Review for Midterm

Chapters 1, 2, 3, 4, & 5

Chapter 1. Introduction to Databases

- ***The Database Management System (DBMS)*** **is the underlying framework of the information system and has fundamentally changed the way in which many organizations operate.**
- **The predecessor to the DBMS was the *file-based system*. Although the file-based system was a great improvement over the manual filing system, it still has significant problems, mainly the amount of data redundancy present and program—data dependence.**

Chapter 1

- **A *database* is a shared collection of logically related data and a description of this data. A *DBMS* is a software system that enables users to define, create, maintain, and control access to the database. An *application program* is a computer program that interacts with the database by issuing an appropriate request (typically a SQL statement) to the DBMS. The more inclusive term *database system* is used to define a collection of application programs that interact with the database along with the DBMS and database itself.**

Chapter 1

- **All access to the database is through the DBMS. The DBMS provides a *Data Definition Language (DDL)*, which allows users to define the database, and a *Data Manipulation Language (DML)*, which allows users to insert, update, delete, and retrieve data from the database.**
- **The DBMS provides controlled access to the database. It provides security, integrity, concurrency and recovery control, and a user-accessible catalog. It also provides a view mechanism to simplify the data that users have to deal with.**

Chapter 1

- **The DBMS environment consists of hardware (the computer), software (the DBMS, operating system, and applications programs), data, procedures, and people. The people include data and database administrators, database designers, application developers, and end-users.**
- **The *hierarchical model* represents the first generation of DBMSs. The *relational model* represents the second generation of DBMSs. The third generation of DBMSs are represented by the *Object-Relational DBMS* and the *Object-Oriented DBMS.***

Chapter 1

- **Some advantages of the database approach include control of data redundancy, data consistency, sharing of data, and improved security and integrity. Some disadvantages include complexity, cost, reduced performance, and higher impact of a failure.**

Chapter 2. Database Environment

- **The ANSI-SPARC database architecture uses *three levels* of abstraction: *external, conceptual, and internal*. The *external level* consists of the users' views of the database. The *conceptual level* is the logical view of the database: it specifies the information content of the entire database, independent of storage considerations. The *internal level* is the computer's view of the database: it specifies how data is represented, how records are sequenced, what indexes and pointers exist, and so on.**

Chapter 2

- The *external/conceptual mapping* transforms requests and results between the external and conceptual levels. *The conceptual/internal mapping* transforms requests and results between the conceptual and internal levels.
- A *database schema* is a description of the database structure. Data independence makes each level immune to changes to lower levels. *Logical data independence* refers to the immunity of the external schemas to changes in the conceptual schema. *Physical data independence* refers to the immunity of the conceptual schema to changes in the internal schema.

Chapter 2

- A data sublanguage consists of two parts: a *Data Definition Language (DDL)* and a *Data Manipulation Language (DML)*. The DDL is used to specify the database schema and the DML is used to both read and update the database. The part of a DML that involves data retrieval is called a *query language*.
- A *data model* describes the data and falls into three broad categories: *object-based* data models (external level), *record-based* data models (conceptual level), and *physical* data models (internal level).

Chapter 2

- ***Conceptual modeling*** **is the process of constructing a detailed architecture for a database that is independent of implementation details, such as the target DBMS, application programs, programming languages, or any other physical considerations. The design of the conceptual schema is critical to the overall success of the system. It is worth spending the time and effort necessary to produce the best possible conceptual design.**

Chapter 2

- ***Functions and services*** **of a multi-user DBMS include data storage, retrieval, and update; a user-accessible catalog; transaction support; concurrency control and recovery services; authorization services; support for data communication; integrity services; services to promote data independence; and utility services.**
- **The *system catalog* is one of the fundamental components of a DBMS. It contains "data about the data," or metadata. The catalog should be accessible to users. It should be able to be shared and transferred from one system to another.**

Chapter 3. Database Architectures and the Web

- *Client–server* architecture refers to the way in which software components interact. There is a *client* process that requires some resource, and a *server* that provides the resource. In the two-tier model, the client handles the user interface and business processing logic, and the server handles the database functionality.
- In the Web environment, the traditional two-tier model has been replaced by a three-tier model, consisting of a user interface layer (the *client* - web browser), a business logic and data processing layer (the *application server*), and a DBMS (the *database server*), distributed over different machines.

Chapter 3

- The three-tier architecture can be extended to *n tiers*, with additional tiers added to provide more flexibility and scalability.
- *Middleware* is computer software that connects different software components or applications in different systems.
- A *Transaction Processing (TP) Monitor* is a specific type of middleware that controls data transfer between clients and servers in order to provide a consistent environment, particularly for online transaction processing (OLTP).

