2021 Database System

Project #1

이름 : 김주호

학과 : 수학과

학번 : 20121277

내용

[1. ER Diagram 3](#_Toc69136521)

[1. Logical Mode 3](#_Toc69136522)

[2. Phycical Mode 4](#_Toc69136523)

[2. Logical / Physical Mode 4](#_Toc69136524)

[1. Entities and Attribute 4](#_Toc69136525)

[1. EMPLOYEE 4](#_Toc69136526)

[2. MEMBER 6](#_Toc69136527)

[3. PUBLISHER 7](#_Toc69136528)

[4. BOOK 8](#_Toc69136529)

[5. BOOK\_DETAILS 10](#_Toc69136530)

[6. STATISTICS 11](#_Toc69136531)

[7. CHECKOUT 12](#_Toc69136532)

[2. Relation and Mapping cardinality 13](#_Toc69136533)

[1. MEMBER and CHECKOUT 13](#_Toc69136534)

[2. CHECKOUT and BOOK 14](#_Toc69136535)

[3. BOOK and PUBLISHER 14](#_Toc69136536)

[4. BOOK and BOOK\_DETAILS 14](#_Toc69136537)

[5. BOOK\_DETAILS and STATISTICS 14](#_Toc69136538)

[3. Constraints and Integrity 14](#_Toc69136539)

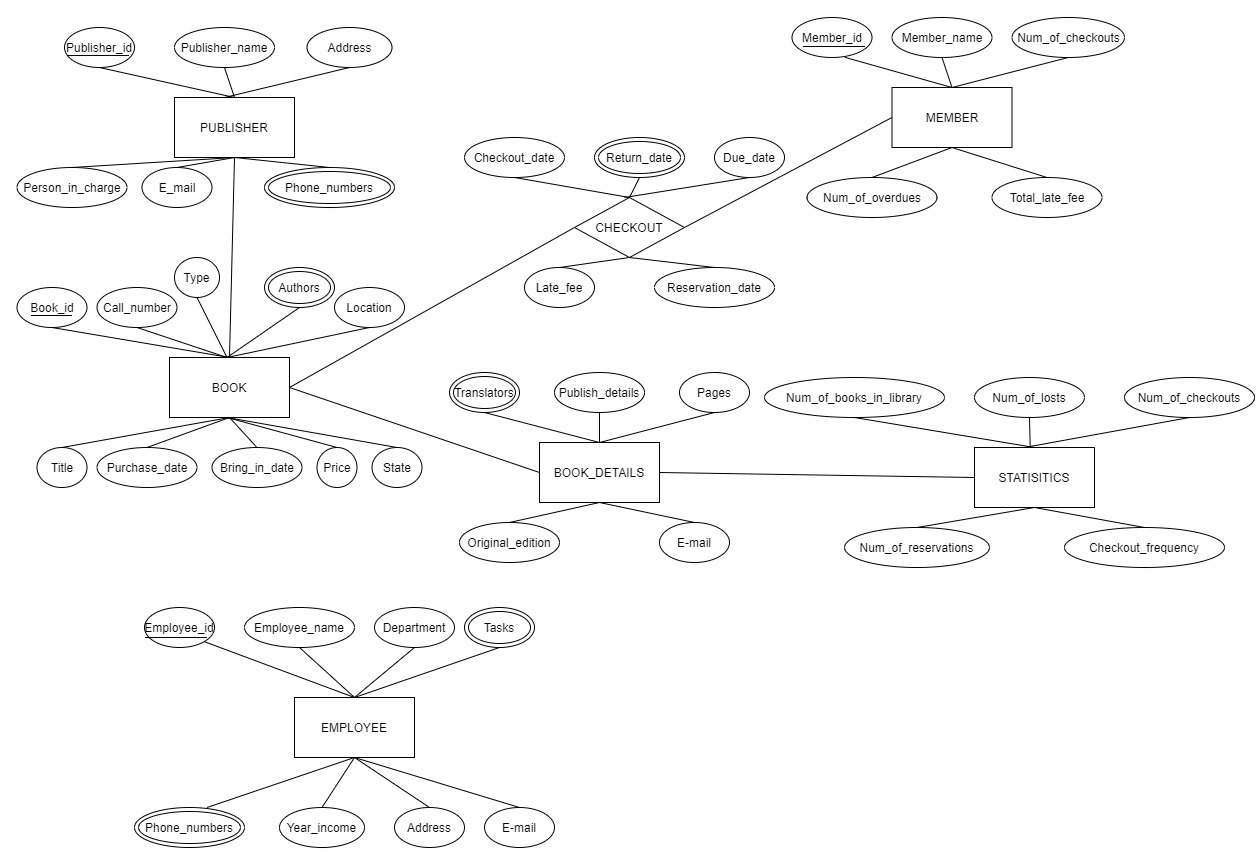
[1. Entity integrity 14](#_Toc69136540)

