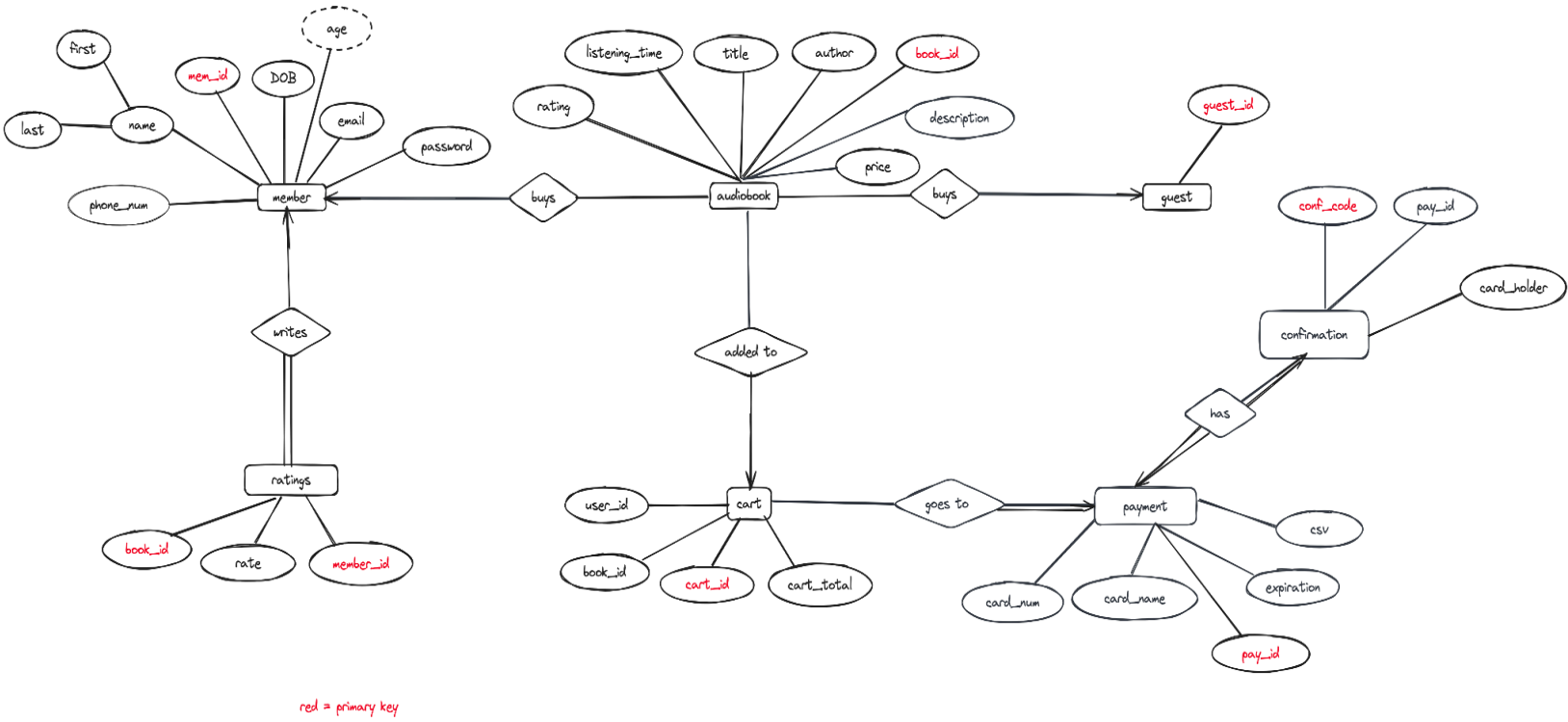
Team Members: Louis Borsato (lcb8vb), James Hawkins (jrh4az), Shelton Kwiterovich (vsk9hw), Jess Saviano (jrs5xrw)
Project Name: Great Listens
Project Concept: A web application that gives users a platform to view, purchase, and rate audio books.

