Database Design - 2nd Edition

Adrienne Watt

Nelson Eng

Chapter 4 Types of Data Models

Adrienne Watt & Nelson Eng

High-level Conceptual Data Models

High-level conceptual data models provide concepts for presenting data in ways that are close to the way people perceive data. A typical example is the entity relationship model, which uses main concepts like entities, attributes and relationships. An entity represents a real-world object such as an employee or a project. The entity has attributes that represent properties such as an employee's name, address and birthdate. A relationship represents an association among entities; for example, an employee works on many projects. A relationship exists between the employee and each project.

Record-based Logical Data Models

Record-based logical data models provide concepts users can understand but are not too far from the way data is stored in the computer. Three well-known data models of this type are relational data models, network data models and hierarchical data models.

- The *relational model* represents data as*relations*, or tables. For example, in the membership system at Science World, each membership has many members (see Figure 2.2 in Chapter 2). The membership identifier, expiry date and address information are fields in the membership. The members are individuals such as Mickey, Minnie, Mighty, Door, Tom, King, Man and Moose. Each record is said to be an *instance* of the membership table.
- The *network model* represents data as record types. This model also represents a limited type of one to many relationship called a *set type*, as shown in Figure 4.1.

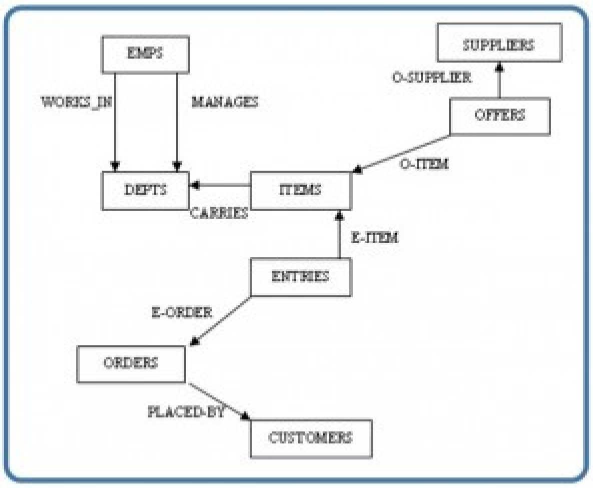

Figure 4.1. Network model diagram.

- The *hierarchical model* represents data as a hierarchical tree structure. Each branch of the hierarchy represents a number of related records. Figure 4.2 shows this schema in hierarchical model notation.

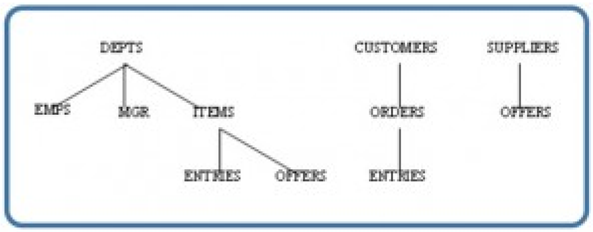

Figure 4.2. Hierarchical model diagram.

Key Terms

hierarchical model: represents data as a hierarchical tree structure

instance: a record within a table

network model: represents data as record types

relation: another term for table

relational model: represents data as relations or tables

set type: a limited type of one to many relationship

Exercises

1. What is a data model?
2. What is a high-level conceptual data model?
3. What is an entity? An attribute? A relationship?
4. List and briefly describe the common record-based logical data models.

Attribution

The following material was written by Adrienne Watt:

1. Key Terms
2. Exercises

Chapter 5 Data Modelling

Adrienne Watt

Data modelling is the first step in the process of database design. This step is sometimes considered to be a high-level and abstract design phase, also referred to as conceptual design. The aim of this phase is to describe:

- The data contained in the database (e.g., entities: students, lecturers, courses, subjects)
- The relationships between data items (e.g., students are supervised by lecturers; lecturers teach courses)
- The constraints on data (e.g., student number has exactly eight digits; a subject has four or six units of credit only)

In the second step, the data items, the relationships and the constraints are all expressed using the concepts provided by the high-level data model. Because these concmepts do not include the implementation details, the result of the data modelling process is a (semi) formal representation of the database structure. This result is quite easy to understand so it is used as reference to make sure that all the user's requirements are met.

