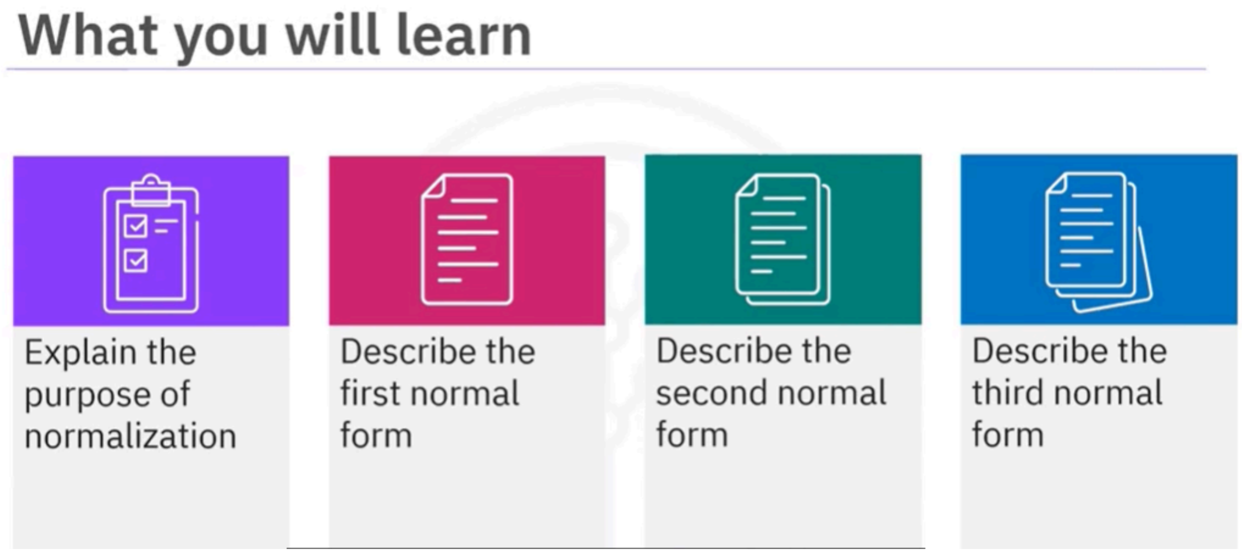
# Normalization

## Normalization: A Simple Explanation

Normalization is like organizing your closet. Imagine you have clothes scattered everywhere, with some items duplicated in different places. This can make it hard to find what you need and can lead to confusion. In the world of databases, normalization helps to tidy up data by reducing duplication and ensuring that each piece of information is stored in the right place. This way, when you need to update something, you only have to do it once, making everything more efficient and consistent.

To illustrate, think of a bookshop's inventory. If a book is listed multiple times for different formats (like paperback and hardback) in the same row, it creates clutter. By normalizing the data, you can create separate entries for each format, ensuring that each piece of information is unique and easy to manage. This organization not only speeds up transactions but also helps maintain the integrity of the data, just like a well-organized closet makes it easier to find your favorite outfit!



