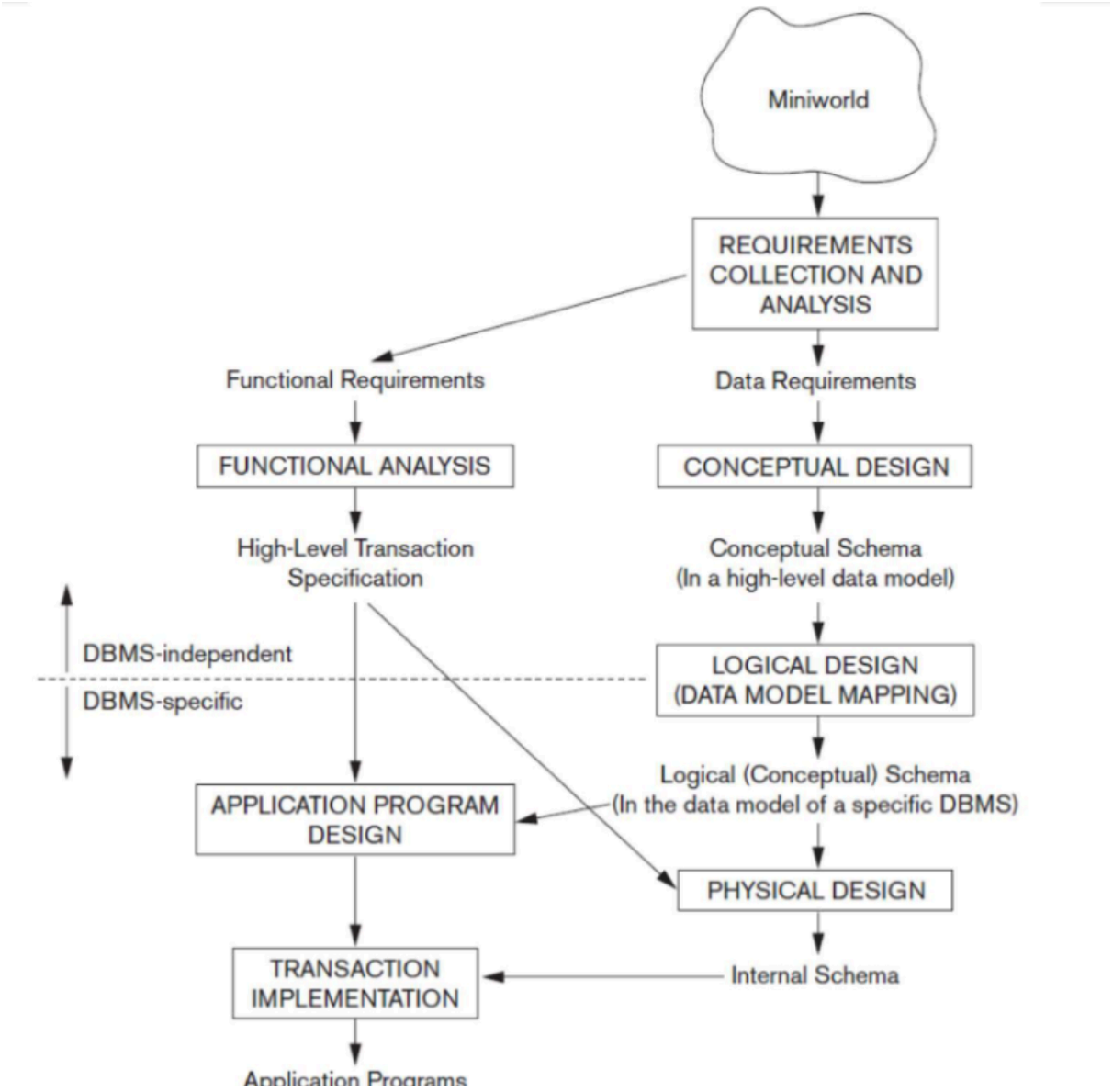
DB03

Database 설계와 응용 프로그램 개발 과정을 도식화하면 아래와 같다.



