|  |
| --- |
|  |
| 데이터베이스 입문 |
| Project #2 |

|  |
| --- |
| 20141334 임재현 |

데이터베이스입문 Project #2

20141334 임재현

1. 개념적 데이터베이스 모델링

1.1Entity 생성

* + 1. catUser : 사이트는 회원 정보를 받아 관리해야 한다. 회원 정보에는 ID, 이름, 이메일, 연령대, 성별, 활동지역, 가입일, 가입 그룹, 가입된 그룹에서의 역할, 획득한 업적이 있어야 한다. 따라서 회원이라는 entity를 생성, catUser 로 명명했다.
    2. catGroup : 사이트는 회원들이 모여 있는 그룹에 대한 정보를 관리한다. 그룹 정보는 그룹 ID, 그룹 이름, 그룹 구성원, 그룹 활동 지역, 그룹 개설일에 대한 정보를 포함한다. 따라서 그룹이라는 entity를 생성, catGroup 으로 명명했다.
    3. Cat : 사이트는 고양이들을 관리한다. 고양이들은 고유 ID를 가지며, 사이트는 고양이의 이름, 사진, 성별, 나이, 서식지, 성격, 사람에 대한 경계, 털빛, 중성화 여부, 유기 여부, 병력, 기타 특이사항 등의 상세정보를 포함해야 한다. 따라서 고양이라는 entity를 생성, Cat 으로 명명했다.
    4. Foodplace : 사이트는 고양이들이 식사를 하는 급식소에 대한 정보를 관리한다. 급식소들은 고유 ID를 가지며, 급식소 위치, 요일별/시간별 급식 담당자, 급식소를 사용하는 고양이, 현재 급식소 사료/물 현황에 대한 정보를 포함한다. 따라서 급식소라는 entity를 생성, Foodplace 로 명명했다.
    5. Food : 각 그룹은 사용하는 사료의 재고를 관리한다. 사료의 ID, 사료명, 사료 구입처, 구입 가격, 사료의 잔량, 사료의 위치에 대한 정보를 포함해야 한다. 따라서 사료라는 entity를 생성, Food 로 명명했다.
    6. Schedule : 운영 시스템에서 급식소의 담당 인원, 담당 요일, 담당 시간, 수행 여부를 확인 할 수 있어야 한다. 따라서 급식 스케줄이라는 entity를 relation의 형태로 생성, Schedule 로 명명했다.
    7. Member : 그룹은 소속 회원 목록을 알 수 있어야 하며 회원은 소속 그룹에서의 역할을 따로 얻는다. 따라서 catUser-Group 사이의 relation attribute를 만들기 위해 associative entity의 형태로 멤버라는 entity를 생성, Member로 명명했다.
    8. catPhoto : 그룹에는 고양이의 사진이 업로드 된다. 따라서 고양이 사진이라는 entity를 생성, catPhoto로 명명했다. 해당 entity에는 고양이의 사진과 사진 속 고양이의 ID, 게시자 ID 와 본문, 사진을 게시한 게시일이 표시된다. (해시태그는 본문에 포함시키고 웹디자인 단계에서 구현하며, 좋아요 및 댓글 기능도 웹디자인 단계에서 구현한다.)
    9. House : 고양이 모임은 겨울철 추위를 견디기 위한 겨울집을 제공한다. 겨울집은 ID, 겨울집 이름, 겨울집 위치, 겨울집 사진, 겨울집 설치일, 겨울집 철거 예정일, 겨울집 사용 고양이에 대한 정보가 포함된다. 따라서 겨울집이라는 entity를 생성, House로 명명했다.
    10. TNR : 고양이 모임에서는 TNR(Trap-Neuter-Return)을 위한 서비스를 제공한다. TNR 은 TNR 을 위 한 일정을 올리고, 신청자를 모집하고, 신청자가 가능한 시간을 체크하는 기능을 제공한다. 또한 TNR 이후 상태를 추적할 수 있는 기능을 제공한다. TNR의 일정을 열람하고 수정할 수 있도록 중성화라는 entity를 생성, TNR로 명명했다.
    11. Rescue : 고양이 모임에서는 유기된 고양이 및 질병이나 상해를 입은 고양이들을 구조하기 위한 서비스를 제공한다. 유기되거나 질병 및 상해를 입은 고양이를 발견할 시에는 해당 고양 이의 사진을 올리고 발견일, 발견 장소, 고양이의 사진, 이름, 특이사항을 업로드 할 수 있고 입양과 관련된 정보들을 제공할 수 있다. 따라서 Cat-catUser 사이의 relation attribute를 만들기 위해 associative entity의 형태로 entity를 생성, Rescue로 명명했다.
    12. Account : 각 그룹의 회계는 회계 정보를 입력할 수 있고, 그룹의 구성원은 회계 정보에 대한 열람이 가능하다. 따라서 catGroup-catUser 사이의 relation attribute를 만들기 위해 associative entity의 형태로 entity를 생성, Account로 명명했다.
  1. Attributes 도출
     1. catUser : 회원 정보에는 ID, 이름, 이메일, 연령대, 성별, 활동지역, 가입일, 가입 그룹, 가입된 그룹에서의 역할, 획득한 업적이 있어야 한다. 이에 따라 필요한 Attribute들을 아래와 같이 정하고 명명했다. 가입한 그룹과 그룹에서의 역할은 Member에서 해당 회원의 ID와 해당 그룹의 ID를 키로 받고 기입한다.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 번호 | Name | Attribute name  (in ERwin) | Domain | 내용 |
| 1 | ID | User\_id | ID | 회원의 ID를 저장 |
| 2 | Name | User\_name | Name | 회원의 이름을 저장 |
| 3 | Email | User\_email | String | 회원의 이메일을 저장 |
| 4 | Age | User\_age | Age | 회원의 연령대를 저장 |
| 5 | Gender | User\_gender | Gender | 회원의 성별을 저장 |
| 6 | Address | User\_address | Address | 회원의 거주지역을 저장 |
| 7 | Joindate | User\_joindate | Date | 회원의 가입일을 저장 |
| 8 | Achievements | User\_ach | String | 회원이 달성한 업적을 저장 |

* + 1. catGroup : 그룹 정보는 그룹 ID, 그룹 이름, 그룹 구성원, 그룹 활동 지역, 그룹 개설일에 대한 정보를 포함한다. 이에 따라 필요한 Attribute들을 아래와 같이 정하고 명명했다. 해당 그룹의 구성원은 Member에서 해당 그룹의 ID를 키로 받아 확인한다.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 번호 | Name | Attribute name  (in ERwin) | Domain | 내용 |
| 1 | ID | Group\_id | ID | 그룹의 ID를 저장 |
| 2 | Name | Group\_name | Name | 그룹의 이름을 저장 |
| 3 | Address | Group\_address | Address | 그룹의 활동 지역을 저장 |
| 4 | Opendate | Group\_date | Date | 그룹의 개설일을 저장 |

