Database Concepts – Normalisation Exercise 02

The following relation (UNF) represents the data requirements of a library administration system:

Rentals (RentId, MemberId, Surname, Forename, phone, BooksOut, (BookId,
Title, Author, BookStatus, DateOut, DueDate, DateReturned, FineAmount))

Where several books may be borrowed in a single transaction. A fine amount is per overdue book.

Transform this relation into one or more 3NF relations.
Explain each transformation as you proceed.

(b) **UNF**: **Rentals** (RentId, MemberId, Surname, Forename, phone, BooksOut, BookId,
Title, Author, BookStatus, DateOut, DateDue, DateReturned, FineAmount)

1NF remove repeating groups:

Since several books may be borrowed in a single transaction (rental)

BookId, Title, Author, BookStatus, DateReturned, FineAmount are repeating attributes

- Remove these attributes to a new relation
- Post the original PK to the new relation
- Name the new relation appropriately
- Assign a PK to the new relation

Rentals (RentId, MemberId, Surname, Forename, phone, BooksOut, DateOut, DateDue)

BooksRented (RentId, BookId, Title, Author, BookStatus, DateReturned, FineAmount)

Ensure that all non-key attributes are functionally dependent on the PK.

2NF Remove partial dependencies

Relation must be in 1NF.
All non-key attributes must be ***fully*** functionally dependent on the PK.

Since the relation ***BooksRented*** has a composite PK, examine this relation.

BookId → Title, Author, BookStatus
RentID, BookID → Date Returned, FineAmount

Title, Author, BookStatus are ***partially*** dependent on the PK.

- Remove these attributes to a new relation
- The determining attribute (subset of PK) becomes the PK of the new relation
- Name the new relation appropriately

Rentals (RentId, MemberId, Surname, Forename, phone, BooksOut, DateOut, DateDue)

BooksRented (RentId, BookId, DateReturned, FineAmount)

Books (BookId, Title, Author, BookStatus)

3NF Remove transitive dependencies

Relation must be in 2NF.
Pairs of non-key attributes must not be transitively dependent on the PK.

Examining the relation ***Rentals*** reveals the following transitive dependencies:

RentId → (MemberId → Surname)
RentId → (MemberId → Forename)
RentId → (MemberId → Phone)
RentId → (MemberId → BooksOut)

OR

RentId → MemberId - > Surname, Forename, Phone, BooksOut

- Remove these attributes to a new relation
- The determining attribute becomes the PK of the new relation
- Name the new relation appropriately

Rentals (<u>RentId</u>, MemberId, DateOut, DueDate, DateReturned)
Members (<u>MemberId</u>, Surname, Forename, phone, BooksOut)
BooksRented (<u>RentId</u>, <u>BookId</u>, FineAmount)
Books (BookId, Title, Author, BookStatus)