Process of creating the conceptual and logical model

BT2102 Group 31

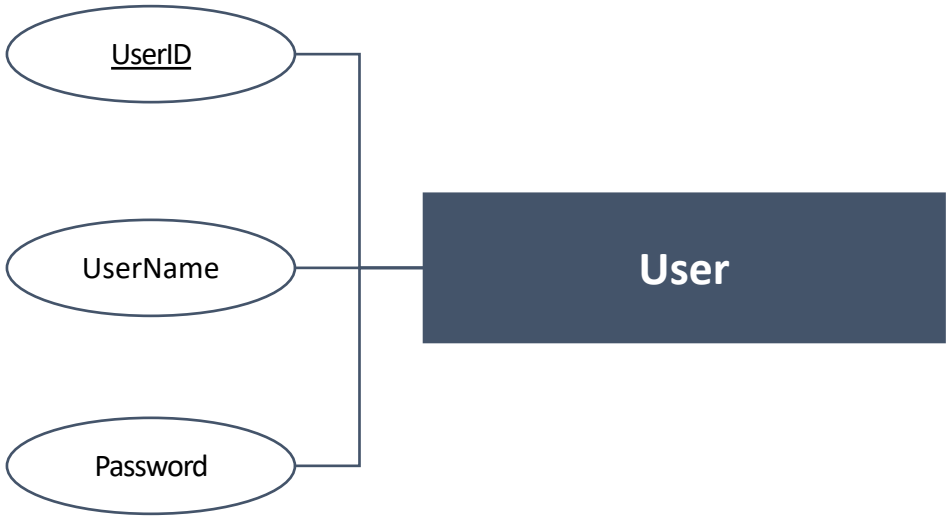
Assignment 1

Introduction

In order to design an efficient SQL database, we have decided to use Entity-Relationship modelling to identify our entities and map out the relationships between the different entities for our project

To begin, we first identified the strong entities present within the assignment

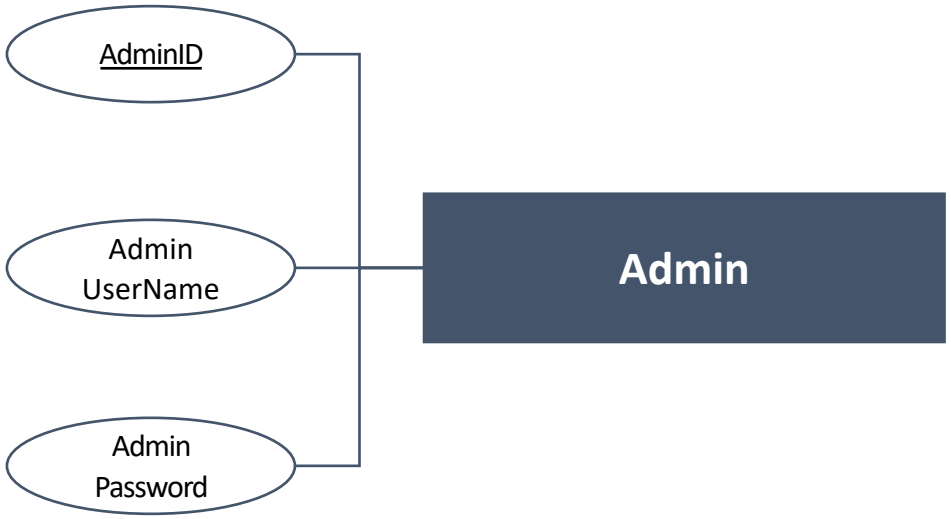
Strong Entity Identification (User)



The first strong entity we identified was the user, who will be the ones using the system. We identified the following 3 attributes for users:

1) **UserID**: Primary Key used to uniquely identify each user
2) **UserName**: Used to login into the system
3) **Password**: Attached to each UserName to verify the identity of the user during login

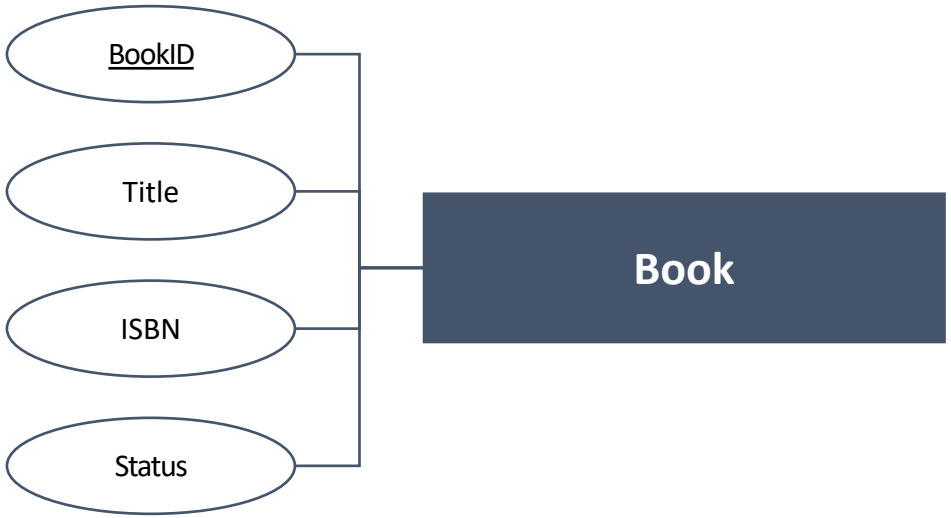
Strong Entity Identification (Admin)



