Database design including integrity constraints-Part-I

Hi! Here in this session we are actually beginning to design a database. So you are going to learn about how a good database could be designed to meet user requirements and also that will cater the needs of a consistent database.

Planning the Relational Database

When you need to build a database, there is a temptation to immediately sit down at the computer, fire up your RDBMS, and start creating tables. Well, don't. There is a process you need to follow to develop a well-designed relational database and, at the start, you're a long way from actually setting up the tables in the database application. Not necessarily a long way in time, but certainly in thought.

A systematic approach to the design will save you, the designer, a lot of time and work and makes it much more likely that the "client" for the database will get something that fits the need. In this topic, you'll look at the steps of a design process that you will follow.

When you get to the point of drafting your tables and fields, you're going to use a very low tech approach to the design process—pencil and paper. You'll find lots of blank spaces in this manual to work in. When you start building databases on your own, if you're the kind of person who just cannot think unless you're looking at a computer screen, there are software tools available for modeling a database. These CASE (computer-aided software engineering) tools can be used to create diagrams and some will create documentation of the design; they can be particularly useful when a team is working on the design of a database. Additionally, some CASE products can generate commands that will actually create the tables in the RDBMS.

The idea is to draw a picture of your tables and fields and how the data in the tables is related. These are generally called entity-relationship, ER, or E/R diagrams. There are various formal systems for creating these diagrams using a specific set of symbols to represent certain objects and types of relationships. At this point in your design career, you should probably use whatever works for you. Again, a formal system becomes more useful when a group of people are working on the same design. Also, using a recognized method is helpful for documenting your design for those who come after you. For additional information on ER diagrams, you may want to read Entity-Relationship Approach to Information Modeling by P. Chen.

The Design Process

The design process includes

- Identify the purpose of the database.
- Review existing database.
- Make a preliminary list of fields.
- Make a preliminary list of tables and enter the fields.
- Identify the key fields.
- Draft the table relationships.
- Enter sample data and normalize the data.
- Review and finalize the design.

Following a design process merely ensures that you have the information you need to create the database and that it complies with the principles of a relational database. In this topic, you're going to use this process to get the point of having well-designed tables and relationships and understand how you can extract data from the tables. After that, you, as the designer, may also have to create additional objects for the application, such as the queries, forms, reports, and application control objects. Most of those tasks are application-specific and are beyond the scope of this topic.

In this topic, you'll review an outline of the process. You'll go through the first few steps of identifying the purpose of the database and, in subsequent topics, will design the tables and relationships. You're the database designer and the information contained represents the client (the person(s) who has expressed the need for a database).If you wish, you can work on a database of your own where you can be both the client and the designer. Then you have nobody to blame but yourself if it doesn't come out right.

<u>So, where to begin?</u>

1. ***Identify the purpose of the database.***

You will rarely be handed a detailed specification for the database. The desire for a database is usually initially expressed as things the client wants it to do. Things like:

- We need to keep track of our inventory.
- We need an order entry system.
- I need monthly reports on sales.
- We need to provide our product catalog on the Web.

It will usually be up to you to clarify the scope of the intended database. Remember that a database holds related information. If the client wants the product catalog on the Web and sales figures and information on employees and data on competitors, maybe you're talking about more than one database. Everyone involved needs to have the same understanding of the scope of the project and the expected outcomes (preferably in order of importance). It can be helpful to write a statement of purpose for the database that concerned parties can sign off on. Something like: "The Orders database will hold information on customers, orders, and order details. It will be used for order entry and monthly reports on sales." A statement like this can help define the boundaries of the information the database will hold.

The early stages of database design are a people-intensive phase, and clear and explicit communication is essential. The process is not isolated steps but is, to a point, iterative. That is, you'll have to keep going back to people for clarification and additional information as you get further along in the process. As your design progresses, you'll also need to get confirmation that you're on the right track and that all needed data is accounted for.

If you don't have it at the beginning of the design process, along the way you'll also need to develop an understanding of the way the business operates and of the data itself. You need to care about business operations because they involve business rules. These business rules result in constraints that you, as the database designer, need to place on the data. Examples include what the allowable range of values is for a field, whether a certain field of data is required, whether values in a field will be numbers or characters, and will numbers ever have leading zeros. Business rules can also determine the structure of and relationship between tables. Also, it will be difficult for you to determine what values are unique and how the data in different tables relates if you don't understand the meaning of the data. The reasons will be clearer when you actually get to those points in the process.

