**Lesson 11**

**Objectives**

* Data Models
  + Structural Part
  + Manipulative Part
  + Set of Integrity Rules
* Categories of Data Models
  + Logical
    - Object Based
    - Record Based
  + Physical

**Database Languages**

We mentioned earlier that a schema is written using a data definition language. In fact, it is written in the data definition language of a particular DBMS. Unfortunately, this type of language is too low level to describe the data requirements of an organization in a way that is readily understandable by a variety of users. What we require is a higher-level description of the schema: that is, a **data model**.

**Data model**

An integrated collection of concepts for describing and manipulating data, relationships between data, and constraints on the data in an organization.

A model is a representation of ‘real world’ objects and events, and their associations. It is an abstraction that concentrates on the essential, inherent aspects of an organization and ignores the accidental properties. A data model represents the organization itself. It should provide the basic concepts and notations that will allow database designers and end-users unambiguously and accurately to communicate their understanding of the organizational data.

A data model can be thought of as comprising three components:

S**tructural part**, consisting of a set of rules according to which databases can be constructed.

M**anipulative part**, defining the types of operation that are allowed on the data (this includes the operations that are used for updating or retrieving data from the database and for changing the structure of the database).

S**et of integrity constraints**, which ensures that the data is accurate.

The purpose of a data model is to represent data and to make the data understandable. If it does this, then it can be easily used to design a database.

There have been many data models proposed in the literature. They fall into three broad categories: **object-based**, **record-based**, and **physical** data models. The first two are used to describe data at the conceptual and external levels; the latter is used to describe data at the internal level.

**Object-Based Data Models**

Object-based data models use concepts such as entities, attributes, and relationships.

An **entity** is a distinct object (a person, place, thing, concept, event) in the organization that is to be represented in the database. An **attribute** is a property that describes some aspect of the object that we wish to record, and a **relationship** is an association between entities.

Some of the more common types of object-based data model are:

* Entity–Relationship
* Semantic
* Functional
* Object-Oriented.

The Entity–Relationship model has emerged as one of the main techniques for database design and forms the basis for the database design methodology used in this book. The object-oriented data model extends the definition of an entity to include not only the attributes that describe the **state** of the object but also the actions that are associated with the object, that is, its **behavior**. The object is said to **encapsulate** both state and behavior. We look at the Entity–Relationship model in depth in this course.

**Record-Based Data Models**

In a record-based model, the database consists of a number of fixed-format records possibly of differing types. Each record type defines a fixed number of fields, each typically of a fixed length. There are three principal types of record-based logical data model: the **relational data model**, the **network data model**, and the **hierarchical data model**. The hierarchical and network data models were developed almost a decade before the relational data model, so their links to traditional file processing concepts are more evident.

**Relational data model**

The relational data model is based on the concept of mathematical relations. In the relational model, data and relationships are represented as tables, each of which has a number of columns with a unique name. A simple example that based on relational model is in Figure 11.1.

Relational data model will be discussed in detail latterly.

Note that the relational data model requires only that the database be perceived by the user as tables. However, this perception applies only to the logical structure of the database, that is, the external and conceptual levels of the ANSI-SPARC architecture. It does not apply to the physical structure of the database, which can be implemented using a variety of storage structures.

**Network data model**

In the network model, data is represented as collections of **records**, and relationships are represented by **sets**. Compared with the relational model, relationships are explicitly modeled by the sets, which become pointers in the implementation. The records are organized as generalized graph structures with records appearing as **nodes** (also called **segments**) and sets as **edges** in the graph.

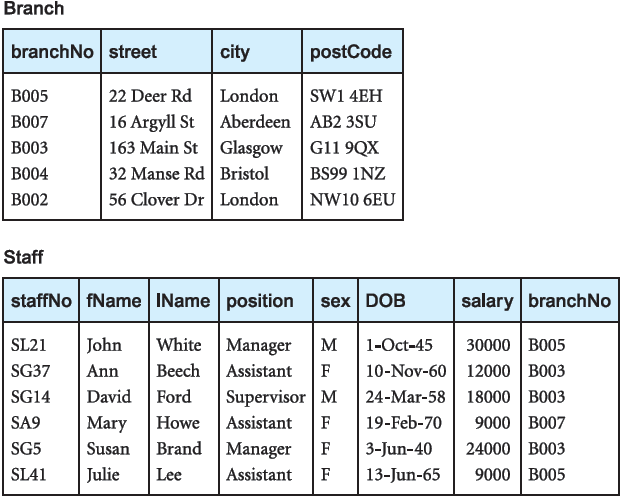


Figure 11.1

Figure 11.2 illustrates an instance of a network schema for the same data set presented in Figure 11.1.

