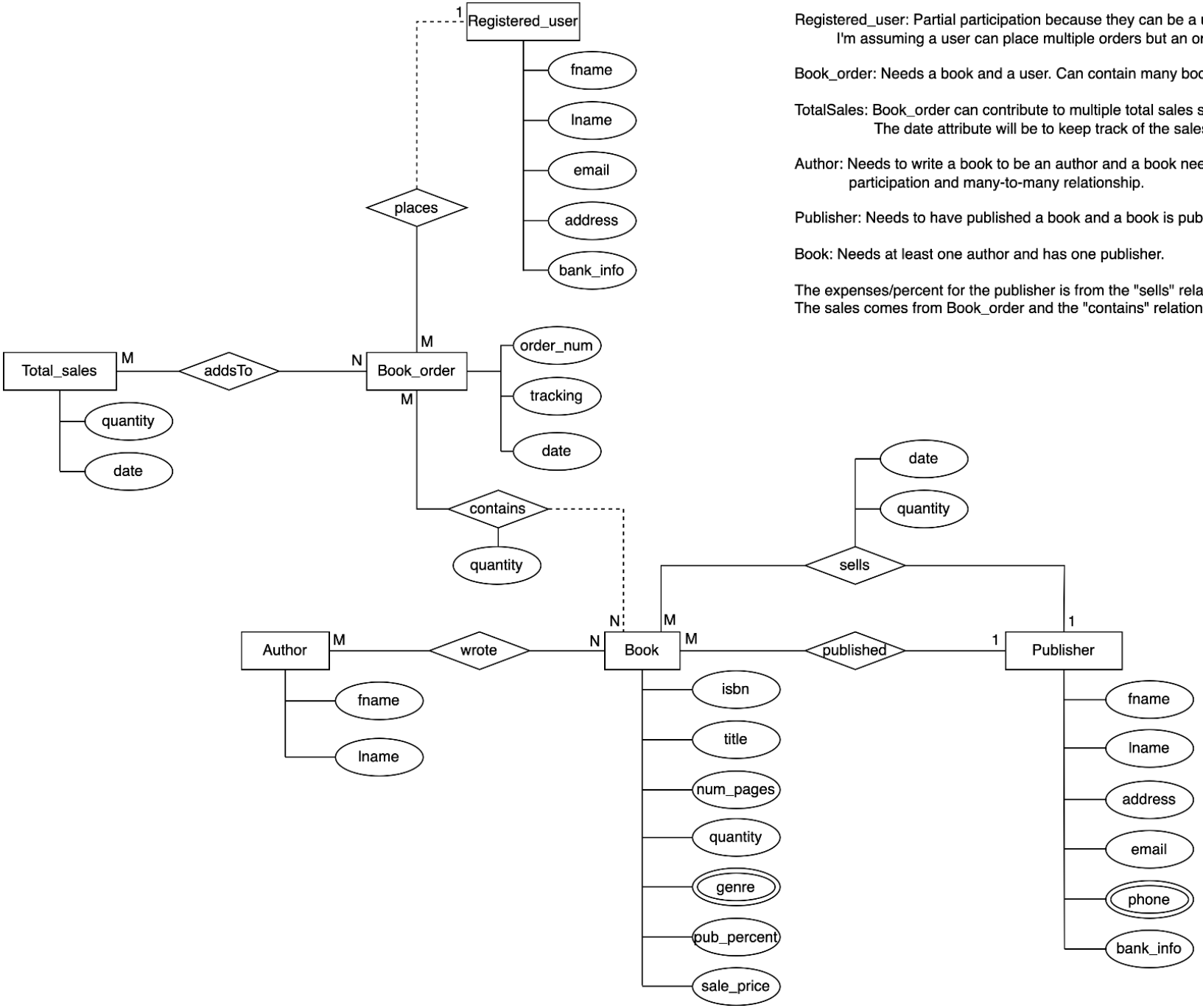
COMP 3005
Final Project
Maykala Salinas-Roy (101145833)

Look Inna Book - Project Report

Conceptual Design



Registered_user: Partial participation because they can be a user and not have an order placed. I'm assuming a user can place multiple orders but an order belongs to only one user.

Book_order: Needs a book and a user. Can contain many books.

TotalSales: Book_order can contribute to multiple total sales since there can be multiple books in an order. The date attribute will be to keep track of the sales by month and year.

