

From Scenario to Database

Find the entities, attributes and relationships, draw the ERD, then normalise the data to 3NF - one question at a time.

CUSTOMER	
CustomerID	PK
CustomerName	
CustomerEmail	

ORDER	
OrderID	PK
OrderDate	
CustomerID	FK

BOOK	
BookID	PK
Title	
Price	

Our scenario: Bookworm Books



THE SCENARIO

Bookworm Books is a small online bookshop.

Customers register with their name and email address.

A customer can place many orders.

Each order has a date and can include several books.

Every book has a title and a price.

For each book on an order, we record how many copies were bought.

ORDER FORM

Order no: O101 **Date:** 01-Oct-2026

Customer: C1 - Aisha Khan

Email: aisha@mail.com

Book	Title	Price	Qty
B1	Python Basics	£20	1
B2	Web Design	£15	2

One order, one customer, several books.

Two parts, one scenario

Part A - Build the ERD

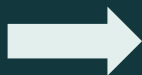
- 1 Find the entities
- 2 List the attributes
- 3 Find the relationships
- 4 Fix the many-to-many
- 5 Draw the ERD

Part B - Normalise the data

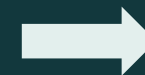
- 1 Spot the unnormalised table (UNF)
- 2 First Normal Form (1NF)
- 3 Second Normal Form (2NF)
- 4 Third Normal Form (3NF)

FOR EVERY STEP

YOUR TURN



think + discuss



ANSWER

+ how to



Build the ERD

Entities, attributes, relationships - then draw the diagram.

Step 1 · Find the entities



THE SCENARIO

Bookworm Books is a small online bookshop.

Customers register with their name and email address.

A customer can place many orders.

Each order has a date and can include several books.

Every book has a title and a price.

For each book on an order, we record how many copies were bought.



Which nouns in the scenario are things we need to store data about?

HINT

An entity is a thing we keep a list of: a person, an object or an event. Ask yourself: will there be many of them?

Step 1 · Find the entities



Customer

A person who registers and buys books



Order

One purchase made by a customer



Book

A title that the shop sells

HOW TO FIND ENTITIES

- 1 Read the scenario and underline the **nouns**.
- 2 Keep the nouns we will have **many of** (many customers, orders, books).
- 3 Drop nouns that are only a **fact about** something else.

NOT ENTITIES

Bookworm Books - the business itself; there is only one.

name, email, date, title, price - these are facts about an entity (attributes, next step).

Step 2 · List the attributes

CUSTOMER	ORDER	BOOK
?	?	?
?	?	?
?	?	?

CLUES IN THE SCENARIO

Customers register with their **name** and **email** address.

Each order has a **date**.

Every book has a **title** and a **price**.

?

What facts (attributes) do we store about each entity?
Which fact could identify each one uniquely?

HINT

If no fact is unique, we can add an ID of our own.

Step 2 · List the attributes

CUSTOMER	
CustomerID	PK
CustomerName	
CustomerEmail	

ORDER	
OrderID	PK
OrderDate	

BOOK	
BookID	PK
Title	
Price	

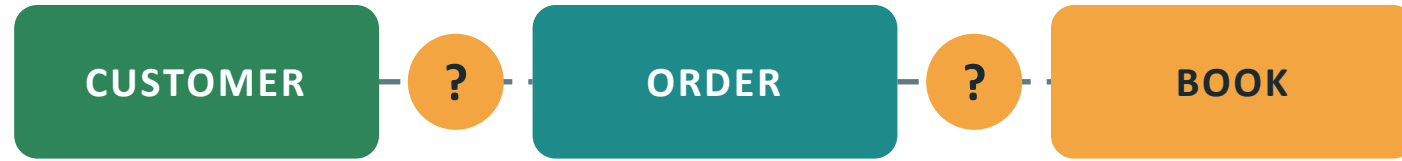
HOW TO LIST ATTRIBUTES

- 1 An **attribute** is one fact about an entity (a future column).
- 2 Pick a **primary key (PK)**: a value that is different for every row.
- 3 If nothing is unique, **add an ID** (CustomerID, OrderID, BookID).

HOLD ON: QUANTITY?

Copies bought is not a fact about a customer, or about a book on its own. We will find its home in Step 4.

Step 3 · Find the relationships



COMPLETE THE SENTENCES

One customer can place ____ order(s).

One order is placed by ____ customer(s).

One order can contain ____ book(s).

One book can appear on ____ order(s).

