로지컬 DB 디자인

4. Requirement Analysis and Conceptual Data Modeling

Intro.

- Logical database design
  - Top-down, Bottom-up, Combined methodologies
- Traditional approach for relational database: Bottom up
- However, for large database
  - Combination of top-down and bottom-up approaches
  - Tables can be defined directly from the requirement analysis
- Conceptual data modeling: ER or UML
  - The most successful tool for communication between the designed and the end user during the requirement analysis and logical design phases
  - Why?
    - Easy to understand
    - Convenient to represent
    - Using entities as an abstraction for data elements and
    - Focusing on the relationships between entities

    → Reduce the number of objects under consideration

  - Dependencies are confined to other attributes within the entity
- Entity Keys
  - The unique identifiers of different entities

Requirement Analysis

- An extremely important step in the database life cycle
- The most labor intensive
- Basic objectives of requirement analysis
  - Delineate the data requirements of the enterprise in terms of basic data elements
  - Describe the information about the data elements and the relationships among them needed to model these data requirements
  - Determine the types of transactions that are intended to be executed on the database
  - Determine the interaction between the transactions and the data elements
  - Define any performance, integrity, security, or administrative constraints
  - Specify any design and implementation constraints
    - Specific technologies, hardware and software, programming languages, policies, standards, or external interfaces
  - Thoroughly document all of the preceding in a detailed requirements specification

Conceptual Data Modeling

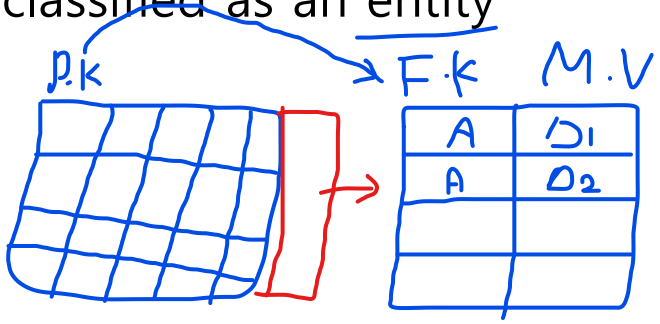
- Classify entities and attributes
- Identify the generalization hierarchies
- Define relationships

Classify Entities

- Project headquarters are located in cities
  - Should "city" be an entity or an attribute?
- Entities should contain descriptive information
- "City"
  - If there is some descriptive information such as "country" and "population" for cities, then "city" should be classified as an entity
  - If only the city name is needed to identify a city, then "city" should be classified as an attribute associated with some entity
- Exception: "State" → 속성이 게만
  - State entity that contains all the valid State instances
    현재 사실이 있는가 = F·K

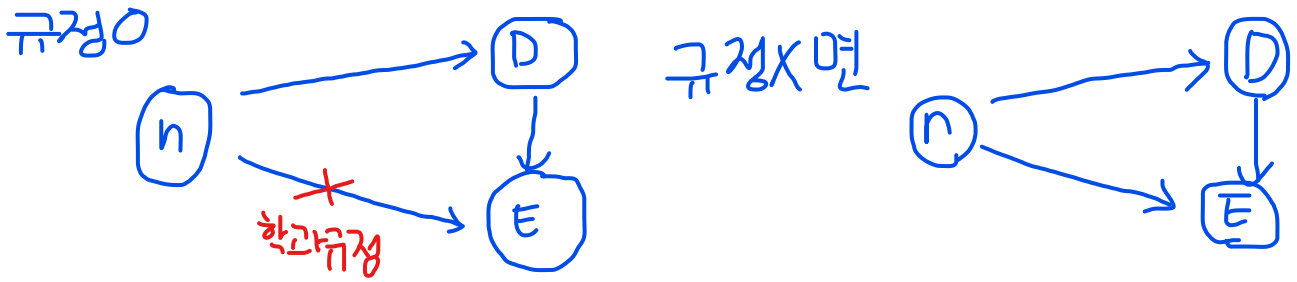
Classify Multivalued Attributes

- Classify multivalued attributes as entities even though it does not have descriptors itself
- Example: "Division"
  - "Division" could be classified as a multivalued attribute of "company"
  - It would be better classified as an entity



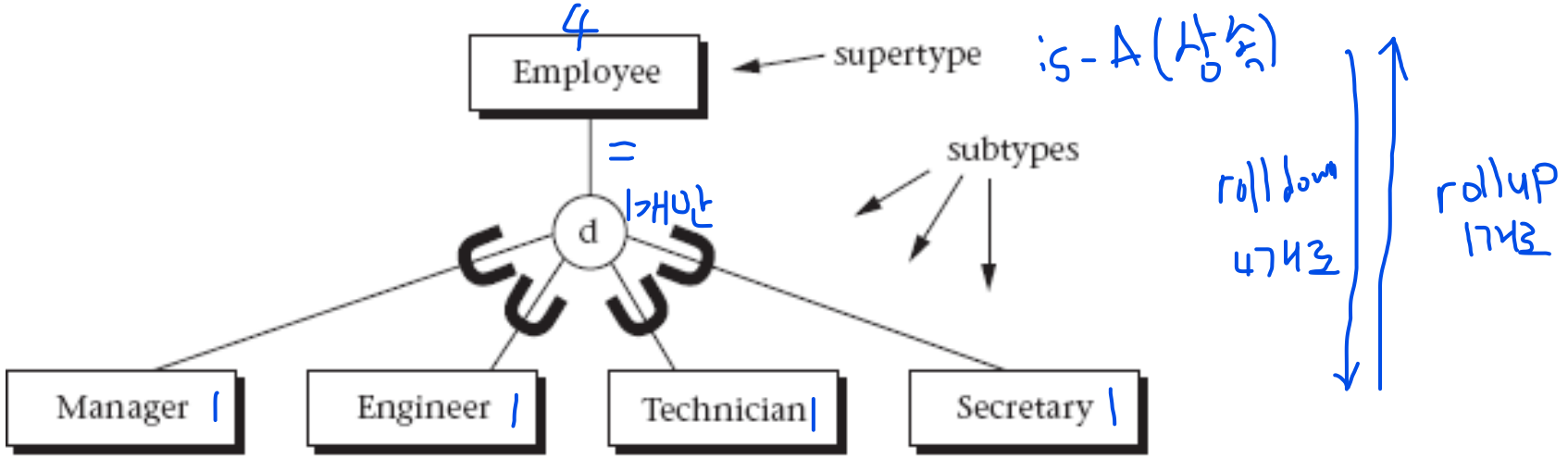
Attribute Attachment

- Attach attributes to the entities they most directly describe
  - "office-building-name" should normally be an attribute of the entity Department, rather than the entity Employee



