3. 데이터 입출력 구현

데이터베이스 개요

1. 데이터 저장소
   - 데이터들을 논리적인 구조로 조직화하거나, 물리적인 공간에 구축한 것
   1. 논리적 데이터 저장소

      데이터 간의 연관성, 제약조건을 식별하여 논리적 구조로 조직화 한 것
   2. 물리적 데이터 저장소

      논리 데이터 저장소를 물리적 특성을 고려하여 실제 저장장치에 저장한 것

2. 데이터 베이스 (Database)
   - 공동으로 사용될 데이터를 중복을 배제하여 통합, 저장장치에 저장하여 항상 사용할 수 있도록 운영하는 운영 데이터

3. DBMS (DataBase Management System : 데이터베이스 관리 시스템)
   - 사용자의 요구에 따라 정보를 생성해주고, 데이터베이스를 관리해주는 소프트웨어
   - 기능
     1. 정의 기능 (Definition)

        데이터 타입과 구조, 제약 조건 정의
     2. 조작 기능 (Manipulation)

        삽입, 삭제, 갱신, 검색 등의 인터페이스 수단 제공
     3. 제어 기능 (Control)

        데이터 무결성, 보안, 권한, 병행 제어 제공

4. 데이터의 독립성

   a. 논리적 독립성

      응용 프로그램과 데이터베이스를 독립시켜, 데이터의 논리적 구조를 변경시키더라도 응용 프로그램은 영향 받지 않음

   b. 물리적 독립성

      응용 프로그램과 물리적 장치를 독립시켜, 디스크를 추가 및 변경하더라도 응용 프로그램은 영향 받지 않음

5. 스키마 (Schema)
    - 데이터베이스의 구조와 제약조건에 관한 전반적 명세를 기술한 것

    1. 외부 스키마

       사용자나 개발자의 입장에서 필요로 하는 데이터베이스의 논리적 구조를 정의
    2. 개념 스키마

       데이터베이스의 전체적인 논리적 구조를 정의
    3. 내부 스키마

       물리적 저장장치 입장에서 본 데이터베이스 구조

       실제로 저장될 레코드 형식, 데이터 표현 방법, 내부 레코드의 물리적 순서를 나타냄

데이터베이스 설계

1. 설계 시 고려사항
    a. 무결성

       삽입, 삭제와 같은 연산 후에도 저장된 데이터는 정해진 제약 조건을 만족해야 함
    b. 일관성

       데이터는 처음부터 끝까지 변함없이 일정해야 함
    c. 회복

       장애가 생겼을 때 복구할 수 있어야 함
    d. 보안

       데이터의 노출, 변경같은 손실로부터 보호되어야 함
    e. 효율성

       응답시간 단축, 저장 공간 최적화가 가능해야 함
    f. 데이터베이스 확장

       지속적으로 데이터 추가가 가능해야 함

2. 데이터베이스 설계 순서
    a. 요구 조건 분석 - 요구 조건 명세서 작성
    b. 개념적 설계 - 개념 스키마, 트랜잭션 모델링, E-R 모델
    c. 논리적 설계 - 목표 DBMS에 맞는 논리 스키마 설계, 트랜잭션 인터페이스 설계
    d. 물리적 설계 - 목표 DBMS에 맞는 물리적 구조의 데이터로 변환
    e. 구현 - 목표 DBMS의 데이터 정의어 (DDL)로 데이터베이스 생성, 트랜잭션 작성

데이터 모델의 구성요소

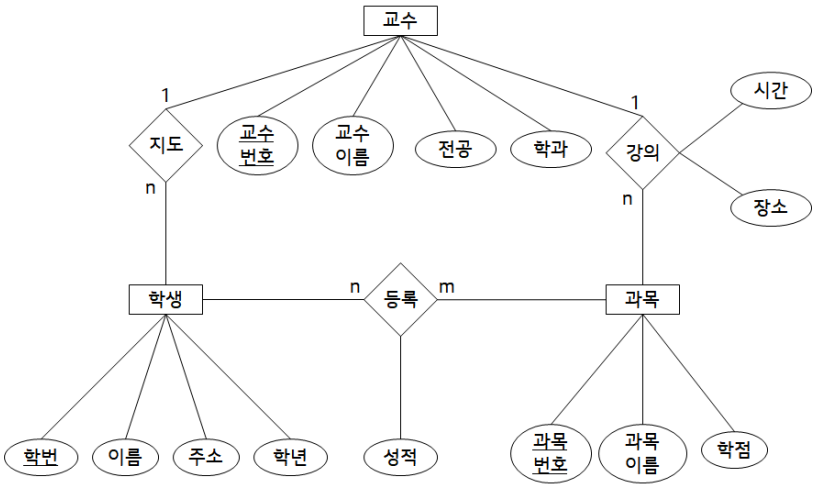
1. 개체 (Entity)
    - 데이터베이스에 표현하려는 것, 개념이나 정보 단위 같은 현실 세계의 대상체
    - 다른 개체와 하나 이상의 관계 (Relationship)를 가짐

