DB06

목표: **Relational Database를 Design**하는 방법에 대해 배운다.

- ER model → Relational model mappint
- ER-to-Relational Mapping

Relational Database Design by ER-to-Relational Mapping

Conceptual design (ER Schema)에 기반하여 Relational database schema를 얻는 과정

- **ER model → Relational model**로 Mapping한다.

Entity, Arrribute, Relationship의 세 가지 Type을 Mapping하는 방법에 대해 순서대로 알아보자.

---

1. Entity type mapping

a. Regular entity types mapping

- **Regular entity type**: Week entity type이 아닌 것을 의미한다.

1. 각 **Regular entity type을 Relation으로 Mapping**한다.
    - Entity relation이라고 부른다.

2. **Simple attribute**(Composite, Multivalued 제외)는 **Relation의 Attribute**가 되도록 한다.
    - Composite, Multivalued는 다른 방법을 사용한다.
3. Entity type에서 **하나의 Key arrtibute를 선택하여 Primary key로 설정**한다.
    - Database designer가 Entity의 key 중 하나를 선택한다.

b. Week entity types mapping

1. **각 Entity type을 Relation으로 매핑**한다.
2. **Simple attribute는 relation의 attribut**e가 된다.
3. Indentifying owner의 **Primary key가 Foreign key**가 되도록 한다.
    - Indentifying owner의 PK를 알아야 하기 때문에 Week entity의 매핑은 Indentifying owner의 매핑 이후에 진행되어야 한다.
4. **Primary key: Entity type의 Key 중 하나를 선택하고 선택한 Key와 Foreign key를 합쳐서 하나의 Primary key**로 사용한다.
    - **Week entity의 Partial key**를 이용한다.

2. Attribute mapping

a. Multivalued attribute mapping

1. 각 Multivalued attribute들을 하나의 새로운 Relation으로 매핑
2. Multivalued attribute가 속한 Entity type의 Relation의 Primary key를 FK로 가져온다.
3. **(1)에서의 Multivalued attribute와 (2)에서의 primary key를 합쳐 Primary key**로 사용한다.
    - Multivalued + FK

b. Composite attribute mapping

두 가지 방법이 있다.

첫 번째 방법

1. 각 Composite attribute들을 하나의 새로운 Relation으로 매핑
2. Composite attribute를 구성하던 Simple attribute들을 Relation에 추가한다.
3. 원래 Composite attribute가 속하던 Entity type의 Primary key를 가져와서 Foreign key로 삼는다.
4. **Primary key**는 (3)에서 가져온 **Foreign key를 그대로 이용**한다.

두 번째 방법

1. Composite attribute를 구성하던 Simple attribute들을 원래의 Entity type에 대응되는 Relation에 attribute로 추가한다.

3. Relationships mapping

a. 1:1 Relationship types mapping

1. Relationship type에 참여하는 Entity type의 Relation을 찾는다.
2. 두 Relation 중 하나를 정해 하나의 PK를 다른 쪽의 FK로 포함시킨다.
    - 이때, **Total participation인 쪽에 FK를 포함**시키는 것이 좋다.
    - Partial participation인 경우, FK로 추가한 Attribute의 값이 NULL이 되는 경우가 많아지고, 이는 Database 공간 낭비이며 비효율적이다.
3. 원래 Replationship의 Simple attributes는 FK를 추가하기로 한 Relation의 Attribute로 포함시킨다.

b. 1:N Relationship types mapping

1. Relationship에서 N-side에 대응되는 Entity type의 Relation을 찾는다.
2. (1)에서 찾는 Entity type이 아니라 다른 Side의 Entity type의 PK를 FK로 (1)에서 찾는 Relation에 넣는다.
3. 원래 1:N Replationship의 Simple attributes는 FK를 추가하기로 한 Relation의 Attribute로 포함시킨다.

c. M:N Relationship types mapping

1. 1:1, 1:N과 다르게 추가하는 방식이 아니라 Independent relation을 새롭게 만든다.
2. Relationship에 대응되는 두 Entity type의 두 PK를 모두 FK로 가져온다.
3. 원래 M:N Replationship의 Simple attributes는 Relation의 Attribute로 포함시킨다.
4. **Primary key**: 모든 Foreign key의 조합

