# COMP3005 Fall 2022 Project

Online Book Store: *Look Inna Book*

Project Report

December 11th, 2022

COMP3005 Fall 2022

Anoushka Singhal, 101100947
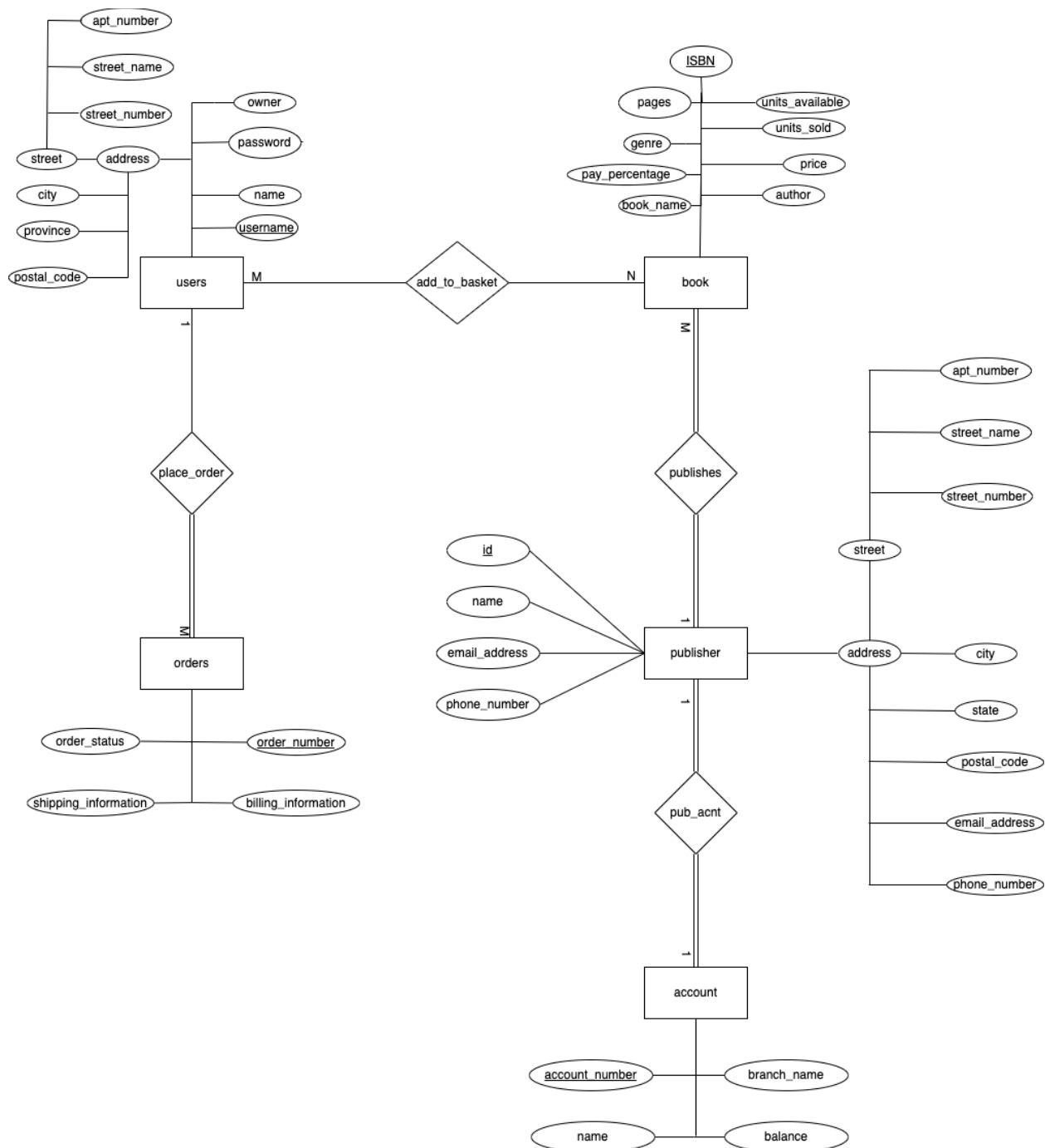
anoushkasinghal@cmail.carleton.ca