Milestone 1 DB Design

Part 1: Develop E-R diagram



Part 2: Convert E-R diagram into tables

Schema Statements:

member(<u>mem_id</u>, name, phone_num, email, password, age)
writes(<u>mem_id</u>, <u>book_id</u>)
ratings(<u>book_id</u>, <u>mem_id</u>, rate)
buys(<u>mem_id</u>, <u>book_id</u>)
guest(guest_id)
buys(<u>guest_id</u>, <u>book_id</u>)
audiobook(<u>book_id</u>, rating, listening_time, title, author, price, description)

added_to(book_id, cart_id)
cart(cart_id, user_id**, book_id, cart_total)
goes_to(cart_id, pay_id)
payment(pay_id, credit_card, card_name, expiration, csv)
has(pay_id, conf_code)
confirmation(conf_code, pay_id, card_holder)

****Clarification: user_id is member id or guest id**
For Example, they will never be the same because:
User_id is M123432 for members
User_id is G138392 for guests

Part 3: Decompose tables using 3NF or **BCNF**

I. Functional dependencies of each table from part 2:

member(mem_id, name, phone_num, email, password, age)
mem_id -> name, phone_num, email, password, age
email -> mem_id, name, phone_num, password, age (assuming email is unique per member)
Phone_num -> mem_id, first,last, DOB, email, password, age

writes(mem_id, book_id)
This is a many-to-many relationship without attributes, so no functional dependencies can be derived from it.

ratings(book_id, mem_id, rate)
book_id, mem_id -> rate (assuming a member can rate a book only once)

buys(mem_id, book_id)
This represents a purchase event. Without additional attributes, we can't define FDs beyond the composite key itself. It is a many-to-many relationship.

guest(guest_id)
guest_id has no other attributes, so no FDs beyond its own identity.

buys(guest_id, book_id)
Similar to the buys for members, this represents a purchase event by a guest. It is also a many-to-many relationship.

audiobook(book_id, rating, listening_time, title, author, price, description)
book_id -> rating, listening_time, title, author, price, description
title, author -> book_id, rating, listening_time, price, description (assuming title and author together are unique)

added_to(book_id, cart_id)
cart_id, book_id -> (no attributes to depend on, this is a linking table for a many-to-many relationship)

cart(cart_id, user_id, book_id, cart_total)
cart_id -> user_id, cart_total

goes_to(cart_id, pay_id)
cart_id -> pay_id (assuming one cart leads to one payment)
pay_id -> cart_id (if the payment is unique to the cart)

payment(pay_id, credit_card, card_name, expiration, csv)
pay_id -> credit_card, card_name, expiration, csv
credit_card, expiration, csv -> pay_id (assuming the combination of credit card number, expiration, and CSV is unique)

has(pay_id, conf_code)
conf_code -> pay_id
pay_id -> conf_code (if each payment has a unique confirmation code)

confirmation(conf_code, pay_id, card_holder)
conf_code -> pay_id, card_holder

II. Process showing how you decompose the tables (using BCNF) or discussion showing that the tables are already in BCNF:
Member table: mem_id, email, and phone_num are all candidate keys based on the dependencies which means it is already in BCNF.
Writes table: Already in BCNF due to an absence of dependencies and attributes.
Ratings: Already in BCNF due to the only dependency having the superkey on the left-hand side.
Buys table: Already in BCNF due to all functional dependencies are satisfied by the candidate keys and there are not any functional dependencies beyond the composite key itself.

Guest table: Already in BCNF due to it not having any functional dependencies beyond its own identity.
Buys table: Already in BCNF due to all functional dependencies being satisfied by the candidate keys and there are not any functional dependencies beyond the composite key itself.
Audiobook Table: Already in BCNF due to book_id as well as the combination of title and author being candidate keys.
Added_to table: Already in BCNF due to non-trivial functional dependencies.
Cart Table: Already in BCNF due to no additional dependencies present from the (cart_id and book_id) dependency.
Goes_to table: Already in BCNF due to both attributes being candidate keys.
Payment Table: Already in BCNF each determinant being a candidate key in both relations.
Has Table: Already in BCNF due to all determinants of the functional dependencies being superkeys.
Confirmation Table: Already in BCNF due to conf_code being the candidate key in the only functional dependency.

III. Schema statements representing the normalized tables with all the necessary details:
member(mem_id, name, phone_num, email, password, age)
writes(mem_id, book_id)
ratings(book_id, mem_id, rate)
buys(mem_id, book_id)
guest(guest_id)
buys(guest_id, book_id)
audiobook(book_id, rating, listening_time, title, author, price, description)
added_to(book_id, cart_id)
cart(cart_id, user_id, book_id, cart_total)
goes_to(cart_id, pay_id)
payment(pay_id, credit_card, card_name, expiration, csv)
has(pay_id, conf_code)
confirmation(conf_code, pay_id, card_holder)