1. playlist는 이름을 가지며 관리를 위한 일련번호를 가진다.
2. 한 playlist는 0개 이상의 곡을 담고 있다.
3. 각 곡(track)에는 일련번호가 부여되고, 곡명, 작곡가 등의 정보가 있다.
4. 한 playlist에 동일한 곡이 두 번 이상 담길 수 없다.
5. 한 곡은 여러 개의 playlist에 담길 수 있다.

어떤 playlist가 있고, 각 playlist에 어떤 곡이 담겨 있는지를 다음과 같은 테이블로 정리한다고 하자.

playlist 일련번호	playlist 이름	track 일련번호	track 이름	작곡가 이름
playlist_id	playlist_name	track_id	track_name	track_composer
1	발라드	1	학교종이 땡땡	미상
1	발라드	3	푸른하늘 은하수	아무개
1	발라드	5	반달	황씨아저씨
2	R&B	2	지우개	알리
2	R&B	3	푸른하늘 은하수	아무개
3	댄스	6	청산리 벽계수	황진이

1. playlist 열의 값이 unique하지 않으므로 playlist 열은 primary key가 될 수 없다.
2. track_id 열의 값도 unique하지 않으므로 track_id 열도 primary key가 될 수 없다.
3. (playlist_id, track_id) 순서쌍을 보면 모든 행에서 unique하다.
4. 이렇게 2개 이상의 열이 모여서 key가 되는 것을 composite key라고 한다.

위 테이블은 테이블 형식이며, primary key가 있으며,
셀에 값이 하나 씩만 들어 있으므로 제1정규형(1st NF, first normal form)이다.

5. 제 2 정규형(second normal form, 2nd NF)은 composite key가 있는 테이블에 대해 적용하는 개념이다.
6. playlist_id가 결정되면 playlist_name이 결정된다.
즉 playlist_name은 composite key 의 일부분인 playlist_id에 전적으로 의존한다.(함수적으로 종속한다.)
7. track_name과 track_composer도 track_id가 결정되면 자동적으로 결정된다.
즉 track_name, track_composer도 composite key의 일부부인 track_id에 전적으로 의존한다.
8. 이처럼 primary key가 아닌 열이 primary key의 일부분에 함수적으로 종속하면 제2정규형이 아니라고 한다.
9. composite key가 있는 테이블이 제2정규형이 아니면 데이터의 중복이 발생한다.
10. 이런 경우에는 테이블을 분해하면 제2정규형이 되도록 할 수 있다.
11. 함수적으로 종속하는 부분을 별도의 테이블로 독립시키면 된다.

playlist_id	playlist_name
1	발라드
2	R&B
3	댄스

track_id	track_name	track_composer
1	학교종이 땡땡	미상
3	푸른하늘 은하수	아무개
5	반달	황씨아저씨
2	지우개	알리
6	청산리 벽계수	황진이

playlist_id	track_id
1	1
1	3
1	5
2	2
2	3
3	6

12. 분해하여 얻어진 3개의 테이블은 주키에 의존하지 않는 열이 없으므로 제3정규형(3rd NF)이다.