1**

20171273 심현우



1. Entity 도출

-Person

-Employee

-Customer

-Publisher

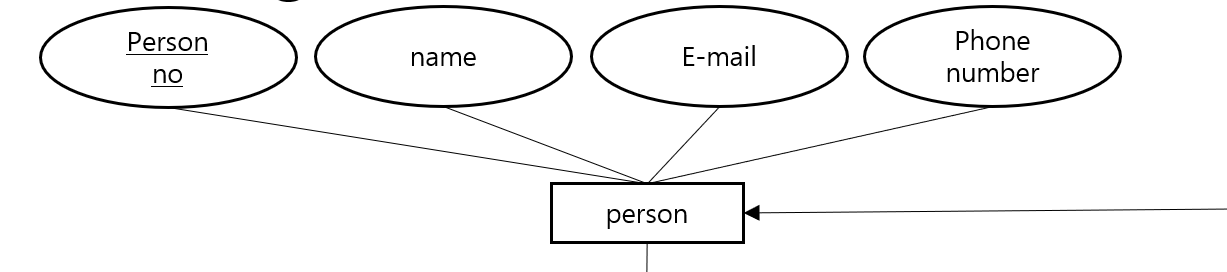
-Book

-Lend

-Department

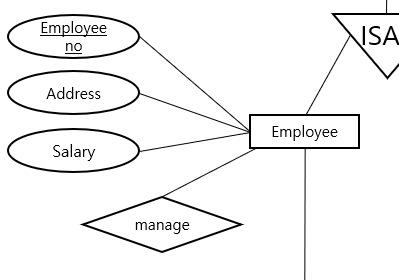
1. Attributes 도출

-Person : Person no(PK), name, E-mail, phone number



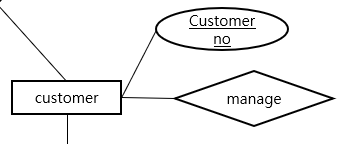
요구사항의 사람들은 직원, 회원, 출판사 담당자로 나뉜다. 그들의 공통적으로 갖게 되는 기본 정보, 즉 이름, 이메일, 전화번호를 묶어서 person entity로 구성했다. Person no는 그 사람의 고유 값, 즉 integer 값을 갖는다.

-Employee : Employ no(PK), Person no(FK), Department no(FK) Address, Salary



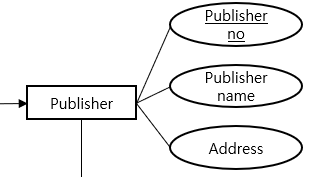
직원은 회원이나, 출판사 담당자가 가지고 있는 정보 외에 추가로 연봉, 주소, 부서, 담당업무를 가져야한다. 따라서 따로 부서 entity를 주어 부서와 담당업무에 대한 정보를 Department no(FK) 외래키를 통하여 받고, 나머지 정보들은 Attributes로 설정해주었다. 직원 고유의 사번도 필요하므로 Employee no(PK)를 통해 부여했다. 또한 직원이 직원 정보에 대해 관리할 수 있는 어플리케이션이 필요하므로 manage relationship을 설정해 필요한 정보를 조회할 수 있도록 설정했다.

-Customer : Customer no(PK), Person no(FK), Lend no, Overdue no, Overdue Payment



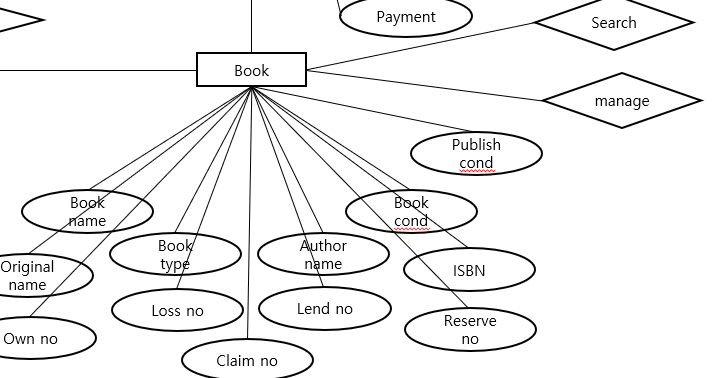
회원 또한 회원 고유 번호가 필요하므로 Customer no(PK)를 통해 부여했다. Person no(FK)를 외래키로 받아 기본적인 정보를 갖도록 설정했다. 이후 Lend no, Overdue no, Overdue Payment는 대출횟수, 연체횟수, 연체대금으로 회원 정보를 열람 시 같이 보여주어야 하는 정보이기에 Attributes로 설정하여 회원에 따른 데이터를 갖도록 설정했다.

