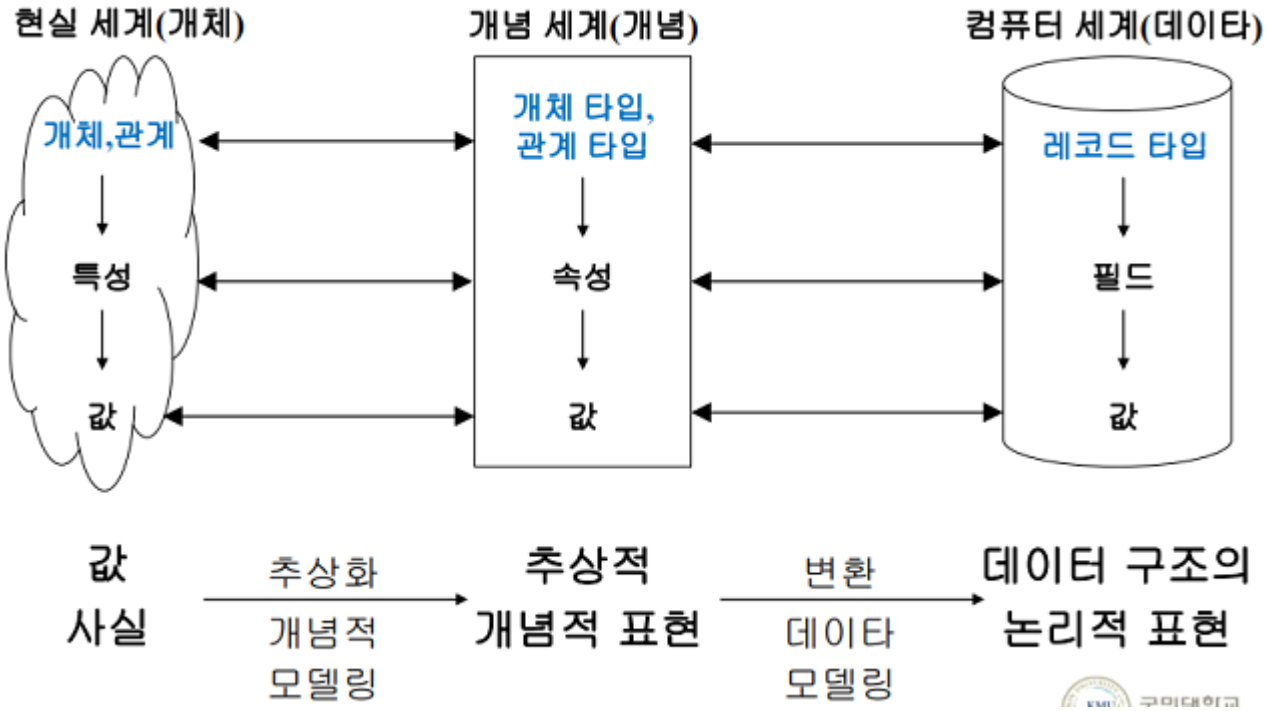
데이터베이스 모델링

데이터베이스 모델링의 단계

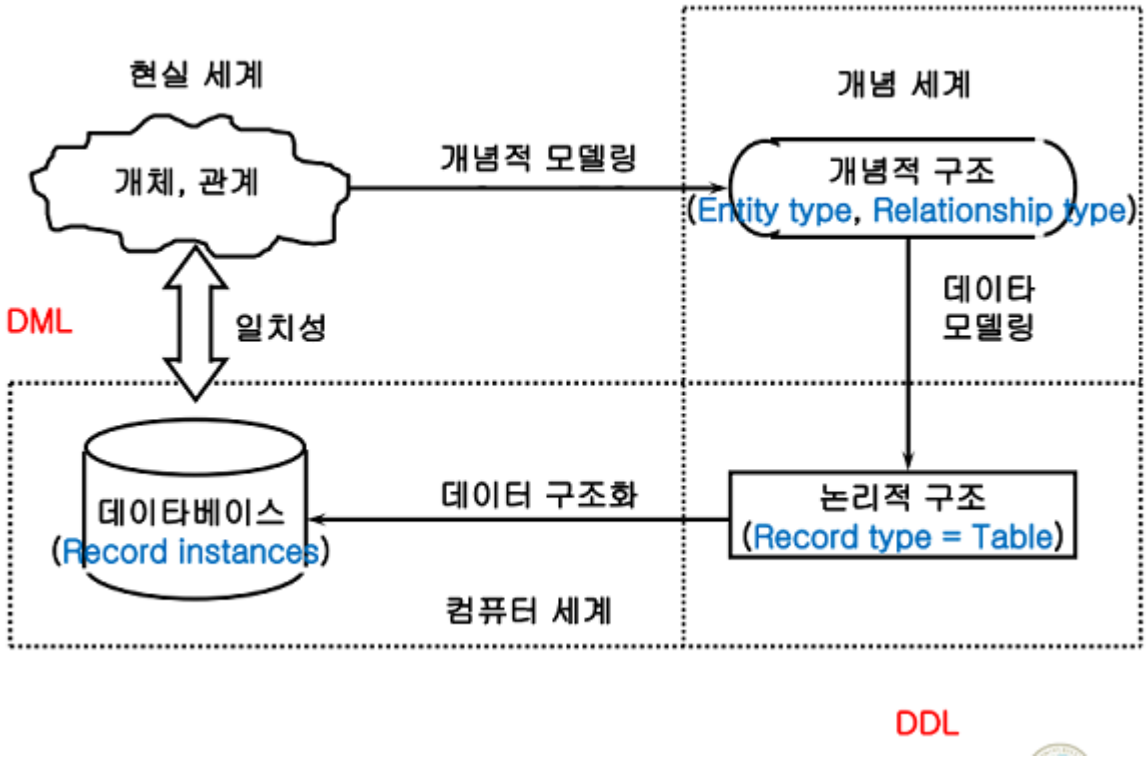
데이터의 세계

- 현실 세계 : 개체, 관계, 사실
- 개념 세계 : 개체타입, 관계타입, 추상적/개념적 표현
- 컴퓨터 세계 : 레코드 타입, 데이터 구조의 논리적 표현



현실세계의 모델링

- 개념적 설계 : 개념적 모델링 -> **개념적 표현** => Entity Type & Relationship Type
- 논리적 설계 : 데이터 모델링 -> **논리적 표현** : 접근방법에 독립적인 표현 => Record Type
- 물리적 설계 : 데이터 구조화 -> 구현 : 저장장치에서의 데이터 표현 => Physical Record Type



데이터 모델

개념 : 데이터 모델(D) = <S,O,C>

- Structure : 데이터의 구조 -> 정적 성질(추상적 개념), 개체타입과 이들 간의 관계를 명세
- Operation : 연산 -> 동적 성질, 개체 인스턴스를 처리하는 작업에 대한 명세, 데이터의 조작 기법
- Constraint : 제약 조건 -> 데이터 조작의 한계를 표현한 규정
    - 데이터의 논리적 제약 : 개체 인스턴스의 허용 조건, 구조(Structure)로부터 파생, 의미상 제약

개체 타입

- 이름과 애트리뷰트로 정의
- 개체 집합 : 특정 개체 타입에 대한 인스턴스 집합
- 애트리뷰트의 유형
    - 단순 / 복합 애트리뷰트
    - 단일값 / 다중값 애트리뷰트
    - 저장 / 유도 애트리뷰트
- null 값은 Not Applicable, Unknown(Missing, Not known)을 의미

관계 타입과 제약조건

관계타입

- 개체 집합들 사이의 대응성, Mapping(사상)
- [NOTE] 기본키(PK) : 모든 값들이 Unique, 흔히 말하는 '키'가 이 기본키 외래키(FK) : 대응되는 기본키의 값을 갖거나 null값을 가짐, 실제로 기본키 역할을 하지는 못함