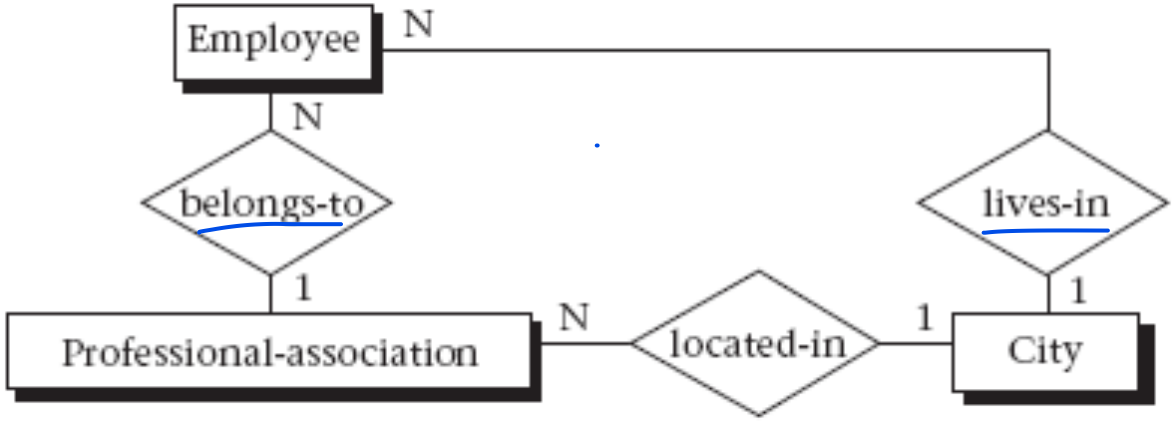
Identify the Generalization Hierarchies

- If there is a generalization hierarchy among entities
  - Put the identifier and generic descriptors in the supertype entity
  - Put the same identifier and specific descriptors in the subtype entities

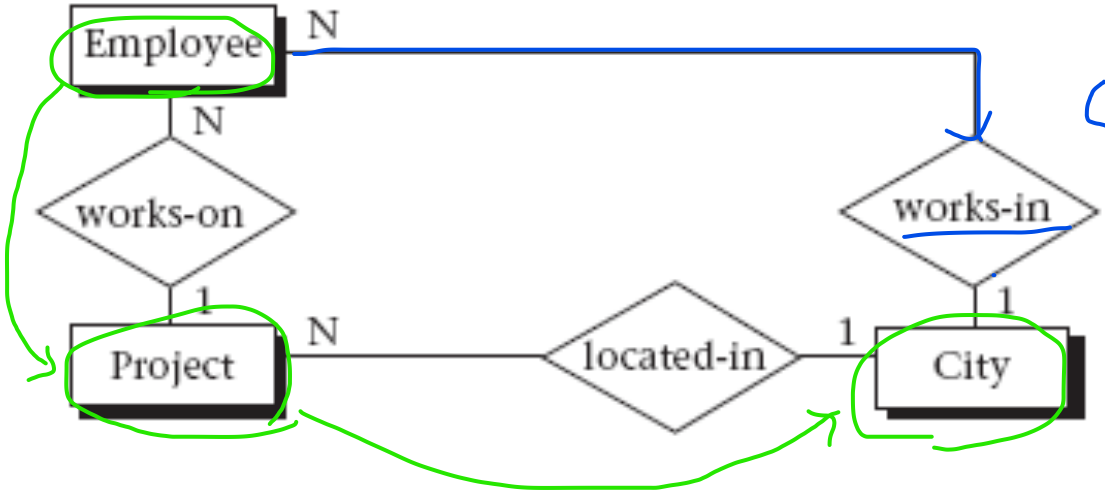


Redundant Relationships

- Nonredundant relationships



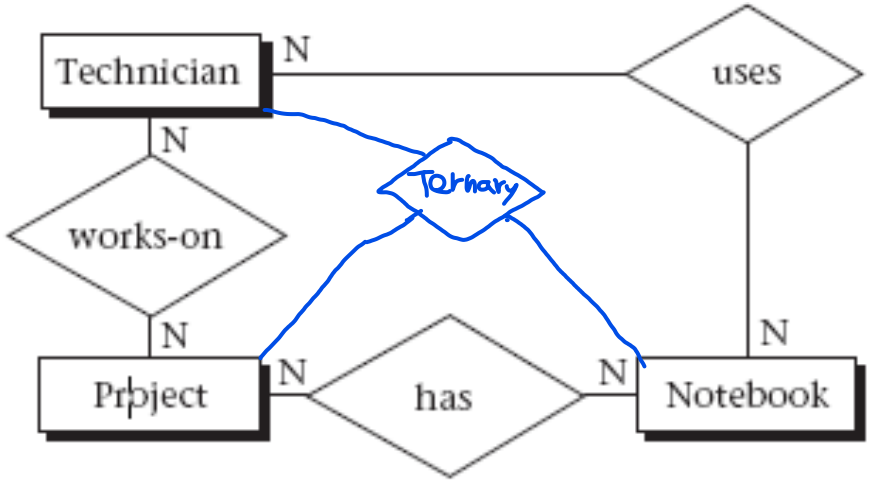
- Redundant relationships using transitivity 추론



(장점) 중복 → 신속, 의심, Documentation

Ternary Relationships ← 잘못된 설명

- We define a ternary relationship among three entities only when the concept cannot be represented by several binary relationships among those entities → ~~이항으로 X할 때만 만든다~~ → 관계들은 제일적이다
  - If each technician can be working on any of several projects and using the same notebooks on each project, then three many-to-many binary relationships can be defined



