

2. RELATIONAL MODEL

Intro to Relational Model (1)

RELATIONAL MODEL

: 관계 모델 = 테이블이랑 비슷함.

- **Relation** : 테이블

- 1) 고유한 이름을 가짐 2) Tuple의 모임

- **Tuple** : 테이블의 행

- 1) 독립된 데이터들 간의 Relationship을 보여줌 2) n-tuple (n개 값의 관계)

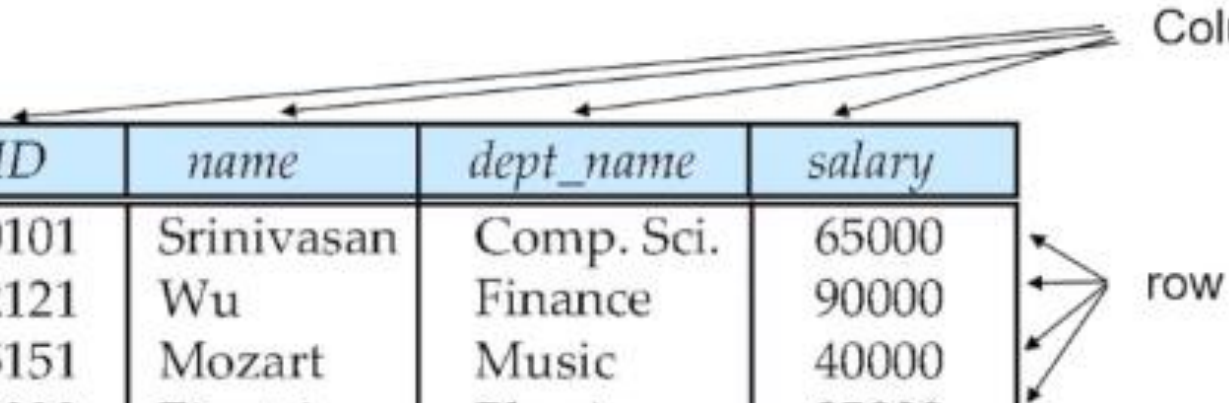
- 3) Cardinality : 하나의 relation의 tuple의 개수

- **Attribute(속성)** : 테이블의 열

- 1) 다음 장 예시 – ID, name, dept_name, salary

Relational DB는 테이블들의 모임임, 여러 개의 relation으로 DB가 구성됨.

EXAMPLE OF A RELATION



<i>ID</i>	<i>name</i>	<i>dept_name</i>	<i>salary</i>
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

RELATIONS ARE UNORDERED

- Relation은 순서를 가지지 않음
= Relation에서 tuple의 순서는 상관이 없다!)
- Tuple들의 구성 내용이 같으면, 동일한 relation으로 간주
: 다른 순서로 정렬되어 있어도 동일한 relation으로 간주한다.
- Relation은 중복된 tuple을 가져서는 안 됨.

ATTRIBUTE TYPES

: 각 attribute들은 Domain을 가짐

- **Domain** : 각 attribute에 허락된 값들의 집합 (데이터 타입같은 느낌)

예) 정수 도메인, 문자형 도메인

- **Atomic** : 논리적으로 쪼갤 수 없는 성질, atomic하다 = 하나의 값(원자 값)을 갖는다

: relation의 attribute들의 도메인은 모두 atomic해야 함!

- **Null**(알 수 없음 or 값이 존재 x)

: 모든 attribute는 null 값이 될 수 있음.

: ! null 값은 연산의 정의를 복잡하게 하므로, null 값을 사용할 때는 주의해야 함 !

RELATIONAL SCHEMA AND INSTANCE

Relational Schema = attribute들 + 해당 도메인들

- 표기법

$: A_1, A_2, \dots, A_n$ | attribute이고 relation의 이름이 name이라면 'name = (A₁, A₂, ..., A_n)'

- Relation이 도메인의 모든 값을 갖는 것이 아니라, 그 중 일부만 relation이 되는 것임.