## Conceptual Design

There are a couple assumptions I made.
- Owner is a boolean value in account that if they are an owner, they will have access to removing and adding books, pay publisher 10%
- Publisher account stores the banking information for publisher to get 10%
- Many users can add many books to the basket
- Many orders can be placed by 1 user or 1 user can place many orders
- 1 publisher can publish many books or many books can be published by 1 publisher
- Each publisher has 1 account on file

# Reduction to Relation Schemas

**orders** | order_number | name | street_number | street_name | username |

**users** | username | name | password | owner | apt_number | street_name | street_number | city | province | postal_code |

**add_to_basket** | username | ISBN |

**book** | ISBN | book_name | author | genre | pages | price | units_sold | units_available | pay_percentage |

**publishes** | ISBN | id |

**publisher-account** | id | name | street_number | street_name | apt_number | city | state | postal_code | email_address | phone_number | account_number | name | branch_name | balance |
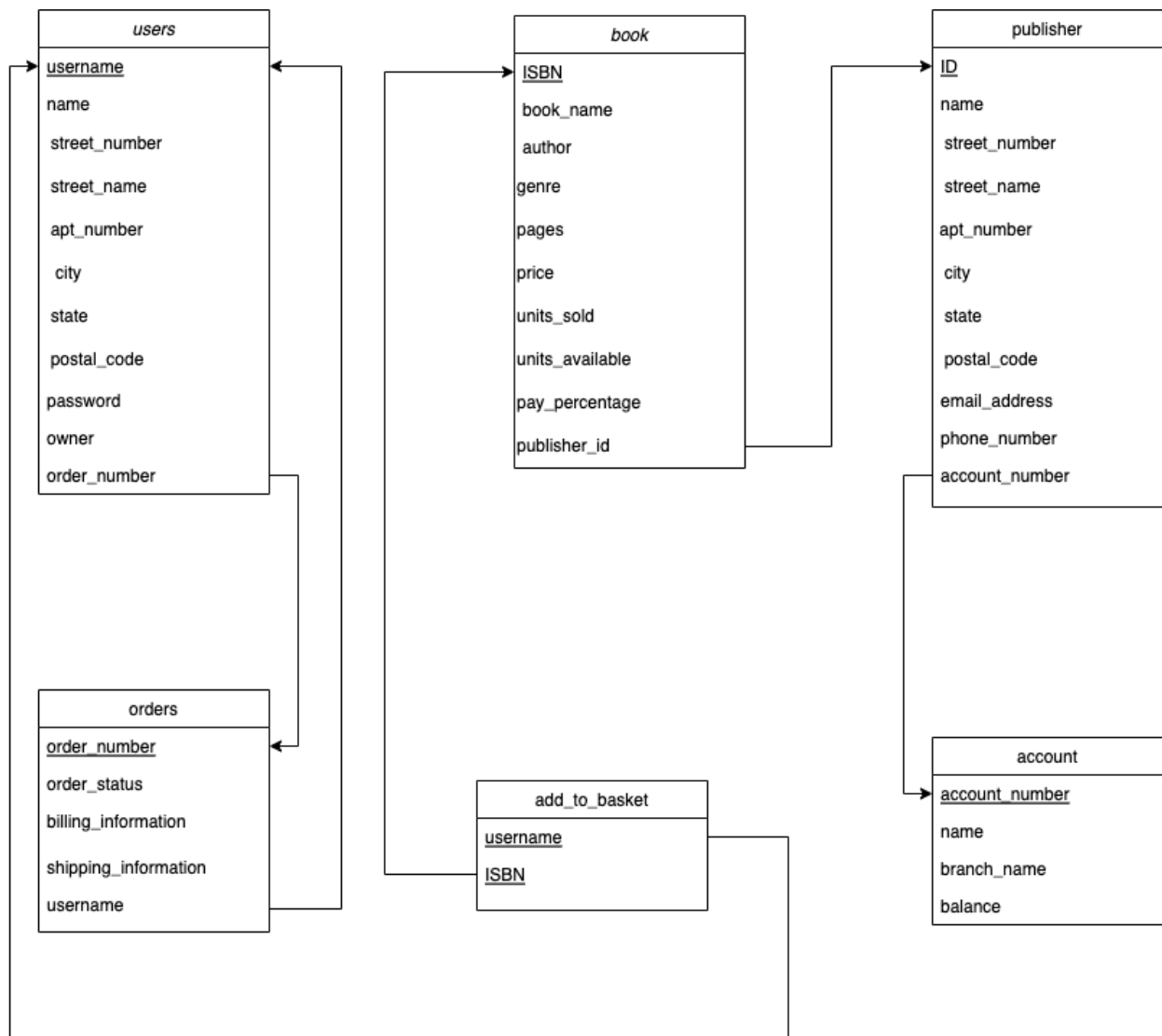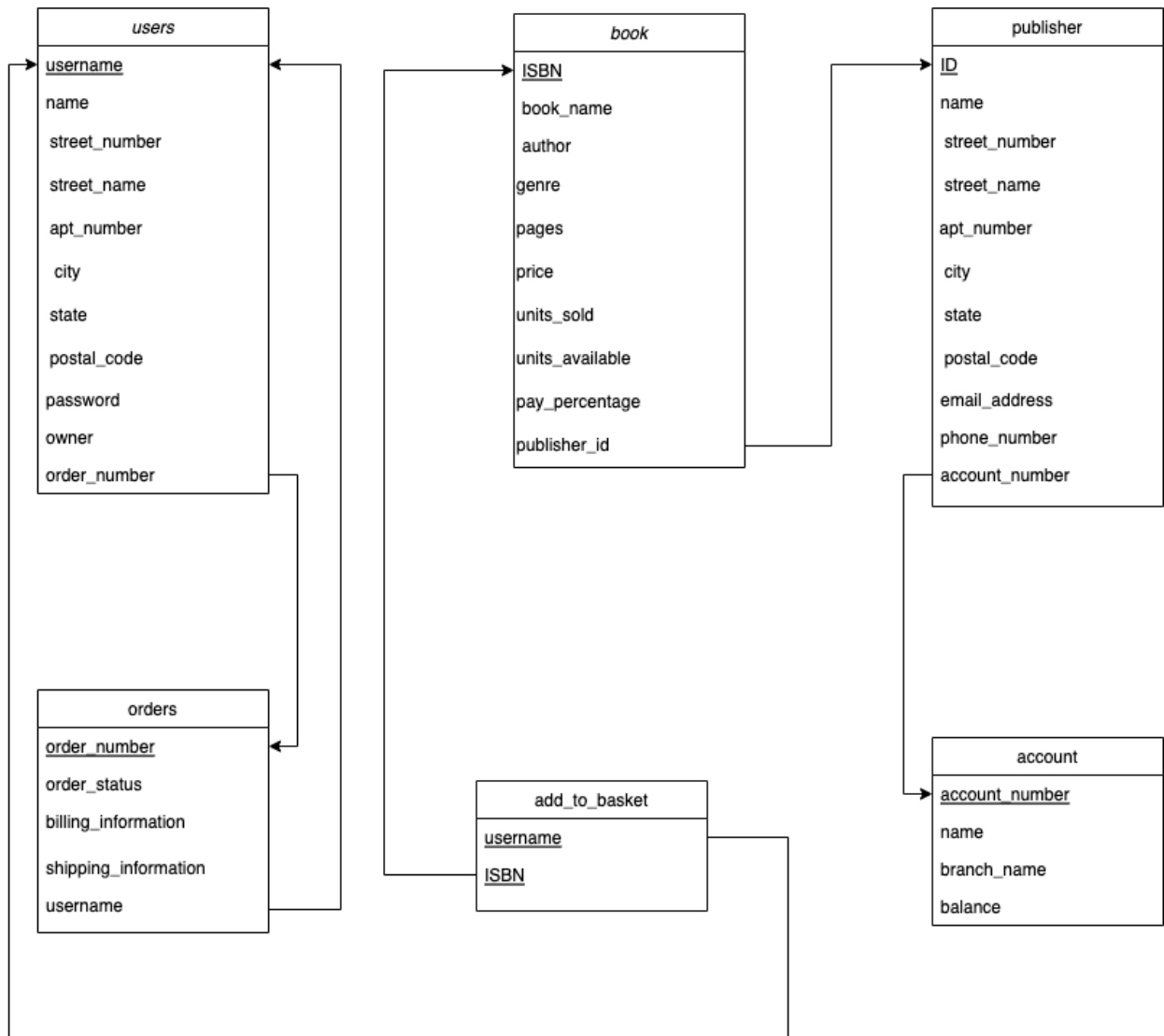
# Normalization of Relation Schemas