[2. Referential integrity 14](#_Toc69136541)

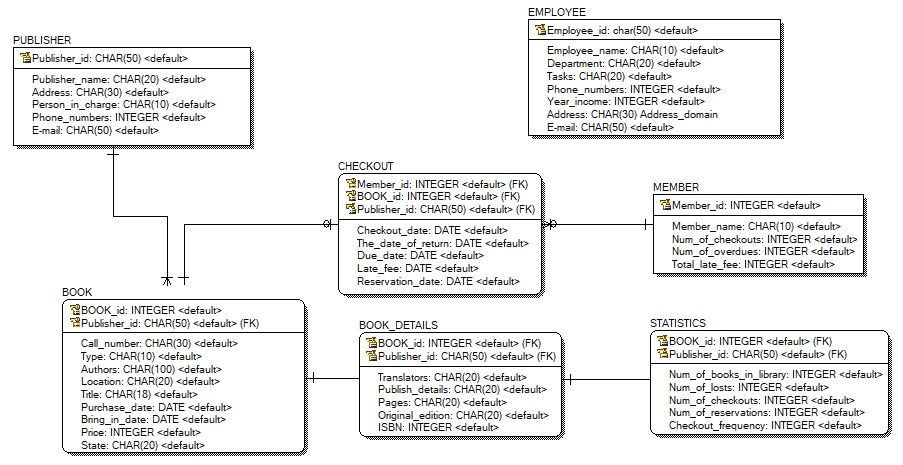
[3. Domain integrity 15](#_Toc69136542)

[3. 설계한 DB가 시스템의 요구사항들을 만족하는지에 대한 설명 15](#_Toc69136543)