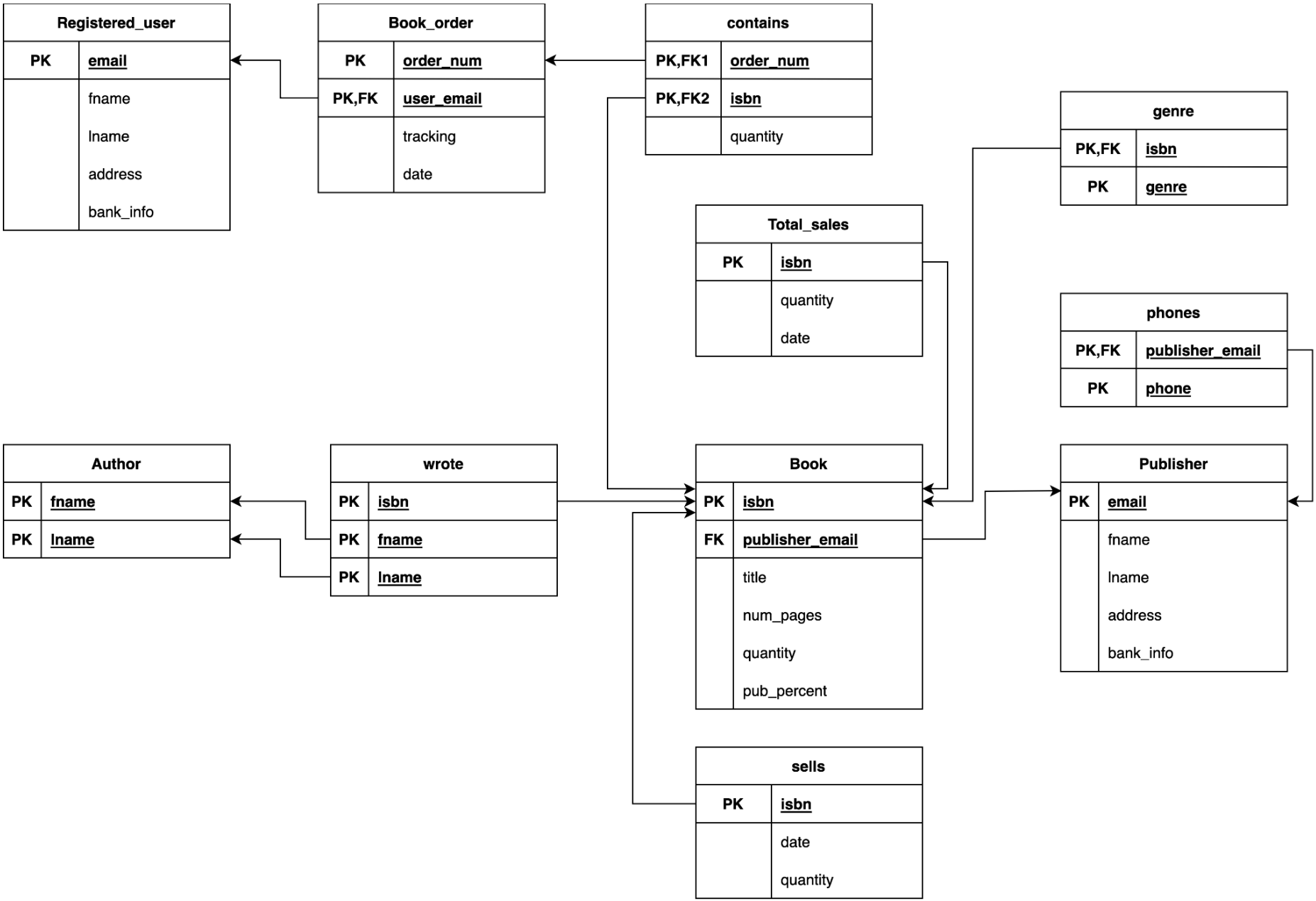
Author: Needs to write a book to be an author and a book needs at least 1 author so they are a full participation and many-to-many relationship.

Publisher: Needs to have published a book and a book is published by only one publisher.

Book: Needs at least one author and has one publisher.

The expenses/percent for the publisher is from the "sells" relationship
The sales comes from Book_order and the "contains" relationship

Relation Schemas



Normalization of Relation Schemas

I will be using the BCNF simplified test because all of my functional dependencies contain all of the attributes for their relations so they are not decompositions.