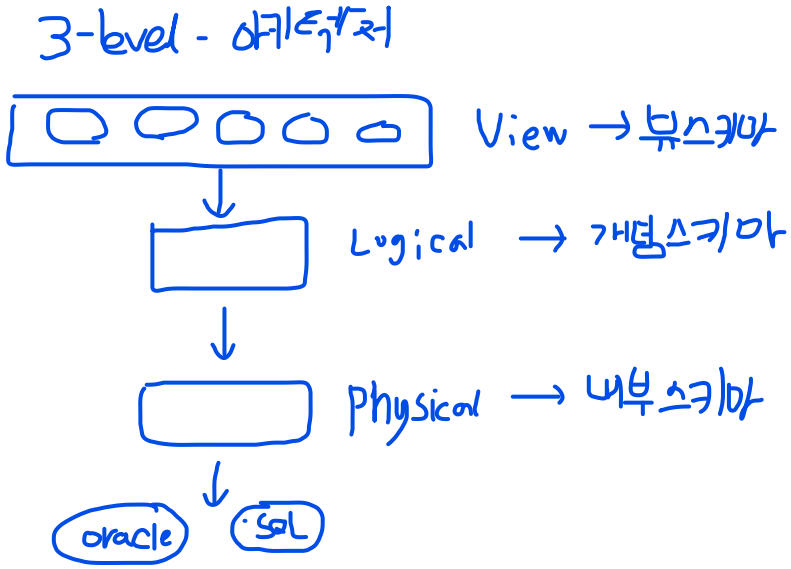
- The approach to take in ER modeling is to first attempt to express the associations in terms of binary relationships; if this is impossible because of the constraints of the associations, try to express them in terms of a ternary

Individual Views Based on Requirements

나눠서 통합

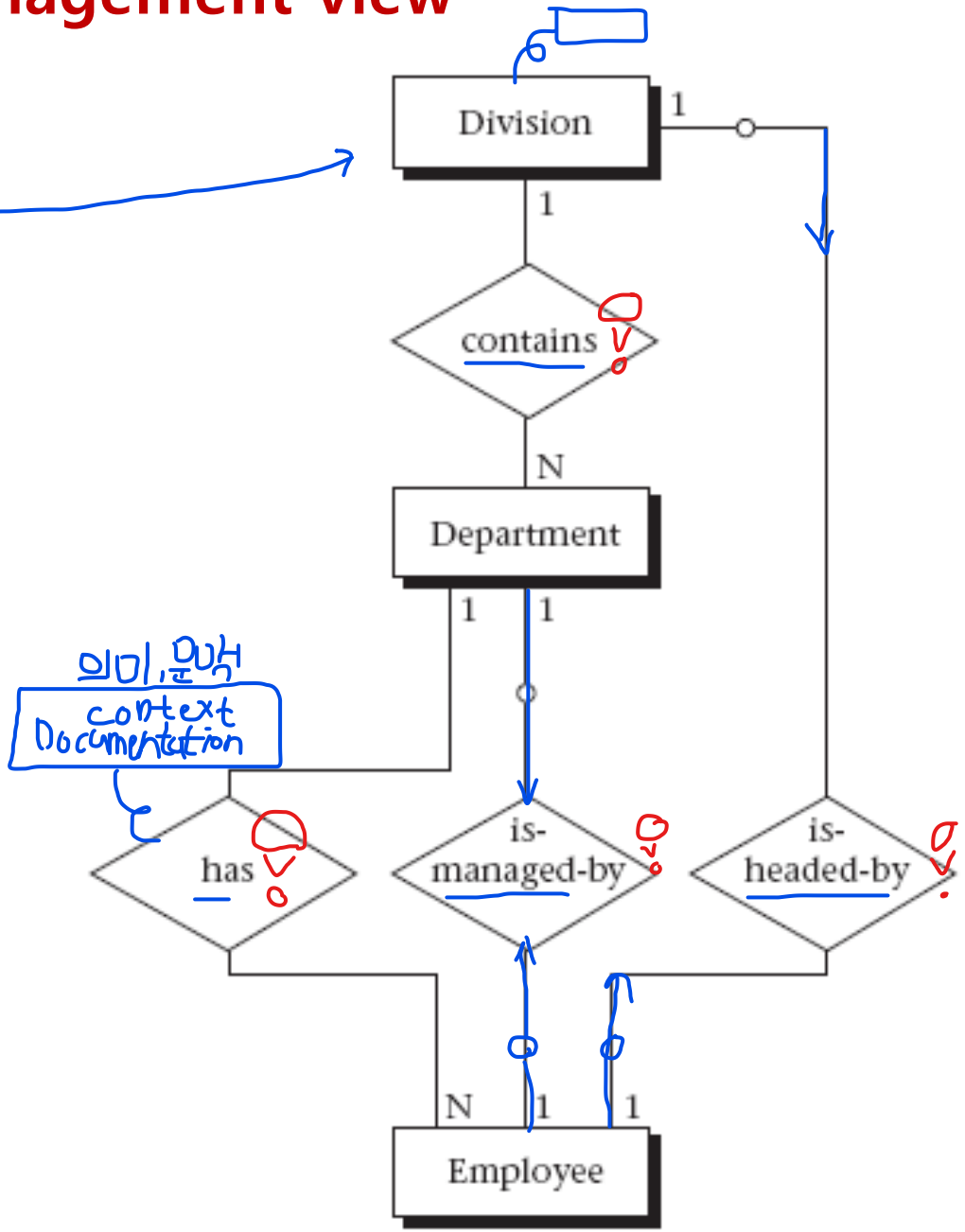
- Let us suppose it is desirable to build a company-wide database for a large engineering firm that keeps track of all full-time personnel, their skills and projects assigned, the departments (and divisions) worked in, the engineer professional associations belonged to, and the engineer desktop computers allocated

- During the requirements collection process, we obtain three views of the database: management view, employee view, Employee assignment view



