1. 데이터베이스 기본 개념

데이터와 정보의 관계

- 데이터: 관찰이나 측정을 통해 수집한 단순한 사실이나 값
- 정보: 데이터를 처리하여 의사결정에 유용한 의미 있는 결과물로 만든 것
- DIKW 피라미드: 데이터 → 정보 → 지식 → 지혜로 발전하는 계층 구조

데이터베이스(DB)와 DBMS

- 데이터베이스: 논리적으로 연관된 데이터를 체계적으로 저장한 것
- DBMS: 데이터베이스를 관리하는 소프트웨어로, 데이터 정의/조작/제어 기능 제공
- 특징: ISOS (통합된/저장된/운영/공용 데이터) 및 R1C3 (실시간 접근성/지속적 변화/동시 공유/내용 참조)

파일 시스템 대비 장점

- 데이터 중복 최소화
- 데이터 공유 가능
- 일관성, 무결성, 보안성 유지
- 데이터 독립성 확보

2. 데이터베이스 유형 및 특징

주요 데이터베이스 유형

1. **계층형 데이터베이스**
    - 트리 구조로 데이터 구성
    - 부모-자식 관계로 표현
    - 현재는 거의 사용되지 않음
2. **네트워크형 데이터베이스**
    - 노드와 집합구조로 데이터 표현
    - OWNER-MEMBER 관계 설정
    - 다대다 관계 표현 가능

3. **키-값 데이터베이스 (NoSQL)**
    - 키와 값을 일대일 대응
    - 스키마 없이 작동
    - 비정형 데이터 저장에 유리
    - 예: Redis, Memcached
4. **관계형 데이터베이스 (RDBMS)**
    - 테이블 형태로 데이터 저장
    - 현재 가장 널리 사용
    - 예: Oracle, MySQL, PostgreSQL, SQL Server

3. 관계형 데이터 모델

릴레이션 구조

- 릴레이션 = 테이블 (행과 열로 구성)
- 스키마: 릴레이션의 구조와 제약조건 정의
- 인스턴스: 실제 저장된 데이터 집합
- 튜플(행), 속성(열), 도메인(값의 범위)

키(Key)의 종류

1. **슈퍼키**: 튜플을 유일하게 식별할 수 있는 속성 집합
2. **후보키**: 최소한의 속성으로 구성된 슈퍼키
3. **기본키(Primary Key)**: 선택된 대표 후보키, NULL 불허
4. **대리키**: 인위적으로 생성한 키
5. **대체키**: 기본키로 선정되지 않은 후보키
6. **외래키(Foreign Key)**: 다른 릴레이션의 기본키를 참조

무결성 제약조건

1. **도메인 무결성**: 속성값이 정의된 도메인 내의 값만 가능
2. **개체 무결성**: 기본키는 NULL 불가, 유일해야 함

3. **참조 무결성**: 외래키는 참조하는 릴레이션의 기본키 값만 가능
    - CASCADE, RESTRICTED, DEFAULT, NULL 옵션

4. 관계대수 (Relational Algebra)

개념

- E.F. Codd 박사가 관계 데이터 모델을 위한 언어로 소개
- 릴레이션에서 원하는 결과를 얻기 위한 절차적 언어
- SQL은 관계해석 기반이지만, DBMS 내부에서는 관계대수 기반으로 연산 수행

주요 연산자

관계 연산 (Relational Operations)

- **셀렉션(Selection, σ)**: 조건을 만족하는 튜플(행) 추출
- **프로젝션(Projection, π)**: 특정 속성(열) 추출
- **조인(Join, ⋈)**: 두 릴레이션의 공통 속성 기준으로 결합
- **디비전(Division)**: 특정 값들을 모두 갖는 튜플 찾기

집합 연산 (Set Operations)

- **합집합(Union, ∪)**: 두 릴레이션의 모든 튜플
- **교집합(Intersection, ∩)**: 공통 튜플
- **차집합(Set-difference, -)**: 한 릴레이션에만 있는 튜플
- **카티션 프로덕트(Cartesian Product, ×)**: 모든 가능한 튜플 조합

5. 트랜잭션과 동시성

트랜잭션 개념

- DBMS의 논리적 작업 단위
- "All or Nothing" 원칙 적용
- BEGIN TRANSACTION과 COMMIT TRANSACTION으로 구분

ACID 특성

1. **원자성(Atomicity)**: 전부 수행되거나 전부 취소
2. **일관성(Consistency)**: 무결성 제약조건 유지
3. **고립성(Isolation)**: 동시 수행 트랜잭션 간 독립성 보장
4. **지속성(Durability)**: 완료된 트랜잭션은 영구 보존

동시성 제어

- **갱신 손실 문제**: 동시 갱신 시 데이터 손실 방지
- **락(Lock) 메커니즘**:
    - 공유락(Shared Lock): 읽기 전용
    - 배타락(Exclusive Lock): 읽기/쓰기 모두 가능