-Publisher : Publisher no(PK), Person no(FK), Publisher name, Address



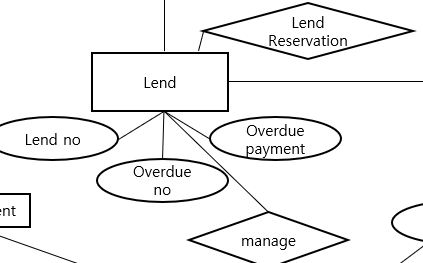
출판사는 출판사 코드를 고유번호로 하고 출판사의 이름, 주소 데이터를 설정하기 위해 attributes를 추가했다. 또한 출판사에는 한 명의 담당자가 필요하므로 person entity와 one to one relationship을 통해 설정했다. 담당자에 대한 전화번호, 이메일, 이름 정보도 person no(FK) 외래키로 받아 설정했다.

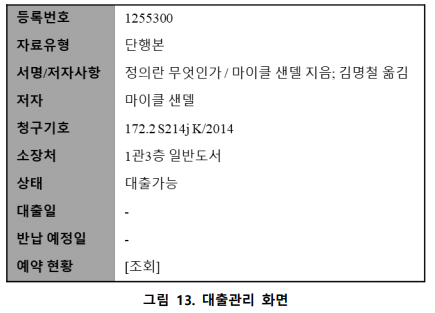
-Book : Book code(PK), Buy Code(FK), Publisher no(FK), Person no(FK), book name, book type, book conditions, publish conditions, author name, original name, loss no, lend no, reserve no, own no, ISBN, Claim Code



책은 포함해야 할 정보가 많아 우선 정리하여 attributes로 설정했다. Book은 출판사에서 구매할 때 등록번호가 생성되므로 book과 publish를 buy라는 relationship으로 연결해주었고 buy를 entity로 설정해 구매할 때 생성되는 buy code(FK)를 외래키로 받아 데이터를 설정했다.

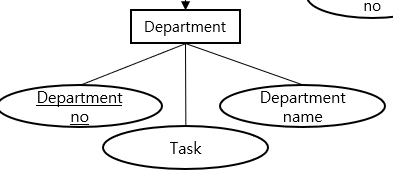
-Lend : Book Code(FK), Buy code(FK), Publisher Code(FK), Person Code(FK), Customer Code(FK), book type, book name, author name, claim code, book location, lend conditions, lend date, return schedule





대출 또한 대출관리화면에서 보면 확인할 수 있듯 요구하는 정보가 많기 때문에 attributes를 추가했다. 대출은 회원이 책을 빌리는 것이므로 회원정보와 책 정보가 필요하기 때문에 customer code(FK)와 Book Code(FK) 외래키를 통해 필요한 정보들을 가져왔다.

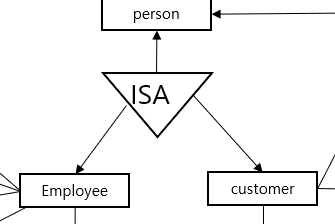
-Department : Department code(PK), Department name, Task



부서와 담당업무는 department no(PK)를 주어 employee에서 정보를 받아갈 수 있도록 설정했고, 부서 이름과 담당업무 이름을 attribute로 설정했다.

1. Relation & Mapping Cardinality

-person(employee, customer)



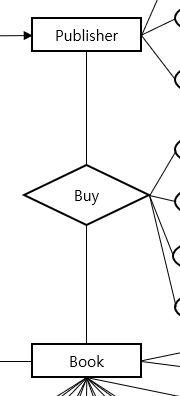
회원과 직원은 모두 사람이기 때문에 사람으로 묶었다. 한 사람은 한 가지 일만 할 수 있고 또한 한 사람은 한 명의 회원이 될 수 있다. 또 하나의 회원 아이디, 하나의 업무 또한 한 사람이 할 수 있으므로 one to one relationship으로 설정했다.

-person, publisher



요구사항에서 publisher는 출판사로 도서를 구입하는 회사를 의미한다. 여기서 출판사는 담당자 한 명을 대표로 그의 이름, 전화번호, 이메일을 정보로 갖는다. 따라서 출판사와 사람의 관계가 형성되고 출판사 하나당 담당자 한 명이 배정된다. 하지만 한 사람은 여러 출판사의 담당자가 될 수 없다는 요구사항이 없으므로 한 사람이 여러 출판사에서 일하며 동시에 담당자가 될 수 있다. 따라서 many to one relationship으로 설정했다.

-Publisher, Book ( Buy )



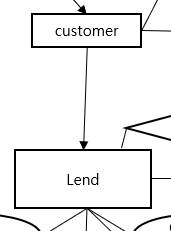
직원이 도서를 구매할 때 출판사를 이용하여 구매한다. 여기서 도서와 출판사는 관계를 갖게 되고 이를 buy relation으로 설정했다. Buy relation으로 설정한 이유는 책을 구매할 때 등록번호, 청구기호, 서명, 구매일, 반입일, 금액 정보가 필요하므로 relationship의 attribute로 설정했다. 여기서 출판사 하나는 여러 책을 판매할 수 있고, 하나의 책도 여러 출판사에서 구매해 올 수 있으므로 many to many relationship으로 설정했다.

