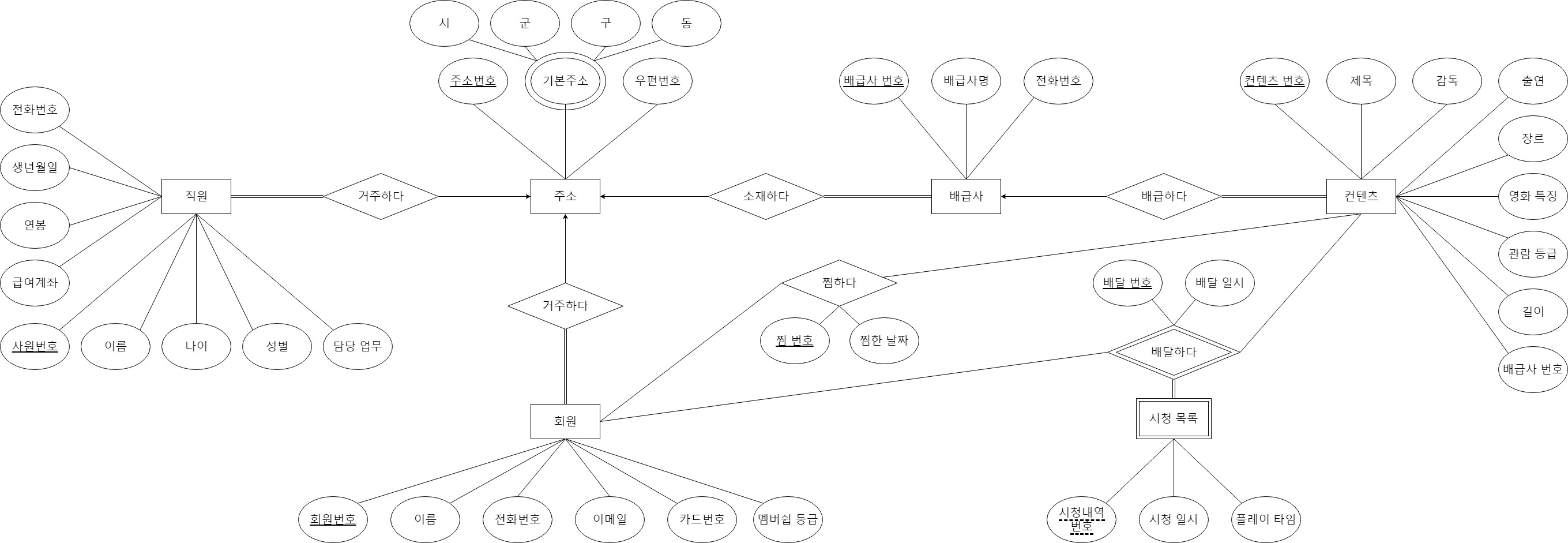
**2019학년도 1학기 데이터베이스 Project 1 보고서**

**담당교수: 서강대학교 컴퓨터공학과 박석**

**1. ER 다이어그램**

이번 프로젝트에서 주어진 모든 요구사항을 효율적으로 수행할 수 있는 데이터베이스를 설계하기 위해 [그림 1]과 같이 ER 다이어그램을 작성했다. 전반적인 ER 다이어그램의 설명은 다음과 같다.

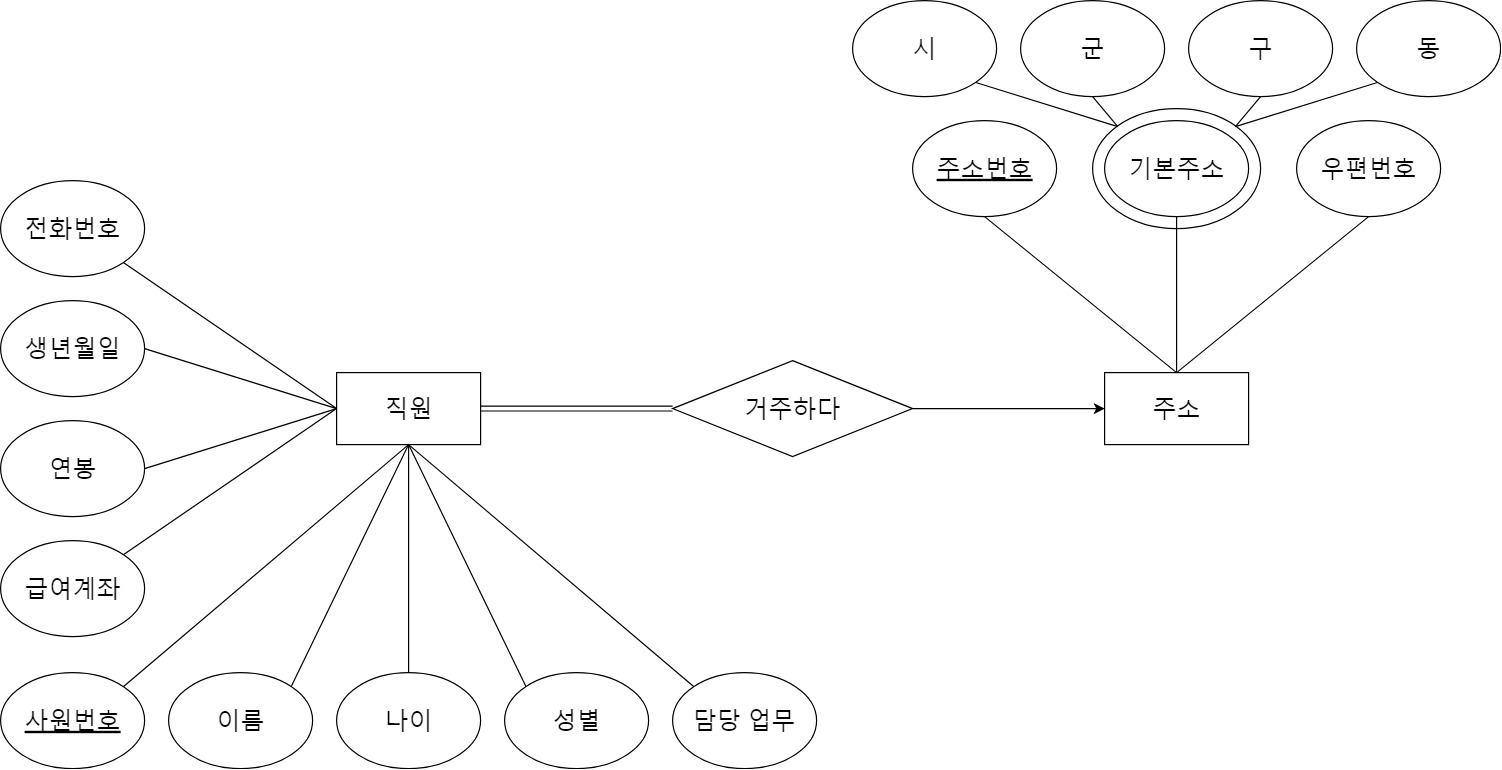
[그림 1]



|  |  |
| --- | --- |
| 1) | 직원 정보, 회원 정보, 배급사 정보, 컨텐츠 정보, 시청 내역 정보를 데이터베이스에 저장하고 관리하기 위해 각각 ‘직원’, ‘회원’, ‘배급사’, ‘컨텐츠’, ‘시청 목록’이라는 5개의 entity를 가진다. |
| 2) | 주소 데이터는 직원, 회원, 배급사에서 모두 사용하며, 동일한 주소에 관해서 동일한 우편 번호를 지니므로 데이터의 redundancy를 줄이고자 ‘주소’라는 entity를 독립적으로 둔다. |
| 3) | 프로젝트에서 주어진 요구사항을 모두 만족하고 각각의 entity와의 관계를 설명하기 위해 ‘배급하다’, ‘찜하다’, ‘배달하다’, ‘소재하다’, ‘거주하다’의 relationship이 존재한다. |

**1.1 직원 - 주소**

[그림 1 - 1]