- Requirements collection and analysis: System analysist가 수행한다.
- Database designer는 **Database의 구조**를 결정하고 Conceptual design, Logical design, physical mapping 세 가지를 수행한다.

Conceptual design

- **사람이 이해하기 쉽도록 하는 표현**하는 것
- DBMS는 이를 이해할 수 없다.
- 예시**: ER model**
- DBMS Independent하다.
    - DBMS Software와는 전혀 상관이 없다.
- 결과로 **Conceptual Schema**를 얻는다.

Logical design

- **DBMS가 제공하는 Logical model을 통해서 DBMS가 이해할 수 있는 형태로 변**환하는 것
- Data model mapping이라고도 한다.
- 예시**: Relation model**
- 결과로 **Logical schema**를 얻는다.
- DBMS dependent하다.

Physical design

- **Database가 물리적으로 어떻게 저장**이 될 지를 결정한다.
- 결과로 **Internal schema**를 얻는다.
- DBMS Independent하다.

Requirements of company database

Departments: 하나의 이름, 하나의 식별 번호, 여러 개의 위치, 매니저, 매니저가 department를 관리하기 시작한 날짜

- Department는 여러 Project를 control한다.

Project: 하나의 이름, 하나의 식별 번호, 하나의 위치

- 하나의 Project는 하나의 Department에 의해서 관리된다.

Employee: 이름, Social security number, 주소, 급여, 성별, 탄생 날짜

- 하나의 Employee는 하나의 Department에 소속된다.
- Employee는 여러 Projects에서 일할 수 있다.
- 각 Project마다 일주일에 몇시간씩 일했는지 알 수 있어야 한다.
- 직속 상사를 추적해야 한다.

Dependent: 이름, 성별, 탄생 날짜, Employee와의 관계

- 각 Employee의 dependent를 추적해야 한다.

ER Model concepts

Entity: 실세계의 독립적인 존재로 존재하는 어떤 것

Attributes: Entity의 특성

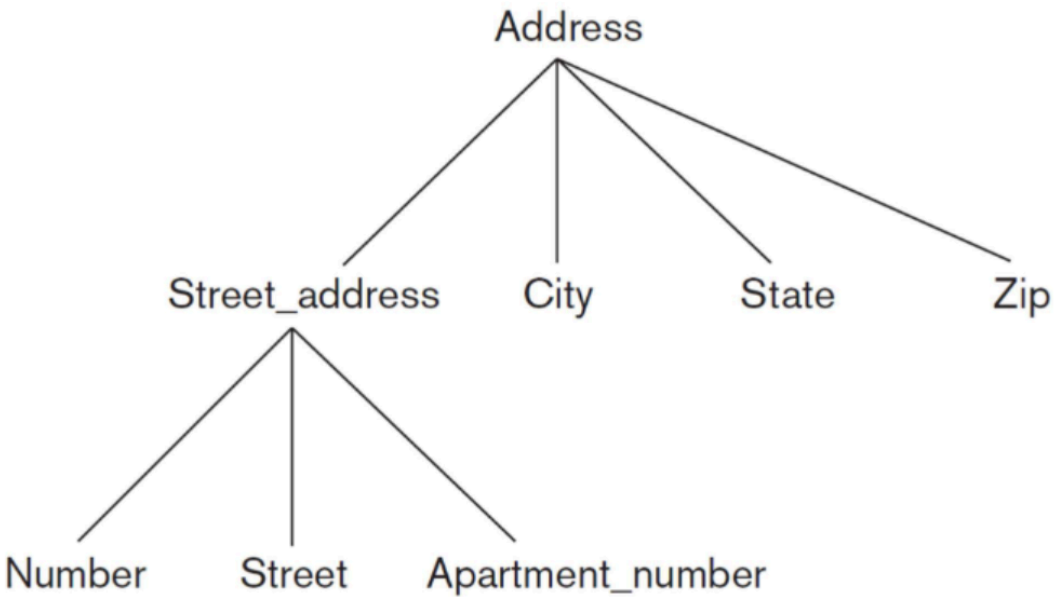
Attribute value: Attribute의 값

Attribute를 세 가지로 분류할 수 있다.