Relations with only trivial dependencies:

- Author
  - R = (fname, lname)
  - F = { fname, lname → fname, lname }
- wrote
  - R = (isbn, fname, lname)
  - F = { isbn, fname, lname → isbn, fname, lname }
- genre
  - R = (isbn, genre)
  - F = { isbn, genre → isbn, genre)
- phones
  - R = (publisher_email, phone)
  - F = { publisher_email, phone → publisher_email, phone)

All of these relations would pass since they are trivial/superkeys

Book Relation:

R = (isbn, publisher_email, title, num_pages, quantity, pub_percent, sale_price)
F = { isbn → publisher_email, title, num_pages, quantity, pub_percent, sale_price
isbn, publisher_email → title, num_pages, quantity, pub_percent, sale_price }

$(isbn)^+$
result = isbn
isbn → publisher_email, title, num_pages, quantity, pub_percent, sale_price: result = isbn, publisher_email, title, num_pages, quantity, pub_percent, sale_price
isbn, publisher_email → title, num_pages, quantity, pub_percent, sale_price: result = isbn, publisher_email, title, num_pages, quantity, pub_percent, sale_price

$(isbn)^+$ contains all of the attributes so it passes the test

$(isbn, publisher\_email)^+$
result = isbn, publisher_email
isbn → publisher_email, title, num_pages, quantity, pub_percent, sale_price: result = isbn, publisher_email, title, num_pages, quantity, pub_percent, sale_price
isbn, publisher_email → title, num_pages, quantity, pub_percent, sale_price: result = isbn, publisher_email, title, num_pages, quantity, pub_percent, sale_price

$(isbn, publisher\_email)^+$ contains all of the attributes so it passes the test as well
Therefore, the Book relation is in normal form.

Publisher Relation:
R = (email, fname, lname, address, bank_info)
F = { email → fname, lname, address, bank_info
bank_info → email, fname, lname, address }

$(email)^+$
result = email
email → fname, lname, address, bank_info: result = email, fname, lname, address, bank_info
bank_info → email, fname, lname, address: result = email, fname, lname, address, bank_info

$(email)^+$ contains all of the attributes so it passes the test
$(bank\_info)^+$
result = bank_info
email → fname, lname, address, bank_info: result = bank_info
bank_info → email, fname, lname, address: result = email, fname, lname, address, bank_info

$(bank\_info)^+$ contains all of the attributes so it passes the test
Therefore, the Publisher relation is in normal form.

Registered_user Relation:
R = (email, fname, lname, address, bank_info)
F = { email → fname, lname, address, bank_info
bank_info → email, fname, lname, address }

$(email)^+$
result = email
email → fname, lname, address, bank_info: result = fname, lname, address, bank_info
bank_info → email, fname, lname, address: result = fname, lname, address, bank_info, email

$(email)^+$ contains all of the attributes so it passes the test

$(bank\_info)^+$
result = bank_info
email → fname, lname, address, bank_info: result = bank_info
bank_info → email, fname, lname, address: result = bank_info, email, fname, lname, address

$(bank\_info)^+$ contains all of the attributes so it passes the test
Therefore, the Registered_user relation is in normal form.

Book_order Relation:
R = (order_num, user_email, tracking, date)
F = { order_num → user_email, tracking, date
order_num, user_email → tracking, date }

(order_num)$^+$
result = order_num
order_num → user_email, tracking, date: result = order_num, user_email, tracking, date
order_num, user_email → tracking, date: result = order_num, user_email, tracking, date

(order_num)$^+$ contains all of the attributes so it passes the test

(order_num, user_email)$^+$
result = order_num, user_email
order_num → user_email, tracking, date: result = order_num, user_email, tracking, date
order_num, user_email → tracking, date: result = order_num, user_email, tracking, date

(order_num, user_email)$^+$ contains all of the attributes so it passes the test
Therefore, the Book_order relation is in normal form.

Contains Relation:
R = (order_num, isbn, quantity)
F = { order_num, isbn → quantity }

(order_num, isbn)$^+$
result = order_num, isbn
order_num, isbn → quantity: result = order_num, isbn, quantity

(order_num, isbn)$^+$ contains all of the attributes so it passes the test
Therefore, the contains relation is in normal form.

Total_sales Relation:
R = (isbn, quantity, date)
F = { isbn, date → quantity }

(isbn, date)$^+$
result = isbn, date
isbn, date → quantity: result = isbn, date, quantity

(isbn, date)$^+$ contains all of the attributes so it passes the test
Therefore, the total_sales relation is in normal form.

The sells relation is the same as the total_sales for this test (They represent different things but their attribute names and their functional dependencies are the same)

Implementation

For my project, I am using Python 3.9.10 with PyQt6 for the GUI and sqlite3 for the database.

My program starts running as a user. To switch to the owner view, click the “Owner View” radio button. To go back to the user view, click the “User View” radio button.

Github Repository

https://github.com/maykalasalinas-roy/comp3005_finalProject