PART III

CONCEPTUAL DATA MODELING

CHAPTER 5

DATA MODELING BASICS

CHAPTER OBJECTIVES

- Understand what exactly a data model is
- Learn why a data model is necessary
- Observe how a data model represents information requirements
- Study how data design is derived from a data model
- Examine how data views are integrated into a data model

Data modeling is an integral part of designing a data system. It has gained importance as more and more businesses have moved into database systems. In a large database project, the data modelers are a distinct group of database practitioners. Professionals responsible for data modeling create models of the data that must reside in the organization's database. They are not concerned with the particular database management system. The data model is independent of the target DBMS. Third-party vendors offer good CASE tools to aid in the creation of data models.

The technique of modeling is nothing new. If a company is building an intricate product, it first creates a model of the product. The model is not the product itself; it is a representation or replica of the product. However, for the model to be a true representation it must display all the major features of the product itself. Building models is common in manufacturing.

Compare the building of a database system to the manufacture of a product. Something intricate is being built in both cases. The idea of building a data model is similar to the creation of a model for the product to be manufactured. The data model serves as a replica of the information content in the proposed database system.

Database Design and Development: An Essential Guide for IT Professionals by Paulraj Ponniah
ISBN 0-471-21877-4

WHAT IS A DATA MODEL?

Let us make the idea of a product model more specific and concrete. Assume that an automobile manufacturer wishes to manufacture the 2003 version of Splendid. The engineers begin with general concepts of the major components; they start with an overall notion of how the components must fit together. From these very general concepts, they put together a representation of how the automobile looks and possesses the required features; they create a 2003 model for Splendid. The model is refined and utilized to come up with the actual design of the automobile.

Relate this analogy to database development. Here we are building a database system. The requirements definition provides a collection of concepts about the information content; the definition gives indications about the data structures, their relationships, and constraints within the structures. The data model is a representation of the collection of these concepts about the information content. Just as the automobile model symbolizes the form and features of the actual automobile, the data model indicates the actual data structures and relationships of the data content.

A data model provides a method and a means for describing real-world information requirements in a form that can be designed and implemented as a computer database system. The method abstracts the characteristics of real-world information; the method enables us to capture the gist and meaning of the required data.

Take a particular example. Let us say that we want to create a data model for a university database. This model must represent the data structures necessary in that database. It should show the characteristics of the structures. The model has to indicate the data elements, relationships, and any constraints. In a university database, one data structure relates to the real-world information content about STUDENT. Even before defining any notation on how to show representations in a model, let us see how real-world information about STUDENT gets represented in the data model for STUDENT. Figure 5-1 shows the creation of this data model from real-world information content.

Why Create a Data Model?

We have noted that you create a data model based on the data requirements gathered and defined. In the requirements definition phase, you adopt various

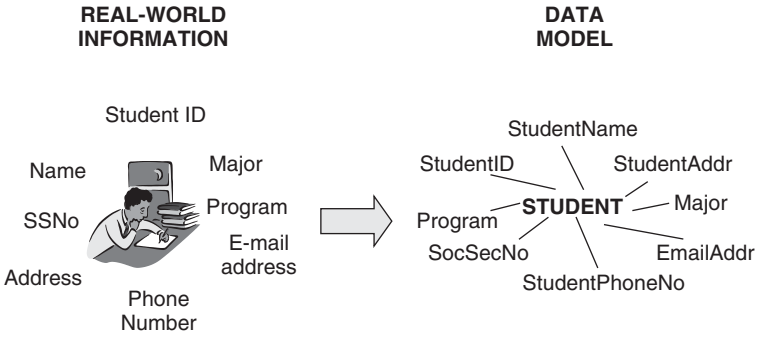

Figure 5-1 STUDENT: real-world information to data model.

methods to collect the information requirements to be implemented in the organization's database system. We have also indicated that these requirements drive the design of the database system. So where does the data model fit in? Why do you need to create a data model for the information requirements and to derive the design from the data model?

In the Chapter 4, we discussed the logical design of a database system. The logical design indicates the data structures and relationships; it shows how the various data elements hang together and are perceived. In a relational database system, these are perceived as two-dimensional tables. On the other hand, in a hierarchical database system, these are reckoned as data segments arranged in a hierarchical, top-down fashion. In a network database system, the arrangement is in the form of data nodes connected to form a network.

The logical design for a relational database system shows the data structures as tables, for a hierarchical database system as data segments, and for a network database system as data nodes. The logical design indicates the structures according to the conventional models—relational, hierarchical, or network. Now, see where the generic data model fits in. It is the preliminary step in the logical design activity. You create a generic data model that simply represents the real-world information requirements without regard to the conventional model of the final, target database. This step produces a replica of what real-world information must be found in the final database, not necessarily according to any conventional model.

Creation of a generic data model as the preliminary step offers several benefits. Here is a list of the major benefits:

- Provides independence so that any real-world information may be freely represented without any restrictions from conventional models
- Captures the essence of the data structures
- Represents real-world objects and their data characteristics
- Defines how real-world objects are related
- Plain and simple representation; easily understandable
- Is at a basic level so that creation and refinement are easy
- Is a very effective method for data abstraction

Real-World Information Requirements

The term "real-world information" has been used a few times, and you may be wondering what exactly it means. Let us clarify. Information or data about the external objects must be present in the database. The database is the internal representation of the data about the external objects. These objects exist in the outside world or the real world, whereas data about these objects exist inside the database. The database residing inside a computer system, therefore, is the internal manifestation of what exists outside in the real world.

Let us go back to the case of the university database. What does the internal university database represent? It indicates the information requirements about the objects present in the real world. What is the real world as far as the information requirements for the university is concerned? What are the objects in this part of

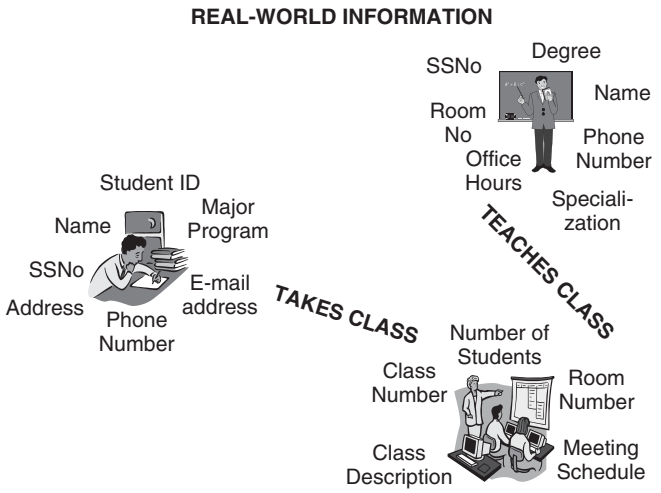

Figure 5-2 University database: real-world information requirements.

the real world that is of concern to the university? Figure 5-2 indicates a few of the objects in the real world for the university. Note the information requirements about these objects. These information requirements must be reflected in the generic data model.