The second strong entity we identified was the admin, who will have access to the administrative functions in the system. We identified the following 3 attributes for admins:

1) **AdminID**: Primary Key used to uniquely identify each user
2) **AdminUserName**: Used by the admin to login into the system
3) **AdminPassword**: Attached to each AdminUserName to verify the identity of the user during login

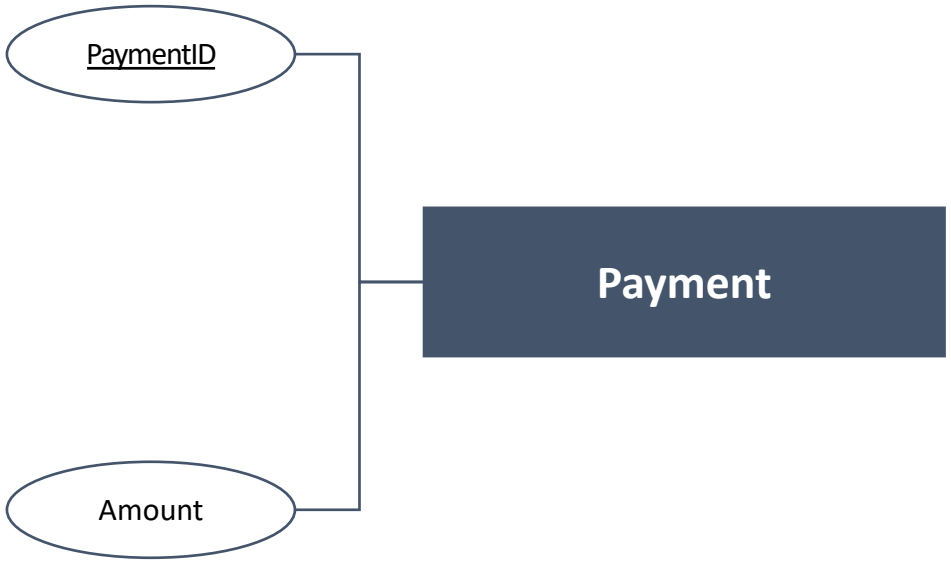
Strong Entity Identification (Book)



The third strong entity we identified was the book. We identified the following 4 attributes for books:

1) **BookID**: Primary Key used to uniquely identify each book
2) **Title**: Title of the book
3) **ISBN**: ISBN attached to each book
4) **Status**: Whether the book is currently in circulation and available to be borrowed within the system

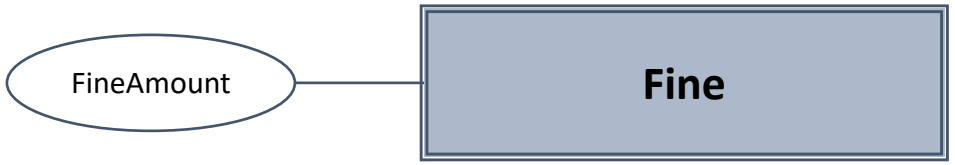
Strong Entity Identification (Payment)



The fourth strong entity we identified was the payments that users make to pay fines. We identified the following 3 attributes for payment:

1) **PaymentID**: Primary Key used to uniquely identify each payment
2) **Amount**: Amount of payment made

Weak Entity Identification (Fine)

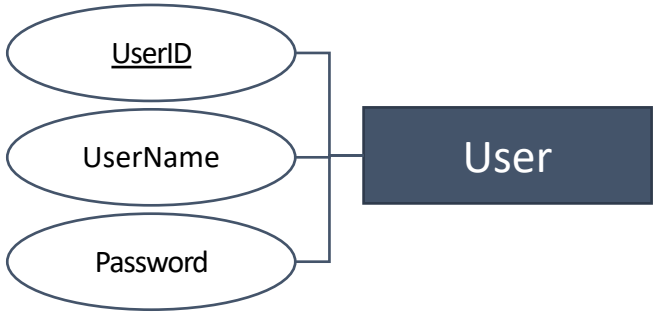


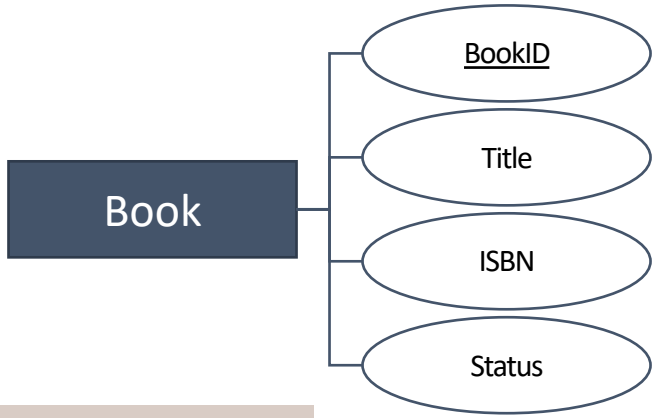
We have also identified one weak entity, fine. Fine is dependent on the User entity. The Fine entity has 1 attribute:

1) **FineAmount**: Amount of Fines owed by the User for overdue books

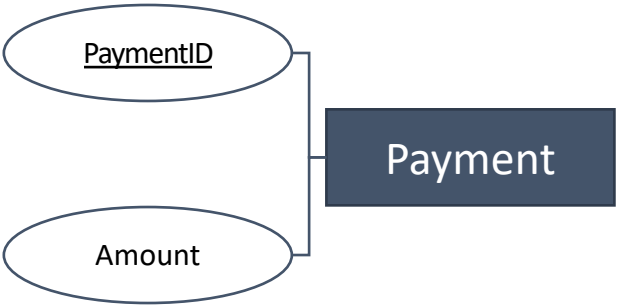
As a weak entity, the Fine entity's primary key is the primary key of User, which is UserID

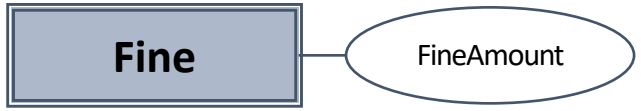
ER diagram with only entities

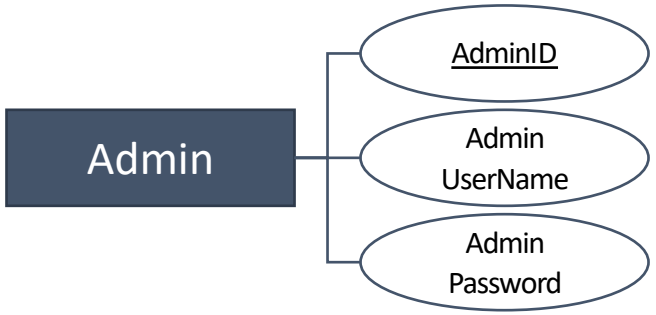




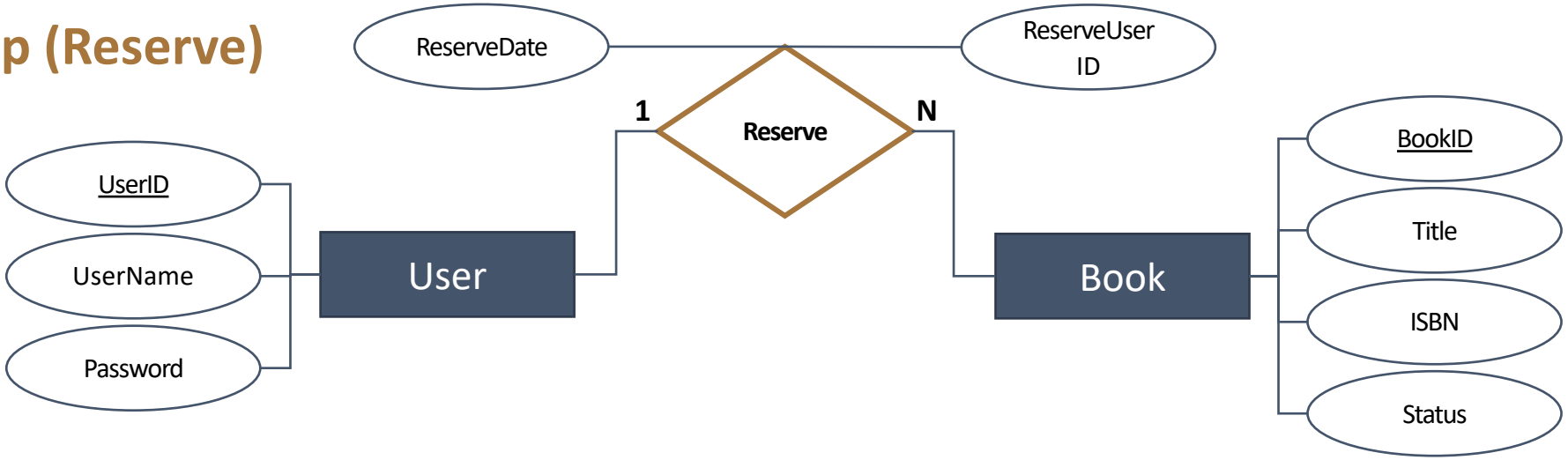
With the entities identified, we proceeded on by identifying the relationships between the different entities

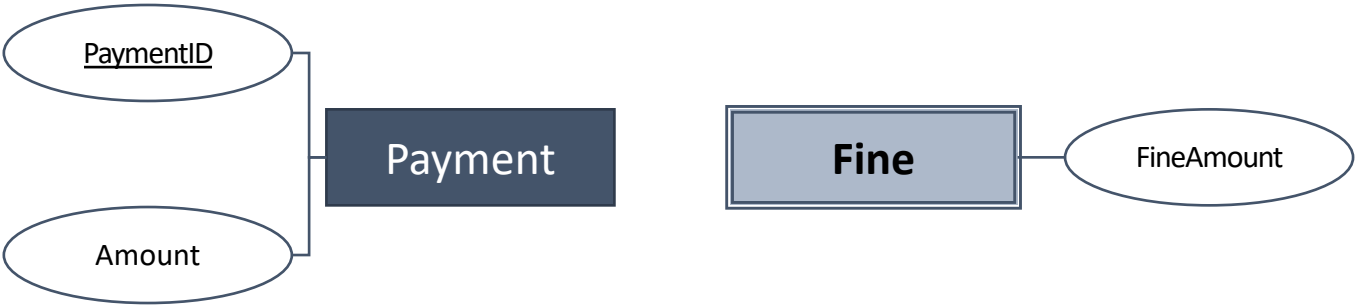






Relationship (Reserve)

