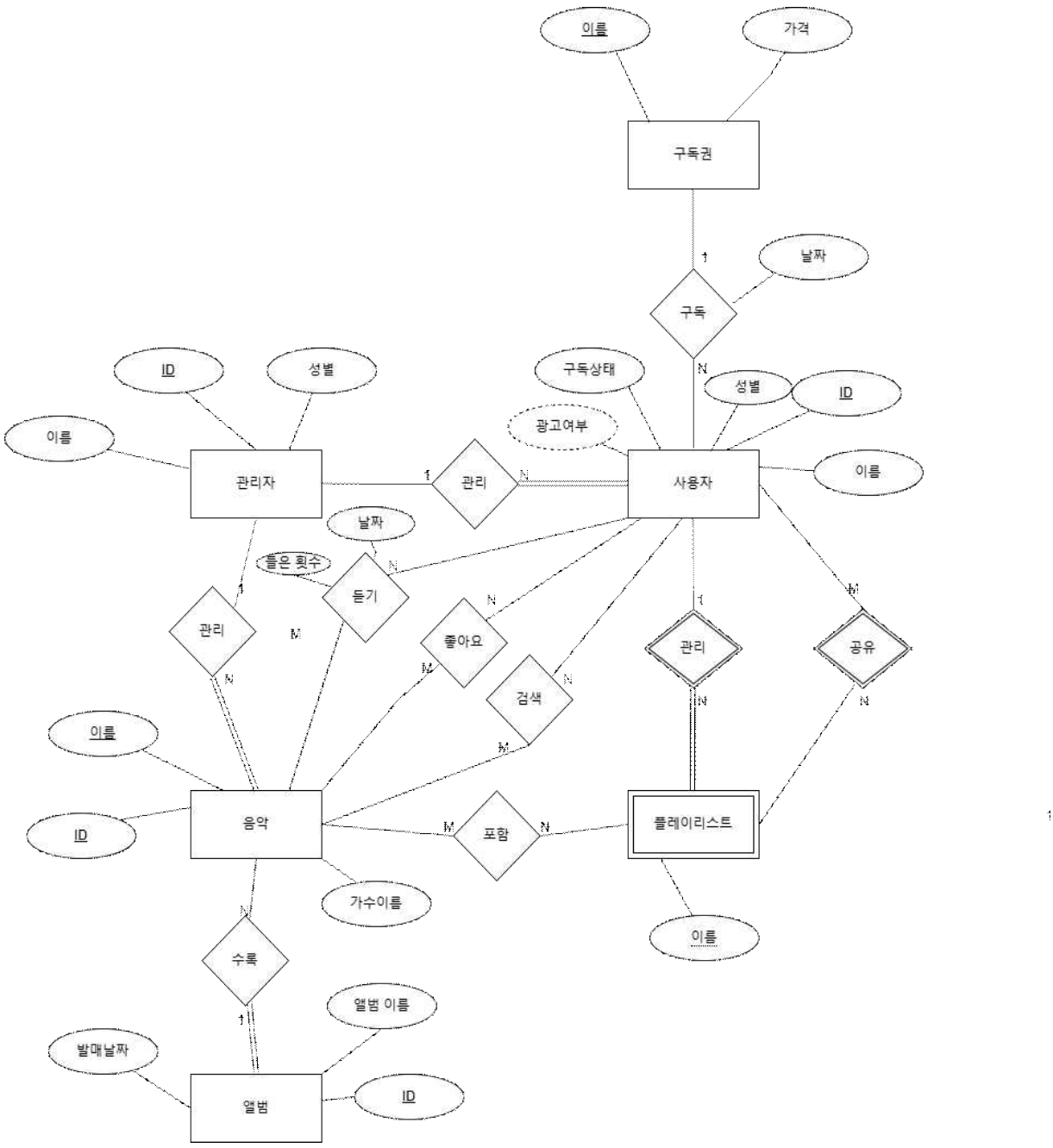
프로젝트2: Entity-Relationship (ER) 모델을 이용한 개념적 (conceptual) DB설계 및 Relation모델을 이용한 논리적(logical)DB설계

과목명: 데이터베이스 시스템 및 응용
2022031994 김준수

2-1



ER Model
수정 사항:
1. ‘알림’ 기능 삭제
2. 재생 횟수 카운트하는 attribute를 ‘듣기’ relationship에 추가
3. ‘구독’ entity 대신 ‘구독권’이라는 entity로 수정 후 구독을 relationship으로 설정
4. 플레이리스트는 사용자가 있어야지 존재할 수 있고, 플레이리스트의 고유 id로 구분하는 것이 아니라 사용자 id로 플레이리스트를 구분하므로 weak entity로 설정
5. 앨범 entity 추가
6. ‘사용자는 서비스에 가입하여 음악을 들을 수 있다’라는 부분은 관리자가 사용자를 등록해주면 가입이 된 것으로 간주하므로 따로 나타내지 않았다.