2. ***Review existing data.***

You can take a huge step in defining the body of information for the database by looking at existing data repositories. Is there an existing database (often called a legacy database) even if it isn't fitting the bill anymore? Is someone currently tracking some of the information in spreadsheets? Are there data collection forms in use? Or are there paper files?

Another good way to help define the data is to sketch out the desired outcome. For example, if the clients say they need a monthly report of sales, have them draft what they have in mind. Do they want it grouped by product line? By region? By salesperson? You can't provide groupings if you haven't got a field containing data you can group on. Do they want calculations done? You can't perform calculations if you haven't stored the component values.

Your goal is to collect as much information as you can about the desired products of the database and to reverse engineer that information into tables and fields.

3. ***Make a preliminary list of fields.***

Take all you have learned about the needs so far and make a preliminary list of the fields of data to be included in the database. Make sure that you have fields to support the needs. For example, to report on monthly sales, there's going to have to be a date associated with each sale. To group sales by product line, you'll need a product line identifier. Keep in mind that the clients for a database have expressed their need for information; it's your job to think about what data is needed to deliver that information.

Each field should be atomic; this means each should hold the smallest meaningful value and, therefore, should not contain multiple values. The most common disregard of this rule is to store a person's first name and last name in the same field.

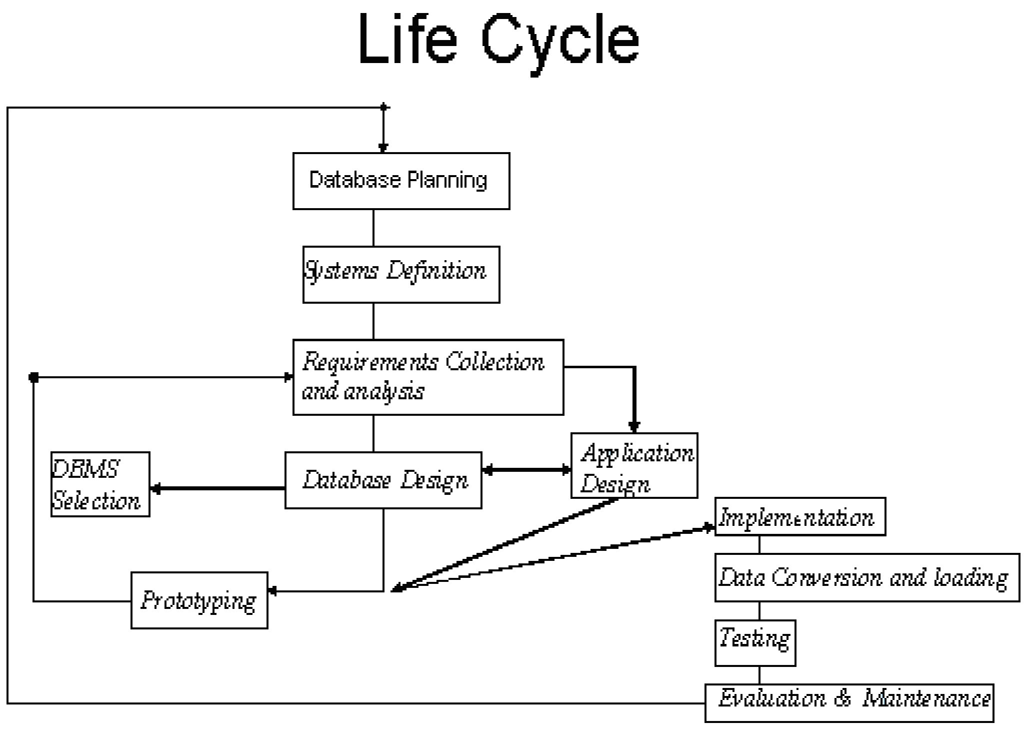
Do not include fields to hold data that can be calculated from other fields. For example, if you had fields holding an employee's hourly pay rate and weekly hours, you would not include a gross pay field.

