학습목표

1. 중복정보 제거
2. 정규형
3. 참조무결성

학습내용

- 중복정보의 제거에 대해 알 수 있습니다.
- 정규형에 대해 알 수 있습니다.
- 참조무결성에 대해 알 수 있습니다.

사전퀴즈

1. 정규화가 필요한 이유는 동일한 정보가 여러군데 나오는 것을 막고 데이터의 무결성(Integrity)를 유지하기 위해서다.
정답 : O
해설 : 정규화를 통해서 데이터의 무결성을 유지하고 중복정보를 최소화한다.

2. 제 1정규형은 테이블의 컬럼들이 기본키와 직접 연관된다는 것을 말한다.
정답 : X
해설 : 테이블의 컬럼들이 기본 키와 직접적으로 연관되는 것을 보장하는 정규형은 제2정규형이다.

수업

1. 중복정보제거

- 테이블 간의 정보는 중복되지 않아야 함
  · 동일한 정보가 여러 군데 테이블에 저장되어 있으면 수정에 대한 부담과 무결성(Integrity)유지가 쉽지 않다.
  · 하나의 정보는 한 군데만 나오도록 한다.
- 이를 위해 정규화(Normalization)을 통해 중복성 제거 (보통 DB modeling 과정에서 함)
  · 제 1정규형, 제2정규형, 제3정규형 ...
- 중복성 제거 후 필요한 정보는 외래키를 통한 조인(JOIN)을 통해 필요한 정보를 구한다.

예제 1-1
city 테이블의 국가코드 외에 국가명(CountryName)을 추가할 경우 생길 문제에 대해 말하시오
→ city 테이블의 국가명(CountryName)테이블을 변경하려면, 해당 값을 country 테이블의 name 컬럼에서 찾아 모두 변경해야함
* foreign key도 중복정보 아닌지?
foreign key는 두 개 테이블을 연결시키기 위한 필수적인 제약조건임
그 외의 정보들은 중복이 되면 안됨

```
1 alter table city add column CountryName varchar(255);
2
3 select *
4 from city
5 ;
6
```

city (6×4,079)

ID	Name	CountryCode	District	Population	CountryName
1	Kabul	AFG	Kabol	1,780,000	(NULL)
2	Qandahar	AFG	Qandahar	237,500	(NULL)
3	Herat	AFG	Herat	186,800	(NULL)

```
7 select *
8 from country
9 ;
```

country (15×239)

Code	Name	Continent	Region
ABW	Aruba	North America	Caribbean
AFG	Afghanistan	Asia	Southern and Central Asia
AGO	Angola	Africa	Central Africa

2. 정규형

중복을 제거하기 위한 테이블 정의 규칙

제1정규형	나눌 수 있을 만큼 쪼개라
제2정규형	테이블의 컬럼들이 기본키와 직접 연관되는 컬럼만으로 구성하라
제3정규형	기본키 외에 다른 컬럼들 간의 종속관계가 있으면 안됨

