CHAPTER 1

DATABASE CONCEPTS

CHAPTER OBJECTIVES

Upon completion of this chapter, you should be able to:

- Define database and database management system
- Describe the steps for the database development process
- Explain the purpose of conceptual, logical, and physical database design
- Explain the relational database model

INTRODUCTION TO DATABASE AND DATABASE MANAGEMENT SYSTEM

A **database management system** (**DBMS**) is a set of software programs that control the organization, storage, and retrieval of data in a database. A **database** is an organized collection of related tables that can be joined to provide information to users. Common relational database management systems that include Structured Query Language (SQL) are IBM® DB2®, Microsoft SQL Server®, MySQL, Oracle®, Sybase, and others.

A DBMS is a software system that uses a standard method of cataloging, retrieving, and running queries on data. The DBMS manages incoming data, organizes it, and provides ways for the data to be modified or extracted by users or other programs.

A DBMS lets users create and access data in a database. The DBMS manages database requests so users and programs do not have to understand where the data are physically located on disk or who else may be accessing the data. In handling database requests, the DBMS ensures the integrity of the data; that is, it makes sure the data are available and are organized as intended. In addition, the DBMS provides security by ensuring that only those users and programs with access privileges can access the data.

RELATIONAL DATABASE MODEL

The foundation of most commercial database management systems today is the **relational database model**, which was first introduced in the paper "A Relational Model of Data for Large Shared Data Banks," published by Edgar Codd in 1970 [*Communications of the ACM* 13 (6): 377–387]. Since the publication of this paper, the relational model has been developed extensively into common relational database management systems such as DB2, Microsoft SQL Server, MySQL, Oracle, and others. In addition to relational databases, there are also hierarchical, network, and object-oriented databases.

THE DB2 DATABASE

DB2 refers to an entire family of IBM databases:

- DB2 for i
- DB2 for z/OS
- DB2 for Linux, UNIX and Windows (also referred to as DB2 LUW)

This book uses the IBM database management system DB2 for i, which is referred to simply as "DB2" throughout the book.

DB2 for i is an integrated part of IBM i, the operating system that runs IBM's System i® hardware platform. DB2 is not purchased as a separate software product, and any application can take advantage of its features. DB2 provides a DBMS that all computer programs use to access data stored in DB2 databases.

As mentioned, a database is an organized collection of related tables that can be joined to provide information to users. In the DB2 database approach, tables are defined independently of any applications that use them. Data remains in independent, simple, linear tables, but indexes are created that indicate how the rows of the different tables relate to each other in the sense of the join concept. These indexes can be created before any queries are run, or they can be created at run time as the join query is executed.

DATABASE TERMINOLOGY

There are two groups of terms used interchangeably when discussing databases. The reason there are two sets of terms is that before the use of SQL, people used traditional system terms. With SQL, a new set of terms more representative of a typical spreadsheet type of relational table became more common. Thus, the non-SQL world sometimes uses different terms than the SQL world uses. The table in Figure 1-1 shows the relationship between the SQL relational database terms and the non-SQL terms.

SQL term	Non-SQL term
Schema. A logical grouping consisting of a library, a journal, a journal receiver, an SQL catalog, and optionally a data dictionary. A schema groups related objects and lets the objects be found by name.	*Library.* Groups related objects and allows the objects to be found by name.
Table. A set of columns and rows.	*Physical file.* A set of records.
Row. The horizontal part of a table containing a set of columns.	*Record.* A set of fields.
Column. The vertical part of a table of one data type.	*Field.* One or more characters of related information of one data type.
View. A subset of columns and rows of one or more tables.	*Logical file.* A subset of fields and records of one or more physical files.
Package. An object type that is used to run SQL statements.	*SQL package.* An object type that is used to run SQL statements.

FIGURE 1-1 Relationships between SQL terms and non-SQL terms

THE IMPORTANCE OF DATABASE DESIGN

Database design is an important stage of a development project that should precede the coding and creation of a database. Before creating database tables, one should know exactly *what* kinds of tables need to be created. Consider how meaningless it would be to create a customer table before knowing what customer data a company needed to have. For example, some companies might need to know their customers' occupations, while others might not care what their customers do for a living. The general requirement for database tables, of course, is to store data that the organization needs so the data can be used effectively. Determining exactly what those needs are and how best to implement a database that meets the requirements clearly should happen *before* coding begins.

