데이터베이스
강의 노트

제 14 회차
논리적 설계 단계 심화

❖ 학습목표

- 개체 통합의 장점에 대해 설명할 수 있다.
- 이력 데이터 발생의 3가지 유형을 나열할 수 있다.
- 코드 데이터가 무엇인지 설명할 수 있다.
- 참조 무결성의 업무 규칙을 설명할 수 있다.
- 데이터 표준화의 필요성을 설명할 수 있다.

❖ 학습내용

- 개체 통합과 이력 및 코드 데이터 모델링
- 데이터 무결성과 표준화

개체 통합과 이력 및 코드 데이터 모델링

1. 개체 타입 통합
2. 이력 데이터 모델링
3. 코드 데이터 모델링

1. 개체 타입 통합

1) 개체 타입 통합의 장·단점

장점

1. 종합적으로 정보를 조회하기 용이함
2. 불필요한 조인이 제거되어 성능이 향상됨
3. 비슷한 속성이 통합되므로 중복이 제거됨
4. ERD가 간결해 짐
5. 물리적으로 관리해야 하는 테이블 수 감소

단점

1. 업무 확장에 따른 데이터 모델의 변경이 유연하지 않음
2. 데이터 모델만으로 업무 흐름을 파악하기 어려움
3. 많은 양의 데이터가 한군데 집약되므로 성능이 저하될 수 있음
4. SQL문에서 체크해야 할 조건 증가

2) 개체 타입 통합 원칙

논리적 설계 단계	가능한 모든 개체를 상세하게 표현
물리적 설계 단계	가능한 개체 통합을 유도해서 표현
트랜잭션 통합 여부	트랜잭션이 통합해서 발생하는지 분리해서 발생하는지 사전에 조사
데이터와 트랜잭션 양이 많지 않은 경우	개체 통합 유도
데이터와 트랜잭션 양이 많은 경우	트랜잭션의 유형에 따라 개체 통합 결정

1. 개체 타입 통합

3) 개체 타입 통합 사례

동일한 기본 키를 갖는 개체 타입의 통합

통합된 개체는 이전 개체의 모든 속성을 포함

예) 부동산소유자와 부동산전세자 개체 타입을 통합해서 부동산관계자 개체 타입을 만듦

부동산소유자(주민등록번호, 소유자명, 주소, 연락처)

부동산전세자(주민등록번호, 전세자명, 주소, 연락처)

통합

부동산관계자(주민등록번호, 관계자구분, 소유자명, 주소, 연락처)

소유자인지 전세자인지 구분하기 위한 속성 추가

기본 키가 상호 식별자가 될 수 있는 개체 타입의 통합

기본 키가 비슷한 개체를 통합하고, 통합된 개체는 이전 개체의 모든 속성을 포함한다.

예) 할인대상고객과 특별고객 개체 타입을 고객 개체 타입에 통합시킨다.

고객(고객번호, 주민등록번호, 이름, 주소, 연락처, 등록일자)

할인대상고객(주민등록번호, 고객번호, 이름, 주소, 마일리지)

특별고객(주민등록번호, 이름, 주소, 등급, 등록일자)

통합

고객(고객번호, 주민등록번호(AK), 이름, 주소, 연락처, 등록일자, 할인마일리지, 특별고객등급)

대체 키 (Alternative Key)

1. 개체 타입 통합

3) 개체 타입 통합 사례

기본 키나 도메인, 속성이 비슷한 개체 타입의 통합

예) 작업요청과 작업완료 개체 타입을 통합해서 작업관리 개체 타입 생성

작업요청(작업요청번호, 요청내용, 요청일자, 작업장소)

작업완료(작업완료번호, 완료내용, 완료일자, 작업장소, 담당자)

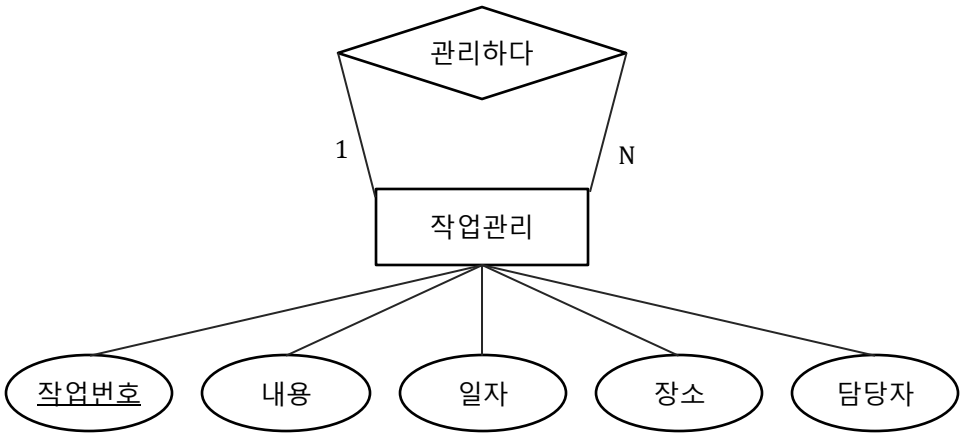
순환 관계로 통합할 수도 있고, 업무에 따라서는 원래대로 분리할 수도 있음

통합

작업관리(작업번호, 내용, 일자, 장소, 담당자, 관련작업번호(FK))

2개의 개체 타입을 순환관계로 통합한 것임

작업관리 개체의 순환관계

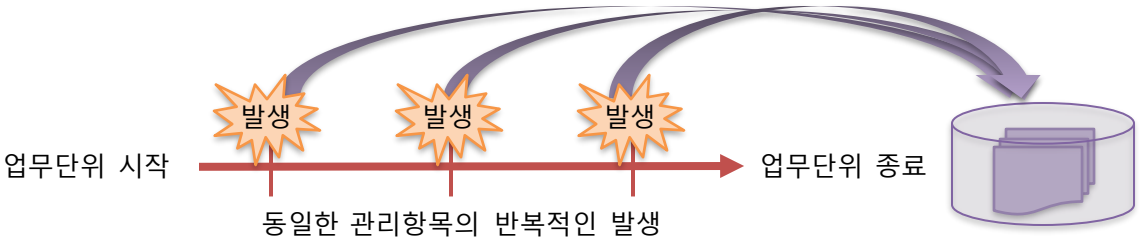


2. 이력 데이터 모델링

1) 이력 데이터(History Data)

이력 데이터(History Data)란?

하나의 업무 단위가 시간의 흐름에 따라 반복적으로 발생한 과거 및 현재 데이터



이력 데이터 모델링 대상

1. 업무적인 활동 — 예) 주문, 입고, 발주, 접수, 예약 등
2. 이력과 간접적인 관계에 있는 대상들 — 예) 고객, 상품 등
3. 최신 정보 — 예) 최신 예약, 최신 환율 등

이력 데이터 모델링의 장점

- 과거 특정 시점의 데이터를 조회할 수 있음
- 변경 내역을 관리할 수 있음
- 오류 발생 시 현재 정보를 가장 최근 이력 정보로 복구할 수 있음

대상 선정 시 고려사항

- 시간이 경과함에 따라 데이터가 변할 수 있나?
- 시간이 경과함에 따라 관계가 변할 수 있나?
- 과거 데이터를 조회할 필요가 있나?
- 변경 내역을 감사할 필요가 있나?
- 과거 버전을 보관할 필요가 있나?