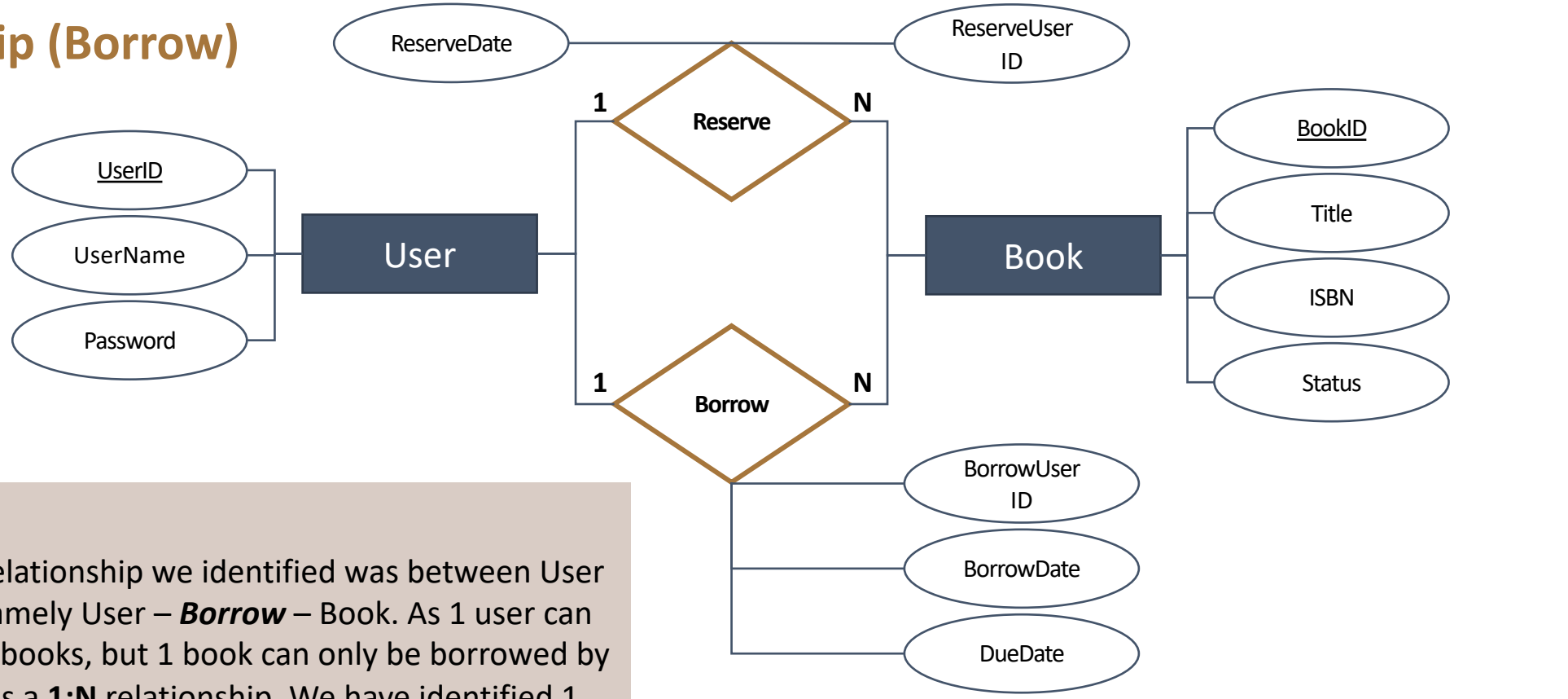




The first relationship we identified was between User and Book, namely User – ***Reserve*** – Book. As 1 user can reserve many books, but 1 book can only be reserved by 1 user, this is a **1:N** relationship. We have identified 1 attribute for this relationship:

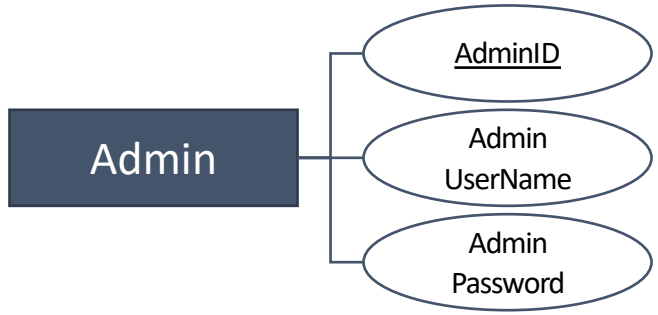
1) **ReserveUserID**: Used to identify which user has reserved the book. This is to distinguish it from the user that borrowed the book.
2) **ReserveDate:** Date which the book was reserved for

Relationship (Borrow)