Determining business requirements and documenting them, including both data and processes that use data, are tasks often collectively referred to as "modeling" because the result of the effort is a model, usually in diagrams, of the way the business works and the data it uses. The part of this effort that concentrates on the organization's data requirements is called **database design**.

Once a data model is completed, there is a specification of what data needs to be stored and at least some of the ways in which the data is used. For example, the data model for a particular company might specify that the company needs to keep track of customers, including their names, shipping and billing addresses, telephone and fax numbers, and credit ratings. This data model might also document that the company needs complete customer lists, in order by the customers' names. From this model or specification, one can then decide which data should be stored in database tables. A data model says *what* the organization needs, not *how* a system will be implemented to address those needs.

A system to provide for storing, updating, and retrieving the data specified in a particular data model might be implemented in any number of ways. The task of deciding how to implement a system for a particular data model is called physical database design or database design. The term "database design" is often used as an abbreviated way to refer to both the logical data modeling and the physical database design processes.

A database model identifies what data are to be stored in a database, how the data will be used, and how the tables in the database are related to each other. A database model that is well designed reduces the need for changes. The final development step, or actual **database implementation**, is the coding and creation of the database tables according to the database design specification.

DATABASE DEVELOPMENT PROCESS

As mentioned, a database is an organized collection of related tables that can be joined to provide information to users. The collection of related tables that comprise a database can be quite complex to develop. Because of this, it is important that a set of steps be followed in the development of a database so users can retrieve the necessary information when needed and in a timely way. The process of doing database design generally consists of a number of steps that are carried out by a systems analyst or database designer. Not all of these steps are necessary in all cases. Logically, the steps flow sequentially; however, in practice it is common to skip around as knowledge about the system being developed is gathered. The database design process may include the following steps:

1. Database planning
2. Requirements analysis—Focused on the requirements that need to be represented in the system
3. Database design—The conceptual and logical database design
4. Physical database design
5. DBMS selection—Based on hardware and operating system
6. Database implementation—Creating the database
7. Testing and evaluation—Testing and evaluating the database to determine whether all user requirements have been met

8. Database deployment—Placing the database into production
9. Database maintenance—Maintaining the database system according to user needs

The primary purpose of this book is to focus on the design and implementation of databases. For that reason, this book focuses on database design and database implementation.

DATABASE PLANNING

The database planning phase begins when a customer submits a request for the development of an information system project. It consists of a set of tasks or activities that decide both the resources required in the database development and the time limits of different activities. During the planning phase, four major activities are performed:

- Review and approve the database project request.
- Prioritize the database project request.
- Allocate resources such as money, people, and tools.
- Arrange a development team to develop the database project.

Database planning should also include the development of standards that govern how data will be collected, how the format should be specified, and what documentation will be needed.

Before embarking on a major data modeling project, the developer should get two things in order: naming conventions and how to store the gathered information. Many business applications involve hundreds of variable names, and keeping them organized in data modeling can be difficult. Establish abbreviation and naming standards for data model objects such as tables, columns, relationships, domains, and programs. Most important, make sure names are consistent and express clearly what an item represents or stores.

Following the naming standard, one table should be created for valid abbreviations that can be used to form names, and another table should be created for valid names. In other words, before starting to generate names, know how they are going to be formed. It may be necessary to record a lot of information about names, including synonyms, preferred usage, a description, and short, standard, and long variations.

Once naming standards have been established and a place to track the names has been settled on, the next step is to create a **data dictionary**. A data dictionary is not one thing. It is many things: all the containers used to hold everything that is recorded during the modeling process. A typical project might have some word-processing documents that contain descriptions of business rules or processes, a database table that contains table and column names and brief descriptions, and some entity relationship diagrams. Together, all these records of the data model make up the data dictionary.

It is good practice to establish naming standards and tables before creating too many other computer-based data dictionary objects, such as documents and files, because as the data model progresses, it is important for the dictionary objects as well as the business objects to follow a rational naming standard.