Category 1

- **Simple attributes**: 더 나눌 수 없는 속성
  - 이름은 성/이름으로 나누어 검색 불가하도록 함
- **Composite attributes**: 여러 작은 파트로 나눌 수 있는 속성

- 이름은 성/이름으로 나누어 검색 가능하도록 함
- **Hierarchy형태로 만들 수 있다.**



Category 2

- **Single-valued attributes**
  - 특정 Entity가 Single value를 갖는다.
- **Multivalued attributes**
  - 같은 Entity가 set of values를 갖는다.

Category 3

- **Stored attributes**
  - Database에 실제로 (물리적으로) 저장되는 속성
- **Derived attributes**
  - 저장되어져 있는 Attribute로부터 유도(추측)할 수 있는 Attirvute
  - Database에 저장되지 않는다.
  - EX) Age는 이미 저장되어져 있는 탄생 날짜로부터 유도된다.

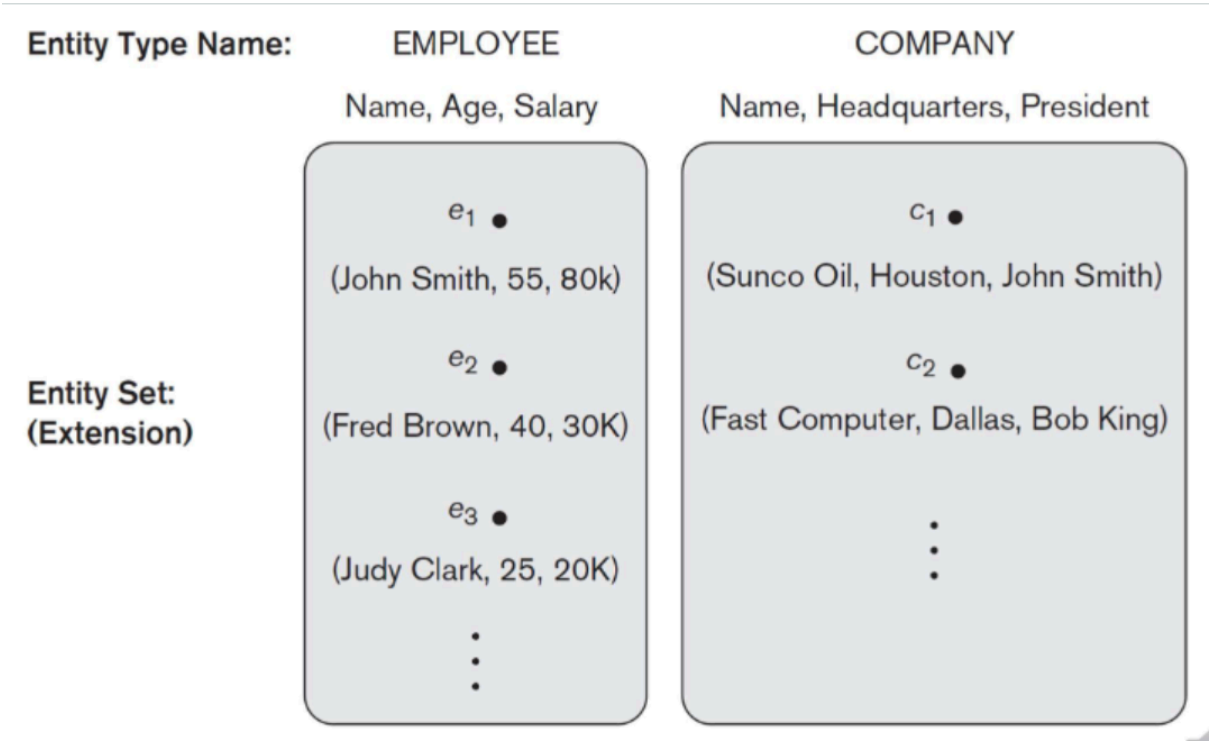
Entity type

같은 Attributes를 갖는 set의 집합

- Set of entity의 Schema (구조)를 나타낸다.
- Name이나 Attributes에 의해 나타나진다.

여러 Entity의 골격을 Define한다.

- 각 Entity는 값은 다르지만 Attribute(골격)은 동일하다.
- Attribute의 집합은 Entity type이다.



- Employee와 Company가 Name이다.
- 2개의 Entity type, 2개의 Entity set이 있다.
- Name, age, salary는 attributes이며 **Name과 attribute 부분을 합쳐 Entity type**이라고 한다.

Domains

- 각 **Entity의 attribute에 할당될 수 있는 값의 집합**
- 어떤 Attribute “a”의 domain은 “a”에 할당될 수 있는 값의 집합을 의미한다.

Key attributes

각 Entity를 구별할 수 있도록 해주는 Attirbute

- Key attribute의 value는 Entity마다 반드시 달라야 한다.
- **Key attribute는 각 Entity를 구별할 수 있어야 한다**

하나의 Key가 여러 개의 Attribute로 구성될 수 있다.

