Definition

Conceptual design of secure databases encompasses the creation of a platform-independent data model that incorporates security requirements and constraints. Following the requirements analysis, conceptual design is the basis for further steps in database design that subsequently transform the database conceptualization into a platform-dependent model and an implementation.

Background

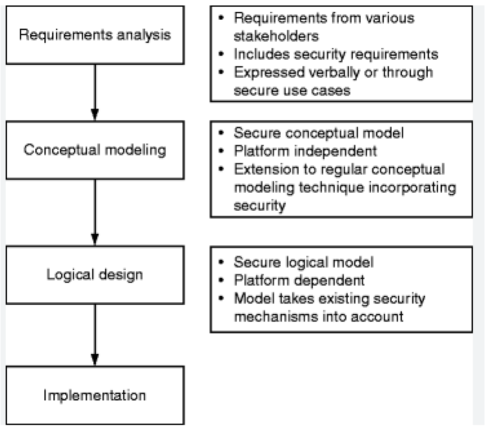
The conceptual design produces a platform-independent model of a universe of discourse and is an important step in any database design methodology. As security concerns are essential for many database applications, it is important to incorporate security requirements early in the design process of a database. Extensions to established data modeling techniques have been proposed to add security semantics.

Conceptual Design in Database Design Process

Conceptual database design is part of the database design process, which consists of the activities requirements gathering and analysis, conceptual...

Requirements Analysis is the stage in the design cycle when you find out everything you can about the data the client needs to store in the database and the conditions under which that data needs to be accessed.

Keep in mind, too, that a single pass through this stage rarely yields all the information the database designer needs. Be prepared to return to the tasks associated with Requirements Analysis several times during the course of designing a database.



Conceptual design is the first stage in the database design process. The goal at this stage is to design a database that is independent of database software and physical details. The output of this process is a conceptual data model that describes the main data entities, attributes, relationships, and constraints of a given problem domain. This design is descriptive and narrative in form. Keep in mind the following minimal data rule:

> *"All that is needed is there, and all that is there is needed".*

In other words, make sure that all data needed are in the model and that all data in the model are needed. All data elements required by the database transactions must be defined in the model, and all data elements defined in the model must be used by at least one database transaction. The conceptual design has four steps, which are as follows.

1. Data analysis and requirements
2. Entity relationship modeling and normalization
3. Data model verification
4. Distributed database design

1. Data Analysis and Requirements:

The first step in conceptual design is to discover the characteristics of the data elements. Appropriate data element characteristics are those that can be transformed into appropriate information. Therefore, the designer's efforts are focused on:

- Information needs. What kind of information is needed—that is, what output (reports and queries) must be generated by the system, what information does the current system generate, and to what extent is that information adequate?
- Information users. Who will use the information? How is the information to be used? What are the various end-user data views?
- Information sources. Where is the information to be found? How is the information to be extracted once it is found?
- Information constitution. What data elements are needed to produce the information? What are the data attributes? What relationships exist among the data? What is the data volume? How frequently are the data used? What data transformations are to be used to generate the required information? The designer obtains the answers to those questions from a variety of sources in order to compile the necessary information. Note these sources:
- Developing and gathering end-user data views. The database designer and the end user(s) interact to jointly develop a precise description of end-user data views. In turn, the end-user data views will be used to help identify the database's main data elements.
- Directly observing the current system: existing and desired output. The end user usually has an existing system in place, whether it's manual or computer-based. The designer reviews the existing system to identify the data and their characteristics.
- Interfacing with the systems design group. The database design process is part of the Systems Development Life Cycle (SDLC). In some cases, the systems analyst in charge of designing the new system will also develop the conceptual database model.

2. Entity Relationship Modeling and Normalization:

Before creating the ER model, the designer must communicate and enforce appropriate standards to be used in the documentation of the design. The process of defining business rules and developing the conceptual model using ER diagrams can be described using the following steps.

1. Identify, analyze, and refine the business rules.
2. Identify the main entities, using the results of Step 1.
3. Define the relationships among the entities, using the results of Steps 1 and 2.
4. Define the attributes, primary keys, and foreign keys for each of the entities.
5. Normalize the entities. (Remember that entities are implemented as tables in an RDBMS.)
6. Complete the initial ER diagram.
7. Validate the ER model against the end users' information and processing requirements.
8. Modify the ER model, using the results of Step 7.

3. Data Model Verification:

The data model verification step is one of the last steps in the conceptual design stage, and it is also one of the most critical ones. In this step, the ER model must be verified against the proposed system processes in order to corroborate that the intended processes can be supported by the database model. Verification requires that the model be run through a series of tests against:

- End-user data views.
- All required transactions: SELECT, INSERT, UPDATE, and DELETE operations.
- Access rights and security.
- Business-imposed data requirements and constraints.

4. Distributed Database Design:

Although not a requirement for most databases, sometimes a database may need to be distributed among multiple geographically disperse locations. Processes that access the database may also vary from one location to another. For example, a retail process and a warehouse storage process are likely to be found in different physical locations. If the database data and processes are to be distributed across the system, portions of a database, known as database fragments, may reside in several physical locations.

A database fragment is a subset of a database that is stored at a given location. The database fragment may be composed of a subset of rows or columns from one or multiple tables.

Introduction to Logical Design

Logical database design is the process of determining the logical data structures that are required to support information resources within an organization. The logical design process helps you to implement a database that satisfies the requirements of your business organization.

Logical design is critical to the implementation of a corporate database. An incomplete or flawed logical design can cause costly changes to the means of data collection, storage, and protection later on. By using a well-conceived preliminary design, you can easily implement and test a database. A sound logical design therefore helps to ensure a successful implementation.