* + 1. Cat : 고양이 정보는 고유 ID, 고양이의 이름, 사진, 성별, 나이, 서식지, 성격, 사람에 대한 경계, 털빛, 중성화 여부, 유기 여부, 병력, 기타 특이사항 등의 상세정보를 포함해야 한다. 이에 따라 필요한 Attribute들을 아래와 같이 정하고 명명했다. 하나의 고양이에 대해 많은 수의 사진이 올라오므로 대표사진만 따로 지정해 저장하고, 게시물의 사진들은 catPhoto에서 다룬다.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 번호 | Name | Attribute name  (in ERwin) | Domain | 내용 |
| 1 | ID | Cat\_id | ID | 고양이의 ID를 저장 |
| 2 | Name | Cat\_name | Name | 고양이의 이름을 저장 |
| 3 | Picture | Cat\_pic | Picture | 고양이의 대표사진을 저장 |
| 4 | Gender | Cat\_gender | Gender | 고양이의 성별을 저장 |
| 5 | Age | Cat\_age | Age | 고양이의 나이를 저장 |
| 6 | Address | Cat\_address | Address | 고양이의 서식지를 저장 |
| 7 | Personality | Cat\_personality | String | 고양이의 성격을 저장 |
| 8 | Alert Level | Cat\_alevel | String | 고양이의 경계정도를 저장 |
| 9 | Color | Cat\_color | String | 고양이의 털빛을 저장 |
| 10 | Surgery | Cat\_surgery | Bool | 고양이의 중성화 여부를 저장 |
| 11 | Drop | Cat\_drop | Bool | 고양이의 유기 여부를 저장 |
| 12 | History | Cat\_history | String | 고양이의 병력을 저장 |
| 13 | Etc | Cat\_etc | String | 고양이의 기타 특이사항을 저장 |

* + 1. Foodplace : 급식소들은 고유 ID를 가지며, 급식소 위치, 요일별/시간별 급식 담당자, 급식소를 사용하는 고양이, 현재 급식소 사료/물 현황에 대한 정보를 포함한다. 이에 따라 필요한 Attribute들을 아래와 같이 정하고 명명했다. 해당 급식소를 사용하는 고양이는 catMember에서 급식소의 ID를 키로 받아 조회한다. 또한 하나의 급식소는 하나의 그룹에 의해서만 관리된다고 가정, 관리하는 그룹의 ID를 Attribute로 채택했다. 급식 담당자, 잔량과 위치정보의 경우 Schedule에 기록해둘 수 있게 했다.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 번호 | Name | Attribute name  (in ERwin) | Domain | 내용 |
| 1 | ID | Foodplace\_id | ID | 급식소의 ID를 저장 |
| 2 | GroupID | Group\_id | ID | 관리 그룹의 ID를 저장 |
| 3 | Address | Foodplace\_address | Address | 급식소의 위치를 저장 |

* + 1. Food : 사료 정보에는 사료의 ID, 사료명, 사료 구입처, 구입 가격, 사료의 잔량, 사료의 위치에 대한 정보를 포함해야 한다. 이에 따라 필요한 Attribute들을 아래와 같이 정하고 명명했다. 잔량과 위치정보의 경우 Schedule에 기록해둘 수 있게 했다.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 번호 | Name | Attribute name  (in ERwin) | Domain | 내용 |
| 1 | ID | Food\_id | ID | 사료의 ID를 저장 |
| 2 | Name | Food\_name | Name | 사료의 이름을 저장 |
| 3 | Seller | Food\_seller | String | 사료의 구매처를 저장 |
| 4 | Price | Food\_price | Price | 사료의 구입 가격을 저장 |

* + 1. Schedule : 운영 시스템에서 급식소의 담당 인원, 담당 요일, 담당 시간, 수행 여부를 확인 할 수 있어야 한다. 이에 따라 필요한 Attribute들을 아래와 같이 정하고 명명했다.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 번호 | Name | Attribute name  (in ERwin) | Domain | 내용 |
| 1 | GroupID | Group\_id | ID | 그룹의 ID를 저장 |
| 2 | FoodplaceID | Foodplace\_id | ID | 급식소의 ID를 저장 |
| 3 | FoodID | Food\_id | ID | 급식소의 사용 사료의 ID를 저장 |
| 4 | Address | Food\_address | Address | 사료의 저장 위치를 저장 |
| 5 | Quantity\_Food | Food\_quantity | String | 사료의 잔량을 저장 |
| 6 | Quantity\_Water | Water\_quantity | String | 물의 잔량을 저장 |
| 7 | Actor | Do\_actor | Name | 담당자의 이름을 저장 |
| 8 | When | Do\_when | Datetime | 담당 요일과 시간을 저장 |
| 9 | Did | Do\_did | Bool | 활동 여부를 저장 |
| 10 | Category | Do\_category | String | 활동 분류를 저장 |
| 11 | Act | Do\_act | String | 활동 내역을 저장 |
| 12 | Detail | Do\_detail | String | 특이사항을 저장 |

* + 1. Member : 그룹은 소속 회원 목록을 알 수 있어야 하며 회원은 소속 그룹에서의 역할을 따로 얻는다. 이에 따라 필요한 Attribute들을 아래와 같이 정하고 명명했다.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 번호 | Name | Attribute name  (in ERwin) | Domain | 내용 |
| 1 | GroupID | Group\_id | ID | 그룹의 ID를 저장 |
| 2 | UserID | User\_id | ID | 회원의 ID를 저장 |
| 3 | Role | User\_role | String | 회원의 역할을 저장 |

* + 1. catPhoto : 고양이의 사진이 업로드 된다. 사진에는 고양이의 사진과 사진 속 고양이의 ID, 게시자 ID 와 본문, 해시태그, 댓글과 좋아요, 조회수, 그리고 사진을 게시한 게시일이 표시된다. 이에 따라 필요한 Attribute들을 아래와 같이 정하고 명명했다.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 번호 | Name | Attribute name  (in ERwin) | Domain | 내용 |
| 1 | CatID | Cat\_id | ID | 고양이의 ID를 저장 |
| 2 | UserID | User\_id | ID | 회원의 ID를 저장 |
| 3 | Datetime | Photo\_time | Datetime | 사진의 게시일을 저장 |
| 4 | Content | Photo\_content | String | 본문 내용을 저장 |
| 5 | Picture | Photo\_pic | Picture | 사진을 저장 |

* + 1. House : 고양이 모임은 겨울철 추위를 견디기 위한 겨울집을 제공한다. 겨울집은 ID, 겨울집 이름, 겨울집 위치, 겨울집 사진, 겨울집 설치일, 겨울집 철거 예정일, 겨울집 사용 고양이에 대한 정보가 포함된다. 이에 따라 필요한 Attribute들을 아래와 같이 정하고 명명했다.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 번호 | Name | Attribute name  (in ERwin) | Domain | 내용 |
| 1 | HouseID | House\_id | ID | 겨울집의 ID를 저장 |
| 2 | Address | House\_address | Address | 겨울집의 위치를 저장 |
| 3 | Picture | House\_pic | Picture | 겨울집의 사진을 저장 |
| 4 | Date | House\_date | Date | 겨울집의 설치일을 저장 |
| 5 | Due | House\_duedate | Date | 겨울집의 철거일을 저장 |

* + 1. TNR : 그룹은 TNR 을 위한 일정을 올리고, 신청자를 모집하고, 신청자가 가능한 시간을 체크하는 기능을 제공한다. 또한 TNR 이후 상태를 추적할 수 있는 기능을 제공한다. 이에 따라 필요한 Attribute들을 아래와 같이 정하고 명명했다. TNR Name의 경우 연도, 학기수, 차수를 표기한다.

