# Database Normalisation Checklist

Thanks for reading my normalisation article and downloading this checklist!

When you're looking to normalise a database, you aim to take to third normal form (as outlined in my article).

Getting to third normal form takes some practice.

I've created this checklist to help you realise if you've made it to third normal form and have a well-designed database.

Use this on any database you're creating (or changes to an existing database) to ensure you have a properly normalised database.

| | |
|---|---|
| **Entities** | |
| | Have you got a table for every entity or object you need to store data about?<br><br>*This usually comes from a description of what your database is meant to do, and represents the nouns (names of things).* |
| | Do each of these tables have a column or set of columns that identifies each row uniquely (and has been labelled as a primary key)?<br><br>*This can either be a column that belongs to the data (e.g. a social security number), a combination of columns, or a new column (e.g. customer_id)* |
| | Are the names of your tables following either a singular or plural naming convention?<br><br>*Either name all of your tables singular (customer, sale, product) or plural (customers, sales, products), to avoid confusion.* |
| **Attributes** | |
| | Are all attributes in each table dependent on the primary key?<br><br>*This avoids update anomalies and is part of "second normal form"* |
| | Are you avoiding storing sensitive information in your database (e.g. credit card numbers) if you really don't need to?<br><br>*This is a wider question about your system. There may be a need for you to store this information.* |
| | Are any fields that are not dependent on the primary key, or dependent on other non-primary-key attributes, moved to another table?<br><br>*This is a "transitive functional dependency" and is part of third normal form.* |

| | |
|---|---|
| | If the business or area you're designing the database for decides to change or delete a value in the database, does it only need to be updated or deleted in one place?<br><br>*This avoids update and delete anomalies.* |
| | Have you specified the data types and sizes for each attribute?<br><br>*This includes maximum length for strings, and digits and decimal places for numbers.* |
| **Relationships** | |
| | Are the tables that need to relate to each other, related to each other?<br><br>*Not all tables need a relationship (e.g. a config table)* |
| | Are there any relationships defined as a many-to-many relationship? If so, are they represented using a joining table? |
| | Do the joining tables have meaningful names where appropriate?<br><br>*For example, enrolment instead of student_subject.* |
| | For each relationship, is there a field in one of the tables that represents the primary key field(s) of the other table? This is known as a foreign key.<br><br>*For example, the employee table may have a department_id column.* |
| | Are there any one-to-one relationships? If so, can they be joined into a single table? |
| | Are there two or more tables that form a hierarchy of a certain type of object? If so, can they be converted to a single table with a self-join?<br><br>*E.g. a table for employee instead of separate tables for manager, team leader, and team member.* |