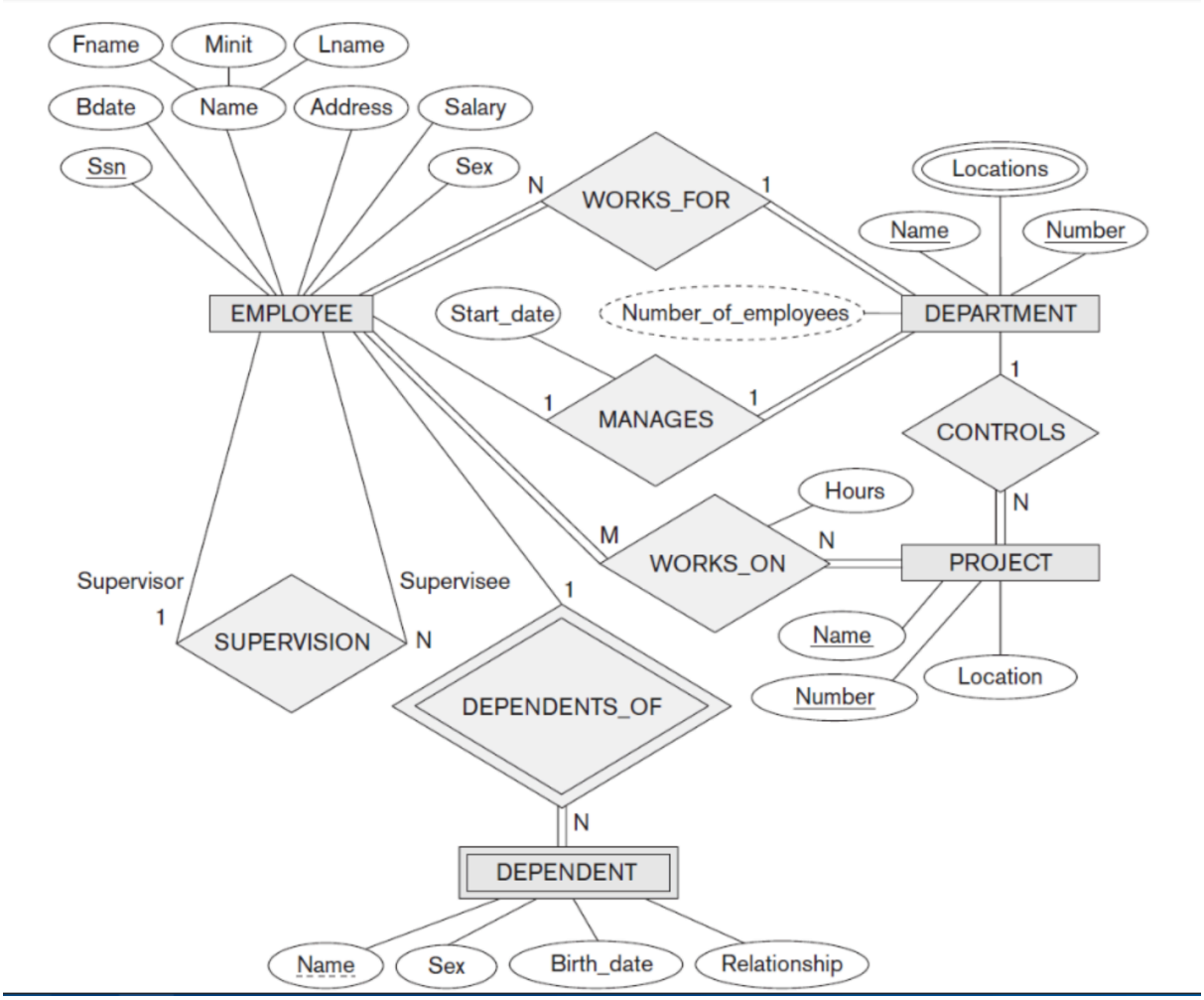
d. N-ary Relationship types (Not binary relationships)

M:N과 **동일한 방식**으로 처리한다.

1. Relation을 새롭게 만든다.
2. Relationship에 대응되는 N개의 Entity type의 PK를 모두 FK로 가져온다.
3. 원래 Replationship의 Simple attributes는 Relation의 Attribute로 포함시킨다.
4. Primary Key: 모든 Foreign key의 조합

Case Study: COMPANY Database

아래 ER Schema를 Relational database로 바꾸어 보자.