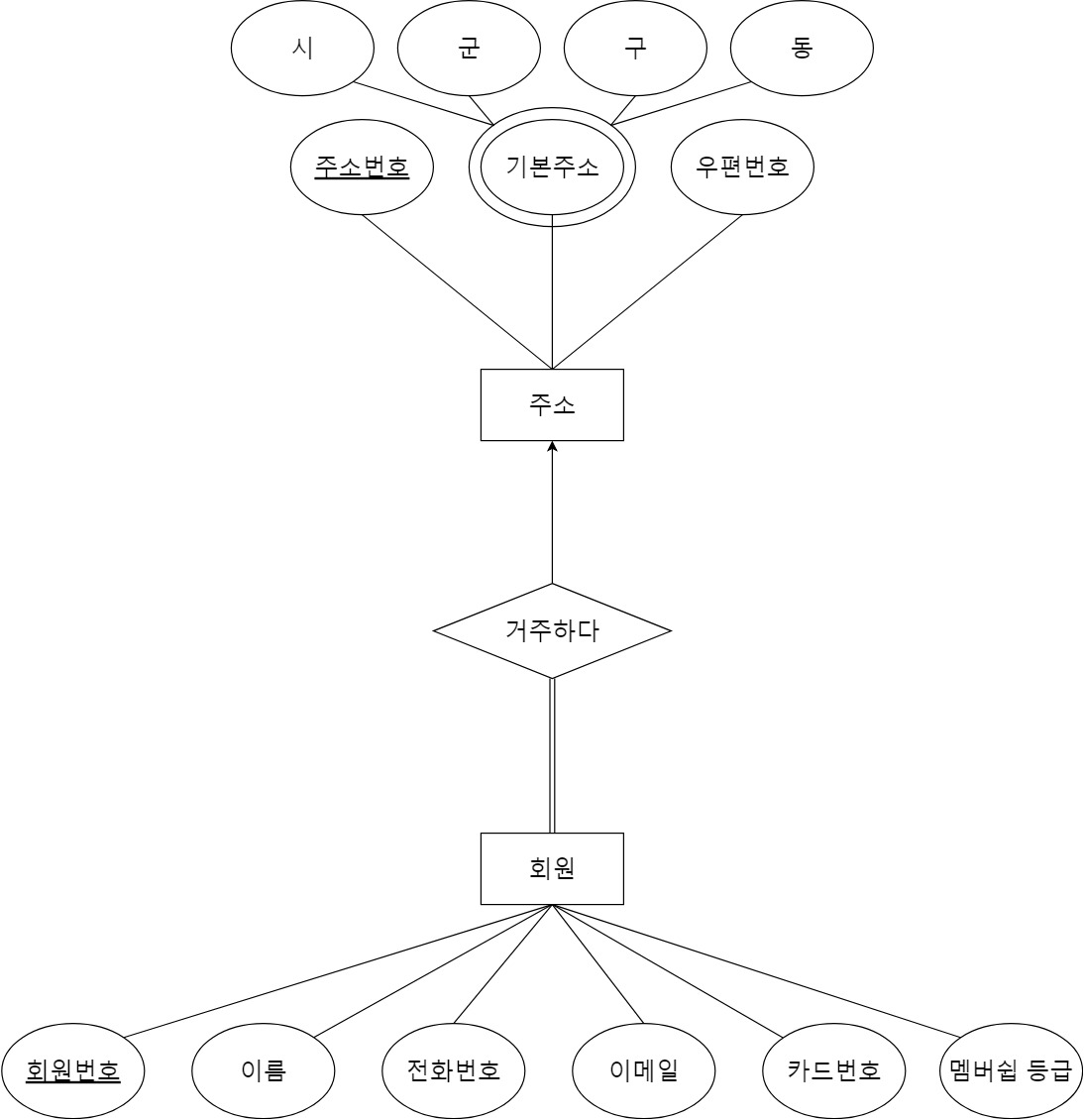


모든 직원의 개인정보를 관리하고 이에 대한 입력 · 삭제 · 수정 · 조회 기능을 수행하기 위해 사원번호, 이름, 나이, 성별, 전화번호, 생년월일, 연봉, 급여계좌, 담당업무를 attribute로 갖는 ‘직원’이라는 entity가 존재한다. 그리고 주소 번호, 기본주소, 우편번호를 attribute로 갖는 ‘주소’라는 entity가 존재한다. 또한 직원과 주소에서 entity 간의 구별이 가능하도록 각각 반영구적인 값을 갖는 사원번호와 주소번호라는 primary key를 가진다.

모든 직원은 거주지에 관한 개인정보를 갖고 있으므로 ‘직원’ entity는 ‘주소’ entity에 관해 total participation이라는 특징을 갖는다. 한 명의 직원은 한 주소에 거주하므로 ‘주소’와 ‘직원’ entity 사이에 ‘거주하다’라는 one-to-many relationship을 지닌다.

**1.2 회원 - 주소**

[그림 1 - 2]

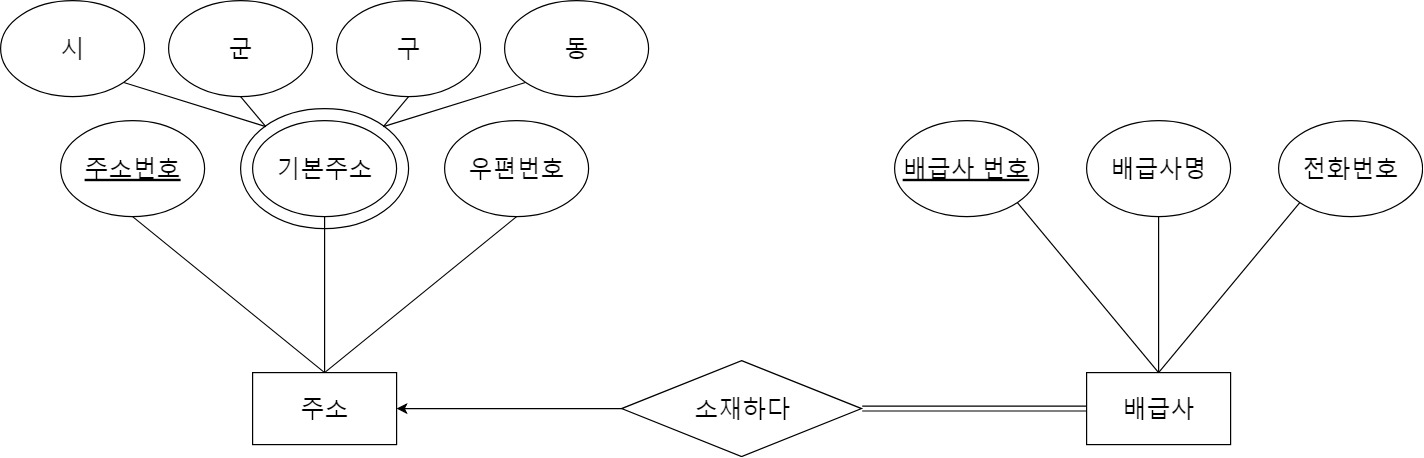


마찬가지로 모든 회원의 개인정보를 관리하고 이에 대한 입력 · 삭제 · 수정 · 조회 기능을 수행하기 위해 회원 번호, 이름, 전화번호, 이메일, 카드번호, 멤버쉽 등급을 attribute로 갖는 ‘회원’이라는 entity가 존재한다. 또한 회원에서 entity 간의 구별이 가능하도록 반영구적인 값을 갖는 회원번호라는 primary key를 가진다.

모든 회원 또한 거주지에 관한 개인정보를 갖고 있으므로 ‘회원’ entity는 ‘주소’ entity에 관해 total participation이라는 특징을 갖는다. 또한 한 명의 회원은 한 주소에 거주하므로 ‘주소’와 ‘회원’ entity 사이에 ‘거주하다’라는 one-to-many relationship을 지닌다.

**1.3 배급사 - 주소**

[그림 1 - 3]

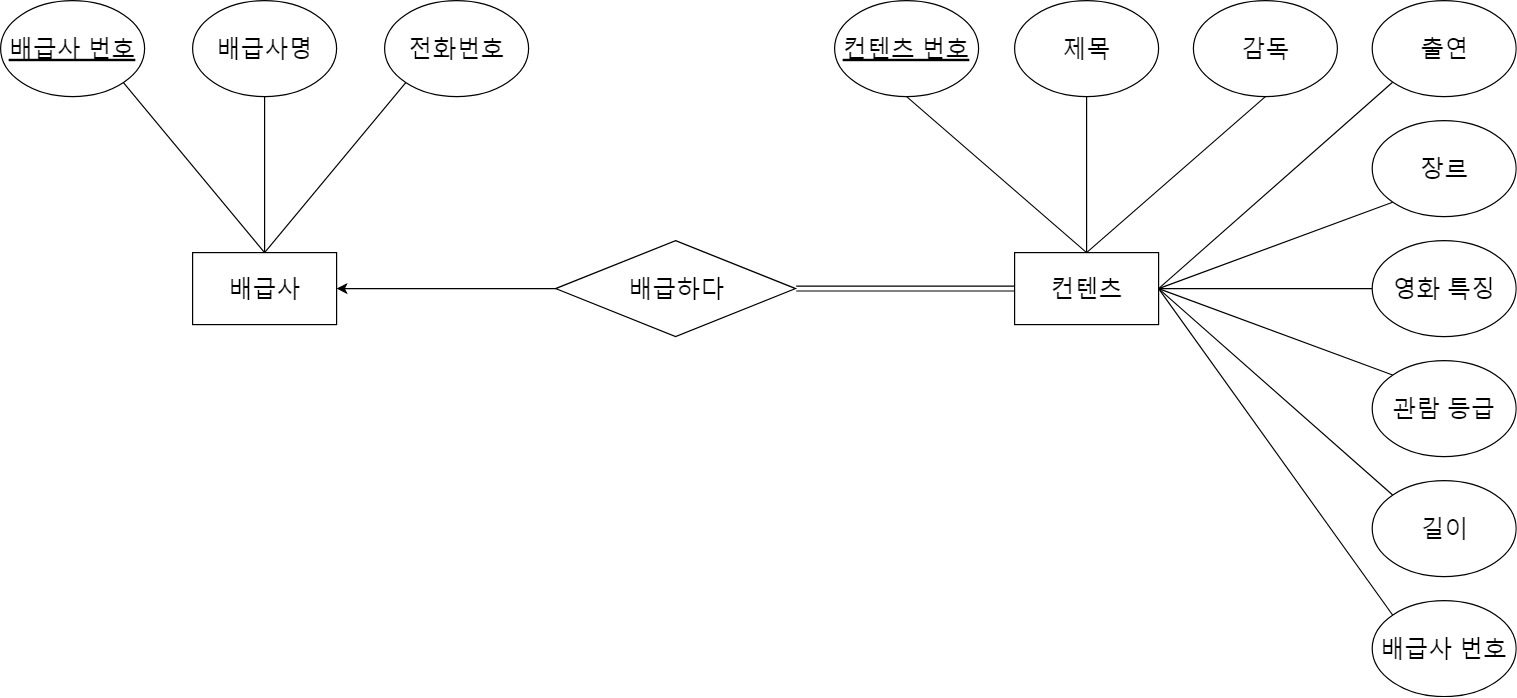


배급사의 정보를 관리하고 이에 대한 입력 · 삭제 · 수정 · 조회 기능을 수행하기 위해 배급사 번호, 배급사명, 전화번호를 attribute로 갖는 ‘배급사’라는 entity가 존재한다. 또한 배급사에서 entity간의 구별이 가능하도록 반영구적인 값을 갖는 배급사 번호라는 primary key를 가진다.

