***데이터 모델링***

***0교시. Introduction***

***1교시. 데이터 모델링의 정의***

***2교시. 구성요소 및 개념***

***4교시. 작업절차 및 기본기법***

*-------------------------------------------------------------*

*교육목표*

-데이터 모델링 관련 기본 Skill을 습득 함

데이터 모델 관련 개념 및 정의에 대한 정확한 이해

데이터 모델 설계 기법 이해

-데이터 모델링 관련 고급Skill을 자습할 준비를 완료 함

참조 데이터 모델 개요 이해

참조 문서 및 학습 방법 이해

*-------------------------------------------------------------*

*왜 데이터 모델링인가?*

-소프트웨어의 본질은 동작과 데이터

소프트웨어는 프로그래밍 언어를 이용해서 데이터를 제어하는 수단

소프트웨어의 가장 중요한 요소인 데이터를 얼마나 잘 이해하는 지가 중요

-소프트웨어는 데이터를 제어하기 위한 언어나 도구를 사용

다양한 프로그래밍 언어와 개발도구를 사용

CRUD로 대표되는 다양한 기능들을 구현

-소프트웨어는 데이터를 제어하는 것

이 CRUD는 결국 데이터를 제어하는 과정

-데이터모델링의 본질도 결국 데이터

추상화하고, 단순화하고, 명확하게 만들어가는 과정

1. 비상연락망 작성해보기(설계서)

2. 요구사항(요건, Requirement)

ex 성명, 휴대폰번호, 이메일주소, 집주소

3. 요구사항 추가

A,B반을 따로 공지할 필요가 있음.

=> 열(컬럼)을 추가

or

=> 데이터 정규화

A반 성명, 휴대폰번호, 이메일주소, 집주소

B반 성명, 휴대폰번호, 이메일주소, 집주소

4. 요구사항 추가2

혈액형별로 공지할 필요가 있음

**데이터 모델링의 정의**

*기본 개념*

-정의

사용자의 요구사항을 분석해서 IT 시스템을 구축하기 위하여 '데이터'요건을 형상화 하는 작업

-필요성

요구사항은 보통은 애매한 것이 많음

사용자는 여러 사람이 있음

데이터는 여러 업무에서 사용됨

따라서 어떤 데이터가 필요한지를 정리하고 명확히 커뮤니케이션 할 필요가 있음

개별 업무 시각에서 벗어나, 전체 시각에서 일관된 개념하에 일관된 방법으로 분석/정리하여야 함

\*데이터베이스 설계(모델링)

요구사항에 맞는 컬럼들이 존재

데이터관점에서 그 데이터를 데이터베이스화할 필요가 있을 때 컬럼으로 추가

*-------------------------------------------------------------*

*데이터 모델링의 중요성*

-파급효과(Leverage)

데이터 모델 변경 → 표준 영향 분석, 응용 변경 영향 분석 등 → 해당 분야의 실제적인 변경 작업→ 전체 시스템 구축 프로젝트에서 큰 위험요소

-복잡한 정보 요구사항의 간결한 표현(Conciseness)

구축 할 시스템의 정보 요구사항과 한계를 가장 명확하고 간결하게 표현할 수 있는 도구.

데이터 모델 : 건축물의 설계도면

-데이터 품질(Data Quality)

데이터 품질의 문제가 야기되는 중대한 이유 중 하나가 데이터 구조의 문제

*데이터 모델의 종류 – level 구분*

통상 Conceptual(개념), Logical(논리), Physical(물리)의 3단계로 구분함



-데이터 개념

정보를 추상화하여 단순화 된 개념으로 분류한 것

-개념모델

핵심 엔티티 및 핵심 엔티티 간의 관계를 표현한 모델(Key 속성만 포함)

-논리모델

어플리케이션에 의한 데이터 사용 형태 및 구현상의 제약 사항을 고려하지 않고 작성된 모델

-업무 논리모델

어플리케이션에 의한 데이터 사용 형태 및 구현상의 제약사항을 고려하여, 도출 속성의 추가, 관계의 고정 및 데이터의 중복 등과 같은 비정규화 작업이 수행된 모델

-물리모델

시스템의 기술요건 및 운영 요건을 반영한 모델

\*개념Conceptual모델링 : 어떤 데이터를 의미있게 관리할 것인가, 추상적으로 분류

DA(Data Architector)는 어떤 조직이든 있어야 함

\*논리Logical모델링 : 개념모델을 기반으로 조금 더 깊이있게 분류

\*물리Physical모델링 : 눈에 보이게 작성 ex.엑셀시트

*데이터 모델링 Approach*

Top-Down

데이터 개념부터 시작하여 상세 모델 작성

일반적으로 (개념) → 개념모델 → 논리모델 → (업무논리모델) → 물리모델로 상세화 단계를 거침

\* 기본적으로 개념 -> 논리 - > 물리

Bottom-Up

데이터 베이스를 Reverse하는 방식으로 작업

일반적으로 데이터베이스 → 물리모델 → 논리데이터 모델→ (개념모델) 단계로 작업함

\* 기존의 시스템을 새로 재개발하려고 할 때

기존의 물리모델을 기반으로 분석하여 추상화해내감 -> 논리 -> 개념

Hybrid

Top-Down과 Bottom-Up을 혼용

Top-Down Approach로 일관된 개념을 가진 통합데이터 모델을 작성하고, Bottom-Up방식으로 누락 없는 상세 모델을 작성

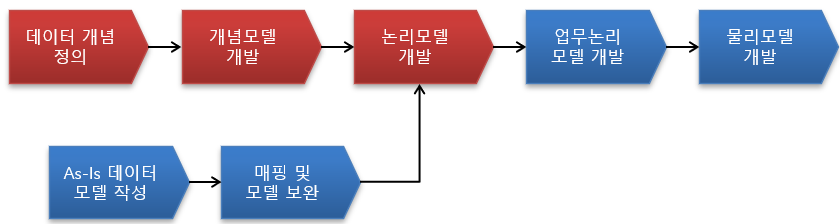
*-------------------------------------------------------------*

*작업 흐름*

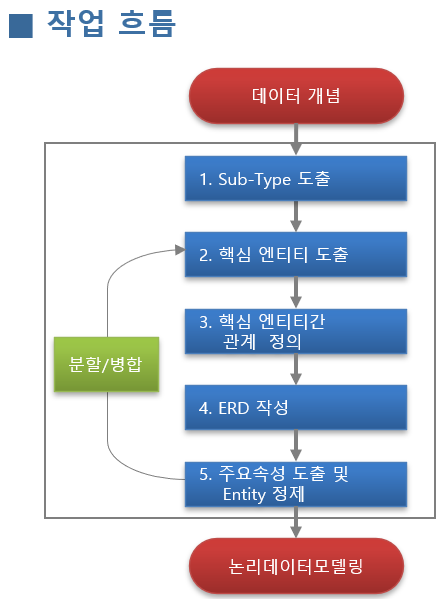
데이터 개념 정의 – 전사 통합 데이터 모델과 같은 대규모의 데이터모델을 개발할 때 선행 필요

개념 모델 개발 – 대규모 통합 데이터 모델 개발 시, 전체 모델의 구조 및 핵심 엔티티간 관계 정의 필요

논리 데이터 모델 개발 – 개념 모델을 상세화하고 As-Is모델을 통한 보완작업 과정을 거쳐 완성 함



*개념 데이터 모델링 작업 절차*



1. Sub-Type의 도출

데이터 개념별로 Sub-Type을 도출 함

예) Involved Party의 Sub-Type으로 개인/법인을 도출

\*부분집합 ex. A반, B반

2. 핵심 엔티티 도출

Sub-Type을 핵심 엔티티로 도출함

\* sub-type별로 관리할지 고민해야 함

ex. 비상연락망 sheet1, sheet2 -> 2개의 entity

3. 핵심 엔티티간 관계 정의

핵심 엔티티간 관계를 분석함

필요시 Relationship Entity를 도출함

ex. 담당교수 - 학생들

4. ERD(Entity Relationship Diagram) 작성

데이터 개념별로 Sub-Type Entity, Relationship Entity간   
관계를 ERD로 작성

ex. entity 안에 column속성.... 실선,점선관계.... 까마귀발톱관계.... o |....

