1. INTRODUCTION TO DATABASE

Data Models, Data Languages, Data Approaches

Generated by 이유림

참고 강의 : https://www.edwith.org/database/lecture/27849/

데이터 모델이란?

- DB의 바탕이 되는 구조, DB의 논리적 구조를 정의

  : Data / Data Relationship / Data Semantics / Data Constraint

- **Data** : 데이터베이스에 저장된 데이터들을 어떻게 부를 것인가?

  예) 1차원 배열, 테이블, raw와 column, node와 edge, key와 value 등

- **Data Relationship** : 데이터들 간의 관계
- **Data Sementics** : 데이터가 갖는 의미
- **Data Constraints** : 유효한 데이터를 위한 제약 조건

데이터 모델의 종류

- Relational Model (데이터베이스의 표준 모델)
- Entity-Relationship Data Model (ER model)
- Object-based data model

  : Object-oriented data model, Object-relational data model
- Semi-structured data model
- Other older Models

  : Network model, Hierarchical model

데이터 모델의 종류 (1)

- Relational Model (데이터베이스의 표준 모델)
  - 테이블(=relation)을 사용하여 데이터를 표현
  - SQL 사용

- SQL
  - 데이터의 삽입, 갱신, 삭제, 검색 가능
  - SQL은 Relational Model이 데이터베이스의 표준 모델이 되는 데 큰 기여

데이터 모델의 종류 (2)

- Entity-Relationship data model (ER model)
  - 객체(entity)와 객체들 사이의 관계(relationship)로 표현
  - 데이터베이스 설계 단계에서 사용
  - relational model로 매핑 가능

데이터 모델의 종류 (3)

Object-based data model

- Object-oriented data model

  - object model을 기반으로 정보 저장 및 검색

  - Relational Model의 단점(비정형 복합정보 표현 힘듦) 해결

  - 기본적인 데이터베이스 기능 ↓ (표준으로 사용 X)

- Object-relational data model

  - Relational DB + Object-oriented model의 장점

데이터 모델의 종류 (4)

- Semi-structured data model

  - 메타데이터 + 스키마 포함

  - 반 정형 데이터 모델 (스키마와 데이터 간의 구분이 없음)

  - 앱에서 많이 사용

  예) XML : 인터넷 상 데이터 교환 포맷으로 많이 사용

데이터 모델의 종류 (5)

Other older models

- Network model

  - 그래프 사용, 객체 사이의 관계 표현

  - 최근 소셜 네트워크가 각광받으며 주목

- Hierarchical model

  - 객체들이 Parent-Child 관계를 갖고, 계층적인 구조를 이룸

  (모든 객체들은 단 하나의 parent만 가질 수 있으며, 가장 꼭대기의 객체를 root라고 함)

DATA LANGUAGE

: DB에서 사용되는 언어

- Data Manipulation Language (DML)
- Data Definition Language (DDL)

  - DMBS와 통신하는 수단

  (데이터언어를 사용하여 스키마를 만들고, 데이터 삽입/삭제/수정/검색 가능)

DATA MANIPULATION LANGUAGE (DML)

: 데이터에 접근(검색) 및 조작(삽입, 삭제, 수정)하기 위한 언어 (query language)

- **Procedural**

  : 원하는 데이터 + 데이터를 얻기 위한 과정 기술

- **Nonprodecural (Declarative)**

  - 원하는 데이터만 명시 (과정 기술 X)

  예) SQL

DATA DEFINITION LANGUAGE (DDL)

- DB의 스키마 정의, 데이터 특성 표현

- 메타데이터 (데이터의 데이터) : 원하는 데이터 + 데이터를 얻기 위한 과정 기술

  1) Database schema

  2) Integrity constraints (무결성 제약조건)

  - Primary key (ID uniquely instructors) : 기본키는 유효해야 하며, null 값 X

  - Referential integrity (references constraint in SQL) : 외부키는 반드시 자신의 기본 키인, 참조되는 relation에 값이 존재해야 함.

  3) Authorization

  - 어떤 자원에 대한 접근을 허가하는 권한

SQL

: Widely used non-procedural language

- Example: Find the name of the instructor with ID 22222

```
select   name
from     instructor
where    instructor.ID = '22222'
```

- Example: Find the ID and building of instructors in the Physics dept.

```
select instructor.ID, department.building
from instructor, department
where instructor.dept_name = department.dept_name and
      department.dept_name = 'Physics'
```

SQL

데이터베이스 응용프로그램을 만드는 방법

- **Embedded SQL** 사용 : 그 언어를 확장
- **Application program interface** 사용 (ODBC / JDBC)

  - SQL query를 데이터베이스로 보냄

  (다음 슬라이드에 예시)