2. 속성 (Attribute)
    - 데이터베이스를 구성하는 가장 작은 논리적 단위
    - 속성의 수 = Degree = 차수

3. 관계 (Relationship)
    - 개체와 개체 사이의 논리적 연결
    - 1:1, 1:N, N:M

E-R 모델

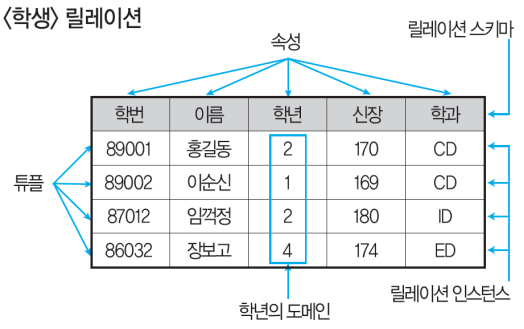
- 개체와 개체 간의 관계를 기본요소로 데이터를 개념적인 논리 데이터로 표현하는 방법
- Peter Chen에 의해 제안
- 개념적 데이터 모델의 가장 대표적인 것



- 개체 : 교수, 학생, 과목
- 관계 : 지도, 등록, 강의
- 기본키 : 교수 번호, 학번, 과목번호

관계형 데이터베이스 구조 / 관계형 데이터 모델

1. 관계형 데이터베이스
    - 표를 이용하여 데이터 상호 관계를 정의하는 데이터베이스



1. 튜플 (Tuple)

   릴레이션의 행

   = 카디널리티

2. 속성 (Attribute)

   데이터베이스를 구성하는 가장 작은 논리적 단위

   = Degree = 차수

3. 도메인 (Domain)

   하나의 속성이 취할 수 있는 같은 타입의 원자값들의 집합

   학년 (1, 2, 4)

2. 릴레이션의 특징

   a. 같은 튜플이 포함될 수 없다.

   b. 튜플 사이의 순서가 없다

   c. 속성들 간의 순서는 중요하지 않다

   d. 속성의 명칭은 유일하지만 구성하는 값은 같은 값이 있을 수 있다.

   e. 유일하게 식별되는 튜플을 위해 속성들의 부분집합을 Key로 설정

   f. 속성의 값은 더 이상 쪼갤 수 없는 원자값만을 저장

3. 관계형 데이터 모델

   - 2차원적인 표를 이용하여 데이터 상호 관계를 정의하는 DB 구조
   - 기본키와 이를 참조하는 외래키로 데이터간의 관계를 표현
   - SQL이 대표적인 언어

관계형 데이터베이스 제약조건

1. 키 (Key)

   - 조건에 만족하는 튜플을 찾거나 순서대로 정렬할 때 기준이 되는 속성

   1. 후보키 (Candidate Key)

      : 속성들 중 유일하게 튜플을 식별하기 위해 사용되는 속성들의 부분집합

      a. 유일성 - 하나의 키는 하나의 튜플을 유일하게 식별

      b. 최소성 - 키를 구성하는 속성 하나를 제거하면 식별할 수 없도록 필요한 속성으로 구성

   2. 기본 키 (Primary Key)

      : 후보키 중 특별히 선정된 주 키 (Main Key)

      : 중복된 값을 가질 수 없음

      : Null을 가질 수 없음

3. 대체 키 (Alternate Key)
   : 후보키가 둘 이상일 때 기본키를 제외한 나머지 후보키

4. 슈퍼 키 (Super Key)
   : 속성들의 집합으로 구성된 키
   : 모든 튜플에 대해 유일성은 만족하지만 최소성은 만족하지 못함

5. 외래 키 (Foreign Key)
   : 다른 릴레이션의 기본키를 참조하는 속성 또는 속성의 집합

2. 무결성 (Integrity)
   - 데이터베이스에 저장된 데이터 값과 현실 세계의 실제값이 일치하는 정확성
   - 종류
     1. 개체 무결성
        : 기본키를 구성하는 어떤 속성도 Null이나 중복값을 가질 수 없다
     2. 참조 무결성
        : 외래키는 Null이거나 참조하는 릴레이션의 기본키 값과 동일해야 함
     3. 도메인 무결성
        : 주어진 속성 값이 정의된 도메인에 속한 값이어야 함
     4. 사용자 정의 무결성
        : 속성 값들이 사용자가 정의한 제약조건에 만족되어야 함
     5. NULL 무결성
        : 릴레이션의 특정 속성 값이 NULL이 되면 안됨
     6. 고유 무결성
        : 특정 속성에 대해 각 튜플이 갖는 속성 값들이 달라야 함
     7. 키 무결성
        : 릴레이션에 적어도 하나의 키가 존재해야 함
     8. 관계 무결성