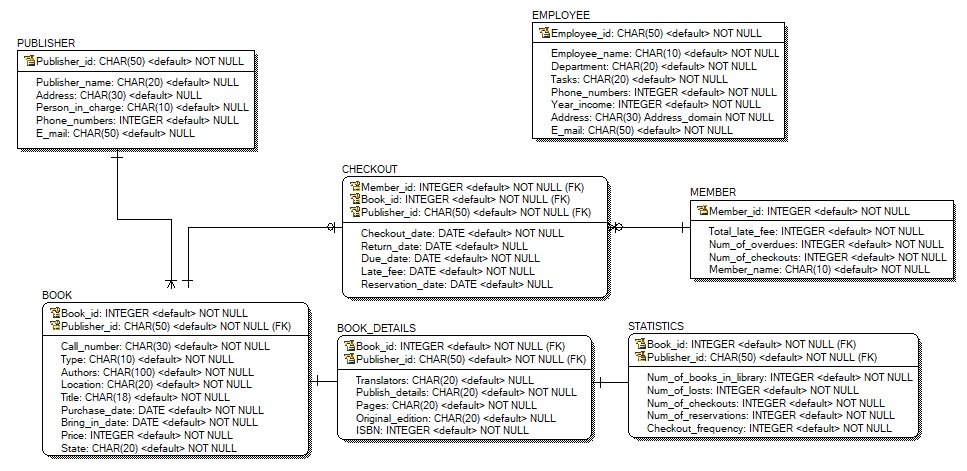
# ER Diagram



## 1. Logical Mode



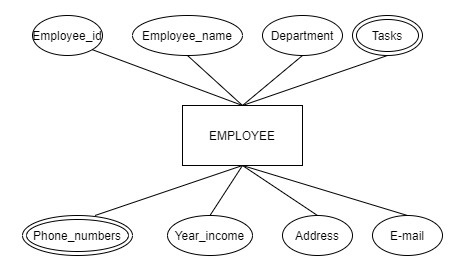
## Phycical Mode



# 2. Logical / Physical Mode

## 1. Entities and Attribute

### 1. EMPLOYEE

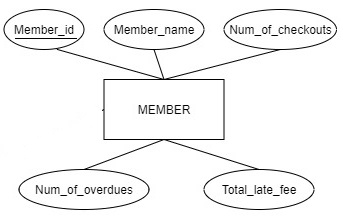


* 학술 정보 기획팀에서 주로 접근하게 될 entity로서, 직원들의 정보를 가지고 있는 entity이다.
* 해당 프로젝트의 경우 특정 '직원'이 특정한 '회원'을 관리 한다거나, 특정한 '직원'이 특정한 '도서'를 관리한다는 개념이 없기 때문에, 도서 관련 entity와 회원 관련 entity와 독립적으로 인적 정보를 저장하는 방향으로 설계하였다.
* 직원 관리를 위해 직원 정보를 조회 및 수정할 수 있는 기능을 제공하는 '직원 관리 시스템'을 설계하는데 활용될 entity이다.

<<Data Type and Constraints of Attribute Value>>

1. Employee\_id : EMPLOYEE의 primary key가 되는 속성 값이다.
   * Data Type : int 형
   * Constraints : 없음
2. Employee\_name : 직원의 이름을 나타내는 attribute이다.
   * Data Type : char형
   * Constraints : 10자 제한
3. Department : 직원이 속한 부서를 나타내는 attribute이다.
   * Data Type : char형
   * Constraints : 20자 제한
4. Tasks : 직원의 담당 업무를 나타내는 attribute이다.
   * Data Type : char형
   * Constraints : 직원 한 명이 여러가지 업무를 가질 수 있다. 해당 attribute를 multivalued attribute로 간주하고, 속성명을 복수형으로 처리하였다. 20자 제한.
5. Phone\_numbers : 직원의 전화번호를 나타내는 attribute이다.
   * Data Type : int형
   * Constraints : 직원 한 명이 전화번호를 여러가지 가질 수 있다. 해당 attribute를 multivalued attribute로 간주하고, 속성명을 복수형으로 처리하였다.
6. Year\_income : 직원의 연봉을 나타낸다.
   * Data Type : int형
   * Constraints : 없음
7. Address : 직원의 주소를 나타내는 attribute이다.
   * Data Type : char형
   * Constraints : domain을 별도로 정의하였다. 주소는 시, 구, 동의 순서로 기입한다는 validation rule을 주었다. 30자 제한
8. E-mail : 직원의 이메일을 나타내는 attribute이다.
   * Data Type : char형
   * Constraints : 30자 제한

### 2. MEMBER

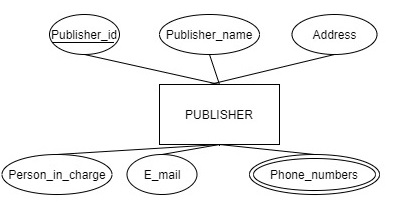


* 회원들의 정보를 가지고 있는 entity로서, 관리 운영팀의 회원 관리 시스템의 근간이 되는 entity이다. 회원 별 대출 내역을 조회할 때 접근하게 될 것이다.

<<Data Type and Constraints of Attribute Value>>

1. Member\_id : MEMBER의 primary key가 되는 속성 값이다.
   * Data Type : int형
   * Constraints : NOT NULL
2. Member\_name : 회원의 이름을 나타내는 attribute이다.
   * Data Type : char형
   * Constraints : 10자 제한, NOT NULL
3. Num\_of\_checkouts : 대출 횟수를 나타내는 attribute이다.
   * Data Type : int형
   * Constraints : 없음, NOT NULL
4. Num\_of\_overdues : 연체 횟수를 나타내는 attribute이다.
   * Data Type : int형
   * Constraints : 없음, NOT NULL
5. Total\_late\_fee : 총 연체 금액을 나타내는 attribute이다.
   * Data Type : int형
   * Constraints : 없음, NOT NULL

### 3. PUBLISHER



* 도서의 출판사 정보를 가지고 있는 entity이다.
* BOOK이 가지고 있는 '속성'이라고도 볼 수 있는데, 출판사 자체로 가지고 있는 정보들이 많아, 이 자체를 하나의 entity로 구성하고, BOOK가 이를 참조하는 방식으로 설계하였다.
* 학술 정보 기획팀이 사용하게 될 '재고 관리 시스템'을 구축하는 데에 해당 entity가 그 설계를 용이하게 해준다.