## 1NF

- For the 1NF, some requirements are:
    - Each table must have a primary key
    - All attributes must be simple with no multi value variable
- So the image below fulfills the requirements listed above.
    - The usernames are all unique to each user, there can't be two users with the same username
    - No two books can have the same ISBN number
    - No two publishers will have the same id
    - The order number for each order done by user is unique
    - Its unique when user adds a book to the basket
    - Each account number for each publisher is unique

| users |
| --- |
| username |
| name |
| street_number |
| street_name |
| apt_number |
| city |
| state |
| postal_code |
| password |
| owner |
| order_number |

| book |
| --- |
| ISBN |
| book_name |
| author |
| genre |
| pages |
| price |
| units_sold |
| units_available |
| pay_percentage |
| publisher_id |

| publisher |
| --- |
| ID |
| name |
| street_number |
| street_name |
| apt_number |
| city |
| state |
| postal_code |
| email_address |
| phone_number |
| account_number |

| orders |
| --- |
| order_number |
| order_status |
| billing_information |
| shipping_information |
| username |

| add_to_basket |
| --- |
| username |
| ISBN |

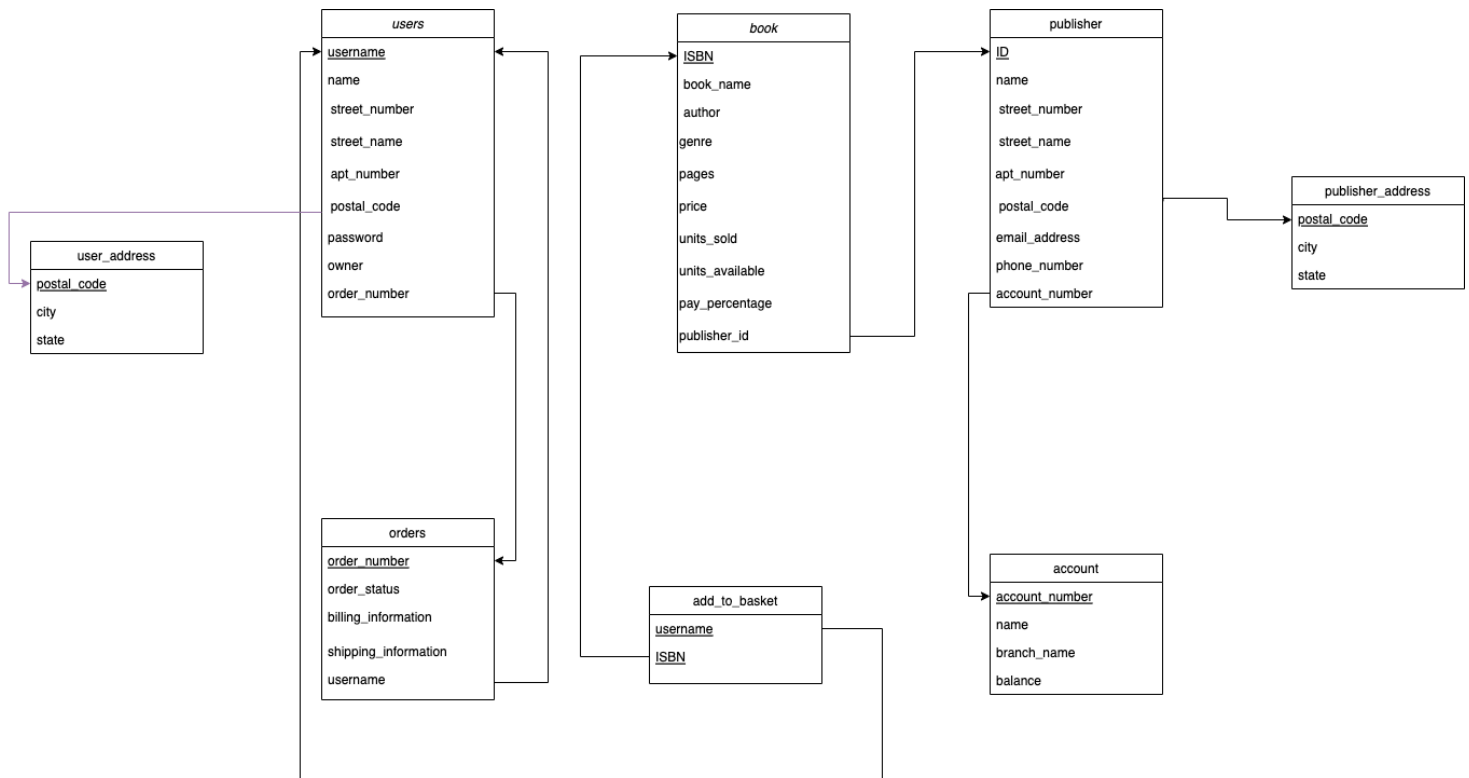| account |
| --- |
| account_number |
| name |
| branch_name |
| balance |