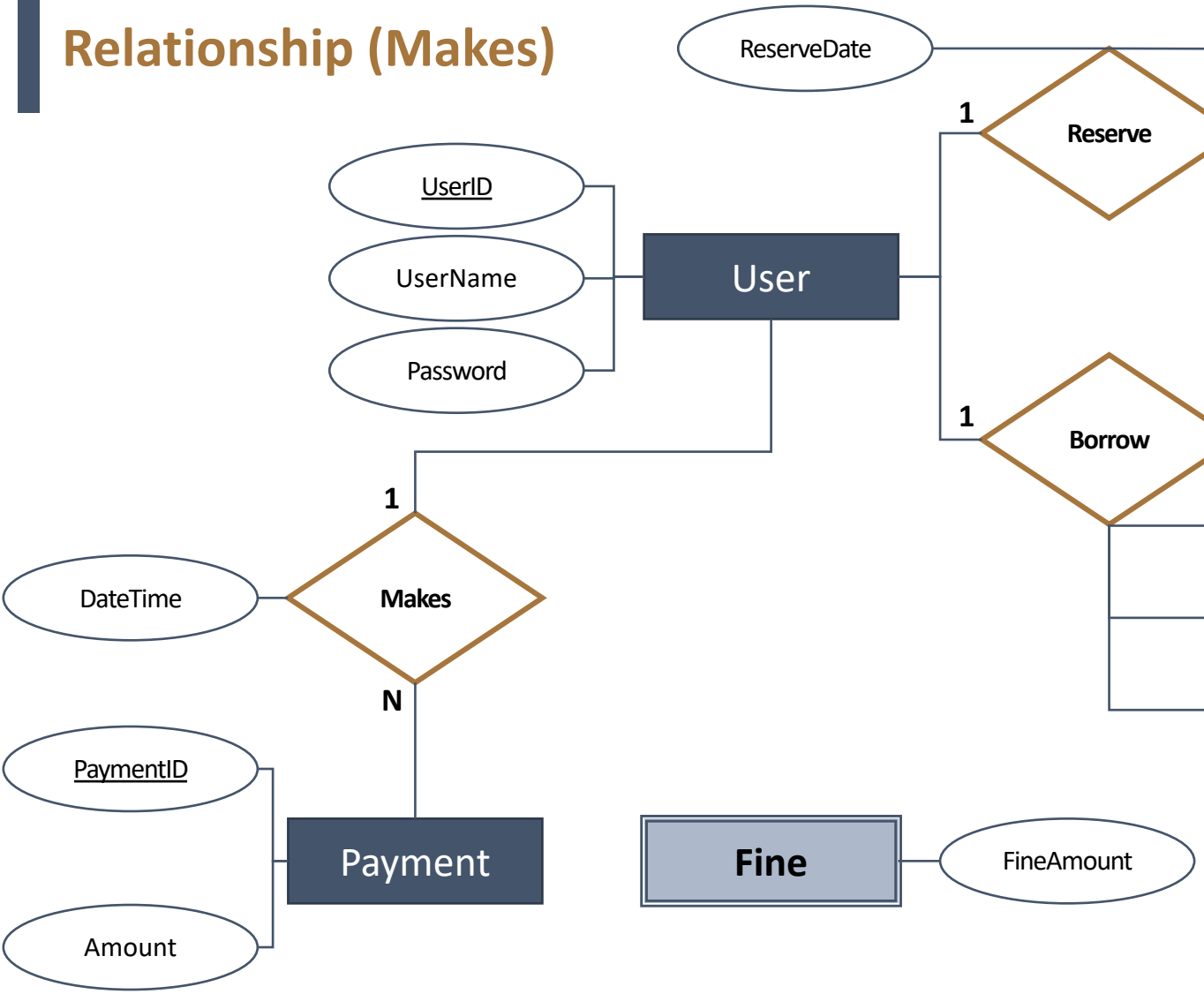


The second relationship we identified was between User and Book, namely User – ***Borrow*** – Book. As 1 user can borrow many books, but 1 book can only be borrowed by 1 user, this is a **1:N** relationship. We have identified 1 attribute for this relationship:

1) **BorrowUserID**: Used to identify which user has reserved the book. This is to distinguish it from the user that reserved the book.
2) **BorrowDate**: A record of when the book was borrowed
3) **DueDate**: A record of when the book is due