\*모델링 정규화

cf. 문자+숫자 = 코드

ex.신입생 입학 -> 명단시트는 변경o, 반시트는 변경x

반시트에는 '영향도가 없다'

'영향도가 쾌적해졌다'

\*명단시트에 반코드가 추가되고 반시트가 생성되며

-> entity 사이에 relationship이 생겼다

-> 영향도가 생겼다

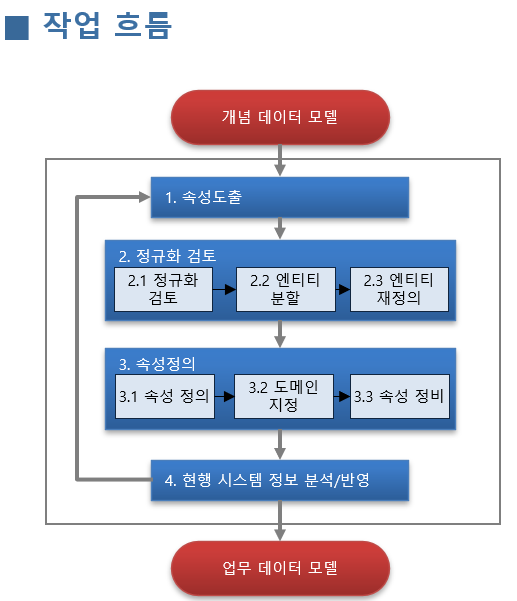
-> 종속관계가 생겼다

5. 주요속성(Column) 도출 및 Entity정제

엔티티의 Key 및 주요 속성을 정의함

필요시 속성형 엔티티를 도출 함(주요 속성형 엔티티로 한정)

*논리 데이터 모델링 작업 절차*

 주제영역정의 > 엔티티정의 > 속성정의 > 관계정의

ㄴ엔티티도출 ㄴ속성도출 ㄴ관계도출

1. 속성 도출

상세 속성을 도출함

필요시 속성형 엔티티를 도출함

2. 정규화 검토

제1, 2, 3 정규화를 수행함

필요시 엔티티를 분리함

3. 속성 정의

속성의 의미를 정의하고 도메인을 지정 또는 생성함

4. 현행 시스템 정보 분석/보완

현행 시스템 정보 반영 여부 검토

필요시 엔티티/속성 보완

**구성 요소 및 개념**

**엔티티(Entity)**

*Entity - Relationship*

개체들을 관계 중심으로 기술하는 이론

\*데이터 모델 3요소 Entity, Relation, Attribute 중

ER ( Entity - Relationship ) 이론

관계 (關係)

둘 이상의 사람, 사물, 현상 따위가 서로 관련을 맺거나 관련이 있음. 또는 그런 관련.

어떤 방면이나 영역에 관련을 맺고 있음. 또는 그 방면이나 영역.

남녀 간에 …

피터 첸 (Dr. Peter Pin-Shan Chen, 陳品山, 1947년 1월 3일~)

분야 : 컴퓨터 과학

수상 : ACM - AAAI Allen Newell Award, 해리 H. 구드 기념상 등

엔티티 관계 모델의 창작자.

현 루이지애나 주립 대학교 컴퓨터 과학과의 M. J. Foster Distinguished Chair 교수

1976년, 『The Entity Relationship Model – Toward A Unified View of Data』

컴퓨터 소프트웨어 분야에서 가장 많이 인용되는 논문 중 하나

컴퓨터과학 분야에서 가장 많은 영향을 미친 38개의 논문에 선정

전문용어의 필요성

Entity, Entity Occurrence, Attribute, Cardinality, Relation, Normalization, Sub-Type

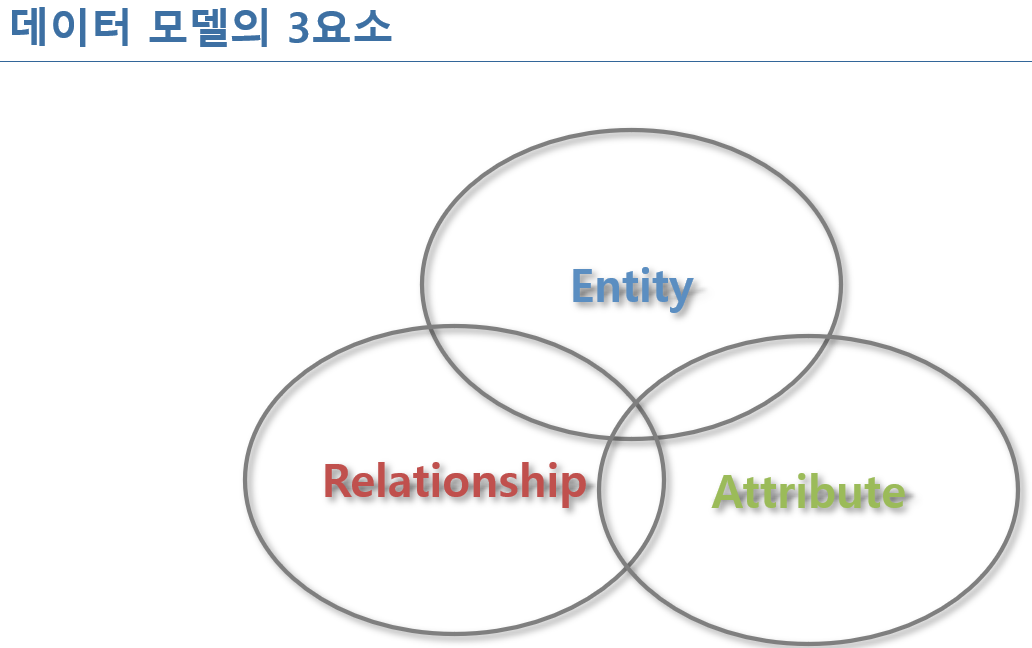
엔티티, 엔티티 어커런스, 속성, 기수, 관계, 정규화, 하위유형…

알쓸신잡 (알아두면 쓸데없는 신비한 잡학)

모델에 대한 개념과 의견들을 정의하고 표현할 수 있다.

설계 프로세스 자체를 정의하고 표현할 수 있다.

논리영역은 물론 물리영역의 DBMS까지 언제나 사용된다.