The third step is database design. During this step, we might have two sub-steps: one called *database logical design*, which defines a database in a data model of a specific DBMS, and another called *database physical design*, which defines the internal database storage structure, file organization or indexing techniques. These two sub-steps are database implementation and operations/user interfaces building steps.

In the database design phases, data are represented using a certain data model. The *data model* is a collection of concepts or notations for describing data, data relationships, data semantics and data constraints. Most data models also include a set of basic operations for manipulating data in the database.

Degrees of Data Abstraction

In this section we will look at the database design process in terms of specificity. Just as any design starts at a high level and proceeds to an ever-increasing level of detail, so does database design. For example, when building a home, you start with how many bedrooms and bathrooms the home will have, whether it will be on one level or multiple levels, etc. The next step is to get an architect to design the home from a more structured perspective. This level gets more detailed with respect to actual room sizes, how the home will be wired, where the plumbing fixtures will be placed, etc. The last step is to hire a contractor to build the home. That's looking at the design from a high level of abstraction to an increasing level of detail.

The database design is very much like that. It starts with users identifying the business rules; then the database designers and analysts create the database design; and then the database administrator implements the design using a DBMS.

The following subsections summarize the models in order of decreasing level of abstraction.

External models

- Represent the user's view of the database
- Contain multiple different external views
- Are closely related to the real world as perceived by each user

Conceptual models

- Provide flexible data-structuring capabilities
- Present a “community view”: the logical structure of the entire database
- Contain data stored in the database
- Show relationships among data including:
  - Constraints
  - Semantic information (e.g., business rules)
  - Security and integrity information
- Consider a database as a collection of entities (objects) of various kinds
- Are the basis for identification and high-level description of main data objects; they avoid details
- Are database independent regardless of the database you will be using

Internal models

The three best-known models of this kind are the relational data model, the network data model and the hierarchical data model. These internal models:

- Consider a database as a collection of fixed-size records
- Are closer to the physical level or file structure
- Are a representation of the database as seen by the DBMS.
- Require the designer to match the conceptual model’s characteristics and constraints to those of the selected implementation model
- Involve mapping the entities in the conceptual model to the tables in the relational model

Physical models

- Are the physical representation of the database
- Have the lowest level of abstractions
- Are how the data is stored; they deal with
  - Run-time performance
  - Storage utilization and compression
  - File organization and access methods
  - Data encryption
- Are the physical level – managed by the *operating system (OS)*
- Provide concepts that describe the details of how data are stored in the computer’s memory

Data Abstraction Layer

In a pictorial view, you can see how the different models work together. Let’s look at this from the highest level, the external model.

The external model is the end user’s view of the data. Typically a database is an enterprise system that serves the needs of multiple departments. However, one department is not interested in seeing other departments’ data (e.g., the human resources (HR) department does not care to view the sales department's data). Therefore, one user view will differ from another.

The external model requires that the designer subdivide a set of requirements and constraints into functional modules that can be examined within the framework of their external models (e.g., human resources versus sales).

As a data designer, you need to understand all the data so that you can build an enterprise-wide database. Based on the needs of various departments, the conceptual model is the first model created.

At this stage, the conceptual model is independent of both software and hardware. It does not depend on the DBMS software used to implement the model. It does not depend on the hardware used in the implementation of the model. Changes in either hardware or DBMS software have no effect on the database design at the conceptual level.

Once a DBMS is selected, you can then implement it. This is the internal model. Here you create all the tables, constraints, keys, rules, etc. This is often referred to as the *logical design*.

The physical model is simply the way the data is stored on disk. Each database vendor has its own way of storing the data.