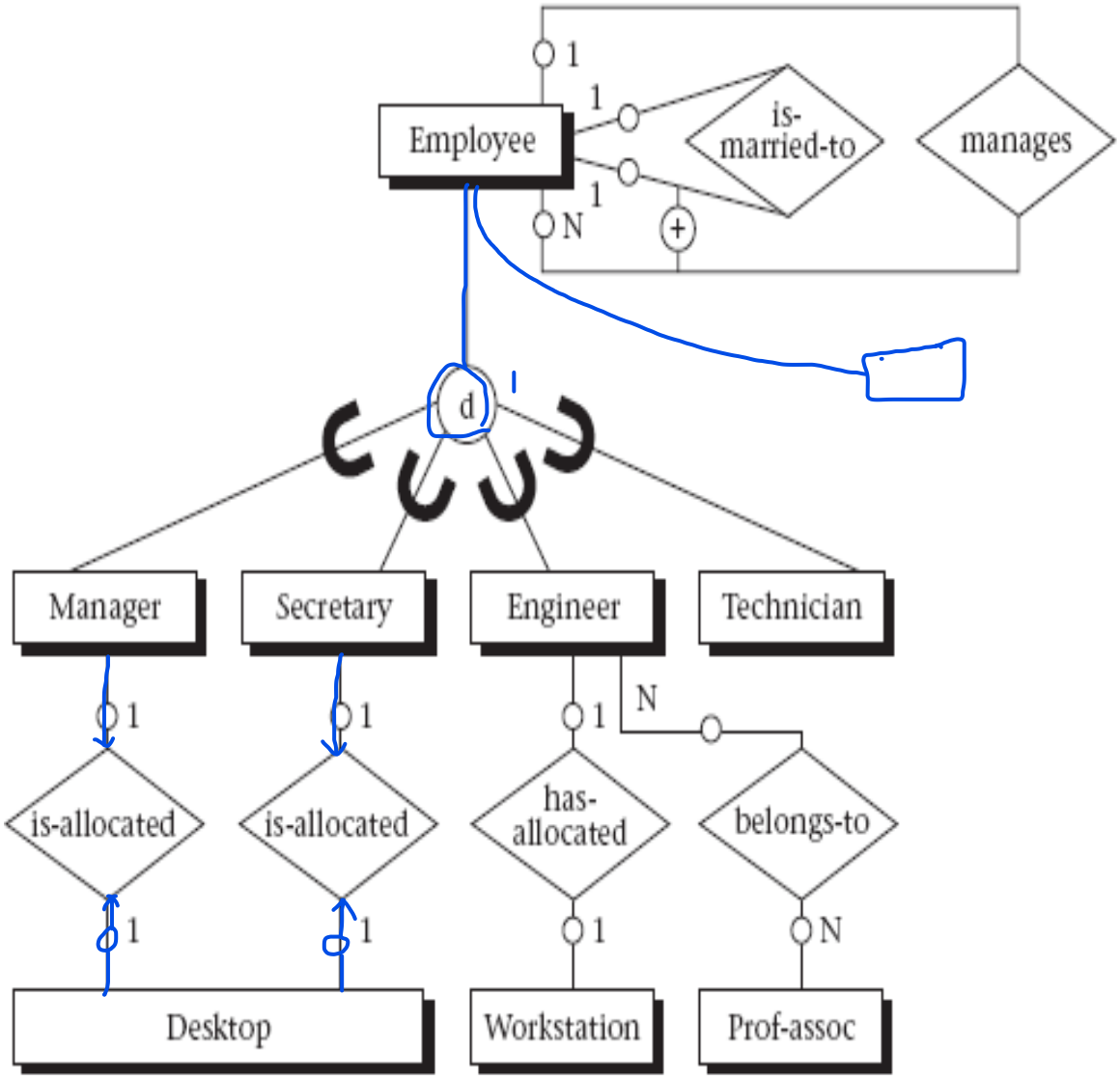
Management view

- Defines each employee as working in a single department
- Defines a division as the basic unit in the company, consisting of many departments
- Each division and department has a manager, and we want to keep track of each manager



Employee view

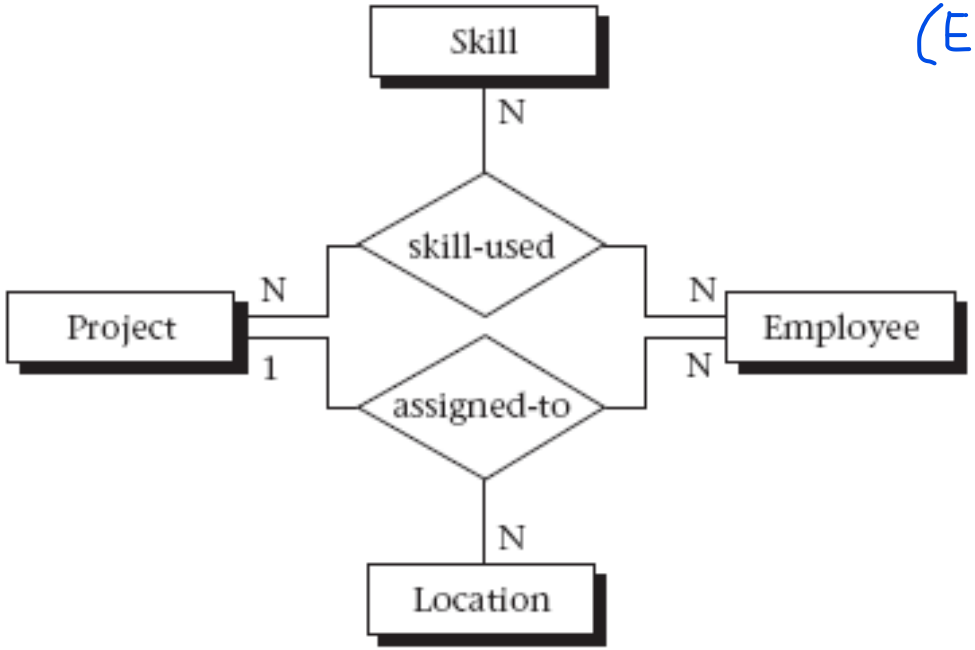
- Defines each employee as having a job title: engineer, technician, secretary, manager, and so on
- Engineers typically belong to professional associations and might be allocated an engineering workstation
- Secretaries and managers are each allocated a desktop computer
- A pool of desktops and workstations is maintained
  - For potential allocation to new employees
  - For loans while an employee's computer is being repaired
- Any employee may be married to another employee
  - We want to keep track of these relationships to avoid assigning an employee to be managed by his or her spouse