이력 관리는 비용이 발생하므로 이력을 관리할 필요가 없는 데이터까지 관리하는 것은 낭비임

2. 이력 데이터 모델링

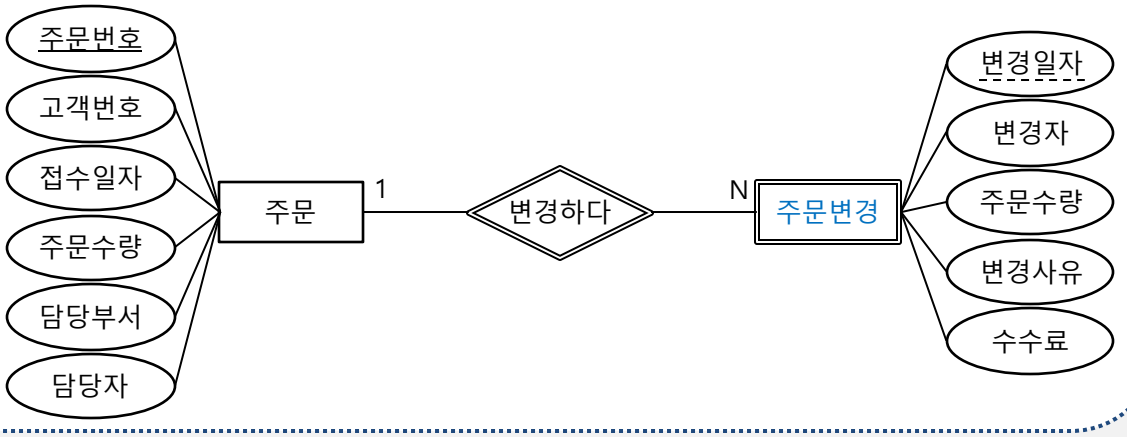
2) 이력 데이터 발생의 3가지 유형

변경 이력

데이터가 변경될 때마다 변경 전후의 차이를 확인해야 하는 경우, 약한 개체로 변경 이력을 저장
예) 주문 변경, 계약 변경, 예약 변경 등

예제

온라인 쇼핑몰에서 고객이 주문한 다음 주문 정보를 변경하는 경우, 이전 주문과 변경된 주문 정보를 관리할 필요가 있을 때 변경된 주문 이력 정보를 저장

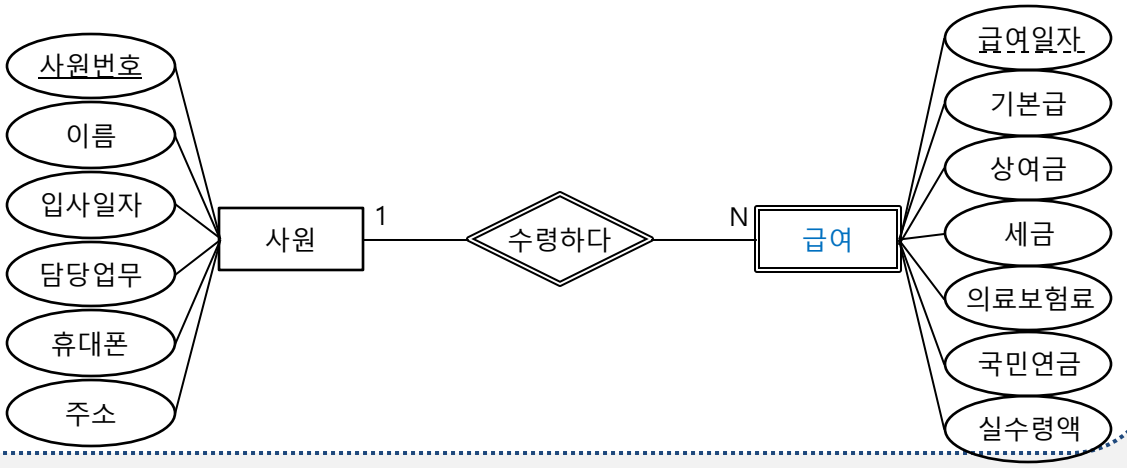


발생 이력

데이터가 발생할 때마다 이력 정보를 남겨야 하는 경우, 약한 개체로 발생 이력을 저장
예) 요금청구, 이자계산, 급여계산 등

예제

사원의 매월 급여 정보를 남겨야 하는 경우, 매월 급여를 지급할 때마다 급여 데이터를 저장



2. 이력 데이터 모델링

2) 이력 데이터 발생의 3가지 유형

진행 이력

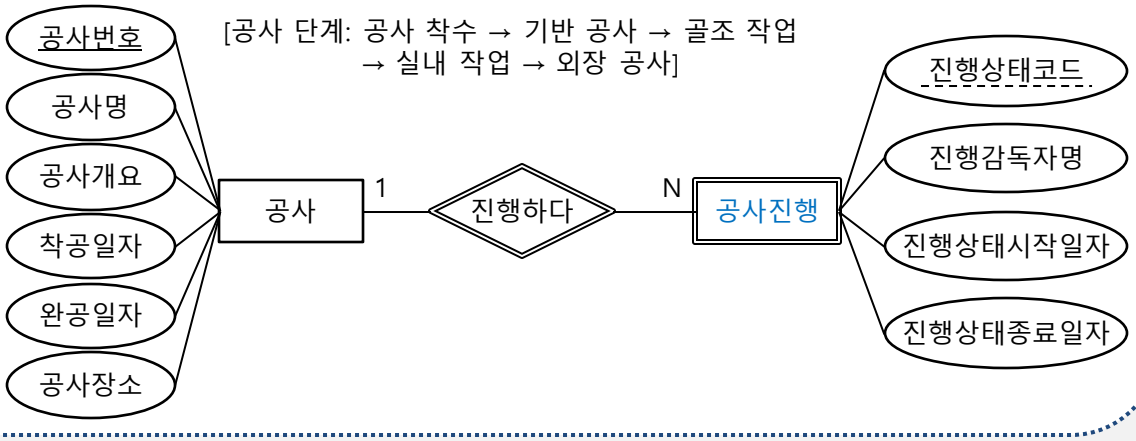
업무가 진행되는 상황을 남겨야 하는 경우 언제, 누가, 어떤 업무를 했는지 , 약한 개체로 진행 이력을 저장
예) 접수 진행, 예약 진행, 공사 진행 등

반드시 현재 단계의 진행 정보도 기록

예제

다음과 같은 단계로 공사 업무가 진행되는 경우, 각 단계별로 누가, 언제 처리했는지, 또 현재 어떤 단계인지를 기록

[공사 단계: 공사 착수 → 기반 공사 → 골조 작업 → 실내 작업 → 외장 공사]



3) 이력 개체의 특징

하나의 개체에서 발생하는 이력을 관리하기 위해서 발생하는 개체

과거의 특정 시점에 대한 정보를 제공하는 것을 목적으로 함

이력 개체(약한 개체)의 기본 키는 부분 키에 소유 개체의 기본 키를 결합해서 지정

필요하면 날짜나 일련번호를 기본 키에 추가

소유 개체의 기본 키는 이력 개체의 외래 키가 됨

하나의 이력 개체에 필요한 대부분의 속성은 소유 개체에 존재

2. 이력 데이터 모델링

4) 이력 데이터 모델링의 형태

시점 이력

▪ 데이터 변경이 발생한 시점만 관리

예) 특정 통화의 환율이 변경되면 그 시점과 환율을 저장해서 환율이 어느 시점에 얼마로 변동되었는지 저장함

▪ 가장 최근의 정보를 추출하기 위해서 MAX 함수를 사용해야 하는 번거로움이 있음

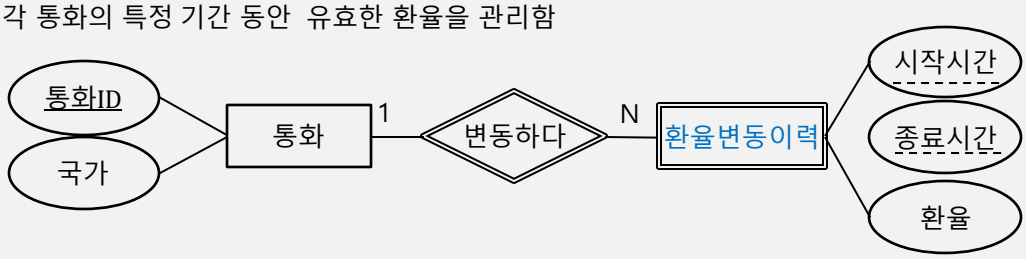
예) 미국달러의 가장 최근 환율을 검색함

```
SELECT   환율
FROM     환율변동이력
WHERE    변동시각 = (SELECT   MAX(변동시각)
                     FROM     환율변동이력
                     WHERE    통화ID = 'USD');
```