EMPLOYEE Entity type

- Regular entity type mapping에 해당한다.
- **EMPLOYEE**라는 **새로운 Relation**을 생성한다.
- **Simple attribute** (Ssn, Bdate, Address, Sex, Salary)를 **Relation에 추가**한다.
- Key가 **Ssn 하나이므로 PK로** 잡는다.

DEPARTMENT Entity type

- Regular entity type mapping에 해당한다.
- **DEPARTMENT**라는 **새로운 Relation**을 생성한다.

- **Simple attribute** (Name, Number)를 **Relation에 추가**한다.
    - 이때, **SQL에서 복잡해지는 것을 방지하기 위해 중복될 수 있는 Attribute의 이름들을 수정**해준다.
    - Dname, Dnumber라고 한다.
- **두 Key (Dname, Dnumber) 중 Dnumber를 PK로** 잡는다.

PROJECT

- Regular entity type mapping에 해당한다.
- **PROJECT**라는 **새로운 Relation**을 생성한다.
- **Simple attribute** (Name, Number, Location)를 **Relation에 추가**한다.
    - 마찬가지로 Pname, Pnumber, Plocation으로 수정한다.
- **두 Key (Pname, Pnumber) 중 Pnumber를 PK로** 잡는다.
    - 가급적 숫자를 PK로 잡는 경향이 있긴 하다.

DEPENDENT

- Week entity type mapping에 해당한다.
- **DEPENDENT**라는 새로운 **Relation**을 생성한다.
- **Simple attribute** (Name, Sex, Birth_data, Relationship)를 **Relation에 추가**한다.
    - 마찬가지로 Dependent_name과 Bdate로
- **Identifying owner**의 **PK를 FK로 추가**한다.
    - EMPLOYEE의 **Ssn**을 Essn으로 renaming하여 가져온다.
- **Owner의 PK + Entity의 Key = Essn + Dependent name**을 **PK로 사용**한다.
    - Dependent name은 Partial key이다.

Locations in DEPARTMENT

- Multi-valued attributes의 mapping이다.
- Locations에 대응되는 새로운 **DEPT_LOCATIONS**라는 Relation을 생성한다.
- Locations가 속한 **DEPARTMENT의 PK를 FK**로 가져온다.

    - Dnumber를 가져온다.
- 원래의 **Multivalued attrbute (Locations)**를 가져온다.
    - Dlocation으로 renaming한다.
- **Owner의 PK + Multi-valued attributie = Dnumber + Dlocation을 PK로 사용**한다.

Name in EMPLOYEE

- Composite attributes의 mapping이다.
- 여기에선 **2번째 방법**을 사용한다.
- Name의 **Simple attribute**인 Fname, Minit, Lname을 **Name이 속한 EMPLOYEE에 대응되는 relation에 추가한다.**

MANAGES

- 1:1 relationship types의 Mapping
- **DEPARTMENT**가 **Total participation**이기 때문에 **DEPARTMENT에 대응되는 Relation에 EMPLOYEE에 대응되는 Relation의 PK를 FK로 추가한다.**
    - Ssn을 Mgr_ssn으로 DEPARTMENT에 추가한다.
- **MANAGES**의 **Simple attribute도 DEPARTMENT에 추가**한다.
    - Start_data를 Mgr_start_date로 DEPARTMENT에 추가한다.

WORKS_FOR

- 1:N relationship types의 Mapping
- **N-side인 EMPLOYEE**에 **DEPARTMENT의 PK를 FK로 추가**한다.
    - Dnumber를 Dno로 하여 추가한다.
- WORKS_FOR의 Simple attribute는 없기에 패스한다.

CONTROLS

- 1:N relationship types의 Mapping
- **N-side인 PROJECT**에 **DEPARTMENT의 PK를 FK로 추가**한다.

    - Dnumber를 Dnum로 하여 추가한다.
- CONTROLS의 Simple attribute는 없기에 패스한다.

