|  |  |  |
| --- | --- | --- |
|  | **Database design principles** |  |
|  | A properly designed database provides you with access to up-to-date, accurate information. Because a correct design is essential to achieving your goals in working with a database, investing the time required to learn the principles of good design makes sense. |  |
|  | 1. Some database terms to know 2. What is good database design? 3. The design process 4. Determining the purpose of your database 5. Finding and organizing the required information 6. Dividing the information into tables 7. Turning information items into columns 8. Specifying primary keys 9. Creating the table relationships 10. Refining the design 11. Applying the normalization rules |  |
|  | **Database design check** |  |
|  | Here are a few things to check for:   1. Did you forget any columns? If so, does the information belong in the existing tables? If it is information about something else, you may need to create another table. Create a column for every information item you need to track. If the information can’t be calculated from other columns, it is likely that you will need a new column for it. 2. Are any columns unnecessary because they can be calculated from existing fields? If an information item can be calculated from other existing columns — a discounted price calculated from the retail price, for example — it is usually better to do just that, and avoid creating new column. 3. Are you repeatedly entering duplicate information in one of your tables? If so, you probably need to divide the table into two tables that have a one-to-many relationship. 4. Do you have tables with many fields, a limited number of records, and many empty fields in individual records? If so, think about redesigning the table so it has fewer fields and more records. 5. Has each information item been broken into its smallest useful parts? If you need to report, sort, search, or calculate on an item of information, put that item in its own column. 6. Does each column contain a fact about the table's subject? If a column does not contain information about the table's subject, it belongs in a different table. 7. Are all relationships between tables represented, either by common fields or by a third table? One-to-one and one-to- many relationships require common columns. Many-to-many relationships require a third table. |  |
|  | 1. **The design process** 2. The design process consists of the following steps: 3. **Determine the purpose of your database.** This helps prepare you for the remaining steps. 4. **Find and organize the information required.** Gather all of the types of information you might want to record in the database, such as product name and order number. 5. **Divide the information into tables.** Divide your information items into major entities or subjects, such as Products or Orders. Each subject then becomes a table. 6. **Turn information items into columns.** Decide what information you want to store in each table. Each item becomes a field, and is displayed as a column in the table. For example, an Employees table might include fields such as Last Name and Hire Date. 7. **Specify primary keys.** Choose each table’s primary key. The primary key is a column that is used to uniquely identify each row. An example might be Product ID or Order ID. 8. **Set up the table relationships.** Look at each table and decide how the data in one table is related to the data in other tables. Add fields to tables or create new tables to clarify the relationships, as necessary. 9. **Refine your design.** Analyze your design for errors. Create the tables and add a few records of sample data. See if you can get the results you want from your tables. Make adjustments to the design, as needed. 10. **Apply the normalization rules.** Apply the data normalization rules to see if your tables are structured correctly. Make adjustments to the tables, as needed. |  |
|  | **Types of Data Models in DBMS** |  |
|  | **Types of Data Models**: There are mainly three different types of data models: conceptual data models, logical data models, and physical data models, and each one has a specific purpose. The data models are used to represent the data and how it is stored in the database and to set the relationship between data items.   1. **Conceptual Data Model:** This Data Model defines **WHAT** the system contains. This model is typically created by Business stakeholders and Data Architects. The purpose is to organize, scope and define business concepts and rules. 2. **Logical Data Model:** Defines **HOW** the system should be implemented regardless of the DBMS. This model is typically created by Data Architects and Business Analysts. The purpose is to developed technical map of rules and data structures. 3. **Physical Data Model**: This Data Model describes **HOW** the system will be implemented using a specific DBMS system. This model is typically created by DBA and developers. The purpose is actual implementation of the database. |  |