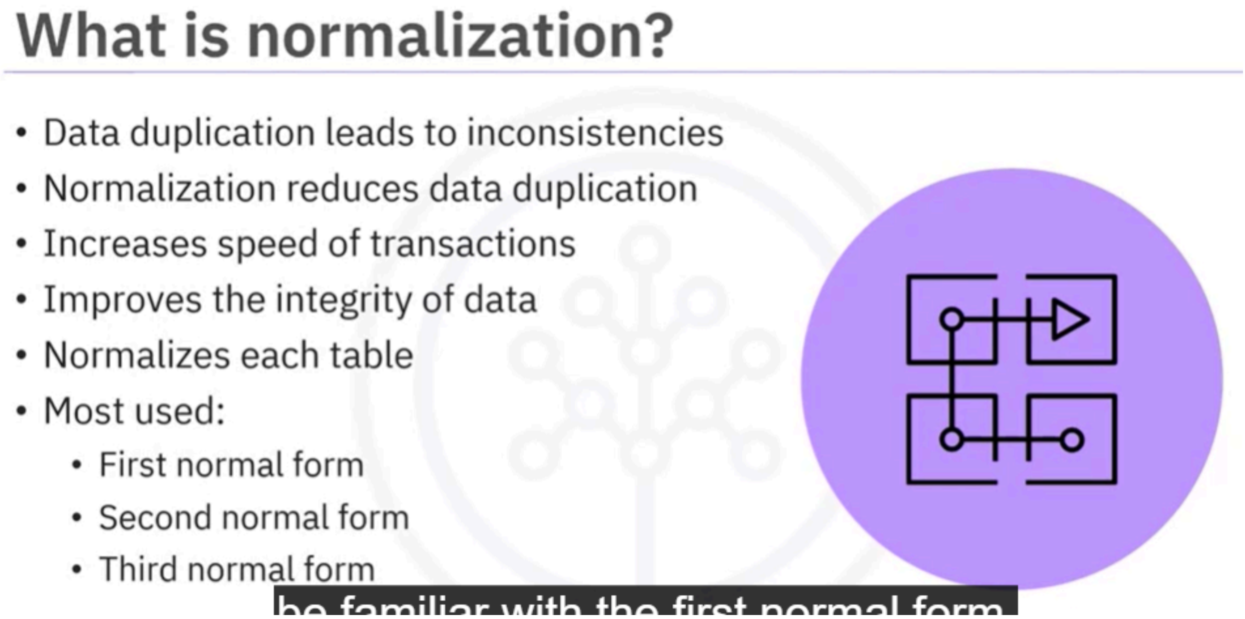
What you will learn

Explain the purpose of normalization

Describe the first normal form

Describe the second normal form

Describe the third normal form

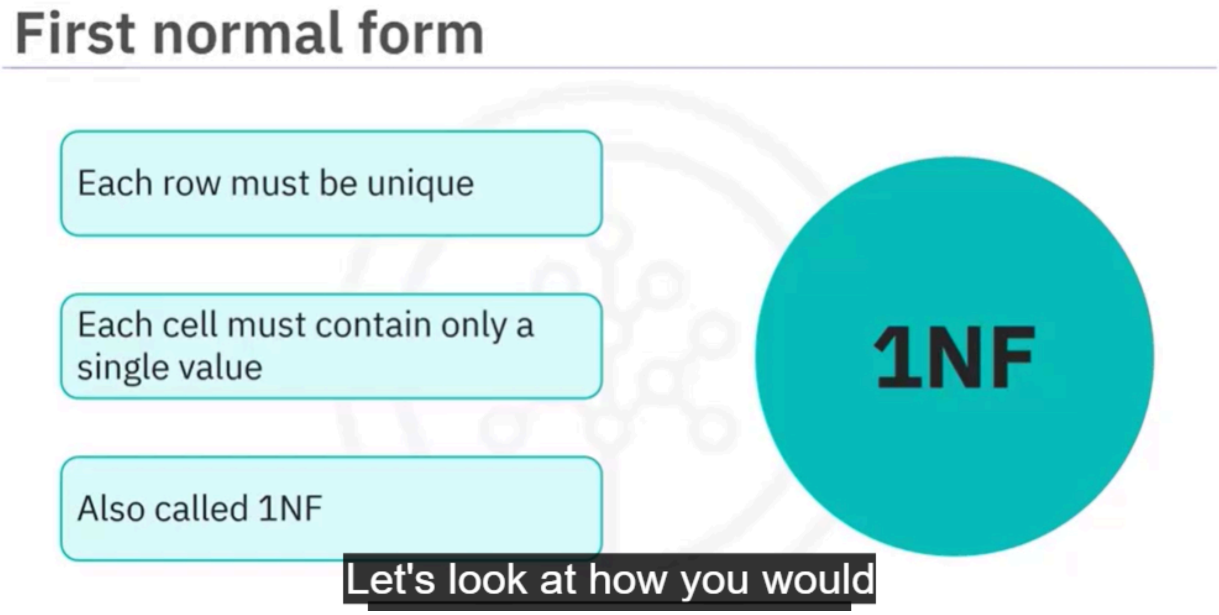


## What is normalization?

- Data duplication leads to inconsistencies
- Normalization reduces data duplication
- Increases speed of transactions
- Improves the integrity of data
- Normalizes each table
- Most used:
  - First normal form
  - Second normal form
  - Third normal form

be familiar with the first normal form

- When you keep records of data such as books in a bookshop, you will inevitably have some inconsistencies and duplicate information. Such duplication can cause extra work and inconsistencies if the data is updated, because you must change it in more than one place. Normalization is the process of organizing your data to reduce redundant data, often by dividing larger tables into multiple relatable tables. Normalization helps speed up transactions because you perform updates, additions, and deletes only once on a normalized database. It also improves data integrity because it reduces the chance of a change being made in one place but not in another. As you begin the normalization process, it's important to recognize that you focus on normalizing each table until you've reached the required normal form level. Normalization usually results in creating more tables, and once all the tables are normalized, you will have a normalized database. There are several forms of normalization, and most data engineers will need to be familiar with the first normal form, second normal form, and third normal form.