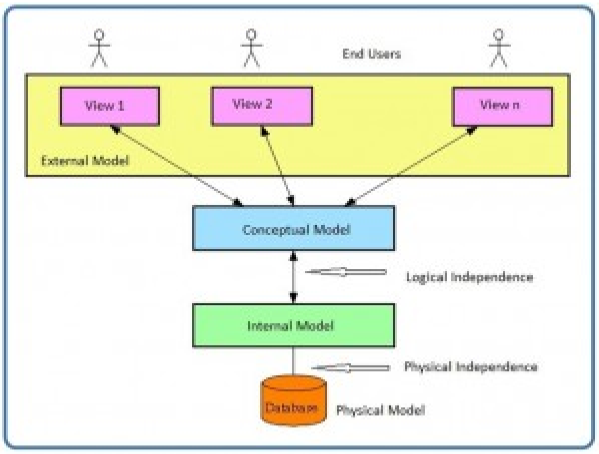

Figure 5.1. Data abstraction layers.

Schemas

A *schema* is an overall description of a database, and it is usually represented by the *entity relationship diagram (ERD)*. There are many subschemas that represent external models and thus display external views of the data. Below is a list of items to consider during the design process of a database.

- External schemas: there are multiple
- Multiple subschemas: these display multiple external views of the data
- Conceptual schema: there is only one. This schema includes data items, relationships and constraints, all represented in an ERD.
- Physical schema: there is only one

Logical and Physical Data Independence

Data independence refers to the immunity of user applications to changes made in the definition and organization of data. Data abstractions expose only those items that are important or pertinent to the user. Complexity is hidden from the database user.

Data independence and operation independence together form the feature of data abstraction. There are two types of data independence: logical and physical.

Logical data independence

A *logical schema* is a conceptual design of the database done on paper or a whiteboard, much like architectural drawings for a house. The ability to change the logical schema, without changing the *external schema* or user view, is called *logical data independence*. For example, the addition or removal of new entities, attributes or relationships to this *conceptual schema* should be possible without having to change existing external schemas or rewrite existing application programs.

In other words, changes to the logical schema (e.g., alterations to the structure of the database like adding a column or other tables) should not affect the function of the application (external views).

Physical data independence

Physical data independence refers to the immunity of the internal model to changes in the physical model. The logical schema stays unchanged even though changes are made to file organization or storage structures, storage devices or indexing strategy.

Physical data independence deals with hiding the details of the storage structure from user applications. The applications should not be involved with these issues, since there is no difference in the operation carried out against the data.

Key Terms

conceptual model: the logical structure of the entire database

conceptual schema: another term for logical schema

data independence: the immunity of user applications to changes made in the definition and organization of data

data model: a collection of concepts or notations for describing data, data relationships, data semantics and data constraints

data modelling: the first step in the process of database design

database logical design: defines a database in a data model of a specific database management system

database physical design*:* defines the internal database storage structure, file organization or indexing techniques

entity relationship diagram (ERD): a data model describing the database showing tables, attributes and relationships

external model: represents the user's view of the database

external schema: user view

internal model: a representation of the database as seen by the DBMS

logical data independence: the ability to change the logical schema without changing the external schema

logical design: where you create all the tables, constraints, keys, rules, etc.

logical schema: a conceptual design of the database done on paper or a whiteboard, much like architectural drawings for a house

operating system (OS): manages the physical level of the physical model

physical data independence: the immunity of the internal model to changes in the physical model

physical model: the physical representation of the database

schema: an overall description of a database

Exercises

1. Describe the purpose of a conceptual design.
2. How is a conceptual design different from a logical design?
3. What is an external model?
4. What is a conceptual model?
5. What is an internal model?
6. What is a physical model?
7. Which model does the database administrator work with?
8. Which model does the end user work with?
9. What is logical data independence?
10. What is physical data independence?

Also see *Appendix A: University Registration Data Model Example*

Attribution

The following material was written by Adrienne Watt:

- Some or all of the introduction, degrees of data abstraction, data abstraction layer
- Key Terms
- Exercises