SQL

ID	name	dept_name	salary
22222	Einstein	Physics	95000
12121	Wu	Finance	90000
32343	El Said	History	60000
45565	Katz	Comp. Sci.	75000
98345	Kim	Elec. Eng.	80000
76766	Crick	Biology	72000
10101	Srinivasan	Comp. Sci.	65000
58583	Califieri	History	62000
83821	Brandt	Comp. Sci.	92000
15151	Mozart	Music	40000
33456	Gold	Physics	87000
76543	Singh	Finance	80000

dept_name	building	budget
Comp. Sci.	Taylor	100000
Biology	Watson	90000
Elec. Eng.	Taylor	85000
Music	Packard	80000
Finance	Painter	120000
History	Painter	50000
Physics	Watson	70000

1) Instructor table : ID, name, dept_name, salary
2) Department : dept_name, building, budget

SQL

ID	*name*	*salary*	*dept_name*	*building*	*budget*
22222	Einstein	95000	Physics	Watson	70000
12121	Wu	90000	Finance	Painter	120000
32343	El Said	60000	History	Painter	50000
45565	Katz	75000	Comp. Sci.	Taylor	100000
98345	Kim	80000	Elec. Eng.	Taylor	85000
76766	Crick	72000	Biology	Watson	90000
10101	Srinivasan	65000	Comp. Sci.	Taylor	100000
58583	Califieri	62000	History	Painter	50000
83821	Brandt	92000	Comp. Sci	Taylor	100000
15151	Mozart	40000	Music	Packard	80000
33456	Gold	87000	Physics	Watson	70000
76543	Singh	80000	Finance	Painter	120000

Dept_name을 기준으로 instructor, department 2개의 relation이 합쳐짐.

SQL

ID	name	salary	dept_name	building	budget
22222	Einstein	95000	Physics	Watson	70000
12121	Wu	90000	Finance	Painter	120000
32343	El Said	60000	History	Painter	50000
45565	Katz	75000	Comp. Sci.	Taylor	100000
98345	Kim	80000	Elec. Eng.	Taylor	85000
76766	Crick	72000	Biology	Watson	90000
10101	Srinivasan	65000	Comp. Sci.	Taylor	100000
58583	Califieri	62000	History	Painter	50000
83821	Brandt	92000	Comp. Sci	Taylor	100000
15151	Mozart	40000	Music	Packard	80000
33456	Gold	87000	Physics	Watson	70000
76543	Singh	80000	Finance	Painter	120000

Building, budget에서 중복된 값을 가짐.

→ 데이터의 중복

→ 데이터의 일관성 해침

→ 데이터의 불일치성

Relation의 primary key = ID (primary key는 null 값을 가질 수 없음!)

DESIGN APPROACHES

- Nomalization Theory (정규화)

  : 불필요한 데이터의 중복을 막기 위해 relation을 나누는 과정

  : 설계된 relation이 얼마나 잘 설계되었는지 확인하기 위해 사용하기도 함.

- Entity Relationship Model (ER model)

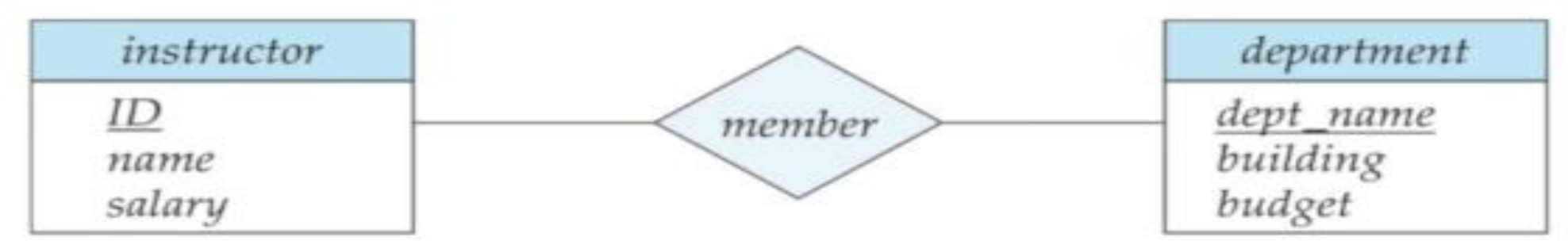
  : 데이터베이스 디자인의 기본이 되는 모델, Entity + Relationship

  1) Entity – 다른 객체들과 구별되는 객체, 자신을 표현할 수 있는 속성 가짐

  2) Relationship – entity들 사이의 연관성

ENTITY RELATIONSHIP MODEL (ER MODEL)

다이어그램으로 표현한 ER model



1) Entity : 사각형을 그리고, 사각형 윗부분에 entit의 이름 적기 (ex. Instructor)

2) Relationship : entity 이름 아래에 적기 (ex. ID, name, salary)

3) Primary key : 해당 attribut에 밑줄을 그어 표시 (ex. ID)

4) Relationship : 마름모를 그리고, relationship의 이름 적기 (ex. Member)

5) Entity-Relationship : Relationship(마름모)에 참여하는 entity(사각형)을 선으로 연결