[IT Essential] Project DB 설계

- 진행: 최호근 컨설턴트
- 날짜: 2020.10.14
- 목차

1. Project DB 설계

> 단계별로 이해하고 설계 하자

1. DB 설계 목적

- 관련 조직의 정보 요구에 대한 정확한 이해
- 분석자, 개발자, 사용자간의 원활한 의사소통 수단
- 데이터 중심의 분석 방법
- 현행 시스템만이 아닌 신규 시스템 개발의 기초 제공
- 설계를 대충하면 기능 한 개 추가 될 때 마다 DB를 뜯어 고쳐야 한다.
    - 마지막 한 번 더 고민하고 DB 를 설계하시기를 당부 드립니다!

2. 설계를 위한 요구사항 분석

- 데이터베이스에 대한 사용자의 요구사항을 수집하고 분석해서 아래와 같이 요구사항(기능) 명세서를 작성합니다.

<예제>

회원으로 가입하려면 아이디, 비밀번호, 성명, 신용카드 정보를 입력해야 한다
회원의 신용카드 정보는 여러 개를 저장할 수 있다 .
신용카드번호, 유효기간을 저장할 수 있다.
회사가 보유한 비행기에 대해 비행기 번호, 출발 날짜, 출발 시간 정보를 저장하고 있다.
비행기 좌석에 대한 좌석 번호, 등급 정보를 저장하고 있다.
회원은 좌석을 예약하는데, 회원 한 명은 좌석을 하나만 예약할 수 있고, 한 좌석은 회원 한명만 예약할 수 있다.

3. 개념적 설계

- 작성한 요구사항 명세서에서 데이터베이스를 구성하는데 필요한 개체, 속성, 개체 간의 관계를 추출하여 ERD 를 생성

1. 개체와 속성을 추출한다.
   - 대부분 명사로 선별한다.
2. 개체 간의 관계를 추출한다.
   대부분 동사로 선별한다. (개체간의 관계를 나타내는 동사이여야 한다.)
   관계에 속한 속성도 있을 수 있다.
   1:1, 1:N, N:M
   필수적인 참여, 선택적인 참여

3-1. 개체와 속성 추출

- 요구사항에서 개체는 대부분 명사로 이루어져 있지만, 속성과 구별할 필요가 있다.
- 빨간색이 개체, 파란색이 속성

<예제>

회원으로 가입하려면 아이디, 비밀번호, 성명, 신용카드 정보를 입력해야 한다
회원의 신용카드 정보는 여러 개를 저장할 수 있다 .
신용카드번호, 유효기간을 저장할 수 있다.
회사가 보유한 비행기에 대해 비행기 번호, 출발 날짜, 출발 시간 정보를 저장하고 있다.
비행기 좌석에 대한 좌석 번호, 등급 정보를 저장하고 있다.
회원은 좌석을 예약하는데, 회원 한 명은 좌석을 하나만 예약할 수 있고, 한 좌석은 회원 한명만 예약할 수 있다.

3-2. 개체 간의 관계 추출

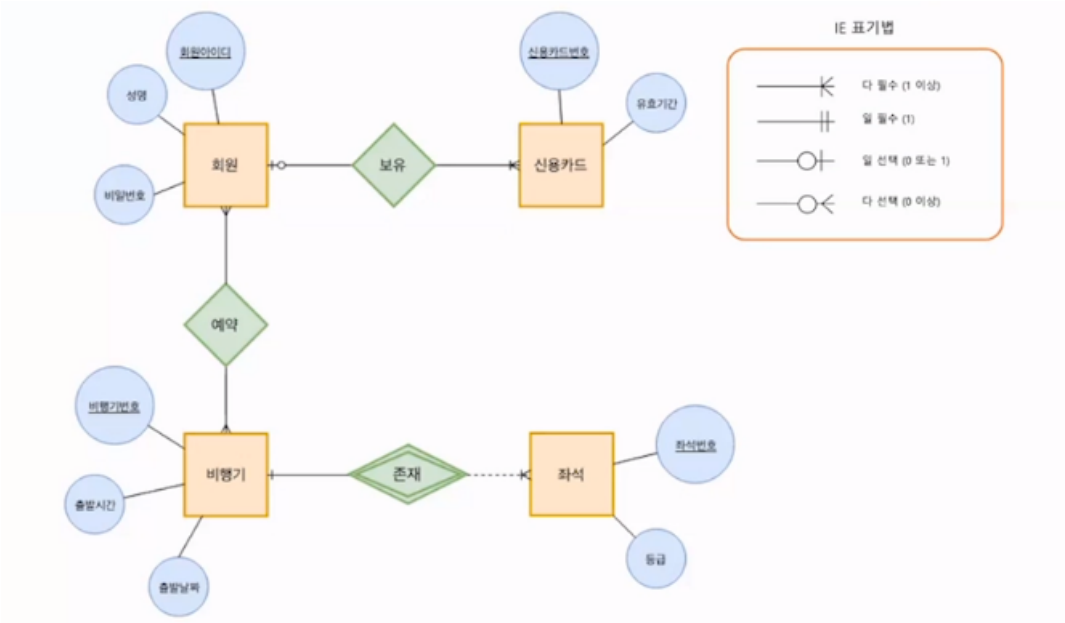
- 개체 간의 관계는 여러가지로 분류해서 정의 된다.