-Book, Lend



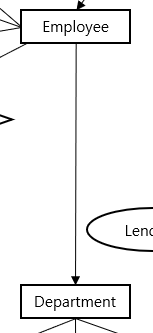
현재 도서를 대출하는 시스템을 운영하고 있는데 이는 회원과 책과의 관계이다. 이때 대출이라는 행위에 필요한 정보량이 많고 또한 대출을 예약하는 시스템이나, 회원 별, 도서 별 대출 현황을 다뤄야하는 경우가 있어 따로 entity로 설정해주었다. Lend와 book과의 관계는 하나의 책은 여러 번 대출이 가능하고, 또한 한 번의 대출은 여러 책을 동시에 할 수 있으므로, many to many relationship으로 설정했다.

-customer, lend



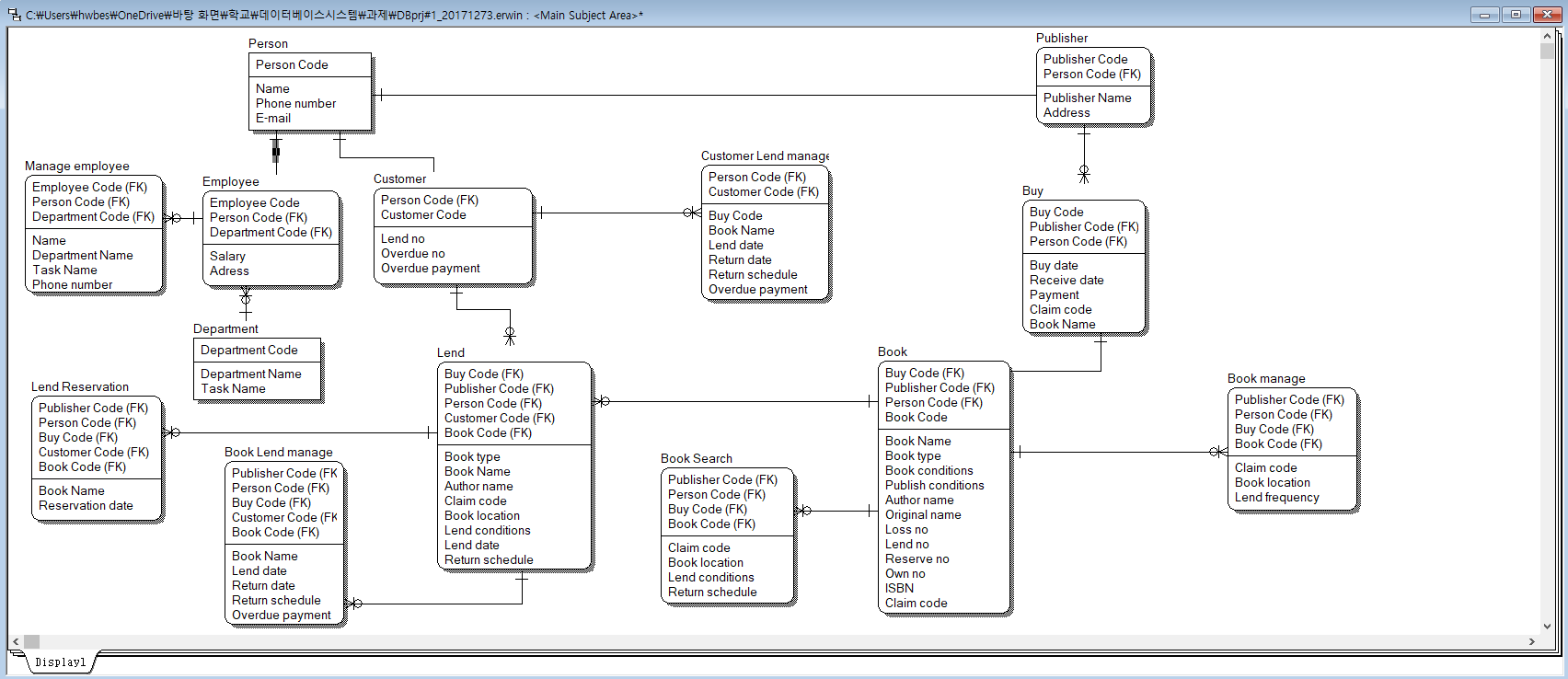
위와 비슷하게 회원과 대출의 관계를 보자. 회원 한 명은 여러 번의 대출을 할 수 있다. 이는 회원 별 대출현황에서 확인할 수 있다. 하지만 한 번의 대출은 한 명의 회원에게서만 할 수 있기 때문에 one to many relationship으로 설정했다.

-employee, department

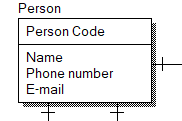


마지막으로 직원과 부서의 관계이다. 요구사항에는 직원과 부서에 관한 다른 이야기는 없으나 직원 정보 조회 화면이나, 전체 직원 리스트 조회 화면의 정보로 미루어 보아 한 명의 직원은 하나의 부서와 담당업무를 갖는 것으로 보인다. 하지만 한 개의 부서에 한 명의 직원만 해당되어야 하는 것은 아니므로, one to many relationship으로 설정했다.