모든 배급사는 소재지에 관한 정보를 갖고 있으므로 ‘배급사’ entitiy는 ‘주소’ entity에 관해 total participation이라는 특징을 갖는다. 또한 한 명의 배급사는 대표적인 한 주소에 소재하므로 ‘주소’와 ‘배급사’ entity 사이에 ‘소재하다’라는 one-to-many relationship을 지닌다.

**1.4 배급사 - 컨텐츠**

[그림 1 - 4]



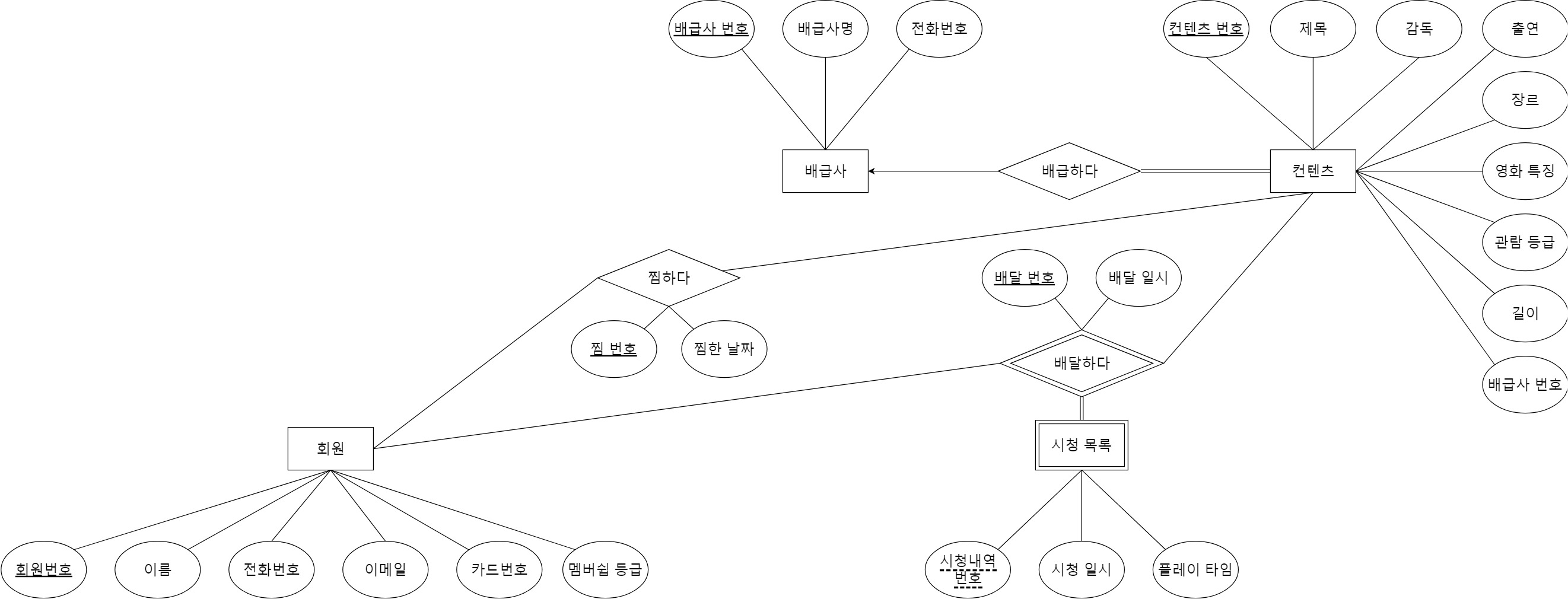
컨텐츠의 정보를 관리하고 이에 대한 입력 · 삭제 · 수정 · 조회 기능을 수행하기 위해 제목, 감독, 출연, 장르, 영화 특징, 관람 등급, 길이, 배급사 번호를 attribute로 갖는 ‘컨텐츠’라는 entity가 존재한다. 또한 ‘컨텐츠’에서 entity 간의 구별을 위해 반영구적인 값을 갖는 컨텐츠 번호라는 primary key를 가진다.

배급사 정보에서 컨텐츠 리스트에 관한 데이터를 저장하기 위해 ‘배급사가 컨텐츠를 배급한다’는 현실 세계의 의미를 반영하여 ‘배급사’와 ‘컨텐츠’ 간의 ‘배급하다’라는 relationship이 존재한다.

모든 컨텐츠는 배급사에 의해 배급되어야 하므로 ‘컨텐츠’ entity는 ‘배급사’ entity에 관해 total participation이라는 특징을 갖는다. 한 컨텐츠는 대표적인 배급사에 의해 배급될 수 있고, 한 배급사는 여러 콘텐츠를 배급할 수 있으므로 ‘배급사’와 ‘컨텐츠’ entity 사이에 one-to-many relationship을 지닌다.

**1.5 회원 - 컨텐츠**

[그림 1 - 5]



회원 정보 중에서 ‘내가 찜한 컨텐츠’에 관한 데이터를 관리하기 위해 ‘회원이 컨텐츠를 찜한다’는 현실 세계의 의미를 반영하여 회원과 컨텐츠 간의 ‘찜하다’라는 relationship이 존재한다. 이 relationship entity는 찜 번호와 찜 날짜라는 attribute를 가지며, 반영구적인 값을 갖는 ‘찜 번호’가 primary key이다.

한 컨텐츠는 여러 회원에 의해 찜이 될 수 있고, 또한 한 회원은 여러 컨텐츠를 찜할 수 있으므로 ‘회원’과 ‘컨텐츠’ entity 사이에 many-to-many relationship이 성립한다.

컨텐츠 배달 내역을 관리하기 위해 ‘회원에게 컨텐츠를 배달한다’는 현실 세계의 의미를 반영하여 회원과 컨텐츠 간의 ‘배달하다’라는 relationship이 존재한다. 이는 배달 번호와 배달 일시라는 attribute를 가지며, 반영구적인 값을 갖는 배달 번호를 primary key로 지닌다.

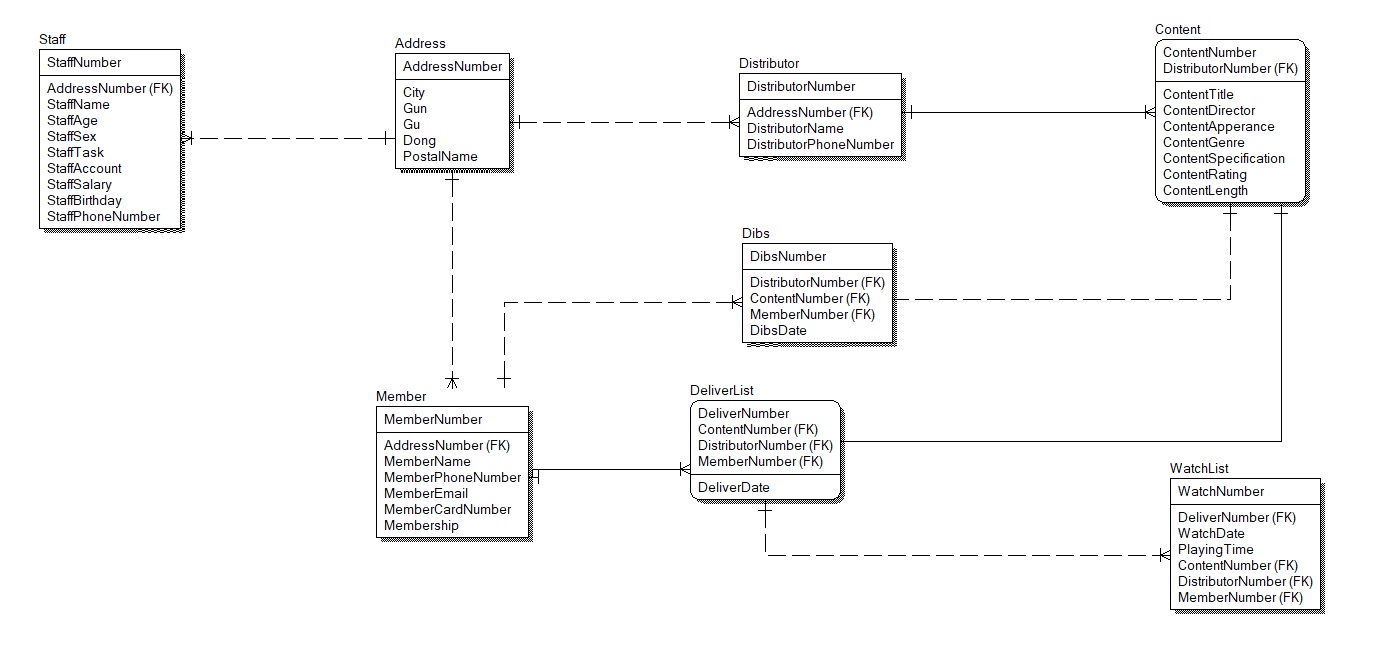
또한 동영상 별 시청 내역을 기록하고 관리하기 위해 시청 내역 데이터를 저장하고 있는 별도의 ‘시청 목록’ entity가 필요한데, 컨텐츠가 회원에게 배달이 되어야 회원이 컨텐츠를 시청할 수 있으므로 ‘배달하다’라는 relationship entity를 ternary relationship으로 만들어서 ‘회원’과 ‘컨텐츠’, 그리고 ‘시청 목록’ entity 사이의 관계로 설정한다. 컨텐츠가 회원에게 배달이 되어야 한다는 선행 조건에 의해 시청 목록이 성립될 수 있으므로 ‘시청 목록’ entity의 존재는 ‘배달하다’의 entity에 의존한다. 따라서 ‘시청 목록’ entity는 weak entity가 되고, ‘배달하다’ entity는 weak relationship이다. ‘시청 목록’ entity는 시청 내역 번호, 시청 일시, 플레이 타임을 attribute로 가지며, 그 중에서 반영구적인 값인 시청 내역이 discriminant이다.