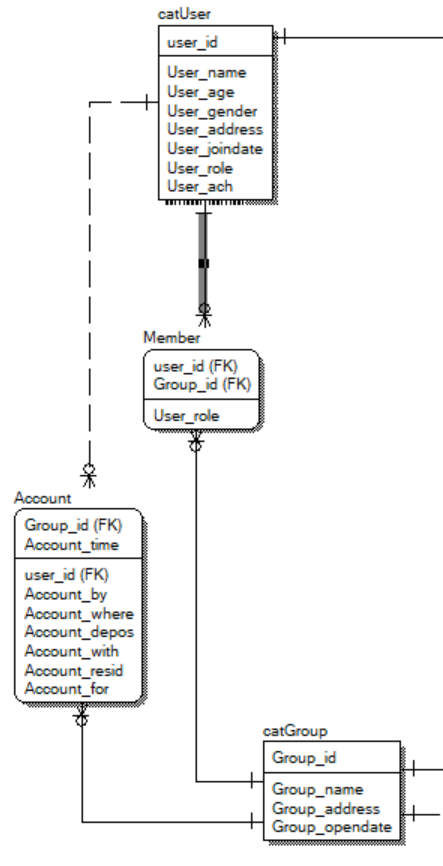
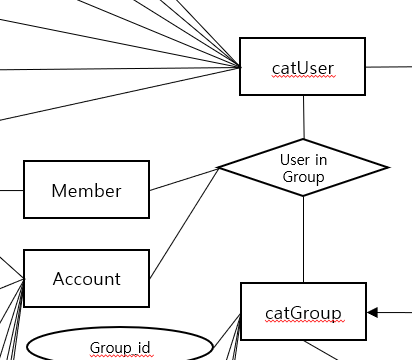
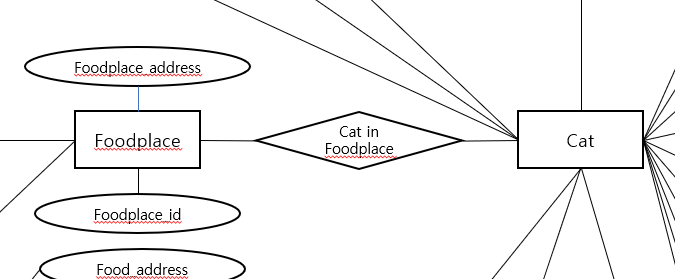
|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 번호 | Name | Attribute name  (in ERwin) | Domain | 내용 |
| 1 | CatID | Cat\_id | ID | TNR시킬 고양이의 ID를 저장 |
| 2 | Date | TNR\_date | Date | TNR 예정일자를 저장 |
| 3 | UserID | User\_id | ID | TNR 담당자의 ID를 저장 |
| 4 | Name | TNR\_name | String | 일정의 이름을 저장 |
| 5 | TNR\_Detail | TNR\_detail | Datetime | 해당 일정의 상세시간대를 저장 |

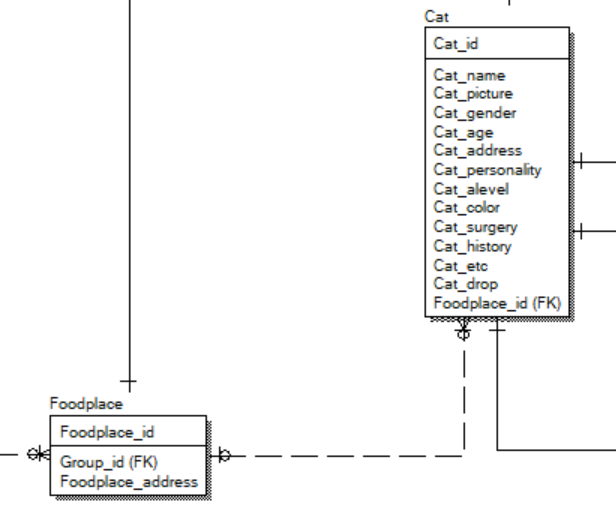
* + 1. Rescue : 그룹에서는 유기되거나 질병 및 상해를 입은 고양이를 발견할 시에는 해당 고양 이의 사진을 올리고 발견일, 발견 장소, 고양이의 사진, 이름, 특이사항을 업로드 할 수 있고 입양과 관련된 정보들을 제공할 수 있다. 이에 따라 필요한 Attribute들을 아래와 같이 정하고 명명했다. 후에 입양 정보(병력, 털빛 등)를 Cat 데이터와 원활하게 연동시키기 위해 이름 대신 ID를 Attribute로 사용한다.

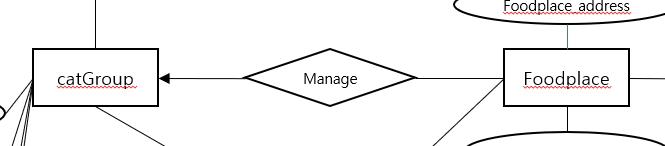
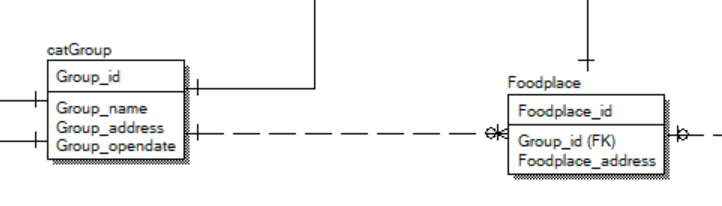
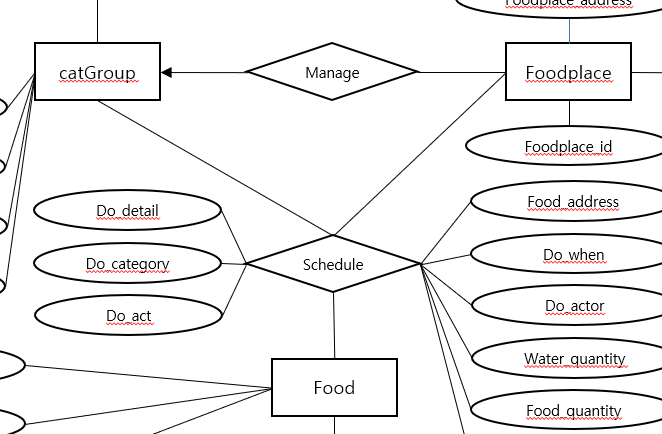
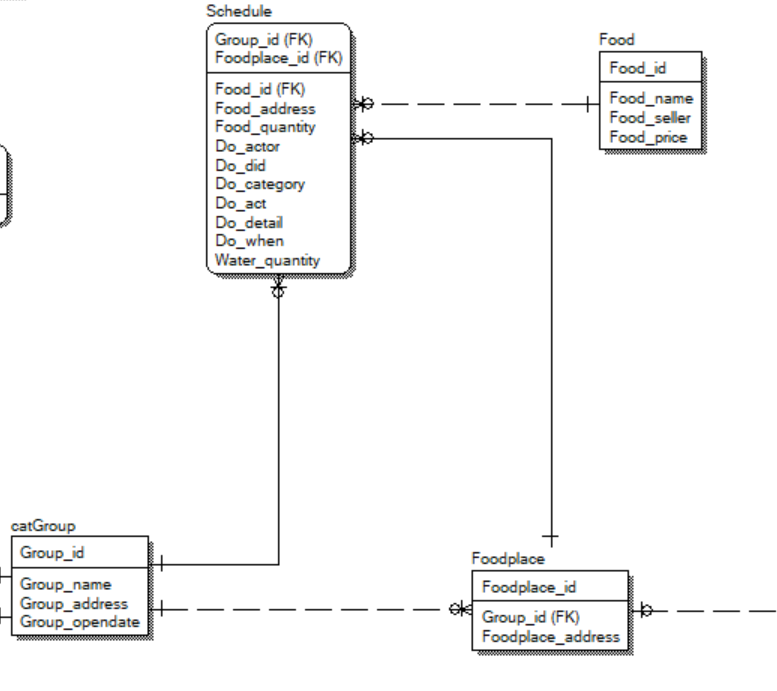
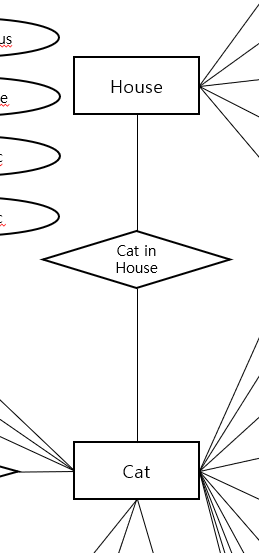
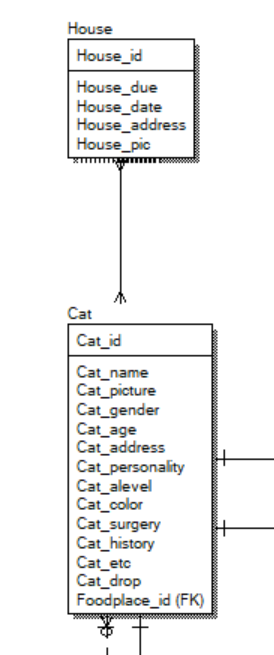
|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 번호 | Name | Attribute name  (in ERwin) | Domain | 내용 |
| 1 | CatID | Cat\_id | ID | 발견한 고양이의 ID를 저장 |
| 2 | Date | Rescue\_date | Date | 발견일을 저장 |
| 3 | Pic | Rescue\_pic | Picture | 발견한 고양이의 사진을 저장 |
| 4 | Etc | Rescue\_etc | String | 특이사항을 저장 |
| 5 | Status | Rescue\_status | String | 현재상태(유기/실종/구조)를 저장 |