**2 Logical mode**

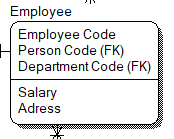


1. entity
2. Person



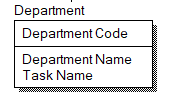
사람이라는 entity이다. Person code(PK)를 프라이머리 키로, 이름, 전화번호, 이메일을 attribute로 가진다.

1. Employee



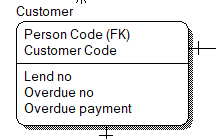
직원 entity이다. 직원은 person entity에서의 사람에 대한 기본 정보를 받아야 하므로 person code를 외래키로 하고, 또한 부서와 업무에 대한 정보도 받아야 하므로 department entity의 PK, 즉 department code를 외래키로 받는다. attribute로는 연봉과 주소가 있다.

1. Department



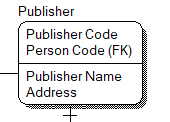
직원의 부서와 업무에 대한 entity로 department code를 PK로 가지며 attribute로는 부서의 이름, 업무의 이름(내용)이 있다.

1. Customer



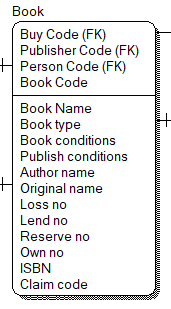
회원에 대한 entity로 직원과 같이 사람에 대한 기본 정보를 person entity에서 받아온다. 따라서 person code를 외래키로 받으며 회원에 대한 상세 정보로 대출횟수, 연체횟수, 연체금액을 따로 attribute로 지정했다. 회원 정보에 대한 열람을 위해 customer code를 PK로 하여 manage할 때 쓰도록 했다.

1. Publisher



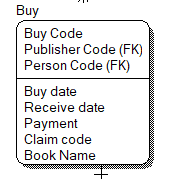
출판사는 출판사 이름, 위치(주소)의 정보를 가지고 있으며 담당자 한 명의 이름, 전화번호, 이메일을 데이터로 갖는다. 따라서 담당자의 정보는 person code를 외래키로 받아 가져왔고, 나머지는 attribute로 설정하여 할당했다.

1. Book



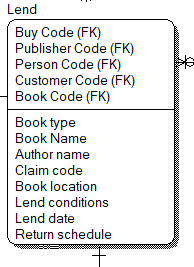
책은 출판사에서 구매하므로 Publisher code와 buy code를 외래키로 받았다. Publisher에서 person code를 외래키로 받았으므로 book과 publisher를 relation할 때 person code도 같이 외래키로 받았다. Book code라는 PK를 설정하여 다른 entity에서도 book의 정보를 사용할 수 있도록 했다. 나머지 책의 이름, 유형, 발행상태, 저자 등등 필요한 정보들을 attribute로 설정했다.

1. Buy



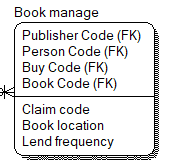
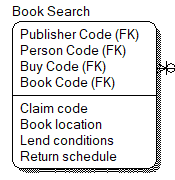
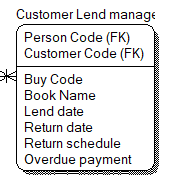
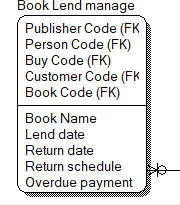
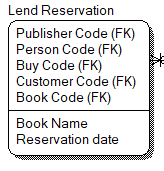
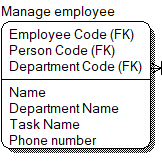
Book과 publisher사이의 relation entity이다. 책을 구매할 때 책의 구매 고유번호인 등록번호가 설정된다. 이를 buy code로 했다. Buy는 publisher와 relation을 맺고 있으므로 publisher entity의 PK와 FK를 외래키로 받아왔다. 그 외 구매일, 반입일, 금액은 구매의 상세 정보로 attribute로 지정했다. 나머지 claim code와 book name은 book entity의 attribute로 책의 정보지만 출판사의 대출리스트를 확인할 때 화면에 띄워야 할 정보이므로 중복되게 attribute로 추가했다.

1. Lend



대출 entity이다. 대출은 회원과 책의 관계로 buy와 같지만 필요한 정보량이 많고 어플리케이션으로 다뤄야 할 내용이 많기 때문에 entity로 설정했다. Lend는 책의 정보와 회원의 정보를 다 담아야 하기 때문에 Book과 customer의 PK와 FK를 외래키로 받았다. 또한 대출만의 상세 정보, 대출일, 반납예정일을 attribute로 추가하였다. 이 외 책 유형, 책 이름, 저자 등등 중복되는 attribute는 요구사항의 대출관리화면 정보를 조회하기 위해 따로 다시 설정했다.