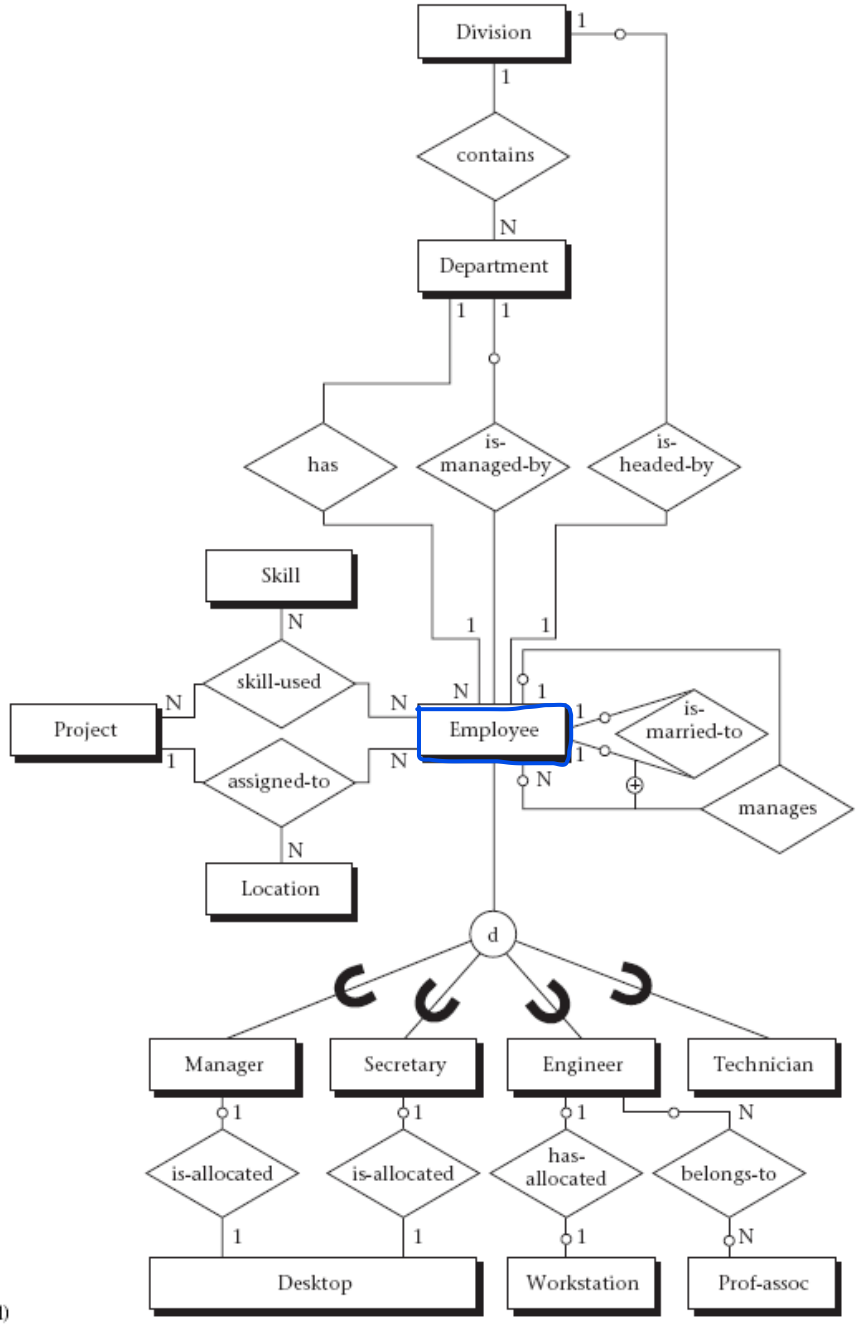
Employee assignment view

- The assignment of employees, mainly engineers and technicians, to projects
  - Employees may work on several projects at one time,
  - Each project could be headquartered at different locations (cities)
  - However, each employee at a given location works on only one project at that location
  - Employee skills can be individually selected for a given project
  - But no individual has a monopoly on skills, projects, or locations

(ERD에) Ternary – optional은 Documentation에



Global ER Schema



(d)

Global ER Schema

- Each relationship in the global schema is based upon a verifiable assertion about the actual data in the enterprise
  - Analysis of those assertions leads to the transformation of these ER constructs into candidate SQL tables
- All relationships, optional existence constraints, and generalization constructs need to be verified with the end user before the ER model is transformed to SQL tables

Benefits of : Application of the ER model to relational database design

- Use of an ER approach focuses end users' discussions on important relationships between entities
- A diagrammatic syntax conveys a great deal of information in a compact, readily understandable form
- A complete set of rules transforms ER constructs into mostly normalized
- SQL tables

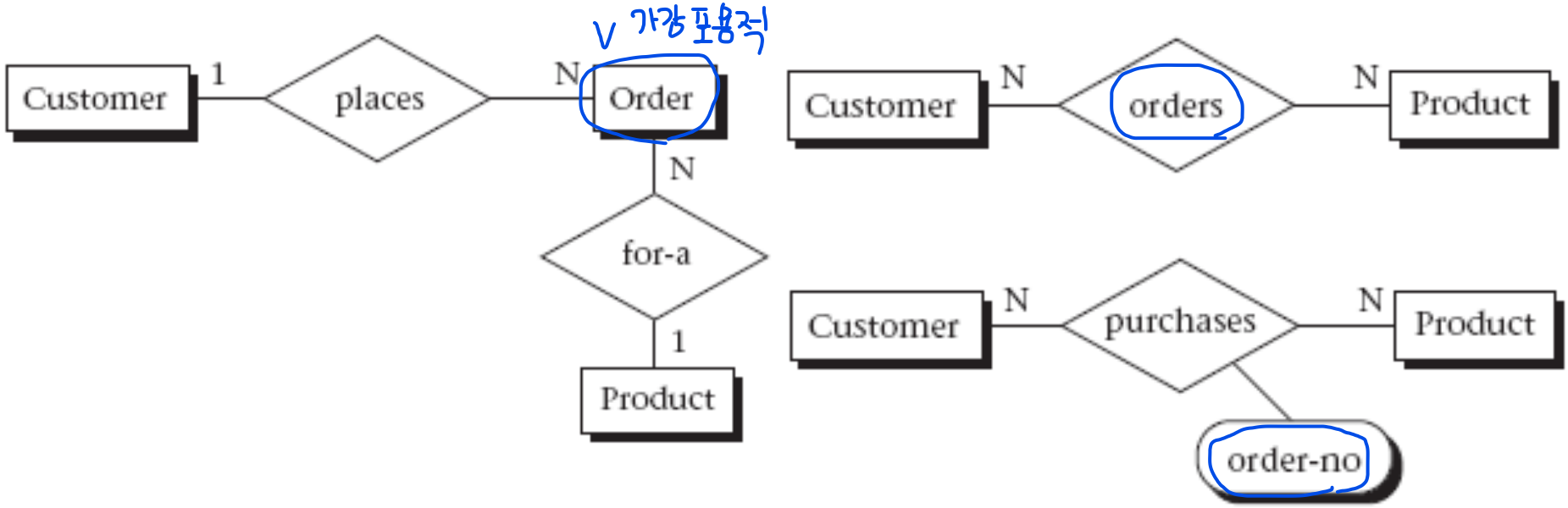
View Integration

- Integration of different user views into a unified, nonredundant global schema
  - The integrated conceptual schema results from sufficient analysis of the end-user views to resolve all differences in perspective and terminology

- Schema diversity occurs when different users or user groups develop their own unique perspectives of the world or, at least, of the enterprise to be represented in the database
  - The marketing division tends to have the whole product as a basic unit for sales
    - But the engineering division may concentrate on the individual parts of the whole product
  - One user may view a project in terms of its goals and progress toward meeting those goals over time
    - But another user may view a project in terms of the resources it needs and the personnel involved
- Such differences cause the conceptual models to seem to have incompatible relationships and terminology