- Relation 인스턴스 : relation의 현재 값, 테이블로 명시됨

- Relation r의 구성요소 t : tuple, 테이블에서 열로 표시

- Relation 변화 \rightarrow relation의 튜플 값 변화 ○

\rightarrow 스키마 변화 X

KEYS

: Relation의 부분집합, relation에서 tuple을 구별하기 위해 사용

- Super key (수퍼키)
- Candidate key (후보키)
- Primary key (기본키)
- Foreign key (외래키)

SUPER KEY

: Tuple을 유일하게 식별할 수 있는 relation의 부분집합

예) dept_name의 값이 Physics인 tuple이 하나 이상이므로, dept_name은 superkey가 될 수 없다.

예) 대학생 정보를 예로 들면, 대학교 내에서 학생들의 학번은 각 학생에게 하나씩 부여되는 unique한 것이므로 Superkey가 될 수 있다.

- Attribute 한 개 또는 여러 개로 구성

예) unique한 속성인 학번을 포함한 모든 attribute의 조합이 수퍼키가 될 수 있다.

- Superkey는 여러 개 존재할 수 있음.

CANDIDATE KEY

: 최소한의 속성들로만 구성된 Superkey

= 구성하는 속성들 중 하나라도 없으면, tuple을 유일하게 구별할 수 없음.

예) ID, name으로 이루어진 Superkey에서 ID가 없으면 name만으로는 tuple을 유일하게 구별할 수 없으므로, ID는 후보키이다.

PRIMARY KEY

: relation들의 관계를 올바르게 표현하기 위해 사용하는 key

- 서로 다른 relation의 공통된 attribute
- Foreign key constraint (참조 무결성)

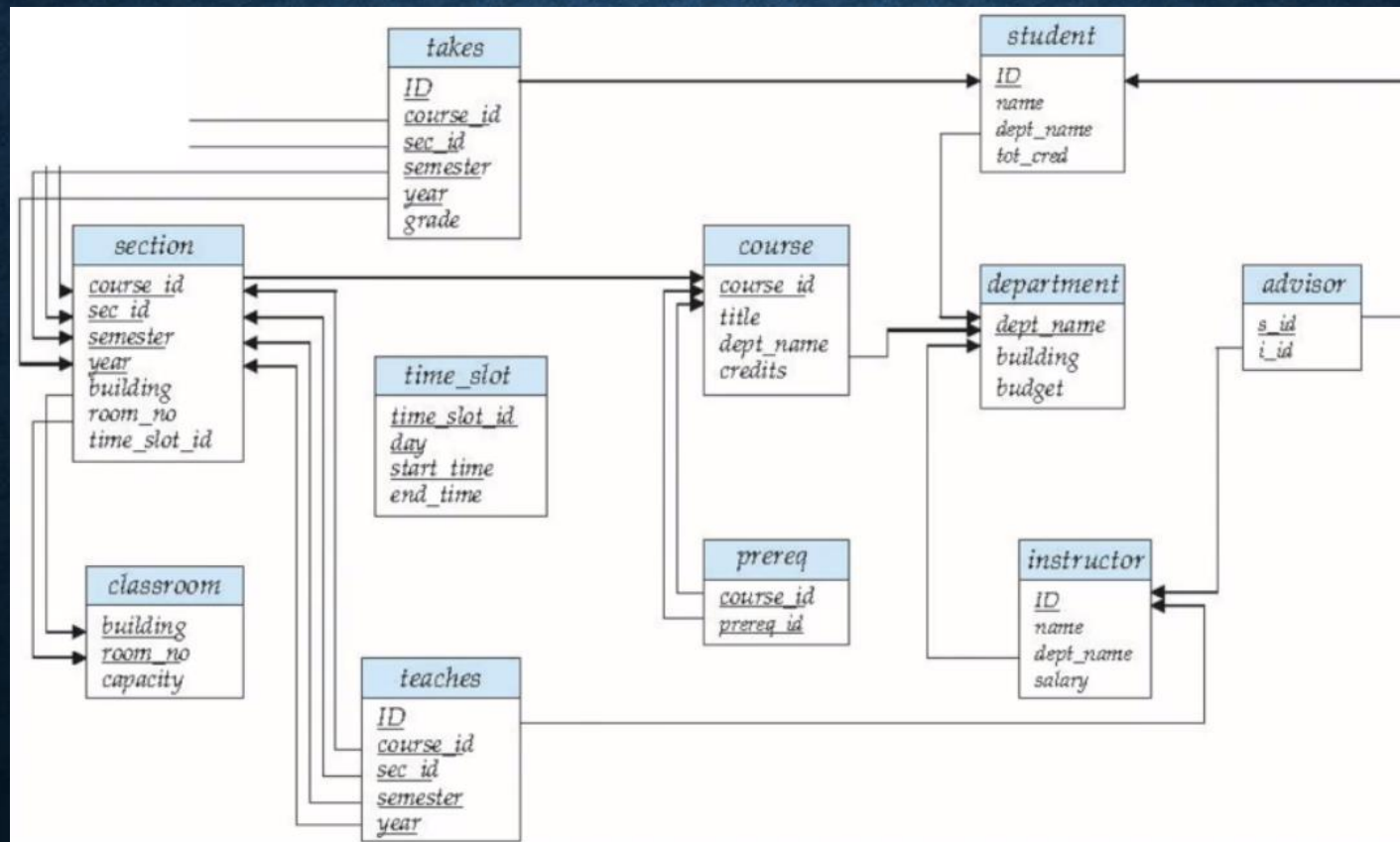
: referencing relation의 foreign key 값은 반드시 referenced relation에 존재해야 한다.