How do you determine the real-world information requirements? The analysts on the database project for the university collect the requirements in the requirements definition phase. We covered that phase of DDLC in great detail. Recall how the information requirements are recorded in the requirements definition document as data groups. The collection of the data groups along with the descriptions of relationships and constraints forms the real-world information requirements for the university.

Data Model as the Replica

We have noted the meaning of real-world information requirements. The following are the requirements that must be satisfied by the proposed database. The database must be implemented to serve the users, providing them with information as required by them. The database must be designed so that it will contain data as required by the users. And the design may be derived from a generic data model that represents the information requirements.

The data model, therefore, is a replica or representation of the information requirements. Examine the information requirements defined in the definition phase of DDLC. Create a model—a data model—from the information requirements. That data model will then be a replica or representation of the information requirements. Figure 5-3 illustrates how a data model is created from the information requirements. Refer back to the Figure 5-2 and observe how the information requirements about objects in the university's real world are represented in the data model.

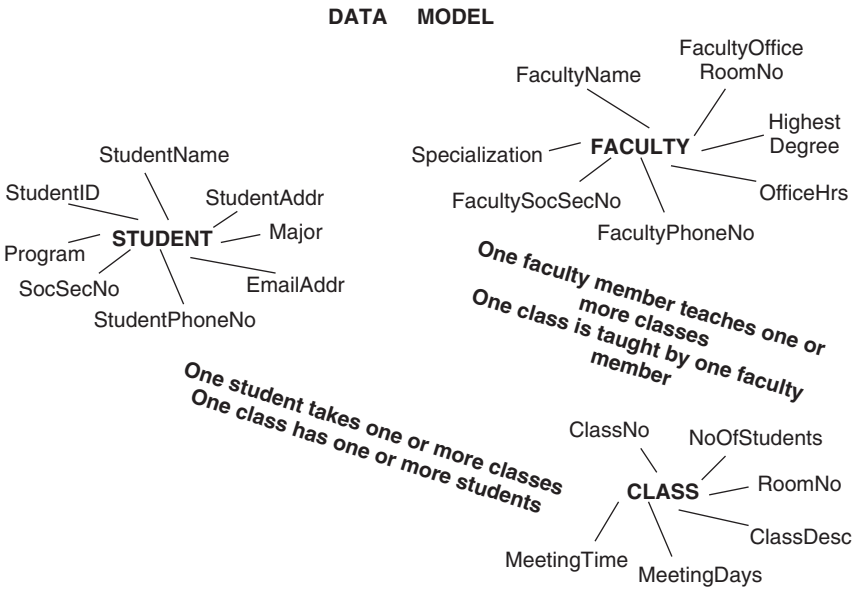

Figure 5-3 University database: creation of data model.

You would also have noticed that to create a data model you need distinct building blocks. Just as you need building blocks or parts to build a model for the manufacture of a product, you require building blocks to construct a data model. Later we will inspect different sets of notations for the building blocks commonly used in data modeling.

Data Model Components

Let us review the characteristics and functions of a data model. The data model is a replica or representation of the information requirements. It must truly reflect the real world of the organization as far as the data requirements are concerned. The data model should provide the meaning and gist of the information content in the proposed database. Whatever components are used to build the data model, when you observe and study the model, you must be able to understand clearly the nature and characteristics of the proposed database. You must be able to discern the types of data and the data structures from the data model.

How should the data model look, and how should it be put together? To match the proposed database correctly, we have to use appropriate building blocks to create the data model. Each building block has to serve a specific role in the overall purpose of representing the real-world information requirements. Also, the combination of all the building blocks and their arrangement must indicate clearly how elements of real-world information interact with one another. Finally, an appropriate symbol will be required for each building block of the data model, as well as a notation to show the relationships.

Based on this discussion, notations or symbols are needed to represent the following in the data model:

- Objects or things about which information must be stored in the database
- Notations for special object types
- Inherent characteristics or properties of each object
- Notations to classify characteristics
- Method to uniquely identify a particular instance of an object
- Indication of relationships among the objects
- The nature of each relationship
- In each relationship between two objects, indication as to how many of one object are related to how many of the other
- Notation to indicate special relationships
- Any constraints on the elements of data

Figure 5-4 presents the types of components or building blocks that are required to create a data model. Compare the components shown in the figure with the points mentioned above and examine whether these components make up a complete list to reflect real-world information requirements in a data model.

DATA MODELING CONCEPTS

By now, you have a fairly clear picture of what a data model is and why a data model becomes necessary as a preliminary step for the logical design. You understand what is meant by real-world information requirements and how the data model must truly reflect these requirements. You have noted the principle that a data model is a replica of the real-world information requirements. Also, you have realized that

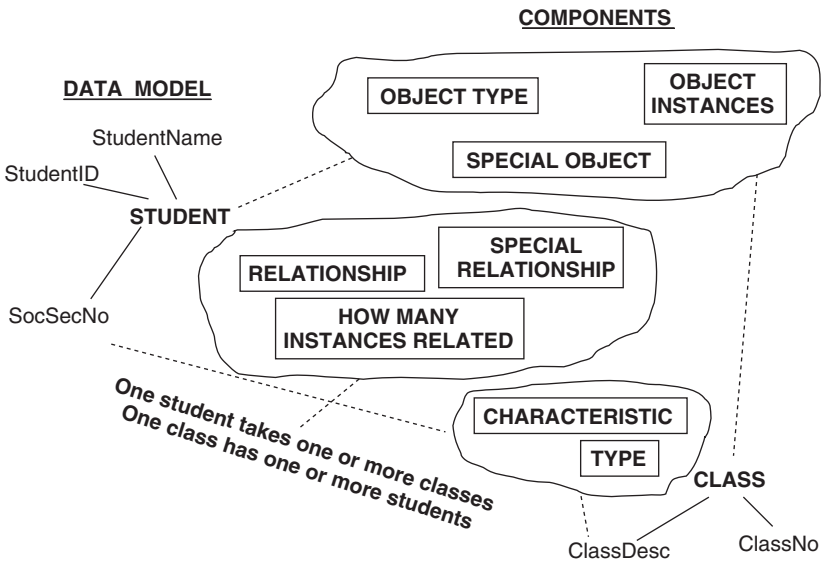

Figure 5-4 Data model components.

appropriate components or building blocks are necessary for the data model to be a true and correct replica.