선분 이력

▪ 데이터 변경이 발생한 시작 시점부터 종료 시점까지 전체 시간을 관리

예) 각 통화의 특정 기간 동안 유효한 환율을 관리함



예) 특정 기간(2012년 6월 1일부터 12월 31일까지) 동안의 미국달러의 환율을 검색

```
SELECT    시작시간, 종료시간, 환율
FROM      환율변동이력
WHERE     시작시간  >=  '12/06/01'
AND       종료시간 <= '12/12/31'
AND       통화ID = 'USD' ;
```

2. 이력 데이터 모델링

5) 이력 데이터 모델링 시 주의사항

최신 데이터를 조회할 수 있도록 변경 이력에 '최신 여부 속성'을 추가함

최신 여부 속성이 없는 경우의 변경 이력 데이터 조회

예) 다음 주문변경 릴레이션에서 사업부코드 '100'인 사업부에서 **가장 최근에 변경된 주문번호와 주문수량을 검색하는 SQL문**

주문변경(주문번호, 변경일자, 변경자, 주문수량, 변경사유, 수수료, 사업부코드)

```
SELECT   주문변경1.주문번호,  주문변경1.주문수량
FROM     주문변경  주문변경1,
         (SELECT  주문번호,  MAX(변경일자)  변경일자
          FROM    주문변경
          WHERE   사업자코드 = '100'
          GROUP   BY  주문번호)  주문변경2
WHERE    주문변경1.주문번호 = 주문변경2.주문번호
AND      주문변경1.변경일자 = 주문변경2.변경일자;
```

최신 여부 속성이 있는 경우의 변경 이력 데이터 조회

예) 다음 주문변경 릴레이션에서 사업부코드 '100'인 사업부에서 **가장 최근에 변경된 주문번호와 주문수량을 검색하는 SQL문**

주문변경(주문번호, 변경일자, 변경자, 주문수량, 변경사유, 수수료, 사업부코드, 최신여부)

```
SELECT  주문번호,  주문수량
FROM    주문변경
WHERE   사업자코드 = '100'
AND     최신여부 = 'Y;
```

2. 이력 데이터 모델링

6) 이력 데이터 모델링 시 고려사항

1. 현재 상황뿐만 아니라 미래의 발생 가능한 상황

   예) 현재 : 사원이 현재 근무하고 있는 부서 정보만 관리
   미래 : 사원의 발령 이력(근무한 적이 있는 모든 부서 포함)을 모두 관리

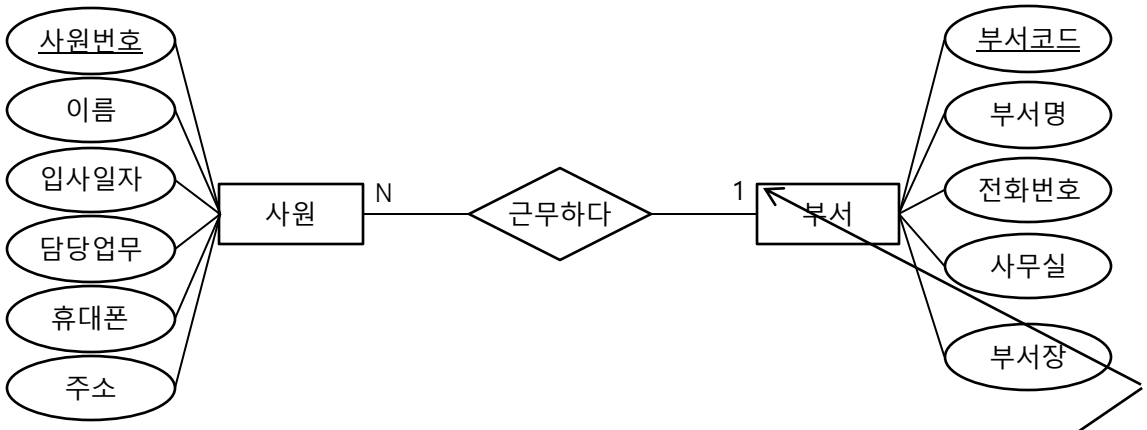
2. 이력 관리를 위해 약한 개체의 추가 또는 기본 키의 변경 여부
3. 데이터의 발생 형태
4. 이력 데이터 모델링의 주기
5. 이력 데이터 모델링의 수위 조절 (어떤 수준까지 이력 관리할 것인가)

2. 이력 데이터 모델링

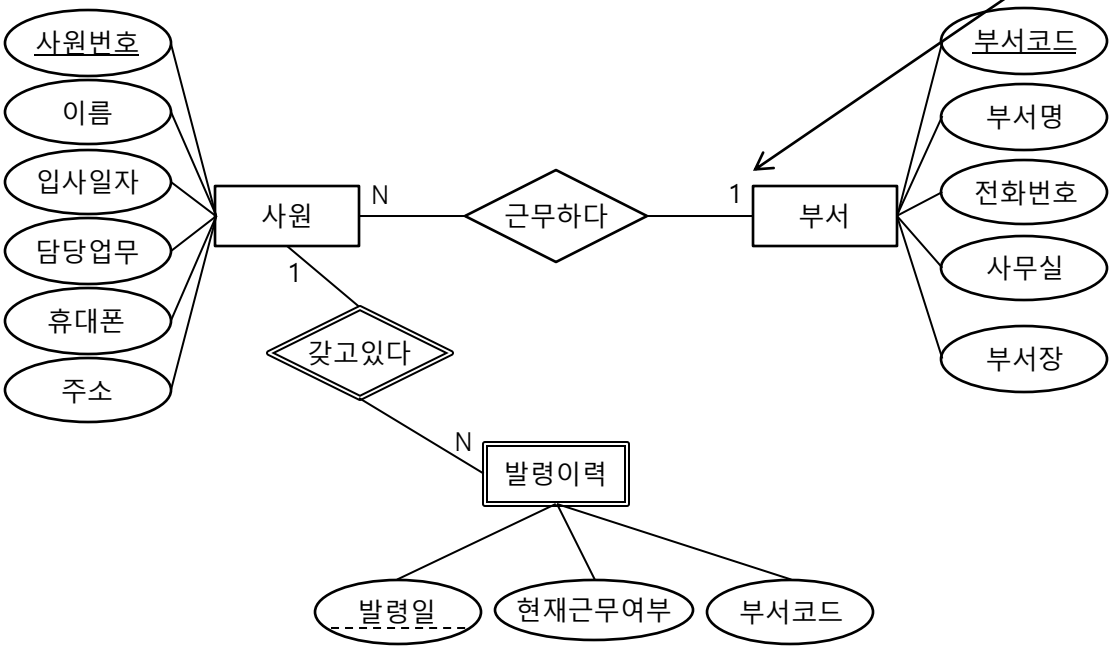
6) 이력 데이터 모델링 시 고려사항

사원관리 ER 다이어그램

- **현재** : 사원이 현재 근무하고 있는 부서 정보만 관리



- **미래** : 사원의 발령 이력(근무한 적이 있는 모든 부서 포함)을 모두 관리



3. 코드 데이터 모델링

1) 코드(Code)

코드(Code)란?