한 회원은 여러 컨텐츠를 배달 받을 수 있고, 한 컨텐츠는 여러 회원에 의해 배달될 수 있으므로 ‘회원’과 ‘컨텐츠’ entity 사이에 many-to-many relationship이 성립한다. 그리고 하나의 배달된 컨텐츠에 관해 한 회원이 여러 번 시청이 가능하고, 또 한 번의 시청 내역은 배달된 한 컨텐츠에 종속되므로 ‘배달하다’와 ‘시청 목록' entity 사이에 one-to-many relationship이 성립된다.

**2. Logical Mode**

Logical Mode에서는 ER 다이어그램에서 정한 entity와 relationship을 바탕으로 실제 table에 대응되는 entity와 attribute를 새로 제작하고 각각의 relationship을 설정하는 작업을 진행했다. [그림 2]는 설계한 Logical Mode의 완성본이며, 전반적인 내용은 다음과 같다.

[그림 2]



|  |  |
| --- | --- |
| 1) | 직원 정보, 회원 정보, 배급사 정보, 컨텐츠 정보, 시청 내역 정보를 데이터베이스에 저장하고 관리하기 위해 각각 ‘Staff’, ‘Member’, ‘Distributor’, ‘Content’, ‘WatchList’이라는 5개의 entity를 가진다. |
| 2) | ‘Staff’, ‘Member’, ‘Distributor’에 중복되는 주소 정보를 따로 저장하여 데이터 간의 중복을 줄이기 위해 ‘Address’라는 entity를 가진다. |
| 3) | ER 다이어그램에서 생성한 ‘거주하다’, ‘소재하다’, 그리고 ‘배급하다’ relationship을 Logical Mode에서는 entity로 제작하지 않고 한 entity에서 foreign key를 이용하여 다른 entity를 참고하도록 했다. |
| 4) | ‘찜하다’와 ‘배달하다’ relationship은 Logical Mode에서 데이터를 효율적으로 따로 관리하기 위해 별개의 entity로 제작했으며, 이는 각각 ‘Dibs’, ‘DeliverList’ entity에 해당된다. |

|  |  |  |  |
| --- | --- | --- | --- |
| **Staff** | | | **직원 정보를 저장하는 entity** |
| **Attribute 이름** | **Attribute 의미** | **Attribute 설명** | |
| **StaffNumber** | 직원 번호 | 직원 개개인의 개별적인 번호이며, 이 번호만 갖고 entity 내의 tuple들을 identifying할 수 있도록 primary key로 지정했다. | |
| **StaffName** | 직원 이름 |  | |
| **StaffAge** | 직원 나이 |  | |
| **StaffSex** | 직원 성별 |  | |
| **StaffTask** | 직원 업무 |  | |
| **StaffAccount** | 직원 계좌 |  | |
| **StaffSalary** | 직원 연봉 |  | |
| **StaffBirthday** | 직원 생일 |  | |
| **StaffPhoneNumber** | 직원 전화번호 |  | |
| **AddressNumber** | 직원 주소 번호 | 직원 정보를 조회할 때 주소에 관한 데이터만을 저장하는 Address라는 entity를 AddressNumber로 참조할 수 있도록 foreign key로 지정했다. 직원 정보에서 주소를 입력하거나 수정할 때는 Address Entity에 데이터를 추가하거나 AddressNumber를 수정하여 변경사항이 적용되도록 한다. | |
| **Member** | | | **회원 정보를 저장하는 entity** |
| **Attribute 이름** | **Attribute 의미** | **Attribute 설명** | |
| **MemberNumber** | 회원 번호 | 회원 개개인의 개별적인 번호이며, 이 번호만 갖고 entity 내의 tuple들을 identifying할 수 있도록 primary key로 지정했다. | |
| **MemberName** | 회원 이름 |  | |
| **MemberPhoneNumber** | 회원 전화번호 |  | |
| **MemberEmail** | 회원 이메일 |  | |
| **MemberCardNumber** | 회원 카드번호 |  | |
| **Membership** | 멤버쉽 |  | |
| **AddressNumber** | 회원 주소 번호 | 회원 정보를 조회할 때 주소에 관한 데이터만을 저장하는 Address라는 entity를 AddressNumber로 참조할 수 있도록 foreign key로 지정했다. 회원 정보에서 주소를 입력하거나 수정할 때는 Address Entity에 데이터를 추가하거나 AddressNumber를 수정하여 변경사항이 적용되도록 한다. | |
| **Distributor** | | | **배급사 정보를 저장하는 entity** |
| **Attribute 이름** | **Attribute 의미** | **Attribute 설명** | |
| **DistributorNumber** | 배급사 번호 | 배급사마다의 개별적인 번호이며, 이 번호만 갖고 entity 내의 tuple들을 identifying할 수 있도록 primary key로 지정했다. | |
| **DistributorName** | 배급사 이름 |  | |
| **DistributorPhoneNumber** | 배급사 전화번호 |  | |
| **AddressNumber** | 배급사 주소 번호 | 배급사 정보를 조회할 때 주소에 관한 데이터만을 저장하는 Address라는 entity를 AddressNumber로 참조할 수 있도록 foreign key로 지정했다. 배급사 정보에서 주소를 입력하거나 수정할 때는 Address Entity에 데이터를 추가하거나 AddressNumber를 수정하여 변경사항이 적용되도록 한다. | |
| **Address** | | | **주소 정보를 저장하는 entity** |
| **Attribute 이름** | **Attribute 의미** | **Attribute 설명** | |
| **AddressNumber** | 주소 번호 | 주소의 개별적인 번호이며, 이 번호만 갖고 entity 내의 tuple들을 identifying할 수 있도록 primary key로 지정했다. Staff, Member, Distributor에서 주소에 관한 정보를 입력하거나 수정할 때 이 entity에 대응되는 AddressNumber가 함께 저장되거나 수정된다. | |
| **City** | 시 이름 |  | |
| **Gun** | 군 이름 |  | |
| **Gu** | 구 이름 |  | |
| **Dong** | 동 이름 |  | |
| **PostalName** | 우편 번호 |  | |
| **Content** | | | **컨텐츠 정보를 저장하는 entity** |
| **Attribute 이름** | **Attribute 의미** | **Attribute 설명** | |
| **ContentNumber** | 컨텐츠 번호 | 컨텐츠의 개별적인 번호이며, 이 번호만 갖고 entity 내의 tuple들을 identifying할 수 있도록 primary key로 지정했다. Dibs, DeliverList, WatchList에서 컨텐츠에 관한 정보를 입력할 때 이 entity에 대응되는 ContentNumber가 함께 저장된다. | |
| **ContentTitle** | 제목 |  | |
| **ContentDirector** | 감독 |  | |
| **ContentApperance** | 출연 |  | |
| **ContentGenre** | 장르 |  | |
| **ContentSpecification** | 영화 특징 |  | |
| **ContentRating** | 관람 등급 |  | |
| **ContentLength** | 길이 |  | |
| **DistributorNumber** | 컨텐츠 배급사 번호 | 컨텐츠 정보를 조회할 때 배급사에 관한 데이터만을 저장하는 Distributor entity를 DistributorNumber로 참조할 수 있도록 foreign key로 지정했다. 컨텐츠 정보에서 배급사를 입력하거나 수정할 때 대응되는 DistributorNumber를 Content에 추가하거나 수정하여 변경사항이 적용되도록 한다. 또한, 하위 entity에서도 DistributorNumber를 사용하여 배급사 정보를 참조할 수 있도록 primary key로 설정했다. | |
| **Dibs** | | | **회원이 찜한 정보를 저장하는 entity** |
| **Attribute 이름** | **Attribute 의미** | **Attribute 설명** | |
| **DibsNumber** | 찜 번호 | 찜한 컨텐츠의 개별적인 번호이며, 이 번호만 갖고 entity 내의 tuple들을 identifying할 수 있도록 primary key로 지정했다. | |
| **DibsDate** | 찜한 날짜 |  | |
| **DistributorNumber** | 찜한 컨텐츠 배급사 번호 | 찜한 컨텐츠의 배급사 정보를 조회할 때 배급사 정보를 출력하기 위해 Distributor entity를 DistributorNumber로 참조할 수 있도록 foreign key로 지정했다. | |
| **ContentNumber** | 찜한 컨텐츠 번호 | 찜한 컨텐츠 정보를 조회할 때 컨텐츠 정보를 출력하기 위해 Content entity를 ContentNumber로 참조할 수 있도록 foreign key로 지정했다. | |
| **MemberNumber** | 찜한 회원 번호 | 찜한 컨텐츠 정보를 조회할 때 찜한 회원에 관한 정보를 출력하기 위해 Member entity를 MemberNumber로 참조할 수 있도록 foreign key로 지정했다. | |
| **DeliverList** | | | **배달 내역 정보를 저장하는 entity** |
| **Attribute 이름** | **Attribute 의미** | **Attribute 설명** | |
| **DeliverNumber** | 배달 내역 번호 | 배달 내역의 개별적인 번호이며, 이 번호만 갖고 entity 내의 tuple들을 identifying할 수 있도록 primary key로 지정했다. WatchList에서 새로운 시청 내역에 관한 정보를 입력하거나 수정할 때 이 entity에 대응되는 DeliverNumber가 참조된다. | |
| **ContentNumber** | 배달 내역 컨텐츠 번호 | 배달 내역을 조회할 때, 해당 컨텐츠가 어떠한 정보를 갖고 있는지 출력하기 위해 Content entity를 ContentNumber로 참조할 수 있도록 foreign key로 지정했다. 또한, 하위 entity인 WatchList에서도 ContentNumber를 사용하여 컨텐츠 정보를 참조할 수 있도록 primary key로 설정했다. | |
| **DistributorNumber** | 배달 내역 컨텐츠의 배급사 번호 | 배달 내역을 조회할 때, 배달된 컨텐츠의 배급사 정보를 출력하기 위해 Distributor entity를 DistributorNumber로 참조할 수 있도록 foreign key로 지정했다. 또한, 하위 entity인 WatchList에서도 DistributorNumber를 사용하여 배급사 정보를 참조할 수 있도록 primary key로 설정했다. | |
| **MemberNumber** | 배달받은 회원 번호 | 배달 내역을 조회할 때, 컨텐츠를 배달받은 회원 정보를 출력하기 위해 Member entity를 MemberNumber로 참조할 수 있도록 foreign key로 지정했다. 또한, 하위 entity인 WatchList에서도 MemberNumber를 사용하여 회원 정보를 참조할 수 있도록 primary key로 설정했다. | |
| **DeliverDate** | 배달 날짜 |  | |
| **WatchList** | | | **시청 내역 정보를 저장하는 entity** |
| **Attribute 이름** | **Attribute 의미** | **Attribute 설명** | |
| **WatchNumber** | 시청 내역 번호 | 시청 내역의 개별적인 번호이며, 이 번호만 갖고 entity 내의 tuple들을 identifying할 수 있도록 primary key로 지정했다. | |
| **ContentNumber** | 시청 내역 컨텐츠 번호 | 시청 내역을 조회할 때, 해당 컨텐츠가 어떠한 정보를 갖고 있는지 출력하기 위해 Content entity를 ContentNumber로 참조할 수 있도록 foreign key로 지정했다. | |
| **DistributorNumber** | 시청 내역 컨텐츠의 배급사 번호 | 시청 내역을 조회할 때, 시청된 컨텐츠의 배급사 정보를 출력하기 위해 Distributor entity를 DistributorNumber로 참조할 수 있도록 foreign key로 지정했다. | |
| **DeliverNumber** | 시청 내역 컨텐츠의 배달 내역 | 대응되는 컨텐츠가 배달된 내역이 있어야 시청 내역 데이터를 새로 입력할 수 있으므로 컨텐츠의 배달 내역이 존재하는지를 저장하기 위해 DeliverList entity를 foreign key로써 참조한다. | |
| **MemberNumber** | 시청한 회원 번호 | 시청 내역을 조회할 때, 컨텐츠를 시청한 회원 정보를 출력하기 위해 Member entity를 MemberNumber로 참조할 수 있도록 foreign key로 지정했다. | |
| **WatchDate** | 시청한 날짜 |  | |
| **PlayingTime** | 플레잉 타임 |  | |