We will briefly discuss some basic concepts of data modeling. When we consider the real-world information content, the first thing that strikes you is the vastness of data that exist. Even in the case of the real world relating to a university, the potential information content is enormous. You know that STUDENT is an object about which information must be stored. But how much information about the object called STUDENT must be stored? Everything there is about each student? Even after arriving at what information about STUDENT must be stored, how do you relate STUDENT and its characteristics to the data model components and represent them? After the data model is created, how do you proceed to the logical design? Let us examine these issues and find the answers.

The progression is from reality to data model and then from data model to design. Figure 5-5 illustrates the movement from reality to design.

Representation of Information Requirements

Let us work with a specific scenario, look at the information content, and examine the process of representing the information requirements in a data model. Take the case of an emergency room in a medical center and the services rendered there. Assume that you are asked to create a data model to represent the information requirements for billing for emergency room services.

Naturally, the first step is to observe the objects and processes in the emergency room and to collect information requirements for the stated purpose. The information requirements are for the purpose of supporting one process, namely, billing for emergency room services.

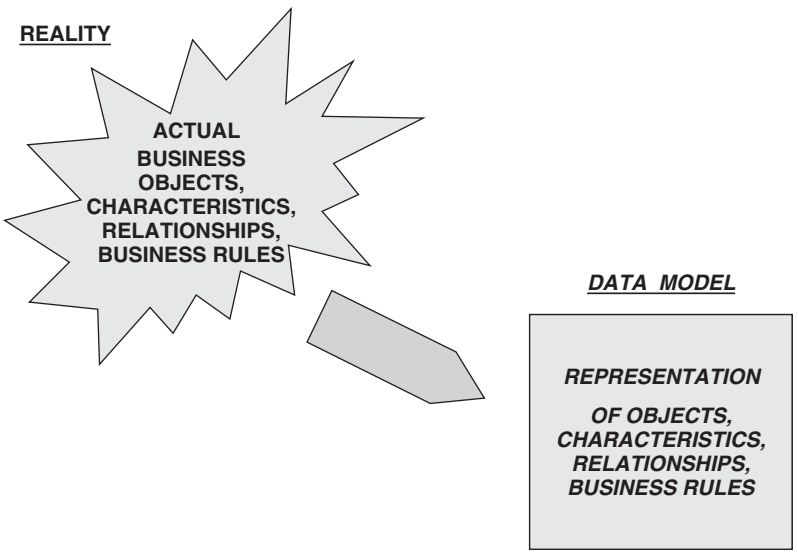

Figure 5-5 Reality to data model and design.

Start by observing the objects in the real world, in this case, the emergency room. The data model must represent the information requirements of this real-world situation. What do you see? What are the objects in the emergency room? In fact, there are many objects, and each object has a set of data relating to it. Begin with the patients in the emergency room waiting for their turn. There is a bunch of data about each patient. Then you see the nurses in attendance. Data elements about the object called NURSE may be enumerated. Next, you notice the physicians attending to patients in the booths behind the waiting area. The object called PHYSICIAN has a set of data relating to it. Then you look around and see the walls and ceiling and the overall lighting. What about the data about the walls and ceiling—the color of the walls and the height of the ceiling? You also hear the supposedly soothing music that comes over the PA system.

There is no need to push the observation farther. Two aspects of the information content of the real world become clear. When you observe the real world for information content, first you notice the existence of a large number of objects about which data are available. Second, you also realize that for each object there exists an abundance of data. Here is the predicament. What do you do with the abundance of data in the real-world situation? How do you represent the real-world information content in a data model?

Filtering Out Irrelevant Data

When you observe the real-world objects and the extent of data about these, how do you translate these into a data model? We mentioned that a data model is a replica or representation of real-world information requirements. A data model is a representation, but with one qualification. It is a representation only insofar as it is relevant to the purposes of the database.

Go back to the scenario of the emergency room. What is the stated purpose of the database in this case? Ability to support billing for emergency services. The types of uniforms worn by the nurses, the color of the walls in the waiting area, the height of the ceiling, the nature of the music heard, and the overall number of doctors in attendance—are these germane to the process of billing for emergency services? Not really. However, if the purpose is to assess the customer service in the emergency room, then the color of the walls, the state of cleanliness, and the nature of the music may all have a bearing.

What, therefore should a data model reflect? Not the entire information content of the real world, only those data elements that are relevant. In the emergency room situation, the data model for billing must exclude data elements such as the nurses' uniforms, wall colors, and ceiling heights. Remember, though, these could have been legitimate and relevant data if the purpose of the database had been different.

Figure 5-6 illustrates this filtering process for creating the data model for billing for emergency services. Note the data items that are excluded and observe how only those relevant to the billing process are included.

Usually, most of the filtering out of irrelevant data happens during the requirements definition phase itself. When you collect information requirements, you observe the primary business processes, review current applications, and examine

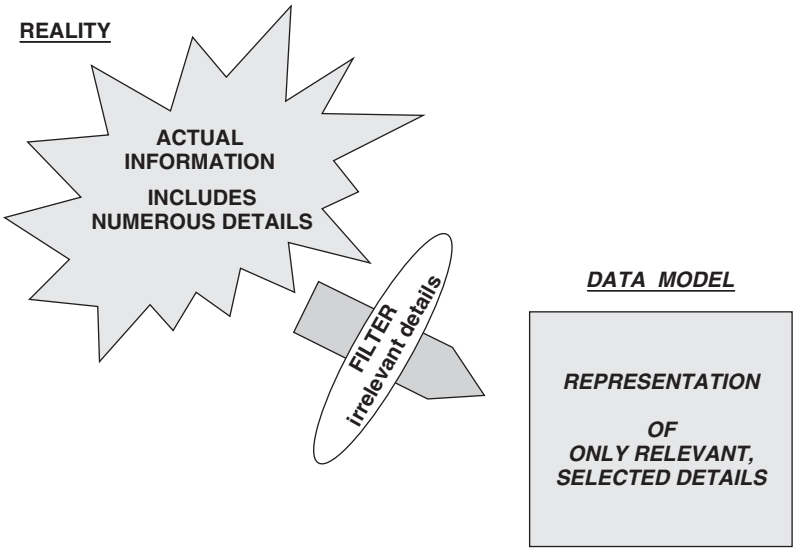

Figure 5-6 Filtering out of irrelevant data.

existing documents. While performing these tasks, automatically only the relevant data are indicated as information requirements.

Mapping of Components

Once you have filtered out the irrelevant information, how do you proceed to create the right data model? After you complete the filtering process, you are left with only the objects and only the relevant elements of data about each object to be included in the data model. You have noted that a data model is built with suitable building blocks or components. These components or building blocks must represent each and every element of the selected information requirements. So the next step for creating the correct data model is to map the information requirements to individual building blocks of the model.

Go back to the model to be built for emergency services billing. In this context, also review the data model components indicated in Figure 5-4. The information content for emergency services billing must be mapped to such components. Figure 5-7 shows how the mapping is done. Note each element of information requirements and the corresponding component of the data model.