업무에서 또는 정보시스템에서 쉽게 구분할 수 있도록 데이터들을 간단하게 구분해 놓은 단위

코드 구분

한 개의 값(속성)이 반복적으로 나타나는 경우

예) 코드명이라는 **한 개의 속성을 갖는 지불방법코드**
- 현금 결재, 선불카드 결재, ..., 계좌 이체 가운데 한 개의 값만 나타남

코드 구분	코드 구분명	코드 값	코드명(의미)
CD001	지불방법코드	1	현금 결재
		2	선불카드 결재
		3	신용카드 결재
		4	휴대폰 결재
		5	계좌 이체

여러 개의 값(속성)이 반복적으로 나타나는 경우

예) 부서명 **외에 위치와 부서장이라는 3개의 속성**을 갖는 부서 코드

코드 구분	코드 구분명	코드 값	속성(의미)		
			부서명	위치	부서장
CD002	부서코드	1	기획팀	서울	김철수
		2	인사팀	서울	이영희
		3	영업팀	서울	박문수
		4	기술팀	대전	최영민
		5	AS팀	천안	장민수

3. 코드 데이터 모델링

2) 코드 데이터 모델링 사례

코드 데이터 모델링 사례

한 개의 값(속성)이 반복적으로 나타나는 코드가 여러 개 존재하는 경우의 모델링

코드구분(소유 개체)과 상세코드(약한 개체)라는 2개의 통합코드 개체를 생성해서 코드 구분을 간소화

예) 한 개의 값을 반복하는 코드가 3개(접수구분 코드, 접수방법 코드, 신청자구분 코드) 있는 경우의 모델링

코드 데이터 모델링 전 초기 ER 모델

코드 데이터 모델링

코드 데이터 모델링 후 ER 모델

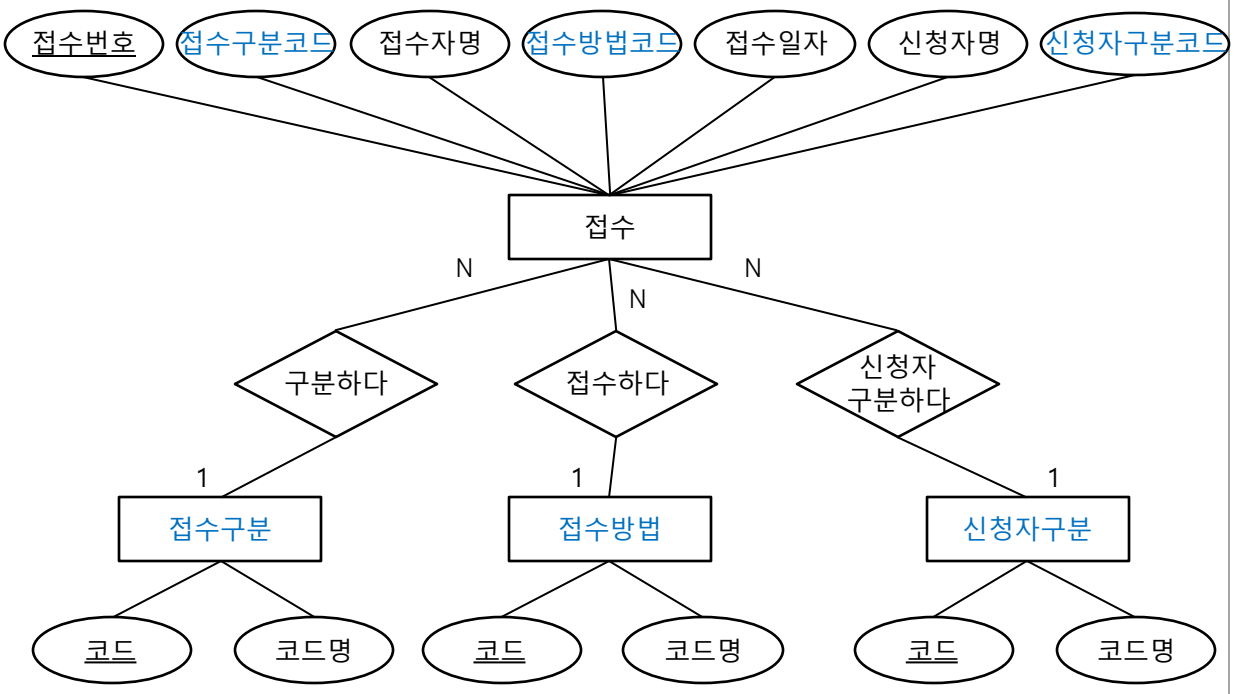
여러 개의 값(속성)이 반복적으로 나타나는 코드의 모델링

3. 코드 데이터 모델링

2) 코드 데이터 모델링 사례

코드 데이터 모델링 사례

코드 데이터 모델링 전 초기 ER 모델



[코드 테이블]

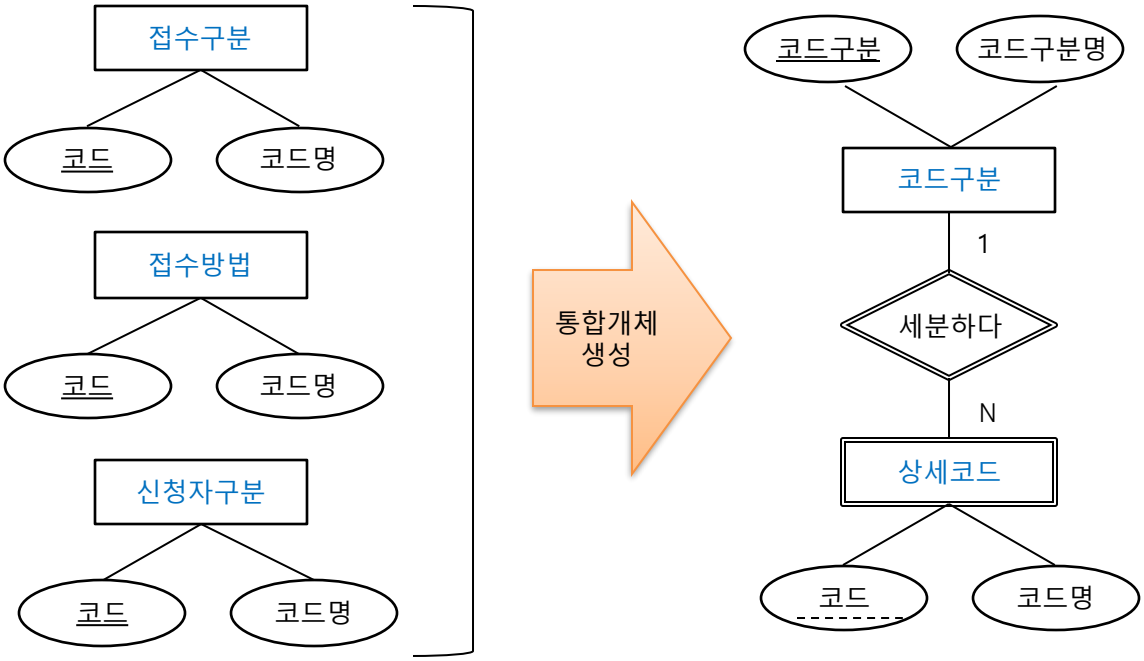
코드 개체명	코드 값	
	코드	**코드명**
접수구분	1	일반
	2	긴급
	3	특별
접수방법	1	방문접수
	2	전화접수
	3	인터넷접수
신청자구분	1	일반인
	2	법인

