BCN (92)

K. H. DAVIS and A. K. ARORA. "A Methodology for Translating a Conventional File System into an Entity-Relationship Model." In J. LIU, ed., *Proc. Fourth International Conference on Entity-Relationship Approach*. Chicago. IEEE Computer Society Press, 1985.

E. G. NILSSON. "The Translation of Cobol Data Structure to an Entity-Relationship Type Conceptual Schema." *Proc. Fourth International Conference on Entity-Relationship Approach*. Chicago. IEEE Computer Society Press, 1985.

These two papers describe methodologies for translating a conventional file into an ER schema. The methodologies proceed by first translating the conventional file declaration into an intermediate data model, removing details about the physical implementation of the file. The intermediate model is then translated into the ER model.

H. BECK and S. B. NAVATHE. "Integrating Natural Language and Query Processing." *Proc. IEEE COMPCON*. San Francisco. Feb.–March 1990.

This paper discusses the mapping of natural language queries into a conceptual structure in the CANDIDE data model proposed by the authors. A lexicon is developed that is specific to the application domain, and reasoning about terms is handled using the conceptual hierarchies of clauses and attributes.

CHAPTER 5

View Integration

View integration is the process of merging several conceptual schemas into a *global conceptual schema* that represents all the requirements of the application. We have seen in Chapter 3 that it may be worthwhile to split the requirements of complex applications into different parts and then proceed by designing them separately. This may be a mandatory strategy when different analysts are involved in the design process.

The main goal of view integration is to find all parts of the input conceptual schemas that refer to the same portion of reality and to unify their representation. This activity is called **schema integration;** it is very complex, since the same portion of reality is usually modeled in different ways in each schema.

Integration is also required in another context, called **database integration,** which involves merging several different databases into a single database; in this case, we must first construct a conceptual schema of each individual database and then integrate those schemas. This activity is required for large information systems consisting of several databases; we discuss it in Section 7.3 (which deals with the development of *data dictionaries*). A special application of database integration occurs in *distributed database systems*, in which individual databases are stored on different computers in a computer network. Users of these systems should be provided with a unified view of the database that is transparent to data distribution and allocation. Besides encountering technological problems, designers of these systems may have to integrate existing databases.

d'que, se supõe que c/bd ho tenha um mod. cptual?

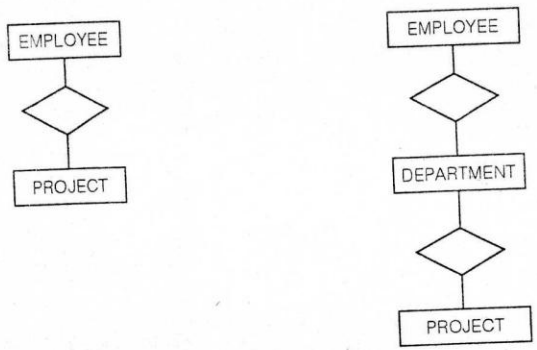
The organization of this chapter is as follows. In Section 5.1 we examine problems and issues that influence the integration activity. Section 5.2 deals with integration *in the large*, that is, with how to organize the integration of several schemas. Subsequent sections deal with integration *in the small*, that is, between two input schemas. Section 5.3 deals with conflict analysis and resolution, and Section 5.4 deals with the merging of views.

5.1 Issues in View Integration

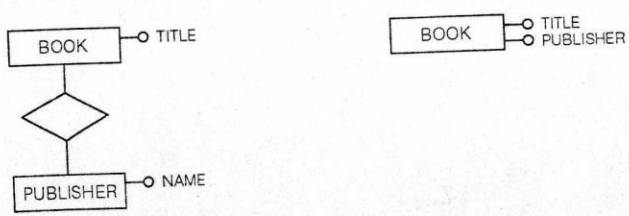
The main difficulty of view integration is discovering the differences in the schemas to be merged. Differences in modeling are due to the following causes.

Different Perspectives. In the design process, designers model the same objects from their own point of view. Concepts may be seen at different levels of abstraction, or represented using different properties. An example is given in Figure 5.1a: the relationship between EMPLOYEE and PROJECT is perceived as one relationship in one schema and as the combination of two relationships in the other schema.

Equivalence among Constructs in the Model. Conceptual models have a rich variety of representation structures; therefore, they allow for different equivalent representations of the same reality. For instance, in Figure 5.1b the association between book and publisher is represented by a relationship between the entities BOOK and PUBLISHER in one schema and as an attribute of the BOOK entity in the other schema. Similarly, in Figure 5.1c the partitioning of persons into males and females is represented by a generalization hierarchy among the entities PERSON, MAN, and WOMAN in one schema and by the attribute SEX of the entity PERSON in the other schema.

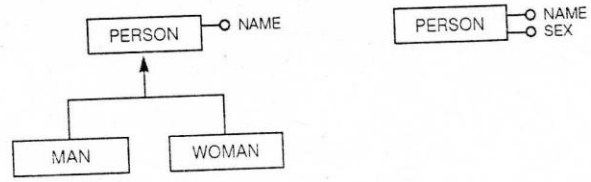


(a) Different perspectives



(b) Equivalent constructs

Figure 5.1 Reasons for schema diversity



(c) Equivalent constructs

(d) Incorrect specification

Figure 5.1 (cont'd) Reasons for schema diversity

Incompatible Design Specifications. Errors during view design regarding names, structures, and integrity constraints may produce erroneous inputs for the integration activity. During integration, these errors should be detected and corrected. For example, in Figure 5.1d the first schema indicates that each employee is always assigned to a unique project, but the second schema indicates that each employee works in multiple projects. Both schemas look correct, but one of them is wrong.