제1정규형

나눌 수 있을 만큼 쪼개라

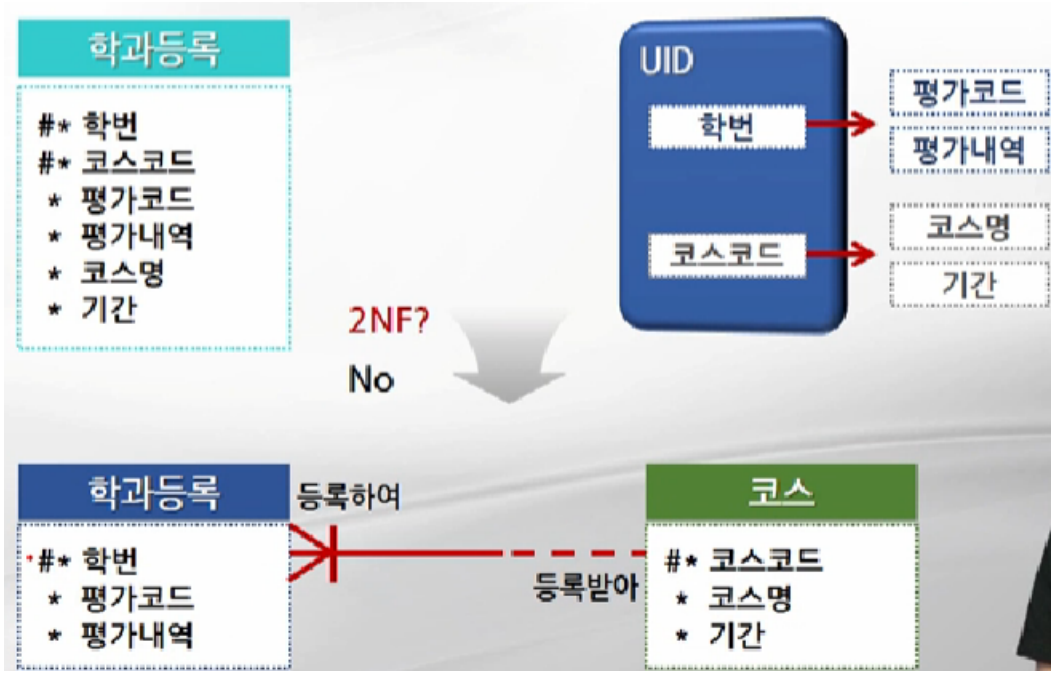
단, 관례적으로 전화번호, 이름, 주소는 하나의 컬럼을 사용한다.

기존 고객테이블에서 고객번호(PK) 는 고객명, 계약일과 1:1관계이다. 즉, 한 명의 고객은 하나의 계약만을 할 수 있다는 것을 전제하고 있다. 하지만 한 명의 고객이 여러 번의 계약(보험, 물건 주문 등)을 할 수 있으므로 고객테이블을 여러 테이블로 나누는게 좋다.

제2정규형

테이블의 컬럼들이 기본키와 직접 연관되는 컬럼만으로 구성하라

학과등록 테이블에서 학번(PK)은 평가코드, 평가내역 컬럼들 하고만 직접적인 연관이 있고, 코스코드(PK)는 코스명, 기간 컬럼들 하고만 직접적인 연관이 있으므로 학번(PK)과 연관있는 테이블과 코스코드(PK)와 연관있는 테이블 2개로 나누는 것이 좋다.

