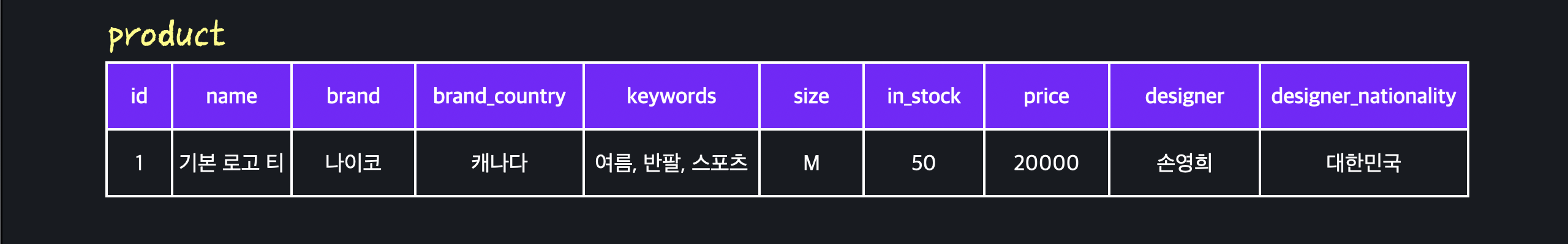
이번 레슨에서는 주어진 테이블을 정규화해보면서 이번 챕터에서 배운 내용을 정리해보겠습니다. 정규화는 단순히 이번 레슨에서 배운 내용들 뿐만 아니라, 저번 챕터에서 배운 내용들도 사용하니까요. 헷갈리시는 내용이 있다면 꼭 앞 레슨들에서 해당 개념을 복습하고 오시는 걸 추천드립니다.

온라인 쇼핑몰 코팡에서 사용하는 상품 (product) 테이블을 예시로 사용하겠습니다. 테이블 구조는 아래와 같습니다.



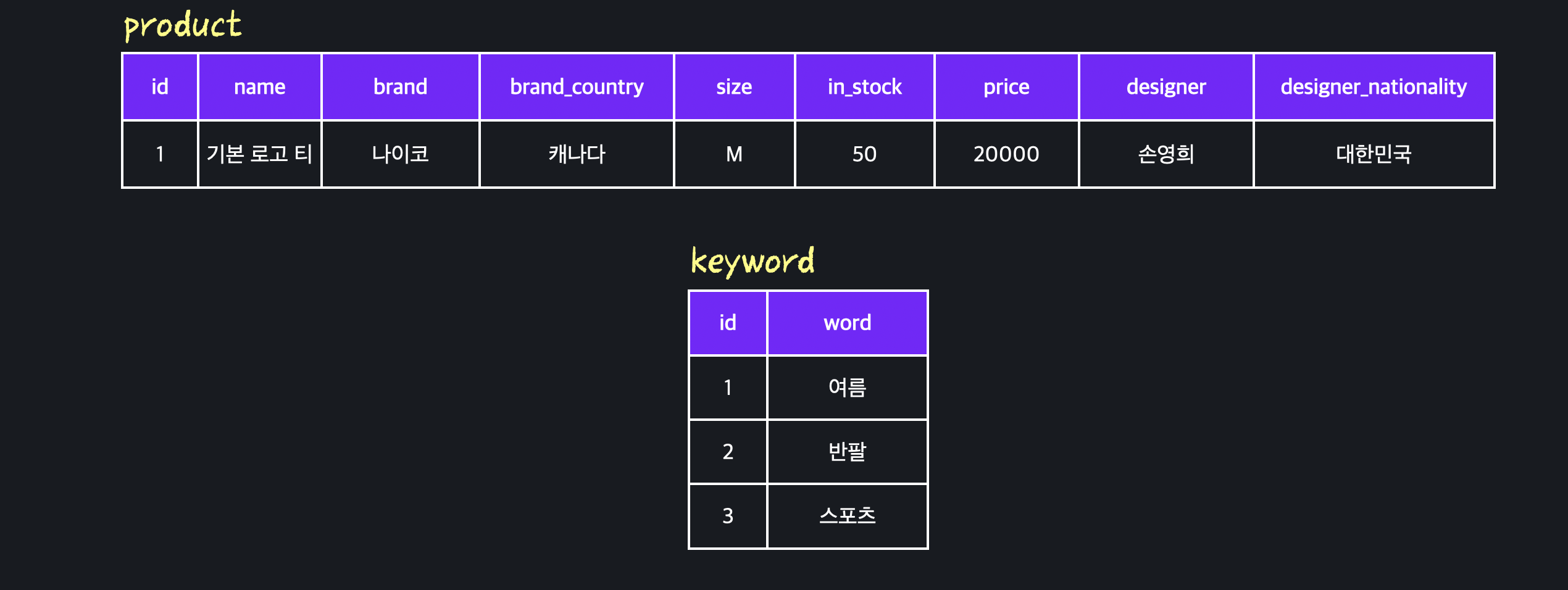
충분히 만들어낼 수 있을 거 같은 테이블인데요. 이제 이 테이블을 정규화해보겠습니다. 아래 내용을 확인하기 전에 먼저 직접 해보시는 걸 추천드립니다.