구조적 제약조건

- 엔티티 인스턴스 간의 사상(Mapping) - Cardinality Ratios
    - 관계의 분류 기준 = 사상 원소수
        - 1:1(일대일 관계) : f(x) : x-> y AND f(y) : y->x
        - 1:n(일대다 관계) : f(x) : x->y **XOR** f(y) : y->x
        - n:m(다대다 관계) : neither f(x) : x-> y nor f(y) : y->x
- 사상에의 참여(Participations)
    - 전체 참여 : A-B 관계에서 개체 집합 B의 모든 개체가 A-B관계에 참여해야 할때 B가 A-B의 전체참여
    - 부분 참여 : A-B 관계에서 개체 집합 B의 일부 개체만 A-B에 참여해도 된다면 B가 A-B의 부분참여

개체-관계 모델

- 현실세계에서의 개념적 표현
- 개체타입과 관계타입을 기본 개념으로 현실 세계를 개념적으로 표현
- 개체 집합 : 한 개체 타입에 속하는 모든 **개체** 인스턴스
- 관계 집합 : 한 관계 타입에 속하는 모든 **관계** 인스턴스

E-R 다이어그램

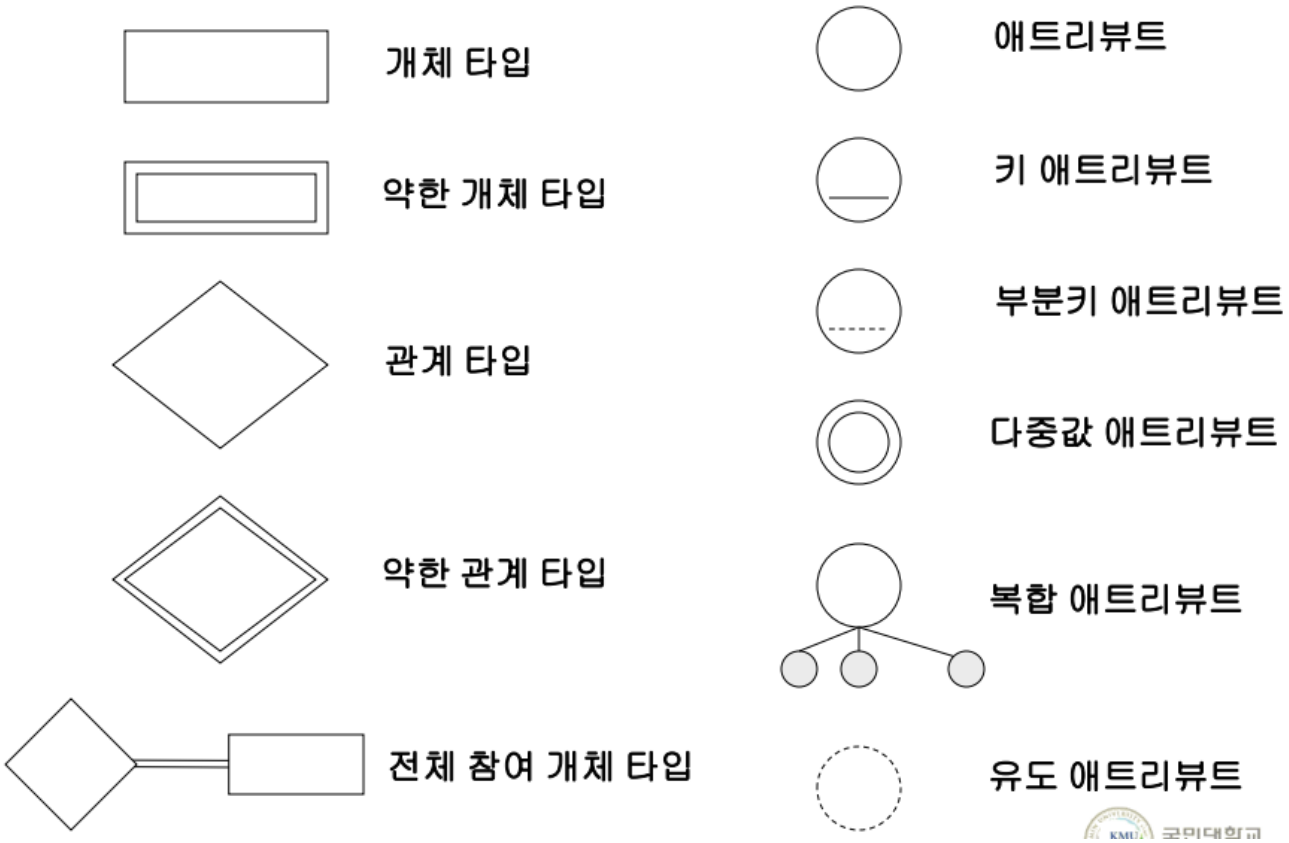
- 개체 타입 : 네모
    - 키 애트리뷰트 : 개체 타입내의 모든 인스터스를 유일하게 식별
    - 동일한 키 값을 가진 두개의 인스턴스는 없음
    - 밑줄로 표시
- 관계 타입 : 마름모
- 속성 : 동그라미
- 링크 : 선
- 레이블 : 관계의 사상을 표현(1:1, 1:n, m:n), 순환적 관계에서 역할 이름

