DATABASES & SCHEMAS

DATABASE

- A database is an organized collection of data
- This is an example of a database arbitrarily called "Users"

id	first_name	last_name	created_at	updated_at
1	Cat	Woman	10/10/1990	10/10/2010
2	Bat	Man	02/02/1990	02/02/2020
3	Bob	Marley	03/03/1990	03/03/2030

RELATIONSHIPS

- But before we talk about building databases, we need to talk about this thing called relationships
- The reason being that databases are very dramatic. They're all about relationships
- Relationships are the backbone of your back-end.
- Before you even think about building a full-blown back-end, you want to make sure you have your relationships down first.

BEFORE THAT...

Table of Orders:-

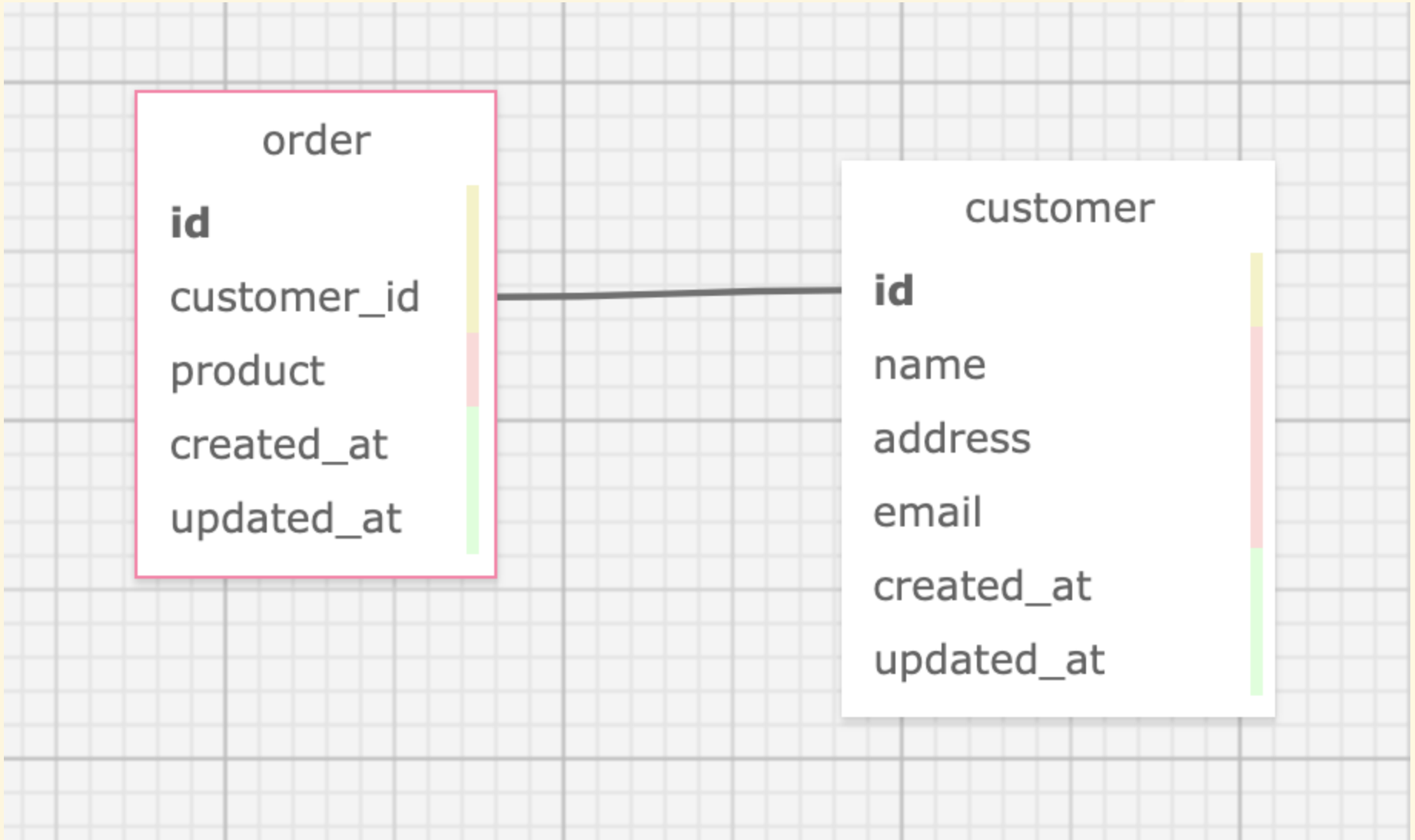
order #	customer	address	products	order date
1	Richard	Bludhaven	Suit	10/10/2010
2	Jason	Gotham	Hand Grenade	02/02/2020
3	Tim	Gotham	Glasses	03/03/2030
4	Damian	Venus	Knife	04/04/2040
5	Damian	Neptune	Another knife	05/05/2050
6	Damian	Uranus	Lemur	06/06/2060

What was the problem with the above table?