REQUIREMENTS ANALYSIS

The first step in developing an information system is the requirements analysis usually conducted by a systems analysis. The purpose of the **requirements analysis** is to gather the business requirements so they can be used in the development of a data model. This step involves gathering information about processes, entities (categories of data), and organizational units. After this information is collected, it is used in the database design step. The diagrams produced should show the processes and data that exist as well as the relationships between business processes and data.

Data modeling goes together with systems analysis, and the two occur together as a development project proceeds. Systems analysis includes discovering such aspects as how orders are entered and fulfilled and how customer credit limits are calculated. Obviously, the description of a process requires referring to the data model to identify where some of the values used in calculations come from and where user input and the results of calculations are stored. The data model may also refer to process definitions for actions that are triggered when some change occurs in the database.

During the requirements analysis phase, the requirements and expectations of the users are collected and analyzed. The collected requirements help to understand what the new system will accomplish. The goal of this phase is essentially to understand how the company works, discover what problems and limitations users have, understand what the company wishes to accomplish, and define the scope and boundaries of the project. The scope and boundaries are essential to make sure the database is created exactly as specified.

Much of the information is developed from interviewing end users or reading procedure manuals, forms, and other sources that explain how the business works. Thus, when talking to an end user or reading a procedure manual, the developer may learn things that are covered in several of the later steps of the process. The first conversation may provide enough information to enable the development of a broad overview of the most important entities and their interrelationships, with details being filled in through further conversations and reading documents.

In most cases, a person who is doing the design of a database is a person with expertise in the area of database design rather than expertise in the area within the company from which the data to be stored is drawn (e.g., financial, manufacturing, transportation). Therefore, the data to be stored in the database must be determined in cooperation with the people who have expertise in that area and are aware of what data must be stored within the system. This process is one that is generally considered part of requirements analysis, and it requires skill on the part of the analyst to obtain the needed information from those with the knowledge. This is because the individuals with the necessary knowledge frequently cannot express clearly what their system requirements for the database are because they are unaccustomed to thinking in terms of the data elements that must be stored.

DATABASE DESIGN

With a good idea of what the organization needs to keep track of and how that data is used, the next step is to decide how best to implement a system to support the organization's requirements. Suppose the business has customers; is it obvious that a customer database table is required? What if the business has fairly independent divisions in different regions of the country? Should there be a single customer table at the home office, or should there be separate tables in each regional office? Questions such as these are answered in the database design stage, which follows the data requirements analysis stage.

After all the requirements have been gathered for a proposed database, they must be modeled. Database design is the process of producing a detailed data model of the database that will meet end-user requirements. Models are created to visually represent the proposed database so that business requirements can easily be associated with database objects to ensure that all requirements have been completely and accurately gathered. Different types of diagrams are typically produced to illustrate the business processes, rules, entities, and organizational units that have been identified. The actual model is frequently called an **entity relationship model (ERM)** because it depicts data in terms of the entities and relationships described in the data. An entity relationship model is an abstract conceptual representation of structured data. Basically, data modeling serves as a link between business needs and system requirements.

Designers create an abstract data model that attempts to model real-world objects by creating a conceptual design. Designers must consider end-user views; define entities, attributes, and relationships; and identify processes and access requirements. The conceptual design is then translated into the logical design, which is DBMS-dependent. Database design includes three design steps:

1. Conceptual database design
2. Logical database design
3. Physical database design

The significance of this approach is that it allows the three perspectives to be relatively independent of each other. Storage technology can change without affecting either the conceptual or the logical model. The table/column structure can change without (necessarily) affecting the conceptual model. In each case, of course, the structures must remain consistent with the other model. The table/column structure may be different from a direct translation of the entity classes and attributes, but it must ultimately carry out the objectives of the conceptual entity class structure. Early phases of many software development projects emphasize the design of a conceptual data model. Such a design can be detailed into a logical data model. In later stages, this model may be translated into a physical data model.

CONCEPTUAL DATABASE DESIGN

Conceptual database design is the process of constructing a data model from the information collected in the requirements analysis phase. The **conceptual model** is a representation of an organization's data, organized in terms of entities, attributes, and relationships; it is independent of any particular DBMS. This phase is independent of physical considerations and involves constructing an ER model and checking the model for redundancy.