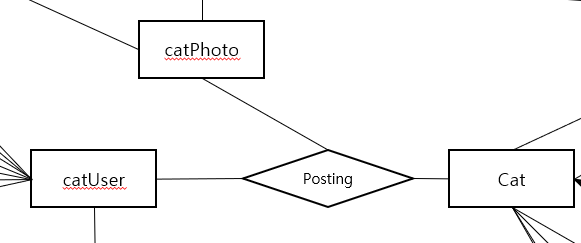
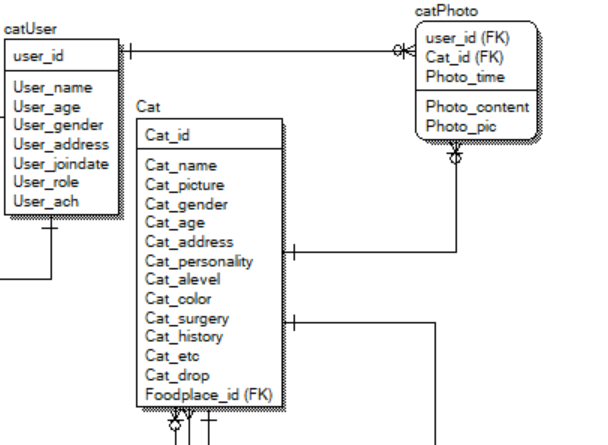
* + 1. Account : 각 그룹의 회계는 회계 정보를 입력할 수 있고, 그룹의 구성원은 회계 정보에 대한 열람이 가능하다. 특정 시간에 기록된 특정 그룹의 회계 정보는 사용 시각, 적요(거래수단), 기재 내용(거래 장소), 찾은(사용한) 금액, 맡긴 금액, 거래 후 잔액, 사용내역을 포함해야 한다. 이에 따라 필요한 Attribute들을 아래와 같이 정하고 명명했다.

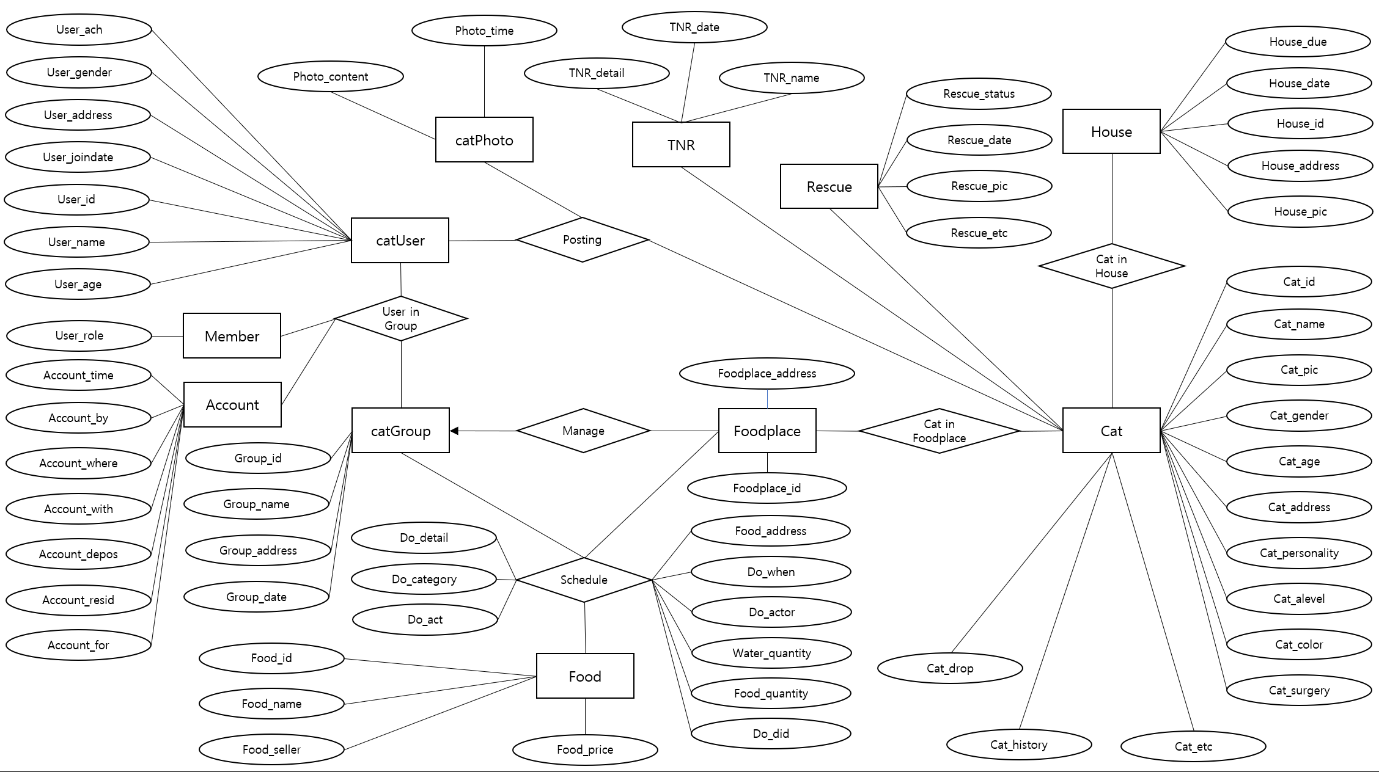
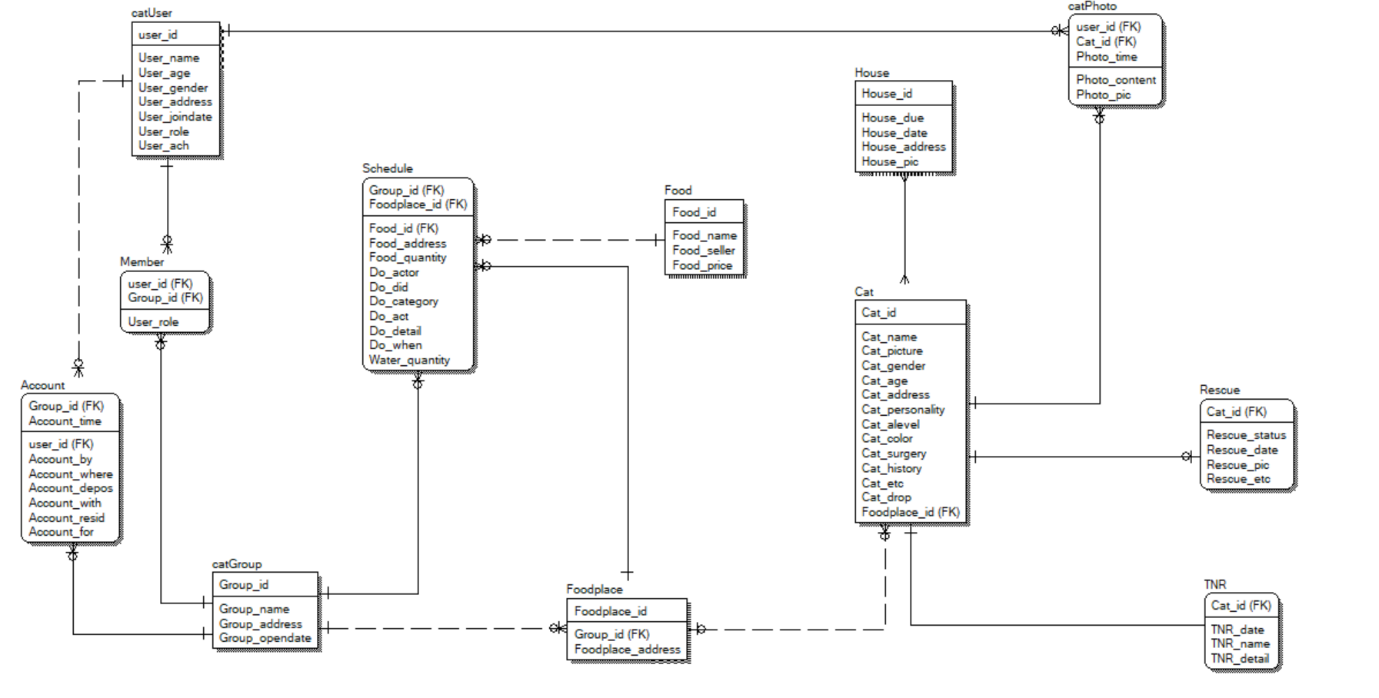
|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 번호 | Name | Attribute name  (in ERwin) | Domain | 내용 |
| 1 | GroupID | Group\_id | ID | 그룹의 ID를 저장 |
| 2 | Time | Account\_time | Datetime | 회계정보의 시간대를 저장 |
| 3 | By | Account\_by | String | 거래수단(적요)을 저장 |
| 4 | Where | Account\_where | String | 거래장소를 저장 |
| 5 | Withdraw | Account\_with | INTEGEReger | 사용한 금액을 저장 |
| 6 | Deposit | Account\_depos | INTEGEReger | 예치한 금액을 저장 |
| 7 | Residual | Account\_resid | INTEGEReger | 거래 후 잔액을 저장 |
| 8 | For | Account\_for | String | 사용내역을 저장 |