Each of these causes may result in **conflicts,** that is, different representations of the same concepts; the presence of conflicts that influence each other makes the integration process a nontrivial one. Our approach is shown in Figure 5.2. During **conflict analysis,** we look for conflicts; our approach highlights the importance of an early identification of conflicts. During **resolution,** one or both schemas are modified to resolve, or eliminate, each conflict; this must be done in consultation with the designer. During **schema merging,** schemas are superimposed and a preliminary integrated schema is obtained. Since all conflicts have already been resolved, this activity is very simple.

5.2 View Integration in the Large

In large database design projects, it is quite common to produce tens or even hundreds of different schemas that must be integrated. This process requires establishing a discipline for selecting an appropriate sequence of individual integrations of views. The most general approach to the integration process is shown in Figure 5.3; this approach proceeds with the integration of several schemas at a time, and therefore involves several coexisting, partially integrated schemas.

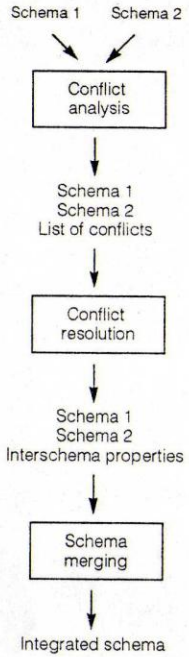


Figure 5.2 Our approach to view integration

The integration of many schemas at the same time is not very convenient, because it is quite difficult to discover conflicts. We suggest instead that only one pair of schemas be considered at a time; further, we suggest that the results of schema integration be accumulated into a single schema, which evolves gradually towards the global conceptual schema. This approach to view integration is shown in Figure 5.4: one schema is progressively enlarged to include new concepts from other schemas.

We must first choose the order of schema comparisons. If the integration process is performed according to a mixed (top-down and bottom-up) design strategy, as described in Section 3.2.4, then the *skeleton schema* should be chosen as the input to the first integration process. Schemas should be progressively aggregated with the skeleton schema, which performs a pivotal function. The order in which the other schemas are considered is not particularly relevant. If no skeleton schema is available, the designer should establish a priority among schemas, based on their importance, completeness, and reliability. The most important schema, called the **managerial schema,** should be considered in the first integration step. An obvious advantage of this policy is that integration of the most relevant schemas is accomplished first, which leads to better convergence and stability of the design process. Furthermore, when conflicts must be resolved, the solution is based on the most relevant views.

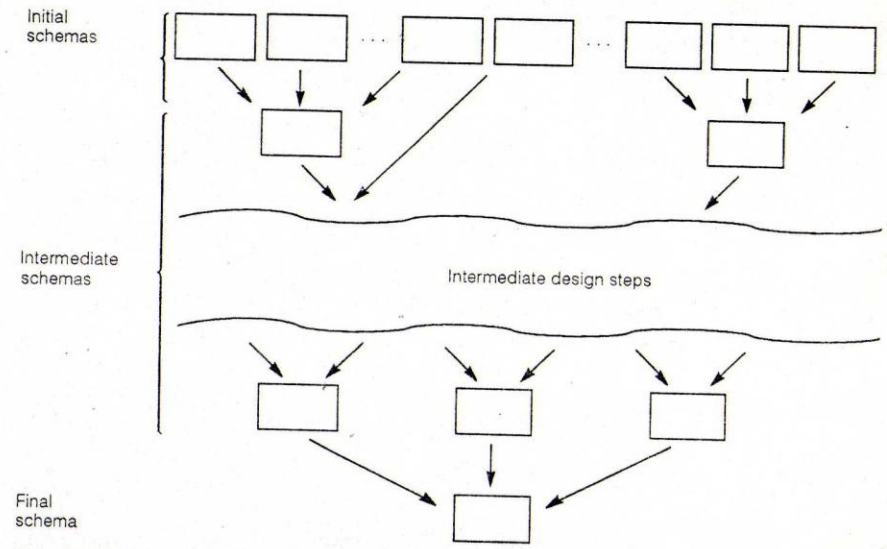


Figure 5.3 The most general approach to view integration

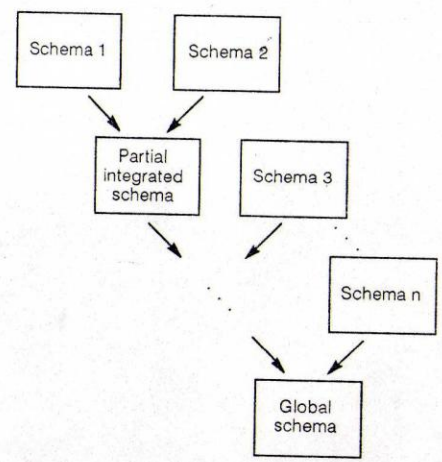


Figure 5.4 The suggested sequence of view-integration activities

5.3 Conflict Analysis and Resolution

We now concentrate on the integration process *in the small*, that is, between a pair of schemas, using an example that deals with the management of a library. The input schemas are shown in Figure 5.5. The scientist schema (Schema 1) describes the structure of the private library of a researcher. The librarian schema (Schema 2) describes the structure of the central library of a department. Tables 5.1 and 5.2 describe concepts of Schemas 1 and 2 whose meanings are not obvious.

Conflict analysis aims at detecting all the differences in representing the same reality in the two schemas. Two main tasks may be distinguished: (1) **name conflict analysis,** in which names of concepts in the schemas are compared and unified; and (2) **structural conflict analysis,** in which representations of concepts in the schemas are compared and unified.

