1.10 Summary

- A database-management system (DBMS) consists of a collection of interrelated data and a collection of programs to access those data. The data describe one particular enterprise.
- The primary goal of a DBMS is to provide an environment that is both convenient and efficient for people to use in retrieving and storing information.
- Database systems are ubiquitous today, and most people interact, either directly or indirectly, with databases many times every day.
- Database systems are designed to store large bodies of information. The management of data involves both the definition of structures for the storage of information and the provision of mechanisms for the manipulation of information. In addition, the database system must provide for the safety of the information stored in the face of system crashes or attempts at unauthorized access. If data are to be shared among several users, the system must avoid possible anomalous results.
- A major purpose of a database system is to provide users with an abstract view of the data. That is, the system hides certain details of how the data are stored and maintained.
- Underlying the structure of a database is the data model: a collection of conceptual tools for describing data, data relationships, data semantics, and data constraints.
- The relational data model is the most widely deployed model for storing data in databases. Other data models are the object-oriented model, the object-relational model, and semi-structured data models.
- A data-manipulation language (DML) is a language that enables users to access or manipulate data. Nonprocedural DMLs, which require a user to specify only what data are needed, without specifying exactly how to get those data, are widely used today.
- A data-definition language (DDL) is a language for specifying the database schema and other properties of the data.
- Database design mainly involves the design of the database schema. The entity-relationship (E-R) data model is a widely used model for database design. It provides a convenient graphical representation to view data, relationships, and constraints.
- A database system has several subsystems.
    - The storage manager subsystem provides the interface between the low-level data stored in the database and the application programs and queries submitted to the system.

  - The query processor subsystem compiles and executes DDL and DML statements.
- Transaction management ensures that the database remains in a consistent (correct) state despite system failures. The transaction manager ensures that concurrent transaction executions proceed without conflicts.
- The architecture of a database system is greatly influenced by the underlying computer system on which the database system runs. Database systems can be centralized, or parallel, involving multiple machines. Distributed databases span multiple geographically separated machines.
- Database applications are typically broken up into a front-end part that runs at client machines and a part that runs at the backend. In two-tier architectures, the front end directly communicates with a database running at the back end. In three-tier architectures, the back end part is itself broken up into an application server and a database server.
- There are four different types of database-system users, differentiated by the way they expect to interact with the system. Different types of user interfaces have been designed for the different types of users.
- Data-analysis techniques attempt to automatically discover rules and patterns from data. The field of data mining combines knowledge-discovery techniques invented by artificial intelligence researchers and statistical analysts with efficient implementation techniques that enable them to be used on extremely large databases.

Review Terms

- Database-management system (DBMS)
- Database-system applications
- Online transaction processing
- Data analytics
- File-processing systems
- Data inconsistency
- Consistency constraints
- Data abstraction
  - Physical level
  - Logical level
  - View level
- Instance
- Schema
  - Physical schema
  - Logical schema
  - Subschema
- Physical data independence
- Data models
  - Entity-relationship model
  - Relational data model
  - Semi-structured data model
  - Object-based data model

- Database languages
  - Data-definition language
  - Data-manipulation language
    - Procedural DML
    - Declarative DML
    - nonprocedural DML
  - Query language
- Data-definition language
  - Domain Constraints
  - Referential Integrity
  - Authorization
    - Read authorization
    - Insert authorization
    - Update authorization
    - Delete authorization
- Metadata
- Application program
- Database design
  - Conceptual design
  - Normalization
  - Specification of functional requirements
  - Physical-design phase
- Database Engine
  - Storage manager
    - Authorization and integrity manager
    - Transaction manager
    - File manager
    - Buffer manager
    - Data files
    - Data dictionary
    - Indices
  - Query processor
    - DDL interpreter
    - DML compiler
    - Query optimization
    - Query evaluation engine
  - Transactions
    - Atomicity
    - Consistency
    - Durability
    - Recovery manager
    - Failure recovery
    - Concurrency-control manager
- Database Architecture
  - Centralized
  - Parallel
  - Distributed
- Database Application Architecture
  - Two-tier
  - Three-tier
  - Application server
- Database administrator (DBA)

Practice Exercises

1.1 This chapter has described several major advantages of a database system. What are two disadvantages?

1.2 List five ways in which the type declaration system of a language such as Java or C++ differs from the data-definition language used in a database.