|  | **Conceptual Data Model** |  |
|  | A **Conceptual Data Model** is an organized view of database concepts and their relationships. The purpose of creating a conceptual data model is to establish entities, their attributes, and relationships. In this data modeling level, there is hardly any detail available on the actual database structure. Business stakeholders and data architects typically create a conceptual data model.  The 3 basic tenants of Conceptual Data Model are   * **Entity**: A real-world thing * **Attribute**: Characteristics or properties of an entity * **Relationship**: Dependency or association between two entities   Data model example:   * Customer and Product are two entities. Customer number and name are attributes of the Customer entity * Product name and price are attributes of product entity * Sale is the relationship between the customer and product |  |
|  | **Logical Data Model** |  |
|  | The Logical Data Model is used to define the structure of data elements and to set relationships between them. The logical data model adds further information to the conceptual data model elements. The advantage of using a Logical data model is to provide a foundation to form the base for the Physical model. However, the modeling structure remains generic.  A logical data model serves to define how a system has to be implemented regardless of the database management system being used. Data architects and business analysts are usually the creators of a logical data model. The goal of creating a logical data model is to develop a highly technical map of underlying rules and data structures. Components of a Logical Data Model. A logical data model has three main components:  1. Entities: Each entity represents a set of things, persons, or concepts relevant to a business 2. Relationships: Every relationship represents an association between two of the above entities 3. Attributes: Each attribute is a descriptive piece, characteristic or any other information that is useful to further describe an entity 4. Dividing the information into tables 5. Turning information items into columns 6. Specifying primary keys 7. Creating the table relationships 8. Applying the normalization rules |  |
|  | **Physical Design Structures** |  |
|  | During the physical design process, you translate the expected schemas into actual database structures. At this time, you have to map:   1. Entities to tables 2. Relationships to foreign key constraints 3. Attributes to columns 4. Primary unique identifiers to primary key constraints 5. Unique identifiers to unique key constraints |  |
|  | Once you have converted your logical design to a physical one, you will need to create some or all of the following structures:   1. [Tablespaces](https://docs.oracle.com/cd/B10501_01/server.920/a96520/physical.htm" \l "97631)  1. [Tables and Partitioned Tables](https://docs.oracle.com/cd/B10501_01/server.920/a96520/physical.htm" \l "97705)  1. [Views](https://docs.oracle.com/cd/B10501_01/server.920/a96520/physical.htm" \l "98127)  1. [Integrity Constraints](https://docs.oracle.com/cd/B10501_01/server.920/a96520/physical.htm" \l "97635)  1. [Dimensions](https://docs.oracle.com/cd/B10501_01/server.920/a96520/physical.htm" \l "98005)   Some of these structures require disk space. Others exist only in the data dictionary. Additionally, the following structures may be created for performance improvement:   1. [Indexes and Partitioned Indexes](https://docs.oracle.com/cd/B10501_01/server.920/a96520/physical.htm" \l "98157)  1. [Materialized Views](https://docs.oracle.com/cd/B10501_01/server.920/a96520/physical.htm" \l "97989) |  |
|  | **Physical ER Diagram** |  |
|  | It happens as the diagram is modified. The issues with the model fall in one of three categories: errors, warnings, and hints, in order of decreasing severity.  **Why use ER Diagrams?**   1. Helps you to define terms related to entity relationship modeling 2. Provide a preview of how all your tables should connect, what fields are going to be on each table 3. Helps to describe entities, attributes, relationships 4. ER diagrams are translatable into relational tables which allows you to build databases quickly 5. ER diagrams can be used by database designers as a blueprint for implementing data in specific software applications 6. The database designer gains a better understanding of the information to be contained in the database with the help of ERP diagram 7. ERD Diagram allows you to communicate with the logical structure of the database to users |  |
|  | Fix Issues with the Physical ER Diagram  The results of the validation may identify issues which need to be fixed. Some of the most common issues are:   1. Missing foreign keys where entity relationships have been defined. 2. Missing primary keys from tables. 3. Unsupported data types for the selected database. |  |