**1NF**

테이블이 1NF에 부합하기 위해서는 테이블 안에 있는 모든 값이 **나눌 수 없는 단일 값**이어야 합니다.

첫 번째 로우의 keywords 컬럼을 살펴보면 여름, 반팔, 스포츠이 저장돼있습니다. 하나가 아니라 여러 개의 키워드들이 저장되고 있잖아요? 그렇기 때문에 현재 테이블은 1NF에 부합하지 않습니다.

테이블을 1NF에 부합하게 만들기 위해서는 keywords 컬럼을 테이블로 분리해야겠죠?



그리고 그 다음에는 두 테이블 사이 관계를 모델링해야 합니다.

비즈니스 룰을 사용해서 관계의 카디널리티를 파악합니다.

하나의 제품은 여러 개의 키워드를 가질 수 있습니다. 마찬가지로 하나의 키워드는 여러 제품에서 사용될 수 있습니다.

두 테이블 사이에는 M:N (다대다) 관계가 있습니다. M:N 관계는 연결 테이블을 사용해서 모델링합니다. 연결 테이블을 추가하면 테이블들을 이렇게 만들어줄 수 있습니다.



1NF에 부합하지 않는 테이블 하나에서 1NF에 부합하는 테이블 세 개를 만들어냈습니다.

**2NF**



다음에는 테이블이 2NF에 부합하게끔 정규화하겠습니다.

현재 테이블은 1NF에 부합하고 있기 때문에 2NF에 부합하기 위해서는 candidate key의 일부분에만 의존하는 non-prime attribute은 없어야 합니다.

**candidate key 찾기**

이걸 알아내기 위해서는 candidate key와 non-prime attribute을 파악해야합니다. id는 attribute 하나만으로 모든 상품을 특정지을 수 있기 때문에 (일부분이란 개념이 없기 때문에) 생각하지 않고요, 여러 attribute의 조합으로 만들어지는 candidate key에 대해서만 생각하겠습니다.

이 테이블에서는 상품 이름 name, 브랜드 이름 brand, 그리고 크기 size, 이 세 attribute이 있으면 하나의 상품을 특정 지을 수 있습니다. 다르게 표현하면 {name, brand, size} 이게 하나의 candidate key가 되는 거죠.

그리고 candidate key에 포함되지 않는 다른 모든 attribute는 ({brand\_country, in\_stock, price, designer, designer\_country}) 모두 non-prime attribute입니다.

**non-prime attribute들의 함수 종속성 찾기**

그럼 테이블이 2NF에 부합하기 위해서는 각각 name, brand, size attribute들 각각, 또는 여기서 두 개의 attribute의 조합에 대한 함수 종속성이 있는 non-prime attribute이 있으면 안 됩니다.

