2. Introduction to DB & Relational Model

▼ Database → relation과 다름

데이터의 집합체 (integrated collection of data)

▼ Database Systems

- centralized control
- uniform access & control of the data.
- redundancy ↓ 중복 X.
- control of integrity

- Database : 데이터의 집합체 → data 관리
- Database Systems : 데이터 저장, 관리, 안정성(복구, 보완)

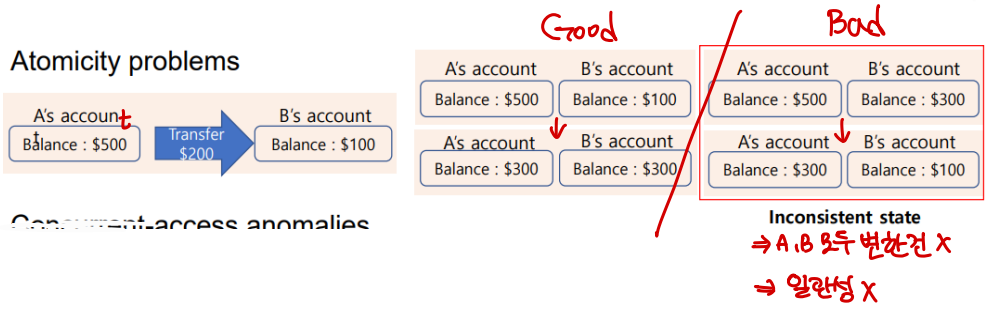
▼ why need? (drawback of file systems) ⇒ 왜 file 사용하지 않고 database?

1. redundant data
   - 중복된 data에 대하여 수정된 data를 효율적으로 보완함(많아지면 관리 어려움)
   - inconsistent 방지 (일관성 유지)

Course number	3178		Course number	4656
Instructor	Wookhee Kim		Instructor	Wookhee Kim
Office	New Eng. Bld. 1217	Redundant Data	Office	Eng. Bld. C422 → 수정 X ⇒ 일관성 X
Email_id	wookhee		Email_id	wookhee Inconsistent (일관성이 깨짐)

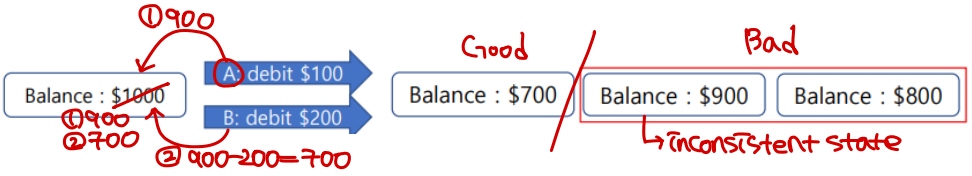
수정된 data

2. difficulty in accessing data : 접근할 때 새로운 프로그램을 필요로 함
3. data isolation : 용량이 너무 클 때 분리함(각각의 file이 다른 format을 가질 수 있음)
4. integrity problems : data가 가져야 할 조건을 명시해야 함 (ex. 학점 = [0, 4.5])
5. atomicity problems(원자성) : 상태가 아예 바뀜 or event가 아예 X



6. concurrent-access anomalies : 동시적으로 접근 → 성능 good(user들을 제어해야 함)

ex) A,B ⇒ 서로의 존재를 인식 X , 같은 Balance에 대해 수행



7. security problems : user에 따라 접근 권한 다르게 설정 ⇒ 보안 ↑

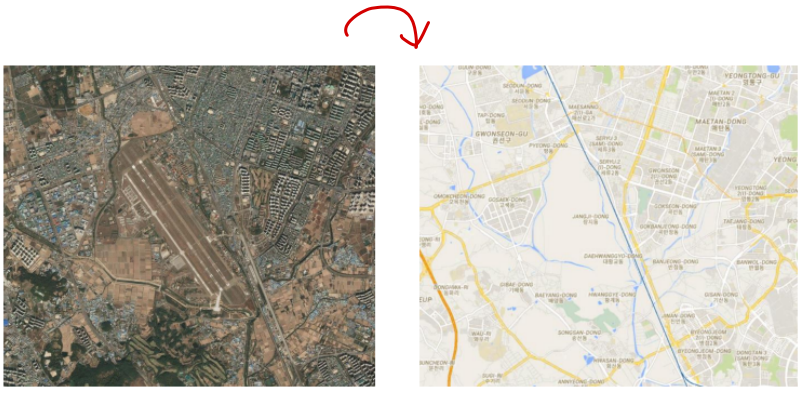
▼ Data models

a collection of conceptual tools for

- data, data relationships, data semantics, consistency constraints

1. **Relational model**(Entity-Relationship data model)
2. Object-based data models
3. Semi-structured data model(XML)

▼ why?