2.6.9 Equivalent Queries

Note that there is often more than one way to write a query in relational algebra. Consider the following query, which finds information about courses taught by instructors in the Physics department:

$$\sigma_{dept\_name = \text{"Physics"}}(instructor \bowtie_{instructor.ID = teaches.ID} teaches)$$

Now consider an alternative query:

$$(\sigma_{dept\_name = \text{"Physics"}}(instructor)) \bowtie_{instructor.ID = teaches.ID} teaches$$

Note the subtle difference between the two queries: in the first query, the selection that restricts *dept_name* to Physics is applied after the join of *instructor* and *teaches* has been computed, whereas in the second query, the selection that restricts *dept_name* to Physics is applied to *instructor*, and the join operation is applied subsequently.

Although the two queries are not identical, they are in fact **equivalent**; that is, they give the same result on any database.

Query optimizers in database systems typically look at what result an expression computes and find an efficient way of computing that result, rather than following the exact sequence of steps specified in the query. The algebraic structure of relational algebra makes it easy to find efficient but equivalent alternative expressions, as we will see in Chapter 16.

2.7 Summary

- The relational data model is based on a collection of tables. The user of the database system may query these tables, insert new tuples, delete tuples, and update (modify) tuples. There are several languages for expressing these operations.
- The schema of a relation refers to its logical design, while an instance of the relation refers to its contents at a point in time. The schema of a database and an instance of a database are similarly defined. The schema of a relation includes its attributes, and optionally the types of the attributes and constraints on the relation such as primary and foreign-key constraints.
- A superkey of a relation is a set of one or more attributes whose values are guaranteed to identify tuples in the relation uniquely. A candidate key is a minimal superkey, that is, a set of attributes that forms a superkey, but none of whose subsets is a superkey. One of the candidate keys of a relation is chosen as its primary key.
- A foreign-key constraint from attribute(s) A of relation r_1 to the primary-key B of relation r_2 states that the value of A for each tuple in r_1 must also be the value of B for some tuple in r_2. The relation r_1 is called the referencing relation, and r_2 is called the referenced relation.

- A schema diagram is a pictorial depiction of the schema of a database that shows the relations in the database, their attributes, and primary keys and foreign keys.
- The relational query languages define a set of operations that operate on tables and output tables as their results. These operations can be combined to get expressions that express desired queries.
- The relational algebra provides a set of operations that take one or more relations as input and return a relation as an output. Practical query languages such as SQL are based on the relational algebra, but they add a number of useful syntactic features.
- The relational algebra defines a set of algebraic operations that operate on tables, and output tables as their results. These operations can be combined to get expressions that express desired queries. The algebra defines the basic operations used within relational query languages like SQL.

Review Terms

- Table
- Relation
- Tuple
- Attribute
- Relation instance
- Domain
- Atomic domain
- Null value
- Database schema
- Database instance
- Relation schema
- Keys
  - Superkey
  - Candidate key
  - Primary key
  - Primary key constraints
- Foreign-key constraint
  - Referencing relation
  - Referenced relation
- Referential integrity constraint
- Schema diagram
- Query language types
  - Imperative
  - Functional
  - Declarative
- Relational algebra
- Relational-algebra expression
- Relational-algebra operations
  - Select σ
  - Project Π
  - Cartesian product $\times$
  - Join $\bowtie$
  - Union $\cup$
  - Set difference $-$
  - Set intersection $\cap$
  - Assignment $\leftarrow$
  - Rename ρ