[표 2 - 1]

**2.1 Entity**

**2.2 Relationship**

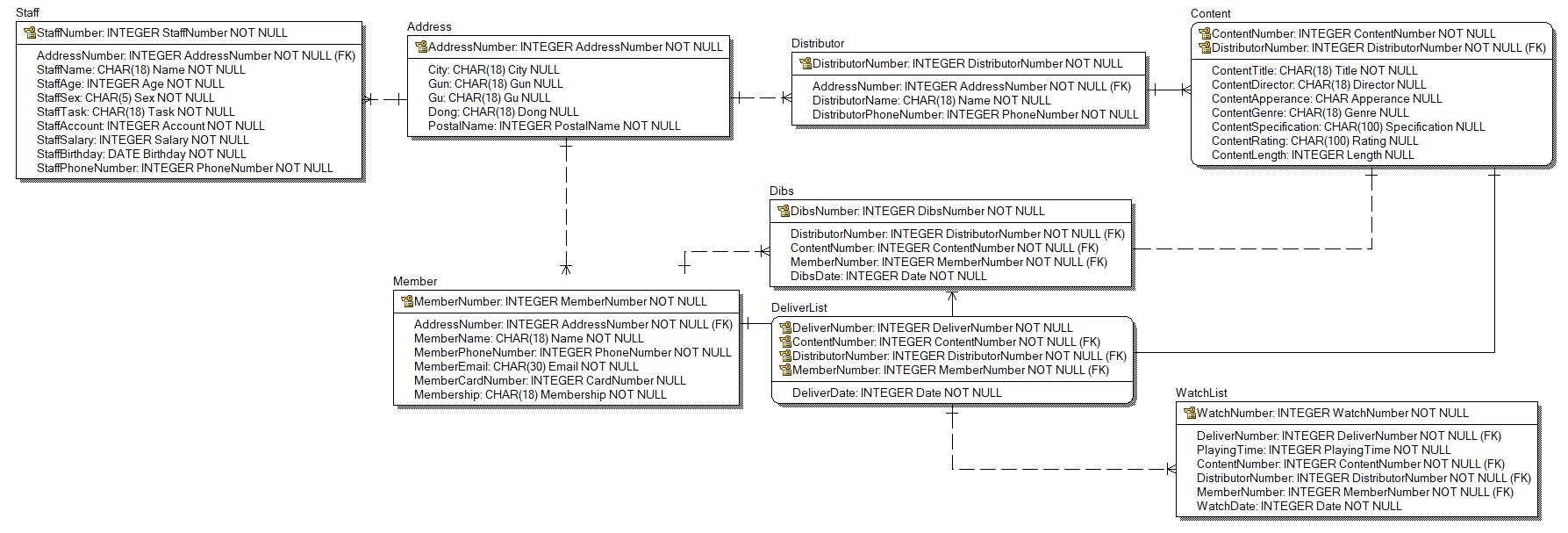
|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Address와 Staff** | | | | | | |
| **Cardinality** | **Relationship Type** | **Nulls** | **RI Actions** | | | |
| One-to-Many | Non-identifying | No Nulls | 자식 삭제 | NONE | 부모 삭제 | RESTRICT |
| 자식 삽입 | RESTRICT | 부모 삽입 | NONE |
| 자식 수정 | RESTRICT | 부모 수정 | RESTRICT |
| **설명** | 한 주소에 여러 직원이 거주할 수 있지만, 한 직원은 오직 하나의 주소에만 거주할 수 있으므로 one-to-many relationship을 갖는다. Staff의 하위 entity가 존재하지 않아 Staff의 AddressNumber가 다른 entity에 의해 참조될 일이 없기 때문에 Address의 PK인 AddressNumber를 Staff에서 PK로는 사용하지 않고 FK로만 사용하는 non-identifying 관계 type으로 설정했으며, 반드시 한 직원은 거주지를 가지고 있어야 한다는 제약 사항을 두기 위해 No Nulls로 정했다. | | | | | |
| **Address와 Member** | | | | | | |
| **Cardinality** | **Relationship Type** | **Nulls** | **RI Actions** | | | |
| One-to-Many | Non-identifying | No Nulls | 자식 삭제 | NONE | 부모 삭제 | RESTRICT |
| 자식 삽입 | RESTRICT | 부모 삽입 | NONE |
| 자식 수정 | RESTRICT | 부모 수정 | RESTRICT |
| **설명** | 한 주소에 여러 회원이 거주할 수 있지만, 한 회원은 오직 하나의 주소에만 거주할 수 있으므로 one-to-many relationship을 갖는다. Member의 하위 entity가 존재하지 않아 Member의 AddressNumber가 다른 entity에 의해 참조될 일이 없기 때문에 Address의 PK인 AddressNumber를 Member에서 PK로는 사용하지 않고 FK로만 사용하는 non-identifying 관계 type으로 설정했다. 또한 반드시 한 회원은 거주지를 가지고 있어야 한다는 제약 사항을 두기 위해 No Nulls로 정했다. | | | | | |
| **Address와 Distributor** | | | | | | |
| **Cardinality** | **Relationship Type** | **Nulls** | **RI Actions** | | | |
| One-to-Many | Non-identifying | No Nulls | 자식 삭제 | NONE | 부모 삭제 | RESTRICT |
| 자식 삽입 | RESTRICT | 부모 삽입 | NONE |
| 자식 수정 | RESTRICT | 부모 수정 | RESTRICT |
| **설명** | 한 주소에 여러 배급사가 소재할 수 있지만, 한 배급사는 오직 대표적인 하나의 주소에만 소재할 수 있으므로 one-to-many relationship을 갖는다. Distributor의 하위 entity는 존재하지만, Member의 AddressNumber가 다른 entity에 의해 참조될 일이 없기 때문에 Address의 PK인 AddressNumber를 Distributor에서 PK로는 사용하지 않고 FK로만 사용하는 non-identifying 관계 type으로 설정했다. 또한 반드시 한 배급사는 대표적인 소재지를 가지고 있어야 한다는 제약 사항을 두기 위해 No Nulls로 정했다. | | | | | |
| **Distributor와 Content** | | | | | | |
| **Cardinality** | **Relationship Type** | **Nulls** | **RI Actions** | | | |
| One-to-Many | Identifying |  | 자식 삭제 | NONE | 부모 삭제 | RESTRICT |
| 자식 삽입 | RESTRICT | 부모 삽입 | NONE |
| 자식 수정 | RESTRICT | 부모 수정 | RESTRICT |
| **설명** | 하나의 배급사는 여러 컨텐츠를 배급할 수 있지만, 한 컨텐츠는 대표적인 하나의 배급사에 의해 반드시 배급되어야 하므로 one-to-many relationship을 갖는다. Content의 하위 entity에 의해 배급사의 정보를 참조하고자 Distributor의 PK인 DistributorNumber가 사용될 수 있어야 하므로 identifying 관계 type을 갖는다. | | | | | |
| **Member와 Dibs** | | | | | | |
| **Cardinality** | **Relationship Type** | **Nulls** | **RI Actions** | | | |
| One-to-Many | Non-identifying | No Nulls | 자식 삭제 | NONE | 부모 삭제 | CASCADE |
| 자식 삽입 | RESTRICT | 부모 삽입 | NONE |
| 자식 수정 | RESTRICT | 부모 수정 | RESTRICT |
| **설명** | 회원은 여러 찜 내역을 가지고 있을 수 있지만, 하나의 찜 내역에 관해서는 한 명의 회원에 의해서만 이루어져야 하므로 one-to-many relationship을 갖는다. Dibs의 하위 entity가 존재하지 않아서 Dibs의 MemberNumber가 다른 entity에 의해 참조될 일이 없기 때문에 Member의 PK인 MemberNumber를 Dibs에서 PK로는 사용하지 않고 FK로만 사용하는 non-identifying 관계 type으로 설정했다. 또한 찜 내역은 반드시 회원에 의해서 선행되어야 존재할 수 있기 때문에 FK인 MemberNumber는 null 값을 가질 수 없으므로 No Nulls로 정했다. 회원 정보가 삭제되면 찜 내역도 같이 삭제되어야 하므로 RI Actions에서 부모의 어떠한 tuple이 삭제될 때 그에 대응되는 자식 tuple도 같이 삭제될 수 있도록 ‘부모 삭제’를 CASCADE로 두었다. | | | | | |
| **Content와 Dibs** | | | | | | |
| **Cardinality** | **Relationship Type** | **Nulls** | **RI Actions** | | | |
| One-to-One | Non-identifying | No Nulls | 자식 삭제 | NONE | 부모 삭제 | CASCADE |
| 자식 삽입 | RESTRICT | 부모 삽입 | NONE |
| 자식 수정 | RESTRICT | 부모 수정 | RESTRICT |
| **설명** | 하나의 컨텐츠에 관해서 여러 개의 찜을 할 수는 없으므로 하나의 찜 내역만이 허락이 된다. 또한 하나의 찜 내역에 관해서도 하나의 컨텐츠만 대응할 수 있기 때문에 둘 사이의 cardinality를 one-to-one relationship으로 정했다. Dibs의 하위 entity가 존재하지 않아서 Dibs의 ContentNumber가 다른 entity에 의해 참조될 일이 없기 때문에 Content의 PK인 ContentNumber를 Dibs에서 PK로는 사용하지 않고 FK로만 사용하는 non-identifying 관계 type으로 설정했다. 또한 찜 내역은 반드시 사전에 컨텐츠가 존재해야 하므로 FK인 MemberNumber는 null 값을 가질 수 없기 때문에 No Nulls로 정했다. 컨텐츠 정보가 삭제되면 찜 내역도 같이 삭제되어야 하므로 RI Actions에서 부모의 어떠한 tuple이 삭제될 때 그에 대응되는 자식 tuple도 같이 삭제될 수 있도록 ‘부모 삭제’를 CASCADE로 두었다. | | | | | |
| **Member와 DeliverList** | | | | | | |
| **Cardinality** | **Relationship Type** | **Nulls** | **RI Actions** | | | |
| One-to-Many | Identifying |  | 자식 삭제 | NONE | 부모 삭제 | RESTRICT |
| 자식 삽입 | RESTRICT | 부모 삽입 | NONE |
| 자식 수정 | RESTRICT | 부모 수정 | RESTRICT |
| **설명** | 한 회원은 여러 배달 내역을 갖고 있을 수 있지만, 하나의 배달 내역에 관해서는 한 명의 회원에게만 대응되어야 하므로 둘 사이의 cardinality를 one-to-many relationship으로 정했다. DeliverList의 하위 entity인 WatchList에서 시청자의 정보를 조회하기 위해 Member의 PK인 MemberNumber를 필요로 하기 때문에 DeliverList에서도 MemberNumber를 FK로 가져야 한다. 따라서 둘 사이를 identifying 관계 type을 갖는다. | | | | | |
| **Content와 DeliverList** | | | | | | |
| **Cardinality** | **Relationship Type** | **Nulls** | **RI Actions** | | | |
| One-to-One | Identifying |  | 자식 삭제 | NONE | 부모 삭제 | RESTRICT |
| 자식 삽입 | RESTRICT | 부모 삽입 | NONE |
| 자식 수정 | RESTRICT | 부모 수정 | RESTRICT |
| **설명** | 하나의 컨텐츠에 관해서 여러 번의 배달이 될 수는 없으므로 하나의 배달 내역만이 허락된다. (한 번 컨텐츠가 배달이 되면 회원은 그 컨텐츠를 다시 배달하지 않아도 영원히 찾아서 볼 수 있다고 가정한다.) 또한 하나의 배달 내역은 한 컨텐츠에만 대응될 수 있으므로 둘 사이의 cardinality를 one-to-one relationship으로 정했다. DeliverList의 하위 entity인 WatchList에서 컨텐츠와 배급사에 관한 정보를 조회하기 위해 Content의 PK인 ContentNumber와 DistributorNumber를 필요로 하므로 DeliverList에서도 ContentNumber와 DistributorNumber를 PK로 가져야 한다. 따라서 둘 사이를 identifying 관계 type을 갖는다. | | | | | |
| **DeliverList와 WatchList** | | | | | | |
| **Cardinality** | **Relationship Type** | **Nulls** | **RI Actions** | | | |
| One-to-Many | Non-identifying | No Nulls | 자식 삭제 | NONE | 부모 삭제 | RESTRICT |
| 자식 삽입 | RESTRICT | 부모 삽입 | NONE |
| 자식 수정 | RESTRICT | 부모 수정 | RESTRICT |
| **설명** | 하나의 배달 내역에 관해서 여러 번의 시청 내역이 존재할 수 있고, 하나의 시청 내역에 관해서는 하나의 배달 내역에만 대응될 수 있으므로 둘 사이의 cardinality를 one-to-many relationship으로 정했다. WatchList의 하위 entity가 존재하지 않아서 attribute가 다른 entity에 의해 참조될 일이 없기 때문에 ContentNumber, DistributorNumber, DeliverNumber, 그리고 MemberNumber를 WatchList에서 PK로는 사용하지 않고 FK로만 사용하는 non-identifying 관계 type으로 설정했다. | | | | | |

[표 2 - 2]

**3. Physical Mode**

Physical Mode에서는 Logical Mode에서 완성한 결과를 바탕으로 attribute들의 domain을 지정하여 어떠한 data type이 각 attribute에 올 수 있는지를 지정한다. [그림 3]은 설계한 Physical Mode의 완성본이다.

