Normalization.

Objectives

By the end of this chapter, you should be able to:

- Explain what normalization is
- Define first, second and third normal form
- Identify where normalization can be done

Introduction

When building applications and designing database systems, it's important to keep in mind how your tables are structured. It's easy to keep adding more and more columns to tables as your data grows, but it's important to make sure your tables don't have bloat. One way to do this is to try to normalize your tables. Normalization is a process of organizing the data in database to avoid data redundancy, and anomalies around insertion, updating and deleting. You can read more about these anomalies here

First Normal Form

As per the rule of first normal form, an attribute (column) of a table cannot hold multiple values. It should hold only atomic values. What do we mean by atmoic? The idea here is that each value in a table should not be able to be subdivided. Let's see what we mean with a classic example

id	name	phone
1	Billy	123-321-1221, 332-123-9400
2	Jenny	800-867-5309
3	Adam	971-121-2121, 191-121-2121

The problem here is that we have a column `phone` which is *not* atomic! This representation of telephone numbers is not in first normal form: our columns contain non-atomic values, and they contain more than one of them. One option is to add more columns, but this would create what is called a "repeating group", which are columns of similar name. The best option here is to create a new table and an association between the original table and a table of phone numbers.

Second Normal Form (2NF)

A table is said to be in 2NF if both the following conditions hold:

- Table is in 1NF (First normal form)
- No non-prime attribute is dependent on the proper subset of any candidate key of table.

It's important to note that this problem only happens when *compound keys* exist. A compound key is a key that consists of 2 or more attributes that uniquely identify an entity occurrence. Each attribute that

makes up the compound key is a simple key in its own right. If this does not exist, we can move straight to 3NF. A solution to this is to separate this single table into multiple tables to reduce the redundancy.

Third Normal Form (3NF)

A table design is said to be in 3NF if both the following conditions hold:

- Table is in 2NF
- All non-key attributes are not dependent on other non-key attributes.

With 3NF, it's similar to 2NF except we are dealing with one primary key. Imagine we have a table of customers and want to have a sales representative for each customer. We might think of storing that information in the same table (with some information about the sales rep like their name), but this will violate 3NF as there are non key attributes (like the sales_rep_number) that are not dependent on the primary key.

The way to fix this would be to move the data related to the sales rep to another table and make it a foreign key inside of the customers table.

Additional Forms

There does exist a 4th, 5th, and 6th normal form, but they are used far less frequently than the three above. You can read more about them here.