*-------------------------------------------------------------*

*entity의 정의*

해당 조직이 업무를 수행하는데 있어 기록관리가 요구되는 유형 무형의 것(사물)의 종류(유형)를 의미함

엔티티를 정의한다는 것

데이터 관리 대상이 되는 것(사물)의 종류(유형)를 도출하는 작업

것(사물)의 종류?

Entity의 원래 용어는 Entity Type이었음

Entity Type이란 Entity를 분류하는 작업을 통해 도출 됨

사물의 종류 (예)

101-111-12345(A고객의 예금계좌), 100-112-12346(B고객의 적금계좌)   
⇒ 수신계좌

요구불예금 상품A, 요구불예금 상품B, 신용카드 상품C, 신용대출 상품D  
⇒ 상품

Entity를 얼마나 추상화하여 보느냐에 따라서, 같은 Entity Type이 되기도 하고, 다른 Entity Type이 되기도 함

예금계좌, 적금계좌 / 수신계좌

\*얼마나 추상화하느냐에 따라 entity type이 달라짐

cf. 보통예금, 적립식예금

*엔티티(Entity)의 특징*

특징

엔티티는 다음과 같은 특징이 있으며, 만약 도출된 엔티티가 다음의 성질을 만족하지 못하면 부적절한 엔티티일 확률이 높음

-업무에서 필요한 정보를 갖는 것이어야 함

-업무를 수행함에 있어서 지속적으로 중요한 의미를 가진 것이어야 함

-구성 엔티티 어커런스를 식별자에 의해 유일하게 구별(Uniquely Identified)할 수 있어야 함

etc. 반코드, 학번

-궁극적으로 속성(Column) 값으로 표현되는 관찰 가능한 특성을 갖고 있어야 함

-------------------------------------------------------------

\* UML ( Unified Modeling Language 통합모델링언어 )

다이어그램

ERD도 UML 중의 하나

-------------------------------------------------------------

*엔티티(Entity)의 명명규칙*

명명규칙

-현장에서 보편적으로 사용하는 업무용어를 사용함

-명사 또는 명사구를 사용함

-모델에서 유일한 이름이어야 함

-뜻이 보존되는 한 이름을 짧게 해야 함

-“~ 기본/상세/내역/이력/관계/목록” 등으로 엔티티의 성격에 따라 명칭을 통일하는 경우도 있음

\*기본(master table, 원장) : 원천을 의미, 중복되지 않음, 식별자가 있음

\*상세, 내역, 이력, 관계, 목록...등 기본을 보조하는 역할

\*관계 : 식별자-식별자의 관계를 매핑mapping해줌

*엔티티(Entity) 관련용어*

엔티티 어커런스(Entity Occurrence)

=엔티티 인스턴스의 집합

엔티티에 소속된 사물의 예

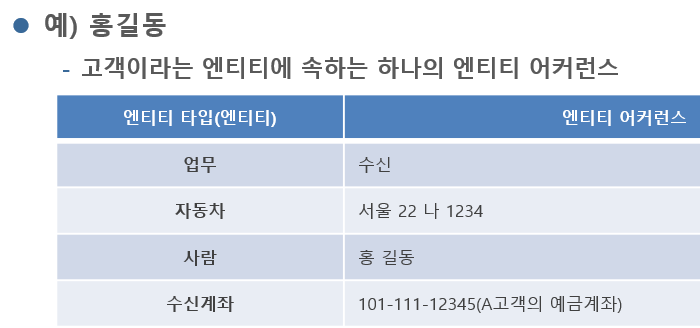
엔티티 인스턴스(Entity Instance)라고도 함

간혹 Entity라고도 함. 이 경우에는 Entity Type이 엔티티이고 Entity는 엔티티 어커런스를 의미함

ex 학생기본 테이블의 학생 13명

예) 홍길동

고객이라는 엔티티에 속하는 하나의 엔티티 어커런스



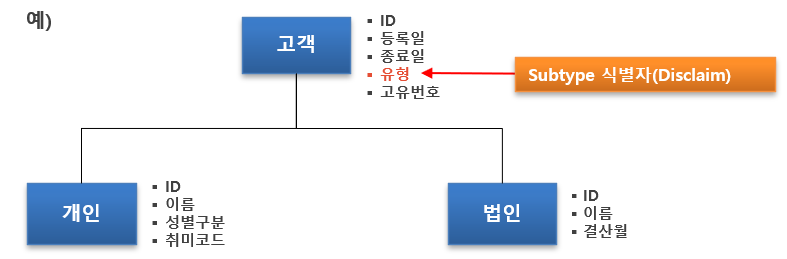
*Super Type 엔티티와 Sub Type 엔티티*

여러 개의 엔티티가 공통 속성이 있고 일부의 속성이나 관계만 다를 경우, 공통 속성을 위한 하나의 엔티티(Super Type)와 개별 속성을 위한 복수개의 엔티티(Sub Type)를 정의 함

Super type Key와 Sub type Key는 동일 함

Sub type의 엔티티 어커런스는 반드시 Super type에 속하여야 함

잠재적으로 null값을 가지는 속성은 Sub type으로 정의됨



\*Super Type 합집합, 추상적, 포괄적

\*Sub Type 부분집합, 추상적인걸 일반화, 좁은 의미로 줄임

ex. 금융서비스에서 고객 1.개인 2.법인

고객의 유형 = subtype식별자(disclaim)로 구분

ex. A면 개인 B면 법인

개인/개인사업자

법인/법인사업자

사업자는 영리추구

ex. 폴리텍-법인, 뱅크-법인사업자

cf. 금융의 5고객: 개인,법인,개인사업자,법인사업자,임의단체 (->sub type)

임의단체 ex. 동창회, 계모임

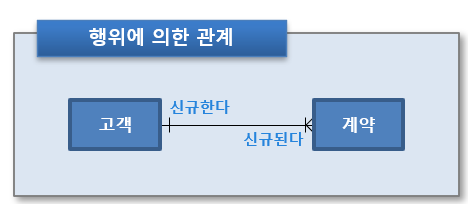
**관계(Relationship)**

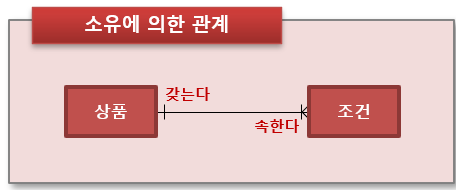
관계가 존재하려면 entity가 2개이상 필요

*관계(Relationship)의 정의*

정의

엔티티 간에 존재하는 상호간의 연관성을 “Cardinality”와 “Relationship Name”으로 표기 함





관계는 업무규칙(Business Rule)을 기술함

예) 1명의 고객은 계약을 신규하거나 안하고, 1개의 계좌는 1명의 고객에 의해서 소유됨

1개의 상품은 조건을 1개 이상 가지며, 하나의 조건은 반드시 어떤 하나의 상품에 관한 것 임

관계는 데이터 상으로 관리 대상이 되는 관계를 의미 함

두 엔티티 간의 관계는 2개 이상일 수 있음

(ERD에는 대표적인 성격 하나만 그리는 것이 일반적)

\*금융에서는 모든 관계가 계약으로 인해 성립

\*고객은 계약을 신규한다 <-> 계약은 고객에게 신규된다

cf 계약(contract) : 주문(order), 보험, 펀드 등

\*엔티티 간에 존재하는 상호간의 연관성을 “Cardinality”와 “Relationship Name”으로 표기 함

\*Cardinality 603호가 소유할 수 있는 경우의 수

ex.책상,의자,선풍기,빔프로젝트...