관계대수 및 관계해석

1. 관계대수
   - 관계형 데이터베이스에서 원하는 정보와 그 정보를 검색하기 위해 어떻게 유도하는가를 기술하는 절차적 언어
   - 릴레이션을 처리하기 위해 연산자와 연산규칙 제공, 피연산자와 결과 모두 릴레이션

1. 순수 관계 연산자
   - Select
     - 릴레이션에 존재하는 튜플 중 선택 조건을 만족하는 튜플들을 구하여 새로운 릴레이션을 만듦
     - σ(조건)(R)
   - Project

        - 주어진 릴레이션에서 속성 리스트에 제시된 속성 값만을 추출하여 새로운 릴레이션을 만듦
        - π(속성 리스트)(R)
    - Join
        - 공통 속성을 중심으로 두 개의 릴레이션을 하나로 합쳐 새로운 릴레이션을 만듦
        - R ⋈(JOIN 조건) S
    - Division
        - Y⊆X인 두개의 릴레이션 R(X), S(Y)가 있을 때 R의 속성이 S의 속성값을 모두 가진 튜플에서 S가 가진 속성을 제외한 속성만을 구하는 연선
        - R[속성r ÷ 속성s]S
2. 일반 집합 연산자 - 수학적 집합 이론에서 사용하는 연산자
    - 합집합 (UNION)
        - 두 릴레이션의 튜플의 합집합을 구하되, 중복되는 튜플은 제거
        - R∪S={t|t∈R ∨ t∈S}
    - 교집합 (INTERSECTION)
        - 두 릴레이션에 존재하는 튜플의 교집합을 구하는 연산
        - R∩S={t|t∈R ∧ t∈S}
    - 차집합 (DIFFERENCE)
        - 두 릴레이션에 존재하는 튜플의 차집합을 구하는 연산
        - R-S={t|t∈R ∧ t∉S}
    - 교차곱 (CARTESIAN PRODUCT)
        - 두 릴레이션에 존재하는 튜플들의 순서쌍 구하는 연산
        - R×S={r · s|r∈R ∧ s∈S}
        - 카디널리티는 두 릴레이션의 카디널리티의 곱

2. 관계해석 (Relational Calculus)
    - 관계 데이터의 연산을 표현하는 방법
    - 원하는 정보가 무엇이라는 것만 정의하는 비절차적 특성을 지님
    - 코드가 수학의 Predicate Calculus (술어 해석)에 기반을 두고 관계 데이터베이스를 위해 제안

이상 / 함수적 종속

1. 이상 (Anomaly)
    - 테이블에서 일부 속성들의 종속으로 인해 데이터 중복이 발생하고 테이블 조작시 문제가 발생하는 현상

    1. 삽입 이상 (Insertion Anomaly)

       : 데이터 삽입시 원하지 않는 값들로 인해 삽입할 수 없는 현상

       : 삽입 시 기본키가 반드시 있어야 하는데 기본키 값 없이 삽입하려는 경우
    2. 삭제 이상 (Deletion Anomaly)

       : 데이터 삭제시 의도치 않았던 값들 까지 같이 삭제되는 현상 (연쇄 삭제)
    3. 갱신 이상 (Update Anomaly)

: 튜플의 속성 값 갱신 시 일부 튜플의 정보만 갱신되어 불일치성이 생기는 현상

2. 함수적 종속 (Functional Dependency)
   - X → Y로 표기, Y는 X에 함수적 종속
   - X : 결정자 (Determinant), Y : 종속자 (Dependent)