6. 데이터베이스 보안과 권한 관리

보안 유형

물리적 보안

- 하드웨어 접근 제한
- 보안 카메라, 잠금 장치
- 이중 전원 장치

권한 관리를 통한 보안

- 로그인 인증 (1차 방어선)
- 사용자별 권한 차등 부여

권한 관리 명령어

GRANT: 권한 부여

GRANT INSERT, DELETE ON 고객 TO PUBLIC;

GRANT UPDATE(등급, 주소) ON 고객 TO Kang;

REVOKE: 권한 취소

REVOKE SELECT ON 고객 FROM Kang CASCADE;

역할(Role) 관리

- 권한들의 집합
- 역할 생성 후 사용자에게 할당
- 권한 관리의 효율성 향상

권한 종류

- 테이블/뷰: INSERT, UPDATE, DELETE, SELECT, REFERENCES
- 함수: EXECUTE, REFERENCE
- 저장 프로시저: EXECUTE

7. 데이터베이스 모델링 개념

데이터 모델링 정의

- 조직의 정보 수집과 관리 시스템을 정의하는 시각적 표현(청사진) 생성 프로세스
- 업무에 필요한 데이터를 분석하고 적절한 표기법(Notation)으로 표현하는 작업
- 데이터 간 관계, 저장 및 분석 방식을 설명

데이터 모델링 필요성

- 조직 전체의 데이터 문서화 및 시스템 설계 일관성 확보
- 업무 조직과 기술 조직 간 의사소통 촉진
- 효율적인 데이터 수집, 저장, 처리 실현
- 데이터베이스 설계 속도와 효율성 향상
- 잠재적 위험 요소 조기 발견 및 해결

모델링 관점 3가지

1. **데이터 관점** (What): 어떤 데이터와 관련 있는지, 데이터 간 관계는 무엇인지
2. **프로세스 관점** (How): 업무를 통해 무엇을 처리하는지
3. **상관 관점** (Interaction): 업무 처리 방법에 따라 데이터가 어떻게 영향받는지

데이터 아키텍트(DA)의 역할

- 비즈니스 요구사항을 기술 요건으로 변환

- 데이터 표준 관리 및 수행
- 전사적 데이터 관리 프레임워크 시각화 및 설계

8. 엔티티 타입 및 속성

엔티티(Entity) 개념

- 업무에 필요한 정보를 저장하고 관리하는 영속적 단위
- 엔티티 타입: 엔티티의 집합
- 사람, 장소, 물건, 개념을 정의할 수 있는 것
- 데이터베이스 구현 시 테이블에 해당

엔티티 타입의 6가지 특징

1. 업무에서 필요하고 관리되어야 하는 정보
2. 유일한 식별자로 식별 가능
3. 2개 이상의 엔티티로 구성된 집합
4. 업무 프로세스에서 사용
5. 반드시 속성(Attribute) 포함
6. 다른 엔티티 타입과 최소 한 개 이상의 관계 포함

엔티티 타입 분류

성격에 따른 분류:

- 유형(Tangible): 물리적 형태 있음
- 개념(Conceptual): 물리적 형태 없음
- 사건(Event): 업무 수행에 따라 발생

발생 시점에 따른 분류:

- 기본 엔티티: 독립적으로 생성
- 중심 엔티티: 기본 엔티티에서 발생
- 행위 엔티티: 두 개 이상 부모 엔티티에서 발생

속성(Attribute)

- 엔티티에서 더 이상 분리되지 않는 최소 데이터 단위
- **분류:**
    - 기본 속성: 업무 분석을 통해 정의
    - 설계 속성: 설계 과정에서 도출
    - 파생 속성: 계산이나 변형으로 생성

9. 식별자 및 관계 개념

식별자(Identifier)

- 엔티티 타입 내에서 각 엔티티를 구분하는 결정자
- 데이터베이스 구현 시 PK, FK 역할 수행

식별자 구분

1. **주/보조 식별자**
    - 주 식별자: 대표성을 나타내는 유일한 식별자 (PK)
    - 보조 식별자: 보조적으로 엔티티 식별 (유니크 인덱스)
2. **내부/외부 식별자**
    - 내부: 자체적으로 생성
    - 외부: 다른 엔티티로부터 상속 (FK)
3. **단일/복합 식별자**
    - 단일: 하나의 속성으로 구성
    - 복합: 두 개 이상 속성으로 구성
4. **원조/대리 식별자**
    - 대리: 복합 식별자를 하나의 속성으로 만들어 사용

관계(Relationship)

- 두 엔티티 타입 사이의 논리적 관계
- 업무 흐름을 나타냄
- 데이터 무결성 보장에 중요

관계 카디널리티

- **1:1 (One to One)**: 각 엔티티가 하나의 관계만 가짐

보안상 분리가 필요한 경우

대용량 컬럼 분리 (BLOB, CLOB)

자주 사용하지 않는 속성 분리

- **1:N (One to Many)**: 한 엔티티가 여러 관계를 가짐

계층 구조 표현

소유/포함 관계

분류/카테고리 관계

- **M:N (Many to Many)**: 양방향 모두 여러 관계 가짐

양방향 다중 관계

관계 자체에 속성이 필요한 경우

시간에 따라 변하는 관계

10. 데이터 모델링 유형 (1)