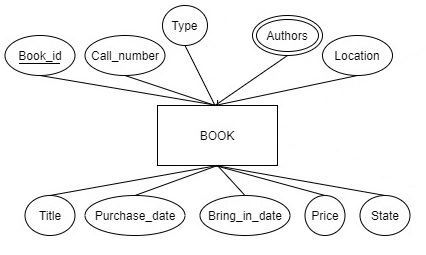
<<Data Type and Constraints of Attribute Value>>

1. Publiser\_id : 출판사의 primary key가 되는 속성이다. 혹여 출판사 이름이 중복되는 경우가 있을 수 있으므로, Publisher\_name을 primary key로 지정할 수 없다. 따라서, 출판사별로 고유의 번호를 주는 방향으로 attribute를 설정하였다.

* Data Type : int형
* Constraints : NOT NULL

1. Publisher\_name : 출판사의 이름을 나타내는 attribute이다.
   * Data Type : char형
   * Constraints : 10자 제한, NOT NULL
2. Address : 출판사의 주소를 나타내는 attribute이다.
   * Data Type : char형
   * Constraints : domain을 별도로 정의하였다. 주소는 시, 구, 동의 순서로 기입한다는 validation rule을 주었다. NOT NULL
3. Person\_in\_charge : 당담자명을 나타내는 attribute이다.
   * Data Type : char형
   * Constraints : 10자 제한, NOT NULL
4. Phone\_numbers : 출판사 전화번호를 나타내는 attribute이다.
   * Data Type : int형
   * Constraints : 출판사의 전화번호가 여러 개일 수 있다. 해당 attribute를 multivalued attribute로 간주하고, 속성명을 복수형으로 처리하였다. NOT NULL
5. E-mail : 출판사의 e-mail을 나태내는 attribute이다.
   * Data Type : char형
   * Constraints : 30자 제한, NOT NULL

### 4. BOOK

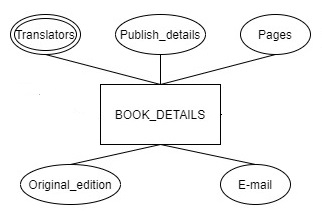


* 도서 정보를 가지고 있는 entity이다.
* 이 entity를 근간으로 도서 정리팀과 회원들이 사용할 도서 정보에 대한 삽입, 삭제, 수정 기능을 수행하는 도서 검색 시스템을 구축할 수 있다. 이 시스템은 도서 정리팀이 구입된 도서를 구입하거나 분류하는 역할을 수월하게 한다.
* PUBLISHER를 참조한다. Publisher\_id를 외래키로 가지고 있다.
* 도서에 관한 attribute가 많은데, 회원들이 검색으로 접근할 수 있는 attribute들을 'basic' 하다고 간주하고, 이 속성을 가지고 있게 하였다. 보다 'detail'한 attribute들은 다른 entity(BOOK\_DETAILS) 를 정의하여 일대일 관계를 가지도록 설계하였다.

<<Data Type and Constraints of Attribute Value>>

1. Book\_id : 도서의 등록 번호를 나타내는 attribute이다. 유일성을 보장하여primary key 로 지정하였다.
   * Data Type : int형
   * Constraints : NOT NULL
2. Call\_number : 도서의 청구 기호를 나타내는 attribute이다.
   * Data Type : char형
   * Constraints : 30자 제한. NOT NULL
3. Type : 자료 유형을 나타내는 attribute이다.
   * Data Type : char형
   * Constraints : 자료 유형은 '단행본' 혹은 '정기간행물' 두 가지 종류의 값만 가질 수 있다. domain을 새로 정의하여, validation rule을 만들고, valid values list로 위의 두가지 속성값을 지정하였다. NOT NULL
4. Authors : 도서의 저자를 나타내는 attribute이다.
   * Data Type : char형
   * Constraints : 책 하나에 여러 명의 저자가 있을 수 있다. 해당 attribute를 multivalued attribute로 간주하고, 속성명을 복수형으로 처리하였다. NOT NULL
5. Location : 도서의 소장처를 나타내는 attribute이다.
   * Data Type : char형
   * Constraints : NOT NULL
6. Title : 도서의 제목을 나타내는 attribute이다.
   * Data Type : char형
   * Constraints : NOT NULL
7. Purchase\_date : 구매일을 나타내는 attribute이다.
   * Data Type : DATE형
   * Constraints : NOT NULL
8. Bring\_in\_date : 반입일을 나타내는 attribute이다.
   * Data Type : DATE형
   * Constraints : NOT NULL
9. Price : 도서의 가격을 나타내는 attribute이다.
   * Data Type : int형
   * Constraints : NOT NULL
10. State : 도서의 상태를 나타내는 attribute이다.
    * Data Type : char형
    * Constraints : 도서의 상태는 '대출중', '분실', '대출가능', '예약' 이렇게 4가지 종류의 값만 올 수 있으므로, domain을 새로 정의하여, 제약 사항을 주었다. NOT NULL