Data Model to Data Design

Requirements definition, filtering out of irrelevant details, creating the data model—these are the steps up to the data model. The data model you have created with some standard building blocks is now ready for the next step in the development process. The data model truly represents the information content of the proposed database. How do you get to database implementation from the data model?

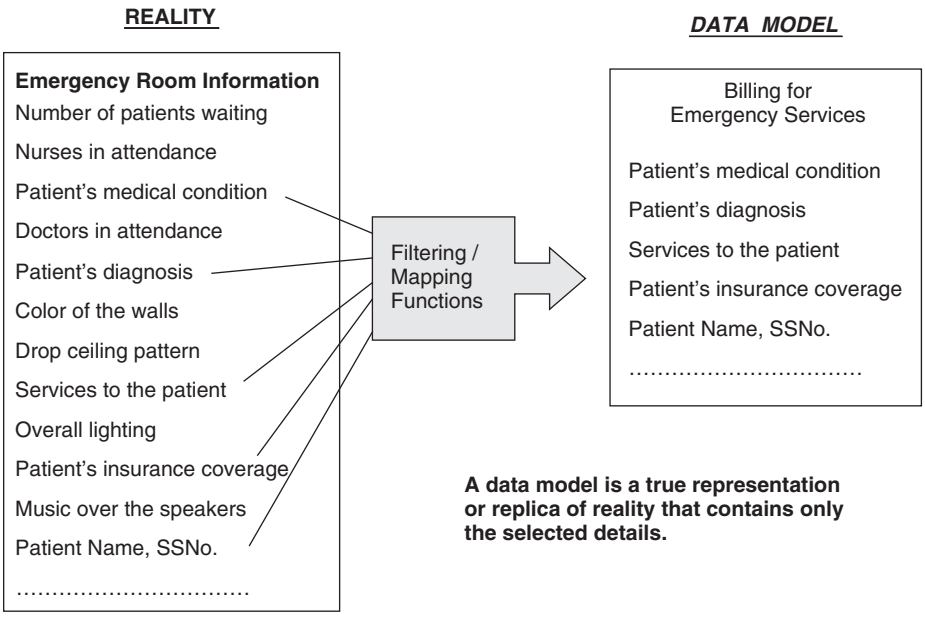

Figure 5-7 Mapping of data model components.

The next step is to formulate the logical design of the proposed database. As you know, the logical design consists of the conceptual view of the database. We referred to the output of the logical design step as the conceptual schema of the database. The conceptual schema provides a representation of the entire database, its structure, the relationships among the data structures, and any constraints. Well, how does the conceptual schema differ from the data model you have created up to this step? The essential difference lies in their basic nature. The data model created up to this step is a generic model built with building blocks independent of any conventional models. Remember that the conventional data models are the hierarchical, network, and relational models. In these models, the data structures and relationships are portrayed with certain defined arrangements—as hierarchical segments, network nodes, or two-dimensional tables.

Let us summarize the steps up to this point:

1. Requirements definition
2. Filtering out of irrelevant details
3. Creation of generic data model
4. Transformation into selected conventional model (hierarchical, network, relational, etc.)

Before we discuss the types of data models and describe their patterns, let us introduce the placement of these models in the database development life cycle. Figure 5-8 shows the phases in which the different types of models are created. Just note the different types of models indicated in the figure. The essential nature of each type is described in the next section.

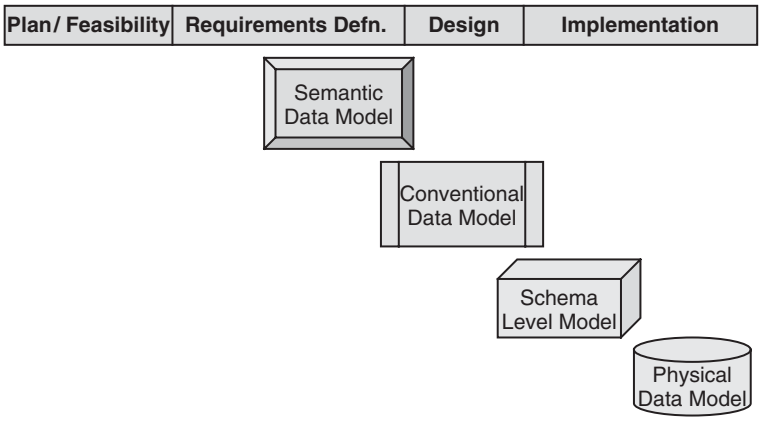

Figure 5-8 Data modeling in DDLC.

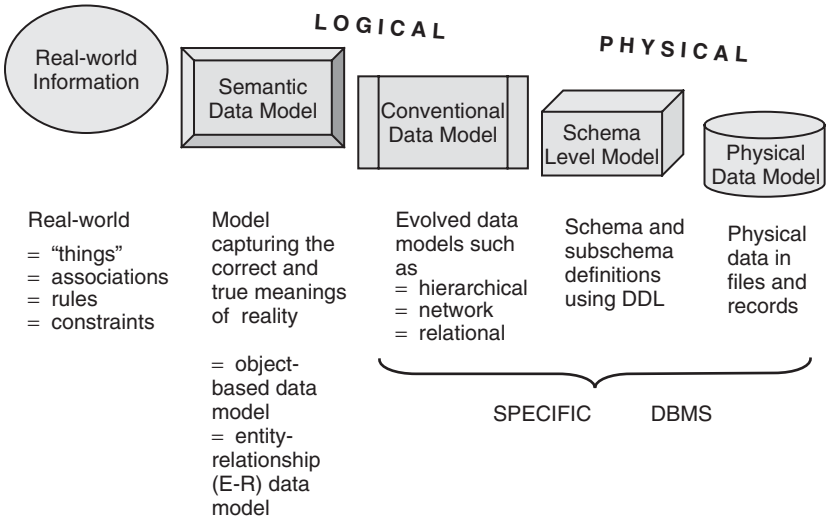

Figure 5-9 Data models at different levels.

PATTERNS OF DATA MODELING

At this point, two aspects of data modeling call for your attention. First, you realize the importance of data modeling in the database development process. You know the benefits and understand the reasons for creating a data model. You need to be clear about the purpose and the place of data modeling in the development life cycle. Second, you need to grasp the essential differences between the patterns of data models mentioned in our discussion.

Figure 5-9 distinguishes the placement of the types of data models at different levels.

Note the data models at the different levels from the semantic data model to the physical model. All of these are data models. Why? Remember that a data model is a representation of real-world information and not the actual real world itself. Each of the models indicated in the figure is a representation of real-world information at a particular level, but they are all representations; they are all data models. Let us describe the distinguishing features of each pattern. Pay special attention to understanding where each pattern fits in, why it is necessary, and how it contributes to the goal of implementing the proposed database.