1. Entity

(1). **관리자**
- ID, 이름, 성별 등의 attribute 가진다.
- 관리자는 1명이라고 가정한다.

(2). **사용자**
- 구독 상태, 광고 여부, 사용자 ID, 이름, 성별 등의 attribute를 가진다.
- 광고 여부는 구독 상태로부터 알 수 있으므로 derived attribute로 설정한다.

(3). **음악**
- 음악 이름, 음악 ID, 가수 이름 등의 attribute 가진다.

(4). **플레이리스트**
- 플레이리스트 이름 attribute를 가진다.
- 사용자가 있어야 플레이리스트도 존재하므로 weak entity로 설정
- 사용자가 identifying owner이다.

(5). **구독권**
- 구독권은 1개라고 가정
- 구독권 이름, 가격 등의 attribute를 가진다.
- 구독 날짜로부터 30일 후를 구독 만료일로 정한다.

(6) **앨범**

- 발매 날짜, 앨범 이름, 앨범 ID 등의 attribute를 가진다.

2. Relationship

사용자 관리(관리자-사용자)
- 관리자 1명이 여러 사용자 관리할 수 있다. (등록, 삭제) (1:N 관계)
- 모든 사용자는 관리자를 가지므로 total paricipation이다.

음악 관리(관리자-음악)
- 관리자 1명이 여러 음악을 관리할 수 있다. (등록, 삭제) (1:N 관계)
- 모든 음악은 관리자를 가지기 때문에 total participation이다.

음악 듣기(사용자-음악)
- 사용자는 여러 음악을 들을 수 있고 음악 하나는 여러 사용자가 들을 수 있다. (M:N 관계)
- 음악을 들은 날짜와 음악을 들은 횟수를 attribute로 가진다.

음악 좋아요(사용자-음악)
- 음악 듣기와 마찬가지로 한 사용자는 여러 음악에 좋아요를 할 수 있고, 음악 하나에 여러 사용자가 좋아요할 수 있다. (M:N 관계)

음악 검색(사용자-음악)
- 한 사용자는 여러 음악을 검색할 수 있고, 음악은 여러 사용자에게 검색되어질 수 있다. (M:N 관계)

플레이리스트 관리(사용자-플레이리스트)
- 사용자는 여러 플레이리스트를 관리할 수 있다. (생성, 삭제) (1:N 관계)
- 플레이리스트는 사용자에 의해 관리되므로 total participation이다.

플레이리스트 공유(사용자-플레이리스트)
- 사용자는 여러 플레이리스트를 공유할 수 있고 플레이리스트도 여러 사용자에게 공유될 수 있다.(M:N 관계)

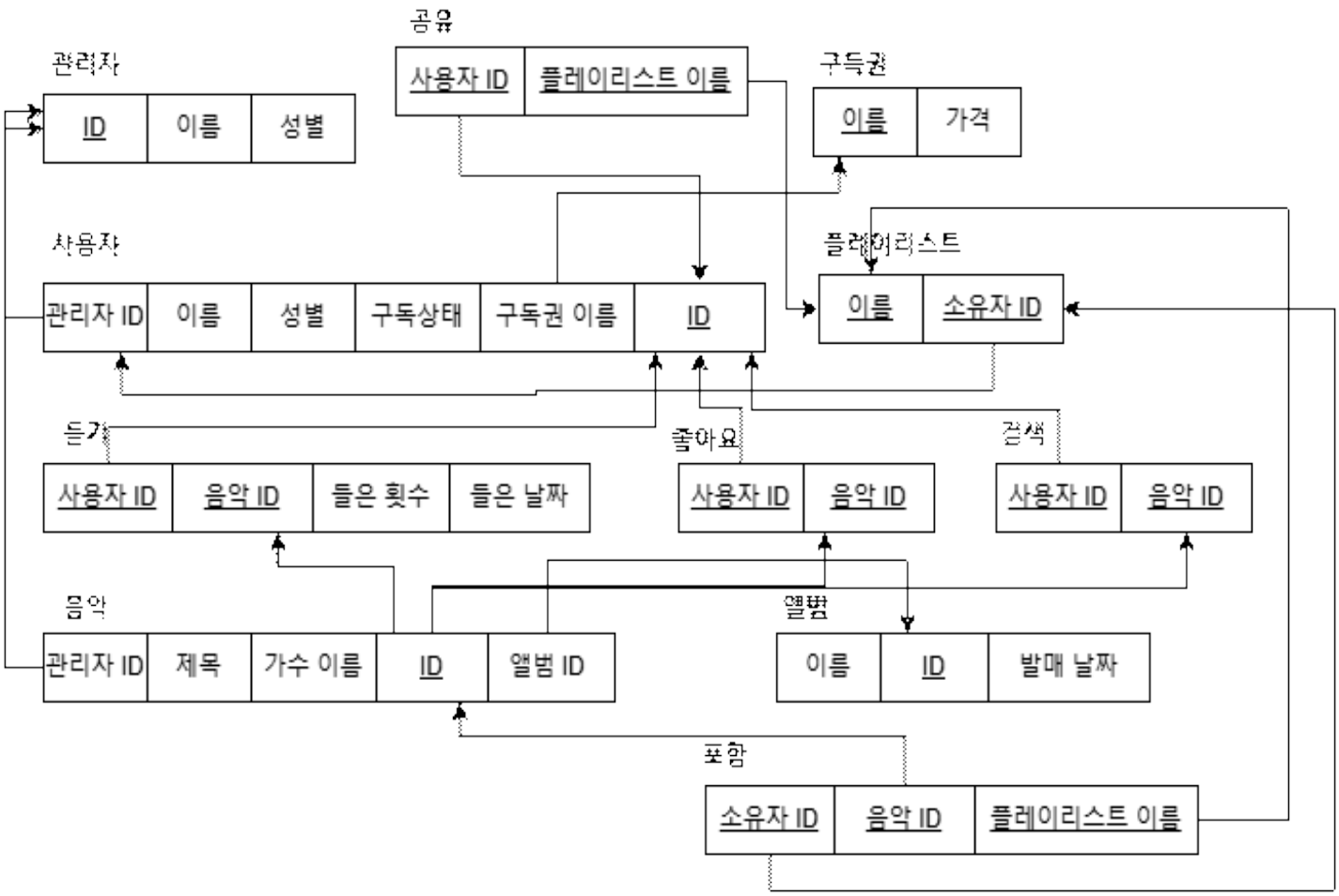
플레이리스트 포함(음악-플레이리스트)
- 여러 플레이리스트는 여러 음악을 포함할 수 있다. (M:N 관계)
- 음악이 없는 플레이리스트가 존재할 수 있으므로 (만들어만 두고 음악을 추가하지 않은 경우) partial이다.