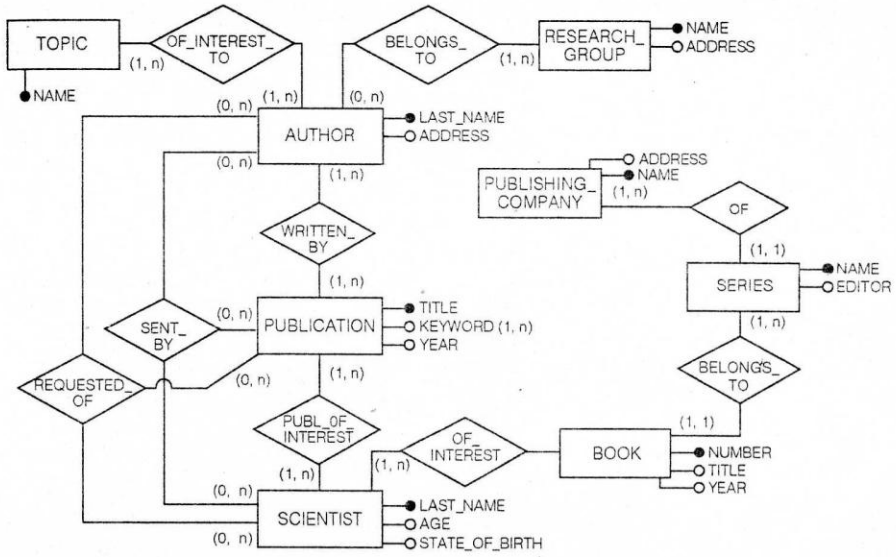
5.3.1 *Name Conflict Analysis*

There are two sources of name conflicts: synonyms and homonyms. **Synonyms** occur when the same objects of the application domain are represented with different names in the two schemas; **homonyms** occur when different objects of the application domain are represented with the same name in the two schemas. In discovering synonyms and homonyms, the designer is guided by a similarity or a mismatch among concepts, which may suggest the presence of a naming conflict. **Concept similarity** arises when concepts with different names have several common properties and constraints in the schemas. Similarity of two concepts indicates that they may be synonyms. **Concept mismatch** arises when concepts with the same name have different properties and constraints in the schemas. Mismatch of two concepts indicates that they may be homonyms.

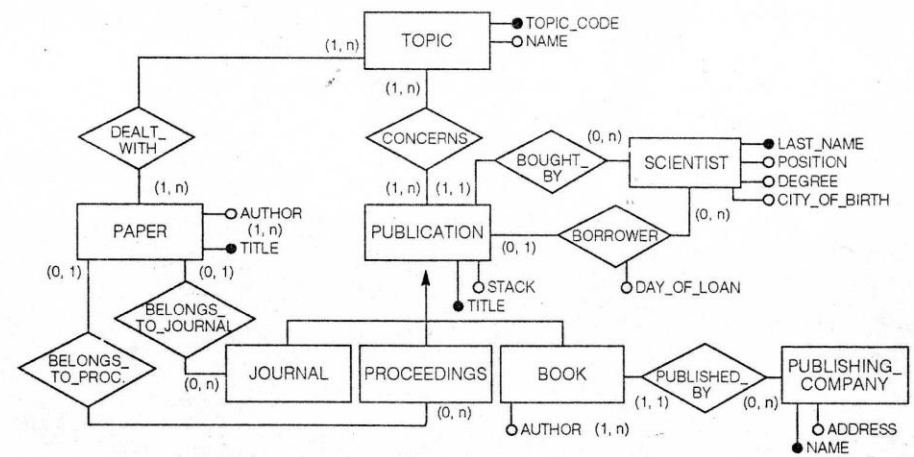
The terms *properties* and *constraints* in these definitions are defined as follows. **Properties** of a concept are all other *neighbor concepts* in the schema. For instance, the properties of a given entity are all its attributes, as well as the relationships, subsets, and generalizations in which it participates. **Constraints** are rules or conditions that limit the set of valid instances of the schema. These include, for instance, cardinality constraints for relationships and generalizations. Table 5.3 shows neighbor properties and constraints for entities, relationships, and attributes.

As a consequence of detecting similar and mismatched concepts, several possible modifications can be performed on the schemas. We call the set of all such possible modifications *modification scenarios*. The modification scenarios for the naming conflict involve renaming the concept; they can also involve adding some *interschema property*. Let us explain both of these ideas.

Concept renaming is performed (i.e., a concept is renamed) whenever we detect a synonym or a homonym. Synonyms should be eliminated to remove ambiguity. For example, when two concepts such as CUSTOMER and CLIENT are synonyms, then we select one of them, say CUSTOMER, and rename CLIENT as CUSTOMER. For the case of a homonym, suppose that REGISTRATION in one view refers to the process of registering a car renter, whereas in another view it means making a reservation for the car. In this case, the concept in the second view should be renamed as RESERVATION.



(a) Scientist schema (Schema 1)



(b) Librarian schema (Schema 2)

Figure 5.5 Input schemas for view integration

Table 5.1 Concepts in Schema 1 (Scientist Schema)

Name	Description
Author	Authors of publications of interest to scientists
Publication	Publications kept by scientists in their private cabinets; they are usually directly obtained by scientists from authors
Topic	Research areas of interest to authors
Requested of	Connects papers that have been requested by some scientist to the author of whom the request has been made
Sent by	Connects papers that have been sent by authors to scientists who have requested them

Interschema properties express mutual constraints between concepts appearing in different schemas. For example the entity PHD_CANDIDATE in one view may be constrained to be a subset of the entity STUDENT in another view. These properties should be annotated as extensions of the two schemas; they are later used in merging the schemas.