<예제>

회원으로 가입하려면 아이디, 비밀번호, 성명, 신용카드 정보를 입력해야 한다.
회원의 신용카드 정보는 여러 개를 저장 할 수 있다 .
신용카드번호, 유효기간을 저장할 수 있다.
회사가 보유한 비행기에 대해 비행기 번호, 출발 날짜, 출발 시간 정보를 저장하고 있다.
비행기 좌석에 대한 좌석 번호, 등급 정보를 저장하고 있다.
회원은 좌석을 예약하는데, 회원 한 명은 좌석을 하나만 예약할 수 있고, 한 좌석은 회원 한 명만 예약할 수 있다.

3-3. 분석 내용으로 ERD 생성

- 네모가 개체, 관계가 마름모, 동그라미는 개체가 가진 속성

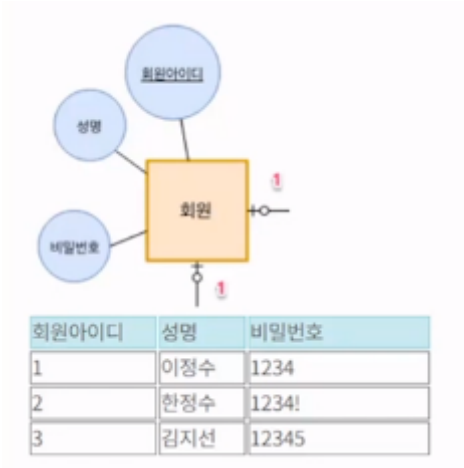


4. 논리적 설계

- 모든 개체는 릴레이션으로 변환
- N:M 관계는 릴레이션으로 변환
- 1:N 관계는 외래키로 표현
- 1:1 관계는 외래키로 표현
- 다중 값 속성은 독립 릴레이션으로 변환

4-1. 모든 개체는 릴레이션으로 변환

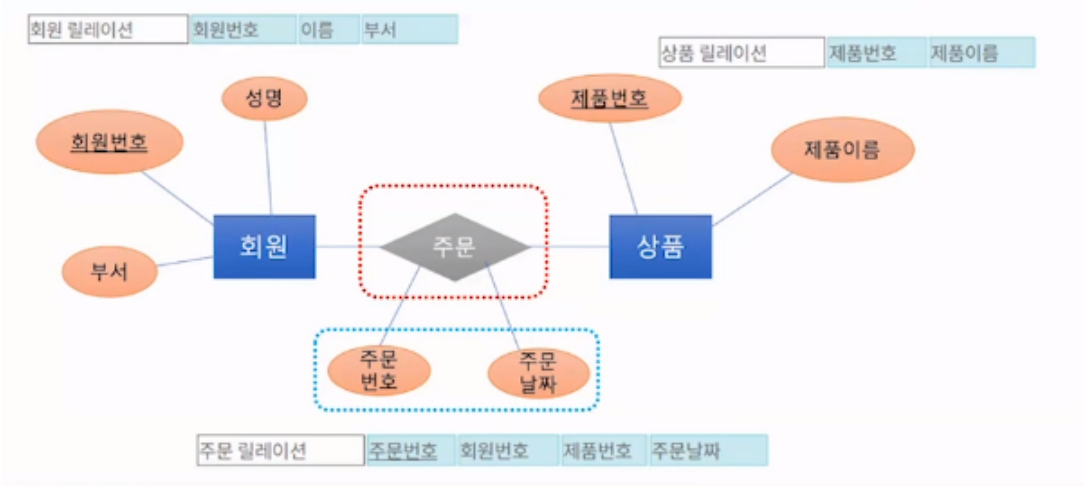
- ER 다이어그램의 각 개체를 릴레이션으로 변환
- 개체 -> 테이블
- 속성 -> 테이블의 속성