각 non-prime attribute의 함수 종속성을 확인해봅시다. 이렇게 정리할 수 있을 거 같은데요.

brand \rightarrow brand\\_country*brand*→*brand*\_*country*

\{name, brand\} \rightarrow \{price, designer, designer\\_country\}{*name*,*brand*}→{*price*,*designer*,*designer*\_*country*}

\{name, brand, size\} \rightarrow \{in\\_stock\}{*name*,*brand*,*size*}→{*in*\_*stock*}

브랜드 국가는 브랜드에 대해서,

가격, 디자이너, 디자이너 국적은 제품 이름과 브랜드에 대해서,

재고는 제품 이름, 브랜드, 크기에 대해서 함수 종속성이 있죠.

**테이블 분리하기**

2NF에 부합하게 만들기 위해서는 product 테이블을, 모든 non-prime attribute이 candidate key의 전체에만 함수 종속성이 있는 세 개의 테이블로 분리해야 합니다.



그리고 이 테이블들 사이 관계를 모델링합니다.

비즈니스 룰을 사용해서 관계의 카디널리티를 파악합니다.

콜라보 제품은 고려하지 않고 상품은 딱 하나의 브랜드만 가질 수 있습니다. 반면 브랜드는 수많은 상품을 가질 수 있죠.

brand와 product 사이에는 1:N 관계가 있습니다. 다수에 해당하는 product 테이블에 foreign key를 넣어줘서 관계를 모델링 합니다.

상품은 여러 개의 크기와 제고를 가질 수 있습니다. 특정 크기와 제고는 항상 하나의 상품에만 대한 내용입니다.

product와 product\_size\_stock은 1:N 관계가 있습니다. 다수에 해당하는 product\_size\_stock 테이블에 foreign key를 넣어줘서 관계를 모델링합니다.



2NF에 부합하지 않는 하나의 테이블을 2NF에 부합하는 세 개의 테이블로 나눠줬습니다.

**3NF**



다음에는 테이블이 3NF에 부합하게끔 정규화하겠습니다. 이미 2NF에 부합하는 테이블이 3NF에 부합하기 위해서는 테이블 안 모든 attribute은 primary key에 대해 직접적인 함수 종속성이 있어야합니다. 이행적 함수 종속성도 있으면 안 되죠..

**이행적 함수 종속성 파악**

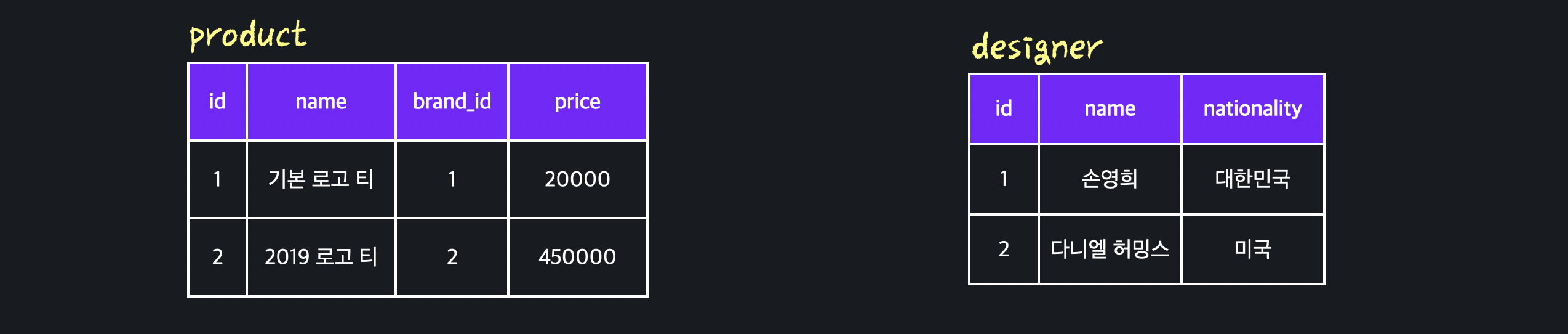
테이블을 살펴보면 name, brand\_id, price, designer, 모두 한 상품에 대한 내용입니다. 직접적으로 id에 대한 함수 종속성이 있습니다. 하지만 디자이너 국적은 직접적으로는 어떤 상품인지가 아니라, 어떤 디자이너인지에 따라 결정되는 값입니다. 함수 종속성이 이행적인 거죠.