3. 코드 데이터 모델링

2) 코드 데이터 모델링 사례

코드 데이터 모델링 사례

코드 데이터 모델링



[코드구분 테이블]

코드 구분	코드 구분명
CD001	접수구분
CD002	접수방법
CD003	신청자구분

[상세코드 테이블]

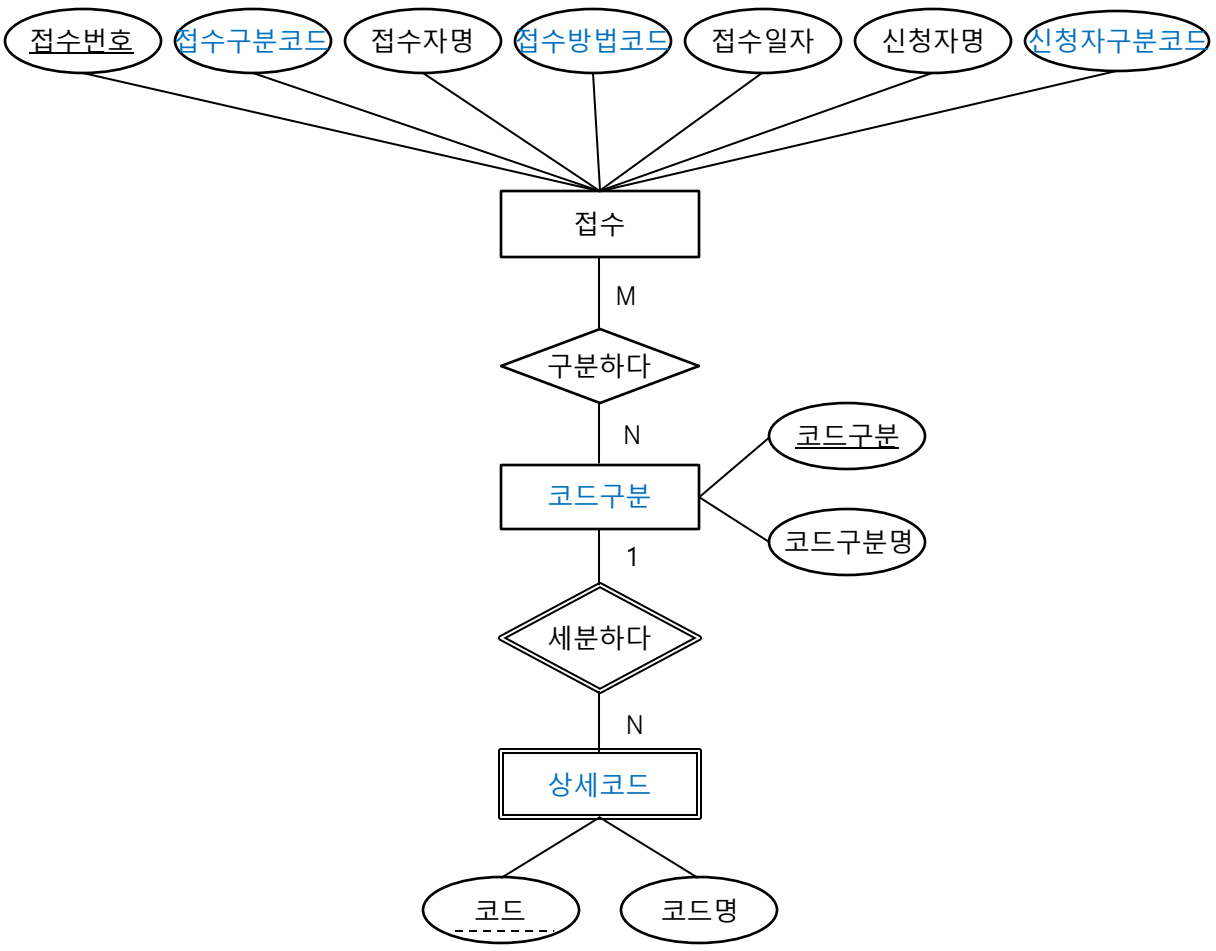
코드 구분	코드	코드명
CD001	1	일반
	2	긴급
	3	특별
CD002	1	방문접수
	2	전화접수
	3	인터넷접수
CD003	1	일반인
	2	법인

3. 코드 데이터 모델링

2) 코드 데이터 모델링 사례

코드 데이터 모델링 사례

코드 데이터 모델링 후 ER 모델



3. 코드 데이터 모델링

2) 코드 데이터 모델링 사례

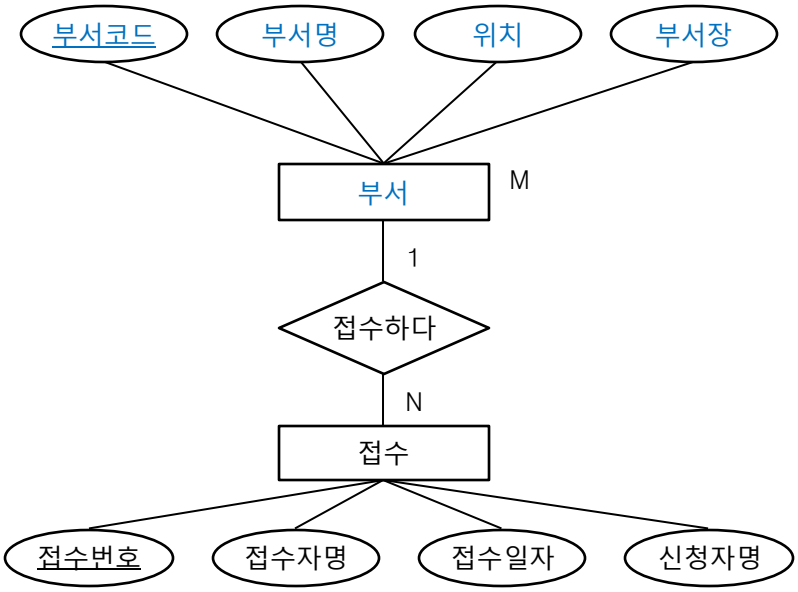
코드 데이터 모델링 사례

여러 개의 값(속성)이 반복적으로 나타나는 코드의 모델링

별도의 통합코드 개체를 생성하지 않고 **일반적인 개체와 동일하게 모델링**

예) 부서명과 위치, 부서장이라는 속성을 갖는 부서코드를 일반적인 부서 개체로 모델링

일반적인 부서 개체 모델링



3. 코드 데이터 모델링

2) 코드 데이터 모델링 사례

코드 데이터 모델링 사례

코드값 변환 시 주의사항

① 코드를 코드 값으로 변환(예: 코드 '1'을 코드 값 '방문접수'로 변환)할 때 조인을 통해서 변환하는 경우, SQL문이 복잡해지고 성능도 저하됨

② SQL문에서 DECODE를 사용해서 코드와 코드 값을 매핑하는 경우, 코드가 변경되거나 추가될 때마다 SQL 문을 수정해야 함

③ 응용 프로그램에서 코드 값을 변환하는 경우, 코드가 변경될 때마다 응용 프로그램을 수정해서 다시 컴파일해야 함

④ 코드 값을 변환하는 가장 효율적인 방법은 DBMS의 함수(FUNCTION)를 사용해서 변환하는 것임

데이터 무결성과 표준화

1. 데이터 무결성
2. 개체 무결성
3. 참조 무결성
4. 데이터 표준화

1. 데이터 무결성

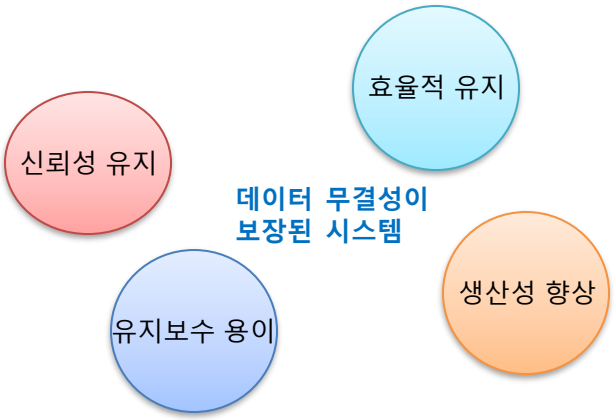
1) 데이터 무결성

데이터 무결성(Data Integrity)이란?