High-Level Data Model

Let us begin at a level where none of the specifics of predefined arrangement of data structures comes into consideration. This is a level at which the data model is as generic as possible. In other words, once you have a data model at this high level, then making the transition to any type of conventional model is easy and straightforward. From a generic model, you can branch out to any of the conventional models, namely, hierarchical, network, or relational. That is the advantage of creating the generic, nonspecific model first.

Some database authors refer to the data model at the highest level as a semantic data model. A semantic model conveys the essential meaning of the data to be stored in the proposed database. This data model is created with commonly accepted building blocks or components. The following is a list of the essential features of the semantic data model:

- Contains relevant, selected details
- Correctly reflects the information requirements by excluding unnecessary details
- Truly conveys the meanings of all relevant "things" in reality
- Exactly represents the associations between "things" in real-world situations
- Correct data model of a real-world system
- Proper mapping of real-world information requirements
- Convenient means for communicating proposed information content to users

Object-Based Modeling Technique

Enough has been said about the semantic, generic data model. You have noted its nature, its characteristics, and its place in the database development cycle. You also understand that the semantic model is built with certain components. Now let us briefly turn our attention to the major techniques for building the semantic model. Two techniques are commonly used: the object-based modeling technique and the entity-relationship modeling technique. When you use either of the techniques, the result is a semantic data model of the proposed database. The techniques are similar and have many common features. They even share common notations or symbols to denote the building blocks or components. Chapters 6 and 7 cover the principles and components used in these two techniques. At this stage, we will just introduce the two techniques and broadly describe them.

First, let us consider the object-based modeling technique. In a real-world situation, every organization deals with business objects. You may even say that the organization's business processes are mere involvement with business objects. In the case of a university, the business objects include STUDENT, PROFESSOR, CLASS, COURSE, REGISTRATION, ASSIGNMENT, TEXTBOOK, and so on. For an airline business, the business objects include PASSENGER, AIRCRAFT, RESERVATION, ROUTE, MANUFACTURER, AIRPORT, AIRPLANE, FLIGHT, and so on.

Object-based modeling technique recognizes and builds on the fact that business processes involve such business objects. It rests on the premise that if your model reflects relevant information about business objects and how they interact with one another, then the model will truly represent the information requirements. Therefore, by adopting the object-based modeling technique, you produce a generic data model composed of the following:

- Business objects
- Characteristics of each object
- Interaction or linkage between pairs of objects
- Number of occurrences in linked pairs
- Method for identifying each unique occurrence of an object
- Special types of object sets
- Linkages within object sets

Entity-Relationship Modeling Technique

The entity-relationship modeling technique is similar to the object-based modeling technique and produces similar results. The main difference is that the two techniques were developed independently for the same purpose. Their evolutions take separate paths. In the object-based modeling technique, representation of information content is centered upon the concept of business objects. Here, in the entity-relationship (E-R) modeling technique, as the name of the technique explicitly indicates, the replica of information requirements is based on business entities and their relationships.

What are called entities in the E-R modeling technique are the objects in the object-based technique. However, a few subtle differences exist. The object-based technique recognizes special types of object sets called supersets and subsets. We will describe these in Chapter 6. The E-R modeling technique had to be enhanced later to include such entity sets. On the other hand, the E-R modeling technique includes concepts such as weak entities. The E-R model technique, more widely used, is built with well-defined and accepted graphics.

By adopting the entity-relationship modeling technique, you can create a generic data model consisting of the following:

- Business entities
- Properties of each entity type
- Relationship between pairs of entities

- Number of occurrences in related pairs
- Method for identifying each unique occurrence of an entity type
- Special entity types

Data Modeling Aids

In practice, how do you create a semantic or generic data model? You know that you need symbols to denote each component you want to use to build the model. The proper arrangement of appropriate symbols constitutes a data model diagram. A data model diagram gives you a graphic picture of the model. When you look at the data model diagram, you observe what the objects are, how they are linked together, and what the characteristics of each object are.

Graphic Tools. You may use any commercial graphic tool to draw the standard symbols and connecting lines to produce a data model diagram. This is still a useful method for creating data model diagrams, but it has limitations. It produces a static diagram; any revisions must be made by adding, changing, or deleting the symbols. Moreover, the diagram itself cannot be used directly to proceed to the next phases of the database development life cycle.

CASE Tools. Computer-aided software engineering (CASE) tools overcome many of the limitations of the plain graphic tools for drawing data model diagrams and creating semantic data models. CASE tools produce the data model diagrams and also keep the model components stored. Revisions may be made easily. Most CASE tools enable you to take the model to the next step. You can create the semantic model and transform it into a conventional data model and then into the physical model for the target commercial database system. This process is called forward engineering of the data model. Some CASE tools also have backward engineering capabilities, with which you can take a physical schema of an Oracle database and convert it back into a semantic model.

DATA VIEWS

In Chapter 4, while discussing the design phase of DDLC, we referred to the external schema. Recall that the external schema relates to the views of the database content by individual user groups. The external schema depicts the database as user groups perceive or view it from the outside—an external view.

What are these external perceptions of the database? What do the external perceptions hold? How do the users see the database? What is the relevance of these external views to data modeling? These external views are also referred to as *data views* or *user views*.

The concept of data views is critical in the development of a database system. It forms the underlying basis for creating a data model, which in turn leads us to the design. Later on, you will also learn about the role of data views in providing security to the database system. We will now explore the significance of data views and their role in data modeling and logical design. We will present different aspects of

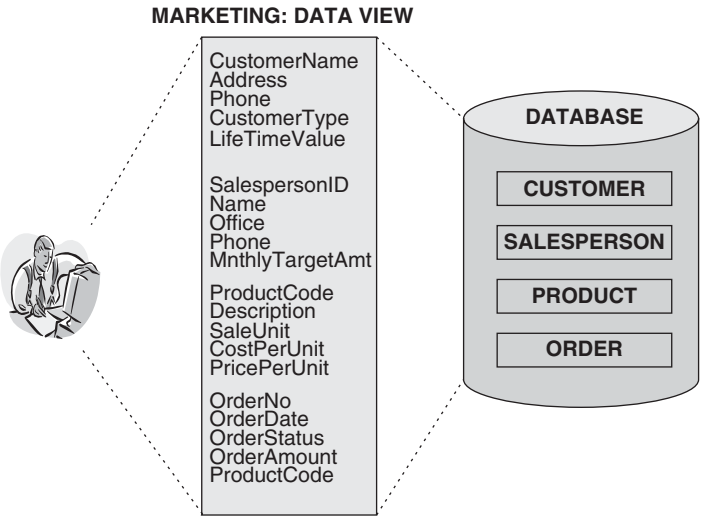

Figure 5-10 Data view: marketing department.

the concept of data views so that you may get a good grasp of its meaning and significance.