id \rightarrow designer \rightarrow designer\\_nationality*id*→*designer*→*designer*\_*nationality*

3NF를 어기고 있습니다.

**테이블 분리**

이걸 해결하기 위해서는 designer에 대한 컬럼들을 따로 테이블로 분리해야 합니다.

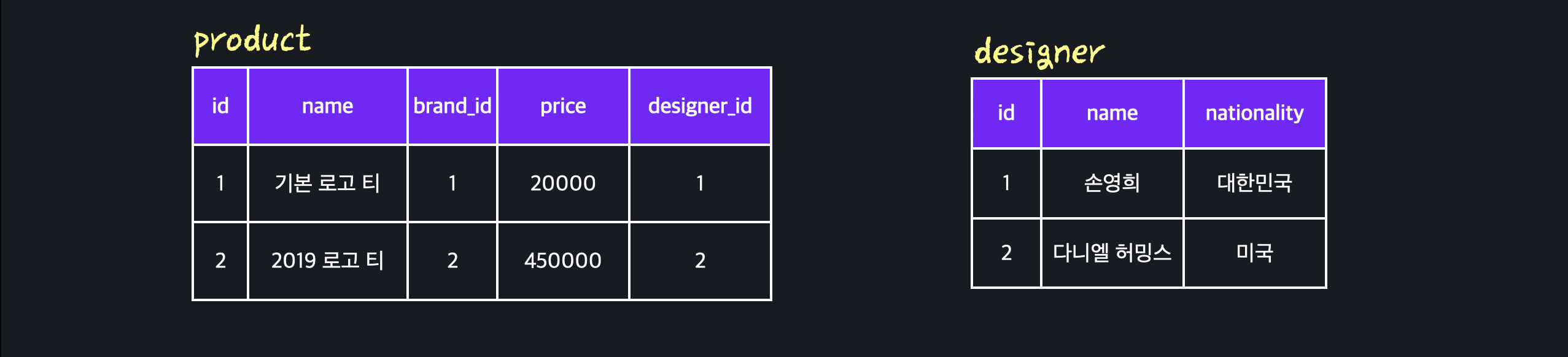


그리고 이 테이블들 사이 관계를 모델링합니다.

비즈니스 룰을 사용해서 관계의 카디널리티를 파악합니다.

하나의 상품은 하나의 디자이너만 있을 수 있고, 한 명의 디자이너는 여러 제품을 디자인할 수 있습니다.

디자이너와 상품은 1:N 관계가 있습니다. 다수에 해당하는 product 테이블에 foreign key를 넣어줘서 관계를 모델링합니다.