수강

학번	이름	학년	학과
400	이순신	4	컴퓨터공학과
422	유관순	4	물리학과
301	강감찬	3	수학과
320	홍길동	3	체육과

   - 학번 → 이름, 학년, 학과

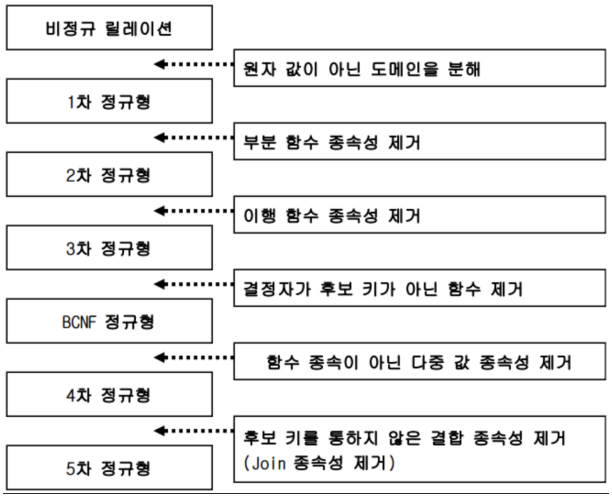
수강

학번	과목번호	성적	학년
100	C413	A	4
100	E412	A	4
200	C123	B	3
300	C321	A	1
300	C324	C	1
400	C123	A	4
400	C312	A	4
400	C324	A	4
400	C413	B	4
400	E412	C	4
500	C312	B	2

   - (학번, 과목번호) → 성적 : 완전 함수 종속
   - 학번 → 학년 : 부분 함수 종속

정규화 (Normalization) / 반정규화 (Denormalization)

1. 정규화 (Normalization)
   - 테이블의 속성들이 상호 종속적인 관계를 갖는 특성을 이용하여 테이블을 무손실 분해 하는 과정
   - 가능한 한 중복을 제거하여 삽입, 삭제, 갱신 이상을 방지



a. 제 1 정규형

: 모든 속성의 도메인이 원자값으로 구성되어있는 정규형

b. 제 2 정규형

: R이 제 1 정규형이고, 모든 속성이 기본키에 대하여 완전함수종속을 만족

c. 제 3 정규형

: R이 제 2 정규형이고, 기본키를 제외한 모든 속성이 기본키에 대하여 이행 함수적 종속을 만족하지 않는 정규형

d. BCNF

: R에서 결정자가 후보키인 정규형

e. 제 4 정규형

: R에 다치 종속 A ↠ B가 존재할 경우 R의 모든 속성이 A에 함수적 종속 관계를 만족하는 정규형

f. 제 5 정규형

: R의 모든 조인 종속이 R의 후보키를 통해 성립되는 정규형

2. 반정규화 (Denormalization)

- 시스템 성능 향상과 편의성을 높이기 위해 정규화된 데이터 모델에서 의도적으로 정규화 원칙을 위배하는 행위
- 시스템 성능이 향상되고 관리 효율성은 증가하지만, 데이터 일관성 및 정합성이 저하
- 과도한 반정규화는 오히려 성능 저하의 요인

a. 테이블 통합

: 두 개의 테이블이 조인되어 사용되는 경우가 많을 경우 아예 하나의 테이블로 만듦

b. 테이블 분할

: 테이블을 수평 혹은 수직으로 분할

1) 수평 분할 - 레코드 기준으로 분할

2) 수직 분할 - 속성 기준으로 분할

c. 중복 테이블 추가

: 작업의 효율을 향상시키기 위해 테이블 추가

d. 중복 속성 추가

: 조인해서 데이터를 처리할 때 데이터를 조회하는 경로를 단축하기 위해 자주 사용하는 속성을 하나 더 추가

시스템 카탈로그

1. 시스템 카탈로그 (System Catalog)

   : 시스템 자체에 관련이 있는 다양한 객체에 관한 정보를 포함하는 시스템 데이터베이스

2. 메타 데이터 (Meta-Data)

   : 시스템 카탈로그에 저장된 정보

3. 데이터 디렉터리 (Data Directory)

   : 데이터 사전에 수록된 데이터에 접근하는 데 필요한 정보를 관리 유지하는 시스템

트랜잭션

1. 트랜잭션 (Transaction)
   - 데이터베이스의 상태를 변환시키는 하나의 논리적 기능을 수행하기 위한 작업의 단위, 한꺼번에 모두 수행되어야 할 일련의 연산

   - 특성
     1. 원자성 (Atomicity)

        : 트랜잭션의 연산은 Commit 되던지 Rollback 되어야 함
     2. 일관성 (Consistency)

        : 트랜잭션의 실행이 완료되면 언제나 일관성 있는 데이터베이스 상태로 변환되어야 함
     3. 독립성 (Isolation)