DB에 저장된 데이터의 정확성과 일관성을 유지하기 위해서 데이터가 항상 만족해야 하는 제약조건

데이터 무결성의 필요성

데이터들 간의 정확성, 유효성, 일관성, 신뢰성을 위해 무분별한 데이터 갱신으로부터 데이터를 보호



2) 데이터 무결성의 종류

구성	특징
개체 무결성 (Entity Integrity)	기본 키를 구성하는 속성은 반드시 값을 가져야 하고, 유일성을 보장하는 최소한의 집합이어야 한다는 제약조건
참조 무결성 (Referential Integrity)	외래 키는 반드시 피참조 릴레이션의 기본 키 값이나 널 값을 가져야 한다는 제약조건
도메인 무결성 (Domain Integrity)	속성의 데이터 타입, 길이, 디폴트 값, 널(Null) 값 허용 여부, 허용되는 값의 범위 등에 대한 제약조건

2. 개체 무결성

1) 개체 무결성 규칙

- 규칙1: 기본 키를 구성하는 속성은 널 값을 가져서는 안됨
- 규칙2: 기본 키는 개체를 유일하게 식별할 수 있어야 함
- 규칙3: 기본 키는 유일성을 보장하는 최소한의 집합이어야 함

기본 키를 정의하면 묵시적으로 정의되지만, 기본 키가 잘못 정의되면 개체 무결성이 보장되지 않음

2) 기본 키 선정 사례

	규칙 1 (Not NULL)	규칙 2 (유일성)	규칙 3 (최소성)
학번	O	O	O
주민등록번호	O	O	O
{이름, 생년월일}	O	X	O
{학번, 이름}	O	O	X
{학번, 전공코드}	X	O	X

후보 키

이름 없이 학번만으로 유일한 식별 가능

1학년은 전공이 결정되지 않아 전공코드는 널 값을 가짐

이름과 생년월일이 같은 사람이 여러 명 있을 수 있음

2. 개체 무결성

3) 기본 키 선정 시 고려사항

1. 기본 키는 개체 인스턴스(Instance)를 유일하게 식별할 수 있어야 함
2. 기본 키는 가능한 변경되지 않아야 함
3. 데이터 보안이 요구되지 않아야 함
   - 주민등록번호는 개인 정보보호 관점에서 보안이 요구되므로 기본 키로 사용하지 않도록 함
4. 기본 키는 반드시 최소성을 만족해야 함
   - 복합 속성이 기본 키가 될 수 있지만, 하나의 속성으로 유일성을 보장하는 속성이 있다면 단일 속성을 기본 키로 선정
5. 후보 키 가운데 보다 적은 저장공간을 차지하는 속성을 선택
   - 기본 키에 대한 인덱스도 자동 생성되므로 저장공간이 적은 속성을 선택
6. 후보 키 가운데 업무 활용도가 높고, 업무상 의미 있는 속성을 선택함
   - 여러 개체에서 사용되는 후보 키보다, 개체의 특성을 나타내는 속성을 선택
7. 기존 업무에서 의미 있게 사용하던 후보 키가 있으면 그대로 유지함
   - 새로운 기본 키를 설정하면 업무에 혼란을 초래할 수 있으므로 가능한 동일한 기본 키를 선정

3. 참조 무결성

1) 참조 무결성

참조 무결성이란?

외래 키(Foreign Key)는 반드시 피참조 릴레이션에 존재하고 있는 기본 키와 연결되거나 널 값을 가져야 한다는 제약조건

참조 무결성의 업무 규칙

구분	적용 시점	세부 업무 규칙
입력 규칙	자식 인스턴스가 입력되거나 외래 키가 수정될 때	- 의존(Dependent) - 지정(Customized) - 자동(Automatic) - NULL - 기본(Default) - 미지정
삭제 규칙	부모 인스턴스가 삭제될 때	- 제한(Restrict) - 연쇄(Cascade)
수정 규칙	부모 인스턴스의 관계에 대응하는 속성이 수정될 때	- 제한(Restrict) - 지정(Customized) - 연쇄(Cascade) - NULL - 기본(Default) - 미지정

3. 참조 무결성

2) 세부 업무 규칙

입력 참조 무결성의 세부 업무 규칙

구분	세부 내용
의존 (Dependent)	자식 테이블에 데이터를 입력할 때 **참조하고 있는 부모 테이블에 기본 키가 존재할 때만 데이터 입력 가능**
자동 (Automatic)	자식 테이블에 데이터를 입력할 때 **참조하고 있는 부모 테이블의 기본 키가 존재하지 않으면,** 기본 키를 생성하고 **자식 테이블에 데이터 입력**
기본 (Default)	자식 테이블에 데이터를 입력할 때 **참조하고 있는 부모 테이블의 기본 키의 값을 기본(디폴트) 값으로 바꾼 후** 자식 테이블에 입력
지정 (Customized)	사용자가 지정한 **일정한 조건을 만족하는 경우**에만 자식 테이블에 데이터를 입력할 수 있음
NULL	자식 테이블에 데이터를 입력할 때 참조하고 있는 **부모 테이블의 기본 키가 없어도 입력할 수 있고, 외래 키 값은 널이 됨**
미지정	자식 테이블에 데이터를 입력할 때 **조건 없이 허용**

삭제 및 수정 참조 무결성의 세부 업무 규칙

구분	세부 내용
제한 (Restrict)	**대응하는 자식 인스턴스가 없는 경우에만** 부모 인스턴스의 삭제 및 수정 가능
연쇄 (Cascade)	부모 인스턴스의 삭제 및 수정을 항상 허용하며, 동시에 **대응하는 자식 인스턴스도 모두 자동으로 삭제 및 수정**
기본 (Default)	부모 인스턴스의 삭제 및 수정을 항상 허용하며, 동시에 대응하는 **자식 인스턴스의 외래 키를 기본(디폴트) 값으로 수정**
지정 (Customized)	**특정한 검증 조건을 만족하는 경우**에만 부모 인스턴스의 삭제 및 수정 허용
NULL	부모 인스턴스의 삭제 및 수정을 항상 허용하며, 동시에 대응하는 **자식 인스턴스의 외래 키를 널 값으로 수정**
미지정	부모 인스턴스의 삭제 및 수정을 **조건 없이 허용**

3. 참조 무결성

2) 세부 업무 규칙

참조 무결성 정의 사례

부모 개체	관계		자식 개체	입력 규칙	삭제 규칙	수정 규칙
	유형	제약				
프로젝트	1: 1	부분 참여	계약금	Dependent	Restrict	Cascade
프로젝트	1: N	부분 참여	보고서	Dependent	Cascade	Cascade
. . .						