1. Manage, Search

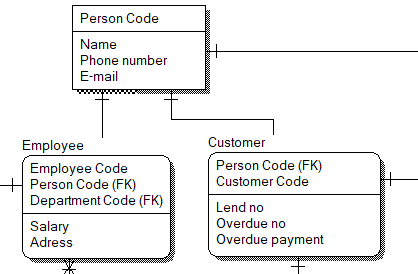


Manage와 search, reservation entity는 각 entity의 relation entity로 도서관 관리 어플리케이션의 전반적인 내용과 화면 출력을 위해 만든 entity이다. 요구사항의 조건대로 직원 전체의 정보를 조회할 때 manage를 사용하여 출력한다. 그 외 직원의 상세정보는 employee entity에서 할 수 있다. 마찬가지로 도서 별 예약 정보 화면을 띄우기 위해선 Customer Lend manage를 통해 원하고자 하는 형식으로 정보를 확인할 수 있다. 요구사항을 만족하기 위한 어플리케이션의 entity이다.

1. Relation & Mapping cardinality

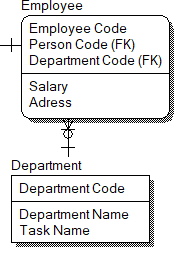
위의 ER diagram의 설명과 동일하다.

1. Person(employee, customer)



사람과 직원, 사람과 회원은 one to exactly one relationship으로 연결했다.

1. Employee, department



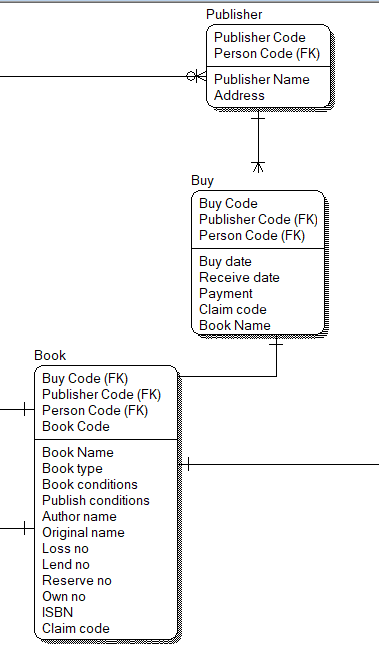
부서에서 직원은 부서 하나당 여러 직원을 가질 수 있으므로 one to zero or one or more relationship을 사용했다.

1. Person, publisher



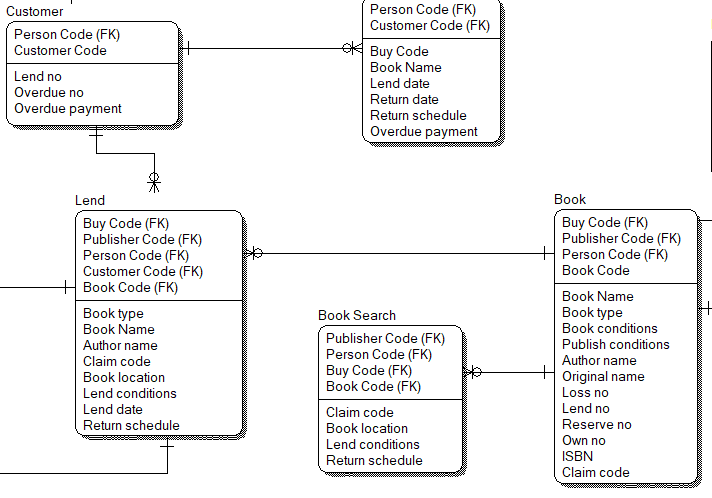
한 사람은 여려 출판사의 담당자가 될 수 있으므로 one to zero or one or more relationship을 사용했다.

1. Publisher, buy, book



출판사와 책의 관계에서 구매는 출판사 하나당 여러 책을 판매할 수 있고 buy code 한 번에 하나의 책을 구매할 수 있으므로 출판사에서 buy는 one to one or more로 buy에서 book은 one to exactly one relationship으로 설정했다.