Figure 5.6 shows an example of possible modification for a naming conflict. Schema 1 contains the entity CLERK; Schema 2 contains the entity MANAGER. In the first scenario, the names are considered as synonyms, and the name in the second schema is changed. In the second scenario, the names refer to different concepts, related by a subset. In the third scenario, the names refer to different concepts that are related by a generalization hierarchy with a third, more general concept. The subset and generalization hierarchy are examples of interschema properties.

Examples of similarity and dissimilarity among concepts can be found in the schemas represented in Figure 5.5:

1. The entity TOPIC appears in both schemas, although with different properties. In Schema 1, TOPIC refers to an author's interest, whereas in Schema 2 it refers to arguments of publications. This is an example of concept mismatch.
2. The entity PUBLICATION refers in the first schema to an object that is requested of and sent by authors; in the second schema, it refers to an object that is bought by (or borrowed by) a scientist. This is also an example of concept mismatch.
3. The attribute KEYWORD with the entity PUBLICATION in the first schema and the entity TOPIC in the relationship CONCERNS with PUBLICATION in the second schema have the same min- and max-cards. This is an example of concept similarity.

Table 5.2 Concepts in Schema 2 (Librarian Schema)

Name	Description
Publication	Publications presently kept in the library
Paper	Papers published in journals or proceedings kept in the library
Topic	Topics of papers
Bought by	Indicates which scientist is responsible for the grant used to purchase the publication

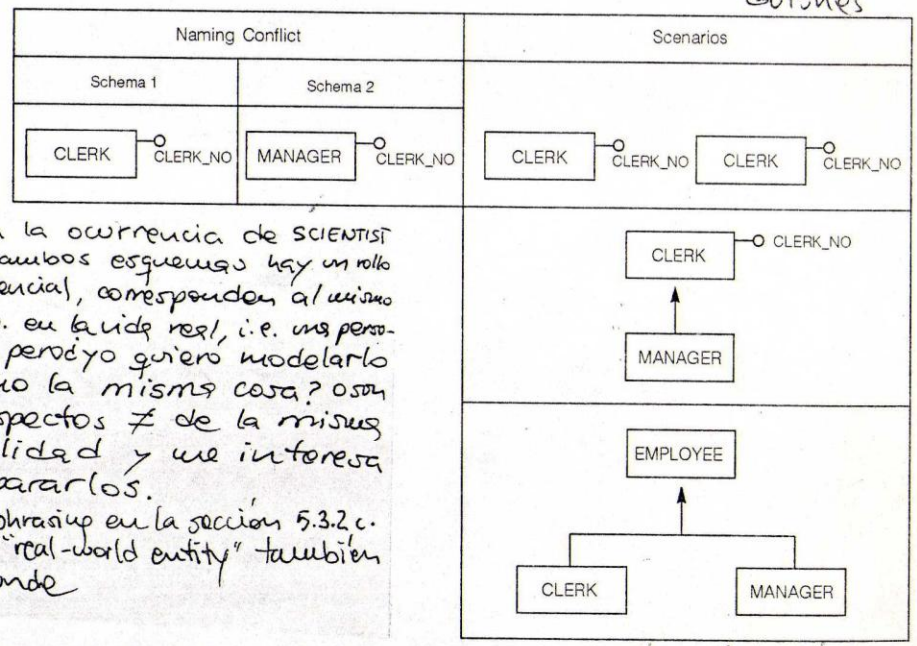
Table 5.3 Neighbor properties and constraints

Schema Element	Neighbor Properties	Constraints
Entity	Their attributes, adjacent relationships, subsets, and generalization hierarchies	Min-card, max-card in relationships where the entity is a participant; identifiers
Relationship	Their attributes; participating entities	Min-card, max-card of the participating entities
Attributes	Entities or relationships to which they belong	Min-card, max-card, value set, identifiers that include the attribute

Homonyms in Cases 1 and 2 are resolved by renaming TOPIC and PUBLICATION in the first schema to RESEARCH_AREA and PAPER respectively. Notice also that the relationship PUBL_OF_INTEREST in the first schema must then be renamed as PAPER_OF_INTEREST. The synonyms in Case 3 are avoided by renaming KEYWORD as TOPIC in the first schema.

Not all naming conflicts can be discovered by using concept similarity or mismatch. For instance, consider the entity SCIENTIST, which in Schema 1 plays the role of getting and

Guiones



Con la ocurrencia de SCIENTIST en ambos esquemas hay un rollo potencial, corresponden al mismo cpto. en la vida real, i.e. una persona, pero yo quiero modelarlo como la misma cosa? o son 2 aspectos ≠ de la misma realidad y me interesa separarlos.
El phrasing en la sección 5.3.2 c. r.a. "real-world entity" también confunde

Figure 5.6 Examples of scenarios for modifying a naming conflict

Nosotros deberíamos aprepare una cuarta para que sea categoría.

being interested in papers and books, whereas in Schema 2 it participates in buying publications. In spite of their dissimilarity, the two entities indeed correspond to the same real-world concept.