데이터 모델링 단계

1. **요구사항 분석**: 업무 및 기능 수행에 필요한 정의 분석
2. **개념 모델링**: 핵심 엔티티 도출 및 관계 정의 (개념 ER 모델)
3. **논리 모델링**: 업무 요건을 명확하게 표현, 설계 (상세 ER 모델)
4. **물리 모델링**: 구현할 DBMS 특징에 맞게 표현
5. **데이터베이스**: 물리적 모델을 데이터베이스에 구현

데이터 모델링 관점

- **데이터 관점**: Entity Type을 활용한 업무와 데이터 간 관계 모델링

- **프로세스 관점**: SQL을 통한 업무 처리 양식 모델링
- **상관 관점**: CRUD Matrix 기법을 적용한 상관관계 모델링

11. 데이터 모델링 유형 (2)

논리적 데이터 모델링

- 개념적 데이터 클래스를 기술적 데이터 구조에 매핑
- 데이터 유형, 엔티티 간 관계, 키 필드 등 상세 내용 제공
- 정규화 과정 포함

물리적 데이터 모델링

- 특정 DBMS 기술에 맞춰 구현
- 데이터 타입, 크기 선정
- 인덱스, 파티션 등 성능 최적화 고려

ERD (Entity Relationship Diagram)

- 시스템의 엔티티와 관계를 시각적으로 표현
- 1:1, 1:N, M:N 관계 표현
- 논리적 ERD와 물리적 ERD로 구분

12. 데이터 모델링 정규화 (1)

정규화 개념

- 데이터 중복을 제거하고 구조화하는 과정
- 1972년 E.F. Codd 박사가 제안

정규화 종류

1. **1차 정규화**: 복수의 속성값을 갖는 속성 분리 (원자값 보장)
2. **2차 정규화**: 부분 종속 속성 분리 (복합 키에서 발생)
3. **3차 정규화**: 이전 종속 속성 분리
4. **BCNF (보이스-코드 정규화)**: 다수의 주식별자 분리

5. **4차 정규화**: 다가 종속 속성 분리
6. **5차 정규화**: 결합 종속 분리

정규화 예시

- 학생-취미 테이블을 1차 정규화하여 분리
- 점수 테이블을 2차 정규화하여 학생 정보 분리

3차 정규화

- **정의**: 속성 간 종속 관계를 분리하여 이행적 함수 종속을 제거
- **목적**: 기본키가 아닌 속성들 간의 종속 관계 제거
- **예시**: 학생 테이블에서 학과 정보를 별도 테이블로 분리

보이스-코드 정규화 (BCNF)

- 3차 정규화를 보완한 형태
- 복잡한 식별자 관계 문제 해결
- 기본키가 아닌 속성이 기본키를 결정할 수 없도록 조정

4차 정규화

- 다치 종속(Multi-valued Dependency) 제거
- 독립적인 관계를 별도 테이블로 분리
- 예: 학생의 강의와 취미를 별도 테이블로 관리

5차 정규화

- 조인 종속 제거
- 데이터 손실 없이 분해 가능한 만큼 완전 분해

13. 데이터 모델링 문제 및 해소

1:1 관계 해소

- 동일 PK 구조는 하나의 엔티티로 통합

- 부분 통합 또는 슈퍼타입 생성으로 해결

M:N 관계 해소

- 관계 엔티티 타입 분리 (매핑 테이블 생성)
- 주 식별자 통합
- 부모 엔티티에 속성 추가

엔티티 타입 통합

- **목적**: 모델 단순화, 중복성 제거
- **문제점**: 확장성 감소, 성능 저하 가능성, SQL 복잡도 증가

이력 엔티티 타입 설계

- **발생 이력**: 시간별 발생 데이터 관리
- **변경 이력**: 수정 전후 데이터 관리
- **진행 이력**: 프로세스 상태 변화 관리
-

14. 성능을 고려한 데이터 모델링

정규화 vs 반정규화

- **정규화**: 입력/수정/삭제 성능 향상
- **반정규화**: 조회 성능 향상 (조인 비용 절약)

성능 향상 기법

1. **PK/FK 순서 조정**
    - 복합 식별자의 컬럼 순서가 성능에 영향
    - 조회 패턴에 맞춰 PK 순서 최적화
    - FK에 인덱스 생성으로 연쇄 삭제 성능 개선
2. **테이블 분리**
    - 컬럼이 많은 테이블을 1:1로 분리
    - 로우 체이닝/마이그레이션 방지

3. **파티셔닝**
    - 대용량 테이블을 파티션 키로 분할
    - 데이터 접근 범위 축소
4. **모델 단순화**
    - 복잡한 모델은 유지보수 어려움
    - 업무 흐름에 따른 설계 권장
5. **데이터 타입 일관성**
    - 동일 속성은 동일 타입 사용
    - 암시적 형변환 방지로 인덱스 활용도 향상

핵심 결론

좋은 데이터 모델링은:

- 정규화를 충실히 따른 후 필요시 반정규화
- 데이터베이스 특성을 반영
- 비즈니스 도메인을 정확히 이해
- 성능과 유지보수성의 균형 고려