<u>2NF</u>

- For 2NF, some requirements are:
    - 1NF must be satisfied
    - All non-key attributes are dependent on primary key
- Which means the same diagram as above is in 1NF
    - All non key attributes are determined by their primary key as there is no other way to determine them

| users |
| --- |
| <u>username</u> |
| name |
| street_number |
| street_name |
| apt_number |
| city |
| state |
| postal_code |
| password |
| owner |
| order_number |

| book |
| --- |
| <u>ISBN</u> |
| book_name |
| author |
| genre |
| pages |
| price |
| units_sold |
| units_available |
| pay_percentage |
| publisher_id |

| publisher |
| --- |
| <u>ID</u> |
| name |
| street_number |
| street_name |
| apt_number |
| city |
| state |
| postal_code |
| email_address |
| phone_number |
| account_number |

| orders |
| --- |
| <u>order_number</u> |
| order_status |
| billing_information |
| shipping_information |
| username |

| add_to_basket |
| --- |
| <u>username</u> |
| <u>ISBN</u> |

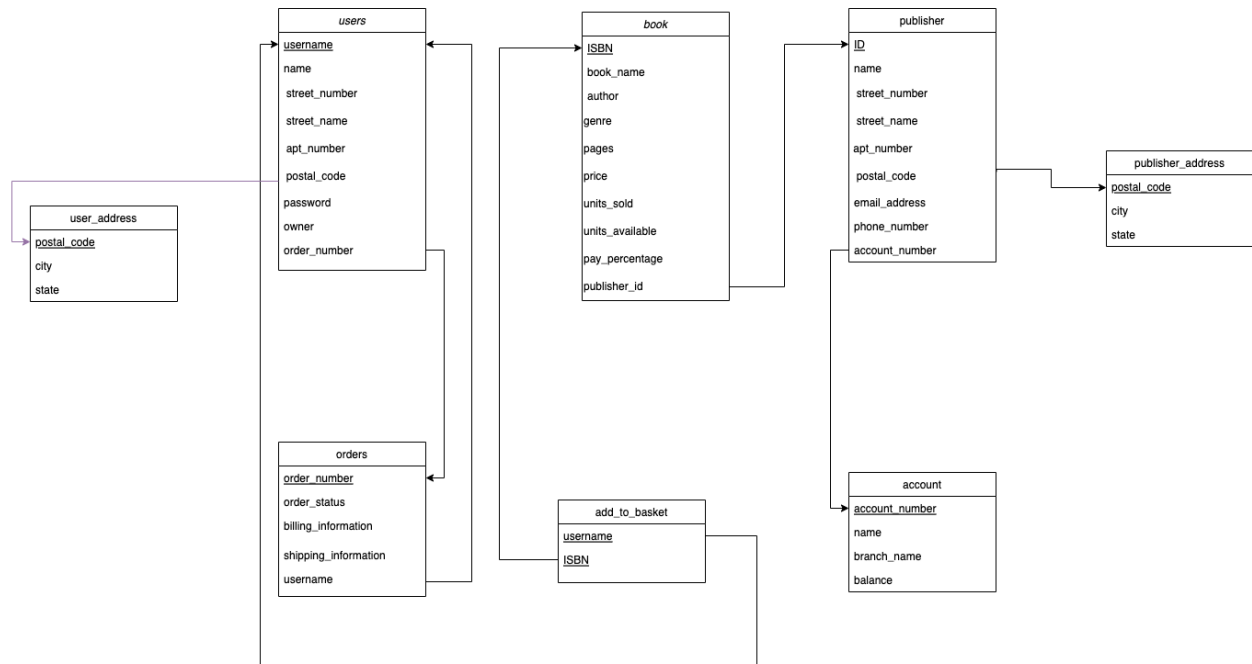| account |
| --- |
| <u>account_number</u> |
| name |
| branch_name |
| balance |

- For 3NF, some requirements are:
  - 2NF should be satisfied
  - No attributes should depend on other attributes that aren't primary keys
- Which means the diagram above is not in 3NF
  - Since Postal Code depends on the city and state you are in and postal code is not a primary key, we can make another table and connect them to each other through primary key

| users |
| --- |
| username |
| name |
| street_number |
| street_name |
| apt_number |
| postal_code |
| password |
| owner |
| order_number |

| book |
| --- |
| ISBN |
| book_name |
| author |
| genre |
| pages |
| price |
| units_sold |
| units_available |
| pay_percentage |
| publisher_id |

| publisher |
| --- |
| ID |
| name |
| street_number |
| street_name |
| apt_number |
| postal_code |
| email_address |
| phone_number |
| account_number |

| publisher_address |
| --- |
| postal_code |
| city |
| state |

| user_address |
| --- |
| postal_code |
| city |
| state |

| orders |
| --- |
| order_number |
| order_status |
| billing_information |
| shipping_information |
| username |