Database Development Life Cycle

Hi, we have reached up to a point where, like the classical software development life cycle, I would like to discuss with you the various phases in the Database Development Life Cycle. Let us explore this interesting concept.

In this phase the database designers will decide on the database model that is ideally suited or the organization's needs. The database designers will study the documents prepared by the analysts in the requirements analysis phase and then will go about developing a system that satisfies the requirements. In this phase the designers will try to find out answers to the following questions:

- What are the problems in the existing system and how they could be overcome?
- What are the information needs to the different users of the system and how could the conflicting requirements be balanced?
- What data items are required for an efficient decision-making system?
- What are the performance requirements for the system?
- How should the data be structured?
- How will each user access the data?
- How is the data entered in to the database?
- How much data will be added to the database on a daily/weekly/monthly basis?



Designing Good Relational Databases

Hi now you have understood the necessity of the Database design phase. Databases have a reputation for being difficult to construct and hard to maintain. The power of modern database software makes it possible to create a database with a few mouse-clicks. The databases created this way, however, are typically the databases that are hard to maintain and difficult to work with because they are designed poorly. Modern software makes it easy to construct a database, but doesn't help much with the design aspect of database creation.

Database design has nothing to do with using computers. It has everything to do with research and planning. The design process should be completely independent of software choices. The basic elements of the design process are:

1. Defining the problem or objective

2. Researching the current database
3. Designing the data structures
4. Constructing relationships
5. Implementing rules and constraints
6. Creating views and reports
7. Implementing the design

Notice that implementing the database design in software is the final step. All of the preceding steps are completely independent of any software or other implementation concerns.

Defining the problem or objective. The most important step in database design is the first one: defining the problem the database will address or the objective of the database. It is important however, to draw a distinction between:

- How the database will be used and
- What information needs to be stored in it?

The first step of database design is to clearly delineate the nature of the data that needs to be stored, not the questions that will be asked to turn that data into information.

This may sound a little contradictory at first, since the purpose of a database is to provide the appropriate information to answer questions. However, the problem with designing databases to answer specific or targeted questions is that invariably questions are left out, change over time, or even become superseded by other questions. Once this happens, a database designed solely to answer the original questions becomes useless. In contrast, if the database is designed by collecting all of the information that an individual or organization uses to address a particular problem or objective, the information to answer any question involving that problem or objective can theoretically be addressed.

Researching the current database. In most database design situations, there is some sort of database already in existence. That database may be Post-it notes, paper order forms, a spreadsheet of sales data, a word processor file of names and addresses, or a fullfledged digital database (possibly in an outdated software package or older legacy system). Regardless of its format, it provides one essential piece of information: the data that the organization currently finds useful. This is an excellent starting point for determining the essential data structure of the database. The existing database information can also provide the nucleus for the content of the new database.

Designing the data structures. A database is essentially a collection of data tables, so the next step in the design process is to identify and describe those data structures. Each table in a database should represent some distinct subject or physical object, so it seems reasonable to simply analyze the subjects or physical objects relevant to the purpose of the database, and then arrive at a list of tables.

This can work successfully, but it's a much better to objectively analyze the actual fields that you have identified as essential in your research and see what logical groupings arise. In many cases, structures that seemed distinct are really reflections of the same underlying subject. In other cases, the complete opposite is true. And to complicate matters, organizations can use the same terms to describe data that they use or collect in fundamentally different ways.

Once the tables have been determined and fields have been assigned to each, the next step is to develop the specifications for each field. The perfect field should be atomic: It should be unique in all tables in the database (unless it is used as a key) and contain a single value, and it should not be possible to break it into smaller components. This is also an appropriate time to start thinking about the kind of data that goes in each field. This information should be fairly clear from the research phase of the project, but sometimes questions remain. Some advance planning can be done to make it easier to implement the database in the software at a later time, such as identifying the type of fields and examining (or re-examining) existing data that you've collected to make sure that the data always fits the model you are constructing. It's much easier and cheaper to fix that now than wait until the database is being rolled out!

Constructing relationships. Once the data structures are in place, the next step is to establish the relationships between the databases. First you must ensure that each table has a unique key that can identify the individual records in each table. Any field in the database that contains unique values is an acceptable field to use as a key. However, it is a much better practice to add an arbitrary field to each table that contains a meaningless, but unique value. This value is typically an integer that is assigned to each record as it is entered and never again repeated. This ensures that each entered record will have a unique key.