View Integration

- These differences show up in conceptual data models as different levels of abstraction, connectivity of relationships (one-to-many, many-to-many, and so on), or as the same concept being modeled as an entity, attribute, or relationship, depending on the user's perspective
- Three different perspectives of the same real-life situation: the placement of an order for a certain product



Four basic steps needed for conceptual schema integration

1. Preintegration analysis
2. Comparison of schemas
3. Conformation of schemas
4. Merging and restructuring of schemas ←

Preintegration Analysis

- Choosing an integration strategy: Binary and N-ary approach
- Binary approach with two schemas merged at one time
  - Attractive because each merge involves a small number of data model constructs and is easier to conceptualize
- *N*-ary approach with *n* schemas merged at one time
  - May require only one grand merge
  - The number of constructs may be so large that it is not humanly possible to organize the transformations properly

Comparison of Schemas: Conflicts

- The designer looks at how entities correspond and
- Detects conflicts arising from schema diversity
- Naming conflicts
  - Synonyms
    - Occur when different names are given for the same concept
    - These can be detected by scanning the data dictionary
  - Homonyms
    - Occur when the same name is used for different concepts
    - These can only be detected by scanning the different schemas and looking for common names
- Type conflicts involve using different constructs to model the same concept
- Dependency conflicts result when users specify different levels of connectivity (one-to-many, etc.) for similar or even the same concepts
  - One way to resolve such conflicts might be to use only the most general connectivity → for example, many-to-many

Comparison of Schemas: Conflicts

- Key conflicts occur when different keys are assigned to the same entity in different views
  - Example: employee's full name, employee ID number, and Social Security number are all assigned as keys 후보키

Conformation of Schemas

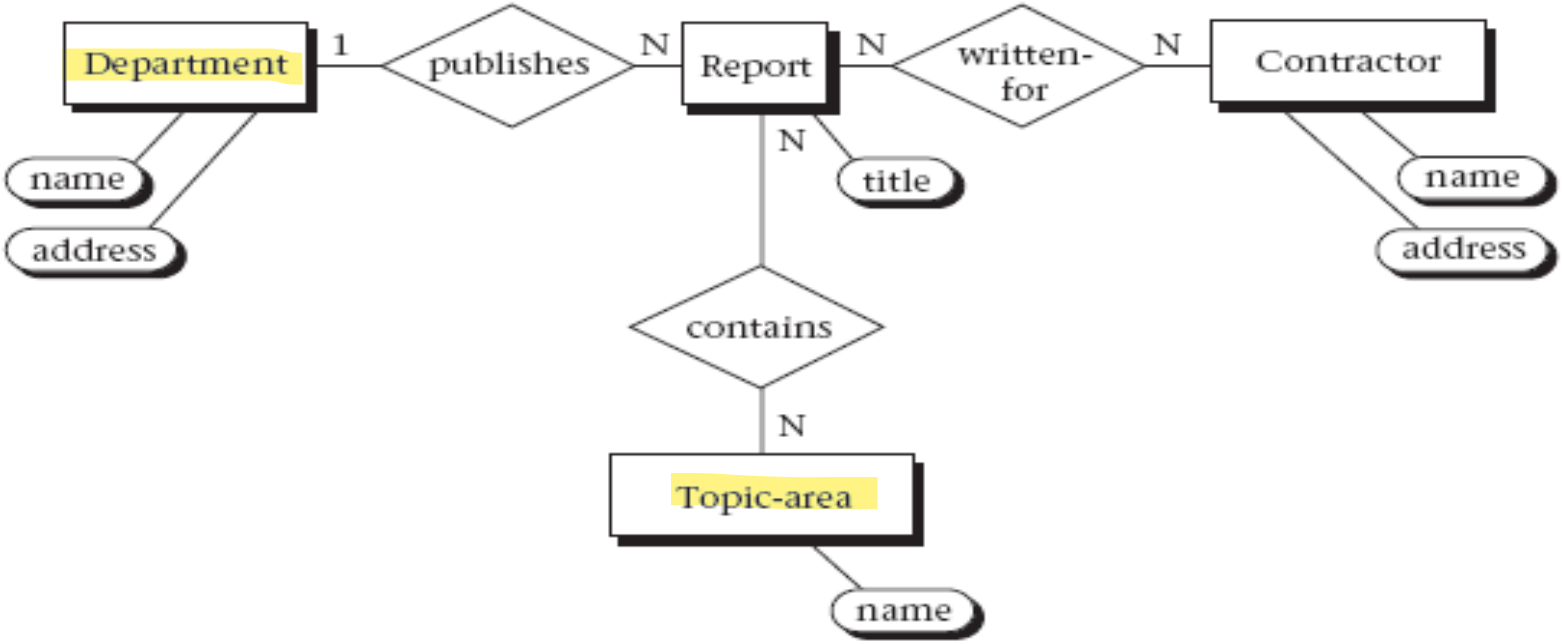
- The resolution of conflicts often requires user and designer interaction
- The basic goal of the third step(Conformation of Schemas)
  - To align or conform schemas to make them compatible for integration
- The entities, as well as the key attributes, may need to be renamed
- Conversion may be required so that concepts modeled as entities, attributes, or relationships are conformed to be only one of them
- Relationships with equal degree, roles, and connectivity constraints are easy to merge
  - Those with differing characteristics are more difficult and, in some cases, impossible to merge
  - Relationships that are not consistent consistent—for example, a relationship using generalization in one place and the exclusive OR in another—must be resolved
- Assertions may need to be modified so that integrity constraints remain consistent
- Techniques used for view integration
  - generalization, aggregation, the introduction of new relationships