### 5. BOOK\_DETAILS

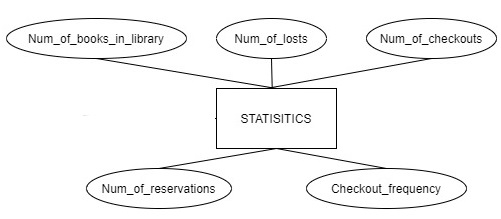


* 도서의 추가 디테일 정보를 가지는 entity이다.
* 회원들의 검색으로 쉽게 접근하는 정보와 구분(완전한 이분법의 구분은 아니다.)하여, 재고 관리 시스템을 구축함에 있어 필요한 도서 상세 정보들을 가지는 entity를 정의하였다.
* BOOK과 1:1 관계를 갖도록 설계하였다.
* Book\_id라는 FK와 Publisher\_id라는 FK를 가지고 있다.
* 학술 정보 기획팀이 보유하고 있는 장서 및 대출 현황을 파악하고, 추가적인 도서 구입 계획 수립을 위해 도서관이 보유하고 있는 전반적인 장서 목록을 파악할 수 있는 '재고 관리 시스템'을 구축하는 데 근간이 된다.

<<Data Type and Constraints of Attribute Value>>

1. Translators : 옮긴이를 나타내는 attribute이다.
   * Data Type : char형
   * Constraints : 도서의 옮긴 이가 없을 수도 있으니 NULL 값을 허용하도록 정의하였다. 해당 attribute를 multivalued attribute로 간주하고, 속성명을 복수형으로 처리하였다. 20자 제한.
2. Publish\_details : 발행 사항을 나타내는 attribute이다.
   * Data Type : char형
   * Constraints : 20자 제한. NOT NULL
3. Pages : 해당 도서의 행태 사항(책의 총 페이지 및 부록 정보)을 나타내는 attribute이다.
   * Data Type : char형
   * Constraints : NOT NULL
4. Original\_edition : 원서명을 나타내는 attribute이다.
   * Data Type : char형
   * Constraints : 도서의 원서가 있지 않은 경우도 있으므로 해당 attribute는 NULL을 허용하였다.
5. ISBN : 국제 표준 도서 번호(13자리 숫자)를 나타내는 attribute이다.
   * Data Type : int형
   * Constraints : 13자 제한. NOT NULL

### 6. STATISTICS

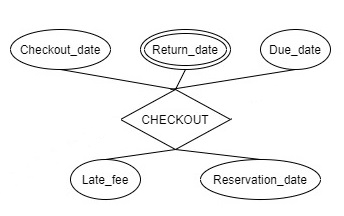


* 도서에 관련한 통계에 대한 정보를 가지는 entity이다.
* 특정한 도서에 대한 도서관 내에서의 이용률 통계에 대한 정보를 기록한다.
* BOOK\_DETAILS와 1:1 관계를 갖도록 설계하였다.
* Publisher\_id라는 외래키와 Book\_id라는 외래키를 가지고 있다.