## First normal form

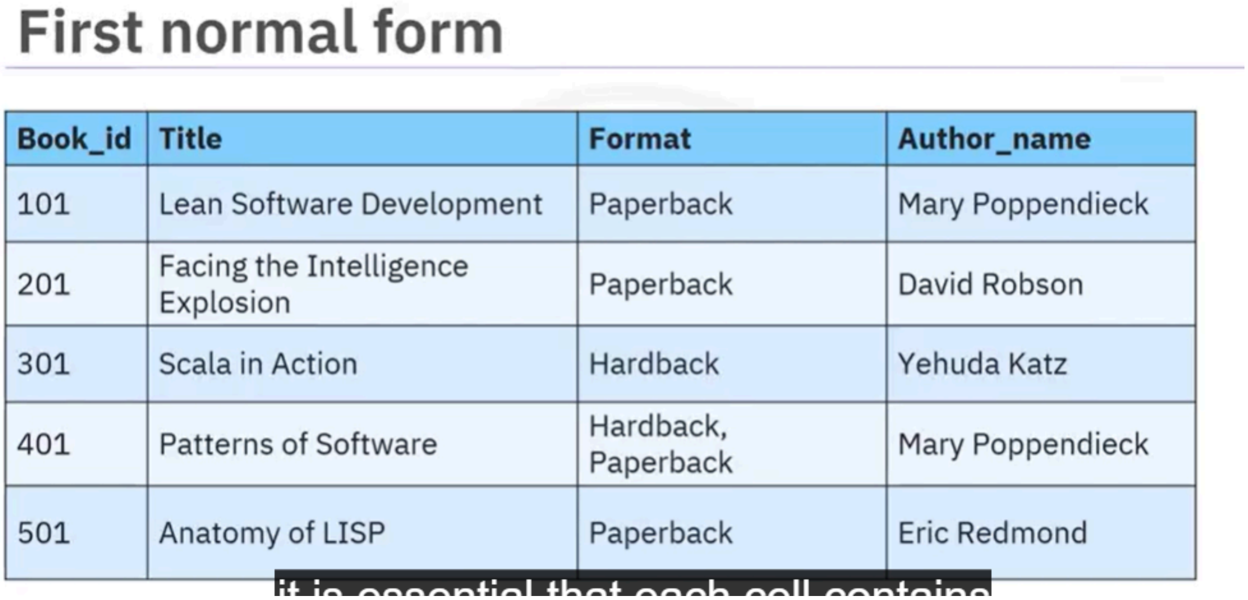
Each row must be unique

Each cell must contain only a single value

Also called 1NF

Let's look at how you would

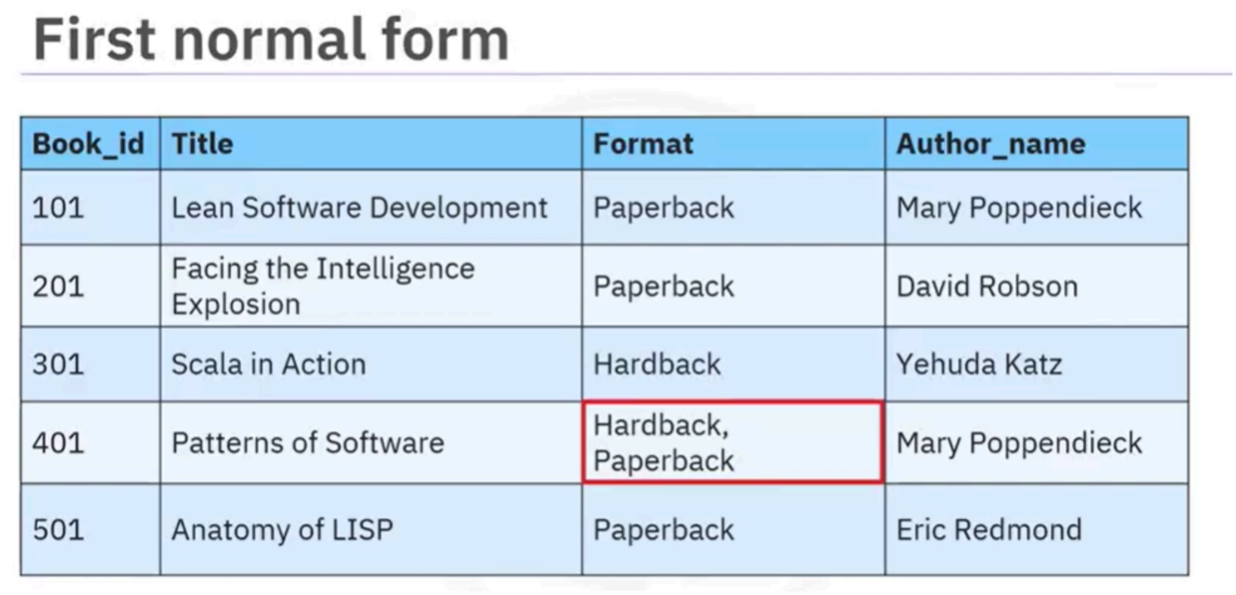
- each row must be unique and each cell must contain only a single value. First normal form is also called 1NF



## First normal form

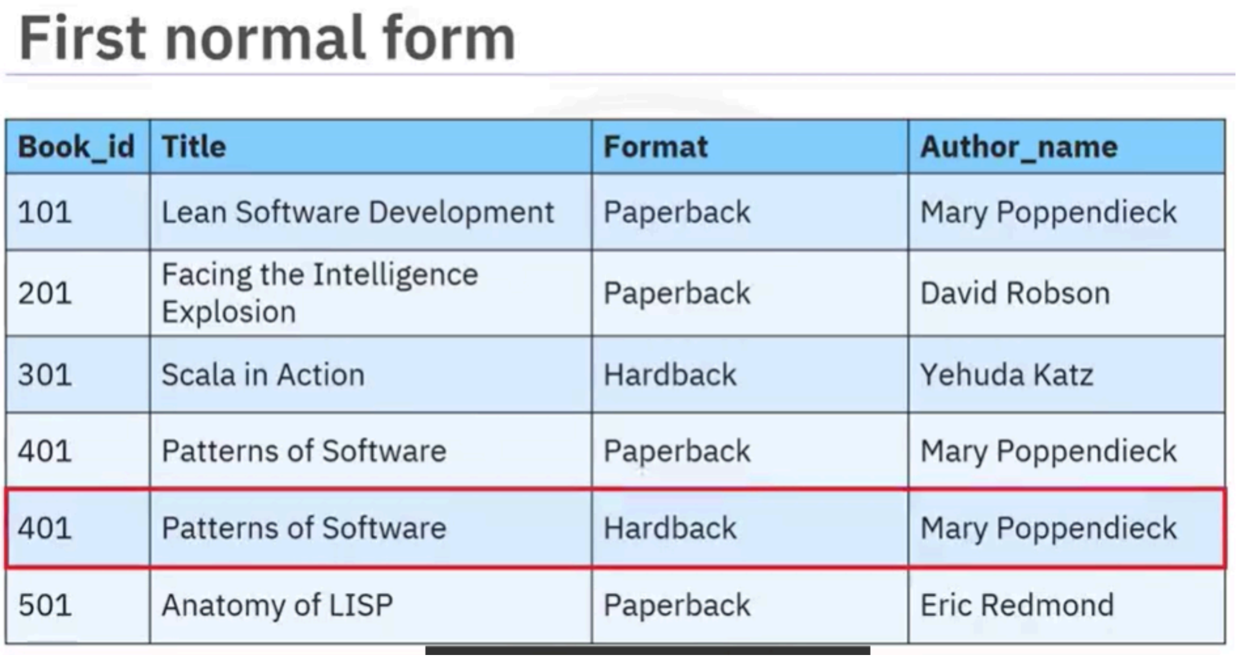
| Book_id | Title | Format | Author_name |
|---|---|---|---|
| 101 | Lean Software Development | Paperback | Mary Poppendieck |
| 201 | Facing the Intelligence Explosion | Paperback | David Robson |
| 301 | Scala in Action | Hardback | Yehuda Katz |
| 401 | Patterns of Software | Hardback, Paperback | Mary Poppendieck |
| 501 | Anatomy of LISP | Paperback | Eric Redmond |

it is essential that each cell contains



## First normal form

| Book_id | Title | Format | Author_name |
|---|---|---|---|
| 101 | Lean Software Development | Paperback | Mary Poppendieck |
| 201 | Facing the Intelligence Explosion | Paperback | David Robson |
| 301 | Scala in Action | Hardback | Yehuda Katz |
| 401 | Patterns of Software | Hardback, Paperback | Mary Poppendieck |
| 501 | Anatomy of LISP | Paperback | Eric Redmond |

