**개념 데이터 모델링**

**개념 데이터 모델링 순서**

- 주제 영역 도출 -> 후보 엔터티 선정 -> 수집된 엔터티 분류 -> 핵심 엔터티 정의 -> 핵심 엔터티 관계 설정

**개념 데이터 모델 정의**

- 개념 데이터 모델은 주요 핵심 엔터티들로 구성

- 핵심 엔터티란 행위의 주체나 목적물이 되는 개체 집합에 해당하는 엔터티

- 부모가 존재하지 않는 창조된 집합 -> 키 엔터티

- 핵심 엔터티는 대체적으로 여러 가지 하위의 행위 엔터티를 탄생

**개념 데이터 모델의 의의**

- 주요 엔터티로 한정을 지을 뿐, 논리 모델링 기법과 동일.

- 개념 데이터 모델링 이후 모델링 상세화가 진행되며 개념 데이터 모델도 상세화가 가능

- 상세화 되더라도 전체적인 골격의 개념적 모델을 벗어나서는 안됨

**데이터 아키텍처 프레임워크 상에서 개념 데이터 모델**

- 개괄 레이어와 논리 레이어 사이

- 개괄 데이터 모델 : 주제 영역 상위

- 개념 데이터 모델 : 주제영역 하위 (주제 영역내의 핵심엔터티)

**관계 정의**

- 엔터티와 엔터티 사이의 관계

**1. 관계 이해**

- 관계도 집합이다.

- 직접 관계를 관계라 한다.

- 두 엔터티 간에는 하나 이상의 관계가 존재할 수 있다.

- 외래키로 정의

- 관계의 관점

**2. 항상 두 엔터티 간에 존재한다.**

- 항상 두 개의 관점을 가지고 있다.

- 데이터의 양방향 업무 규칙을 표현

- 관계를 통하여 정보로서의 활용 가치 상승

- 외래키로 구현되어 참조 무결성으로 데이터의 정합성 유지

- 참조 무결성

**3. 관계 표현**

- 선택 사양

1) 일반적인 형태

- 1측 엔터티 : 선택적 / -M쪽 엔터티 : 필수적

- 가장 많이 나타나는 형태일 뿐, 꼭 바람직한 형태는 아니다. => 집계 엔터티 경우 반대

2) 바람직한 형태

- 가능한 직선 관계를 가지도록 노력. 특히 자식 엔터티

- 부모 엔터티에 개체가 존재하지 않는 자식 엔터티의 개체가 많이 발생할 경우 정보의 정합성에 많은 문제점 발생

**관계명**

1) 두 개의 관계 멤버십에 각각 부여 – 각자 상대방 입장에서의 관계명 기술

2) 현업에서 사용하는 간결한 동사형으로 표현

- 두 엔터티간의 업무적 연관성을 나타내는 이름 부여

- 현재 시제 사용

- 다른 관계명과의 유일성은 확보되지 못함

- 능동 / 수동형의 사용은 가급적 배제

3) 업무적 의미가 없거나 애매모호한 용어는 배제

**4. 관계 정의 방법 : 역할 + 부모속성 명칭**

**5. 관계형태**

|  |  |  |  |
| --- | --- | --- | --- |
|  | 1 : 1 | M : 1 | M : M |
| 특징 | - 현실에서 매우 드뭄  - 업무의 흐름에 따라 설계된 형태에서 많이 나타남  - 엔터티 수직 분할 시에 많이 나타남 | - 가장 흔하게 나타나는 관계의 형태 | - 양쪽 모두 점선인게 기본형  - 의외로 빈번하게 발생하나 모델링 완료후에는 존재 X.  - 분해시 항상 1:M 이 되는건 아니며 이때 M:M 해소될 때까지 분해 필요 |
| M : O | 컴퓨터 - 메인보드 | - 현실 세계에서 가장 흔한 형태  - 부서 – 사원 관계 |  |
| M : M | 두 엔터티는 동치 | 주문 – 주문상품 관계 |  |
| O : O | 자주 발생 X | 현실세계에서 가장 흔한 형태 중의 하나. 선택성을 해소해야 함 |  |
| O : M |  | 대금청구하는 경우 |  |

다중 관계 처리 (병력식 vs 직렬식)

|  |  |  |
| --- | --- | --- |
| 구분 | 병렬식 | 직렬식 |
| 장점 | * 테이블이 될 때 여러 개의 컬럼으로 나열 * 하나의 로우에서 관리되므로 새로운 테이블 추가 필요 | * 관계들이 로우 형태로 나타남 * 인덱스 수가 감소하고 SQL이 단순해진다. * 새로운 관계의 추가, 관계 형태의 변경 등에 매우 유연한 형태 * 관계 내용별로 상세 정보 관리 가능 = 자식 엔터티를 거느릴 수 있다 |
| 단점 | * 인덱스 수가 증가되고 SQL 복잡 * 새로운 관계의 추가, 관계 형태 변경 등에 매우 취약한 형태 * 관계 내용별로 상세 정보를 관리할 수 없다. | * 관계들을 관리하는 새로운 엔ㅋ터티가 추가 필요 |

**특수한 형태 관계**

1) 순환 관계

- 하나의 순환 엔터티는 각 엔터티의 모든 속성을 포함해야 한다.

- 각 계층에 있는 속성은 동일하게 하는 것이 가장 좋다.

- 순환 모델은 필수 관계로 취급 될 수 없고 반드시 선택 사양 (필수일 시 무한 루프)

- 순환 모델의 특징은 변경에 쉽게 대응할 수 있다.

- 최상위 노드의 경우 성능 향상을 위해 dummy node혹은 임의의 값을 넣을 수 있다.

2) Bill of Materials

- M : M 순환 관계

- 상세 모델링 관계에서 새로운 관계 엔터티를 추가하여 두 개의 1:M관계로 구성

3) Arc 관계

- 아크 내에 있는 릴레이션십은 보통 카디널리티가 동일

- 아크 내에 있는 릴레이션십은 항상 피수이거나 선택사양

- 아크는 반드시 하나의 엔터티에만 속해야 함

- 어떤 엔터티는 다수의 아크를 가질 수 있다. 그런 지정된 릴레이션십은 단 하나의 아크에서만 사용되야 한다.