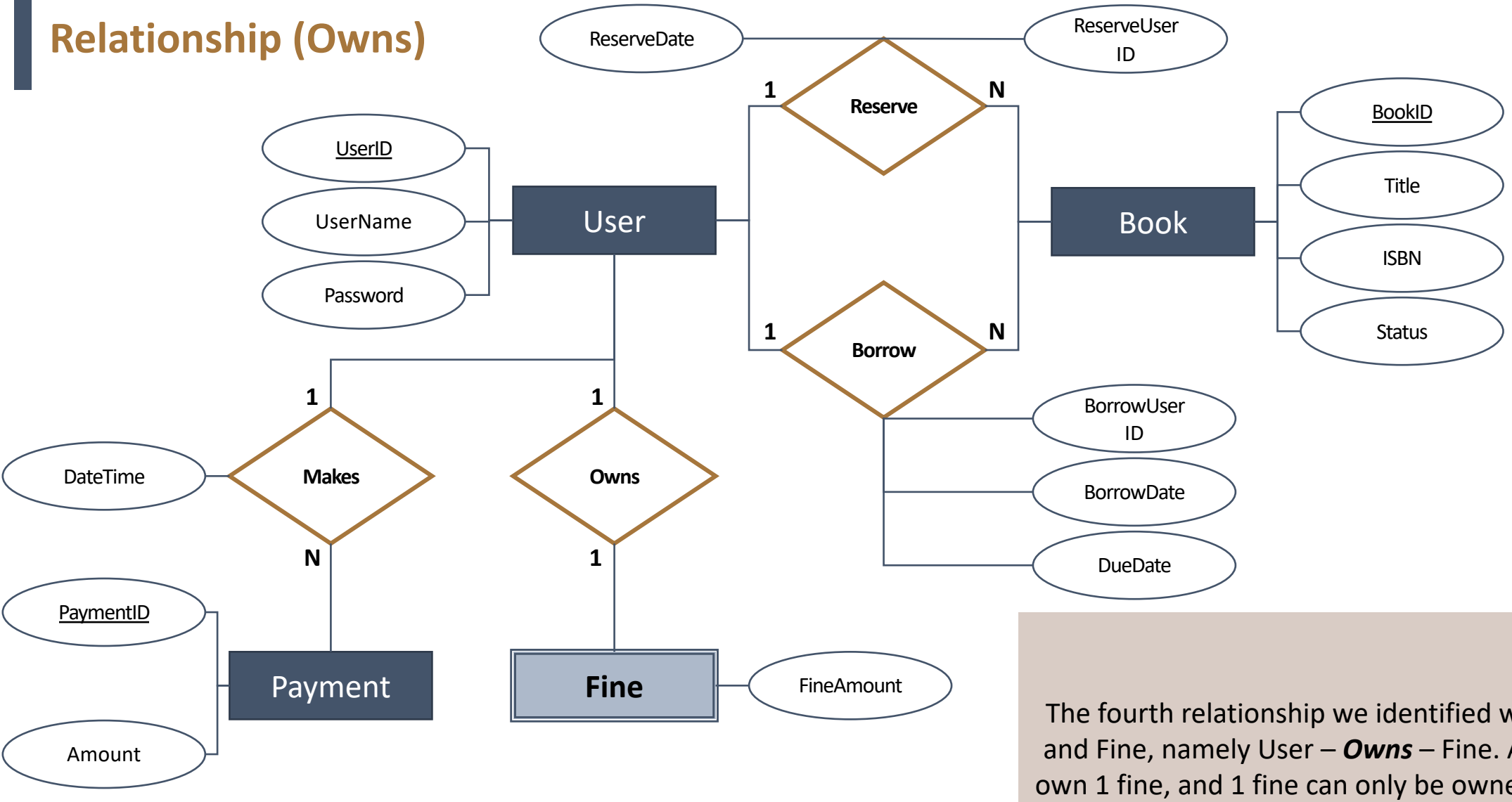
Relationship (Makes)



The third relationship we identified was between User and Payment, namely User – ***Makes*** – Payment. As 1 user can make many payments, but 1 payment can only be made by 1 user, this is a **1:N** relationship. We have identified 1 attribute for this relationship:

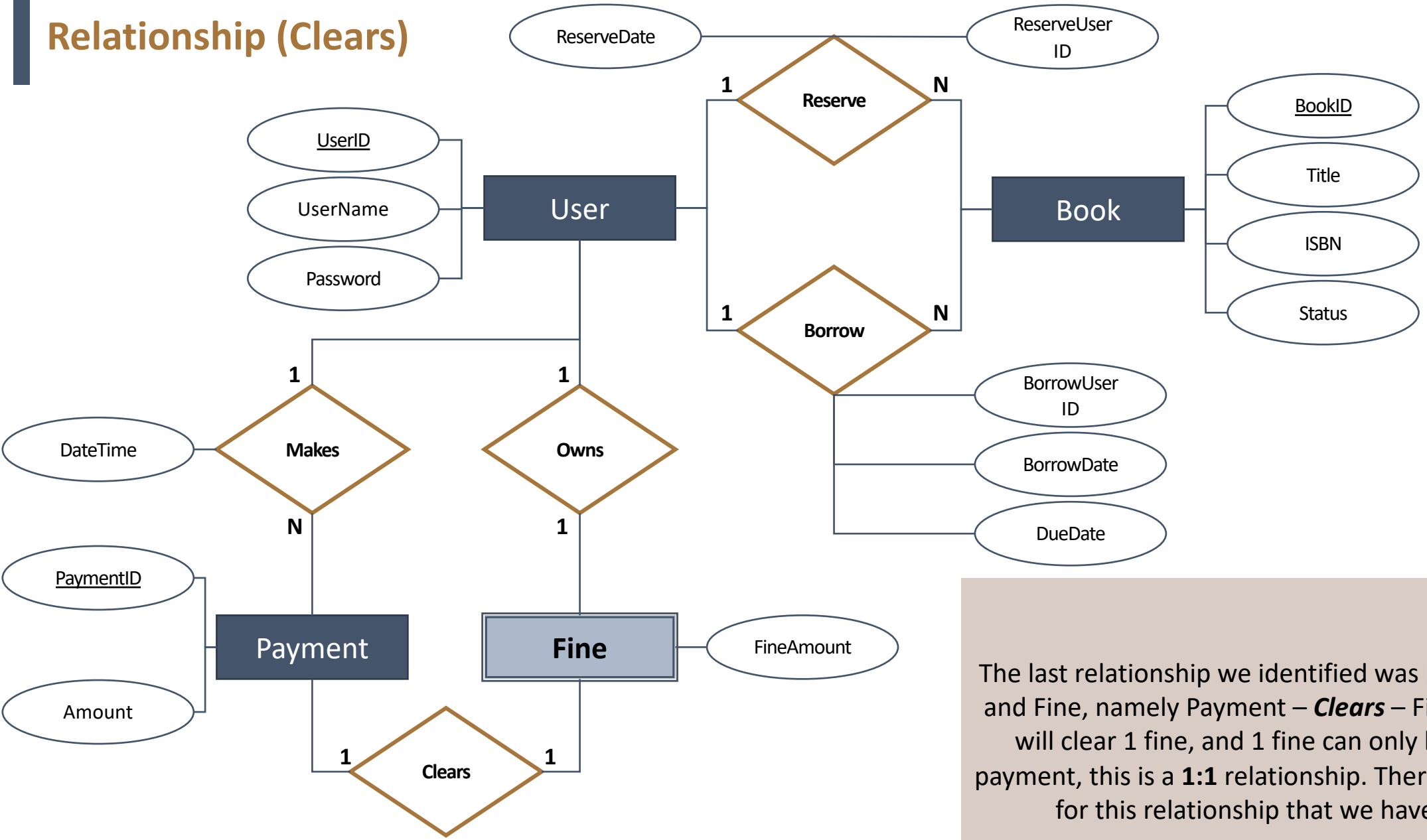
1) **DateTime**: The date and time for when the payment was made