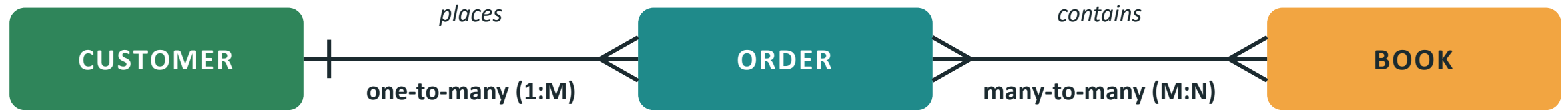
?

How are the entities connected? What is the link, and is it ONE or MANY in each direction?

HINT

Look for the verbs in the scenario: 'place' and 'include'.

Step 3 · Find the relationships



Customer and Order: one-to-many

One customer places **MANY** orders.
Each order is placed by **ONE** customer.

Order and Book: many-to-many

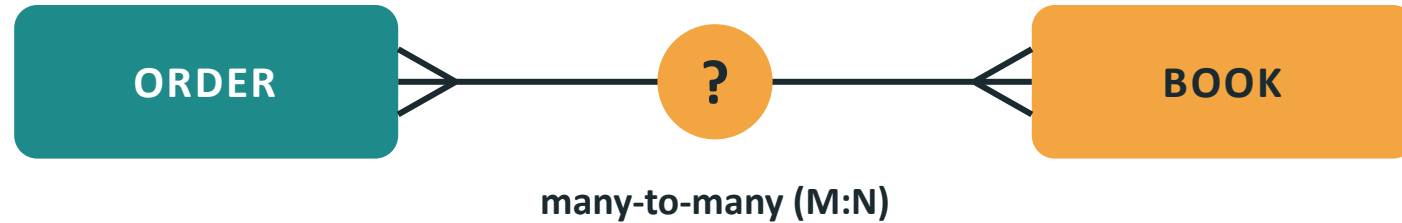
One order contains **MANY** books.
One book can appear on **MANY** orders.

HOW TO READ THE LINES

—+ = ONE

—> = MANY (crow's foot)

Step 4 · Fix the many-to-many



THE ORDER FORM, LINE BY LINE

Order	Book	Quantity
O101	B1	1
O101	B2	2
O102	B1	1

Each line = one order + one book + a number.

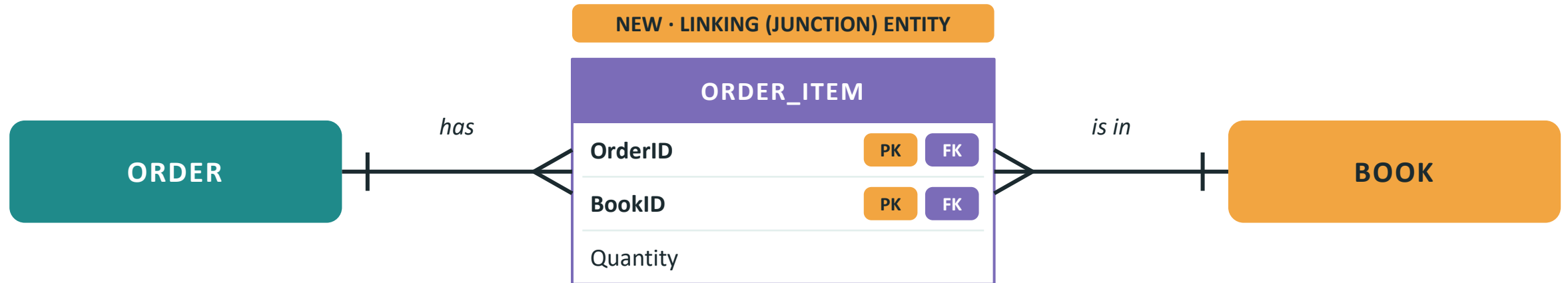


A database can't store many-to-many directly. What can we add to solve it? Where does Quantity belong?

HINT

Each line of an order form links ONE order to ONE book.

Step 4 · Fix the many-to-many



HOW TO FIX A MANY-TO-MANY

- 1 Add a **linking entity** between the two (ORDER_ITEM).
- 2 Copy both IDs into it: these are **foreign keys (FK)** and together they form its PK.
- 3 Move facts about the **pair** (Quantity) into it.



Both links are now one-to-many.