Merging and Restructuring of Schemas

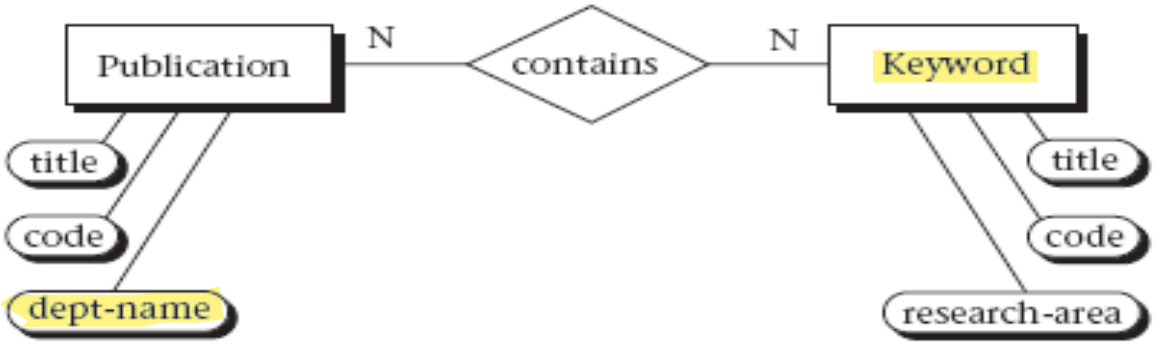
- Goal: Completeness, Minimality, and Understandability
- Completeness 완전성
  - Requires all component concepts to appear semantically intact in the global schema
- Minimality 최소성 (형태를 정할 때 이미 만족해야함)
  - Requires the designer to remove all redundant concepts in the global schema
  - Examples of redundant concepts are overlapping entities and truly semantically redundant relationships
    - Ground-Vehicle and Automobile might be two overlapping entities
    - A redundant relationship might occur between Instructor and Student. The relationships "direct research" and "advise" may or may not represent the same activity or relationship, so further investigation is required to determine whether they are redundant or not
- Understandability 이해성
  - Requires that the global schema make sense to the user.

Example of View Integration

- Find meaningful ways to integrate



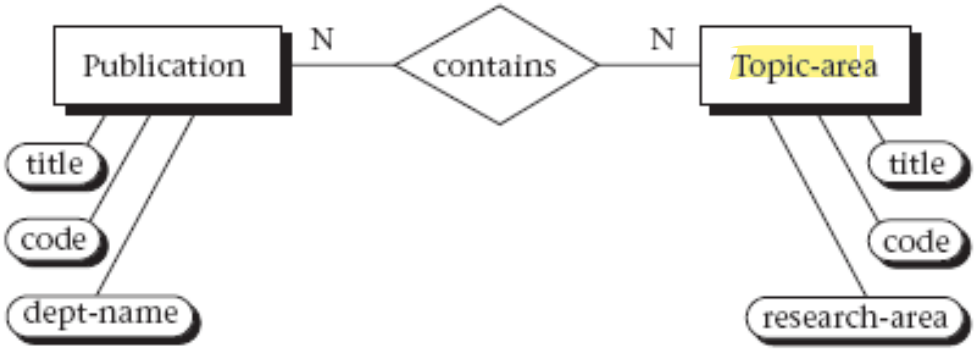
(a) Original schema 1, focused on reports



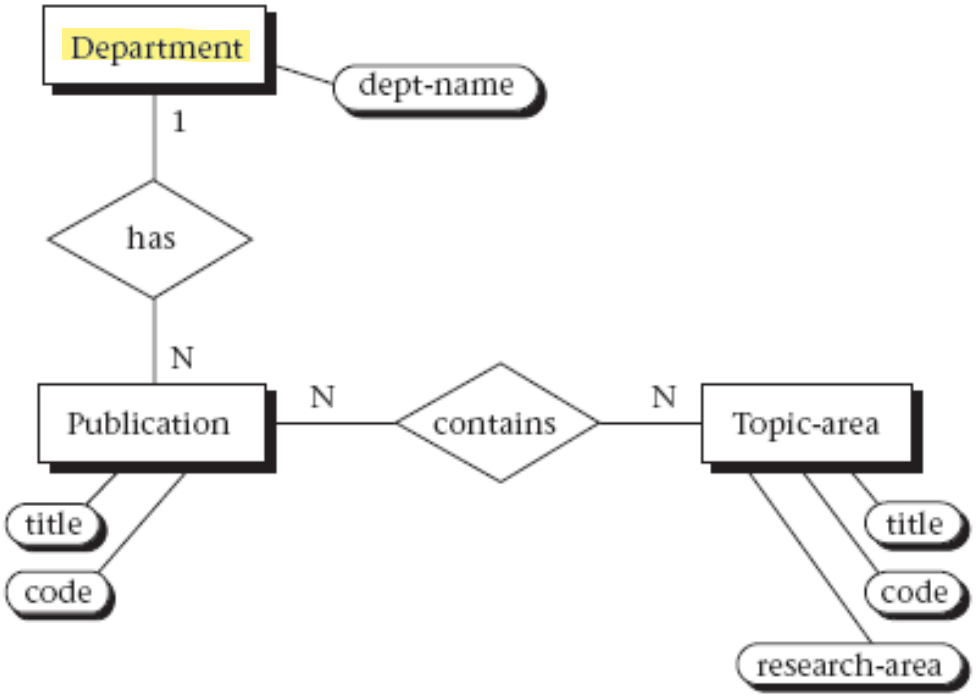
(b) Original schema 2, focused on publications

Example of View Integration

- We first look for synonyms and homonyms
  - A synonym exists between the entities Topic-area in schema 1 and Keyword in schema 2, even though the attributes do not match
  - However, we find that the attributes are compatible and can be consolidated.
- We look for structural conflicts between schemas
  - A type conflict is found to exist between the entity Department in schema 1 and the attribute "dept-name" in schema 2.1
  - The conflict is resolved by keeping the stronger entity type, Department, and moving the attribute type "dept-name" under Publication in schema 2 to the new entity, Department, in schema 2.2 (see Figure 4.6b)