ER MODEL

A pictorial representation of the real-world problem in terms of entities, attributes, and relations between the entities is referred as an entity relationship diagram (ERD). An ERD includes:

- *Entities:* An entity is a class of distinct identifiable objects or concepts.
- *Attributes:* Attributes are properties or characteristics of entities.
- *Relations:* Relations are associations between or among entities.

LOGICAL DATABASE DESIGN

Logical database design consists of converting the entity relationship diagram from the conceptual model into tables or relational schemas in normal forms using a technique called normalization.

Once the relationships and dependencies amongst the various pieces of information have been determined in the conceptual model, it is possible to arrange the data into a logical structure, which can then be mapped into the storage objects supported by the database management system. In the case of relational databases, the storage objects are tables that store data in rows and columns.

Each table may represent an implementation of either a logical object or a relationship joining one or more instances of one or more logical objects. Relationships between tables may then be stored as links connecting child tables with parents. Because complex logical relationships are themselves tables, they will probably have links to more than one parent.

The logical data model contains all the needed logical and physical design choices and physical storage parameters needed to generate a design in a data definition language, which can then be used to create a database. A fully attributed data model contains detailed attributes for each entity.

NORMALIZATION OF TABLES

Normalization is used to check the entity relationship model and help eliminate redundancy and other anomalies in the database. Some splitting and even recombination of entity types may result from normalization, and the entity relationship model will have to be updated accordingly. The entity relationship model and the table definitions resulting from normalization should be consistent.

PHYSICAL DATABASE DESIGN

Physical database design consists of creating tables in the selected DBMS according to the requirements that were established during logical modeling. Conceptual and logical modeling mainly involves gathering the requirements of the business, with the latter part of logical modeling directed toward the goals and requirements of the database. Physical modeling deals with the conversion of the logical, or business, model into a relational database model. When physical modeling occurs, objects are being defined at the schema level. A schema is a group of related objects in a database. A database design effort is normally associated with one schema.

During physical modeling, objects such as tables and columns are created based on entities and attributes that were defined during the conceptual and logical design phases of database design. Constraints are also defined, including primary keys, foreign keys, other unique keys, and check constraints. Views can be created from database tables to summarize data or to simply provide the user with another perspective of certain data. Other objects, such as indexes, can also be defined during physical modeling. Physical modeling is when all the pieces come together to complete the process of defining a database for a business.

Physical modeling is database-software–specific, meaning that the objects defined during physical modeling can vary depending on the relational database software being used. For example, most relational database systems have variations in the way data types are represented and the way data is stored, although basic data types are conceptually the same among different implementations. In addition, some database systems have objects that are not available in other database systems.

Physical database design is the process that weighs the alternative implementation possibilities and carefully lays out exactly how tables and other elements of the system will be implemented. For the database implementation, the most important tangible result of the design is a set of table layouts, including detailed column specifications. Other important elements of the database design are specific end-user views of the data and various integrity constraints that must be enforced. The database design details how these elements will be implemented with a particular database system.

The physical design of the database specifies the physical configuration by which data are stored on the storage media. This step involves describing the base relations, file organizations, and index design used to achieve efficient access to the data, as well as any associated integrity constraints and security measures.

DATA MODEL VS. PHYSICAL DESIGN

The logical data model and the physical database design are quite different in their objectives, goals, and content. The table shown in Figure 1-2 provides a comparison between logical data modeling and physical database design.

Logical data modeling	**Physical database design**
Includes entities, attributes, and relationships	Includes tables, columns, keys, data types, validation rules, database triggers, stored procedures, and access constraints
Uses business names for attributes	Uses abbreviated column names that are limited by the DBMS being used
Is independent of the technology that will be used to implement the database	Defines primary keys and indexes for fast data access
Is normalized to at least third normal form	May be denormalized to meet performance requirements

FIGURE 1-2 Differences between logical data modeling and physical database design

DBMS SELECTION

In many situations, the DBMS selection phase is a formality. The reason for this is many companies have already determined and implemented the hardware and a DBMS, and all future information systems are developed for their specific DBMS. Once a company determines the DBMS and the hardware on which it is going to run, the expense and person-hours required to change systems prohibits such a change.