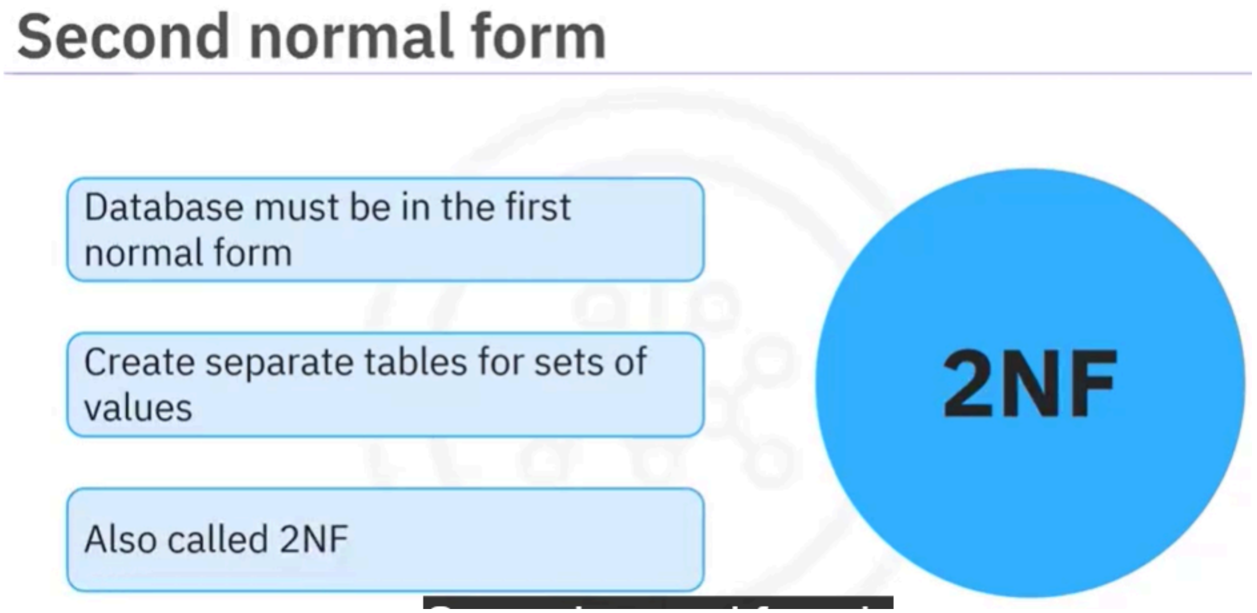


## First normal form

| Book_id | Title | Format | Author_name |
|---|---|---|---|
| 101 | Lean Software Development | Paperback | Mary Poppendieck |
| 201 | Facing the Intelligence Explosion | Paperback | David Robson |
| 301 | Scala in Action | Hardback | Yehuda Katz |
| 401 | Patterns of Software | Paperback | Mary Poppendieck |
| 401 | Patterns of Software | Hardback | Mary Poppendieck |
| 501 | Anatomy of LISP | Paperback | Eric Redmond |

- Let's look at how you would normalize a simple table. In this example, the book table contains some basic information about books, including titles, formats, and authors. To adhere to the requirements of the first normal form, it is essential that each cell contains a singular value rather than a list. In this example, you can observe that all variations of a book's format are recorded within the same cell. To normalize this table, you can add an extra row and split the two formats of patterns and software into their own row. So now you have a row for the paperback version and a row for the hardback version.

Each cell in the table now has only one entry, and so the table is in the first normal form.