5.3.2 *Structural Conflict Analysis*

After performing an analysis of naming conflicts, we achieve *name unification*. In this situation we assume that two concepts (attribute, entity, or relationship) with the same name represent the same concept in the real world. During structural conflict analysis, concepts with the same name in the input schemas are compared to see if they can be merged. We use the following categories:

1. *Identical* concepts, which have exactly the same representation structure and neighbor properties.
2. *Compatible* concepts, which have different representation structures or neighbor properties that are not contradictory. For instance, the use of an entity and an attribute to represent the same concept (see Figure 5.1b) is an example of the compatibility of concepts. Compatibility conflicts are easily solved by changing one of the two representations.
3. *Incompatible* concepts, which have properties that are contradictory. Sources of incompatibility must be eliminated before merging the schemas. Some of the possible incompatibilites in the ER model follow:
   a. *Different cardinalities* for the same attribute or entity.
   b. *Different identifiers:* an identifier in one schema is not an identifier in the other one.
   c. *Reverse subset relationships:* Entity A is a subset of Entity B in one schema, while B is a subset of A in the other one.

Possible solutions to incompatibility include: selecting one representation over the other or building a common representation such that all the constraints of the two schemas are supported in the integrated schema.

OPIC is
y an ex-
n Schemas!
?

Let us consider again our librarian and scientist schemas, and look for compatible and incompatible concepts. Two compatible concepts are:

1. SCIENTIST, which is an entity in both schemas, although it has different attributes and relationships. It represents the same object and needs no restructuring activity.
2. AUTHOR and TOPIC, which have different representation structures in the two schemas. AUTHOR should be transformed into an entity in Schema 2, whereas TOPIC should be transformed into an entity in Schema 1. In both cases suitable relationships are introduced.

Two cases of concept incompatibility arise. The first concerns the min-card of the entity PUBLISHING_COMPANY in the PUBLISHED_BY relationship with BOOK: it is 0 in Schema 2, but it is 1 (through the entity SERIES) in Schema 1. The former representation includes publishing companies that have not published any of the books currently in the library,

6:48 → 7:22
→ 7:27 →

while the latter case excludes these publising companies. We select the former alternative, since it is less restricted.

The second incompatibility refers to the AUTHOR concept. Schema 1 includes any author (as an entity), as long as there is a scientist who is interested in any of his or her publications. Schema 2 includes only authors (as attributes of PAPER) having at least one paper currently available in the library. Again, we select the former alternative, since it is less restrictive. The final products of conflict analysis are the two modified schemas of Figure 5.7.

5.4 Merging of Views

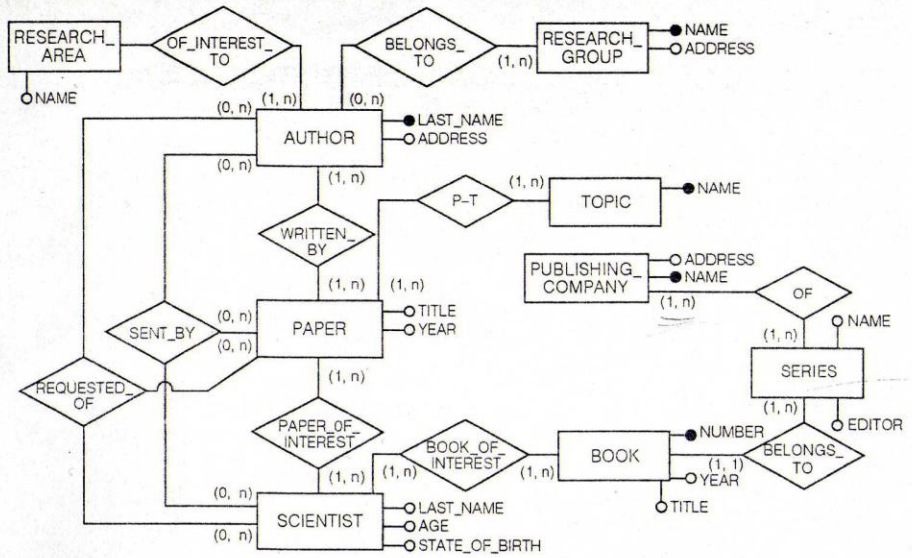
View merging acts on two input schemas and produces a schema that includes all concepts represented in the input schemas. At this stage of the design all conflicts have been solved, so schema merging is a simple superimposition of common concepts. Entities that fully match are superimposed directly (e.g., PUBLISHING_COMPANY). Entities that correspond to the same real-world objects but have different attributes are superimposed by taking the union of their attributes; all identifiers are reported in the result schema. If generalization hierarchies are present as interschema properties, they are added at this point. When entities are exhaustively superimposed, all relationships and generalizations from the input schemas are reported to the result schema.

Interschema properties may also lead to restructuring, additions, or deletions of concepts in the resulting schema. For example, consider the two entities TOPIC and RESEARCH_AREA. In the librarian application, research areas can be subdivided into topics, which represent a finer subdivision of disciplines. Hence, we can add a logical link between the entities RESEARCH_AREA and TOPIC by introducing a one-to-many relationship between them. The global schema of the example generated by the merging activity is shown in Figure 5.8.

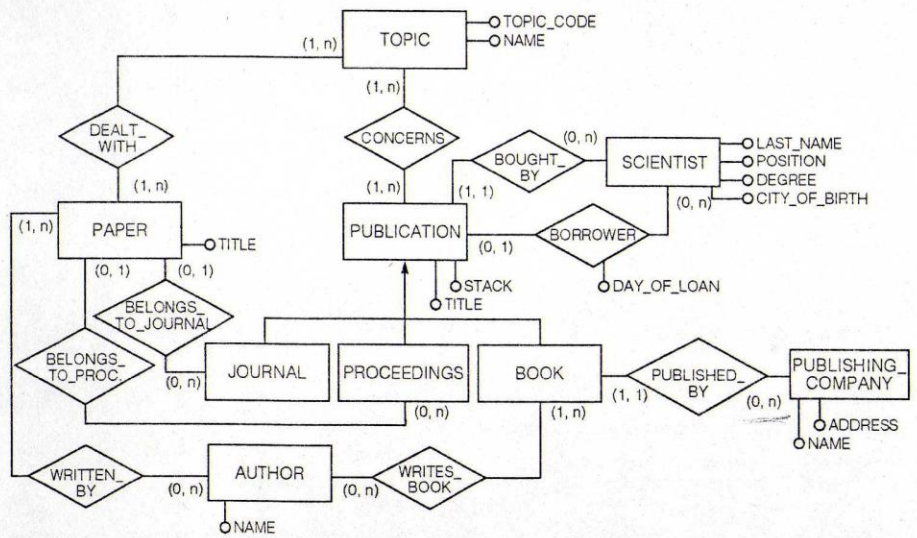
At the end of view merging we are ready to evaluate the quality of the integrated schema and then to try to improve it by performing restructuring operations. This activity is extensively described in Chapter 6.