나한주 고객이 소유하고 있는 예금통장의 수 ex...

\*Relationship Name관계명

ex.신규한다, 소유한다, 갖는다...

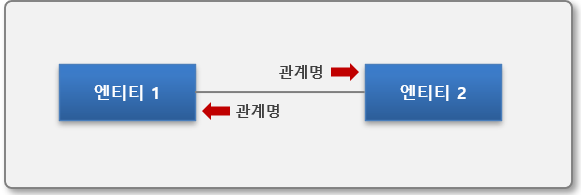
-------------------------------------------------------------

*관계명(Relationship Name)*

두 엔티티 사이의 관계는 선으로 표시 함

관계선의 좌우 또는 상하에 관계명을 기록함 (시계 방향으로 기록)

통상 Application 데이터 모델(C’ Level) 이하 단계에서는 크게 의미가 없어 표기하지 않음  
(이유 : 엔티티 관계가 관계명 없이도 명확하게 파악되는 경우 관계명을 기술할 필요는 없으므로)

*관계명(Relationship Name)의 명명규칙*

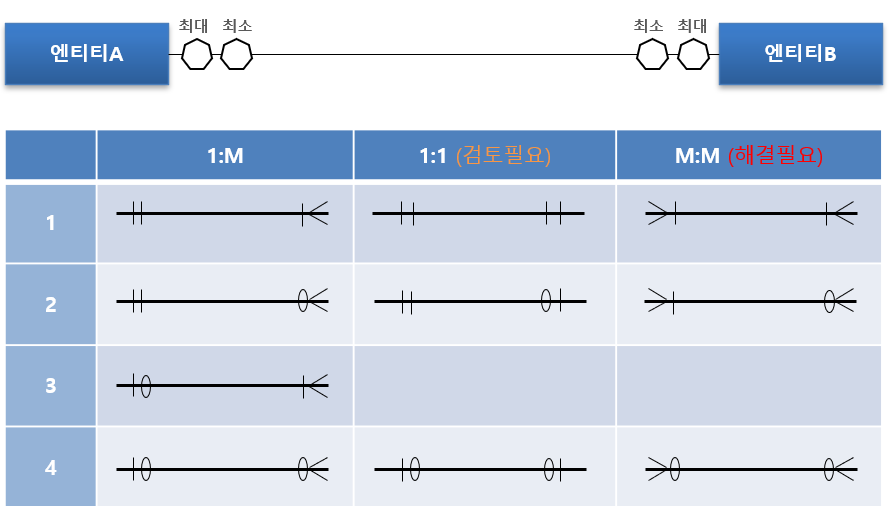
애매한 동사는 사용하지 않아야 함.

예) 관계된다, 관련있다.

현재형으로 표현 함. 예) 신청한다.

*관계(Relationship)의 기수*

*기수(Cardinality)의 발생 유형*



1:N 일대다관계

|| (최대1 최소1) ---------------------- (최소1 최대1) ||

|| (최대1 최소1) --------------------- (최소0 최대多) 0<

|0 (최대1 최소0) --------------------- (최소1 최대多) |<

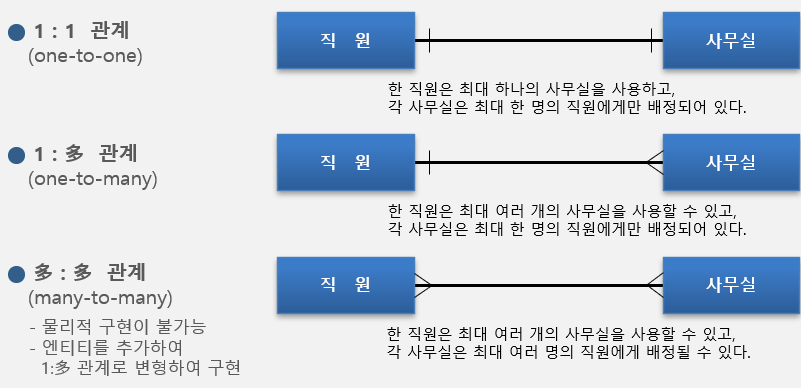
|0 (최대1 최소0) --------------------- (최소0 최대多) 0<

*최대 기수 - 3가지 유형*

최대 기수(Cardinality) : 두 개의 엔티티 간 관계에서, 한 엔티티의 한 어커런스를 기준으로 상대 엔티티의 관계 된 어커런스가 최대로 몇 개까지 있을 수 있는가를 표현

1:1, 1:多, 多:多

=관계차수



*최소 기수 - 3가지 유형*

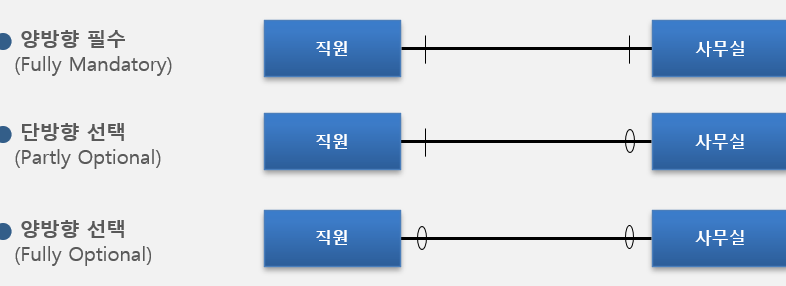
두 개의 엔티티 간 관계에서, 한 엔티티의 한 어커런스를 기준으로 상대 엔티티의 관계 된 어커런스가 적어도 몇 개까지 있을 수 있는가를 표현

0 or 점선

: 발생할 수도 있고 발생 안 할 수 도 있는 경우 (Optional)

1이상 or 실선

: 꼭 발생해야 하는 경우 (Mandatory) 두 개의 엔티티 간 관계에서, 한 엔티티의 한 어커런스를 기준으로 상대 엔티티의 관계 된 어커런스가 적어도 몇 개까지 있을 수 있는가를 표현



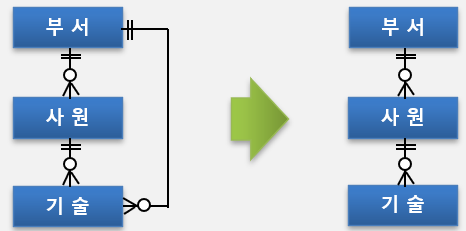
-------------------------------------------------------------

*관계 검증*

-원형관계(Circular Relationship) 제거



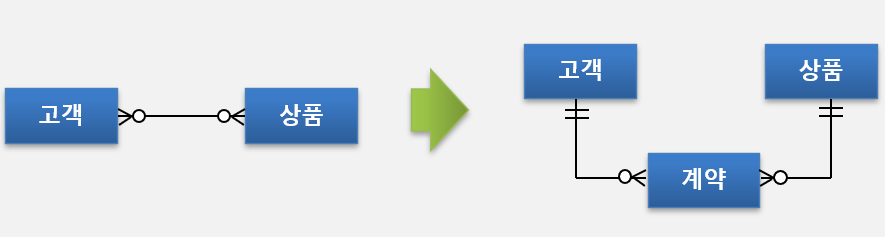
-사슬관계(Chain Relation) 제거

**

-M:N 관계 단순화

관계를 별도 엔티티로 분리

숨겨진 정보로부터 연관 엔티티를 추출

다대다 관계는 엔티티를 추가하여 정규화해주어야 함

-1:1 관계 정제

데이터 모델링 과정에서 엔티티 간의 관계가 1:1로 대응되었다는 사실은 해당 엔티티들의 기본키가 동일하다는 의미하며 1:1 관계를 가진 모델은 3가지 방법으로 표현할 수 있음