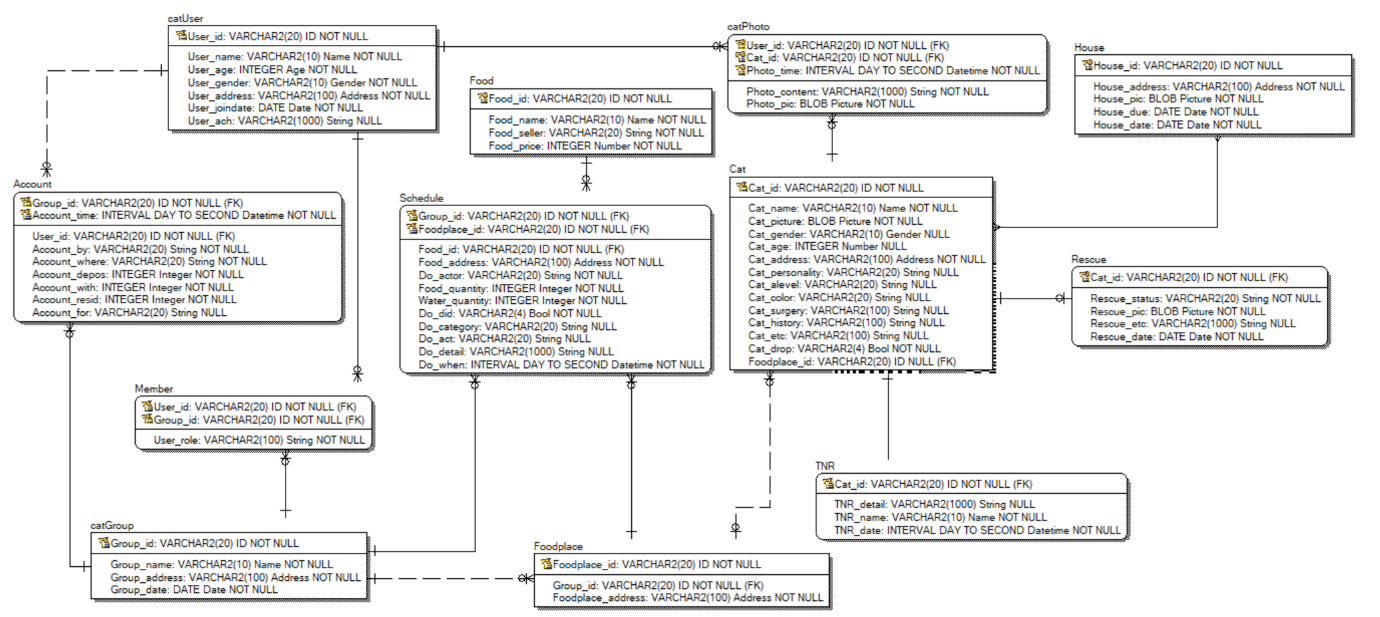
* 1. Key 식별
     1. catUser : 유저의 id는 고유한 값이므로 User\_id를 Primary Key(PK)로 선택했다.
     2. catGroup : 그룹의 id는 고유한 값이므로 Group\_id를 PK로 선택했다.
     3. Cat : 고양이의 id는 고유한 값이므로 Cat\_id를 PK 로 선택했다.
     4. Foodplace : 급식소의 id는 고유한 값이므로 Foodplace\_id를 PK로 선택했다.
     5. Food : 사료의 id는 고유한 값이므로 Food\_id를 PK로 선택했다.
     6. Schedule : 활동내역을 식별하기 위해서는 그룹의 id, 급식소의 id가 모두 필요하며 이는 각각 catGroup과 Foodplace의 id 값들(Group\_id, Foodplace\_id)을 Foreign Key(FK)로 받은 값이다. 따라서 두개의 FK를 PK로 선택했다. 또한 사용되는 사료의 id는 Food의 id 값을 FK로 받은 값이다.
     7. Member : 그룹에 대해 속한 회원을 조회해야 하며, 반대로 회원이 어떤 그룹에 속하는지도 확인할 수 있어야 한다. 또한 특정 회원이 어떤 그룹에서 맡은 역할을 식별하기 위해서는 회원의 id와 그룹의 id가 모두 필요하다. 이는 각각 catUser와 catGroup의 id값들(User\_id, Group\_id)을 FK로 받은 값이다. 따라서 두 개의 FK를 PK로 선택했다.
     8. catPhoto : 특정 사진을 식별하기 위해서는 사진속 고양이의 id, 게시자의 id, 올린 시간대(특정 게시자가 같은 고양이에 대해 여러 번에 걸쳐 사진을 올리는 경우를 식별하기 위함)이 모두 필요하며 이는 각각 Cat, catUser의 id값들(Cat\_id, User\_id)을 FK로 받은 값과 게시글의 작성시각(Photo\_time)이다. 따라서 총 세 개의 Attribute를 PK로 선택했다.
     9. House : 겨울집의 id는 고유한 값이므로 House\_id를 PK로 선택했다.
     10. TNR : TNR은 한 마리의 고양이에 대해 한 번만 이루어 지게 된다. 따라서 고양이의 id값은 고유한 값이므로 Cat\_id를 PK로 선택했다.
     11. Rescue : 특정 고양이에 대해 구조와 분양이 이루어 지게 되므로 고양이의 id값은 고유한 값이며 따라서 Cat\_id를 PK로 선택했다.
     12. Account : 특정 거래내역을 열람하기 위해서는 그룹과 시간대를 모두 특정해야 한다. 이는 catGroup에서 id값을 FK로 받은 값(Group\_id)과 시간대(Account\_time)이다. 이 두개의 값들을 PK로 선택했다.
  2. Relation 설정 및 Catdinality 설정
     1. User in Group : User entity와 Group entity 사이의 Relationship이다. Account는 특정 그룹에 대해 생성되는 레코드로, 그룹과는 식별관계이나 유저를 특정할 필요는 없으므로 유저와는 비식별관계이다. 또한 하나의 유저가 여러 개의 그룹에 가입할 수 있으므로 유저-멤버와 그룹-멤버는 각각 1 : N cardinality, N은 1 or More로 설정했다. Account의 경우에도 하나의 그룹에 여러 개의 레코드가 생기며, 유저는 다양한 그룹의 회계 내역을 조회할 수 있으므로 1 : N cardinality, N은 1 or More로 설정했다.
     2. Cat in Foodplace : Cat entity와 Foodplace entity 사이의 Relationship으로, 고양이가 사용하게 될 급식소를 통해 고양이의 정보를 관리할 수 있도록 설정하였다. 하나의 급식소에 다수의 고양이가 올 수 있으므로 1 : N의 Cardinality를 설정했다. Cat은 Foodplace와는 독립적인 관계이며 따라서 비식별관계이고, Null도 허용한다(오지 않을 수 있으므로). 따라서 N을 zero, one or more로 설정했다.