1) Referencing relation : foreign key를 가지고 있는 relation

2) Referenced relation : 다른 relation의 foreign key를 primary key로 가지고 있는
relation

SCHEMA DIAGRAM (1)

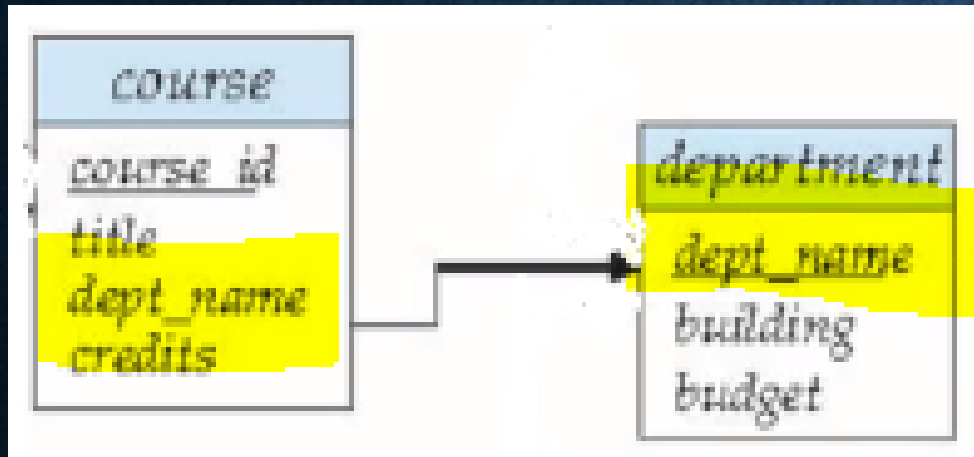
: Schema Diagram은 DB가 Primary key와 Foreign key에 가지는 종속성을 시각적으로 나타냄.