Relationship (Owns)



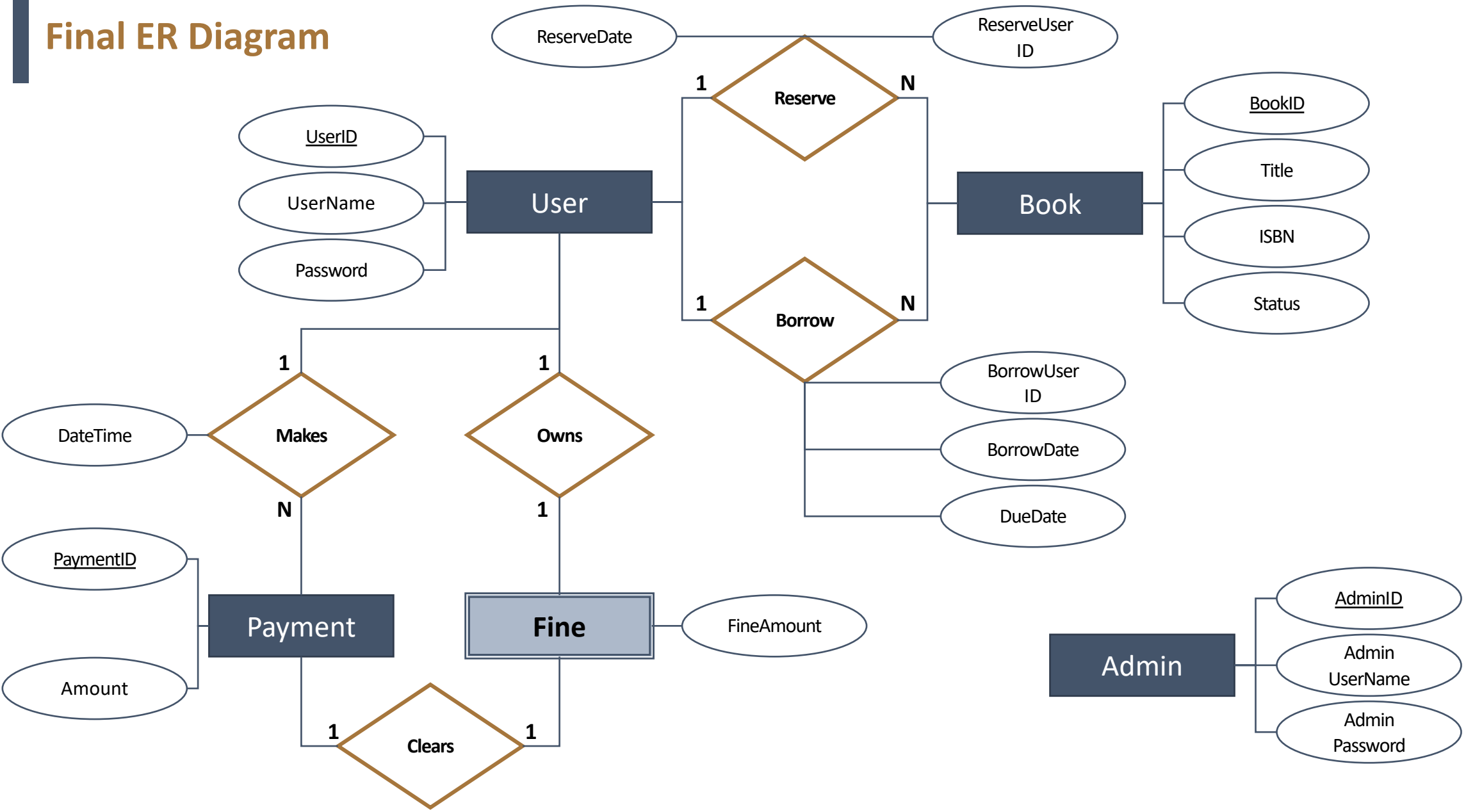
The fourth relationship we identified was between User and Fine, namely User – ***Owns*** – Fine. As 1 user can only own 1 fine, and 1 fine can only be owned by 1 user, this is a **1:1** relationship. There are no attributes for this relationship that we have identified.

