**Database System Development Lifecycle**

As a database system is a fundamental component of the larger organization-wide information system, the database system development lifecycle is inherently associated with the lifecycle of the information system. The stages of the database system development lifecycle are shown in Figure.

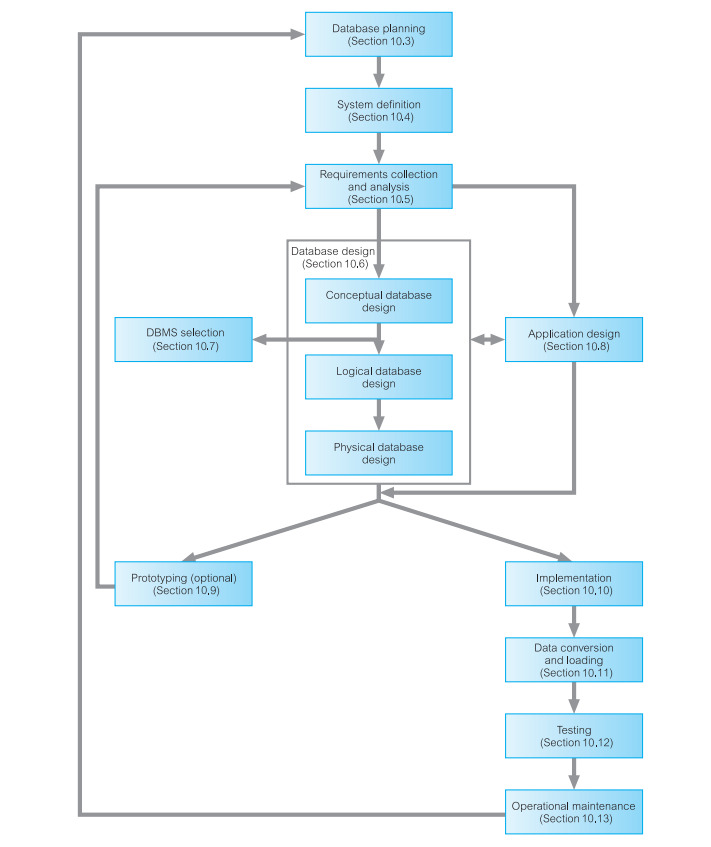
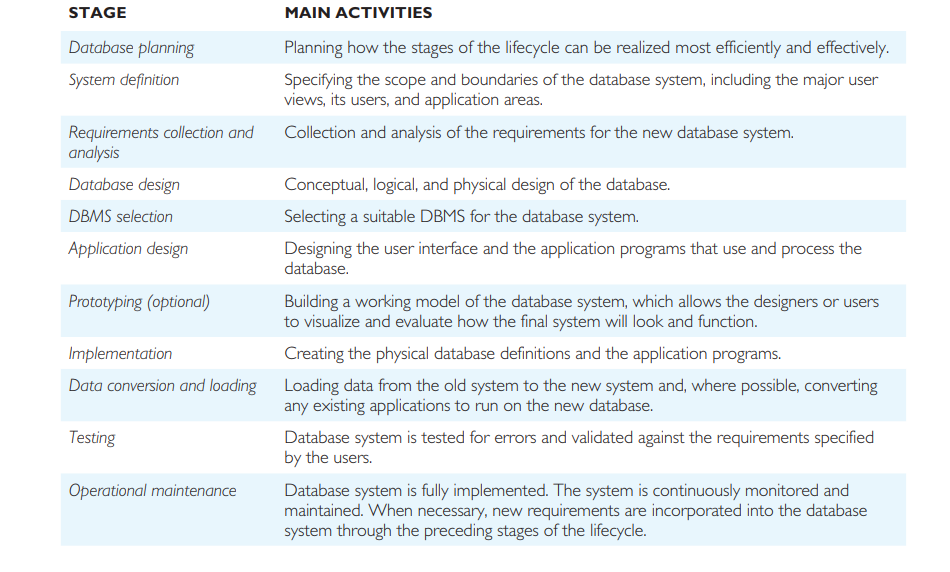
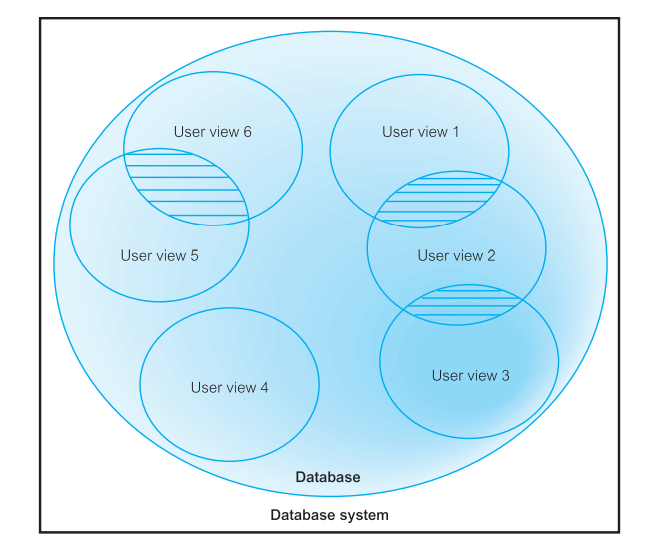


Fig:The stages of the database system development lifecycle



**Requirements Collection and Analysis**

The process of collecting and analyzing information about the part of the organization that is to be supported by the database system, and using this information to identify the requirements for the new system.