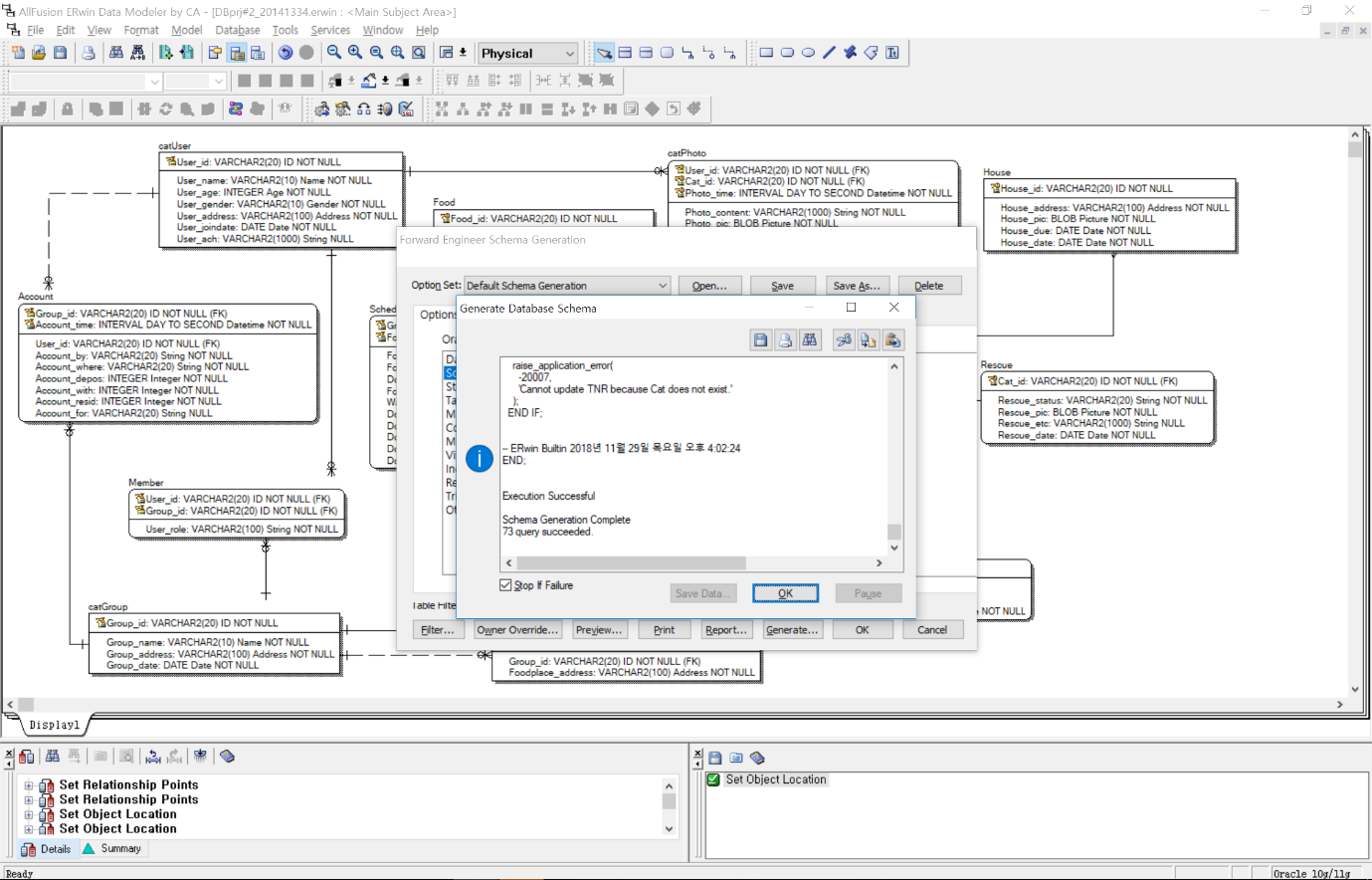
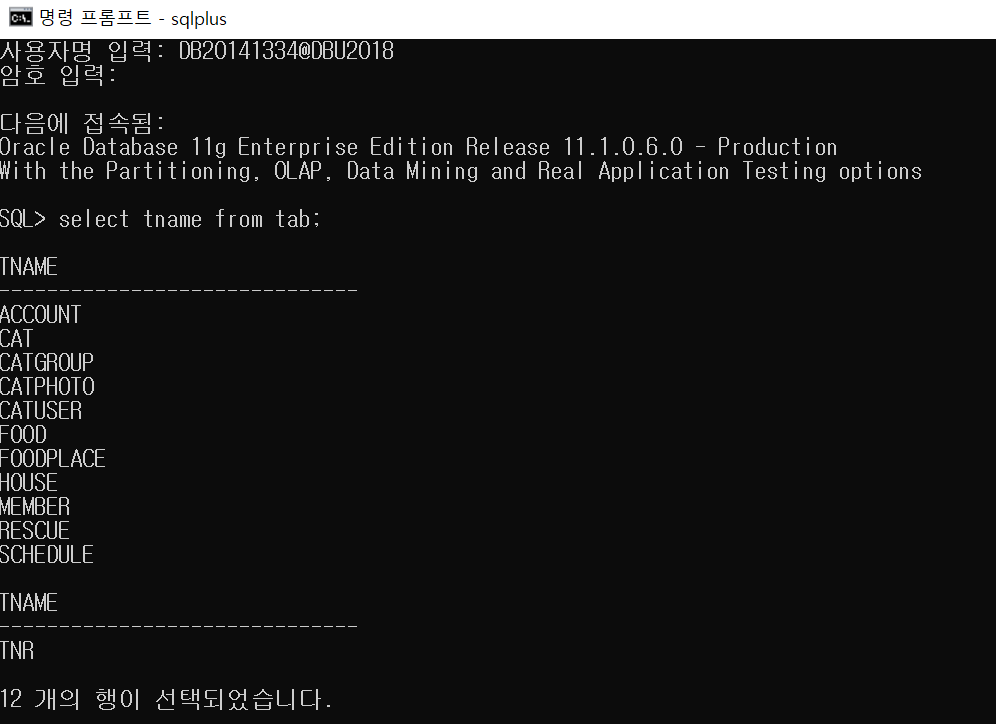