- Redundant data
- Repetitive data
- Might not be accurate
- Difficult to update
- Conflicting information
- Difficult to query through (especially since you probably have a LOT more orders than the above)

So what do??? 🤔

LET'S SHIP IT INSTEAD ❤️



SCHEMAS & KEYS

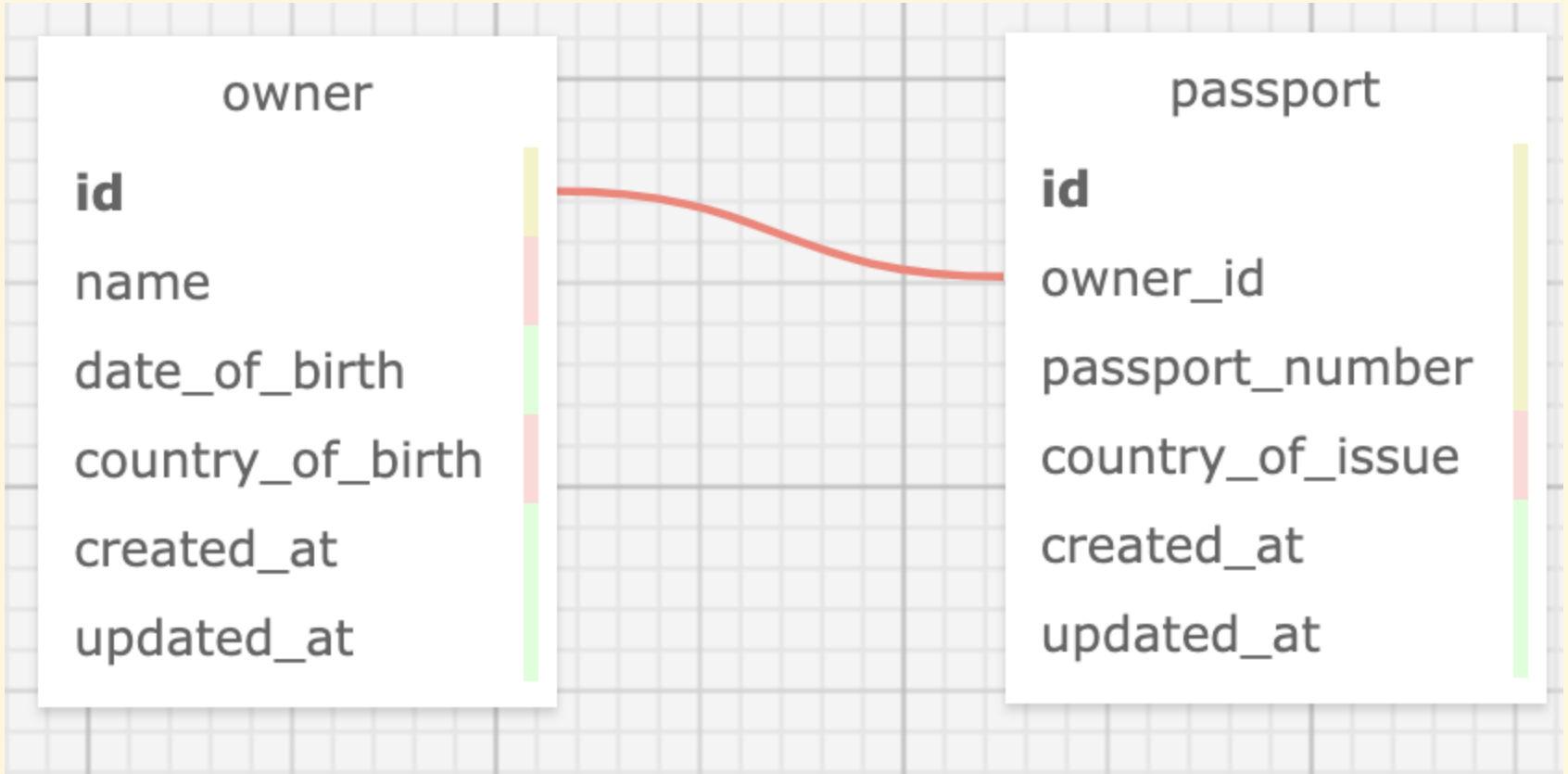
- Primary Key
  - Uniquely identifies a record in a table e.g. customer_id, user_id etc.
  - Cannot be null
  - Can only have one

SCHEMAS & KEYS contd...