특징

- 관계타입도 속성을 가질 수 있음
- 다원관계 표현 : 두개 이상의 개체 타입이 하나의 관계에 관련 가능
- 다중관계 표현 : 두 개체 타입 사이에 둘 이상의 관계가 존재 가능
- 존재종속 : 어떤 객체 b의 존재가 객체 a의 존재에 달려있는 경우
    - 객체 a를 삭제하면 b도 삭제 해야하는 제약조건 적용
    - Strong Entity Type(Regular/Owner) : 기본키가 있는 엔티티
    - Weak Entity Type : 기본키는 없고 Partial 키만 있음
    - Identifying Relationship
        - Strong-Weak Entity를 연결하는 관계
        - Strong Entity의 기본키가 **Identifying Attribute**
    - Weak Entity Type은 Identifying Relationship에 **항상 전체 참여**
    - 더블마름모로 표현
- 순환적 관계 : 역할 이름 표현

- 1(상급자):n(하급자) 같은 식으로
- 하나의 테이블로 표현(1에 해당하는 값을 n 인스턴스의 필드로 추가)

표기법



확장된 개체-관계 모델(Extended E-R Model)

- 복잡한 현실세계를 조금 더 정교하게 모델링
- 주요개념 : 세분화 / 일반화 / 집단화
- 복잡한 표기법 필요

ERD의 용도

- 개념적 디자인 용도로 설계 됐으나, 논리적 디자인 용도로 사용 가능
  - 개념 : PK 표현, FK 표현 X
    - Chen, Min-Max, IE, Barker's Notation
  - 논리 : PK, FK 모두 직접 표현
    - IDEF1X, EER Diagram(MySQL)

논리적 설계 : 관계 데이터 모델

- 개념적 데이터 모델 : 개체-관계 데이터 모델, 현실 세계를 추상적으로 표현한 구조(ERD)
- 논리적 데이터 모델
  - 개념적 구조를 저장 데이터베이스에 사상시키기 위해 논리적 구조로 표현하기 위한 일련의 규칙
  - 관계 데이터 모델

관계 데이터 모델

- 데이터베이스 = 릴레이션의 집합, 개체 릴레이션 / 관계 릴레이션 / 애트리뷰트 릴레이션
- 관계 스킴 : 개체와 관계성의 테이블 정의(검색 효율성 고려 X)