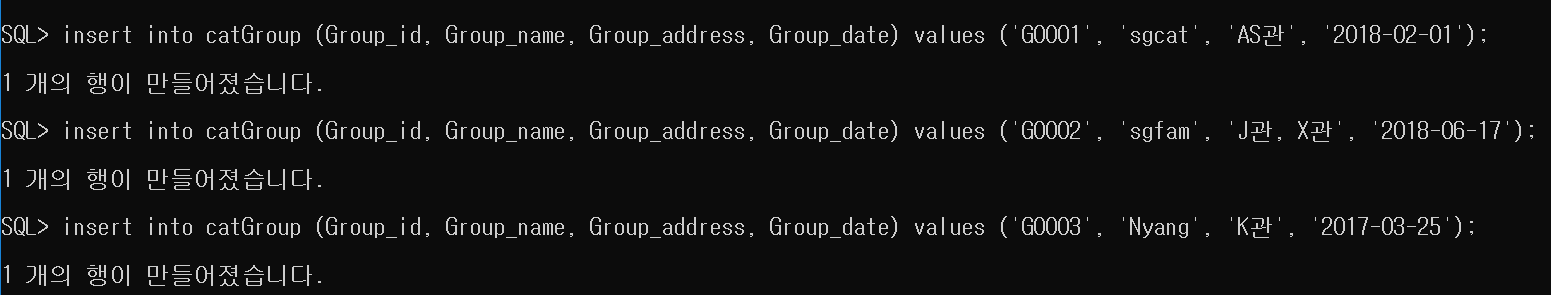
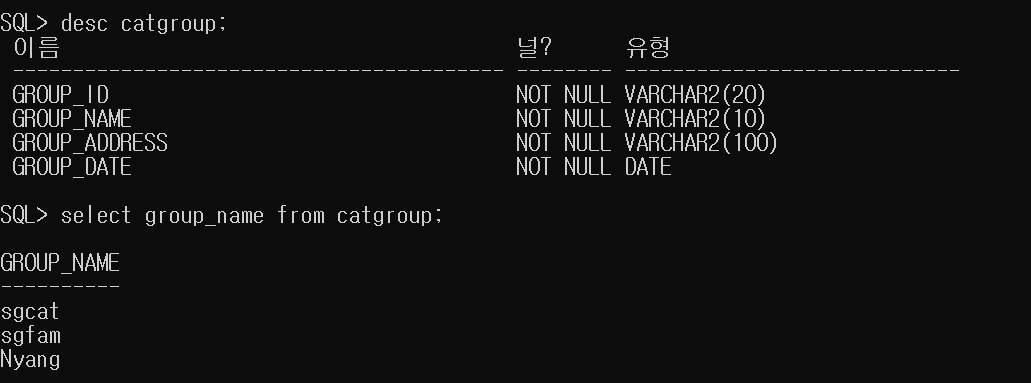
* + 1. Manage : Foodplace entity와 Group entity 사이의 Relationship으로, 그룹이 담당하는 Foodplace를 관리할 수 있도록 설정하였다. 하나의 그룹이 여러 개의 Foodplace를 관리할 수 있도록 1 : N의 Cardinality를 설정했다. 그룹에서 관리하는 Foodplace는 없을 수도, 다수일 수도 있으므로 N을 zero, one or more로 설정했다. Foodplace 관점에서는 급식소가 그룹이 생성하거나 관리하는 급식소가 아닐 수 있으므로 비식별관계로 두고, Null도 허용했다.
    2. Schedule : 급식 시스템 스케줄을 운영, 관리할 수 있도록 Foodplace, Food, Group 사이에 Relationship을 설정했다. 각 entity 들이 다양한 조합으로 연결될 수 있으므로 Cardinality는 Many-to-Many 일 것이다.
    3. Cat in House : Cat entity와 House entity 사이의 Relationship으로, 고양이가 사용하게 될 겨울집의 정보를 관리할 수 있도록 설정하였다. 하나의 겨울집에 다수의 고양이가 올 수 있으며, 하나의 고양이가 다수의 겨울집에 갈 수 있으므로 many to many Cardinality를 설정했다. Cat은 House와는 독립적인 관계이고 many to many c ardinality이므로 자연스럽게 비식별관계이고, Null도 허용 한다(오지 않을 수 있으므로).
    4. Posting : catUser entity와 Cat entity 사이의 Relationship으로, 사용자가 올린 고양이 사진 게시글을 관리할 수 있도록 설정하였다. 하나의 게시글에 하나의 게시자가 있으므로 1:1 Cardinality를 설정했고, 하나의 게시글에 올라오는 사진들은 같은 고양이라는 전제 하에 1:1 Cardinality를 설정했다. 모두 식별관계이며, Null은 허용하지 않는다.