1. Customer, Lend, Book

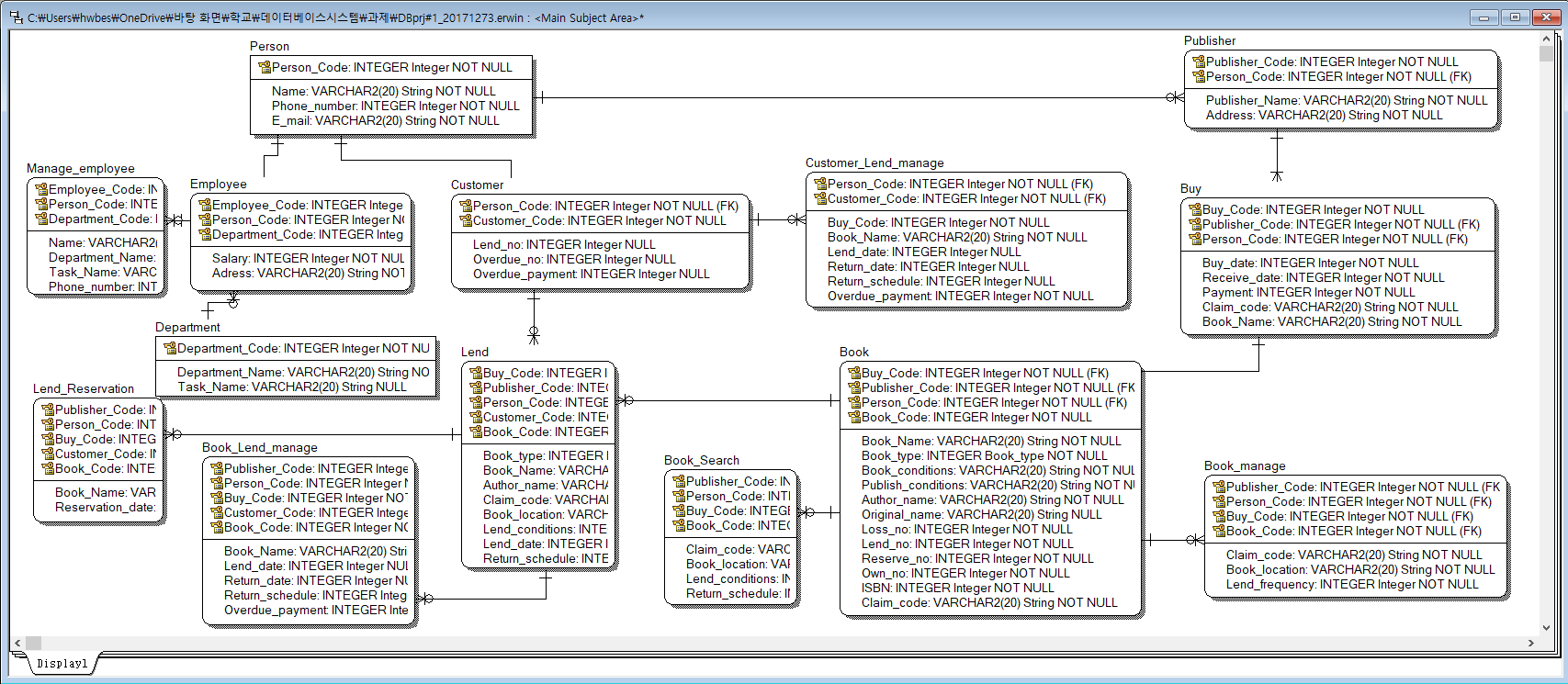


회원과 책, 그리고 대출의 관계는 회원 한명은 여러 대출을 할 수 있고, 책 한권은 여러 번 대출될 수 있으므로 one to zero or one or more로 설정했다.

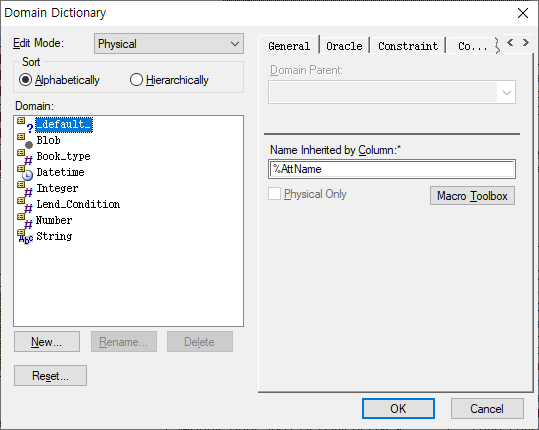
1. 그 외 manage, search, reservation

그 외 어플리케이션에 대한 entity들은 여러 번 사용할 수 있으므로 one to zero or one or more으로 설정했다.

**3 Physical Mode**



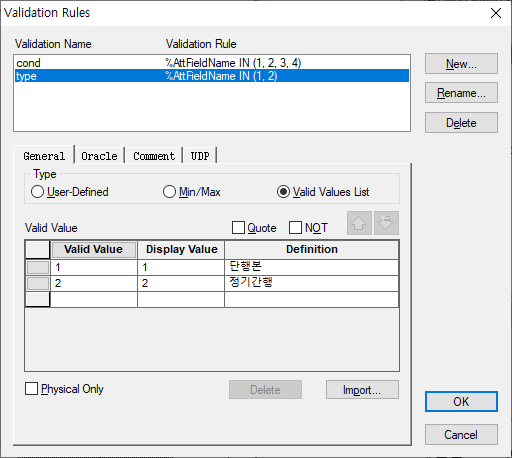
1. Domain 설정



도메인은 Integer와 String을 사용했다.

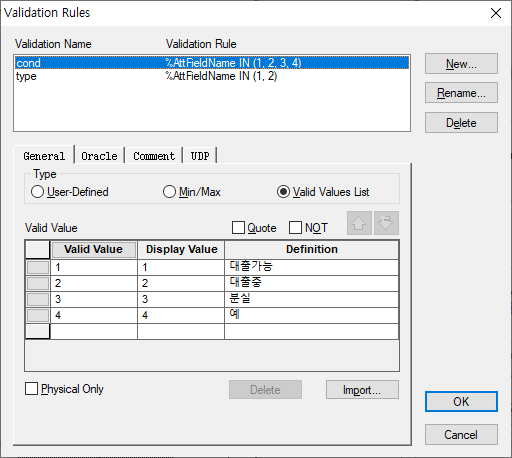
새로 지정한 도메인은 Book type과 Lend Condition이다.