Step 5 · Draw the ERD

TOOLBOX

CUSTOMER

ORDER

ORDER_ITEM

BOOK

Draw your ERD here

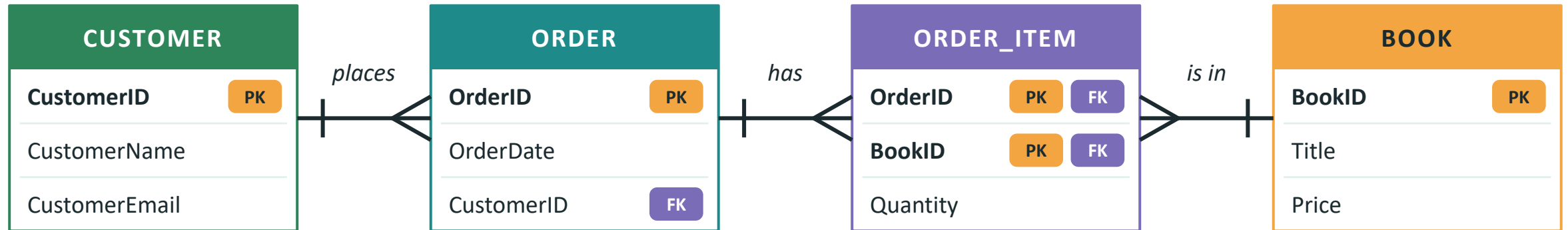
?

Draw the full ERD with the 4 entities. Show PK, FK and the ONE / MANY ends.

HINT

Put ORDER_ITEM in the middle. Copy each PK into the MANY side as an FK.

Step 5 · Draw the ERD



PK

Primary key: different for every row.

FK

Foreign key: a copy of another table's PK.



Bar = ONE end, crow's foot = MANY end.

Rule: copy the PK from the ONE side into the MANY side as an FK.

B

Normalise the data

UNF, then 1NF, 2NF and 3NF: one small fix at a time.

Why do we normalise?



Repeated data

Aisha's name and email are typed again on every order she makes. Change one copy and the others disagree.



Can't add data

A new book cannot be stored until somebody orders it.



Lose data

Delete Tom's only order and we lose Tom's details too.

Normalisation: organise the data so every fact is stored **once**.
We do it by splitting one big table into several small, linked tables.

Step 1 · Unnormalised data (UNF)

OrderID	OrderDate	CustomerID	CustomerName	CustomerEmail	BookID	Title	Price	Quantity
O101	01-Oct-2026	C1	Aisha Khan	aisha@mail.com	B1	Python Basics	£20	1
					B2	Web Design	£15	2
O102	03-Oct-2026	C2	Tom Evans	tom@mail.com	B1	Python Basics	£20	1
O103	05-Oct-2026	C1	Aisha Khan	aisha@mail.com	B3	Databases	£25	1



This is our order data typed into one table. Is it ready for a database? What is wrong with it?

HINT

Look closely at the Book columns in the first row.

Step 1 - Unnormalised data (UNF)

					REPEATING GROUP			
OrderID	OrderDate	CustomerID	CustomerName	CustomerEmail	BookID	Title	Price	Quantity
O101	01-Oct-2026	C1	Aisha Khan	aisha@mail.com	B1	Python Basics	£20	1
					B2	Web Design	£15	2
O102	03-Oct-2026	C2	Tom Evans	tom@mail.com	B1	Python Basics	£20	1
O103	05-Oct-2026	C1	Aisha Khan	aisha@mail.com	B3	Databases	£25	1



ANSWER

This table is in **UNF** - Unnormalised Form.
Order O101 has two books, so the Book columns hold **more than one value** in one cell.

HOW TO SPOT UNF

- 1 Look for cells that hold a **list** of values.
- 2 Ask: would I have to write 'B1, B2' in one cell?

Step 2 · First Normal Form (1NF)

OrderID	OrderDate	CustomerID	CustomerName	CustomerEmail	BookID	Title	Price	Quantity
O101	01-Oct-2026	C1	Aisha Khan	aisha@mail.com	B1	Python Basics	£20	1
					B2	Web Design	£15	2
O102	03-Oct-2026	C2	Tom Evans	tom@mail.com	B1	Python Basics	£20	1
O103	05-Oct-2026	C1	Aisha Khan	aisha@mail.com	B3	Databases	£25	1



How do we remove the repeating group? Which columns make each row unique (the primary key)?

HINT

Give each book its own row. Which pair of columns is never repeated?