[그림 3]



**3.1 Domain Rule과 Attribute**

|  |  |  |  |
| --- | --- | --- | --- |
| **Domain Name** | **Data Type** | **Null 허용** | **설명** |
| **Account** | INTEGER | Not Null | 직원의 계좌번호를 저장하기 위한 StaffAccount attribute의 도메인이며, 계좌번호는 ‘-’가 없으면 숫자의 나열이기 때문에 INTEGER를 data type으로 가진다. 직원의 계좌번호에 관한 데이터는 무조건 갖고 있어야 한다고 가정하여 Not Null로 설정한다. |
| **AddressNumber** | INTEGER | Not Null | Address Entity에서 tuple들을 개별적으로 identifying 할 수 있도록 고유의 반영구적인 번호를 부여한 primary key의 도메인이며, 수로 이루어진 번호이기 때문에 INTEGER를 data type으로 가진다. PK는 빈 상태여서는 안 되기 때문에 Not Null로 설정한다. |
| **Age** | INTEGER | Not Null | 직원의 나이를 저장하기 위한 StaffAge attribute의 도메인이며, 나이는 수로 이루어지기 때문에 INTEGER를 data type으로 가진다. 직원에 관한 정보는 빠짐없이 무조건 갖고 있어야 한다는 것을 전제로 가정하기 때문에 Not Null로 설정한다. |
| **Apperance** | CHAR | Null 가능 | 컨텐츠의 출연자들을 저장하기 위한 ContentApperance attribute의 도메인이며, 출연자들의 이름을 저장해야 하므로 CHAR를 data type으로 가진다. 컨텐츠의 출연자 정보가 반드시 있어야 하지는 않는다고 가정하여 Null도 허용한다. |
| **Birthday** | DATE | Not Null | 직원의 생일을 저장하기 위한 StaffBirthday attribute의 도메인이며, 생일 데이터는 연도, 월, 일의 8자리 수로 된 정수로 다루므로 CHAR를 data type으로 가진다. 직원의 생일에 관한 데이터는 무조건 갖고 있어야 한다고 가정하여 Not Null로 설정한다. |
| **CardNumber** | INTEGER | Null 가능 | 회원의 카드 번호를 저장하기 위한 MemberCardNumber attribute의 도메인이며, 카드 번호는 ‘-’를 제외하고 여러 숫자가 나열되어 하나의 수를 이루기 때문에 INTEGER를 data type으로 가진다. 회원의 카드 번호는 굳이 없어도 된다고 가정하여 Null도 허용한다. |
| **City** | CHAR(18) | Null 가능 | 주소 데이터에서 시에 관한 데이터를 저장하는 City attribute의 도메인이며, 문자열로 이를 저장한다고 가정하여 최대 18자리의 CHAR를 data type으로 가진다. 주소 체계에서는 시가 있을 수도 있지만 시 대신 군이 존재할 수도 있기 때문에 Null을 허용한다. |
| **ContentNumber** | INTEGER | Not Null | Content Entity에서 tuple들을 개별적으로 identifying 할 수 있도록 고유의 반영구적인 번호를 부여한 primary key의 도메인이며, 수로 이루어진 번호이기 때문에 INTEGER를 data type으로 가진다. PK는 빈 상태여서는 안 되기 때문에 Not Null로 설정한다. |
| **Date** | INTEGER | Not Null | 내가 찜한 컨텐츠 데이터를 저장하는 Dibs와 시청 이력을 기록하는 WatchList, 그리고 컨텐츠가 배달된 내역을 기록하는 DeliverList entity에서 각각 찜한 날짜와 시청한 날짜인 DibsDate, WatchDate, DeliverDate attribute의 도메인이다. 컨텐츠를 찜하거나 배달, 또는 시청할 때의 날짜뿐만이 아니라 해당 시점의 시와 분에 관한 데이터까지 기록해야 하므로 DATE가 아니라 연도, 월, 일, 시간, 분의 순으로 된 12자리의 수를 저장해야 한다고 가정하여 INTEGER를 data type으로 갖는다. 찜하거나 시청할 때 무조건 그 때의 시각을 기록한다고 전제하여 Not Null로 설정한다. |
| **DeliverNumber** | INTEGER | Not Null | DeliverList Entity에서 tuple들을 개별적으로 identifying 할 수 있도록 고유의 반영구적인 번호를 부여한 primary key의 도메인이며, 수로 이루어진 번호이기 때문에 INTEGER를 data type으로 가진다. PK는 빈 상태여서는 안 되기 때문에 Not Null로 설정한다. |
| **DibsNumber** | INTEGER | Not Null | Dibs Entity에서 tuple들을 개별적으로 identifying 할 수 있도록 고유의 반영구적인 번호를 부여한 primary key의 도메인이며, 수로 이루어진 번호이기 때문에 INTEGER를 data type으로 가진다. PK는 빈 상태여서는 안 되기 때문에 Not Null로 설정한다. |
| **Director** | CHAR(18) | Null 가능 | 컨텐츠의 제작진 정보를 저장하기 위한 ContentDirector attribute의 도메인이며, 제작진 이름을 저장해야 하므로 최대 18자까지 가능한 CHAR를 data type으로 가진다. 컨텐츠의 제작진 정보가 반드시 있어야 하지는 않는다고 가정하여 Null도 허용한다. |
| **DistributorNumber** | INTEGER | Not Null | Distributor Entity에서 tuple들을 개별적으로 identifying 할 수 있도록 고유의 반영구적인 번호를 부여한 primary key의 도메인이며, 수로 이루어진 번호이기 때문에 INTEGER를 data type으로 가진다. PK는 빈 상태여서는 안 되기 때문에 Not Null로 설정한다. |
| **Dong** | CHAR(18) | Null 가능 | 주소 데이터에서 동에 관한 데이터를 저장하는 Dong attribute의 도메인이며, 문자열로 이를 저장한다고 가정하여 최대 18자리의 CHAR를 data type으로 가진다. 주소 체계에서는 동이 있을 수도 있지만 동 대신 면, 리 등이 존재할 수 있기 때문에 Null도 허용한다. |
| **Email** | CHAR(30) | Not Null | 회원의 이메일을 저장하기 위한 MemberEmail attribute의 도메인이며, 이메일 주소는 문자로 이루어져 있으므로 최대 30자의 CHAR을 data type으로 갖는다. 이메일은 회원이 반드시 입력해야 할 데이터라고 전제하여 Not Null로 설정한다. |
| **Genre** | CHAR(18) | Null 가능 | 컨텐츠의 장르 정보를 저장하기 위한 ContentGenre attribute의 도메인이며, 장르명을 저장해야 하므로 최대 18자까지 가능한 CHAR를 data type으로 가진다. 컨텐츠의 장르가 반드시 있어야 하지는 않는다고 가정하여 Null도 허용한다. |
| **Gu** | CHAR(18) | Null 가능 | 주소 데이터에서 동에 관한 데이터를 저장하는 Gu attribute의 도메인이며, 문자열로 이를 저장한다고 가정하여 최대 18자리의 CHAR를 data type으로 가진다. 주소 체계에서 모든 케이스에 구가 존재하지는 않을 수 있기 때문에 Null도 허용한다. |