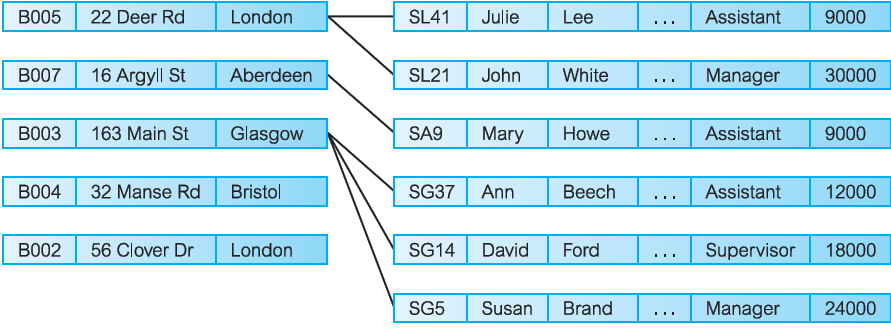


Figure 11.2

**Hierarchical data model**

The hierarchical model is a restricted type of network model. Again, data is represented as collections of **records** and relationships are represented by **sets**.

However, the hierarchical model allows a node to have only one parent. A hierarchical model can be represented as a tree graph, with records appearing as **nodes** (also called **segments**) and sets as **edges**. Figure 11.3 illustrates an instance of a hierarchical schema for the same data presented in Figure 11.2 and 11.3.

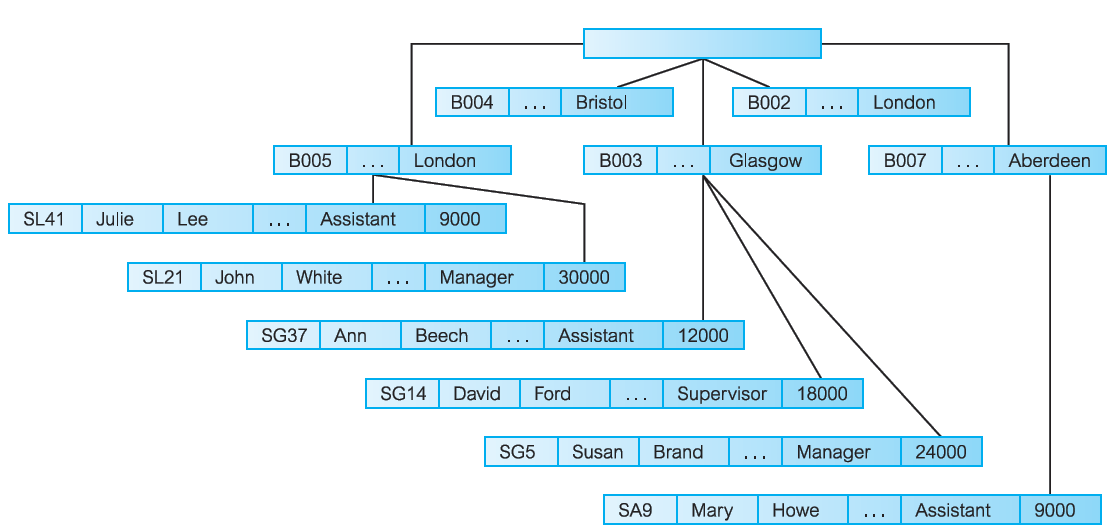


Figure 11.3

**Physical Data Models**

Physical data models describe how data is stored in the computer, representing information such as record structures, record orderings, and access paths. There are not as many physical data models as logical data models, the most common ones being the *unifying model* and the *frame memory*.