- Foreign Key
  - Is a field in ONE table that corresponds to a PRIMARY key in ANOTHER table
  - Can be null i.e. you don't HAVE to have a foreign key if there's no relationship.
  - Can have multiple
  - id in customer table => primary key
  - customer_id in the orders table => foreign key

ONE-TO-ONE RELATIONSHIP

Think of it as a marriage. One husband. One wife. That's it. Do not pass go, do not collect $200, do not cheat on your partner.

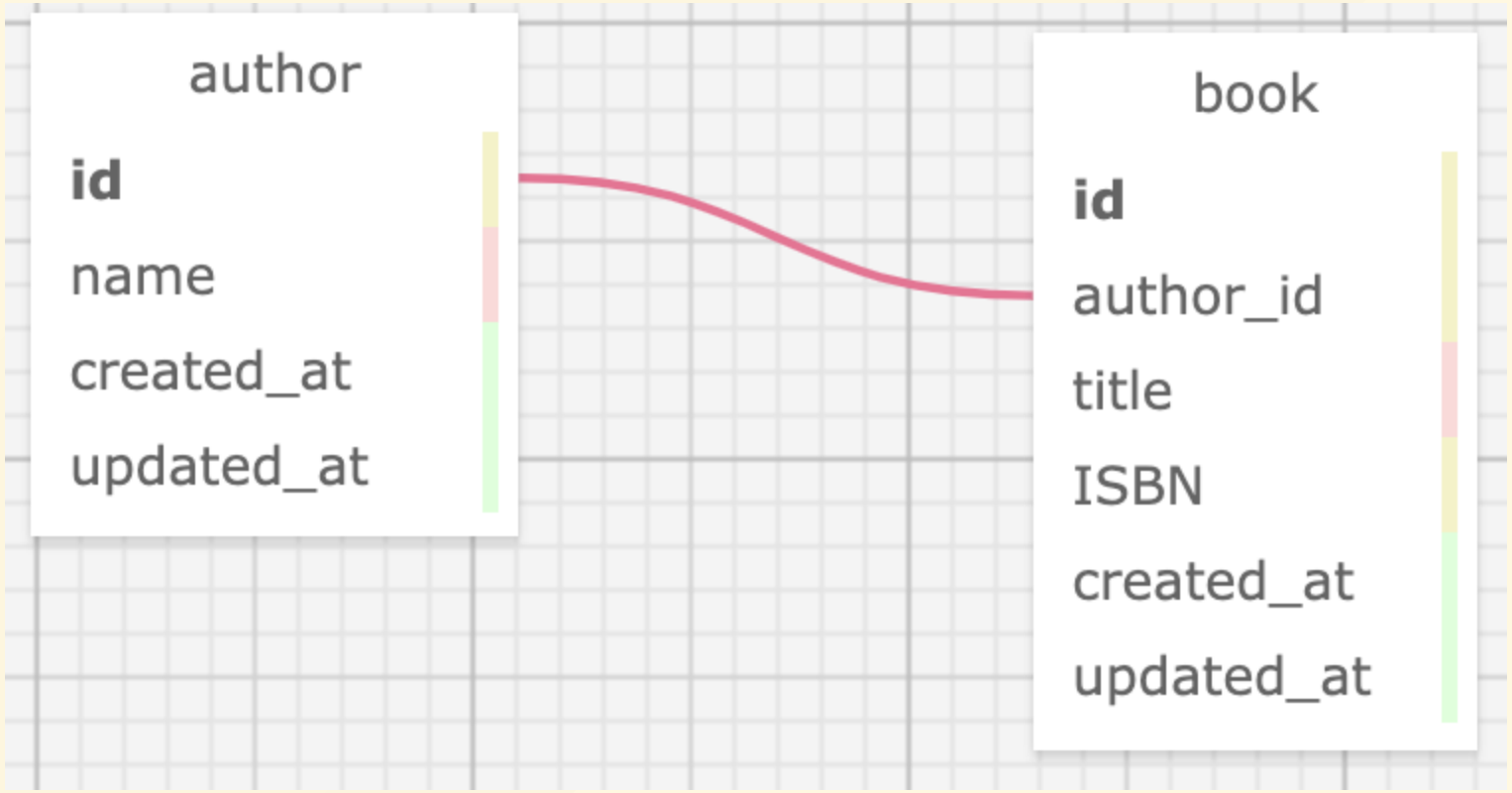


So your tables might look something like this:-

		Passport Table		
id	passport_number	owner_id	country_of_issue	created_at
1	12345	1	Malaysia	10/10/2010
2	23456	2	Malaysia	02/02/2010
3	34567	3	Hong Kong	03/03/2010
		User Table		
id	name	date_of_birth	country_of_birth	created_at
1	Steve	10/10/2010	Malaysia	10/10/2010
2	Stefan	02/02/2010	Malaysia	02/02/2010
3	Stephen	03/03/2010	Malaysia	03/03/2010

ONE-TO-MANY RELATIONSHIP

- Might look a little like one-to-one at first...