| add_to_basket |
| --- |
| username |
| ISBN |

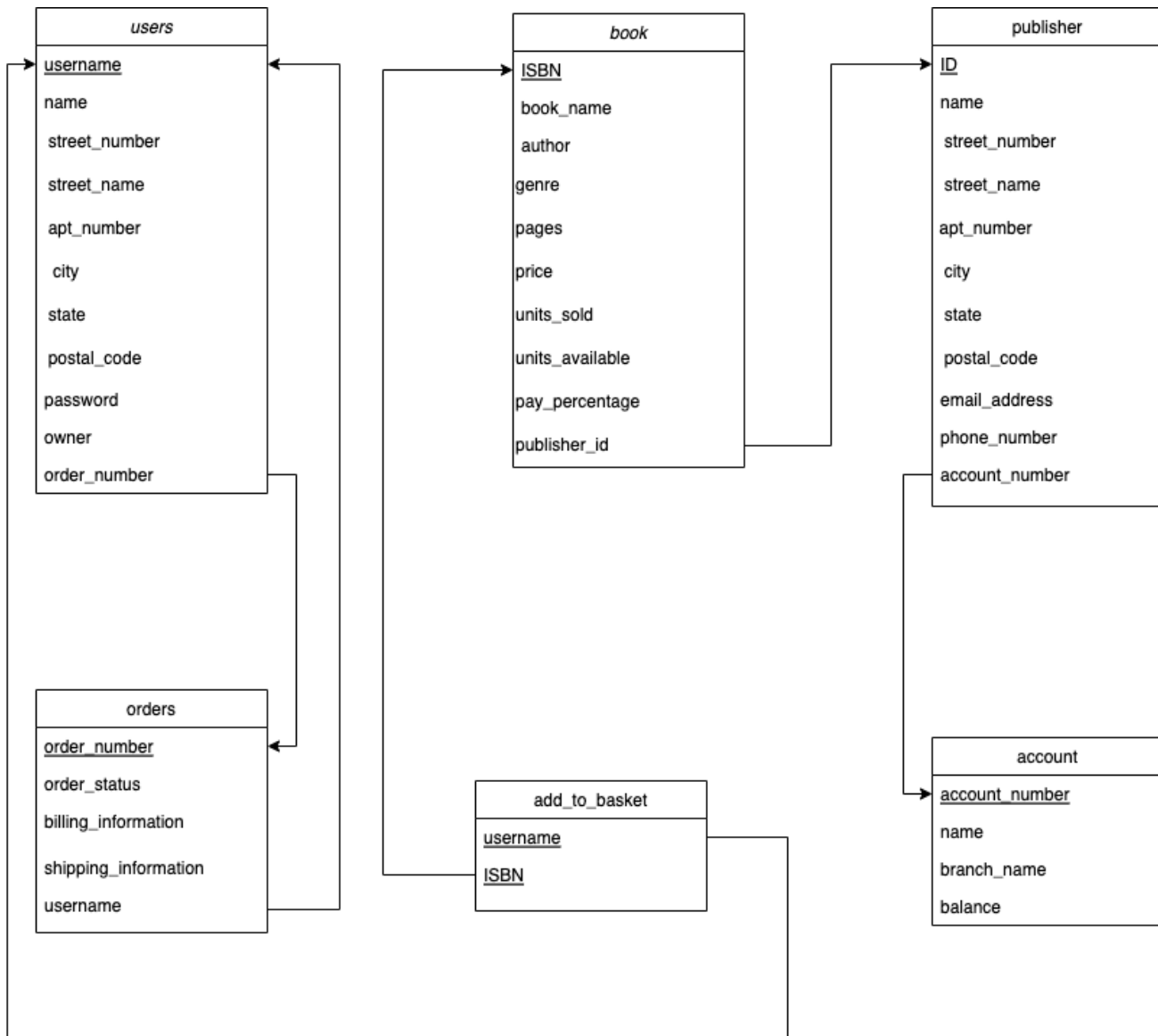| account |
| --- |
| account_number |
| name |
| branch_name |
| balance |

Since 3NF is very thorough with its good normal form, we don't have to go further, however there is the BCNF form that we will look at.

<u>BCNF</u>

- For BCNF, some requirements are:
  - 3NF is satisfied
  - If the primary key is a super key that all attributes are dependent on
- My diagram above is in BCNF good form since:
  - Publisher_id in the book table can be determined by the ISBN number which is the super key
  - Order_number is can be found through the username of user
  - Account number of publisher can be determined by their id

# Database Schema Diagram

**users**

<u>username</u>
name
street_number
street_name
apt_number
city
state
postal_code
password
owner
order_number

**book**

<u>ISBN</u>
book_name
author
genre
pages
price
units_sold
units_available
pay_percentage
publisher_id

**publisher**

<u>ID</u>
name
street_number
street_name
apt_number
city
state
postal_code
email_address
phone_number
account_number

**orders**

<u>order_number</u>
order_status
billing_information
shipping_information
username

**add_to_basket**

<u>username</u>
<u>ISBN</u>

**account**

<u>account_number</u>
name
branch_name
balance

## Bonus Features

I wanted to implement a bonus feature where the user searches for Lord of the Rings and finds multiple matches. The idea was similar to netflix where the user will be able to view books with similar words in it name.

Another bonus feature I wanted to add was a rating system, where users can provide feedback on the books they have read. As a strong and avid reader, this system would be helpful as I can rate and give feedback to all the books I've read, which will help organize my sticky notes and excel sheet into an online database.

## GitHub Repository

https://github.com/AnoushkaSinghal/3005Fall2022Project.git