하나의 **Entity type이 두 개 이상의 Key**를 가질 수 있다.

Key attribute를 설정할 때, Attirbute에 값을 할당한 이후에, 겹치는 값이 없다고 그 Attribute를 Key로 설정하는 것이 아니라, **값을 할당하기 전에 Key attribute로 사용하겠다고 미리 지정하고 사전에 겹치는 값을 갖지 못하도록 강제하는 것**이다.

CAR

Registration (Number, State), Vehicle_id, Make, Model, Year, {Color}

CAR_1

((ABC 123, TEXAS), TK629, Ford Mustang, convertible, 2004 {red, black})

CAR_2

((ABC 123, NEW YORK), WP9872, Nissan Maxima, 4-door, 2005, {blue})

CAR_3

((VSY 720, TEXAS), TD729, Chrysler LeBaron, 4-door, 2002, {white, blue})

⋮

- **Registraion**은 Number, State가 합쳐진 형태로 Key로 사용될 수 있다.
  - Number만으로는 Key가 될 수 없다.
- **Vehicle_id**는 Key가 될 수 있다.

위에서 Requirement를 작성한 **Company database의 Entity type**을 살펴보자.

DEPARTMENT

Name, Number, {Locations}, Manager, ManagerStartDate

PROJECT

Name, Number, Location, ControllingDepartment

EMPLOYEE

Name(FName, MInit, LName), SSN, Sex, Address, Salary,
BirthDate, Department, Supervisor, {WorksOn (Project, Hours)}

DEPENDENT

Employee, DependentName, Sex, BirthDate, Relationship

{}: multivalued attribute, (): composite attribute

- __: Key attribute를 표시
- **WorksOn** 같은 경우에는 Composite이면서 Multivalued인 경우이다.
- Employee의 이름은 겹칠 수 있으므로 DEPENDENT의 Key는 Employee와 DependentName을 합쳐 사용해야 한다.