회원아이디	성명	비밀번호
1	이정수	1234
2	한정수	1234!
3	김지선	12345

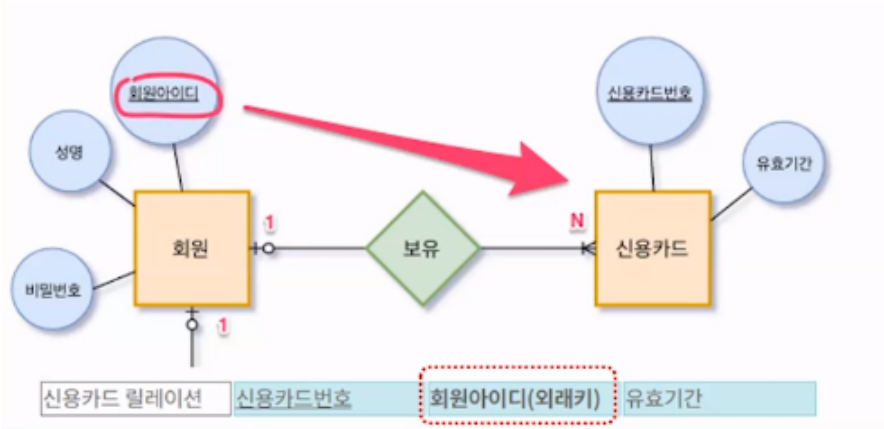
4-2. N:M 관계는 릴레이션으로 변환

- 관계 -> 릴레이션 이름
- 관계 속성 -> 릴레이션 속성



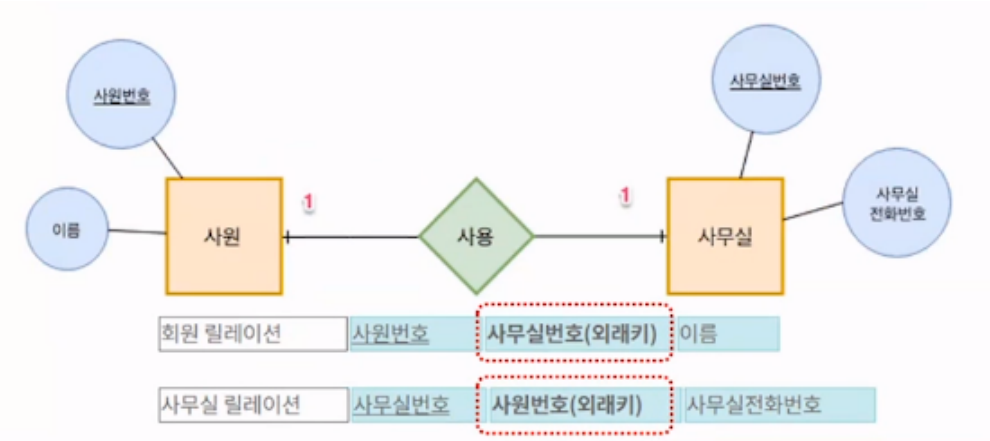
4-3. 1:N 관계는 외래키로 표현

- 일반적으로 1:N 관계에서 1측 개체의 기본키를 N측 릴레이션에 포함시키고 외래키로 지정



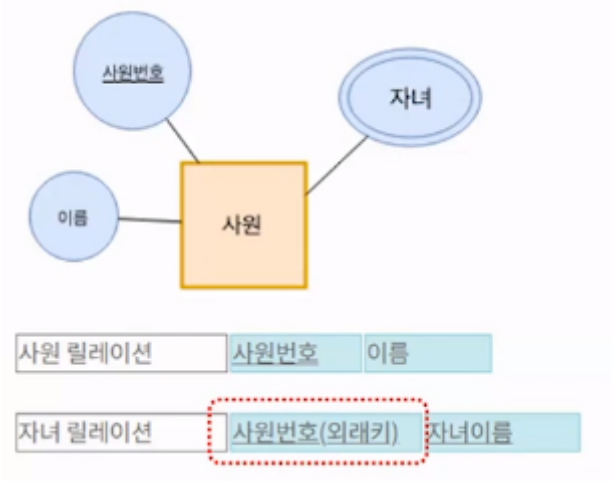
4-4. 1:1 관계는 외래키로 표현

- 일반적 1:1 관계는 외래키를 서로 주고 받는다



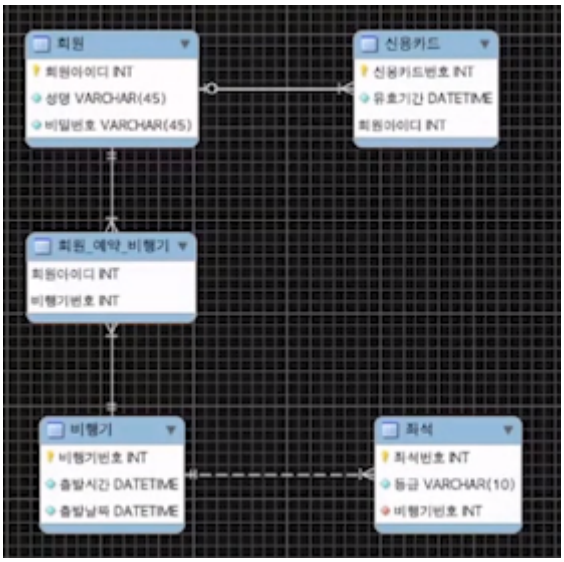
4-5. 다중값 속성은 독립 릴레이션으로

- 릴레이션에서는 다중 값 속성을 가질 수 없으므로 다중 값 속성을 별도의 릴레이션으로 생성해야 함



5. 물리적 스키마 및 구현

- ERD 를 실제 테이블로 생성한다 (Workbench 또는 SQL 스크립트 사용으로도 가능해야 함)