1.하나로 통합

Mandatory 관계, 기본키가 동일,

대부분의 속성과 관계가 동일

2.그대로 유지

Optional 관계이면서 대부분의 속성 및 관계가 상이한 경우, 대응되는 속성이 값이 서로 상이한 경우

3.새로운 Super Type 엔티티 추가

기본키가 동일하고 속성의 일부만 상이한 경우

\*학생기본 - 학생휴대폰기본

1:1(한쪽이 수정되면 다른쪽도 수정됨), 공유하는 컬럼(학생식별자)이 있음-> 관계가 있음

\*학생기본 - 학생반관계

1:다, 공유하는 컬럼(학생식별자)이 있음-> 관계가 있음

\*학생반관계 - 반기본

1:다, 공유하는 컬럼(반식별자)이 있음-> 관계가 있음

\*학생기본 -x- 반기본

공유하는 컬럼이 하나도 없음-> 관계가 없음

*관계(Relationship)별 엔티티 종류*

관계를 기준으로 한 엔티티 종류

기본형 엔티티 (Fundamental Entity)

속성형 엔티티 (Attributive Entity)

관련형 엔티티 (Associative Entity)

**속성(Attribute)**

*속성(Attribute)의 정의*

-정의

엔티티가 관리해야 되는 데이터 항목으로 더 이상 분리될 수 없는 최소의 데이터 보관 단위

향후 물리 데이터 모델링 작업을 통하여 컬럼으로 변환됨

-속성의 명명규칙

해당 업무에서 사용하는 이름을 부여 함

엔티티에서 유일하게 식별가능 하도록 지정해야 함

가능하면 약어를 사용하지 않아야 함

속성은 전체 데이터모델에서 하나의 의미만을 가져야 함

“수식어 + 도메인 명” 형식을 사용하여 표준화 함

예) 데이터 항목 이름이 다르고 의미가 같은 경우

: 계좌개설일, 계좌신규일

데이터 항목 이름이 같고 의미가 다른 경우

: 일자(계좌신규일, 계좌해지일)

-------------------------------------------------------------

*속성 타입 (Attribute Type)*

-근본적으로는 속성(Attribute)과 같은 의미 임

-엔티티가 갖는 속성의 유형을 의미함

예 : (고객이라는 엔티티에서) 고객의 성명, 주소, 소속회사명, 전화번호 등

-------------------------------------------------------------

*속성 인스턴스 (Attribute Instance)*

-속성 타입의 값(Value)을 의미 함

예) 128만원 -- 대출총액이라는 속성타입에 속하는 하나의 속성 인스턴스

\*엔티티 인스턴스 = 레코드

-------------------------------------------------------------

*속성(Attribute)의 도메인*

*도메인(Domain)*

-속성이 가질 수 있는 값에 대한 업무적 제약요건으로부터 파악된 특성

-속성에 대한 데이터 Type과 크기, 제약사항을 지정한 것

항목 표준화에 따르는 사항으로 데이터관리자가 설정

모든 업무영역에서 같은 도메인 집합을 사용

-도메인의 활용 (중요성)

속성 값의 비교 기준

공통 검증 규칙

-도메인(Domain)

어떤 한 속성(Attribute)에 허용되는 업무적으로 의미 있는 인스턴스 값의 집합

-도메인의 종류

열거형 도메인

코드와 같이 인스턴스 값이 정해져 있으며, 개수가 한정되어 있는 것으로서 업무의 관리체제가 변하지 않는 한 인스턴스는 일정함.

제약형 도메인

인스턴스 값이 하한가와 상한가와 같이 값의 범주가 정해져 있으나 특정한 값을 정할 수 없고 개수도 제한이 없는 것

예) 금액, 계좌번호, 부품번호

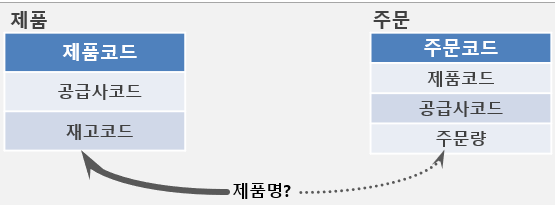
-도메인의 구성요소

도메인명, 데이터 타입, 자릿수, 인스턴스 기술서(열거형 도메인 경우)

*속성(Attribute)의 배치*

기본키에 종속되는 속성을 해당 엔티티에 배치

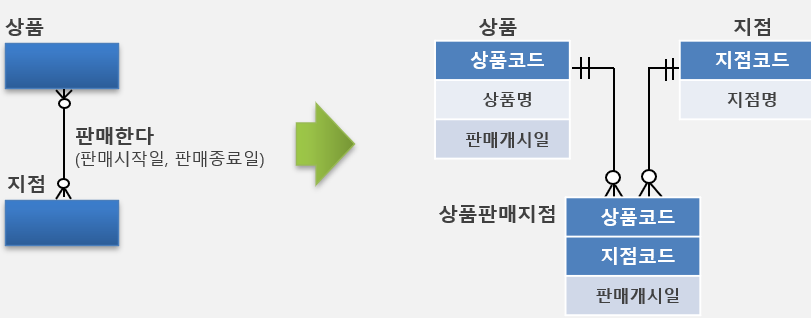
가능한 범위 내에서 부모 엔티티 쪽으로 배치



속성이 여러 개의 값을 가질 수 있는 경우, 새로운 자식 엔티티를 만들어 자식 엔티티의 속성으로 배치



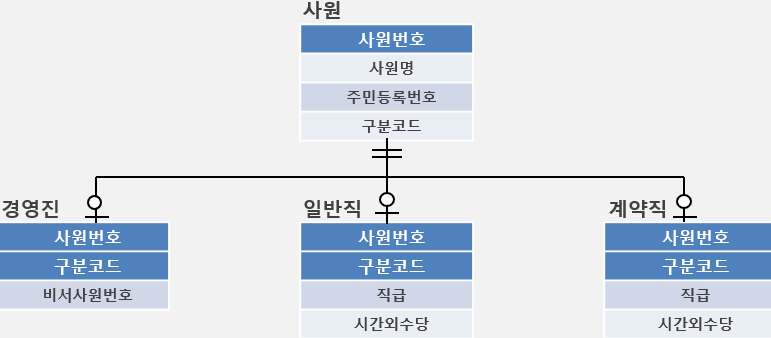
관계를 구체화 시켜주는 속성이 존재할 경우 관계를 엔티티로 전환시켜 해당 속성 배치



Super Type 엔티티, Sub Type 엔티티 내의 속성 배치

Super Type 엔티티에 Sub Type 엔티티의 구분항목을 속성으로 등록

Sub Type 엔티티 모두에 포함되는 속성을 Super Type 엔티티로 이동



**식별자(Key)**

*식별자(Key)의 정의*

정의

식별자란 한 엔티티 내에서 각각의 어커런스를 유일하게 구분할 수 있는 단일 속성 또는 속성 그룹을 말함

관련 용어

후보키(Candidate Key)