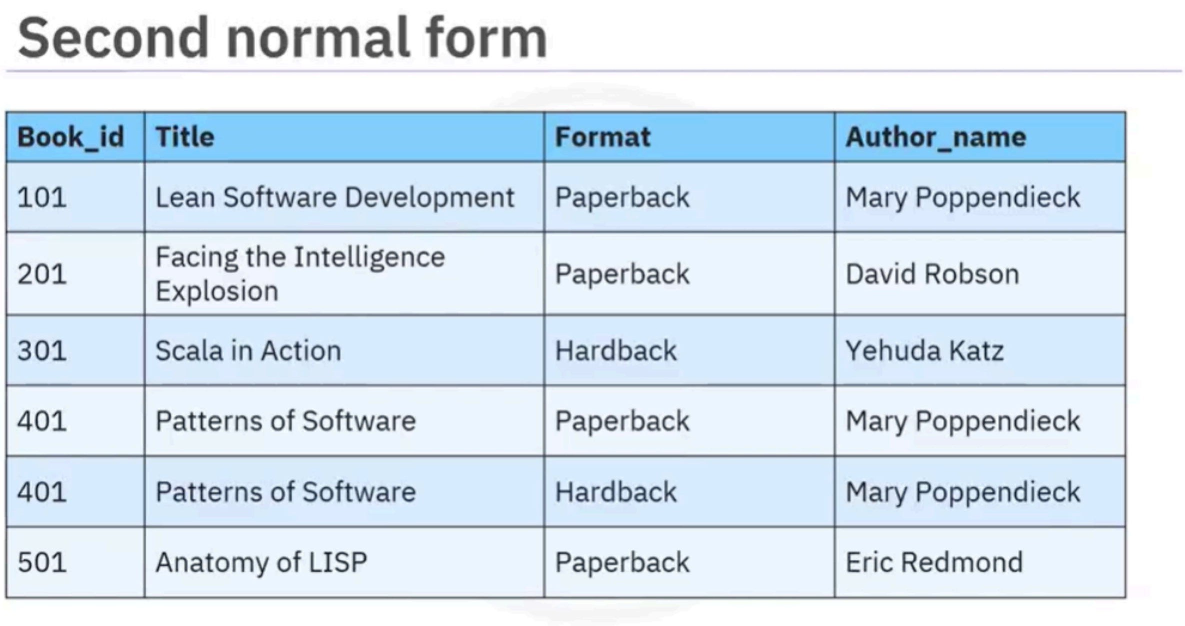
## Second normal form

Database must be in the first normal form

Create separate tables for sets of values

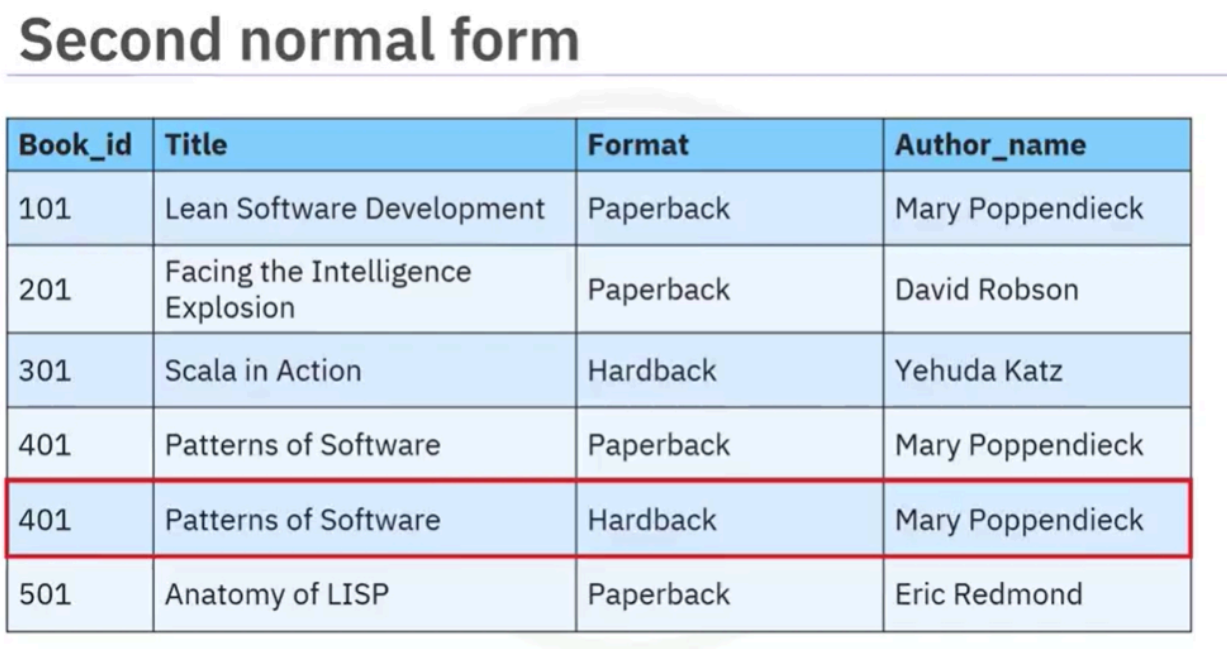
Also called 2NF

- To be in the second normal form, the database must already be in first normal form, which involves ensuring that every row in the table is unique, and that each cell contains only a single value. Second normal form specifies that you should separate groups of values that apply to multiple rows by creating new tables. Second normal form is also referred to as 2NF.



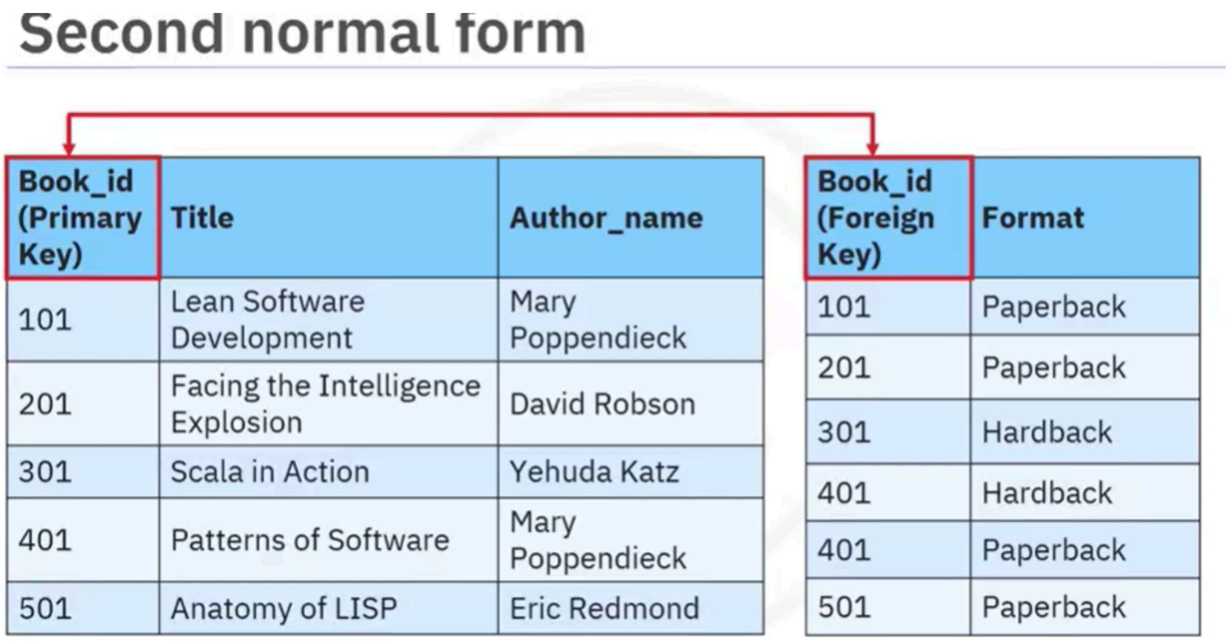
## Second normal form

| Book_id | Title | Format | Author_name |
|---|---|---|---|
| 101 | Lean Software Development | Paperback | Mary Poppendieck |
| 201 | Facing the Intelligence Explosion | Paperback | David Robson |
| 301 | Scala in Action | Hardback | Yehuda Katz |
| 401 | Patterns of Software | Paperback | Mary Poppendieck |
| 401 | Patterns of Software | Hardback | Mary Poppendieck |
| 501 | Anatomy of LISP | Paperback | Eric Redmond |



## Second normal form

| Book_id | Title | Format | Author_name |
|---|---|---|---|
| 101 | Lean Software Development | Paperback | Mary Poppendieck |
| 201 | Facing the Intelligence Explosion | Paperback | David Robson |
| 301 | Scala in Action | Hardback | Yehuda Katz |
| 401 | Patterns of Software | Paperback | Mary Poppendieck |
| 401 | Patterns of Software | Hardback | Mary Poppendieck |
| 501 | Anatomy of LISP | Paperback | Eric Redmond |