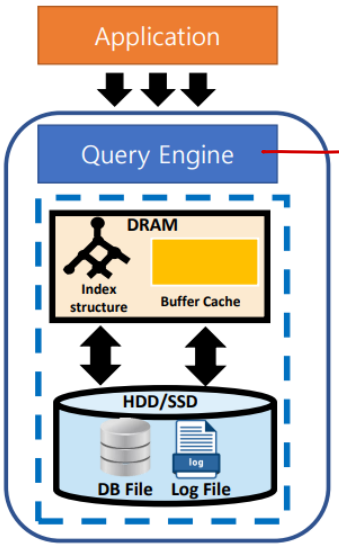
Real World (Satellite Image) Model (Map)

- examine or manage parts of the real world → 더 저렴하게, 효율적으로 가능

ex) 비행기 시뮬레이터, 지도, 빌딩 구조 ...

▼ Data abstraction (데이터 추상화)

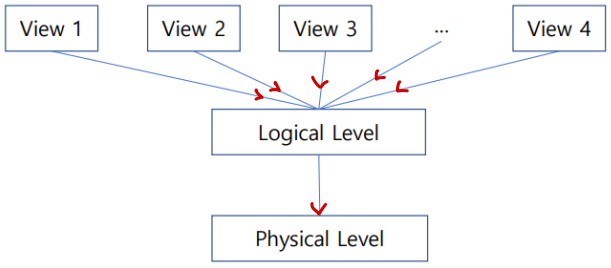
data에 대한 abtract view를 user에게 제공



query 요청
language : SQL

- database system : data가 어떻게 저장, 관리되는지에 대한 특정한 디테일을 숨김
- 효율을 위해 복잡한 data structures 사용 ⇒ 효율이 가장 중요!
  - but 대부분의 user는 not-computer-trained
- user별로 view를 제공(data abstraction level 제공)

내부정보 다 보이지 않도록!



추상화 High
User 최적화 Level
사람이 이해할 수 있는 Level
추상화 Low
HW에 최적화 되어있는 Level

① Physical level : 저장방식 정의
  - ex) index structures(자료구조, slotted page)
② Logical level ↪ SQL문 같은 느낌
③ View level(highest abstraction level)
  - logical level에 대해 detail 숨김, security mechanism 제공

ex) 학생지원팀 : 교수의 연봉은 볼 수 없음.

- schema : programming language의 type, variable과 유사함
  - logical schema : the overall logical strucutre of the database
  - physical schema : the overall physical structure of the database

  ⇒ physical data independence : logical schema의 변경 없이 physical schema 수정 가능

application : depend on the logical schema

사용에서 상속하게 영향을 미치지 않음 → design good

data가 구조화되는 방식

- instance : 같은 database에 대한 특정 시점을 의미
  → Analogous to the value of variable

▼ Relational Model

(table) relation

attributes (or columns)

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

tuples (or rows) → 하나의 entity에 대한 정보를 담고 있음.

▼ attribute types

- **domain** : attribute type (DT) ex) ID에는 숫자만!
- **attribute values** : to be atomic(항상) ⇒ invisible(최소 단위이기에 나눌 수 x)
- **null**(special value) : a member of every domain

▼ Relation schema and instance

- **attributes** : A1, A2, A3 , ... , An
- **relation schema** : R = (A1, A2, ... , An) → relation 관련
  - ex. instructor = (ID, name, dept_name, salary)
- relation instance : r(R) → relation, schema
- **a table** : the curret values at relation → specified!
- an element **t(tuple)** of relation of **r**
  - tuple : row에 의해 represented

▼ relations are unordered → 정렬 기준에 따라 정렬 필요

instructor relation

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

▼ Database ⇒ 여러 개의 Relation

- table(relation) 정보 별로 생성 → 1개 이상의 집합 ex) instructor, student, advisor
  가능
- repetition of information, need for null values → 잘 고려해야 함.

▼ keys

데이터 분류할 때 기준이 됨

Let. $K \subseteq R$

- **superkey** : tuple 구분하는 key (K is a superkey of R)
  - ex. {ID} and {ID,name} are both superkeys of instructor
- **candidate key** : 만약 k가 minimal이라면 candidate key → 최소의 구성으로 구분 가능
  - ex. {ID} is a candidate key for instructor
- **primary key** : 기준키, candidate 중 하나 설정
- **foreign key** : 다른 table에 존재
  - 같은 attribute여도 foreign key면 다른 table에서도 똑같은 값을 가지는 것이 존재