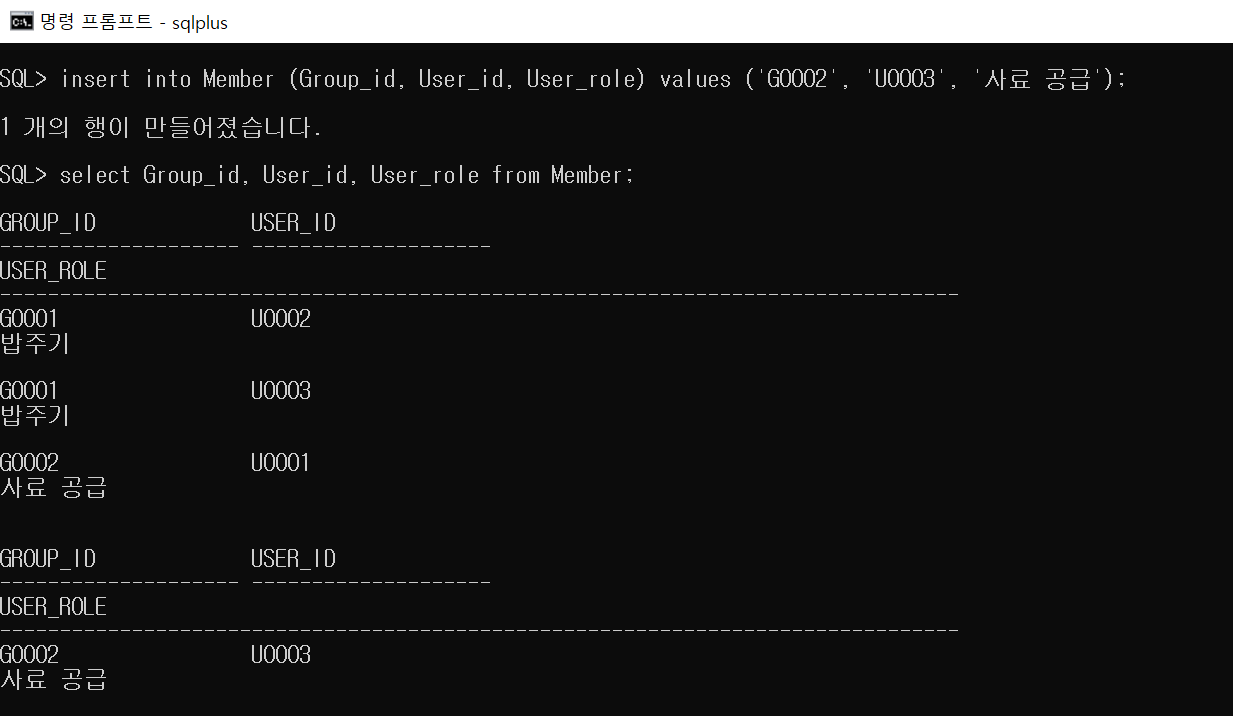
1. ER diagram, Logical Mode
   1. ER diagram : 1에서 생성한 Entity들과 그 Attributes를 총괄하는 전체 다이어그램은 다음과 같다.
   2. Logical Mode : ER win을 이용해 Logical Mode로 생성한 Entity 사이의 관계는 다음과 같다.
2. Physical Mode
   1. 도메인 설정
      1. ID : ID에 쓰이는 Domain은 String이며, Primary Key가 되므로 Not Null이다.
      2. Name : Name에 쓰이는 Domain은 String이며, 이름은 항상 존재하므로 Not Null이다.
      3. Bool : Bool의 경우, Yes/No라는 값 만을 출력하므로 String type이다. 수술 여부나 수행 여부는 항상 존재해야 하므로 Not Null이다
      4. Age : 연령을 기입하므로 INTEGER type로 지정했다. 사람의 경우는 Not Null, 고양이의 경우는 Null 허용으로 설정했다.
      5. String : 글을 집어넣는 란이므로 VARCHAR2() type로 지정했고, 길이제약은 각 Attribute에 따라 적당히 10~1000 사이의 값을 지정했다. Null 허용은 각 Attribute에 따라 적당히 지정했다.
      6. Address : 주소에 사용되는 Domain으로 String과 같이 지정했다. 길이제약은 100, Not Null이다.
      7. Gender : 성적 정체성을 표기한다. String과 같이 VARCHAR2() type로 지정, 길이제약은 10으로 지정했다. 사람의 경우는 Not Null, 고양이의 경우는 Null 허용으로 설정했다.
      8. Date : 날짜를 표기한다. 가입일/생성일의 경우, 연도/월/일에 대한 데이터를 가지고 있어야 하므로 Domain은 DATE type이다.
      9. Datetime : 날짜와 정확한 시각까지 표기한다. 시간(시, 분, 초) 데이터를 포함해야 하므로 INTEGERERVAL DAY TO SECOND type으로 지정한다.
      10. Price : 가격을 기입한다. INTEGER type로 지정했다.
      11. Integer : 각종 숫자를 기입한다. INTEGER type로 지정했다.
      12. Picture : Picture의 경우 사진을 저장해야 하므로, 이진 데이터 타입 BLOB type으로 지정한다. Null은 각 Attribute에 따라 적당히 지정한다.
   2. Physical Mode : Domain을 고려해 Physical Mode 로 구현한 전체 데이터베이스의 모습은 다음과 같다.

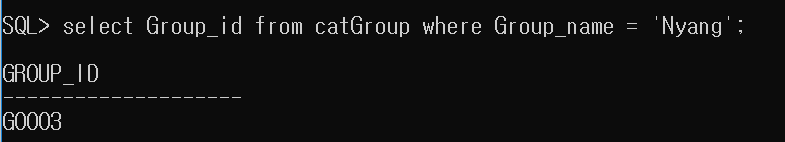
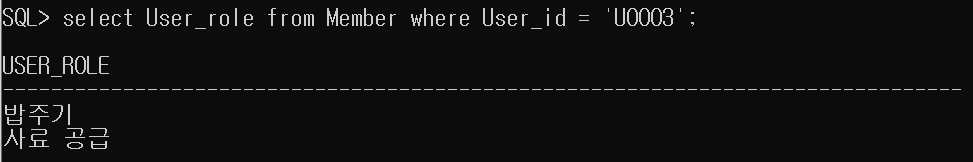
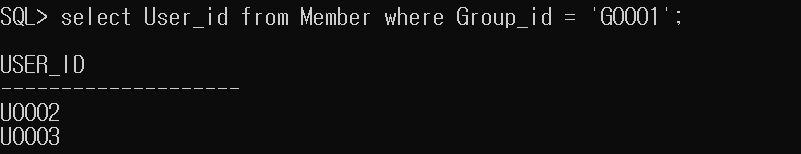


1. 설계에 대한 세부 요구사항 지정
   1. 메인 화면 : 메인화면에서 보이게 되는 회원정보는 catUser Entity, 그룹정보는 catGroup Entity, 고양이 정보는 Cat Entity에서 관리한다. 그룹 검색과 고양이 검색은 각각의 Entity에서, 회원 가입, 내 정보, 내 그룹은 Member Entity를 통해 이용할 수 있다.
   2. 그룹 화면 : 그룹 화면을 띄우면 catGroup Entity와 연결된 Cat, Schedule Entity를 통해 그룹에서 관리중인 고양이의 사진, 정보, 급식 정보를 열람할 수 있다. 또한, TNR, Rescue, House Entity를 통해 정보를 표시한다.
   3. 내 정보 화면 : 내 정보 화면에서 catUser Entity를 통해 내 정보를 확인할 수 있다.
   4. 회원 가입 화면 : 기입한 정보를 catUser Entity에 추가함으로써 회원 정보를 등록할 수 있다.
   5. 사진 화면 : catPhoto Entity를 통해 고양이의 사진과 게시글, 게시자 등의 정보를 확인할 수 있다.
   6. 고양이 열람 화면 : Group Entity를 통해 Schedule에서 그룹이 관리하는 Foodplace id를 확인하고, 해당 Foodplace를 이용하는 고양이들의 정보를 Cat Entity를 통해 확인할 수 있다.
   7. 사료 열람 화면 : Group Entity와 연결된 Schedule, Food Entity에서 사료에 대한 정보를 확인 할 수 있다.
   8. 급식소 열람 화면 : Group Entity와 연결된 Schedule, Foodplace Entity에서 그룹이 관리하는 급식소에 대한 정보를 확인 할 수 있다.
   9. 급식 일정 열람 화면 : Schedule table에서 급식 일정을 가져와 확인할 수 있으며, 급식 신청 시 table에 새로운 레코드를 추가한다.
   10. 유기 고양이 입양 화면 : Rescue Entity에서 Cat\_id로 해당 고양이를 찾아 특징을 확인 할 수 있다. 입양 신청시 Status를 입양과 관련된 값으로 하여 테이블에 추가한다.
   11. 구조 고양이 열람 화면 : Rescue Entity에서 Cat\_id로 해당 고양이를 찾아 특징을 확인 할 수 있다.
   12. TNR 일정 열람 화면 : TNR Entity를 통해 일정을 열람할 수 있다. TNR 신청 시 담당자의 이름 또한 화면에 보이게 한다.
   13. 겨울집 열람 화면 : House Entity를 통해 겨울집에 관련된 정보를 확인할 수 있다.
   14. 회계 화면 : 해당 그룹에 대한 Account Entity를 통해 회계 내역을 확인할 수 있다.
2. 테스트
   1. 테이블 생성 : 다음과 같이Erwin과 오라클을 연동시켜 테이블을 생성했다.
   2. 테이블에 데이터 추가1 : catGroup에 세 개의 그룹을 등록시켰다.



* 1. 테이블에 데이터 추가 2 : catUser에 세 명의 유저를 등록시켰다.
  2. 테이블에 데이터 추가 3 : Member 테이블에 데이터를 추가해 유저들이 그룹에 속하도록 등록시켰다.



* 1. 쿼리문 수행 1 : 특정 그룹의 ID를 확인한다.
  2. 쿼리문 수행 2 : 특정 유저가 하는 작업을 확인한다.
  3. 쿼리문 수행 3 : 특정 그룹의 멤버들을 확인한다.