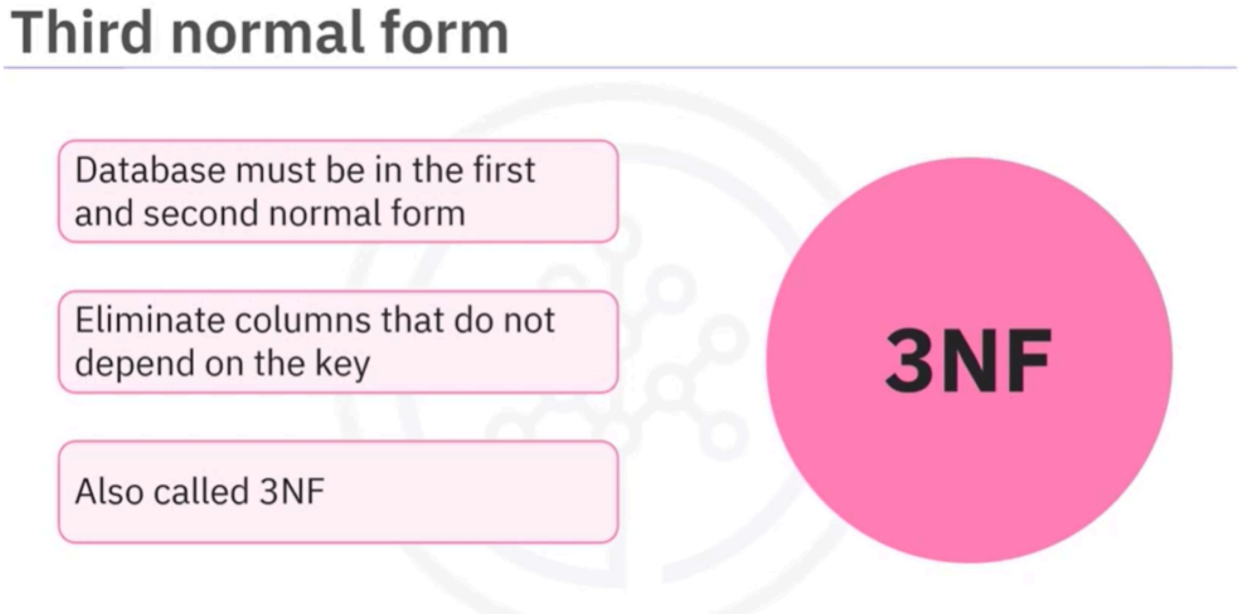


## Second normal form

| Book_id (Primary Key) | Title | Author_name |
|---|---|---|
| 101 | Lean Software Development | Mary Poppendieck |
| 201 | Facing the Intelligence Explosion | David Robson |
| 301 | Scala in Action | Yehuda Katz |
| 401 | Patterns of Software | Mary Poppendieck |
| 501 | Anatomy of LISP | Eric Redmond |

| Book_id (Foreign Key) | Format |
|---|---|
| 101 | Paperback |
| 201 | Paperback |
| 301 | Hardback |
| 401 | Hardback |
| 401 | Paperback |
| 501 | Paperback |

- For clarity, this example shows just a subset of the data in the book table. Book 401 comes in both paperback and hardback format, so in its current form, it must be listed twice, once for each format. In this case, the format column contains values that apply to both rows that reference book 401, so there is some data duplication. To meet the requirements for the second normal form and achieve just one row for book 401. You can split the book table so that the format information for the book is separated from unrelated information such as title and author, each resulting table is in 1NF. To maintain a relationship between the two tables, identify a primary key for one table that will be used as a foreign key in the other. In this example, the book ID is unique to each book, so you can make that the primary key in the book table and include it as a foreign key in the format table. Now you can use it to link between the two tables to find the different formats of each of the unique books



## Third normal form

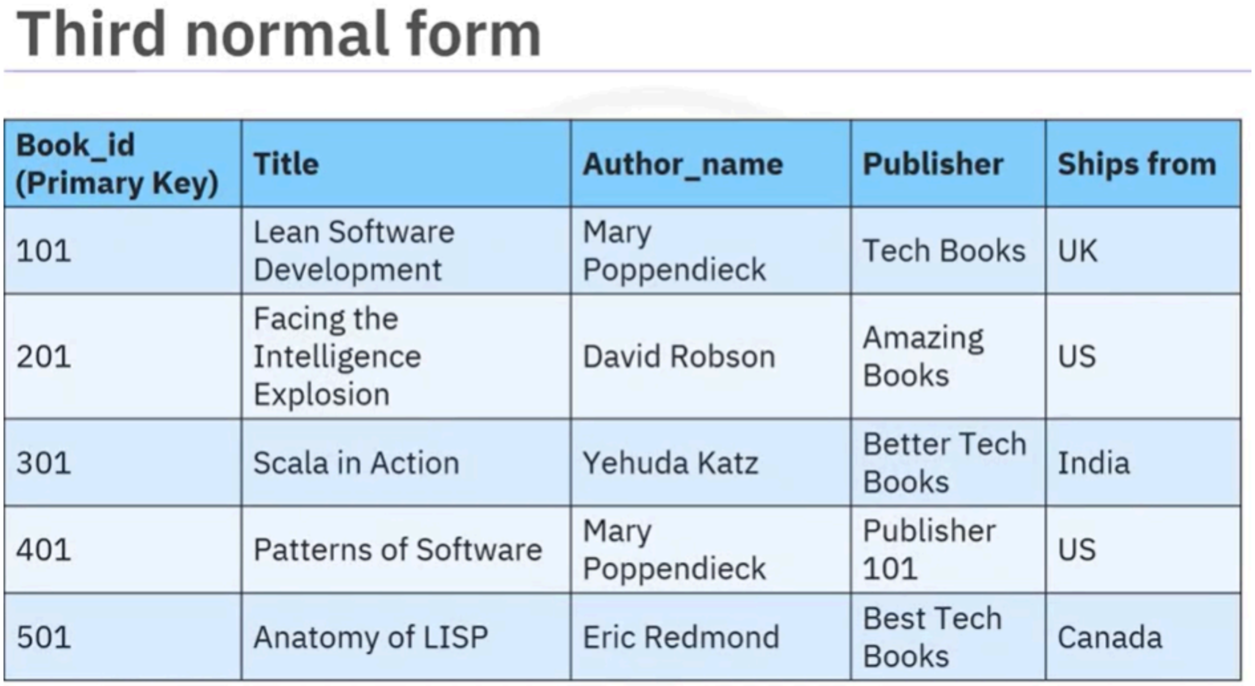
Database must be in the first and second normal form