Fig:Representation of a database system with multiple user views: user views (1, 2, and 3) and

(5 and 6) have overlapping requirements (shown as hatched areas), whereas user view 4 has

distinct requirements.

This stage involves the collection and analysis of information about the part of the enterprise to be served by the database. There are many techniques for gathering this information, called fact-finding techniques. Information is gathered for each major user view including:

• A description of the data used or generated;

• The details of how data is to be used or generated;

• Any additional requirements for the new database system. This information is then analysed to identify the requirements (or features) to be included in the new database system. These requirements are described in documents collectively referred to as requirements specifications for the new database system.

Requirements collection and analysis is a preliminary stage to database design. The amount of data gathered depends on the nature of the problem and the policies of the enterprise. The information collected at this stage may be poorly structured and include some informal requests, which must be converted into a more structured statement of requirements.

This is achieved using requirements specification techniques, which include, for example, Structured Analysis and Design (SAD) techniques, Data Flow Diagrams (DFD), and Hierarchical Input Process Output (HIPO) charts supported by documentation. Computer-Aided Software Engineering (CASE) tools may provide automated assistance to ensure that the requirements are complete and consistent. Unified Modeling Language (UML) supports requirements analysis and design.

Identifying the required functionality for a database system is a critical activity, as systems with inadequate or incomplete functionality will annoy the users, which may lead to rejection or underutilization of the system. However, excessive functionality can also be problematic, as it can overcomplicate a system, making it difficult to implement, maintain, use, or learn.

Another important activity associated with this stage is deciding how to deal with the situation in which there is more than one user view for the database system.

There are three main approaches to managing the requirements of a database system with multiple user views:

• The centralized approach

• The view integration approach

• A combination of both approaches

**Centralized Approach**

**Requirements for each user view are merged into a single set of requirements for the new database system. A data model representing all user views is created during the database design stage**.

The centralized (or one-shot) approach involves collating the requirements for different user views into a single list of requirements. The collection of user views is given a name that provides some indication of the application area covered by all the merged user views. In the database design stage, a global data model is created, which represents all user views. The global data model is composed of diagrams and documentation that formally describe the data requirements of the users. A diagram representing the management of user views 1 to 3 using the centralized approach is shown in Figure 10.3. Generally, this approach is preferred when there is a significant overlap in requirements for each user view and the database system is not overly complex.

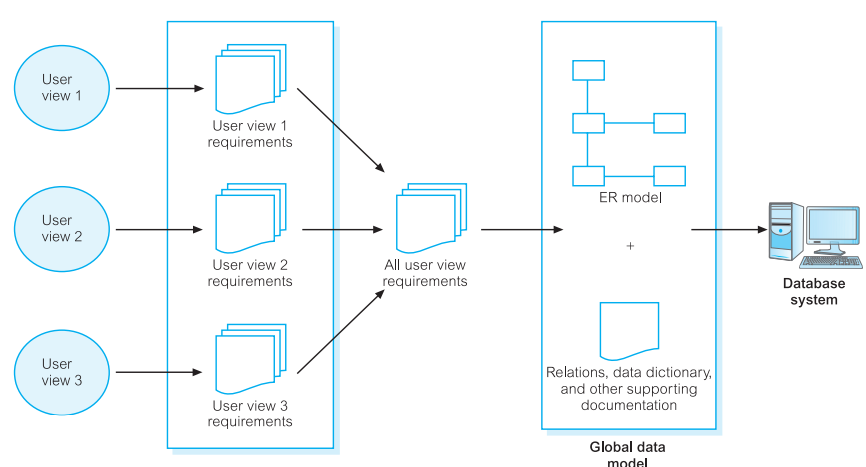


Fig: The centralized approach to managing multiple user views 1 to 3

**View Integration Approach**

Requirements for each user view remain as separate lists. Data models representing each user view are created and then merged later during the database design stage.