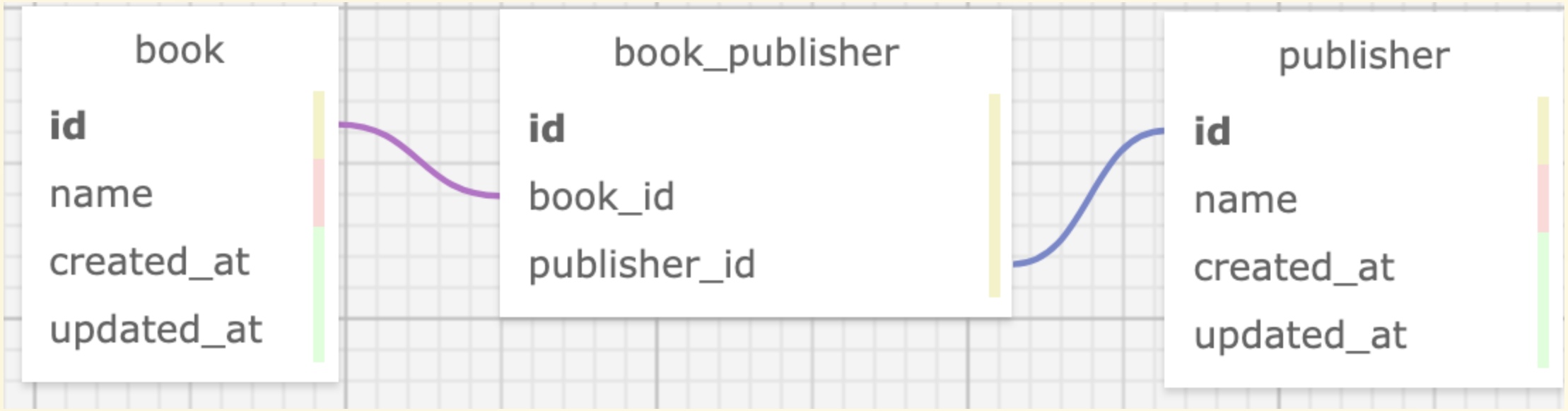


But take a look at your tables. See the difference?

		Author			
id	name	created_at	updated_at		
1	JK Rowling	10/10/2010	10/10/2010		
2	George RR Martin	02/02/2010	02/02/2010		
		Book			
id	author_id	title	ISBN	created_at	updated_at
1	1	Harry Potter & SS	12345	10/10/2010	10/10/2010
2	1	Harry Potter & CoS	23456	02/02/2010	02/02/2010
3	1	Harry Potter & PoA	34567	03/03/2010	03/03/2010
4	2	Game of Thrones	45678	04/04/2010	04/04/2010

MANY-TO-MANY RELATIONSHIP

- We create a JOIN TABLE to connect the two to store the RELATIONSHIP between two entities



- If we just connected the tables together we wouldn't get to see the PROPERTY of the relationship itself and your database might look something complicated like this:-

		Book		
id	name	publisher_id	created_at	updated_at
1	Harry Potter	1, 2,3	10/10/2010	10/10/2010
2	Game of Thrones	2,3	02/02/2010	02/02/2010
		Publisher		
id	title	book_id	created_at	updated_at
1	Bloomsbury	1,2,5,6,7,8	10/10/2010	10/10/2010
2	Penguin	2,5,7,8	02/02/2010	02/02/2010
3	HarperCollins	2,3,4	03/03/2010	03/03/2010

- Whereas this is much easier to look at, maintain and update:-

		Book			
id	name	created_at	updated_at		
1	Harry Potter	10/10/2010	10/10/2010		
2	Game of Thrones	02/02/2010	02/02/2010		
		Book_Publisher			
id	book_id	publisher_id	created_at	updated_at	
1	1	1	10/10/2010	10/10/2010	
2	1	2	02/02/2010	02/02/2010	
3	1	3	03/03/2010	03/03/2010	
4	2	**1**	04/04/2010	04/04/2010	
5	2	3	10/10/2010	10/10/2010	
		Publisher			
id	title	created_at	updated_at		
1	Bloomsbury	10/10/2010	10/10/2010		
2	Penguin	02/02/2010	02/02/2010		
3	HarperCollins	03/03/2010	03/03/2010		

TO DO

- Practice basic schemas - one-to-one, one-to-many, many-to-many
- In addition to schemas, draw out the actual tables as well for visualisation
- To-Do schema
- University Course schema