Eliminate columns that do not depend on the key

Also called 3NF

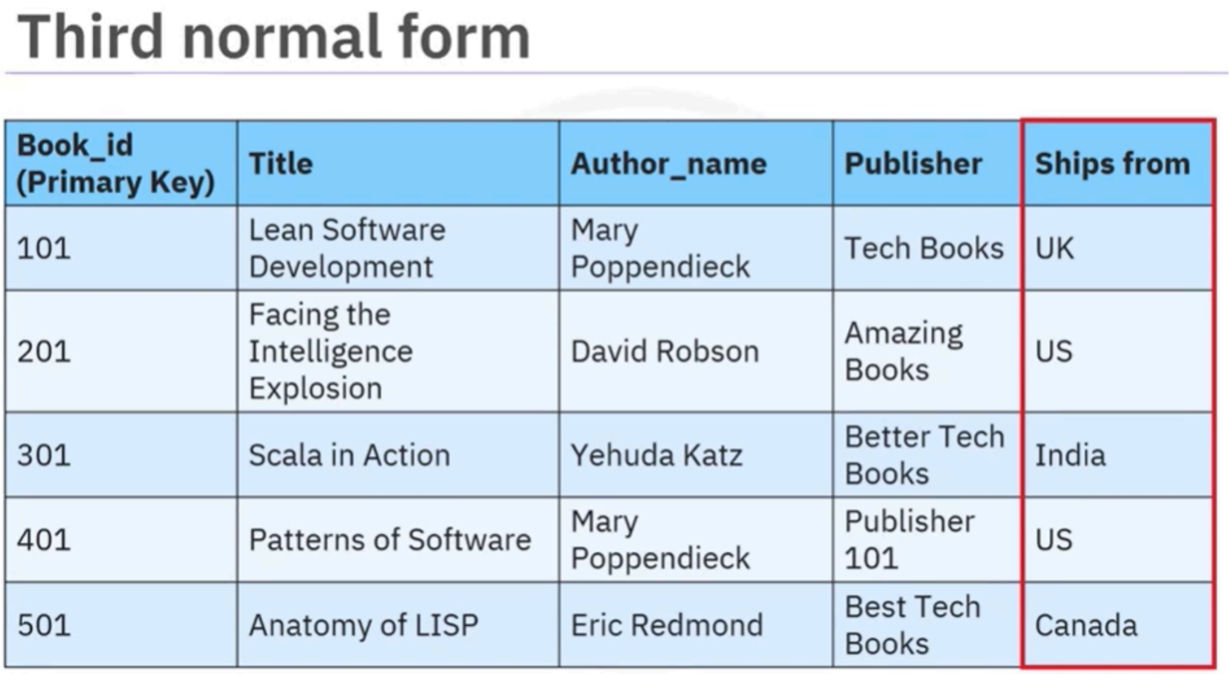
3NF

- The database must already be in the first and second normal forms to meet the requirements for the third normal form. Next, you must eliminate any columns that do not depend on the key. Third normal form is also referred to as 3NF.



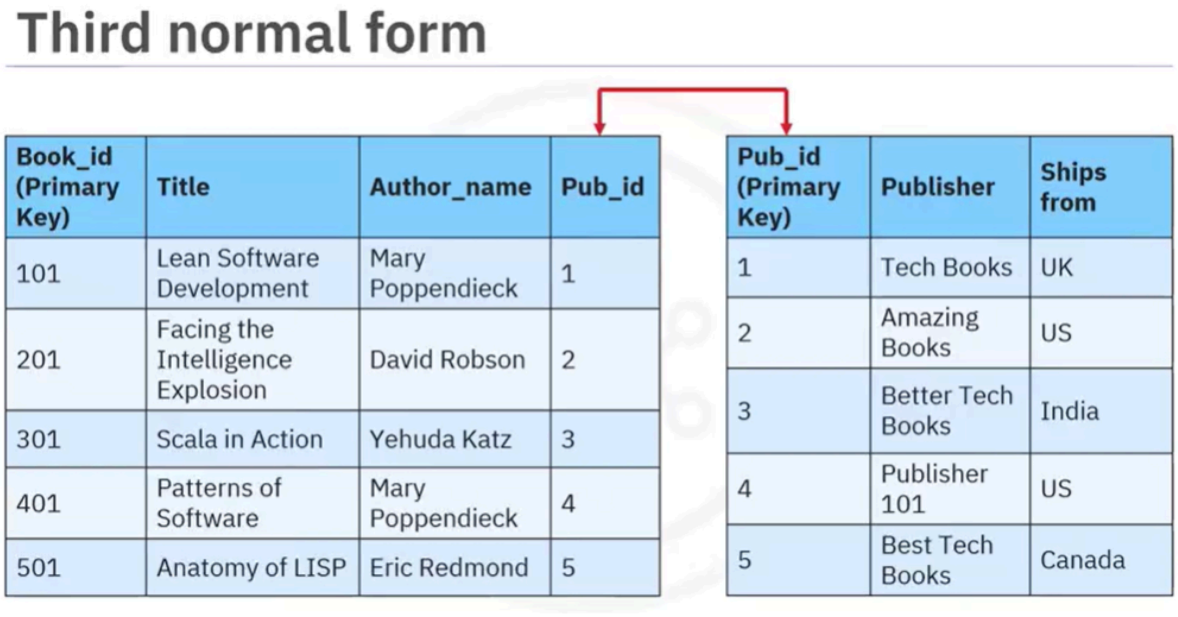
## Third normal form

| Book_id (Primary Key) | Title | Author_name | Publisher | Ships from |
|---|---|---|---|---|
| 101 | Lean Software Development | Mary Poppendieck | Tech Books | UK |
| 201 | Facing the Intelligence Explosion | David Robson | Amazing Books | US |
| 301 | Scala in Action | Yehuda Katz | Better Tech Books | India |
| 401 | Patterns of Software | Mary Poppendieck | Publisher 101 | US |
| 501 | Anatomy of LISP | Eric Redmond | Best Tech Books | Canada |



## Third normal form

| Book_id (Primary Key) | Title | Author_name | Publisher | Ships from |
|---|---|---|---|---|
| 101 | Lean Software Development | Mary Poppendieck | Tech Books | UK |
| 201 | Facing the Intelligence Explosion | David Robson | Amazing Books | US |
| 301 | Scala in Action | Yehuda Katz | Better Tech Books | India |
| 401 | Patterns of Software | Mary Poppendieck | Publisher 101 | US |
| 501 | Anatomy of LISP | Eric Redmond | Best Tech Books | Canada |