기본키(Primary Key), 주요키, 일차키

대체키(Alternate Key)

복합키(Concatenate Key), 연결키

외부키(Foreign Key)

합성키(Compound Key)

-------------------------------------------------------------

기본키(Primary Key)

어느 엔티티 어커런스를 식별하기 위한 속성을 의미 함

조건

유일하게 구분이 되어야 함 (Unique)

반드시 데이터 값이 존재해야 함(Not Null)

가능하면 변하지 않아야 함 (Never Change)

가능하면 업무적으로 활용도가 높은 것으로 정의하여야 함

가능하면 길이가 짧은 것을 선정하여야 함

-------------------------------------------------------------

후보키(Candidate Key)

엔티티 내에서 각각의 어커런스를 유일하게 구분할 수 있는 속성으로 하나 또는 하나 이상의 속성으로 구성됨

즉 기본키(Primary Key)가 될 수 있는 후보 속성 임

-------------------------------------------------------------

대체키(Alternate Key)

대체키는 후보키 중에서 기본키로 선정되지 않은 속성을 말 함

즉, Primary Key 이외에 엔티티 인스턴스를 Unique하게 식별하는 Key 임

-------------------------------------------------------------

복합키(Concatenate Key)

2개 이상의 속성으로 Primary Key를 구성한 Key

-------------------------------------------------------------

외부키(Foreign Key)

어떤 엔티티 A가 다른 엔티티 B와 관계를 갖는 경우, B 엔티티와 관계때문에 속성으로 갖게 되는 B 엔티티의 Primary Key를 말 함

-------------------------------------------------------------

합성키(Compound Key)

엔티티와 엔티티의 관계가 Many to Many 관계일 때 그 관계를 나타내는 엔티티의 Key

\*두 엔티티의 'primary key'들을 합친것

-------------------------------------------------------------

\*개발자는 NULL과 중복을 싫어함

\*NULL을 없애려면 entity를 쪼개라 = 데이터를 정규화해라

\*데이터 모델러(아키텍터)는 선호, 개발자는 비선호

**주제영역(Subject Area)**

정의

엔티티를 도출하고 관리하는 단위 영역

일반적으로 데이터 중심으로 정의하나, 경우에 따라서 업무 흐름 중심으로 정의되기도 함

계층이 있음

최하위 주제영역은 ER Diagram 의 작도 단위 임

주제영역의 명명규칙

현업에서 보편적으로 사용하는 업무용어를 사용

데이터의 그룹을 의미하는 이름을 부여

계층을 표현하기 위하여 xxx\_yyy 형식으로 사용하기도 함

-------------------------------------------------------------

**표현방법(Notation)**

데이터 모델의 표현

ER 모델의 표기법은 다양하고, 모델링 도구 업체별로 다름

Peter Chen 표기법, Information Engineering 표기법, Barker Ellis 표기법 등

**작업절차 및 기본기법**

**작업절차**

*논리/물리모델의 차이*

논리모델 -> 물리모델(테이블레이아웃)

*성능을 고려*

-서브타입 모델의 변환

-엔티티 합체와 분해

-비정규화 (역정규화)

-PK(Primary Key) 확정

-테이블 파티션 확정 : 가장 최근에 쌓인 데이터는 빨리 불러오고 오래된 데이터를 불러올 때의 속도는 포기하는 기법

(파티션이 없으면 모든 데이터가 느려짐)

why? 성능에 한계가 있어서

DBMS에 따라 파티션 기능을 지원하기도 안하기도 한다

-데이터 저장 방법 확정 : 일자별로 할지 역순으로 할지 등등

-인덱스 설계

-뷰 설계 : 두 테이블을 조인해서 셀렉트한 쿼리가 있을 때 결과물이 항상 한 화면에만 나타난다면

두 테이블을 조인하지 않고 하나로 만들면 속도는 빠르지만

데이터 관점에서 구분이 되므로 엔티티를 분리해서 관리하는 것을 지향

->이해관계 상충

->가상의 테이블 view를 만들어서 select하면 테이블을 조인하지 않아도 됨

-시스템 속성 추가

cf. 스키마 : 테이블에 접근할 수 있는 권한을 조정할 때

물리스키마 ex. 테이블

가상스키마 ex. 뷰

ex. 700개의 데이터 중 50개는 고객, 100개는 정산

담당자가 서로 접근할 수 없게 하고 싶음

고객데이터 50개를 스키마로, 정산데이터 100개를 스키마로 설정하여 접근 권한을 조정

\*point. 물리모델은 DBMS에 종속되어있다

논리모델은 DBMS에 영향을 받지 않는다

**데이터의 개념**

*업무 데이터를 이해한다는 것*

선행관계, 종속성 규칙 등을 관찰

*업무데이터 분석 과정*

업무의 이해 -> 업무 데이터의 이해 -> 데이터 구조 모델링 -> DB생성

*FSDM(Financial Services Data Model)*

금융기관의 데이터는 크게 9가지의 Data 개념(Concept)을 근간으로 함

핵심 내용

금융기관의 데이터는 크게 9가지의 Data 개념(Concept)을 근간으로 함

금융기관의 데이터는 9가지 Data 개념에 관한 데이터와 9가지 Data 개념간의 관계에 관한 정보로 구성 됨

9가지 Data 개념에 관한 데이터는 Classification 정보와 Description 정보로 구성 됨

9 Concept

계약(AR) - 상품,서비스 등의 가치를 교환하는데 관련된 룰이나 의무/권리를 나타내는 둘 이상의 개인이나 조직간의 잠재적, 실제적인 약정

전략(BD) - 금융기관의 업무/재무 계획, 정책, 절차와 일정 등 업무를 수행하고자 하는 방식과 경영환경에 대한 관계당사자의 지침/전략

분류(CL) - 특정한 업무정보를 분류하는 카테고리의 구조

조건(CD) - 업무처리에 관한 구조화된 요건과 전제조건

이벤트(EV) - 업무 수행의 한 부분으로 데이터에 영향을 미치는 사건

이해관계자(IP) - 금융기관과 접촉을 하고 관심이 있는 모든 당사자

장소(LO) - 금융기관이 정보를 유지하고자 하는 장소, 목적지, 또는 지역

상품(PD) - 금융기관이 제조, 판매하는 금융상품 또는 서비스

자원(RI) - 업무를 수행하는 동안 금융기관이 운영,소유하거나 사용하는 유형/무형의 가치 있는 것

cf. ITA 정보기술아키텍처

데이터 개념의 예

HL7 Reference Information Model

-------------------------------------------------------------

**또다시 Entity**

범주화(Classification)와 추상화(Abstraction)

본질

대상의 가장 핵심적이고 필수적인 속성

엔티티는 집합이다. 엔티티는 본질이다.

*Entity 모델링의 고충*

-데이터 집합을 정의하기 쉽지 않다.

“여신 연체관리 진행”이라는 은행 업무

여신연체관리진행, 여신연체관리 + 진행

그 기준이 무엇인지 명확히 밝히고 객관화

-데이터의 본질을 볼 줄 알아야 한다.