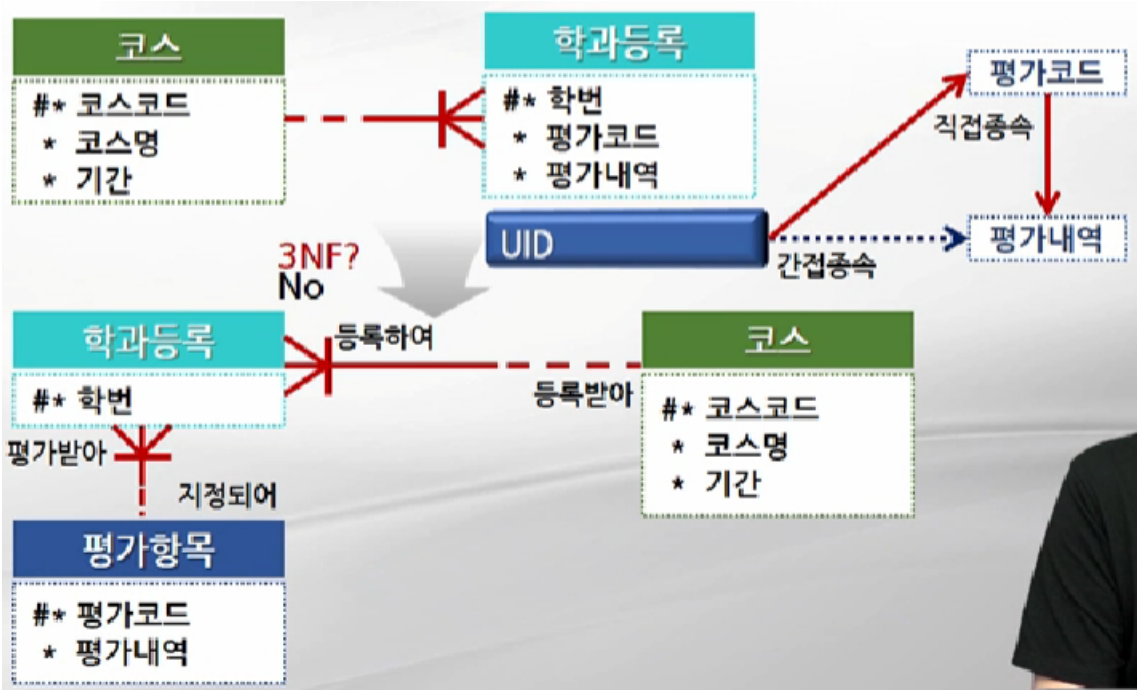


제3정규형

컬럼들 간의 종속관계가 있으면 안 됨

평가코드: A+, A ...

평가내역: 평가코드에 대한 상세설명 (뛰어남, 잘했음 ...)



3. 참조무결성 (Referential Integrity)

- 외래키(FK)에 적용되는 규칙
- 외래키와 외래키가 참조하는 원래 테이블의 키 사이의 관계를 명시
- 외래키가 참조하는 원래 테이블에 해당 레코드 값이 **반드시** 존재해야 함
- 만약 원래 레코드를 삭제하려면 참조하는 외래키(FK)값을 먼저 NULL로 만들어야 함
- 외래키 참조관계가 있을 경우에 레코드 추가/삭제시 선후관계를 나타냄

예제 3-1

city 테이블(CountryCode)과 country 테이블(Code)과의 관계를 이용해 새로운 국가코드 'ZZZ'를 country에 추가하고 도시 'YYY'를 city에 추가/삭제하라

```
insert into country(code, name)
values ('ZZZ', 'ZZZ')
;

insert into city(name, CountryCode)
values('YYY', 'ZZZ')
;

delete
from city
where name = 'YYY'
and CountryCode = 'ZZZ'
;

delete
from country
where code = 'ZZZ'
and name = 'ZZZ'
;
```

*** 참고 ***

desc [테이블명]; 명령어로는 외래키 제약조건을 알 수 없다.

그래서 show create table [테이블명][₩G]; 명령어로 실제 테이블을 생성했을 때의 제약조건을 확인한다.

```
MariaDB [world]> desc city;
+-------------+-------------+------+-----+---------+----------------+
| Field       | Type        | Null | Key | Default | Extra          |
+-------------+-------------+------+-----+---------+----------------+
| ID          | int(11)     | NO   | PRI | NULL    | auto_increment |
| Name        | char(35)    | NO   |     |         |                |
| CountryCode | char(3)     | NO   | MUL |         |                |
| District    | char(20)    | NO   |     |         |                |
| Population  | int(11)     | NO   |     | 0       |                |
| CountryName | varchar(255)| YES  |     | NULL    |                |
+-------------+-------------+------+-----+---------+----------------+
6 rows in set (0.021 sec)

MariaDB [world]> show create table city₩G;
*************************** 1. row ***************************
       Table: city
Create Table: CREATE TABLE `city` (
  `ID` int(11) NOT NULL AUTO_INCREMENT,
  `Name` char(35) NOT NULL DEFAULT '',
  `CountryCode` char(3) NOT NULL DEFAULT '',
  `District` char(20) NOT NULL DEFAULT '',
  `Population` int(11) NOT NULL DEFAULT 0,
  `CountryName` varchar(255) DEFAULT NULL,
  PRIMARY KEY (`ID`),
  KEY `CountryCode` (`CountryCode`),
  CONSTRAINT `city_ibfk_1` FOREIGN KEY (`CountryCode`) REFERENCES `country` (`Code`)
) ENGINE=InnoDB AUTO_INCREMENT=20002 DEFAULT CHARSET=latin1
1 row in set (0.000 sec)
```

데이터를 추가/삭제할 때 참조무결성을 체크하는 방법

전문가의견

데이터를 추가하기 전에 미리 외래 키 컬럼의 값을 원래 테이블에서 검색해 존재 여부를 확인한 후 존재할 경우에 레코드를 삽입한다.

데이터를 삭제할 경우 외부에서 해당 레코드를 참조(reference)하는 테이블의 컬럼이 존재하는지 확인한 후 참조하는 경우가 없다면 레코드를 삭제한다.