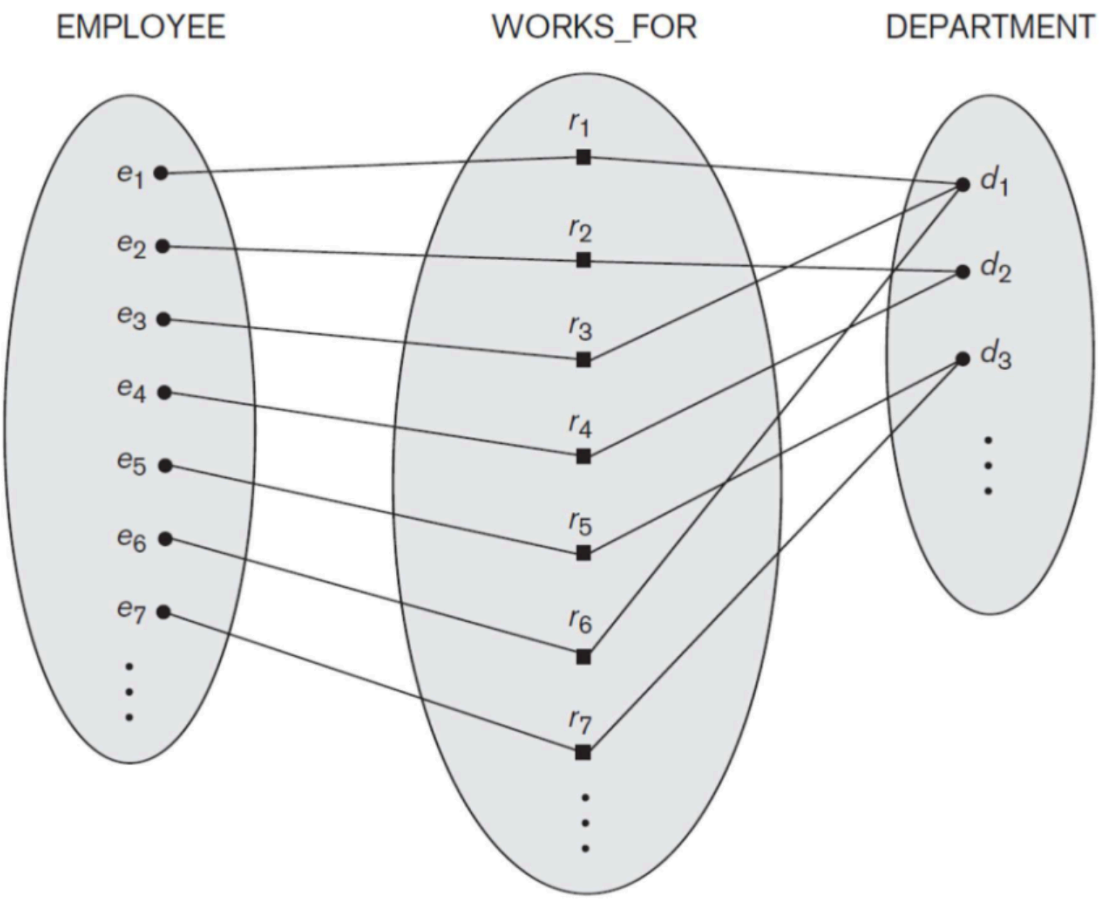
Relationship

Relationship instances: 두 Entity간의 Relation을 표현하는 것이다.

- Entity sets에서 뽑은 Entity간의 Association (실제 연결 관계)

Relationship type

- Entity type에서의 Relationship instance의 집합을 나타낸 것
- 참여하는 Entity types를 표현한다. (틀)



- Employee와 Department는 Entity type의 Entity set이다.
- **WORKS_FOR**는 Relationship type이다.
  - Relationship instance의 집합이다.
- r_i와 선은 Relationship instance를 의미한다.