4. 데이터 표준화

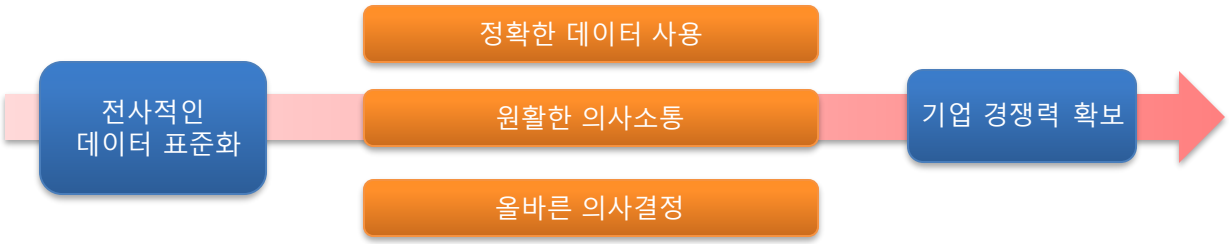
1) 데이터 표준화

데이터 표준화란?

시스템 별로 산재되어 있는 데이터의 명칭, 정의, 규칙, 형식 등에 대한 원칙을 수립해서 전사적으로 적용하는 것

데이터 표준화의 필요성

데이터가 기업의 전략적 의사소통의 핵심 요소이므로 데이터의 품질을 확보하기 위해 데이터 표준화가 필수적임



2) 데이터 비표준화의 문제점

현실적으로 데이터 표준화가 어려운 여러 가지 이유가 있는데, 그로 인해 데이터를 표준화하지 않으면 추후 훨씬 더 심각한 문제를 유발할 수 있음

데이터 표준화가 어려운 이유

① 여러 정보시스템을 동시에 개발
② 전사적인 데이터 관리에 대한 마인드 부족
③ 전사적인 데이터 관리 인력 부재
④ 전사적인 데이터 표준 관리 도구 부재

데이터 비표준화의 문제점

① 데이터의 중복 및 불일치 발생
② 데이터에 대한 의미 파악이 어려워 정보 제공의 적시성 결여
③ 데이터 통합의 어려움
④ 정보시스템 유지보수의 비효율성

4. 데이터 표준화

3) 데이터 표준화 대상

1. 표준 단어
2. 표준 용어
3. 표준 도메인
4. 표준 코드

4) 데이터 표준화의 기대효과

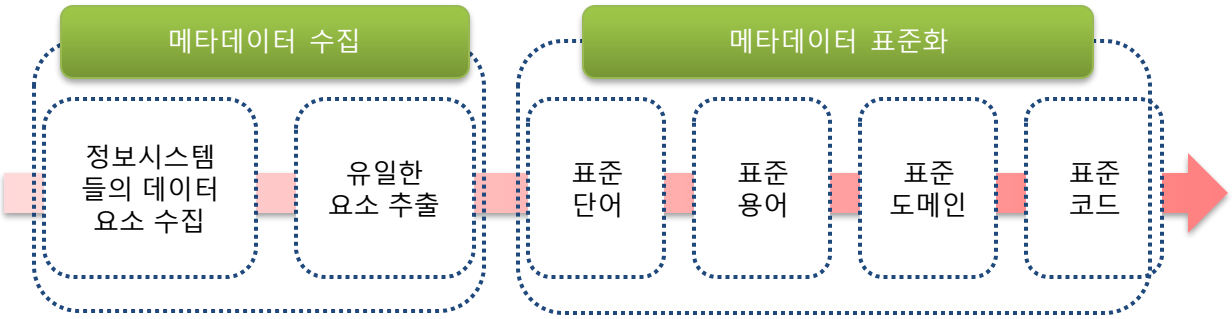
- 명칭의 통일로 인한 원활한 의사소통 가능
- 일관된 데이터 형식과 규칙의 적용으로 인한 데이터 품질 향상
- 필요한 데이터의 소재 파악에 소요되는 시간과 노력 감소
- 정보시스템이 상호 인터페이스 할 때 데이터 변환 및 정제 비용 감소

4. 데이터 표준화

5) 데이터 표준화 단계

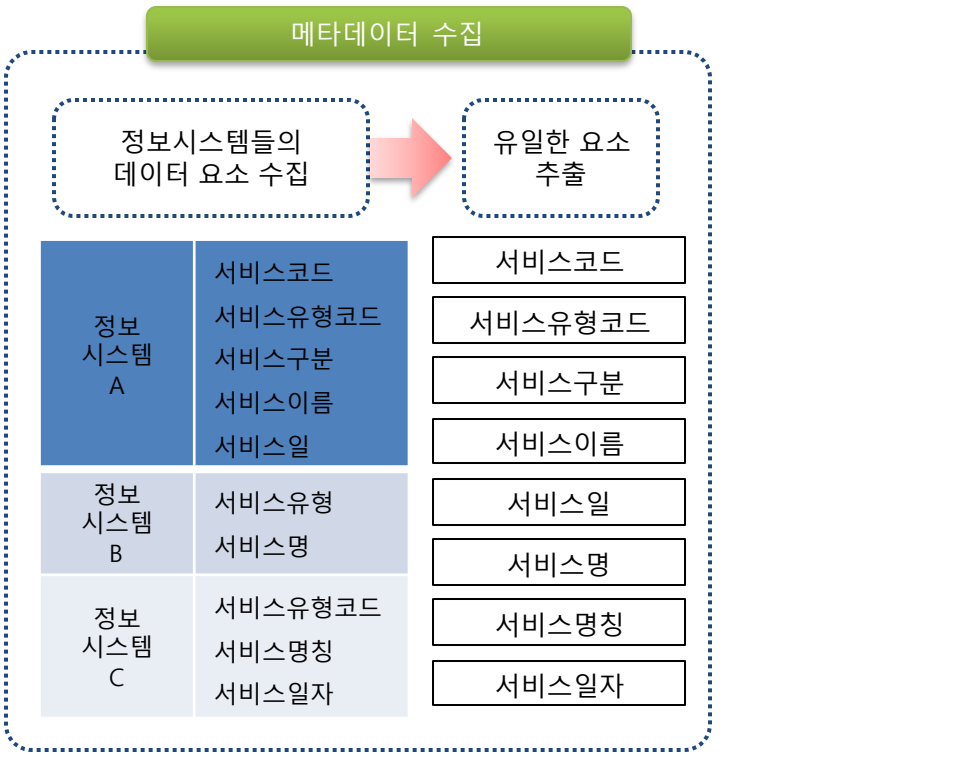
메타데이터(Meta data)*를 수집한 다음, 단계별로 메타데이터를 표준화함