What Are Data Views?

Let us begin with a simple explanation. A data view represents one view of the database by a single user group. Figure 5-10 provides a simple explanation of data views. See how the users in the marketing department would view the database of a supermarket chain. The set of data in the database they are interested in constitutes their view of the database.

What we see here is that a data view is the description of a specific data set—a data set relevant to a single user group. As seen in the figure, the data set consists of those data elements the marketing department is interested in and needs to perform its business processes. Also, note that the specific data set for the marketing department does not just relate to data of a single business object. The set includes data elements from more than one business object.

Collection of Information Requirements

Recall how information requirements are gathered. The analysts review the processes, examine the existing documents, and interview user groups. The output from the requirements definition phase contains a collection of data sets or data groups. Each data set or data group supports a specific process.

Let us say that you are gathering information for the data system of a medical center. Take examples of two processes: providing services to patients in the emergency room and billing Medicare for services to patients. For each of these processes, you need data. Each process requires a set of data for its completion. Each data set forms a part of the overall information requirements of the medical

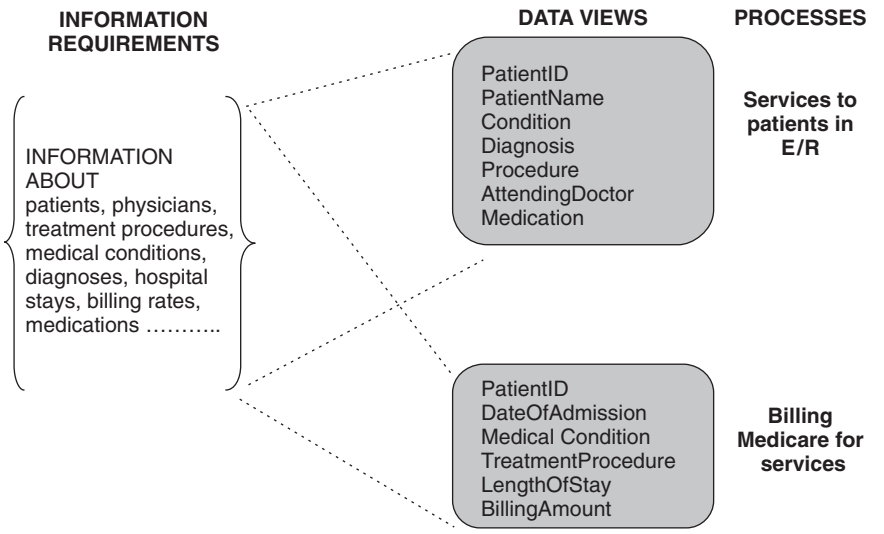

Figure 5-11 Data views as parts of information requirements.

center. A single data set constitutes one particular view of the entire collection of data. Figure 5-11 indicates how each data set forms a part of the information requirements.

Several processes, similar to the two described here, form the entire business of the medical center. You have seen that each process requires a data set or a data view. When you put all these separate views together, you arrive at the entire set of information requirements. Therefore, you may perceive the combination of all the data views as the collection of information requirements for an organization.

Windows into the Data System

Move on to another way of understanding data views. Imagine the database system to be structure like a building with many windows. If you look into the database system through a single window, you are likely to see a specific set of data elements. If you look through a second window, you will see another set of data elements. A third window will provide you with a view of yet another set of data elements. Just pause to note that some of the data elements observed through one window may overlap with a few of the elements seen through another window. A second significant note is that the windows do not represent the actual data; they are just means for looking at sections of the data itself.

Let us understand how this analogy relates to data views. Get back to our university database. This database supports the various processes performed at the university. As examples, take two processes such as course registration and assignment of final grades. Consider the computer screens and reports that enable these two processes to be performed. The screens and reports for each process contain a set of data elements from the university database. What the user is looking at from the screens and reports of a particular process is a set of data elements from the database. This is like looking into the database system through a specific window. The view of data for the process is the view of data through a specific window.

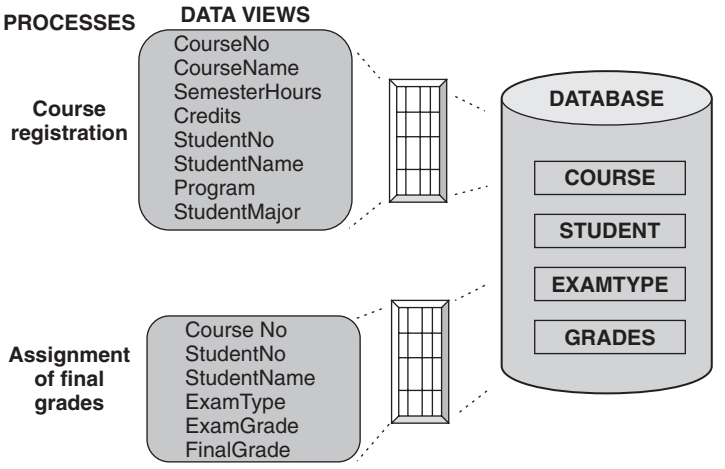

Figure 5-12 Data views as windows into database.

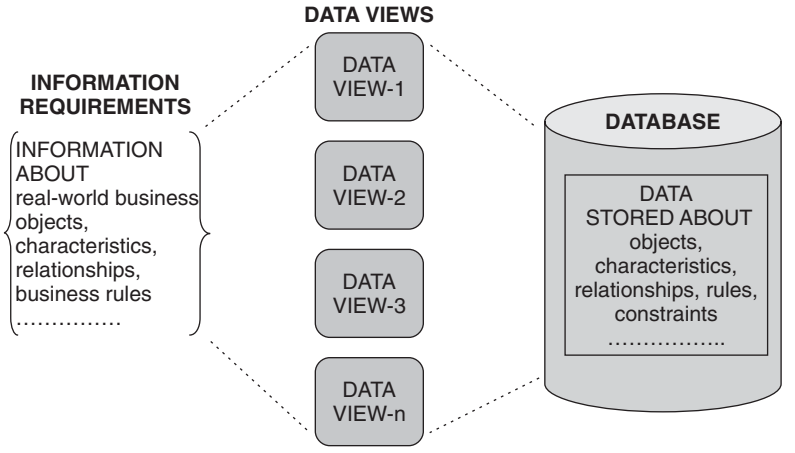

Figure 5-13 Data views: two perceptions.

Figure 5-12 illustrates this comparison of data views with windows into the university database system. Notice the data elements for the two processes seen through the two windows.

Data Views: Two Perceptions

You have now perceived data views from two sides of the entire database development process. Let us summarize our discussion of data views so far using two approaches. On one side, you have seen the real-world information requirements as a collection of individual data views. From the other side, you have perceived data views as windows into the completed database system. Figure 5-13 demonstrates the two perceptions of data views from both sides.