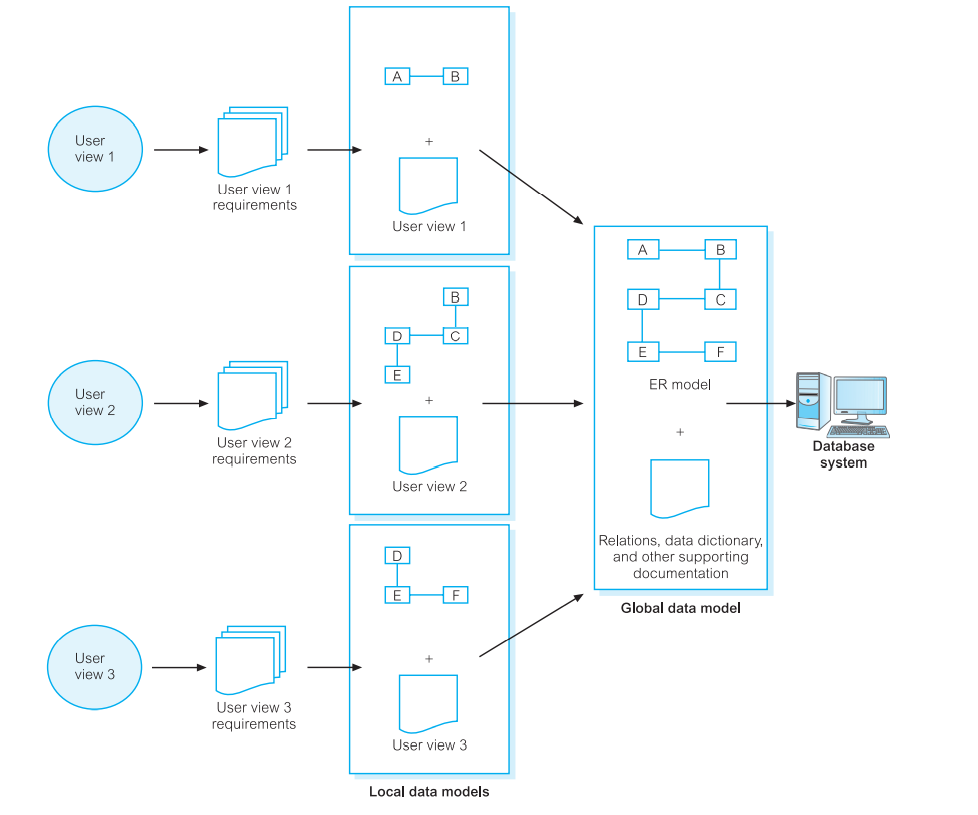


Fig: The view integration approach to managing multiple user views 1 to 3

**Database Design**

The process of creating a design that will support the enterprise’s mission statement and mission objectives for the required database system.

**i)Approaches to Database Design**

The two main approaches to the design of a database are referred to as “**bottomup” and “top-down.”**

The bottom-up approach begins at the fundamental level of attributes (that is, properties of entities and relationships), which through analysis of the associations between attributes are grouped into relations that represent types of entities and relationships between entities.

The bottom-up approach is appropriate for the design of simple databases with a relatively small number of attributes. However, this approach becomes difficult when applied to the design of more complex databases with a larger number of attributes, where it is difficult to establish all the functional dependencies between the attributes.

A more appropriate strategy for the design of complex databases is to use the top-down approach. This approach starts with the development of data models that contain a few high-level entities and relationships and then applies successive top-down refinements to identify lower-level entities, relationships, and the associated attributes. The top-down approach is illustrated using the concepts of the Entity-Relationship (ER) model, beginning with the identification of entities and relationships between the entities, which are of interest to the organization.

For example, we may begin by identifying the entities PrivateOwner and PropertyForRent, and then the relationship between these entities, PrivateOwner Owns PropertyForRent, and finally the associated attributes such as PrivateOwner (ownerNo, name, and address) and PropertyForRent (propertyNo and address).

**ii) Data Modeling**

The two main purposes of data modeling are to assist in the understanding of the meaning (semantics) of the data and to facilitate communication about the information requirements. Building a data model requires answering questions about entities, relationships, and attributes. In doing so, the designers discover the semantics of the enterprise’s data, which exist whether or not they happen to be recorded in a formal data model. Entities, relationships, and attributes are fundamental to all enterprises.

A data model makes it easier to understand the meaning of the data, and thus we model data to ensure that we understand:

• Each user’s perspective of the data

• The nature of the data itself, independent of its physical representations

• The use of data across user views

Data models can be used to convey the designer’s understanding of the information requirements of the enterprise.

**iii) Phases of Database Design**

Database design is made up of three main phases: conceptual, logical, and physical design.

Conceptual database design

The process of constructing a model of the data used in an enterprise, independent of all physical considerations.

Logical database design

The process of constructing a model of the data used in an enterprise based on a specific data model, but independent of a particular DBMS and other physical considerations.

Physical database design

The process of producing a description of the implementation of the database on secondary storage; it describes the base relations, file organizations, and indexes used to achieve efficient access to the data, and any associated integrity constraints and security measures.