<<Data Type and Constraints of Attribute Value>>

1. Num*of*books*in*library : 소장 부수를 나타내는 attribute이다.
   * Data Type : int형
   * Constraints : NOT NULL
2. Num\_of\_losts : 분실 건수를 나타내는 attribute이다.
   * Data Type : int형
   * Constraints : NOT NULL
3. Num\_of\_checkouts : 대출 건수를 나타내는 attribute이다.
   * Data Type : int형
   * Constraints : NOT NULL
4. Num\_of\_reservations : 예약 건수를 나타내는 attribute이다.
   * Data Type : int형
   * Constraints : NOT NULL
5. Checkout\_frequency : 대출 빈도를 나타내는 attribute이다.
   * Data Type : int형
   * Constraints : NOT NULL

### 7. CHECKOUT



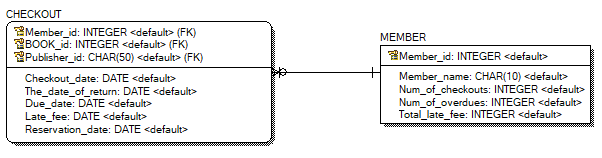
* 대출에 관련한 정보를 가지는 entity이다.
* CHECKOUT은 MEMBER를 참조하고 있고, BOOK과는 1:1 관계를 이루면서, MEMBER와 BOOK이 1:N 관계를 가질 수 있게끔 설계되었다. MEMBER와 BOOK을 CHECKOUT이 매개하는 구조가 되었다.
* Member\_id라는 FK와 Book\_id라는FK, Publisher\_id라는 FK 총 3개의 FK를 가지고 있다.
* 관리 운영팀이 대출 현황을 조회하는 데 필요한 대출 관리 시스템을 구현하는 데 있어 근간이 되는 entity이다.

<<Data Type and Constraints of Attribute Value>>

1. Checkout\_date : 대출일을 나타내는 attribute이다.
   * Data Type : DATE형
   * Constraints : NOT NULL
2. Return\_date : 반납일을 나타내는 attribute이다.
   * Data Type : DATE형
   * Constraints : 미납인 상태에서는 값을 가질 수 없으므로 해당 attribute는 NULL을 허용하였다.
3. Due\_date : 반납 예정일을 나타내는 attribute이다.
   * Data Type : DATE형
   * Constraints : NOT NULL
4. Late\_fee : 연체료를 나타내는 attribute이다.
   * Data Type : int형
   * Constraints : NOT NULL
5. Reservation\_date : 예약날짜를 나타내는 attribute이다.
   * Data Type : DATE형
   * Constraints : 도서에 대한 예약이 없을 수도 있으므로 NULL을 허용하였다.

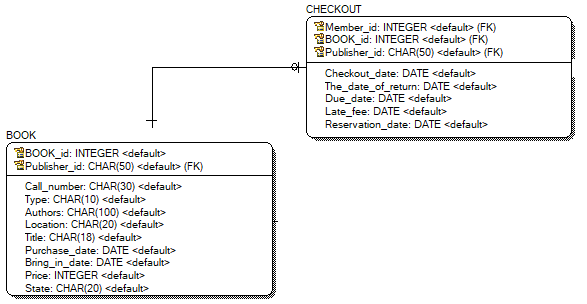
## 2. Relation and Mapping cardinality

### 1. MEMBER and CHECKOUT



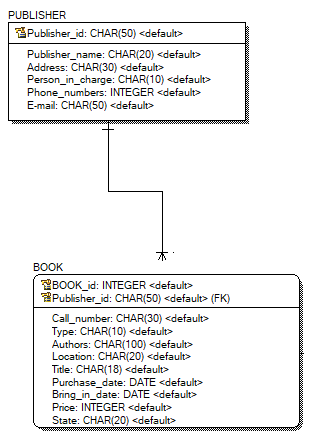
'회원' 한 명이 여러 개수의 '대출'을 발생시킬 수 있다. 하지만, '대출' 하나가 여러 '회원'에 대응할 수는 없다. 즉, MEMBER와 CHECKOUT은 1:N 관계에 있는 one to N relationship이라고 할 수 있다. 또한, 회원이 대출을 전혀 하지 않는 상태일 수 있으므로, MEMBER가 CHECKOUT을 전혀 발생시키지 않을 수도 있다. 이를 반영한 zero, one or more cardinality이다.

### 2. CHECKOUT and BOOK



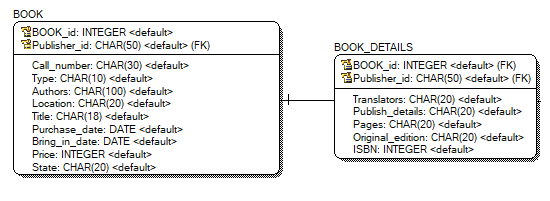
하나의 '대출'은 특정한 하나의 '도서'에 대해서 발생할 수 있다. 마찬가지로 '도서'는 많아야 하나의 '대출'만을 발생시킨다. 즉, CHECKOUT과 BOOK은 one to one relationship이라고 할 수 있다. BOOK은 대출되지 않은 상태로 존재할 수 있으므로, BOOK이 CHECKOUT을 전혀 발생시키지 않을 수도 있다. 이를 반영한 zero or one cardinality이다.