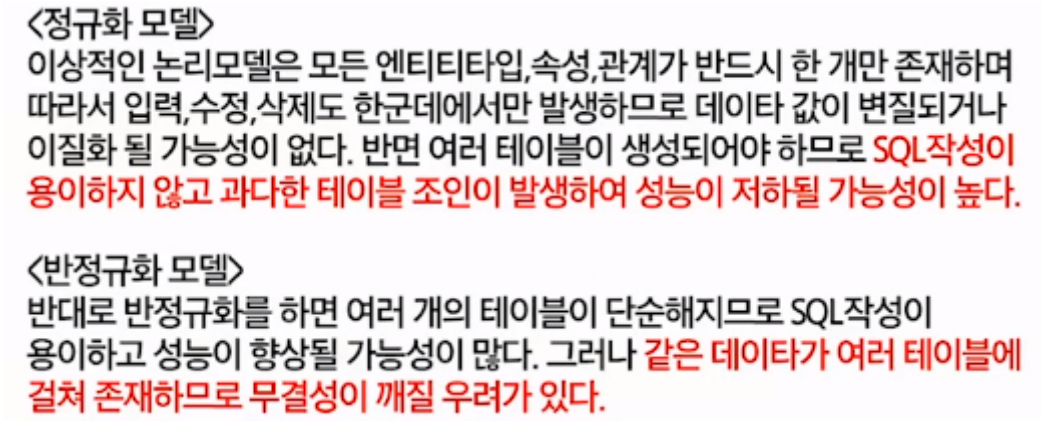


2. 반정규화

> 어떤 경우에 반정규화를 사용하는가

1. 반정규화란?

- 정규화 된 엔티티타입, 속성, 관계를 시스템의 성능 향상, 개발과 운영의 단순화를 위해 모델을 통합하는 프로세스
    - 프로젝트를 진행하며 5정규화 단계 중 보통은 3정규화 정도까지 반영하여 DB 를 설계하게 됩니다.
    - 조회를 많이 하는 항목이 포함 된 경우, 또는 속도를 중요하게 생각하는 서비스의 경우 등에서 반정규화 모델을 사용하게 됩니다.

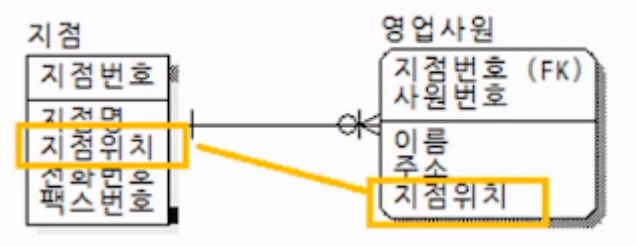


2. 테이블 반정규화

- 1:1 관계의 테이블 병합
- 1:N 관계의 테이블 병합
- 수퍼/서브 타입 테이블 병합
- 수직 분할 (집중화 된 일부 컬럼을 분리)
    - 300개 컬럼을 분리하여 나누어 저장하는 경우
- 수평 분할 (행으로 구분하여 구간별 분리)
    - 1억건의 데이터를 5천개 씩 나누어 저장하는 경우
- 테이블 추가 (중복 테이블, 통계 테이블, 이력 테이블, 부분 테이블)
    - 이력이나 통계만 보는 화면을 위해 해당 데이터를 분리하여 만드는 경우