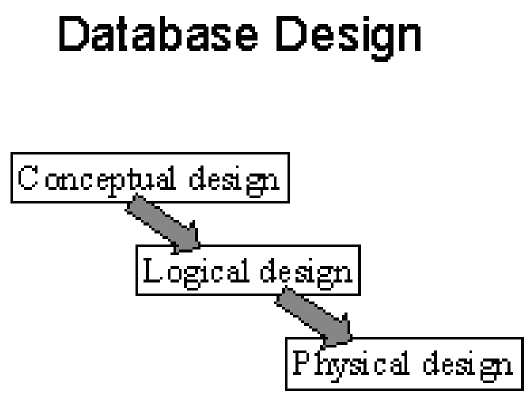
Implementing rules and constraints. In this step, the fields in the database are still fairly amorphous. Defining the fields as text or numeric and getting a rough feel for the types of data that the client needs to store has narrowed them down, but there is room for further refinement. Rules and constraints typically lead to cleaner data entry and thus better information when using the data. Business rules and constraints limit the format that data can take or the ways that data tables can be related to other data tables.

Some of these constraints are imposed by the nature of the data itself; social security numbers are always in the same nine-digit format. This type of constraint is normally implemented to make sure that data is complete and accurate. In other cases, the situation itself explicitly constrains the data. The possible values for the data are usually checked against a list or the choice of values is otherwise constrained. This type of constraint is usually easy to implement and easy to change.

Creating views and reports. Now that the data design is essentially complete, the penultimate step is to create the specifications that help turn the data into useful information in the form of a report or view of the data. *Views* are simply collections of the data available in the database combined and made accessible in one place. It could be as simple as a subset of an existing data table or as complicated as a collection of multiple tables joined on particular set of criteria. Reports on the other hand, are typically snapshots of the database at a particular point in time.

Implementing the design in software. All of the work to this point has been accomplished without explicitly worrying about the details of the program being used to produce the database. In fact, the design should only exist as diagrams and notes on paper. This is especially important at a later point when you or someone else need to update the database or port it to another package. Now it's time to boot the computer and get started.

Database Design



Conceptual Design

Now we will start with the actual design, the first stage of database designing.

In the conceptual design stage, data model is used to create an abstract database structure that represents the real world scenario.

The Conceptual Design

- Complete understanding of database structure, semantics, constraints, relationships etc
- DBMS independent
- Stable description
- Database users and application users views; aids their understanding
- Communication with users
- True representation of real world

The different steps in the conceptual design are as follows

1. Data Analysis and requirements definition
2. Data Modelling and normalization

Data Analysis and requirements definition

In this step the data item and their characteristics are determined. The data items that are required for successful information processing and decision making are identified and their characteristics are recorded. Questions like what kind of information is needed, what outputs (reports and queries) should the system generate who will use the information, how and for what purpose it will be used, what are the sources of the information, etc ; will be answered in this stage.

Data Modelling and normalization

In this step the database designer creates a data model of the system. The business contains entities and relationships. Each entities will have attributes. In this stage the business entities and relationships are transformed in to a data model usually an E-R Model, using E-R Diagrams. Now many designers have started using data modeling using UML (Unified Modeling Language) instead of E-R diagrams. Once the data model is created then the data will be available in a structured form.

All objects (Entities, relations and so on) are defined in a data dictionary and the data is normalized. During he process the designer will group the data items, define the tables, identify the primary keys, define the relationships (One to One, One to Many and many to Many), Create the data model, normalize the data model and so on. Once the data model is created it is verified against the proposed system in order to ascertain that the proposed model is capable of supporting the real world system. So the data model is tested to find out whether the model can perform various database operations and whether the data model takes care of the issue of the data security, integrity, and concurrency and so on.

Review Questions

1. How will you go about in planning the database?
2. Explain the Database Development Life cycle?
3. Explain the conceptual design?

References

http://www.microsoft-accesssolutions.co.uk

Date, C, J, Introduction to Database Systems, 7th edition

Leon, Alexis and Leon, Mathews, Database Management Systems, LeonTECHWorld.