| **Gun** | CHAR(18) | Null 가능 | 주소 데이터에서 군에 관한 데이터를 저장하는 Gun attribute의 도메인이며, 문자열로 이를 저장한다고 가정하여 최대 18자리의 CHAR를 data type으로 가진다. 주소 체계에서 군이 존재할 수도 있지만, 군 대신 시로 존재할 수도 있기 때문에 Null도 허용한다. |
| **Length** | INTEGER | Not Null | 컨텐츠의 길이 정보를 저장하기 위한 Length attribute의 도메인이며, 몇 시간 몇 분의 길이를 갖는지를 저장해야 하므로 INTEGER를 data type으로 가진다. 컨텐츠의 길이는 반드시 존재한다고 가정하여 Not Null로 설정한다. |
| **MemberNumber** | INTEGER | Not Null | Member Entity에서 tuple들을 개별적으로 identifying 할 수 있도록 고유의 반영구적인 번호를 부여한 primary key의 도메인이며, 수로 이루어진 번호이기 때문에 INTEGER를 data type으로 가진다. PK는 빈 상태여서는 안 되기 때문에 Not Null로 설정한다. |
| **Membership** | CHAR(18) | Not Null | 회원의 멤버쉽에 관한 데이터를 갖는 Membership attribute의 도메인이며, ‘일반’, ‘우수’ 등 특정한 문자열 이름을 갖는 등급을 저장해야 하므로 최대 18자로 이루어진 CHAR를 data type으로 갖는다. 모든 회원은 처음 가입한 상태일 때 ‘일반’ 멤버쉽을 갖고, 그 이후 승급할 수 있다고 가정하여 Not Null로 설정한다.  회원의 멤버쉽은 최소한 ‘일반(Normal)’ 또는 ‘우수(VIP)’라는 두 가지 선택지밖에 존재하지 않는다고 전제하기 때문에 Validation Constraint Rule로서 Membership\_Validation\_Rule을 지정했다. 입력 또는 저장 가능한 값은 ‘Normal’ 또는 ‘VIP’ 밖에 되지 않도록 한다. |
| **Name** | CHAR(30) | Not Null | 직원, 회원, 배급사의 이름을 데이터로 갖는 StaffName, MemberName, DistributorName attribute의 도메인이며, 문자열로 된 이름을 데이터로 저장해야 하므로 최대 30자로 이루어진 CHAR를 data type으로 갖는다. 모든 직원과 회원, 그리고 배급사는 이름을 반드시 갖고 있어야 한다고 가정하여 Not Null로 설정한다. |
| **PhoneNumber** | INTEGER | Not Null | 직원, 회원의 전화번호, 그리고 배급사의 연락처를 데이터로 갖는 StaffPhoneNumber, MemberPhoneNumber, DistributorPhoneNumber의 도메인이며, 숫자로 이루어진 수열을 하나의 정수 데이터로서 저장해야 하므로 INTEGER를 data type으로 갖는다. 모든 직원과 회원, 그리고 배급사는 전화번호 또는 연락처를 반드시 갖고 있어야 한다고 가정하여 Not Null로 설정한다. |
| **PlayingTime** | INTEGER | Not Null | 회원이 얼마의 시간동안 컨텐츠를 시청했는지에 관한 데이터를 갖는 WatchList의 PlayingTime attribute의 도메인이며, 시간에 관한 정보를 저장해야 하므로 이를 분 단위로 기록한다고 가정하여 INTEGER를 data type으로 갖도록 한다. 시청을 하면 시청 시간은 반드시 기록된다고 가정하여 Not Null로 설정한다. |
| **PostalName** | INTEGER | Not Null | 우편 번호 데이터를 갖는 Address entity의 PostalName attribute의 도메인이며, ‘-'를 제외하고 숫자로 이루어진 수열을 하나의 정수로서 저장한다고 가정하여 INTEGER를 data type으로 갖도록 한다. 우편 번호는 주소 데이터에서 반드시 갖고 있다고 전제하여 Not Null로 설정한다. |
| **Rating** | CHAR(18) | Null 가능 | 컨텐츠의 관람 등급에 관한 데이터를 갖는 ContentRating attribute의 도메인이며, ‘전체 관람가’, ‘12세 이상’과 같이 문자열로 되어 있기 때문에 최대 18자의 CHAR를 data type으로 갖는다. 컨텐츠의 관람 등급은 반드시 갖고 있어야 하지는 않는다고 가정하여 Null도 허용한다. |
| **Salary** | INTEGER | Not Null | 직원의 연봉을 저장하기 위한 StaffSalary attribute의 도메인이며, 연봉이 1000만원일 때 ‘10000000’처럼 정수로 데이터를 저장하기 위해 INTEGER를 data type으로 가진다. 관리자가 직원들의 효율적으로 관리하기 위해 직원의 연봉에 관한 데이터는 무조건 갖고 있어야 한다고 가정하여 Not Null로 설정한다. |
| **Sex** | CHAR(10) | Not Null | 직원의 성별을 데이터로 갖는 StaffSex attribute의 도메인이며, ‘Man' 또는 ’Woman‘을 데이터로 가져야 되므로 최대 5자리의 CHAR를 data type으로 가진다. 직원들의 성별에 관한 정보는 직원의 개인정보 확인을 위해 반드시 필요하다고 가정하여 Not Null로 설정한다.  직원의 성별은 ‘Man’ 또는 ‘Woman’의 두 가지 선택지밖에 존재하지 않기 때문에 Validation Constraint Rule로서 StaffSex\_Validation\_Rule을 지정했다. 입력 또는 저장 가능한 값은 ‘Man’ 또는 ‘Woman’ 밖에 되지 않도록 한다. |
| **Specification** | CHAR(100) | Null 가능 | 컨텐츠의 특징을 저장하기 위한 ContentSpecification attribute의 도메인이며, 컨텐츠 특징을 길게 작성할 수 있도록 최대 100자까지 가능한 CHAR를 data type으로 가진다. 컨텐츠의 특징 데이터가 반드시 있어야 하지는 않는다고 가정하여 Null도 허용한다. |
| **StaffNumber** | INTEGER | Not Null | Staff Entity에서 tuple들을 개별적으로 identifying 할 수 있도록 고유의 반영구적인 번호를 부여한 primary key의 도메인이며, 수로 이루어진 번호이기 때문에 INTEGER를 data type으로 가진다. PK는 빈 상태여서는 안 되기 때문에 Not Null로 설정한다. |
| **Task** | CHAR(18) | Not Null | 직원의 담당 업무가 무엇인지를 저장하는 StaffTask attribute의 도메인이며, ‘재무 담당’, ‘홍보’, ‘홈페이지 관리’처럼 문자열로 이루어진 데이터를 갖도록 최대 18자의 CHAR를 data type으로 설정한다. 또한, 직원의 업무 등 개인정보는 반드시 존재한다고 가정하여 Not Null로 설정한다. |
| **Title** | CHAR(18) | Not Null | 컨텐츠의 제목을 저장하기 위한 ContentSpecification attribute의 도메인이며, 컨텐츠 제목은 문자열로 이루어져 있으므로 최대 100자까지 가능한 CHAR를 data type으로 가진다. 컨텐츠의 제목 데이터는 반드시 있어야 한다고 가정하여 Not Null로 설정한다. |
| **WatchNumber** | INTEGER | Not Null | WatchList Entity에서 tuple들을 개별적으로 identifying 할 수 있도록 고유의 반영구적인 번호를 부여한 primary key의 도메인이며, 수로 이루어진 번호이기 때문에 INTEGER를 data type으로 가진다. PK는 빈 상태여서는 안 되기 때문에 Not Null로 설정한다. |