### 3. BOOK and PUBLISHER



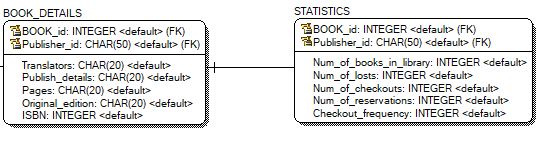
하나의 '도서'는 하나의 '출판사'에서 발행된다. 그러나 하나의 '출판사'는 여러 권의 '도서'를 발행할 수 있다. 따라서, PUBLISHER와 BOOK은 one to N relationship이라고 할 수 있다. 출판사 없는 도서는 허용되지 않도록 설계하였다. one or more cardinality이다.

### 4. BOOK and BOOK\_DETAILS



BOOK\_DETAILS는 BOOK의 세부사항이기 때문에 이 둘은 반드시 1:1 관계를 가져야한다. BOOK\_DETAILS와 BOOK은 one to one relationship이다. BOOK\_DETAILS 의 속성들은 optional하지 않고 도서가 반드시 가져야하는 정보들이며, exactly one cardinality 이다.

### 5. BOOK\_DETAILS and STATISTICS



BOOK\_DETAILS는 하나의 'STATISTICS'을 가져야하고, 반대의 경우도 마찬가지이다. BOOK\_DETAIL와 'STATISTICS'는 one to one relationship이다. 같은 이유로 exactly one cardinality이다.

## 3. Constraints and Integrity

### 1. Entity integrity

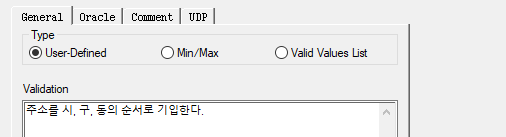
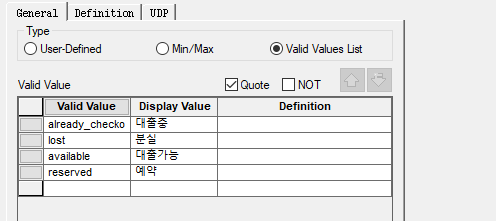
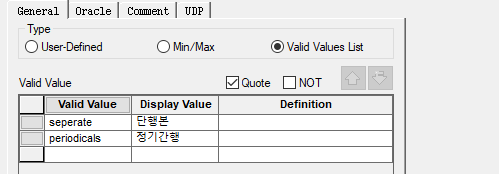
모든 테이블이 기본 키(primary key)로 선택된 필드(column)를 가져야 한다. 기본 키로 선택된 필드는 고유한 값을 가져야 하며, 빈 값은 허용되지 않는다. 모든 entity가 유일한 primary key 값을 가질 수 있도록 설계하였다.

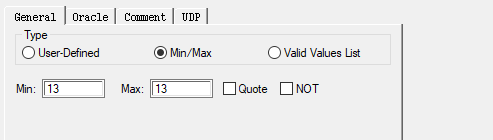
### 2. Referential integrity

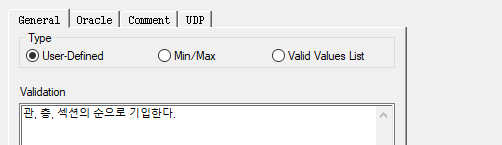
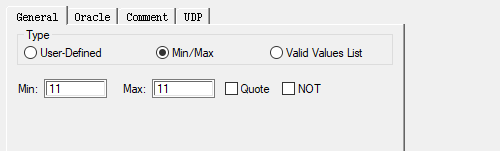
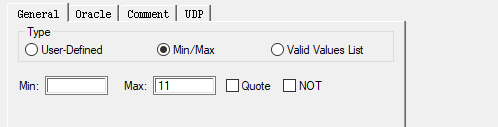
기본 키로 정의된 Publisher\_id, Book\_id, Member\_id, Employ\_id를 외부에서 외래키로 사용할 때, 반드시 그 안에서의 값들만 입력해야 한다. 다른 값들을 입력하려는 시도를 막는 코드를 작성하면 된다.