        : 둘 이상의 트랜잭션이 동시에 병행 실행되는 경우 다른 트랜잭션의 연산이 끼어들 수 없음
     4. 영속성 (Durability)

        : 성공적으로 완료된 트랜잭션 결과는 장애가 발생하더라도 영구적으로 반영되어야 함

뷰 / 클러스터

1. 뷰 (View)
   - 접근이 허용된 자료만을 제한적으로 보여주기 위해 테이블로부터 유도된 가상 테이블
   - 물리적으로 존재하지는 않음
   - 뷰가 정의된 기본 테이블을 삭제하면 뷰도 같이 삭제
   - 뷰를 삭제하면 그 뷰를 기반으로 생성된 다른 뷰도 같이 삭제
   - 정의 : CREATE, 삭제 : DROP

   - 장점
     1. 논리적 데이터 독립성 제공

2. 동일 데이터에 대해 동시에 여러 사용자의 요구 반영
3. 사용자의 데이터 관리를 간단하게 해줌
4. 접근 제어를 통한 자동 보안

- 단점
  1. 독립적 인덱스 가질 수 없음
  2. 뷰의 정의 변경 불가
  3. 뷰로 구성된 내용의 삽입, 삭제, 갱신 연산에 제약이 따름

2. 클러스터 (Cluster)
   - 데이터 저장시 데이터 액세스 효율을 향상시키기 위해 동일 성격의 데이터를 동일 데이터 블록에 저장하는 물리적 저장 방법

스토리지

1. DAS (Direct Attached Storage)

   : 서버와 저장장치를 전용 케이블로 직접 연결
2. NAS (Network Attached Storage)

   : 서버와 저장장치를 네트워크를 통해 연결
3. SAN (Storage Area Network)

   : DAS의 빠른 처리와 NAS의 파일 공유 장점을 혼합한 방식

   : 서버와 저장장치를 연결하는 전용 네트워크를 별도로 구성

자료구조

1. 배열 (Array)

   : 크기와 데이터 타입이 동일한 자료들이 순서대로 나열된 자료의 집합
2. 연속 리스트 (Contiguous List)

   : 배열 처럼 연속되는 기억장소에 저장되는 자료구조
3. 연결 리스트 (Linked List)

   : 자료들을 임의의 기억공간에 기억시키되 순서에 따라 노드의 포인터를 이용하여 서로 연결시킨 자료구조
4. 스택 (Stack)

   : 리스트의 한쪽 끝으로만 자료의 삽입, 삭제가 이루어지는 자료구조

: LIFO, 오버플로우, 언더플로우

5. 큐 (Queue)

   : 리스트의 한쪽 끝에서는 삽입, 다른쪽 끝에서는 삭제가 이루어지는 자료구조

   : FIFO

   : 포인터 두 개 - Front, Rear

6. 그래프 (Graph)

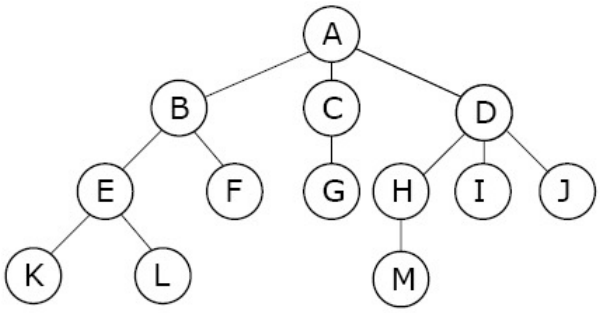
   : 정점 (Vertex)와 간선 (Edge)의 두 집합으로 이루어진 자료구조

   : 방향 그래프 간선 수 - n(n-1)

   : 무방향 그래프 간선 수 - n(n-1) / 2

트리 / 이진트리

1. 트리 (Tree)
   - 정점 (Node)과 선분 (Branch)를 가지고 사이클을 이루지 않는 그래프의 특수 형태



   - 노드 : A, B, C, D, E, F, G, H, I, J, K, L, M
   - 루트 노드 : A
   - 디그리 (차수) - 자식이 있는 노드중 자식의 갯수

     : A=3, B=2, C=1, D=3, E=2, H=1
   - 단말 노드 (Leaf Node) : K, L, F, G, M, I, J
   - 조상 노드 : M의 조상 H, D, A
   - 자식 노드 : B의 자식 F
   - 부모 노드 : E, F의 부모 B
   - Level - 맨 위가 Level 1이 됨

     : H의 Level = 3
   - 깊이 (Depth) - 최대 레벨

     : Depth = 4

2. 이진 트리
   - 트리 운행법
   - 수식 표기법

정렬