5.5 Summary

View integration is a frequent activity in database design. In this chapter we have shown a methodology for view integration; in the large, we suggest producing the integration of several schemas by performing a sequence of integrations of two schemas at a time; the skeleton schema or the *managerial* view is taken as the starting point, and concepts are progressively aggregated to it. In the small, we suggest that each integration step be performed by first recognizing conflicts and then resolving them. We present one possible alternative for dealing with the detection and resolution of conflicts among views. Conflicts originate from different viewpoints or errors in naming or in structuring data; they can be resolved by renaming or by local data reorganization. We maintain information about conflicts as interschema properties. After conflict resolution, the final step of merging the two schemas is relatively easy.



(a) Scientist schema



(b) Librarian schemas

Figure 5.7 Modified schemas

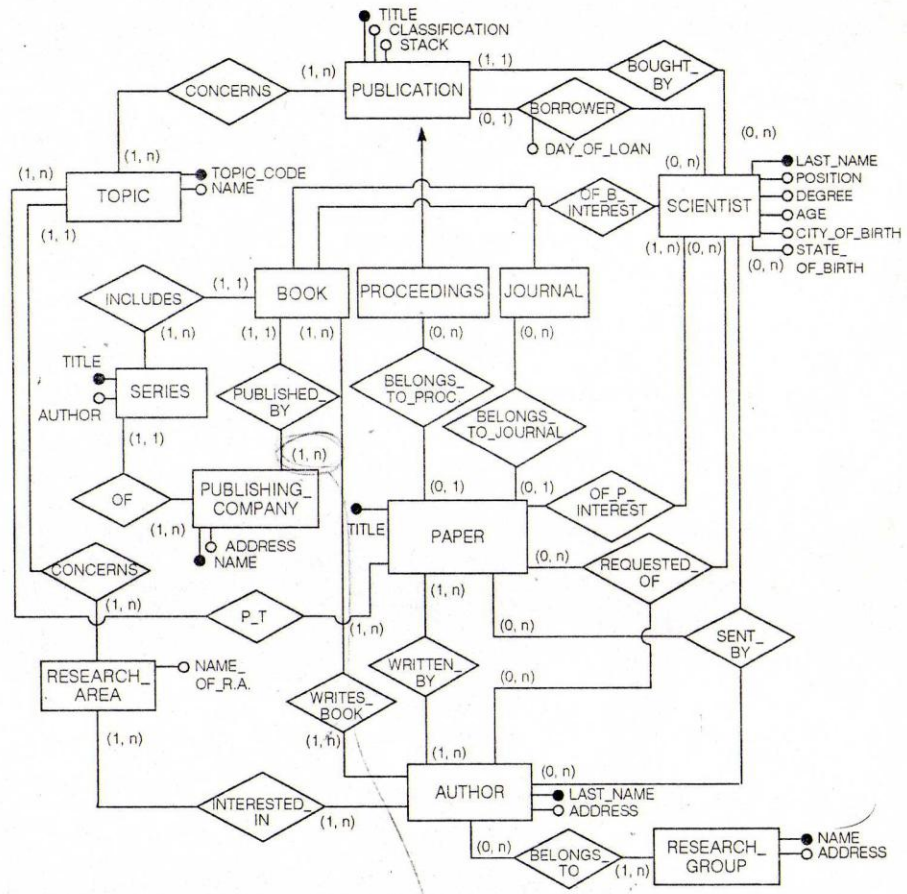


Figure 5.8 Global schema after merging

acababan de decir q' esto debia ser (0,n) Y no lo ponen

Exercises

5.1. Create several examples of the following:

a. Equivalent representations allowed in the Entity-Relationship model for the same real-world concept

b. Different perspectives for the same concepts in different schemas

c. Incompatible design specifications that result in different representations of the same properties

5.2. Perform this experiment with one of your classmates or colleagues. Identify an application domain well known to both of you (e.g., the structure of the computer center, the football team, a restaurant, etc.). Discuss the features of this application domain together, but not very precisely. Then produce two conceptual models. Now compare your schemas to see if they include the following:

a. Portions that have been modeled in exactly the same way

b. Portions that have been modeled with equivalent constructs

c. Parts that are not common and the interschema properties existing between them

d. Synonyms and homonyms

e. Errors

Then merge the schemas, and indicate the following graphically:

f. The subschemas containing all the concepts that are present in both input schemas