However, there are situations in which this phase applies. Consider a company that decides to implement a new database structure and also decides to change hardware vendors and thus the DBMS. In this phase, an appropriate DBMS is selected to support the information system.

A number of factors are involved in DBMS selection, including technical and economic factors. The technical factors concern the suitability of the DBMS for information systems. The following technical factors are considered:

- Type of DBMS (e.g., relational, object-oriented)
- Storage structure and access methods that the DBMS supports
- Available user and programmer interfaces
- Type of query languages
- Development and other tools

DATABASE IMPLEMENTATION

After the design phase and selection of a suitable DBMS, the database system is implemented. The purpose of the implementation phase is to construct and install the information system according to the plan and design as described in previous phases. In this phase, the DBMS software is installed, the database (or databases) is produced, and the tables are populated with data. This phase also requires that the database performance is evaluated, security standards are set up, backup and recovery procedures are put in place, and data integrity is enforced. Finally, the database administrator must ensure that company standards are being followed by implementing and enforcing them in the database.

TESTING AND EVALUATION

The testing and evaluation phase requires that the database is tested again for performance. It is tested during implementation; however, in this phase it is tested again and fine-tuned. Testing also requires that the administrator test integrity, security, and multi-user load.

DATABASE MAINTENANCE

Once the database system is implemented, the maintenance phase of the database system begins. **Database maintenance** is the process of monitoring and maintaining the database system. Maintenance includes activities such as adding new columns to tables, changing the size of existing columns, adding new tables, adding new constraints, adding views, and so on. As the database system requirements change, it becomes necessary to add new tables or remove existing tables and to reorganize some tables by changing primary access methods or by dropping old indexes and constructing new ones. Some queries or transactions may be rewritten for better performance. Database tuning or reorganization continues throughout the life of the database and while the requirements continue to change.

OPERATION

Essentially at this point, the database is fully functional. Users are allowed to fully use the system and report any issues. Any problems are resolved according to severity.

END-OF-CHAPTER

CHAPTER SUMMARY

1. A database management system (DBMS):
   a. Is a set of software programs that controls the organization, storage, and retrieval of data in a database
   b. Uses a standard method of cataloging, retrieving, and running queries on data
   c. Manages incoming data, organizes it, and provides ways for the data to be modified or extracted by users or other programs

2. A database is an organized collection of related tables that can be joined to provide information to users.

3. SQL has its own set of database terminology:

SQL terms
Schema. A logical grouping consisting of a library, a journal, a journal receiver, an SQL catalog, and optionally a data dictionary. A schema groups related objects and lets the objects be found by name.
Table. A set of columns and rows.
Row. The horizontal part of a table containing a set of columns.
Column. The vertical part of a table of one data type.
View. A subset of columns and rows of one or more tables.
Package. An object type that is used to run SQL statements.

4. Before implementing database tables, the developer needs to know what the tables should contain and how they are related. This information is developed by three processes that should occur before implementation:
   a. Conceptual database design
   b. Logical database design
   c. Physical database design

5. Conceptual database design:
   a. Is the process of constructing a data model from the information collected in the requirements analysis phase
   b. Is a representation of an organization's data, organized in terms of entities, attributes, and relationships
   c. Is independent of any particular database management system
   d. Involves constructing an ER model and checking the model for redundancy

6. Logical database design:
   a. Consists of converting the entity relationship diagram from the conceptual model into tables or relational schemas
   b. Uses normal forms using a technique called normalization
7. Physical database design:
   a. Evaluates the anticipated volume of data, types of access, and available hardware, software, and personnel to decide the most effective way to implement the capabilities specified in the data model
   b. Results in a specification for the database tables, columns, and other related items
   c. Provides a design specification that is used as the basis for coding SQL and creating the actual tables

KEY TERMS

column	entity relationship model (ERM)
conceptual database design	logical database design
conceptual model	package
data dictionary	physical database design
database	relational database model
database design	requirements analysis
database implementation	row
database maintenance	schema
database management system (DBMS)	table
DB2	view
DB2 for i	