ER다이어그램 -> 관계스킴 변환

- 원칙 : 검색 효율성을 고려
    - 릴레이션의 수가 작을수록 검색 연산이 효율적 -> 전체 릴레이션의 수를 줄이는 방향으로 변환
- 변환 방법
    - Entity Type : 릴레이션으로 변환
    - 1-1 / 1-n 관계타입 : 외래키로 변환
    - n-m 관계타입 : 릴레이션으로 변환
    - 다중값 애트리뷰트 : 릴레이션으로 변환
- 주의사항 : 엔티티 타입의 유도 애트리뷰트는 레코드 타입의 **애트리뷰트 제외**
    - 유도 애트리뷰트는 ER Model(개념적 레벨)에서 사용하는 개념
    - 관계 모델에서는 해당 애트리뷰트를 포함하지 않음

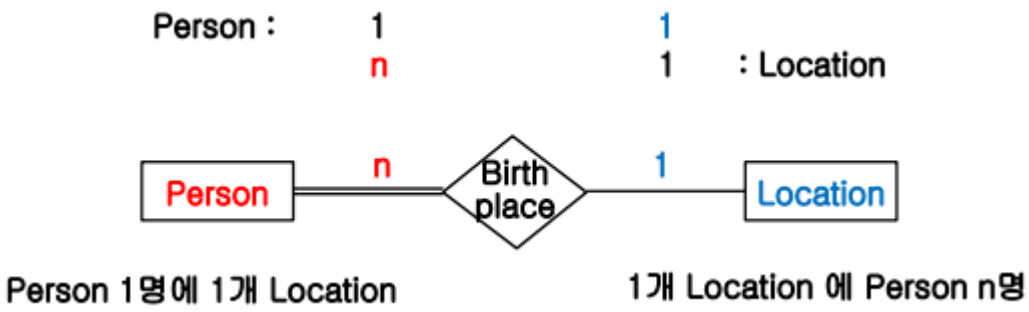
변환 알고리즘

- Entity Relation
    - 1 : Regular/Strong Entity Type
        - 릴레이션으로 생성
        - 기본키는 키 애트리뷰트 그대로 유지
        - 복합 애트리뷰트는 단순 애트리뷰트만 유지 (하나로 통합하거나, 일일히 다 나눔)
    - 2 : Weak Entity Type
        - 릴레이션으로 생성
        - Strong Entity의 키 애트리뷰트를 외래키로 포함시킴
        - 기본키 = 부분키 + 외래키(Strong Entity의 키)
        - 어느 관계와 관련된 Weak Entity 인지 확인해야 함 기본키가 다른 테이블의 외래키로 쓰일 수 있기 때문
    - 8 : Subtype
        - 별도의 릴레이션으로 생성
        - Supertype에서 Subtype의 기본키들을 각각 외래키로 추가(Surrogate KEY)
        - 또는 Subtype에 외래키로 Supertype의 기본키를 외래키로 추가(Natural KEY)
- Foreign Key
    - 3 : 1:1 Relationship Type
        - 별도의 릴레이션을 생성하지 않고, 외래키 추가
        - 두 Entity 중 하나를 선택(S), 한쪽의 키를 다른 쪽 엔티티(T)의 릴레이션에 외래키로 추가
        - 관계에 속한 속성은 모두 외래키가 추가 된 엔티티(T) 쪽으로 포함
        - 완전 참여가 있는 쪽, 없다면 튜플 개수가 현저히 작은 쪽을 T로 생각 -> 불필요한 널값의 최소화
    - 4 : 1-n Relationship Type
        - 별도의 릴레이션을 생성하지 않고, n쪽에 외래키/관계 속성 추가
    - 이 규칙에서 관계에 속하는 속성이 많을 경우, 별도의 릴레이션으로 생성 할 수도 있음
- Relationship Relation

