The designers and users of present-day information systems deal with more and more complex applications that have to meet stringent quality requirements. It is no longer enough to capture the static aspect of the world – modeling the dynamics, i.e., time, change, and concurrency becomes equally important. Also, there is a need for multiple modalities to distinguish between what is true, known, believed, permitted, obligatory, past, present, and future.

Stimulated by the new requirements, the database technology has been evolving as well. New data models, integrity constraints, deductive and active databases, database programming languages are providing the necessary extensions of the traditional database framework. However, the new technology has brought about a host of novel features whose interaction is still not well understood. For example, it is difficult to make sense of any sizable set of active database rules.

Logic, in the form of many different logical formalisms, is a suitable tool to address at least some of the above problems. Logic has simple, unambiguous syntax and semantics. It is thus ideally suited to the task of specifying informa-
tion systems. Logic typically comes with an associated proof procedure which makes it possible to check the consistency of specifications and determine their consequences. In databases, since the pioneering work of Codd logic has been recognized as a basic kind of query language. Although hard to use for a novice, logic serves as a reference point for the design of practical query languages. They would do well by emulating the simplicity of its syntax and semantics! Also, logic is the most natural language to formulate integrity constraints – conditions that database states have to satisfy. Since another pioneering work of Kowalski, van Emde, and Colmerauer, syntactically restricted logic (Horn clauses) has also been studied and used as a programming language leading to the concepts of logic programming and deductive databases.

The purpose of this book is to bear witness to the different uses of logic in databases and information systems. We have organized it as a set of fairly broad case studies. Each study, written by one or more experts in the field, addresses a specific aspect of databases or information systems, or a specific logical formalism.

Chapter 2 is a brief introduction to the basic concepts of first-order logic, modal logic, and logic programming. This chapter can be used as an entry point to the book, although its primary purpose is to gather in one place various basic definitions used in later chapters and provide an easy-to-use reference for the reader.

Logics of time are the topic of Chapter 3. This chapter describes how time and temporal databases can be modeled in logic and situates first-order temporal logic among query and integrity constraint languages for temporal databases. Although temporal logic has been studied by logicians for a long time, its use in this area is new and leads to many interesting research problems. For example, one gains a new perspective on the properties of existing temporal query languages.

Chapter 4 addresses the issue of formalizing permission and obligation using deontic logic. Again, deontic logic is a well-known formalism but only recently it has been applied in the context of information systems. Using deontic logic one is able to introduce fine semantic distinctions into the specification of information systems. For example, one can vary the force with which integrity constraints hold to obtain “soft” constraints.

Database dynamics is a problem notoriously hard to formalize in logic. Chapter 5 describes a new logic of updates and transactions, called Transaction Logic, that addresses this very issue. In the Horn fragment of Transaction Logic, deduction coincides with transaction execution, thanks to an original, path-based semantics. A complete proof theory for this fragment is presented in the chapter, together with many examples and a detailed discussion of related work.