A complete and accurate logical design for a database helps to ensure:

Data independence
The logical design process yields a database model that is independent of program or physical storage requirements. This model represents the way data structures appear to users. The model does not specify how data structures are maintained or processed by the computer.

Physical database flexibility
Logical design is independent of storage and performance requirements. Therefore, you can use it to implement a database that is used with any hardware or software system. During the physical design process, the logical design can be tailored to satisfy the needs of particular users or to suit a particular data processing environment.

Integrity
The logical design identifies both the data that is maintained in your corporation and the rules of the business. These business rules can be used later to define integrity rules for the physical design.

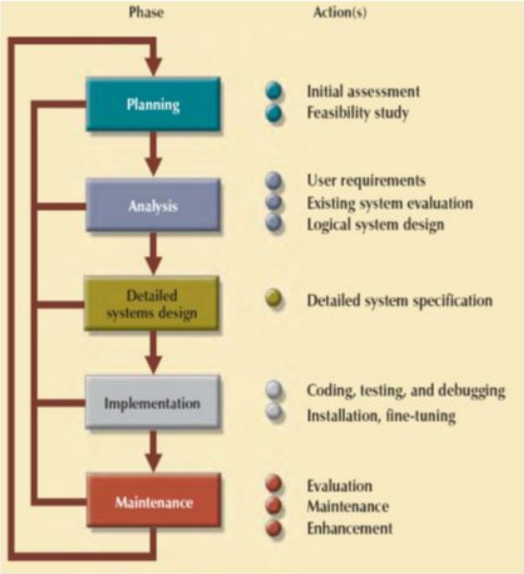
User satisfaction
The logical design represents data structures in a simple, understandable format. You can show the design to users at any stage of development without intimidating them. The logical design can be easily modified to incorporate user suggestions and feedback.

Many viable approaches exist for logical database design. In this section, we combine several design techniques, including systems analysis, the entity-relationship approach, and normalization.

Systems Development Life Cycle (SDLC)

The Systems Development Life Cycle (SDLC) traces the history (life cycle) of an information system. Perhaps more important to the system designer, the SDLC provides the big picture within which the database design and application development can be mapped out and evaluated.

The traditional SDLC is divided into five phases: planning, analysis, detailed systems design, implementation, and maintenance. The SDLC is an iterative rather than a sequential process. For example, the details of the feasibility study might help refine the initial assessment, and the details discovered during the user requirements portion of the SDLC might help refine the feasibility study.



The Systems Development Life Cycle (SDLC)

Because the Database Life Cycle (DBLC) fits into and resembles the Systems Development Life Cycle (SDLC), a brief description of the SDLC stages is as follows.

1. Planning:

The SDLC planning phase yields a general overview of the company and its objectives. An initial assessment of the information flow-and-extent requirements must be made during this discovery portion of the SDLC. Such an assessment should answer some important questions:

- Should the existing system be continued? If the information generator does its job well, there is no point in modifying or replacing it.
- Should the existing system be modified? If the initial assessment indicates deficiencies in the extent and flow of the information, minor (or even major) modifications might be in order. When considering modifications, the participants in the initial assessment must keep in mind the distinction between wants and needs.
- Should the existing system be replaced? The initial assessment might indicate that the current system's flaws are beyond fixing. Given the effort required to create a new system, a careful distinction between wants and needs is perhaps even more important in this case than it is when modifying the system.
- Participants in the SDLC's initial assessment must begin to study and evaluate alternative solutions. If it is decided that a new system is necessary, the next question is whether it is feasible. The feasibility study like The technical aspects of hardware and software requirements, The system cost, The operational cost must be evaluated.

2. Analysis:

Problems defined during the planning phase are examined in greater detail during the analysis phase. A macro analysis must be made of both individual needs and organizational needs, addressing questions such as:

- What are the requirements of the current system's end users?
- Do those requirements fit into the overall information requirements?

The analysis phase of the SDLC is, in effect, a thorough audit of user requirements.

The existing hardware and software systems are also studied during the analysis phase. The result of analysis should be a better understanding of the system's functional areas, actual and potential problems, and opportunities. End users and the system designer(s) must work together to identify processes and to uncover potential problem areas. Such cooperation is vital to defining the appropriate performance objectives by which the new system can be judged.

The analysis phase also includes the creation of a logical systems design. The logical design must specify the appropriate conceptual data model, inputs, processes, and expected output requirements.

3. Detailed Systems Design

In the detailed systems design phase, the designer completes the design of the system's processes. The design includes all the necessary technical specifications for the screens, menus, reports, and other devices that might be used to help make the system a more efficient information generator. The steps are laid out for conversion from the old to the new system. Training principles and methodologies are also planned and must be submitted for management's approval.

4. Implementation:

During the implementation phase, the hardware, DBMS software, and application programs are installed, and the database design is implemented. During the initial stages of the implementation phase, the system enters into a cycle of coding, testing, and debugging until it is ready to be delivered. The actual database is created, and the system is customized by the creation of tables and views, user authorizations, and so on.

The database contents might be loaded interactively or in batch mode, using a variety of methods and devices:

- Customized user programs.
- Database interface programs.
- Conversion programs that import the data from a different file structure, using batch programs, a database utility, or both.

5. Maintenance:

Almost as soon as the system is operational, end users begin to request changes in it. Those changes generate system maintenance activities, which can be grouped into three types:

- Corrective maintenance in response to systems errors.
- Adaptive maintenance due to changes in the business environment.
- Perfective maintenance to enhance the system.

Because every request for structural change requires retracing the SDLC steps, the system is, in a sense, always at some stage of the SDLC.