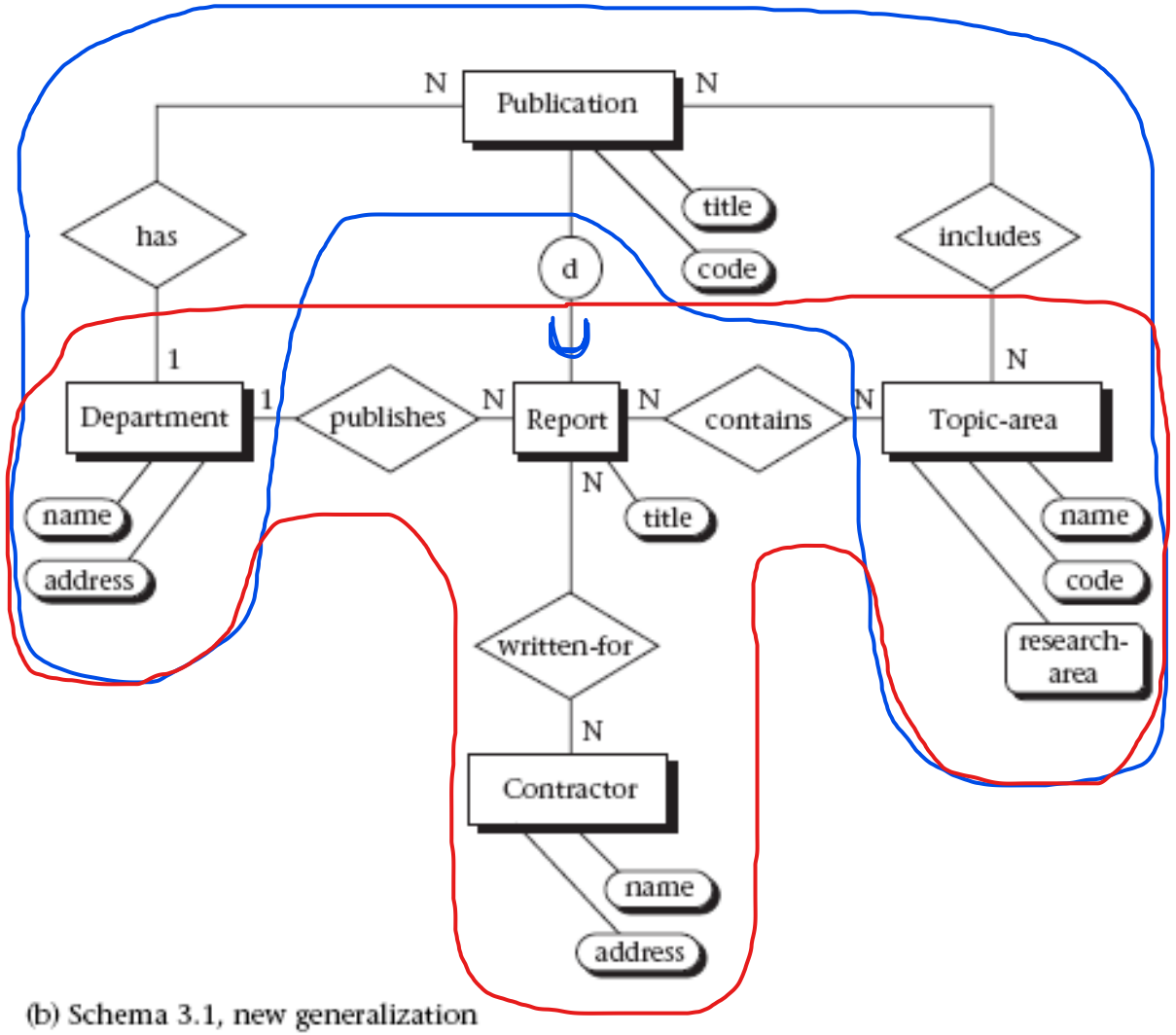
(a) Schema 2.1, in which Keyword has changed to Topic-area



(b) Schema 2.2, in which the attribute dept-name has changed to an attribute and an entity

Example of View Integration

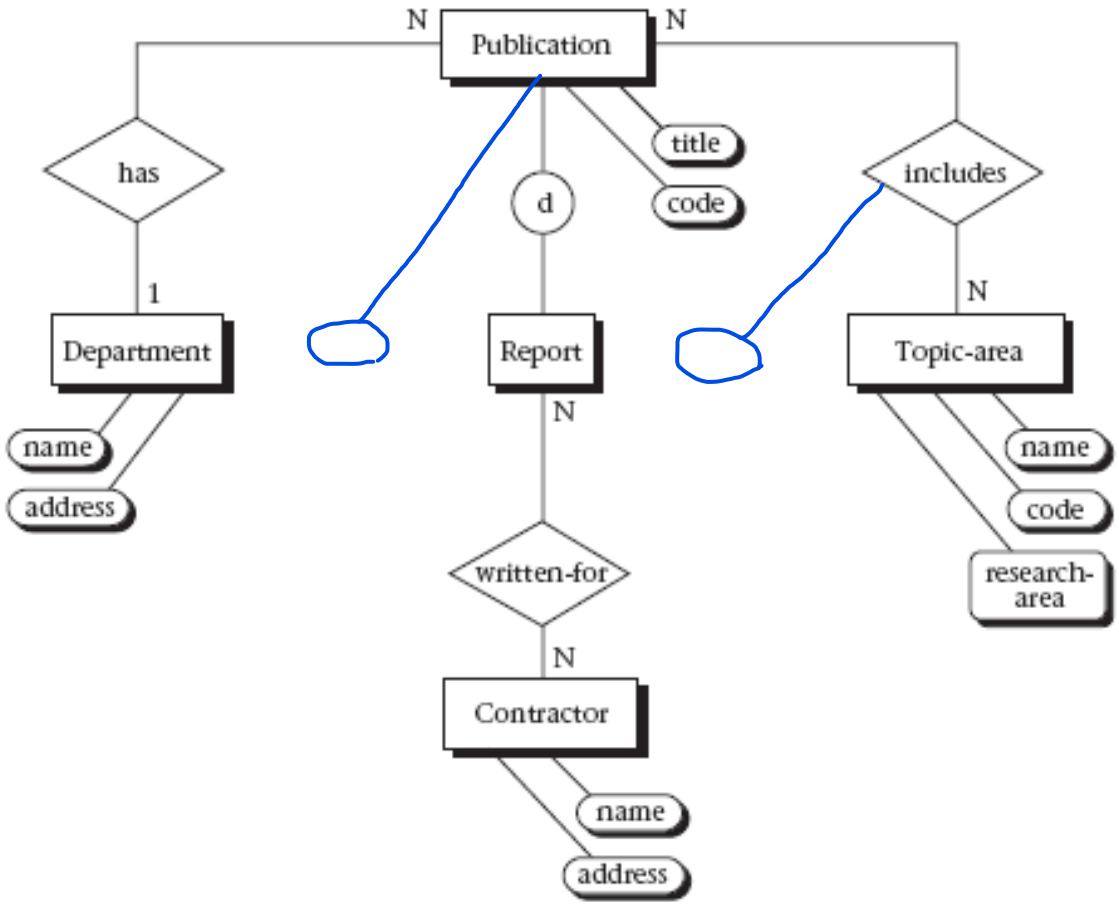
- At this point we have sufficient commonality between schemas to attempt a merge
  - In schemas 1 and 2.2 we have two sets of common entities, Department and Topic-area
  - Other entities do not overlap and must appear intact



(b) Schema 3.1, new generalization

Example of View Integration

- There is some redundancy between Publication and Report in terms of the relationships with Department and Topic-area
  - Such a redundancy can be eliminated if there is a supertype/subtype relationship between Publication and Report, which does in fact occur in this case because Publication is a generalization of Report



(c) Schema 3.2, elimination of redundant relationships