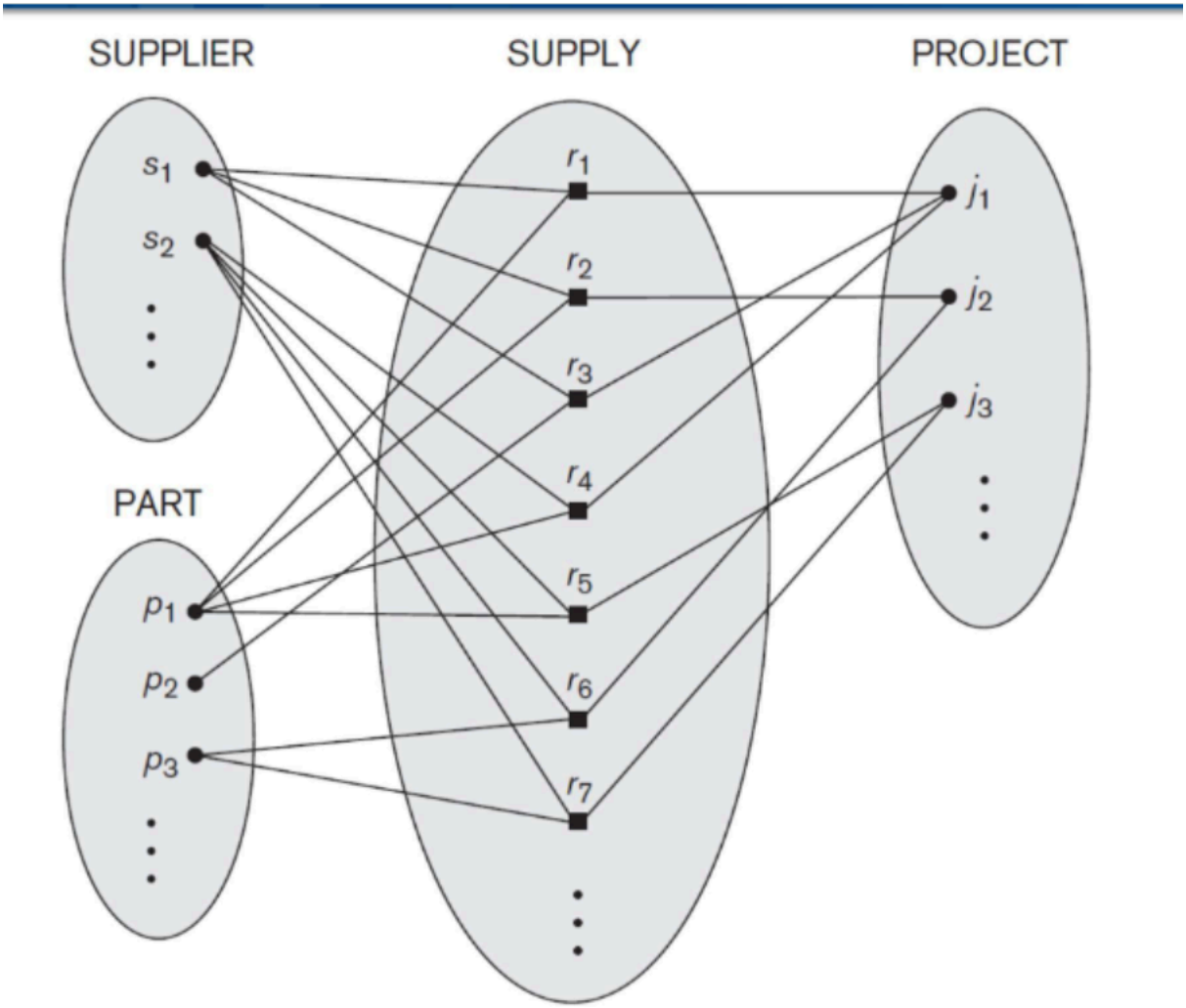
Relationship degree

- 참여하는 Entity type의 수를 의미한다.
- WorksOn의 경우에는 Binary이다.
- Binary, Ternary...

Binary Relationship

- Relationship degree가 2인 경우
- 가장 흔하게 사용된다.