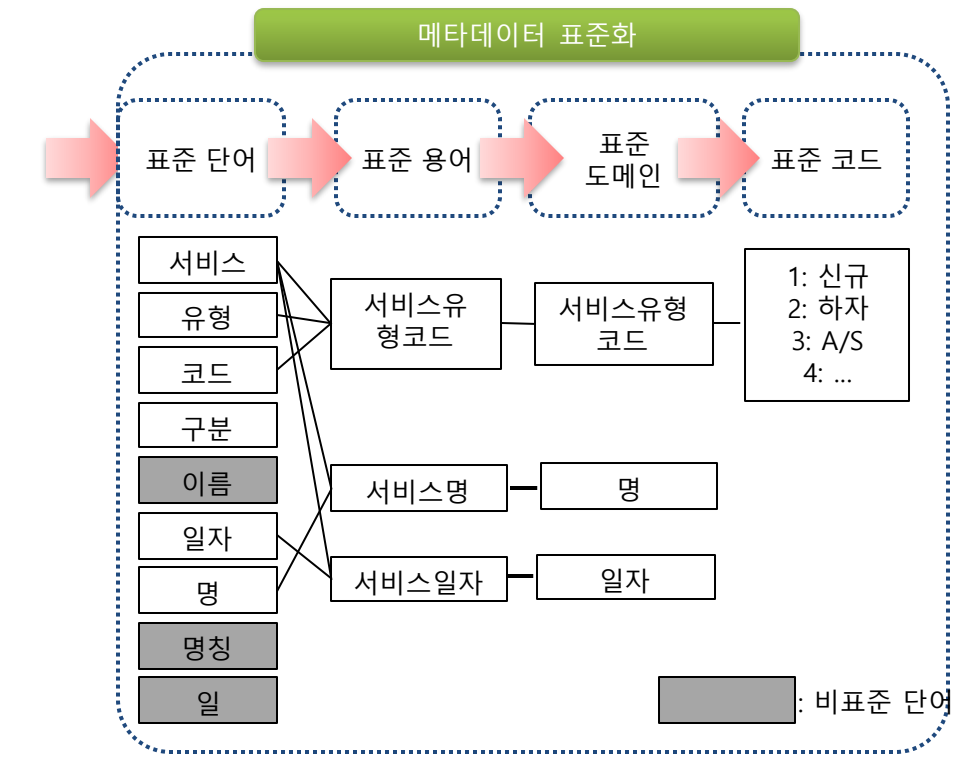
데이터에 대한 데이터



4. 데이터 표준화

데이터 표준화





4. 데이터 표준화

6) 데이터 표준화 지침 수립

데이터 표준화를 정의하기 전에 전사적으로 지켜야 하는 지침을 미리 정해야 함

공통 지침 사례

① 업무에서 사용되고 있는 관용화된 용어를 우선해서 사용함
② 물리적 설계 단계에서 영문명으로 전환할 때 문법에 맞는 영어 단어를 사용함
③ 한글명이나 영문명을 부여할 때 특수문자나 띄어쓰기는 사용하지 않음
④ 하나의 한글명에 대해서 하나의 영문명만 사용함 (동음이의어 사용 불가)
⑤ 하나의 영문명에 대해서는 하나 이상의 한글명을 허용함 (이음동의어 허용)

표준 용어 지침 사례

① 수식어를 활용해서 용어의 의미가 명확하도록 함
② 용어의 길이가 너무 길지 않게 하고, 너무 긴 경우 약어를 사용함
③ 논리적 모델에서는 한글명, 물리적 모델에서는 영문명을 사용함
④ 영문명을 사용할 때는 파스칼 명명 규칙*을 사용함
⑤ 단일 식별자 속성인 경우에는 'ID'라는 접미어를 사용함
⑥ 이력 개체인 경우에는 '이력'이라는 접미어를 사용함
⑦ '이름'을 다른 단어와 조합할 때는 '명'이라는 접미어를 사용함
⑧ 날짜만 의미할 때는 '일자'라는 접미어를, 시간까지 의미할 때는 '일시'라는 접미어를 사용함

첫 번째 문자와 그 뒤에 이어지는 단어들의 첫 문자를 모두 대문자로 표시하는 방법
예) OfficePhone

표준 도메인 지침 사례

① 도메인은 우선 논리적인 데이터 형식인 문자, 숫자, 날짜 등으로 구분해서 정의함
② 이후 논리적 데이터 형식을 가능한 최소의 물리적 데이터 타입을 사용해서 저장공간이 최소화되도록 정의함
③ 도메인이 범위에 따라 두 개 이상의 데이터 타입으로 구분되는 경우, '도메인명(데이터타입)' 형태의 이름으로 도메인을 분리함

표준 코드 지침 사례

① 코드는 가능한 최소의 숫자 데이터 형식을 사용해서 저장공간이 최소화되도록 함
② 코드 속성은 의미를 분명히 하기 위해 구분, 유형, 상태, 여부 등의 접미어를 사용함
③ 코드 사용의 의미를 분명히 하기 위해 동일한 코드라도 수식어를 사용해서 코드 속성명을 다르게 정의할 수 있음
④ 도메인 값의 범위가 명확한 경우, 별도로 코드화해서 관리하지 않음
예) '이원코드 도메인'의 경우, 남녀, 찬반 등에 대해 1, 2라는 값만 가질 수 있으므로 별도로 코드를 정의하지 않음
⑤ 코드는 전체 DB 설계에서 유일하게 정의되어야 함

4. 데이터 표준화

7) 데이터 표준화 방법

공통 지침 정의 방법

1. 현재 사용 중인 모든 용어를 수집하고, 필요한 경우 추가 용어를 정의
2. 수집 및 정의된 용어들을 기본 단어로 분할
3. 기본 단어들을 정제 - 정제 대상: 중복 단어, 동음이의어, 이음동의어 등
4. 데이터 표준화 지침을 참고해서 **'표준 단어 사전'**을 작성

예) 대학 관련 표준 단어 정의

표준 단어	영문약어명	영문전체명	설명
대학교	UNIV	University	2~4년제 대학교
주소	ADDR	Address	집 주소
. . .			

표준 용어 정의 방법

1. 업무에서 사용하는 용어를 선택해서 업무 용어를 정의하고, 유사 용어는 표준 용어로 대체
2. 표준 단어를 이용해서 표준 용어를 구성하고, 표준 도메인과 표준 코드를 기준으로 데이터 타입과 길이를 결정해서 표준 용어 정의
3. 데이터 표준화 지침을 참고해서 **'표준 용어 사전'**을 작성

예) 대학 관련 표준 용어 정의

용어	영문명	데이터타입	길이	표준 도메인	표준 코드
대학	UNIV	CHAR	3	코드	대학코드
등록일자	REG_DATE	CHAR	8	일자	-
. . .					

4. 데이터 표준화

7) 데이터 표준화 방법

표준 도메인 정의 방법

1. 현재 사용 중인 용어에서 유사한 속성의 용어 그룹핑
2. 그룹핑된 유사 속성에 대해 표준 도메인을 정의
3. 데이터 표준화 지침을 참고해서 **'표준 도메인 정의서'**를 작성

예) 대학 관련 표준 도메인 정의

분류	도메인명	설명	데이터타입	길이
코드	이원코드	남녀, 유무	CHAR	1
코드	대학코드	대학구분 코드	CHAR	3
. . .				

표준 코드 정의 방법

1. 현재 사용 중인 코드와 필요하다면 추가 코드 정의
2. 표준 도메인을 적용해서 표준 코드 정의
3. 데이터 표준화 지침을 참고해서 **'표준 코드 정의서'**를 작성

예) 대학코드에 대한 표준 코드 정의

그룹코드	그룹코드명	데이터타입	길이	코드값	코드설명
GC001	대학코드	CHAR	3	01	한기대
GC001	대학코드	CHAR	3	02	서울대
. . .					

쉼터

처음 출근하는 이에게

-고두현-

잊지 마라
지금 네가 열고 들어온 문이
한때는 다 벽이였다는 걸

쉽게 열리는 문은
쉽게 닫히는 법
들어올 땐 좁지만
나갈 땐 넓은 거란다

집도 사람도 생각의 그릇만큼
넓어지고 깊어지느니
처음 문을 열 때 그 떨림으로
늘 네 집의 창문을 넓혀라

그리고 창가에 앉아 바라보라
세상의 모든 집에 창문이 있는 것은
바깥 풍경을 내다보기보다
그 빛으로 자신을 비추기 위함이니

생각이 막힐 때마다
창가에 앉아 고요히 사색하라
지혜와 영감은 창가에서 나온다

어느 집에 불이 켜지는지
먼 하늘의 별이 어떻게 반짝이는지
그 빛이 내게로 와서
어떤 삶의 그림자를 만드는지

시간이 날 때 마다
그곳에 앉아 너를 돌아보라
그리고 세상의 창문이 되어라
창가에서는 누구나 시인이 된다