-Book type



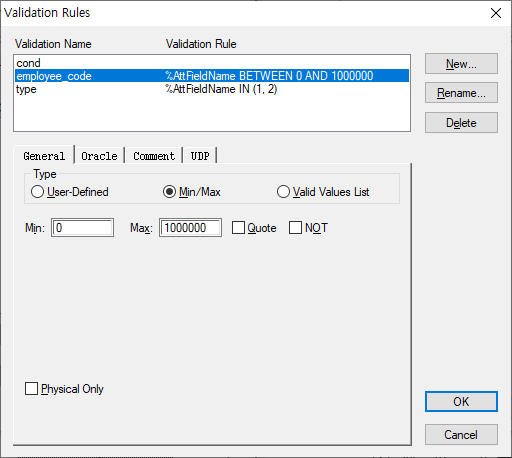
Book type은 integer값으로 1이면 단행본을, 2이면 정기간행을 표현하도록 했다.

-Lend condition



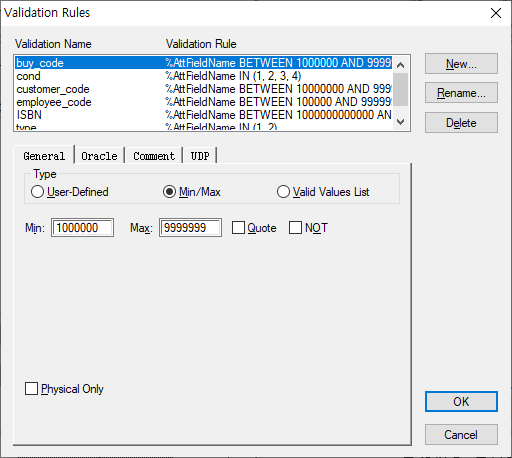
Lend condition 역시 integer값으로 1이면 대출가능, 2이면 대출 중, 3이면 분실, 4면 예약을 표현하도록 했다.

-employ code



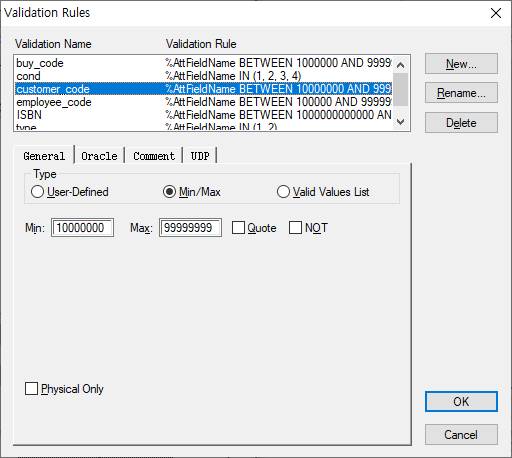
직원 고유 번호는 6자리이므로 설정했다.

-Buy code



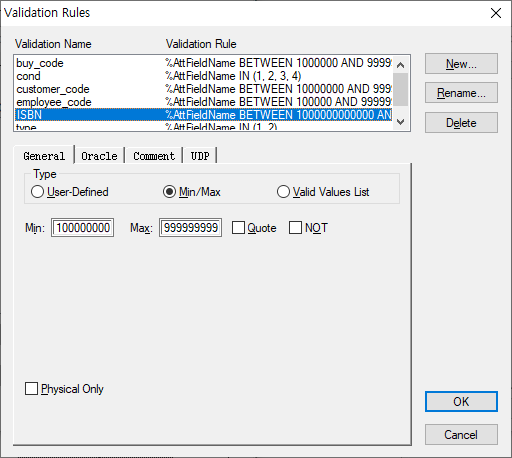
도서 구매 시 도서에 부여되는 고유번호로 7자리로 설정했다.

-customer code



회원 고유 번호는 8자리이므로 설정했다.

-ISBN



ISBN은 13자리이므로 설정했다.

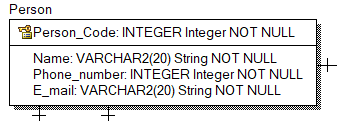
또 다른 제약사항은 원서명, 반납일, 반납예정일, 대출일 4가지가 이고, NULL값이 가능하다.

1. 제약사항 및 무결성

무결성은 부모entity에서 자식entity로의 relation에서 FK, 즉 자식entity가 부모entity의 PK를 FK로 받을 때 PK의 속성을 그대로 유지해야 한다는 의미이다. PK의 속성에서 벗어난 값들을 입력하면 에러가 발생한다. 이는 Erwin 모델링을 할 때도 알 수 있는데 PK의 도메인을 설정해 주면 자식entity의 FK에서도 도메인이 자동적으로 설정되는 것을 확인할 수 있다. 따라서 무결성은 Primary key에 대해서만 고려해주면 된다.

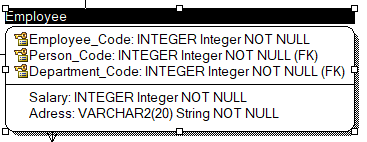
현재 Person code, Employ Code, Customer Code, Department Code, Publisher Code, Book Code, Buy Code가 primary key로 사용되고 있으며 이를 외래키로 받는 여타 entity에 대해서 주어진 범위 내의 값들을 입력하도록 해야 한다.