### 3. Domain integrity

특정 attribute의 값이 그 속성이 정의된 domain에 속한 값이어야 한다는 규정이다. 속성의 성격에 맞게 값이 할당될 수 있도록, 5가지 domain을 정의하고, 정의한 각각의 domain에 validation rule을 정의하였다.

* Address domain
* 
* Address validation rule을 적용하였으며, 주소를 시, 구, 동의 순서로 기입하도록 지령하였다. 직원의 주소, 출판사의 주소가 해당 domain을 갖는다.
* Book state domain
* 
* BOOK entity의 state 속성을 위한 domain 정의이다. 해당 속성은 '대출 중', '분실', '대출 가능', '예약' 단 4가지의 값만을 가질 수 있으므로, 이에 대한 제약 조건을 갖도록 Book state validaion rule을 적용한 domain을 정의하였다.
* Book type domain
* 
* BOOK entity의 type 속성을 위한 domain이다. 해당 속성은 '단행본', '정기간행물' 2단 2가지의 값만을 가질 수 있으므로, 이에 대한 제약조건을 갖도록 Book type validatoin rule을 적용한 domain을 정의하였다.
* ISBN domain



* ISBN은 국제 표준 도서 번호로 13자리 숫자로 표현되므로, 12자리가 입력되거나 14자리가 입력되어서는 안된다. 이에 대한 제약조건을 갖도록 ISBN validation rule을 적용한 domain을 정의하였다.
* Location domain
* 
* BOOK 개체의 location 속성을 위한 domain이다. 도서의 위치가 정의되는 방식은 그 규칙(x관 x층 [section])이 정해져 있는데, 그 규칙을 따라 값을 할당할 수 있도록, location validation rule을 적용한 domain을 정의하였다.
* Phone number domain, Phone number domain2
* 
* 
* 직원들의 전화번호가 가져야하는 domain과 출판사의 전화번호가 가져야하는 domain이 다르도록 설계하였다. 직원들의 전화번호는 개인 휴대 전화를 기록하여, 업무에 활용해야 하는 반면, 출판사의 번호는 회사 대표번호 혹은 휴대 전화일 수 있어서 서로 다른 validation rule을 적용하였다. 개인 휴대전화의 경우 정확히 11자리의 숫자를 입력하도록 제약을 걸었고, 출판사의 번호는 11자리 이하의 숫자가 입력되도록 제약을 걸었다.

# 3. 설계한 DB가 시스템의 요구사항들을 만족하는지에 대한 설명

도서 정리팀에서는 도서 정보에 대한 삽입, 삭제, 수정 기능을 필요로 한다. 회원들을 위한 검색 시스템에서도 도서 정보에 접근해야 하며, 학술 정보 기획팀에서도 도서 구입 계획을 위해 도서 정보에 접근해야 한다. 그러나 학술 정보 정보 기획팀에서 도서 관리를 위해 필요로 하는 도서 정보는 옮긴이나 발행사항, 행태 사항 같은 보다detail한 정보일 수 있다. 따라서, 도서 정보들을 단 하나의 entity로 구성하여 많은 attribute를 넣은 big size로 만들기 보다, 검색 시스템에서 주로 조회하게 될 정보들과 관리침에서 조회하게 될 정보들을 서로 다른 entity로 분리하여 구성한다면, 가독성 측면에서 혹은 데이터들의 관리적인 측면에서 더욱 효율적이다.

통계 table을 따로 구성한 것도 시스템의 요구사항에 부응하는 방향이다. 만약에 사용자가 특정 도서의 소장 부수를 원한다고 하였을 때, table의 모든row를 탐색하면서, 해당 도서가 몇 권이 있는지를 count하는 것보다, STATISTICS entity에 접근하여 값을 한번에 가져오는 것이 훨씬 빠를 것이다. 조회 및 삽입, 삭제가 빈번하게 일어나는 시스템의 요구사항 하에서 속도 적인 측면에서 큰 효율을 얻을 수 있다.

더불어 도서 정보들과 분리 되어 직원들을 위한 Employee entity가 별도로 있기 때문에, 직원 관리를 위해 직원 정보를 조회 및 수정하는 직원 관리 시스템도 빠르게 동작할 수 있을 것이다.