[표 3 - 1]

**4. 세부 요구사항**

앞에서 세부적인 요구사항을 모두 설명하였으나 이를 재정리하여 명료하게 설명하면 다음과 같다.

|  |  |
| --- | --- |
| 1) | Staff, Member, Distributor entity의 주소와 관련한 데이터는 서로 중복될 수 있으므로 데이터의 redundancy를 줄이기 위해 주소에 관한 데이터를 별도로 저장하고 관리하는 Address Entity를 둔다. Staff, Member, Distributor entity의 주소 데이터 attribute는 Address Entity의 PK를 값으로 가지는 FK가 되어 tuple의 주소 데이터를 관리할 때 해당 PK값을 가지고 있는 Address Entity의 tuple을 참고할 수 있도록 한다. 따라서 직원 · 회원 · 배급사의 정보를 조회 · 수정 · 삭제할 때 FK 값에 해당하는 Address Entity의 tuple도 같이 참조가 된다. |
| 2) | 회원 정보를 조회할 때 ‘내가 찜한 컨텐츠’에 관한 데이터는 별도의 테이블인 Dibs Table에 저장되어 있기 때문에 회원 번호를 가지고 Dibs Table에서 해당 회원 번호를 FK로 갖고 있는 tuple들을 찾아서 조회할 수 있도록 한다. 또한 해당 내역을 찜한 회원뿐만이 아니라 그 컨텐츠에 관한 데이터도 조회할 수 있어야 하므로 Content의 PK를 FK로 갖도록 한다. |
| 3) | 회원이 어떤 컨텐츠를 찜할 때 하나의 찜 내역에 대해서는 한 회원에 대응이 되지만, 한 회원은 여러 개의 찜 내역을 생성할 수 있으므로 Member와 Dibs 간의 one-to-many relationship을 갖는다. 또한 하나의 컨텐츠는 하나의 찜 내역에 대응이 되고, 마찬가지로 하나의 찜 내역도 한 컨텐츠에 대응되므로 Dibs와 Content 사이에는 one-to-one relationship을 갖도록 한다. |
| 4) | 컨텐츠는 배급사에 의해 배급이 되어야 한다는 것이 선행되어야 한다. 그래서 배급사별 동영상 리스트 내역을 조회할 때 이에 관한 데이터는 Content 테이블과 별도의 테이블에 저장하지 않고, 바로 Content 테이블에서 배급사 번호를 가지고 Content Table에서 해당 배급사 번호를 FK로 갖고 있는 tuple들을 찾아서 조회할 수 있도록 한다. 한 배급사 당 여러 개의 컨텐츠를 배급할 수 있고, 한 컨텐츠는 대표적인 하나의 배급사에 의해 배급된다는 점을 반영하여 Distributor와 Content entity 간의 one-to-many relationship을 갖게 한다. |
| 5) | 컨텐츠 배달 내역을 관리할 때 이에 관한 데이터는 별도의 테이블인 DeliverList Table에 저장되어 있는데, 배달 내역에는 배달을 받은 회원의 정보와 배달이 된 컨텐츠 정보를 모두 출력하기 위해 회원 번호와 컨텐츠 번호 두 가지의 FK를 갖는다. 이를 통해 배달 내역을 조회할 때 어느 회원이 어떠한 컨텐츠를 배달받았는지 확인할 수 있도록 한다. |
| 6) | 회원이 어떤 컨텐츠를 배달받을 때 하나의 배달 내역에 대해서는 한 회원에 대응이 되지만, 한 회원은 여러 개의 배달 내역을 생성할 수 있으므로 Member와 DeliverList 간의 one-to-many relationship을 갖는다. 또한 하나의 컨텐츠는 하나의 배달 내역에 대응이 되고, 마찬가지로 하나의 배달 내역도 한 컨텐츠에 대응되므로 DeliverList와 Content 사이에는 one-to-one relationship을 갖도록 한다. |
| 7) | 회원이 어떠한 컨텐츠를 시청한 데이터를 갖는 내역을 저장하는 WatchList는 회원에게 컨텐츠가 배달되어야 한다는 상황이 선행되는 점을 고려하여 DeliverList의 배달 내역을 WatchList가 상속받을 수 있도록 부모-자식 관계의 relationship을 형성한다. 또한 회원별 시청 내역을 조회하는 기능이 구현 가능하도록 WatchList에서 각각의 tuple들이 회원 번호와 컨텐츠 번호를 FK로 가지게 하여 이들을 통해 외부에 저장된 Member와 Content의 데이터를 참조할 수 있게끔 한다. |
| 8) | 모든 entity에서 각각의 attribute의 valid data type과 같은 domain 지정하고 attribute별로 Null 값을 가질 수 있는지의 여부를 설정하여 무결성을 보장하면서 저장된 데이터의 일관성을 유지할 수 있도록 한다. 각각의 attribute에 대응되는 domain의 세부적인 내용은 [표 3 - 1]에 자세히 서술되어 있다. |