1. Person



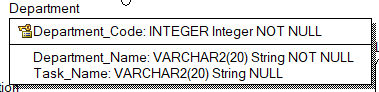
우선 사람 entity를 보면 person code를 PK로 가지며 요구사항에서 제한사항이 없는 임의로 만든 고유번호이기에 integer값으로 고유번호를 부여하고 사람이 모두 가져야할 값이므로 NOT NULL로 설정해주었다. 이름, 전화번호, 이메일도 제약사항이 없으므로 각각 string, integer, string으로 도메인을 설정했고 한 사람은 이름, 전화번호, 이메일을 각각 무조건 가져야 한다고 생각하여 NOT NULL로 설정했다.

1. Employee



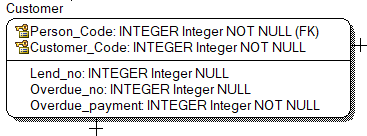
직원이 갖는 FK들은 부모entity에서 도메인을 설정해주기 때문에 생략한다. 위에 설명처럼 도메인도 상속받는다. Employ code는 직원 고유 번호로 6자리 숫자라는 제약사항이 있기에 integer로 도메인을 설정하고 constraint를 부여했다. Salary는 연봉으로 제약 없는 integer로 설정했고, Address는 주소로 string으로 설정했다. 모두 NOT NULL로 설정했다.

1. Department



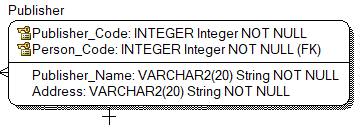
부서는 임의로 설정해 놓은 attribute들이기에 도메인도 integer와 string으로 부여하는 거 외에는 건드리지 않았다. 모두 무조건 값을 가져야 하기에 NOT NULL로 설정했다.

1. Customer



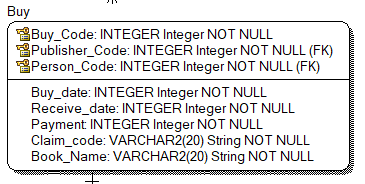
Customer code는 customer code constraint를 갖는 integer로 설정했다. 나머지 대출횟수, 연체 횟수, 연체 금액은 integer로 설정했고 여기서 대출은 하지 않아 대출 횟수와 연체 횟수가 없는 사람이 있을 수 있으므로 NULL이 가능하도록 했고 payment는 없는 경우 0으로 표시하기에 NOT NULL로 했다.

1. Publisher



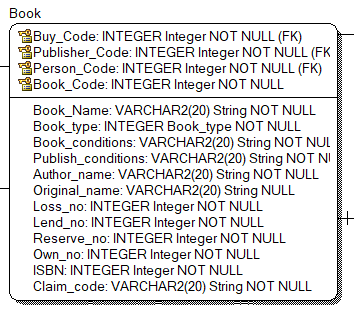
Publisher code 또한 아무런 제약 없는 임의로 설정한 값이기에 integer외에 다른 제약은 주지 않았다. 출판사 이름과 주소는 string으로 부여했고 무조건 값을 가져야 하기에 NOT NULL로 설정했다.

1. Buy



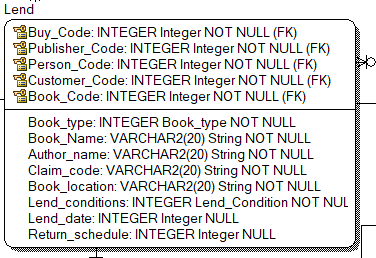
책 구매 시 고유번호 buy code는 integer와 buy code constraint를 통해 7자리 정수로 부여했고, 나머지 구매일, 반입일, 금액은 integer 외에 제약이 없다. 또한 구매 시 값을 무조건 갖으므로 NOT NULL로 부여했다. 청구번호와 책 이름은 string으로 하고 이것 역시 책 구매 시 값이 정해지므로 NOT NULL로 설정했다.

1. Book



Book code는 제약사항이 없기에 integer값을 가진다는 것 외에 제약을 주지 않았다. attribute에서 ISBN에만 제약이 들어있는데 이는 정수 값을 가지면서 13자리이므로 ISBN constraint를 설정하여 제약을 걸어주었다. 나머지 attribute들은 정수 값은 integer, 문자나 숫자의 배열은 string으로 하여 설정했고 original name을 제외한 나머지 값들은 책마다 무조건 있어야하는 값이므로 NOT NULL로 설정했다. Original name은 NULL값도 가능하므로 NULL로 해주었다.

1. Lend



Lend는 모두 FK이기 때문에 설명을 생략하고 attribute를 확인해보자.

Integer와 string을 부여해주고 대출을 하지 않았을 경우 lend date와 return schedule은 생성되지 않기 때문에 NULL값을 가질 수 있도록 해주었다. 나머지 값들은 대출이 없더라도 책 고유 정보이기 때문에 NOT NULL로 설정해주었다.