Designing a Database

by Sophia

WHAT'S COVERED

Recall that a database is an organized collection of related information of which there are two types: flat file and relational. In a relational database, tables are connected, thereby simplifying the process of searching for and retrieving data. Because a relational database maintains a separate table for certain pieces of data, the possibility of errors or redundant information in the database is eliminated. In this tutorial, we will explore the mechanisms employed by relational databases in order to connect its tables, as well as the ways in which data is represented.

Our discussion will break down as follows:

1. Primary Key/Foreign Key

In a relational database, all the tables are related by one or more fields, so that it is possible to connect all the tables in the database through the field(s) they have in common. To connect tables in a relational database, one of the fields is identified as a primary key. A **primary key** is the unique identifier for each record in the table. A primary key must contain a unique value for each row of data. Additionally, a **foreign key** is a field in one table that links to the primary key in another table. To help you understand these terms further, let's walk through the process of designing a database.

EXAMPLE Suppose a university wants to create an information system to track participation in student clubs. After interviewing several people, the design team learns that the goal of implementing the system is to give better insight into how the university funds clubs. This will be accomplished by tracking how many members each club has and how active the clubs are. From this, the team decides that the system must keep track of the clubs, their members, and their events. Using this information, the design team determines that the following tables need to be created:

- **Clubs**: this will track the club name, club president, and a description of the club.
- **Students**: student name, e-mail, and year of birth.
- **Memberships**: this table will correlate students with clubs, allowing us to have any given student join multiple clubs.
- **Events**: this table will track when the clubs meet and how many students showed up.

Now that the design team has determined which tables to create, they need to define the specific information that each table will hold. This requires identifying the fields that will be in each table. For example, Club Name would be one of the fields in the Clubs table. First Name and Last Name would be fields in the Students table. Finally, since this will be a relational database, every table should have a field in common with at least one other table (in other words, they should have a relationship with each other). In order to properly create this relationship, a primary key must be selected for each table. This key is a unique identifier for each record in the table. For example, in the Students table, it might be possible to use a student's last name as a way to uniquely identify him or her. However, it is more than likely that some students will share a last name (like Rodriguez, Smith, or Lee), so a different field should be selected. A student's e-mail address might be a good choice for a primary key, since e-mail addresses are unique. However, a primary key cannot change, so this would mean that if students changed their e-mail address we would have to remove them from the database and then re-insert them — not an attractive proposition. Our solution is to create a value for each student — a user ID — that will act as a primary key. We will also do this for each of the student clubs. This solution is quite common and is the reason you have so many user IDs!

BIG IDEA

A relational database's primary key can never be changed. It should, therefore, be a unique identifier for each record. A common primary key is a user ID, student ID number, or customer ID number, as these types of identifiers are unique and rarely change.

TERMS TO KNOW

Primary Key
Unique identifier for each record in the table.

Foreign Key
A field in one table that links to the primary key in another table.

2. Data Types

When defining the fields in a database table, we must give each field a data type. A**data type** is a classification of the type of data that a field will hold. For example, the field Birth Year is a year, so it will be a number, while First Name will be text. Most modern databases allow for several different data types to be stored. There are two important reasons why we must properly define the data type of a field. First, a data type tells the database what functions can be performed with the data. For example, if we wish to perform mathematical functions with one of the fields, we must be sure to tell the database that the field is a number data type. So, if we have, say, a field storing birth year, we can subtract the number stored in that field from the current year to get age. The second important reason to define data type is so that the proper amount of storage space is allocated for our data. For example, if the First Name field is defined as a text (50) data type, this means 50 characters are allocated for each first name we want to store. However, even if the first name is only five characters long, 50 characters (bytes) will be allocated. While this may not seem like a big deal, if our table ends up holding 50,000 names, we are allocating 50 * 50,000 = 2,500,000 bytes for storage of these values. It may be prudent to reduce the size of the field so we do not waste storage space.

Some common data types and examples are listed here:

Data Type	Description	Example
	Used to store non-numeric	

Text	data that is brief, generally under 256 characters. The database designer can identify the maximum length of the text.	“Hello world”
Number	Used to store numbers. There are usually a few different number types that can be selected, depending on how large the largest number will be.	1 -4 7.3
Yes/No	A special form of the number data type that is (usually) one byte long, with a 0 for “No” or “False” and a 1 for “Yes” or “True.”	0 for “No” or “False” 1 for “Yes” or “True”
Date/Time	A special form of the number data type that can be interpreted as a number or a time.	9-6-17 10:04 AM
Currency	A special form of the number data type that formats all values with a currency indicator and two decimal places.	$45.00 £38.82
Paragraph Text	Used to store text longer than 256 characters.	Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur.
Object	Used to store data that cannot be entered via keyboard, such as an image or a music file.	Music file Image file

TERM TO KNOW

Data Type
A classification of the type of data that a field will hold.

SUMMARY

In this tutorial we took a closer look at the **relational database**, and how utilizing a primary key can connect **information** held in one table to the information held in another table. We also explored**data**

types, and the importance of specifying the type of data a field will hold, as this will communicate to the **database** what **type** of operations can be performed on the data.

Source: Derived from Chapter 4 of “Information Systems for Business and Beyond” by David T. Bourgeois. Some sections removed for brevity.
https://www.saylor.org/site/textbooks/Information%20Systems%20for%20Business%20and%20Beyond/Textbook.html

TERMS TO KNOW

Data Type
A classification of the type of data that a field will hold.

Foreign Key
Field or group of fields in one table that uniquely identifies a row of another table or the same table.

Primary Key
Unique identifier for each record in the table.