Chapter 3

- A *distributed database* is a logically interrelated collection of shared data (and a description of this data), physically distributed over a computer network.
- A *distributed DBMS* is the software system that permits the management of the distributed database and makes the distribution transparent to users.

Chapter 3

- ***Cloud computing*** **is a model for enabling ubiquitous, convenient, on-demand network access to a shared pool of configurable computing resources (e.g., networks, servers, storage, applications, and services) that can be rapidly provisioned and released with minimal management effort or service provider interaction. The three main service models are: Software as a Service (SaaS), Platform as a Service (PaaS), and Infrastructure as a Service (IaaS). Cloud-based database solutions fall into two basic categories: Data as a Service (DaaS) and Database as a Service (DBaaS).**

Chapter 4. The Relational Model

- The Relational Database Management System (RDBMS) has become the dominant data-processing software in use today. This software represents the second generation of DBMSs and is based on the relational data model.
- A mathematical *relation* is a subset of the Cartesian product of two or more sets. In database terms, a relation is any subset of the Cartesian product of the domains of the attributes. A relation is normally written as a set of n-tuples, in which each element is chosen from the appropriate domain.

Chapter 4

- Relations are physically represented as *tables*, with the rows corresponding to individual tuples and the columns to attributes.
- The structure of the relation, with domain specifications and other constraints, is part of the *intension* of the database; the relation with all its tuples written out represents an *instance* or *extension* of the database.
- Properties of database relations: each cell contains exactly one atomic value, attribute names are distinct, attribute values come from the same domain, attribute order is immaterial, tuple order is immaterial, and there are no duplicate tuples.

Chapter 4

- The *degree* of a relation is the number of attributes, and the *cardinality* is the number of tuples. A *unary* relation has one attribute, a *binary* relation has two, a *ternary* relation has three, and an *n-ary* relation has n attributes.
- A *superkey* is an attribute, or set of attributes, that identifies tuples of a relation uniquely, and a *candidate key* is a minimal superkey. A *primary key* is the candidate key chosen for use in identification of tuples. A relation must always have a primary key. A *foreign key* is an attribute, or set of attributes, within one relation that is the candidate key of another relation.

Chapter 4

- A *null* represents a value for an attribute that is unknown at the present time or is not applicable for this tuple.
- *Entity integrity* states that in a base relation no attribute of a primary key can be null.
- *Referential integrity* states that foreign key values must match a candidate key value of some tuple in the home relation or be wholly null.
- Apart from relational integrity, integrity constraints also include domain constraints; other integrity constraints are called *general constraints.*

Chapter 4

- A *view* in the relational model is a *virtual* or *derived relation* that is dynamically created from the underlying base relation(s) when required. Views provide security and allow the designer to customize a user's model. Not all views are updatable.

Chapter 5. Relational Algebra and Relational Calculus

- The *relational algebra* is a procedural language: it can be used to tell the DBMS how to build a new relation from one or more relations in the database. The *relational calculus* is a nonprocedural language: it can be used to formulate the definition of a relation in terms of one or more database relations. However, formally they are equivalent to one another: for every expression in the algebra, there is an equivalent expression in the calculus (and vice versa)

Chapter 5

- **The relational calculus is used to measure the selective power of relational languages. A language that can be used to produce any relation that can be derived using the relational calculus is said to be *relationally complete*. Most relational query languages are relationally complete but have more expressive power than the relational algebra or relational calculus because of additional operations such as calculated, summary, and ordering functions.**

Chapter 5

- The five fundamental operations in relational algebra—*Selection*, *Projection*, *Cartesian product*, *Union*, and *Set difference*—perform most of the data retrieval operations that we are interested in. In addition, there are also the *Join*, *Intersection*, and *Division* operations, which can be expressed in terms of the five basic operations.
- The *relational calculus* is a formal nonprocedural language that uses predicates. There are two forms of the relational calculus: tuple relational calculus and domain relational calculus.

Chapter 5

- In the *tuple relational calculus*, we are interested in finding tuples for which a predicate is true. A tuple variable is a variable that "ranges over" a named relation: that is, a variable whose only permitted values are tuples of the relation.
- In the *domain relational calculus*, domain variables take their values from domains of attributes rather than tuples of relations.

Chapter 5

- The relational algebra is logically equivalent to a safe subset of the relational calculus (and vice versa).
- Relational data manipulation languages are sometimes classified as *procedural* or *nonprocedural*, *transform-oriented*, *graphical*, *fourth-generation*, or *fifth-generation*.