Step 2 · First Normal Form (1NF)

PK					PK			
OrderID	OrderDate	CustomerID	CustomerName	CustomerEmail	BookID	Title	Price	Quantity
O101	01-Oct-2026	C1	Aisha Khan	aisha@mail.com	B1	Python Basics	£20	1
O101	01-Oct-2026	C1	Aisha Khan	aisha@mail.com	B2	Web Design	£15	2
O102	03-Oct-2026	C2	Tom Evans	tom@mail.com	B1	Python Basics	£20	1
O103	05-Oct-2026	C1	Aisha Khan	aisha@mail.com	B3	Databases	£25	1

1 One value in **every cell**.

2 No repeating groups.

3 A **primary key**: OrderID + BookID together.

Still repeated: Aisha's details (twice) and Python Basics (twice). 2NF and 3NF will fix that.

Step 3 - Second Normal Form (2NF)

PK					PK			
OrderID	OrderDate	CustomerID	CustomerName	CustomerEmail	BookID	Title	Price	Quantity
O101	01-Oct-2026	C1	Aisha Khan	aisha@mail.com	B1	Python Basics	£20	1
O101	01-Oct-2026	C1	Aisha Khan	aisha@mail.com	B2	Web Design	£15	2
O102	03-Oct-2026	C2	Tom Evans	tom@mail.com	B1	Python Basics	£20	1
O103	05-Oct-2026	C1	Aisha Khan	aisha@mail.com	B3	Databases	£25	1



The key is OrderID + BookID. For each other column: do we need BOTH parts of the key, or only one?

HINT

If I only know the BookID, can I find the Price? If I only know the OrderID, can I find the OrderDate?

Step 3 - Second Normal Form (2NF)

PK		Depends on OrderID only			PK	BookID only		Both
OrderID	OrderDate	CustomerID	CustomerName	CustomerEmail	BookID	Title	Price	Quantity
O101	01-Oct-2026	C1	Aisha Khan	aisha@mail.com	B1	Python Basics	£20	1
O101	01-Oct-2026	C1	Aisha Khan	aisha@mail.com	B2	Web Design	£15	2
O102	03-Oct-2026	C2	Tom Evans	tom@mail.com	B1	Python Basics	£20	1
O103	05-Oct-2026	C1	Aisha Khan	aisha@mail.com	B3	Databases	£25	1



ANSWER

- OrderDate, CustomerID, Name, Email need only OrderID.
- Title, Price need only BookID.
- Quantity needs both - it can stay.

THE 2NF TEST

- 1 Cover one part of the key.
- 2 Can you still find the value? That's a **partial dependency**.

Step 3 · Second Normal Form (2NF)

Key: OrderID

?

Which columns go here?

Key: BookID

?

Which columns go here?

Key: OrderID + BookID

?

Which columns go here?

?

**Split the table: one new table for each group.
Which columns go in each table, and what is its
key?**

HINT

Each column follows the key it depends on.
Quantity needs both IDs.

Step 3 - Second Normal Form (2NF)

ORDER

OrderID	OrderDate	CustomerID	CustomerName	CustomerEmail
O101	01-Oct-2026	C1	Aisha Khan	aisha@mail.com
O102	03-Oct-2026	C2	Tom Evans	tom@mail.com
O103	05-Oct-2026	C1	Aisha Khan	aisha@mail.com

BOOK

BookID	Title	Price
B1	Python Basics	£20
B2	Web Design	£15
B3	Databases	£25

ORDER_ITEM

OrderID	BookID	Quantity
O101	B1	1
O101	B2	2
O102	B1	1
O103	B3	1

WHAT CHANGED

- 1 Each group of columns moved to its **own table**.
- 2 The key columns are copied so the tables can be **linked** again.
- 3 Every non-key column now depends on its **whole key**.

Step 4 · Third Normal Form (3NF)

ORDER

<u>OrderID</u>	OrderDate	CustomerID	CustomerName	CustomerEmail
O101	01-Oct-2026	C1	Aisha Khan	aisha@mail.com
O102	03-Oct-2026	C2	Tom Evans	tom@mail.com
O103	05-Oct-2026	C1	Aisha Khan	aisha@mail.com

WHERE WE ARE

ORDER's key is just OrderID, so every column depends on the whole key. It is in 2NF. Is it in 3NF?

?

Look at the non-key columns. Does any non-key column decide the value of another non-key column?

HINT

If we know the CustomerID, do we automatically know the name and email? Where does the same customer appear twice?

Step 4 · Third Normal Form (3NF)

ORDER

<u>OrderID</u>	OrderDate	CustomerID	CustomerName	CustomerEmail
O101	01-Oct-2026	C1	Aisha Khan	aisha@mail.com
O102	03-Oct-2026	C2	Tom Evans	tom@mail.com
O103	05-Oct-2026	C1	Aisha Khan	aisha@mail.com



ANSWER

Yes! **CustomerID** decides **CustomerName** and **CustomerEmail**.