구독권 구독(구독권-사용자)
- 여러 사용자는 하나의 구독권을 구독할 수 있다.

음악 수록(음악-앨범)
- 하나의 앨범은 여러 음악을 수록한다.(1:N 관계)
- 하나의 앨범에는 하나 이상의 음악이 포함되어야 하므로 total paricipation

이고, 모든 음악은 하나의 앨범에 포함되어야 하므로 total paricipation이다.

2-2



관리자

- ID, 이름, 성별을 attribute로 가지고 있다.
- ID를 primary key로 설정한다.

사용자

- ID, 이름, 성별, 구독 상태를 attribute로 가지고 있다.
- ID를 primary key로 설정한다.
- 관리자의 primary key인 ID를 '관리자 ID'라는 이름으로 foreign key로 가져온다.
- 구독권의 primary key인 이름을 '구독권 이름'이라는 이름으로 foreign key로 가져온다.

구독권

- 이름과 가격을 attribute로 가진다.
- 이름을 primary key로 설정한다.

음악

- 제목(위 ER diagram에서 '이름'이라는 attribute를 '제목'으로 수정), 가수 이름, ID를 attribute로 가진다.
- ID를 primary key로 설정한다.
- 앨범의 primary key인 ID를 '앨범 ID'라는 이름으로 foreign key로 가져온다
- 관리자의 primary key인 ID를 '관리자 ID'라는 이름으로 foreign key로 가져온다.

앨범

- 이름, 발매 날짜, ID를 attribute로 가진다.
- ID를 primary key로 설정한다.

플레이리스트

- 이름을 attribute로 가진다.
- 사용자의 primary key인 ID를 '소유자 ID'라는 이름으로 foreign key로 가져온다.
- (소유자 ID+이름)을 primary key로 설정한다.

듣기

- 들은 날짜와 들은 횟수를 attribute로 가지고 있다.
- 사용자와 음악의 M:N 관계이므로 새로운 relation으로 만들어준다.
- 사용자의 primary key인 ID와 음악의 primary key인 ID를 '사용자ID', '음악 ID'라는 이름으로 foreign key로 가져온다.

- (사용자 ID+음악 ID)를 primary key로 설정한다.

좋아요, 검색

- 사용자와 음악의 M:N 관계이므로 새로운 relation으로 만들어준다.
- 사용자의 primary key인 ID와 음악의 primary key인 ID를 '사용자ID', '음악 ID'라는 이름으로 foreign key로 가져온다.
- (사용자 ID+음악 ID)를 primary key로 설정한다.

공유

- 사용자와 플레이리스트 간의 M:N 관계이므로 새로운 relation으로 만들어준다.
- 사용자의 primary key인 ID와 플레이리스트의 partial key인 이름을 '사용자 ID, 플레이리스트 이름'이라는 이름으로 foreign key로 가져온다.
- (사용자 ID+플레이리스트 이름)은 primary key가 된다.

포함

- 음악과 플레이리스트 간의 M:N 관계이므르 새로운 relation으로 만들어준다.
- 음악의 primary key인 ID와 플레이리스트의 primary key인 (플레이리스트 이름+사용자 ID)를 foreign key로 가져온다.
- (음악 ID+플레이리스트 ID+사용자 ID)를 primary key로 설정한다.