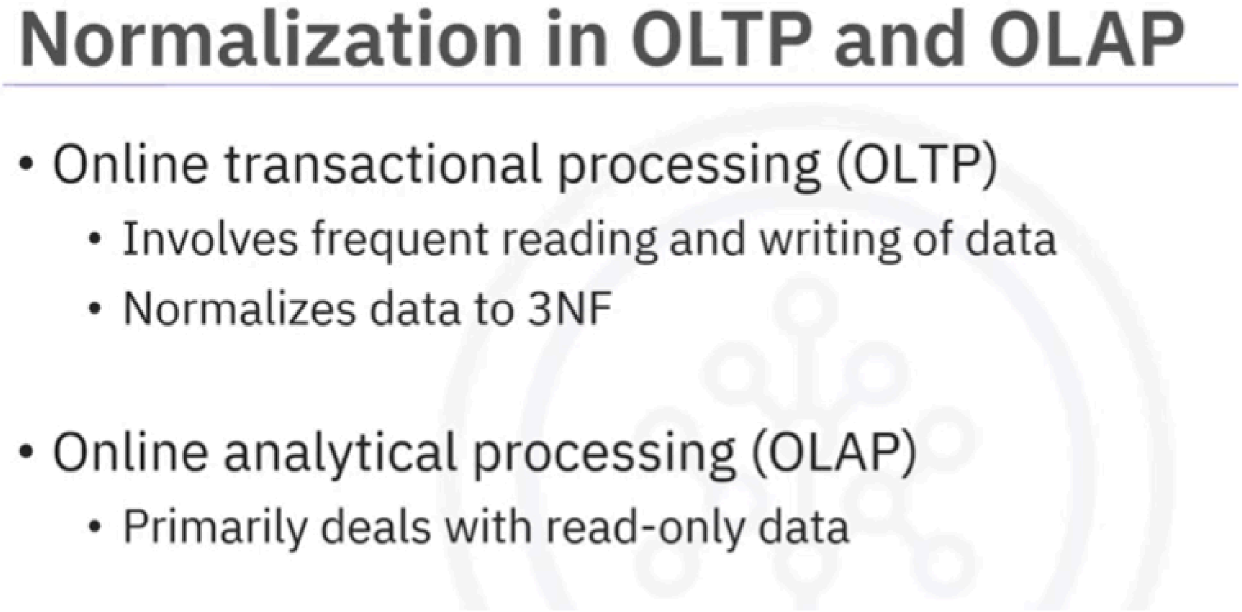


## Third normal form

| Book_id (Primary Key) | Title | Author_name | Pub_id |
|---|---|---|---|
| 101 | Lean Software Development | Mary Poppendieck | 1 |
| 201 | Facing the Intelligence Explosion | David Robson | 2 |
| 301 | Scala in Action | Yehuda Katz | 3 |
| 401 | Patterns of Software | Mary Poppendieck | 4 |
| 501 | Anatomy of LISP | Eric Redmond | 5 |

| Pub_id (Primary Key) | Publisher | Ships from |
|---|---|---|
| 1 | Tech Books | UK |
| 2 | Amazing Books | US |
| 3 | Better Tech Books | India |
| 4 | Publisher 101 | US |
| 5 | Best Tech Books | Canada |

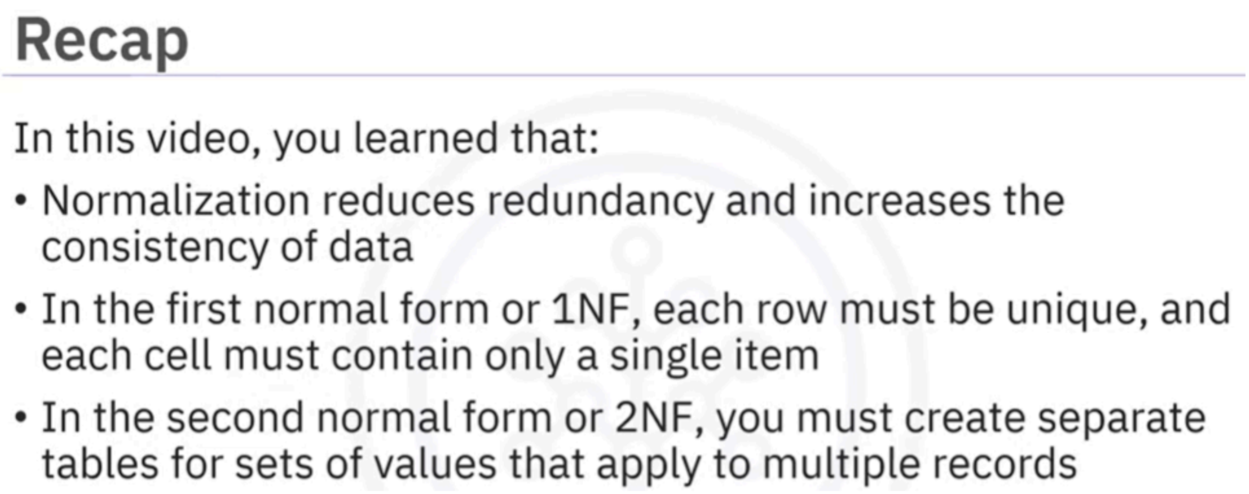
- Let's consider some additional data about books, the publisher, and where the book ships from. Each publisher ships books from warehouses in their own location. So where the book ships from depends on the publisher, not the book ID. Therefore, the book table is not in 3NF because the ships from data does not depend on the primary key. To fulfill the criteria of the third normal form, 3NF, it is necessary to segregate the publisher and ship's details into a dedicated publisher's table. Both tables are now in third normal form, which is as far as most relational databases go. There are also higher normal forms such as Boyce-Codd normal form, or BCNF, which is an extension to the third normal form, as well as fourth and fifth normal forms, which may be needed for specific scenarios.



Normalization in OLTP and OLAP

- Online transactional processing (OLTP)
  - Involves frequent reading and writing of data
  - Normalizes data to 3NF
- Online analytical processing (OLAP)
  - Primarily deals with read-only data

- In transactional systems, OLTP, where data is both read and written frequently, you typically normalize the data to 3NF. OLTP systems need to process and store transactions as well as query transactional data, and normalizing the data to 3NF helps the database to process and store individual transactions efficiently. In analytical OLAP systems, where users primarily read data, databases prioritize read performance over write integrity. Hence, the data may have undergone some denormalization to a lower normal form before being loaded into the analytical system, such as a data warehouse. In data warehousing, data, engineers focus on performance, which can benefit from having fewer tables to process.



## Recap

In this video, you learned that:

- Normalization reduces redundancy and increases the consistency of data
- In the first normal form or 1NF, each row must be unique, and each cell must contain only a single item
- In the second normal form or 2NF, you must create separate tables for sets of values that apply to multiple records