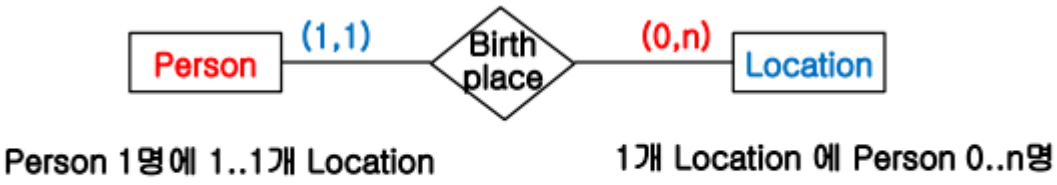
    - 5 : n-m Relationship Type
        - 별도의 릴레이션으로 생성
        - 양쪽 엔티티의 키속성을 릴레이션의 외래키로 포함
        - 기본 키는 양쪽의 키속성(외래키)들의 조합으로 구성 (해당 관계에도 키 속성이 있다면, 이것도 조합에 추가)
        - 관계 모델에서는 릴레이션 생성을 통해 두 개의 1:N관계로 표현 (직접 표현 불가능)
    - 6 : n-ary Relationship Type(3차 이상)
        - 별도의 릴레이션으로 생성
        - 모든 참여 엔티티의 키속성을 외래키로 포함
        - 참여 엔티티의 키속성들의 조합으로 기본키 구성
- Attribute Relation
    - 7 : Multi-valued attribute
        - 별도의 릴레이션으로 생성
        - 다중값 애트리뷰트 A, 소속 엔티티 타입의 키 K로 구성
        - K는 외래키
        - 개념적 레벨에서 사용하는 개념으로, 관계모델에서는 사용 X
- Identifying Relationship
    - Parent Table 없이는 Uniquely Identify 될 수 없는 Child Table
    - Child Table의 기본키가 다른 외래키를 포함 할 경우를 확인
        - n-m Relationship인 경우
        - 존재종속 관계인 경우
        - 다중값 애트리뷰트인 경우

다른표기법

- 상호간 기술적으로 전환 가능
- 단일선-이중선 표기법(Chen) : 관계 기준



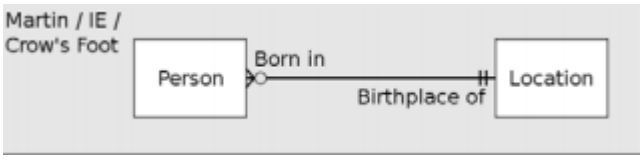
    - 단일선-이중선 -> Min-Max 변환
        - Max값은 반대방향의 숫자, Min값은 부분참여시 0, 전체참여시 1
- Min-Max 표기법 : **Entity Type** 기준 - 얘만 이럼



☛ **Note: min 값이 0이면 부분참여, 1이면 전체참여.**

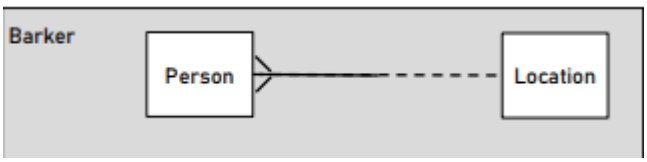
  - (0,1) : 부분참여, (0,n) : 부분참여, (1,1) : 전체참여, (1,n) : 전체참여
  - Min-Max -> IE 변환
    - Max 값 : 그대로 Crow's foot notation
    - Min 값 : **상대편이** 부분참여면 Optional, 전체 참여면 Mandatory(Line)

- IE Notation(Crow's Foot Notation) : 가장 많이 사용



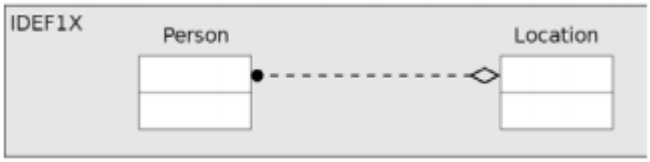
  - 까마귀 발 모양 : N을 뜻
  - 세로 줄 하나 : 하나
  - 빈 원 : Optional(없을수도 있을수도 있다는 뜻)
  - | o : Zero or One, || : One and only one, > o : Zero or many, >| : One or many
  - One to One : || - ||
  - One to Many : || - o<
  - Many to Many : >o - o<
  - 엔티티는 이름 / 속성들로 구성