This is a **transitive dependency**.

*decides**decides*

OrderID

CustomerID

CustomerName + CustomerEmail

3NF rule: no non-key column may depend on another non-key column.

Step 4 · Third Normal Form (3NF)

ORDER (after the split)		
OrderID	OrderDate	?
O101	01-Oct-2026	?
O102	03-Oct-2026	?
O103	05-Oct-2026	?

NEW TABLE: ?		
?	?	?
?	?	?
?	?	?
?	?	?



Move the customer columns out. What new table do we need, what is its key, and how does ORDER still point to the customer?

HINT

The column that decides the others becomes the new table's key, and stays in ORDER as the link (FK).

Step 4 · Third Normal Form (3NF)

CUSTOMER		
<u>CustomerID</u>	CustomerName	CustomerEmail
C1	Aisha Khan	aisha@mail.com
C2	Tom Evans	tom@mail.com

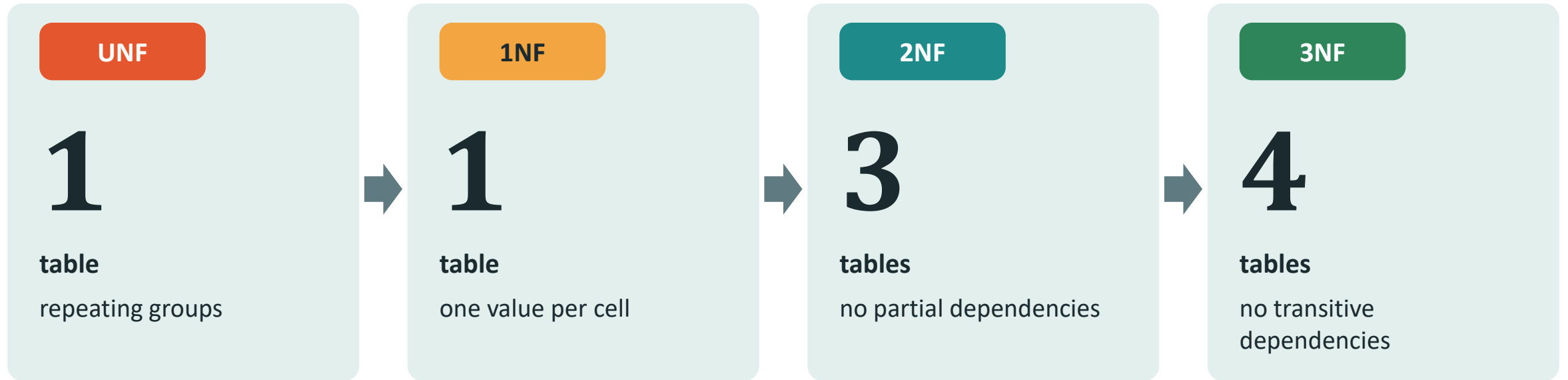
ORDER		
<u>OrderID</u>	OrderDate	CustomerID (FK)
O101	01-Oct-2026	C1
O102	03-Oct-2026	C2
O103	05-Oct-2026	C1

BOOK		
<u>BookID</u>	Title	Price
B1	Python Basics	£20
B2	Web Design	£15
B3	Databases	£25

ORDER_ITEM		
<u>OrderID</u> (FK)	<u>BookID</u> (FK)	Quantity
O101	B1	1
O101	B2	2
O102	B1	1
O103	B3	1

Underlined = primary key. FK = foreign key. Every fact is now stored once.

Two routes, the same four tables



CHECK YOUR WORK

The four 3NF tables are exactly the four entities in the ERD.

CUSTOMER

ORDER

ORDER_ITEM

BOOK

The normal forms at a glance

	RULE	ASK YOURSELF	FIX
UNF	Cells can hold lists (repeating groups).	Is there a list inside any cell?	Give each value its own row.
1NF	One value per cell, no repeating groups, a primary key.	Does every cell hold exactly one value?	Split repeating groups into rows; pick a key.
2NF	1NF + every non-key column depends on the WHOLE key.	Do I need all of the key to find this value?	Move partial dependencies to new tables.
3NF	2NF + no non-key column depends on another non-key column.	Does any column decide another non-key column?	Move them to a new table; keep the key as FK.

Every column depends on **the key** (1NF), **the whole key** (2NF) and **nothing but the key** (3NF).