What you understand now is that data views relate to the real-world information on one side and to the completed database system on the other side. The objective of a database development project is the implementation of a database system that matches with the information requirements. Now you see the connection and role of data views in the development process.

VIEW INTEGRATION

Data views offer a useful link from real-world information requirements to the finished database system. You can assemble the data views from the side of real-world information, consolidate them, and derive the proposed database system. How are these data views assembled? What is the process? The combining of the data views results in a consolidated data model for the proposed database system. Let us now discuss this topic.

We have established the connection between data views and data model. The data model is derived from the data views. Go back to the real-world information. This, as we have seen, is a collection of individual user views. You get the resultant data model by combining the separate user views. How can you accomplish this and produce the entire data model? Analysts adopt one of two distinct approaches depending on the circumstances and skills of the data modelers. Here are the two approaches or methods:

Merge individual user views. You take all the individual user views, merge these user views, and then create the complete data model from the complete set of merged data views.

Integrate partial data models. You create a partial data model with standard building blocks for each user view and then integrate these partial data models into the complete data model.

Figure 5-14 illustrates the two approaches showing three user views. The principle can be extended to any number of data views.

Merging Individual User Views

This approach for creating the consolidated data model by merging individual user views generally consists of the following major tasks. Amend and adapt this list to suit your database project.

1. Take each user view and group the data elements by business objects.
2. Within each user view, identify the relationships among the data groups.
3. Take three or four user views at a time and combine the data groups eliminating duplicate elements.
4. For each such combination, mark the relationships among the data groups.
5. Repeat tasks 3 and 4 until all user views are combined.
6. Separate out objects and their characteristics.
7. Establish relationships among objects.

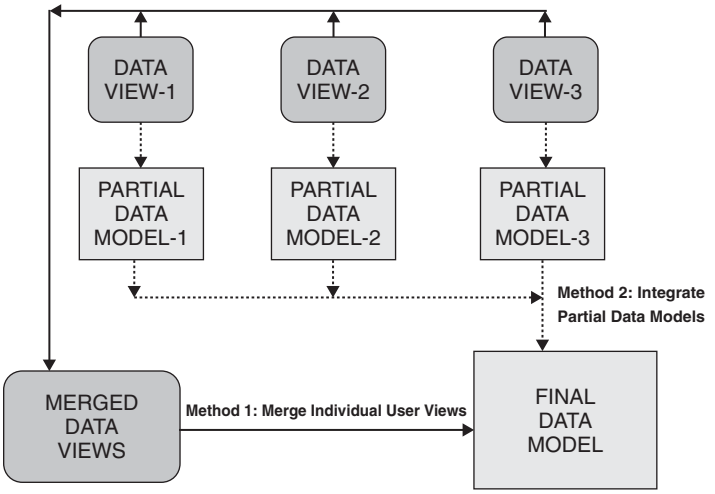

Figure 5-14 View integration: two approaches.

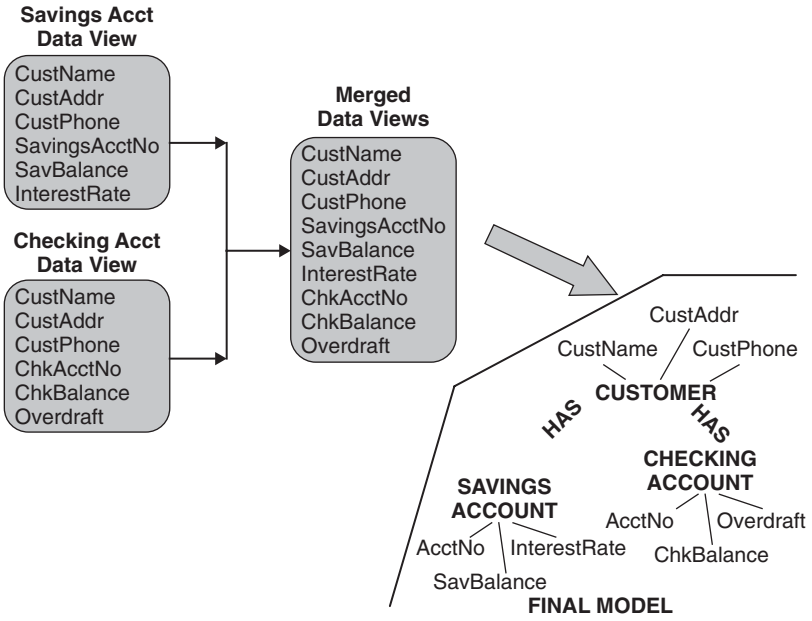

Figure 5-15 Merging individual user views.

8. Indicate how an instance of each object will be uniquely identified.
9. Create complete or global data model from the complete set of objects and relationships.

Figure 5-15 describes the approach of merging individual user views for a banking example. Notice how each task is indicated and how the tasks work together to produce the final, global data model.

Integrating Partial Data Models

In this approach, you do not wait to apply data modeling techniques until all the requirements are merged. You create a mini-model or partial data model from a few related user views. How do you form these sets of related user views? Usually, user views of a department or a natural user group can be combined to form a related group. This approach for creating the consolidated data model by integrating partial data models generally consists of the following major tasks. Amend and adapt this list to suit your database project.

1. Combine the data elements of related user views (usually those of a department or user group).
2. Eliminate duplicate data elements.
3. Separate out objects and their characteristics for the combined set of elements.
4. Establish relationships among objects.
5. Create a local or mini data model for the objects and relationships for the first combination.
6. Repeat tasks 1 through 5 until local data models are created for all combinations.
7. Integrate all the local models, two at a time.
8. Indicate how an instance of each object will be uniquely identified.
9. Review the final, global data model created from the partial model for completeness.

Figure 5-16 describes the approach of integrating partial or local data models for the banking example. Observe how each task is shown in the figure and how the tasks work together to produce the final data model.

Enhancement and Validation

As soon as you produce the global or complete data model with either of the two methods, your data model is almost ready to complete the logical design. Before you use the data model, you have to review the data model to ensure completeness. This is the time to enhance the data model by removing discrepancies and to validate the model.

Generally, enhancement of the data model includes the following tasks:

1. **Reconcile naming conflicts**. If two or more different names are used to refer to the same data element, then these names are synonyms. Review the data model and reconcile synonyms. If the same name is used to refer to two or more data elements, then the name is a homonym. Reconcile homonyms.
2. **Resolve conflicts between data element types**. If a data element in the model refers to an object in one place and to a characteristic of another object in a different place, then there is conflict between the two types of data elements. Resolve such discrepancies.