Tenary Relationship



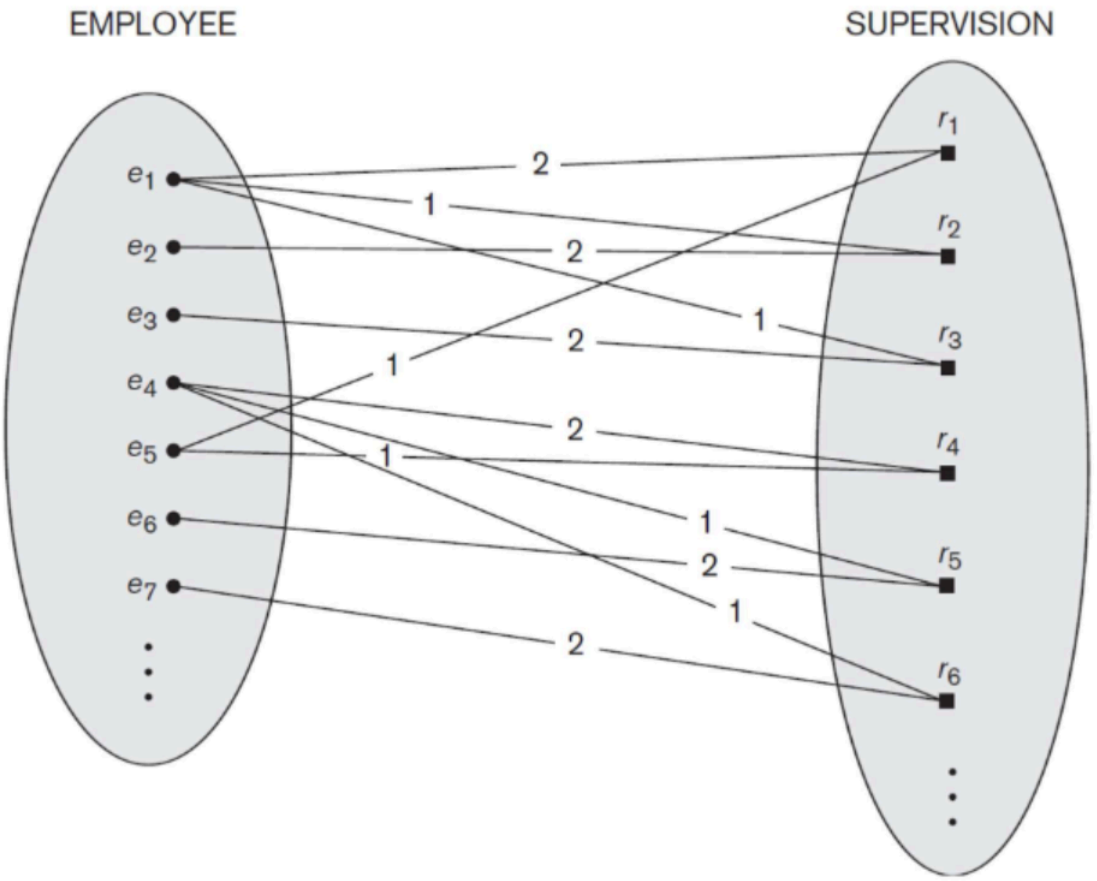
- s_i가 j_i에게 p_i를 공급하는 경우이다.

Role name

- **각 Relationship instance에서 Entity들이 하는 역할**을 명시한 것
- 위에서 배운 Relationship은 Relationship을 형성하는 Entity type이 달랐기 때문에 굳이 Role name을 명시할 필요가 없었다.
- 하지만 아래에서 볼 Recursive relationship에서는 Role name이 필수적이다.

Recursive relationship type

- **같은 Entity type이 다른 Role으로 두 번 이상 Relation type에 참여**하는 경우
- 반드시 Role name을 명시해야 한다.



- Employee는 Entity type, Supervision은 Relation type이다.
- **(1)는 상사를 의미하고 (2)는 부하직원을 의미**한다. (Role name)
- 예를 들어 r_2**는** e_1**을 상사로 갖고,** e_2**를 부하직원으로 갖는 Relationship instance**이다.