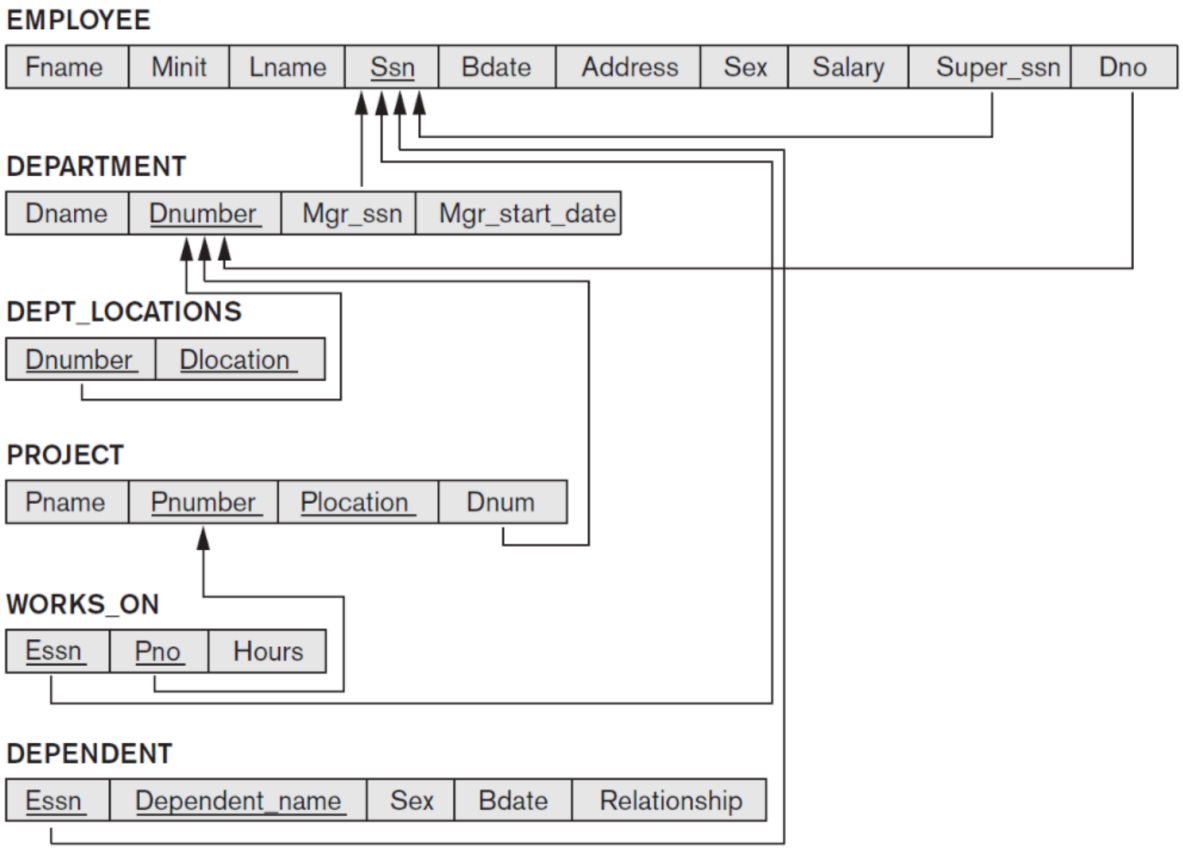
SUPERVISION

- 1:N relationship types의 Mapping
- 1-side(Supervisor)와 N-side(Supervisee) 모두 EMPLOYEE이다.
- N-side(Supervisee) 쪽인 EMPLOYEE에 1-side(Supervisor)인 EMPLOYEE의 PK를 FK로 추가한다.
    - Ssn을 Super_ssn으로 하여 추가한다.
- SUPERVISION의 Simple attribute는 없기에 패스한다.

WORKS_ON

- M:N relationship types의 Mapping
- **WORKS_ON**이라는 새로운 Relation을 생성한다.
- M-side의 EMPLOYEE와 N-size의 PROJECT에 각각 대응되는 Relation의 PK를 FK로 가져온다.
    - Ssn는 Essn으로, Pnumber는 Pno로 가져온다.
- **WORKS_ON**의 Simple attributes는 없으므로 패스한다.
- **가져온 모든 FK의 조합** = **Ssn + Pno를 PK로** 잡는다.

Relational Database schema를 나타내면 다음과 같다.



- 밑줄은 Primary key (PK)를 의미한다.
- PROJECT의 Dnum에서 DEPARTMENT의 Dnumber으로 가는 화살표는 1:N Relationship인 CONTROL에 의한 것이다.
- EMPLOYEE의 Super_ssn에서EMPLOYEE의 Ssn으로 가는 화살표는 1:N Relationship인 SUPERVISION에 의한 것이다.
- EMPLOYEE의 Dno에서 DEPARTMENT의 Dnumber으로 가는 화살표는 1:N Relationship인 WORKS_FOR에 의한 것이다.
- WORKS_ON relation은 M:N Relationship type에 의해 새로 생긴 Relation이다.