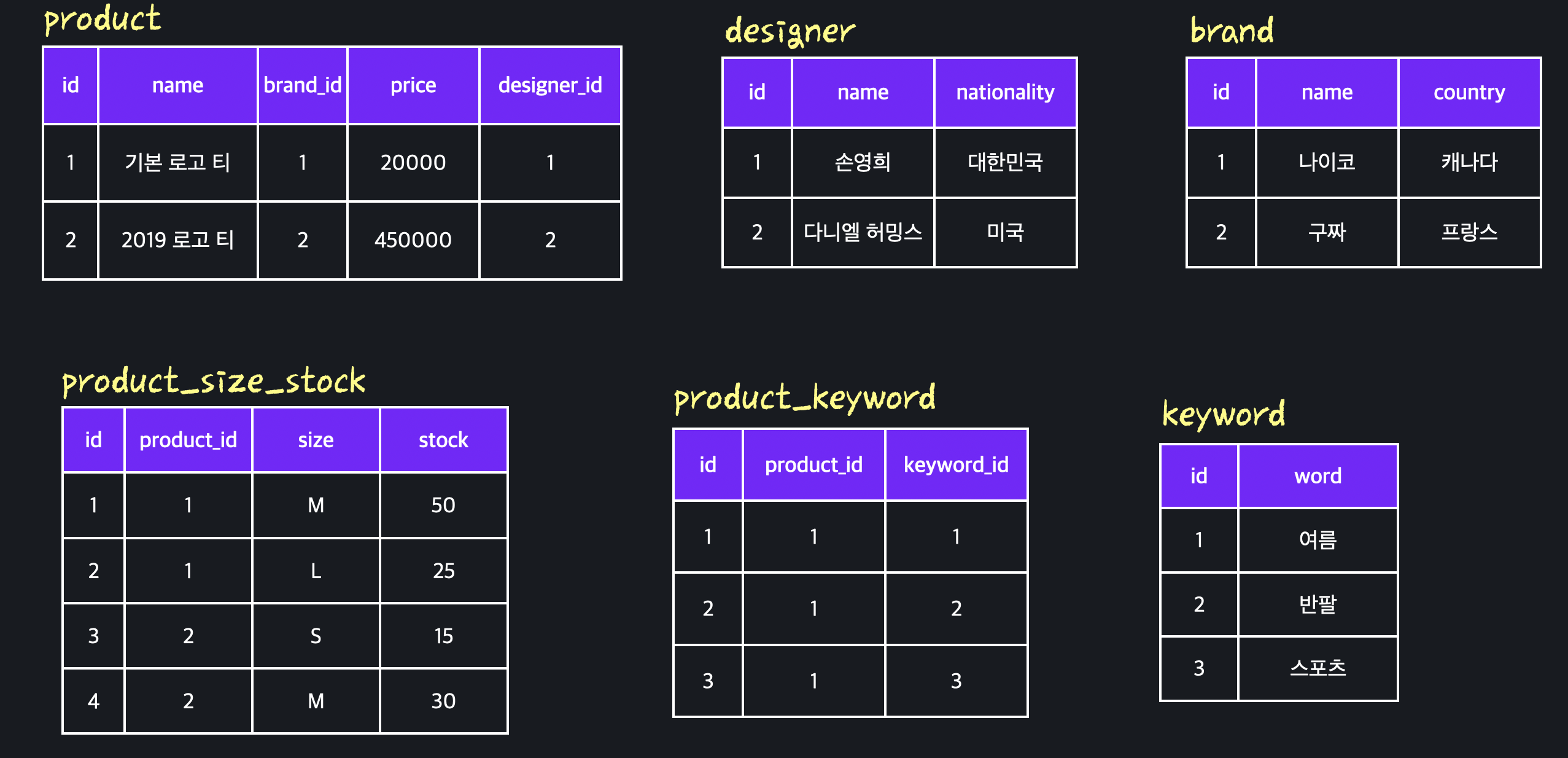


**정리**



이번 레슨에서는 정규화되지 않은 위에 있는 테이블을 3NF에 부합하게끔 바꾸는 과정을 살펴봤습니다.

정규화 되지 않은 하나의 테이블을 3NF에 부합하는 6개의 테이블로 나눠줬죠.



3NF까지 정규화를 하고 나면, 중복되는 데이터 등 모델링 실수로 생기는 데이터베이스 이상 현상을 웬만해서는 다 예방할 수 있습니다.

일반적으로 3NF에 부합하는 데이터베이스를 보고 "정규화된 데이터베이스"라고 부릅니다.

더 높은 정규형들은 데이터베이스를 학문적으로 공부하는 학자, 또는 연구원들 이외에는 크게 관심을 갖지 않는데요. 저희도 이번 챕터에서는 여기까지만 알아보도록 하겠습니다.