3. 컬럼 반정규화

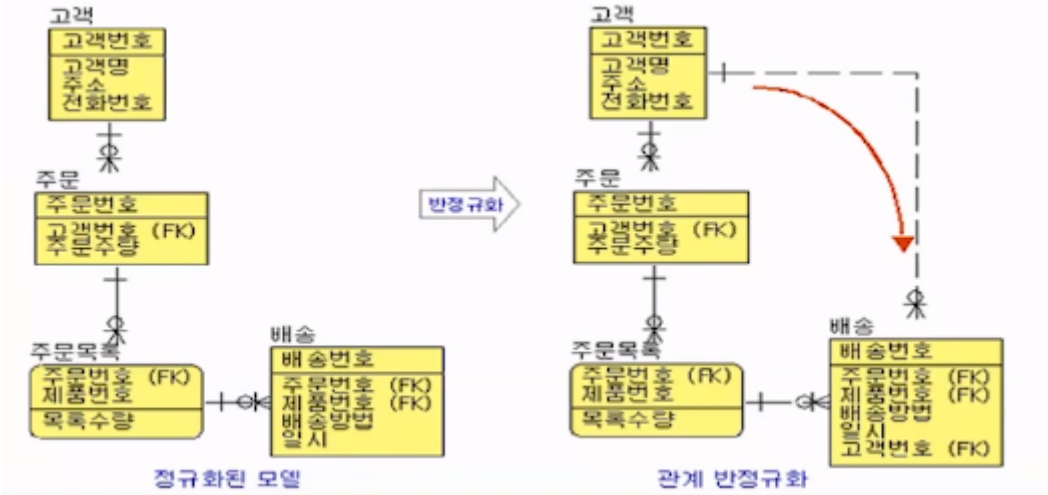
- 중복 컬럼 추가 (자주 조회하는 컬럼이 있는 경우)
    - 지점 위치를 자주 조회하는 경우에 대해 파생한 경우



- 파생 컬럼 추가 (미리 계산한 값)
- PK 에 의한 컬럼 추가
- 응용 시스템 오작동을 위한 컬럼 추가 (이전 데이터 임시 보관)

4. 관계 반정규화

- 중복 관계 추가
    - 이미 A 테이블에서 C 테이블의 정보를 읽을 수 있는 관계가 있음에도 관계를 중복하여 조회(Read) 경로를 단축
    - 고객 번호로 배송 정보를 자주 조회하기 때문에 관계를 반정규화 한 경우



5. 요약

> 설계만 잘 해도 개발이 수월합니다.
> DBA 가 DB 를 설계 할 때 5분을 더 고민하면 개발자는 1시간은 덜 고생할 수 있습니다.

두 줄 요약

1. 정규화? 반정규화? 정답은 그때그때 다르다.
2. 개념을 잘 알고 있어야 적절하게 쓸 수 있다.

3. 추가 자료

NoSQL

- 삼성SDS, NoSQL이란 무엇인가? 대량데이터 동시처리위한 DBMS 종류와 특징
- 잔재미코딩, NoSQL 이해

정의

- NoSQL이 무엇의 약자인지는 사람에 따라 No SQL, Not Only SQL, Non-Relational Operational Database SQL로 엇갈리는 의견들이 있습니다만, 현재 Not Only SQL로 풀어 설명하는 것이 다수를 차지하고 있습니다.
- 위키피디아
  - non SQL 또는 non relational
    - http://nosql-database.org/ "NoSQL DEFINITION: Next Generation Databases mostly addressing some of the points: being non-relational, distributed, open-source and horizontally scalable"
  - SQL 계열 쿼리 언어를 사용할 수 있다는 사실을 강조한다는 면에서 `Not only SQL`

관계형 데이터베이스와의 차이점

- 관계형 모델을 사용하지 않으며 테이블간의 조인 기능 없음
- 직접 프로그래밍을 하는 등의 비SQL 인터페이스를 통한 데이터 액세스
- 대부분 여러 대의 데이터베이스 서버를 묶어서(클러스터링) 하나의 데이터베이스를 구성
- 관계형 데이터베이스에서는 지원하는 Data처리 완결성(Transaction ACID 지원) 미보장
- 데이터의 스키마와 속성들을 다양하게 수용 및 동적 정의 (Schema-less)
- 데이터베이스의 중단 없는 서비스와 자동 복구 기능지원
- 다수가 Open Source로 제공
- 확장성, 가용성, 높은 성능