대상에서 비즈니스 관점을 제거하는 것

-추상화 수준의 결정

서브타입별로 나누어 관리할지, 통합하여 관리할지

-하위 트랜잭션 데이터로 상위 논리집합 발견하기

**속성 분석**

*속성이 충분히 세분화되었는가?*

분리 가능한 속성은 최대한 분리해야 함

ex. 전표번호 -> 일자 + 텔러번호 + 일련번호

(일자와 텔러번호는 변동 가능한 값)

*속성이 일관성 있고 배타적인 값을 갖는가?*

값의 배타성을 고려하여 배타적이지 않은 경우, 속성을 분리해야 함

\*sub-type에 대한 고려

*속성이 같은 어커런스에 대해 복수의 값들을 갖는가?*

특정 시간에?

계속해서 변하는가? 과거의 기록이 중요한가?

그렇다면 별도의 엔티티로 도출하여야 한다

ex. 집주소이력이라는 엔티티로 관리

*복수 어커런스가 이 속성에 대해 같은 값을 갖는다면, 그 값들이 업무에 의미가 있는 것인가?*

만약 그렇다면 별도의 엔티티로 분리하여야 함

*이 속성이 도출 가능한 것이라면 그 값이 모든 어커런스에 적용 가능한 것인가?*

도출 가능한 속성은 제외함

**정규화(Normalization)**

*정의*

효율적인 관리를 위한 데이터구조를 도출하기 위하여, 속성간의 관계를 분석하여 별도 엔티티 도출 여부를 결정해 나가는 절차에 관련된 이론 임

\*엔티티를 얼마나 잘게 찢을것인가

*효과*

정규화의 기본 원칙은 하나의 테이블에는 중복된 데이터가 없도록 하는 것이며 효과는 다음과 같음

-데이터 중복에 따른 정합성 오류 발생가능성 제거

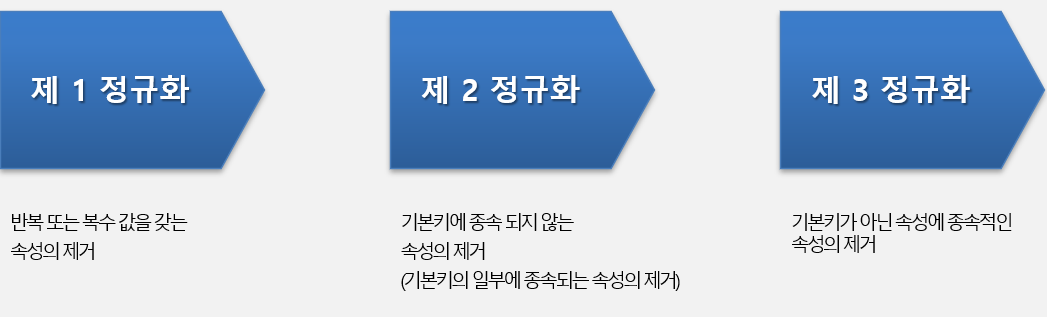
-성능 개선

\*정규화의 기본 원칙은 하나의 테이블에는 중복된 데이터가 없도록 하는 것

-------------------------------------------------------------

*정규화 규칙*

효율적인 관리를 위한 데이터구조를 도출하기 위하여, 속성간의 관계를 분석하여 별도 엔티티 도출 여부를 결정해 나가는 절차에 관련된 이론 임



-------------------------------------------------------------

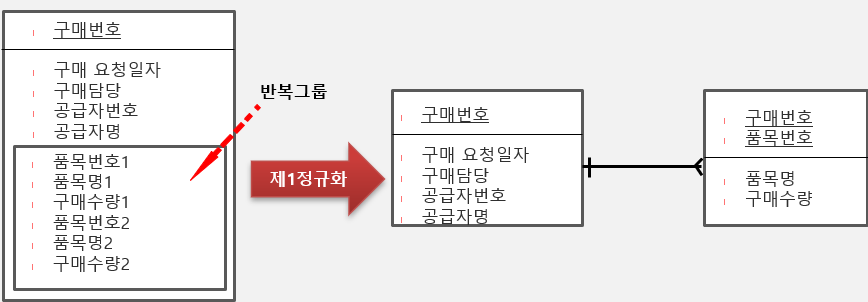
*제1정규화*

-반복그룹의 속성을 별도의 엔티티로 분리

-작업 절차

반복하는 속성 그룹으로 새로운 엔티티 작성

새로운 엔티티 키(Concatenate Key) 및 관계를 정의

\* 반복되는 컬럼이 있으면 새로운 엔티티로 만들어 관계(두 엔티티 사이에 중복되는 컬럼이 있어야 함) 설정

-------------------------------------------------------------

*제2정규화 \*복합키가 있는 경우*

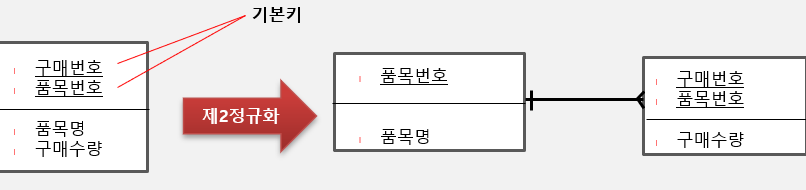
-기본키에 종속되지 않는 모든 속성을 분리하여 새로운 엔티티를 구성

-Concatenated Key가 있는 엔티티에 적용됨.

-작업 절차

속성 중에 Key의 일부에만 종속하는 속성을 분리한다. -> 새로운 엔티티를 정의한다. -> 새로운 엔티티 키 및 관계를 정의

\*(1)반복 (2)엔티티를 갈라서 복합키를 설정한다



-------------------------------------------------------------

*제3정규화 \*복합키가 아닌 경우*

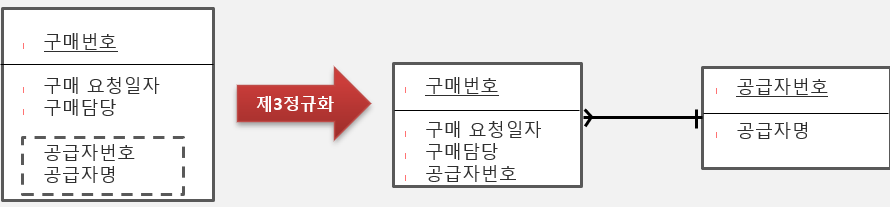
-엔티티의 속성 중에 기본 키가 아닌 속성에 종속하는 모든 속성을 분리하여 새로운 엔티티를 생성

-작업 절차

속성 중에 Key가 아닌 속성에 종속하는 속성을 분리한다.

새로운 엔티티를 정의한다.

새로운 엔티티 키 및 관계를 정의



\*관련성이 떨어지는 식별가능한 속성들을 새로운 엔티티에 정의

\*새로운 엔티티의 key가 기존 엔티티의 FK로 연결고리

**메타 데이터의 관리**

cf.

데이터 조작어 DML : SELECT, INSERT, UPDATE, DELETE

데이터 정의어 DDL : CREATE, ALTER, DROP, RENAME

-------------------------------------------------------------

*데이터 표준화*

표준화란? 이음동의어와 동음이의어를 관리하는 것

메타데이터란? 데이터들의 데이터

통일시켜 일관되게 사용되도록 하는 것

주로 데이터를 담는 최소 단위인 속성의 표준화

-------------------------------------------------------------

cf.