SCHEMA DIAGRAM (2)

- Relation
: 네모 칸에 이름 적기
- Relation의 attribute
: 이름 아래의 네모 칸에 적기
- Primary key
: 밑줄 그어 표시
- Foreign key의 종속성
: 참조하는 relation → 참조되어지는 relation으로의 화살표

SCHEMA DIAGRAM EXAMPLE (1)

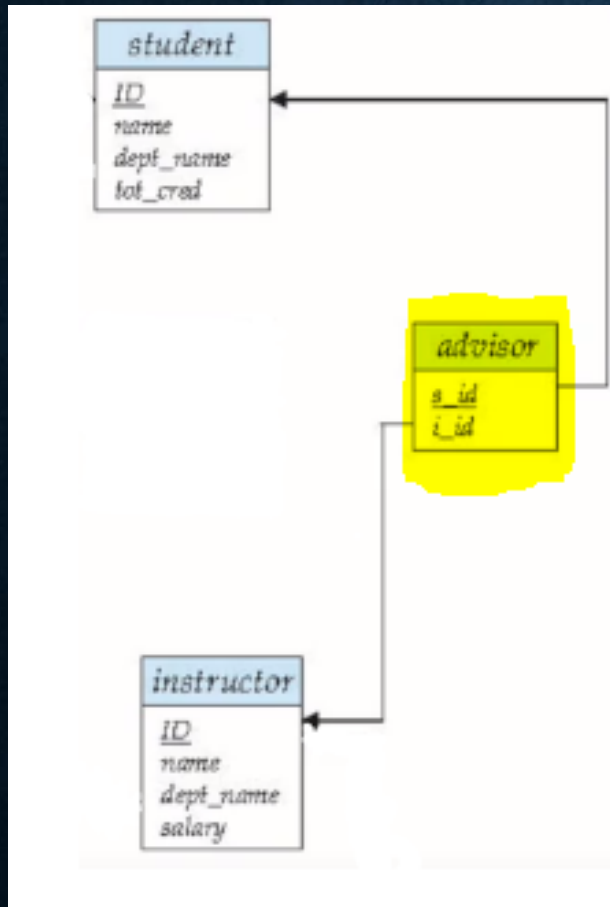


예) course relation의 dept_name은 department relation의 dept_name을 참조하고 있음.

예) department relation의 dept_name은 primary key로, 유일한 값을 가짐.

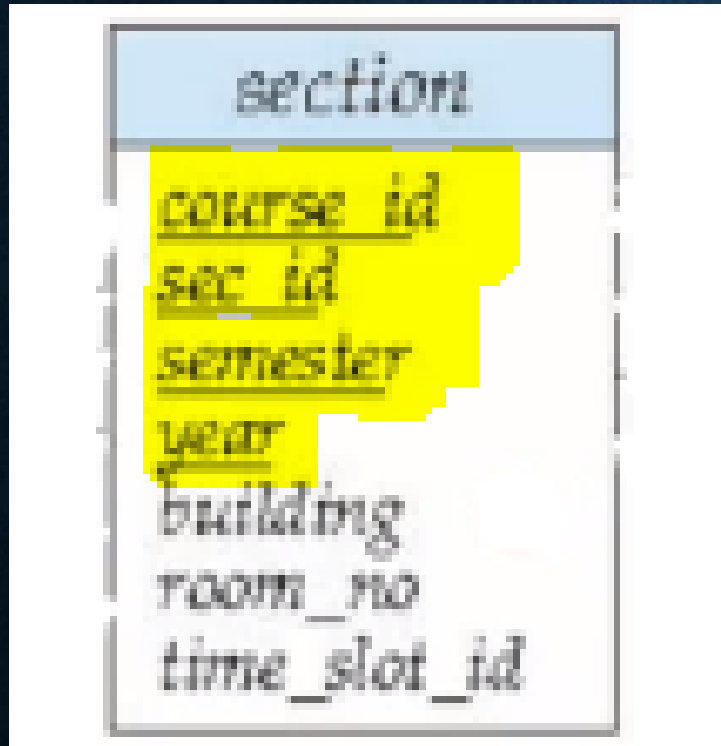
예) **foreign key constraint** : course relation의 모든 dept_name 속성값은 department relation의 dept_name 속성값 중에 존재해야 한다.

SCHEMA DIAGRAM EXAMPLE (2)



예) advisor relation은 student relation과 instructor relation을 연결해주는 relation임

SCHEMA DIAGRAM EXAMPLE (3)



예) **composite key** (복합키) : 여러 개의 attribute들로 구성된 키