- Barker's Notation : 오라클에서 사용

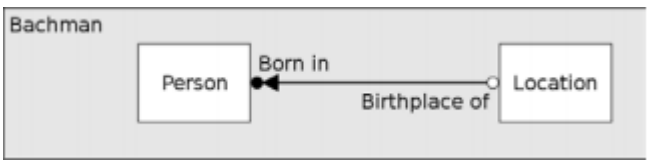


  - Entity 속성 표시 : # = UID, * = Mandatory(NOT NULL), O = Optional(NULL 허용)
  - 관계 표시
    - 전체참여 = 실선, 부분참여 = 점선
    - 1:1 = —(그냥 선), 1:n = -< (선 + 까마귀발), n:m = >-< (까마귀발 + 선 + 까마귀발)
    - 숫자 표시 앞의 UID Bar(|)는 관계가 해당 엔티티의 Identifying Relationship이라는 뜻 (반대쪽의 기본키가 해당 엔티티의 기본키에 외래키로 포함 됨)
  - 같은 데이터를 2번 쓸 필요 없음
  - 이진 관계만 표현 가능
  - Subtype Entity는 Supertype Entity 안에 그려넣음
    - 최소 하나의 외래키는 NOT NULL이여야 하고, 하나의 FK만 NOT NULL일 수 있다

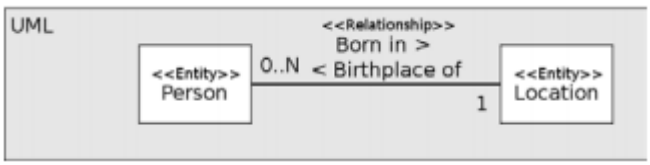
- IDEF1X



    - 미국 공군에 의해서 개발
    - IDEF(Integration Definition)
        - IDEF0 : Function Modeling
        - IDEF1 : Information Modeling
        - IDEF2 : Dynamics Modeling
    - 이론적 근간 : 관계 모델, ER Model
- EER Diagram : MySQL에서 사용
    - Crow's foot notation 사용 - 논리적 디자인을 위해서
        - 다대다 관계가 없고, 외래키가 명쾌하게 나와있다
    - Identifying Relationship 표현
        - 실선 = Identifying Relationship
        - 점선 = Non-Identifying Relationship
        - 다대다인 경우 존재종속이 걸린쪽은 1~n : 1~1로 표현
- Bachman



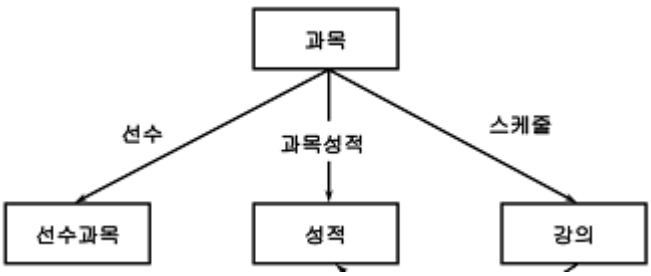
- UML



논리적 설계 : 계층 및 네트워크 모델

자료구조도

- Bachman Diagram



- 레코드 타입 간의 관계에 대한 도형적 표현
    - 화살표 : 레코드 타입간의 1:N관계를 표현
- 구성요소
    - 사각형 노드 : 레코드 타입(개체 타입)

    - 직선 : 레코드 타입 간의 일대다 관계
    - 레이블 : 관계 이름
- Schema Diagram으로도 사용 가능함 : 스키마의 도형적 표현
    - 트리형태 -> 계층 데이터 모델, 그래프 형태 -> 네트워크 데이터 모델

네트워크 데이터 모델

- 스키마 다이어그램이 **네트워크(그래프)**
- 허용되는 레코드 타입, 관계성을 명세
- Owner - Member 관계 (두 레코드 타입의 1:N 관계)

계층 데이터모델

- 스키마 다이어그램이 **트리**
- 사이클이 없음
- 루트 레코드, 자식 레코드, 레벨 존재
- 자식 - 부모 관계 (1:N 관계의 두 레코드 타입)