FROM A팀테이블 a, B팀테이블 b -> ',' : inner join

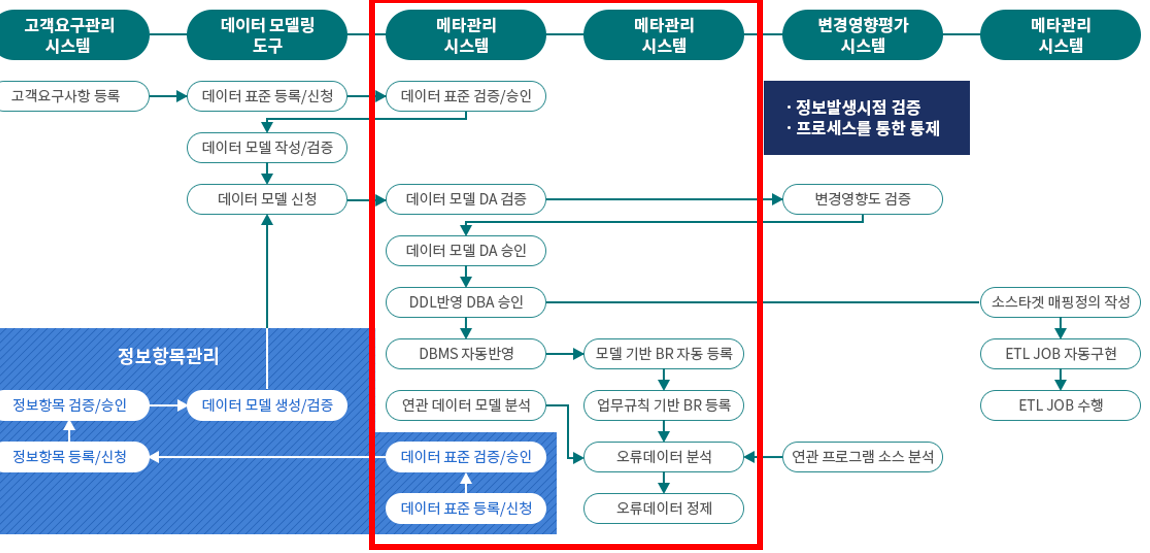
WHERE a.코드 = b.코드

(=)

A팀테이블 a

INNER JOIN B팀테이블 b

ON a.코드 = b.코드



데이터타입이 다르면 조인이 안됨

cf.

인덱스기법 : 데이터를 빠르게 찾게 해주는 효율적인 방법 (PK와 다름)

Full Scan : 무식하게 찾는 방법 스캔성능 쓰레기 속도가 극악..

-------------------------------------------------------------

*코드*

식별자(보통 PK)와 식별자가 아닌 것을 구분해야 한다.

코드 : 오직 릴레이션의 튜플들에 대해 특정 기준 중심의 분류/집합.

표준화 – 식별자에는 ID나 번호, 코드에만 코드

-------------------------------------------------------------

*개별코드와 통합코드*

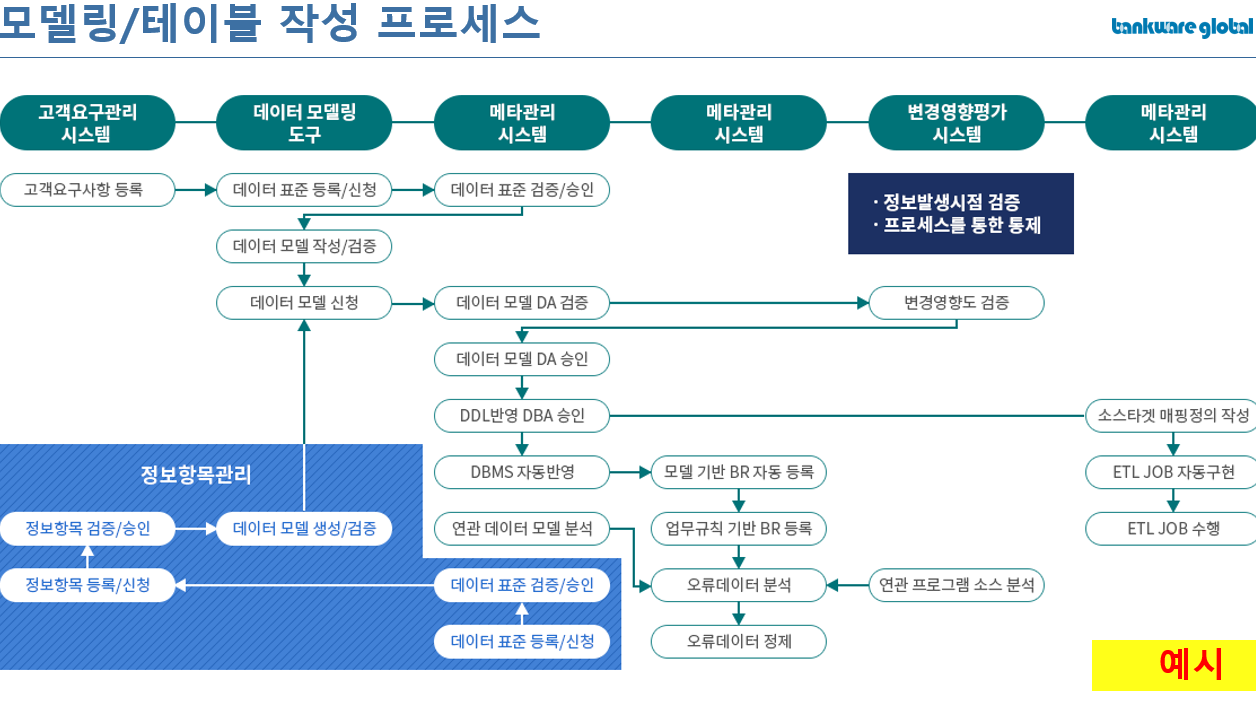
**

데이터를 통합 관리함으로써, 등록/참조/분배 등 관리상 장점

코드 사용빈도가 하나의 테이블에 응집 – 메모리 성능상의 이점 기대

어플리케이션 아키텍처를 고려한 선택 필요

**모델링 프로세스 실무**



**데이터의 품질 관리**

*품질관리의 기준*

모델링에는 정답이 없다. 평가 및 측정 또한 개발 후 운영되어지면서 증명

현황에 대한 다양한 기준을 마련하여, 해당 기준 충족여부를 통해 품질 가늠

모델링 품질 기준

-표준화 관점

업무에서 결정된 표준항목의 준수 여부로 판단

표준용어와 표준도메인 사용여부 확인

네이밍룰과 필 수 입력항목의 준수여부 확인

-모델링 방법론 관점

일반적인 모델링 룰의 준수 여부로 판단

중복여부 확인

관계선 지정 룰 준수 여부 확인

인덱스 룰 확인

정규화 여부 확인

-업무적 관점

기업의 업무를 정확하게 표현했는지의 여부로 판단

담당 업무 DA들에 의한 업무 표현의 정확도 확인

*역할별 품질관리 범위*

**

**데이터 모델링 도구**

Exerd – 토마토시스템

DA# - 엔코아

MS Visio 등

**인증 제도**

데이터 인증 제도 – 품질, 관리, 보안인증

데이터 전문가 자격 검정

데이터 요건분석, 데이터 표준화, 데이터 모델링, 데이터베이스 설계와 이용에 대한 전문지식 및 실무적 수행 능력 평가

