Table Normalization

In short: Normalization is the process of organizing data so that the same information is not repeated unnecessarily and a single change does not cause inconsistencies in multiple places.

1. Starting Point: Everything in One Table

Imagine an RPG game database. We store players, their characters, classes, and items. The simplest approach would be to put everything in a single table:

Player	Character	Class	Items
Tom	Geralt	Witcher	Silver sword, armor
Harry	Yennefer	Mage	Wand, ring
Bart	Ciri	Warrior	Steel sword, armor

Problems arise quickly: the list of items is difficult to search, and with a large inventory, a single cell becomes unwieldy. If we want to manage each item separately, we need to change how the data is stored.

2. First Normal Form (1NF)

Rule: One cell should contain one value, not a list of different items.

Character	Item
Geralt	Silver sword
Geralt	Armor
Ciri	Steel sword
Ciri	Armor

It is now easier to find all characters that have armor or count Geralt's items. This is the first step: every item we want to search for and manage separately is stored in its own record.

3. Second Normal Form (2NF)

Suppose that in the inventory table we store the character ID, character name, item ID, item name, and its weight all at once. Geralt's name will be repeated for each of his items, while the armor's name and weight will be repeated for every character who has it.

Rule: Data should depend on the entire record key, not just on part of it.

If an inventory entry is identified by the pair “character + item”, then the character's name depends only on the character, while the item's name and weight depend only on the item. That is why we separate this information:

CHARACTERS	ITEMS	INVENTORY
1 — Geralt	101 — Silver sword (3)	1 — 101
2 — Ciri	102 — Armor (10)	1 — 102
	103 — Steel sword (4)	2 — 103
		2 — 102

The CHARACTERS table stores character details, the ITEMS table stores item details, and the INVENTORY table stores information about who owns what. A change to the armor's weight is made in just one place.

4. Third Normal Form (3NF)

Information about a class can also be unnecessarily duplicated in the characters table:

Character ID	Name	Class ID	Class	Mana Bonus
1	Geralt	10	Witcher	20
2	Yennefer	20	Mage	100
3	Triss	20	Mage	100

The mana bonus depends on the class, not on a specific character. If it were changed from 100 to 120, we would have to remember to update every mage. It is better to move the class details to a separate table.

CHARACTERS	**KLASY**
1 — Geralt	10 — Witcher — bonus 20
2 — Yennefer — class 20	20 — Mage — bonus 100
3 — Triss — class 20	

Rule: The characters table describes characters, while the classes table describes classes. Information about another entity should be stored in the table that describes that entity.

5. How Are These Tables Related?

One character can have many items, and the same type of item can belong to many characters. This is a many-to-many relationship. The INVENTORY table therefore links character IDs to item IDs without duplicating names and descriptions.

Table	What Does It Store?
CHARACTERS	Character data, e.g. name and class ID
KLASY	Class data, e.g. name and mana bonus
ITEMS	Item data, e.g. name and weight
INVENTORY	Relationships: which character has which item

6. Why Do This?

A well-organized database makes everyday changes in the game easier:

- **Changing stats:** you update the weight of the steel sword in one place.
- **Adding a new character:** you assign an existing class instead of copying all the information about that class.
- **Deleting a character:** you can remove its inventory relationships without deleting the definitions of items used by others.

Quick Reference

Normal Form	Simplest Explanation	In-Game Example
1NF	Do not put a list in a single cell.	Each inventory item has its own row.
2NF	Information depends on the entire key.	The character's name does not depend on the character + item pair.
3NF	Do not store details about another entity in this table.	A mage's bonus belongs to the class, not to each character.

Most importantly: Normalization is not about creating as many tables as possible. It is about giving each piece of information its proper place, avoiding unnecessary duplication, and keeping the data consistent.