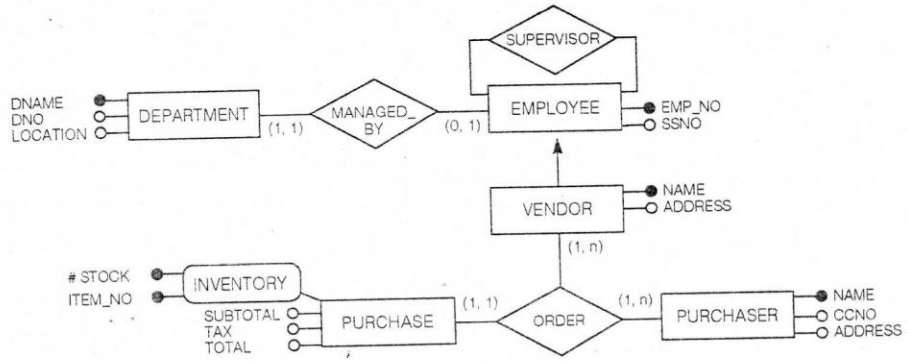
g. The subschemas containing all the concepts that are present in only one schema

5.3. Integrate the two schemas in Figure 5.9, which represent sales in a company and the structure of its departments and personnel, producing a single schema.

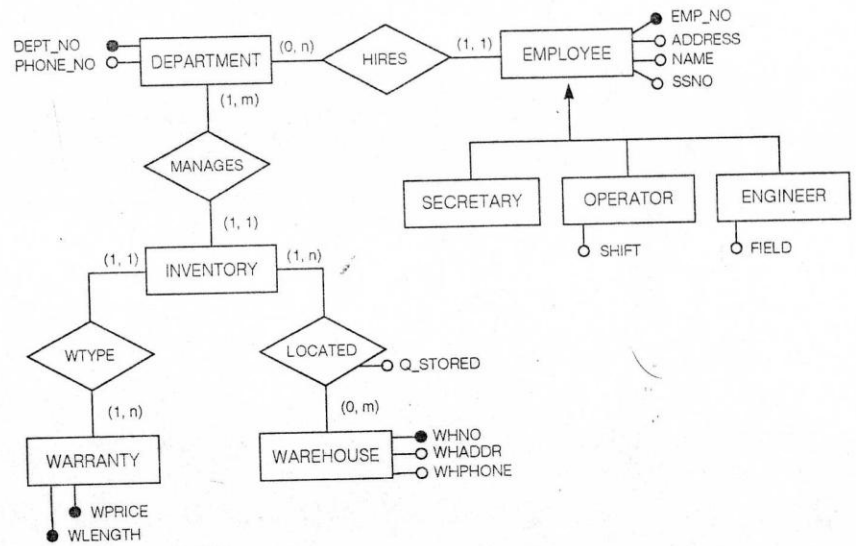
5.4. Integrate the three schemas in Figure 5.10, which represent special tours, daily travels, and reservations for daily travels, producing a single schema.

5.5. When a global schema has been produced as a result of an integration activity, it is useful to remember the relationships between ER constructs in the input schemas and ER constructs in the global schemas. Define a language that allows expressing such mapping.

5.6. The methodology discussed in this chapter produces a unique representation as a result of the integration process. Define a methodology that allows preserving several different user views of the same reality, relaxing the rules for merging concepts.