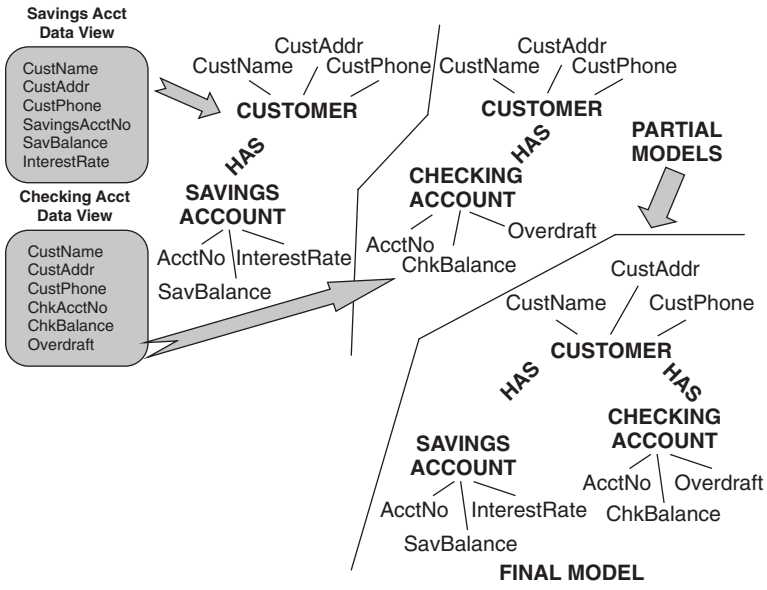

Figure 5-16 Integrating partial data models.

3. **Ensure correct value sets for each characteristic**. If a characteristic of an object has a defined set of values in one user view and a different set of values in another user view, this characteristic cannot be taken as the same characteristic for the object. If the characteristic *employee height* is given in inches in one user view and in centimeters in another user view, then there is a discrepancy if the two user views are merged without reconciling the units of measurement. Review the data model and ensure that the value sets for each characteristic are correct.
4. **Resolve discrepancies in the constraints**. Suppose that, in one user view, the relationship between employee object and department object is such that one employee may belong to only one department. On the other hand, another user view such as project assignments may allow one employee to be involved in multiple departments, that is, assigned to projects of multiple departments. This is a discrepancy in the constraint on the relationship. Resolve such discrepancies.

One more task remains before the global model is finally complete and ready. This is the task of validating the data model with the users. Usually, user representatives knowledgeable in the organization's business processes are selected to perform the review of the data model. The project team walks the user representatives through the global data model, reviewing each object, each relationship, and each type of constraint.

Consolidated Data Model

After enhancement and validation, the global data model must be reviewed one last time to check for any data redundancies. After verification and modification to set

right any data redundancies, the data model is ready for completing the logical design.

Let us summarize the steps and tasks from requirements definition to preparing the consolidated data model.

- Filter out unnecessary details; collect the selected data requirements.
- Adopt one of the following methods to create the global data model:
  - Merge user views
  - Integrate partial data models
- Enhance the global model.
- Validate the global model.
- Review the global model to ensure that redundancies have been removed.
- Create the final consolidated data model.

Figure 5-17 summarizes these steps. Note the flow of the tasks from one step to the next.

CHAPTER SUMMARY

- A data model is the representation or replica of real-world information.
- A data model is built with defined and accepted building blocks or components.
- Before a data model is built from information requirements, irrelevant details must be filtered out.
- A generic or semantic data model captures the true meaning of real-world information.

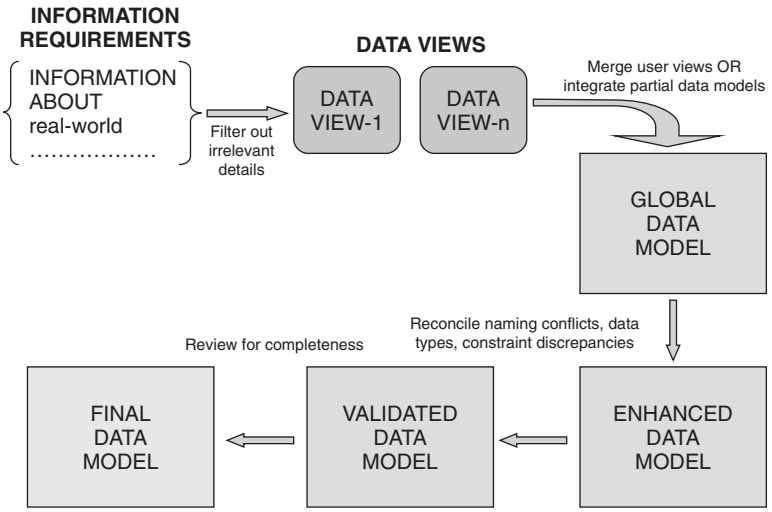

Figure 5-17 Consolidated data model: steps for creation.

- There are two techniques for creating a generic or semantic data model: (1) the object-based modeling technique and (2) the entity-relationship modeling technique.
- Data views form the underlying basis for creating a data model; data views can be perceived as a collection of information requirements and as windows into the database system.
- There are two methods for data view integration: (1) merging individual user views and (2) integrating partial data models.

REVIEW QUESTIONS

1. Give a simple definition of a data model. Explain the connection between real-world information and the data model.
2. State any four reasons for creating a data model during DDLC. Is creating a data model absolutely essential?
3. What do you understand by "real-world information"? What would real-world information be for a banking organization?
4. Data model components need notations or symbols to represent them. List any such five data model components.
5. What is filtering out of irrelevant data? How is this important in data modeling?
6. What is your understanding of a generic, semantic data model? How does it differ from a conventional data model as specified in the text?
7. Name the two common modeling techniques for creating a semantic data model. What are the differences between the two techniques?
8. How do CASE tools aid data modeling?
9. Explain briefly the role of data views in data modeling.
10. What is view integration? Briefly describe any one method for view integration.

EXERCISES

1. Match the columns:

1. real-world information	A. technique for creating semantic data model
2. semantic data model	B. reconcile naming conflicts
3. special object type	C. window into data system
4. E-R modeling	D. expresses true meaning of data
5. CASE tools	E. integrated from partial data models
6. data view	F. based on data models
7. merging user views	G. represented by data model

8. data model enhancement — H. allow forward engineering of data model
9. logical database design — I. technique to create partial data models
10. consolidated data model — J. a data model component

2. As the project manager, write a memo to the Executive VP of information technology explaining the significance of data modeling and justifying the purchase of a CASE tool.
3. As the senior data modeler in the database project for an insurance company, describe the business objects you would consider in your data model.
4. Describe with examples from a banking business how data views represent both a collection of information requirements and windows into the proposed data system.
5. You are a senior analyst responsible for view integration in the database project of a department store. Which method will you choose? Describe how you will proceed to produce the consolidated data model.