Relationship (Clears)



The last relationship we identified was between Pyament and Fine, namely Payment – ***Clears*** – Fine. As 1 payment will clear 1 fine, and 1 fine can only be cleared by 1 payment, this is a **1:1** relationship. There are no attributes for this relationship that we have identified.

Final ER Diagram



Refining Relational Schema

With our strong entities and relationships identified, we now looked to refine the relational schema and reduce the number of attributes that we need to store. To do so, we first listed out the relations from entities and relations from relationships

Relations from Entities

User(UserID, UserName, Password)

Admin(AdminID, AdminUserName, AdminPassword)

Book(BookID, Title, ISBN, Status)

Payment(PaymentID, Amount)

Fine(UserID, FineAmount)

Relations from Relationships

Reserve(BookID, UserID, ReserveDate)

Borrow(BookID, UserID, BorrowDate, DueDate)

Makes(PaymentID, UserID, DateTime)

Owns and **Clears** has no attributes and hence are not included

Refining Relational Schema (Reserve)

Relations from Entities

User(UserID, UserName, Password)

Admin(AdminID, AdminUserName, AdminPassword)

Book(BookID, Title, ISBN, Status, **ReserveUserID, ReserveDate**)

Payment(PaymentID, Amount)

Fine(UserID, FineAmount)

Relations from Relationships

~~**Reserve**(BookID, UserID, ReserveUserID, ReserveDate)~~

Borrow(BookID, UserID, BorrowUserID, BorrowDate, DueDate)

Makes(PaymentID, UserID, DateTime)

As User – *Reserve* – Book is a 1:N relationship, it can be combined into the Book entity. The UserID is the same as the ReserveUserID and hence has been removed as well. The primary key for Book remains as BookID; however, now ReserveUserID acts as the foreign key to connect Book to User.

Refining Relational Schema (Borrow)

Relations from Entities

User(UserID, UserName, Password)

Admin(AdminID, AdminUserName, AdminPassword)

Book(BookID, Title, ISBN, Status, ReserveUserID, ReserveDate, **BorrowUserID, BorrowDate, DueDate**)

Payment(PaymentID, Amount)

Fine(UserID, FineAmount)

Relations from Relationships

~~**Reserve**(BookID, UserID, ReserveUserID, ReserveDate)~~

~~**Borrow**(BookID, UserID, BorrowUserID, BorrowDate, DueDate)~~

Makes(PaymentID, UserID, DateTime)

As User – *Borrow* – Book is a 1:N relationship, it can be combined into the Book entity. The UserID is the same as the BorrowUserID and hence has been removed as well. The primary key for Book remains as BookID; however, now BorrowUserID acts as the foreign key to connect Book to User.

Refining Relational Schema (Makes)

Relations from Entities

User(UserID, UserName, Password)

Admin(AdminID, AdminUserName, AdminPassword)

Book(BookID, Title, ISBN, Status, ReserveUserID, ReserveDate, BorrowUserID, BorrowDate, DueDate)

Payment(PaymentID, Amount, **UserID, DateTime**)

Fine(UserID, FineAmount)

Relations from Relationships

~~**Reserve**(BookID, UserID, ReserveUserID, ReserveDate)~~

~~**Borrow**(BookID, UserID, BorrowUserID, BorrowDate, DueDate)~~

~~**Makes**(PaymentID, UserID, DateTime)~~

As User – *Makes* – Payment is a 1:N relationship, where Payment's participation in User is total, it can be combined into the Payment entity. The primary key for Payment remains as PaymentID; however, now UserID acts as the foreign key to connect Payment to User.

Refining Relational Schema (3NF)

Relational Schema

User(UserID, UserName, Password)

Admin(AdminID, AdminUserName, AdminPassword)

Book(BookID, Title, ISBN, Status, ~~ReserveUserID, ReserveDate~~, BorrowUserID, BorrowDate, DueDate)

Payment(PaymentID, Amount, UserID, DateTime)

Fine(UserID, FineAmount)

Reserve(ReserveUserID, ReserveDate)

Looking at ReserveDate under book, we see that it is dependent on ReserveUserID and not the primary key BookID. Hence, we can normalize it by creating a new table named Reserve with ReserveUserID as the primary key.

Refining Relational Schema (3NF)

Relational Schema

User(UserID, UserName, Password)

Admin(AdminID, AdminUserName, AdminPassword)

Book(BookID, Title, ISBN, Status, ~~BorrowUserID, BorrowDate, DueDate~~)

Payment(PaymentID, Amount, UserID, DateTime)

Fine(UserID, FineAmount)

Reserve(ReserveUserID, ReserveDate)

Borrow(BorrowUserID, BorrowDate, DueDate)

Looking at BorrowDate and DueDate under book, we see that it is dependent on BorrowUserID and not the primary key BookID. Hence, we can normalize it by creating a new table named Borrow with BorrowUserID as the primary key.

Final Relational Schema

Relational Schema

User(UserID, UserName, Password)

Admin(AdminID, AdminUserName, AdminPassword)

Book(BookID, Title, ISBN, Status)

Payment(PaymentID, Amount, UserID, DateTime)

Fine(UserID, FineAmount)

Reserve(ReserveUserID, ReserveDate)

Borrow(BorrowUserID, BorrowDate, DueDate)

*These attributes have been written in lower case for in the SQL files, and FineAmount has been represented as amount for simplicity in SQL.