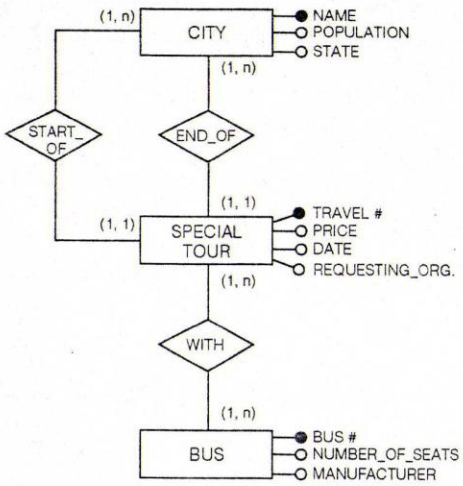


(a) First schema

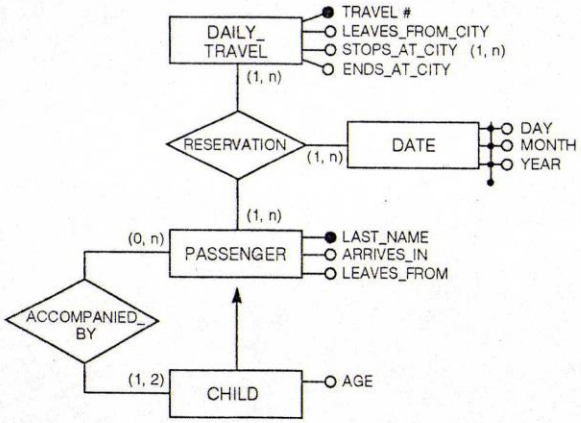


(b) Second schema

Figure 5.9 Company database

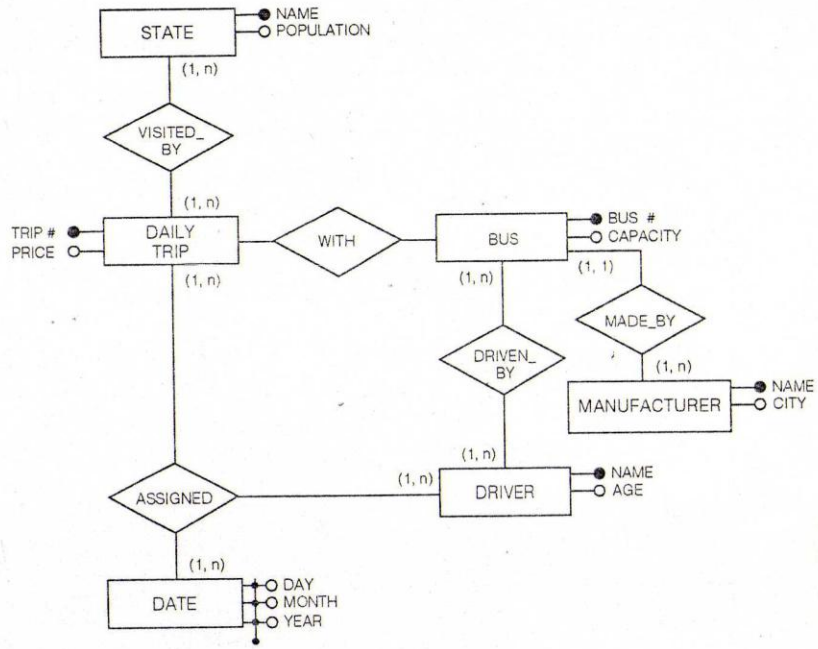


(a) First schema



(b) Second schema

Figure 5.10 Trip database



(c) Third schema

Figure 5.10 (cont'd) Trip database

Annotated Bibliography

C. BATINI, M. LENZERINI, and S. B. NAVATHE. "A Comparative Analysis of Methodologies for Database Schema Integration." ACM *Computing Surveys* 18, no. 4 (December 1986): 323–64.

This paper is a detailed survey of methodologies for database schema integration. The analysis covers both view integration and database integration, which concerns the problem of producing in a distrubuted environment an integrated schema from local schemas of existing databases. The first part of the paper discusses the role of the integration process in databases, office systems, and knowledge-based systems and then examines the influence of the conceptual model on the integration process. The central part of the paper compares methodologies according to the following criteria: order of schema constructs to be integrated, inputs and outputs considered, integration processing strategy, conflicts considered, and procedurality of the integration process. Future research directions are discussed.

C. BATINI and M. LENZERINI. "A Methodology for Data Schema Integration in the Entity-Relationship Model." *IEEE Transactions on Software Engineering* SE-10, no. 6 (November 1984).

R. ELMASRI and G. WIEDERHOLD. "Data Model Integration Using the Structural Model." In P. BERNSTEIN, ed., *Proc. ACM-SIGMOD International Conference*. Boston. 1979.

S. B. NAVATHE and S. G. GADGIL. "A Methodology for View Integration in Logical Data Base Design." In *Proc. Eighth International Conference on Very Large Databases*. Mexico City. 1982.

These publications describe three approaches to view integration that emphasize different aspects of the integration process. The first paper suggests several indicators to guide the designer in the investigation of conflicts. For every conflict, several scenarios, that is, typical solutions of the conflict, are proposed. A specific activity is also suggested to improve the readability of the global schema. The second paper gives a taxonomy for types of conflicts and for schema integration operations. The problem of automating the view-integration process is also considered, distinguishing activities that can be solved automatically and those that require interaction with the designer/user. The third paper considers the structural data model, which has a rich variety of modeling constructs. This variety of constructs gives rise to an extensive set of conflicts that are presented and analyzed.

A. MOTRO and P. BUNEMAN. "Constructing Superviews." In Y. LIEN, ed., *Proc. ACM- SIGMOD International Conference*. Ann Arbor, Mich. 1981.

This paper presents a methodology for data schema integration. The main feature of the methodology is that it provides a large and powerful set of restructuring primitives. Processing specifications and queries are also considered.

J. BISKUP and B. CONVENT. "A Formal View Integration Method." In C. ZANIOLO, ed., *Proc. ACM-SIGMOD International Conference*. Washington, D.C. 1986.

View integration is discussed in the framework of a formal model, which serves as a theoretical basis for a clear definition of several operations involved in the integration process (e.g., comparisons of information content, local schema transformations, and schema merging).

R. ELMASRI, J. LARSON, and S. B. NAVATHE. "Schema Integration Algorithms for Federated Databases and Logical Database Design." Honeywell Computer Sciences Center, Technical Report CSC-86-9:8212. 1986.

This paper is a comprehensive description of a methodology and a tool for n-ary integration based on the Entity-Category-Relationship model. A prototype has been implemented. It incorporates a variety of assertions among attributes and among entities to indicate correspondences.

M. MANNINO, S. B. NAVATHE, and W. EFFELSBERG. "A Rule-Based Approach for Merging Generalization Hierarchies." *Information Systems* 13, no. 3 (1988): 257–72.

This paper addresses a special problem: merging generalization hierarchies of entities from different views. A classification of all possible cases is presented, coupled with a set of rules for merging a pair of hierarchies in each case.

F. N. CIVELEK, A. DOGAC, and S. SPACCAPIETRA. "An Expert-System Approach to View Definition and Integration." In C. BATINI, ed., *Entity-Relationship Approach: A Bridge to the User: Proc. Seventh International Conference on Entity-Relationship Approach*. Rome. North-Holland, 1988.

This paper presents an expert system that assists the user in several activities involved in the integration process. The system provides support for several integration activities, a help facility for novice users, and an explanation of the results achieved by the system.

A. SAVASERE, A. SHETH, S. K. GALA, S. B. NAVATHE, and H. MARCUS, "On Applying Classification to Schema Integration." In *Proc. First International Workshop on Interoperability in Multi-Database Systems*. Kyoto, Japan. IEEE, April 1991.

This recent paper describes an approach to schema integration that combines an extended ER model as a front end with an internal formal model, CANDIDE, for schema integration. A variant of the ER model used in this book is used to input user views into the schema integration tool. These views are mapped into internal CANDIDE schemas. In CANDIDE, the concept of classification is used to place classes from a variety of schemas into a single acyclic graph of classes. The paper describes a set of operations used in conjunction with the classifier to achieve integration.