Database Design

We will begin the design of the Database by creating the appropriate Entity – Relationship diagram.

According to the requirements, there are five main concepts that should be represented:

- Books
- Authors
- Publishers
- Users
- Reviews

These five concepts should be represented as separate entity sets to avoid redundancy, update and deletion anomalies.

The central entity set is 'Book', which is connected with all other entity sets. The rest of the entity sets are independent from one another. However, the cardinality ration of each connection is different:

Relationship	Connected Entity Sets	Cardinality Ratio
CREATE	Book – Author	N:M
PUBLISH	Book – Publisher	N:1
ORDER	Book – User	N:M
SUBMIT	Book – Review	1:N

Most of the attributes that are described in the requirements can be directly related to one of the above entity sets. However, some of them should be related to one of the relationships instead, given that these attributes essentially describe a process, rather than an object:

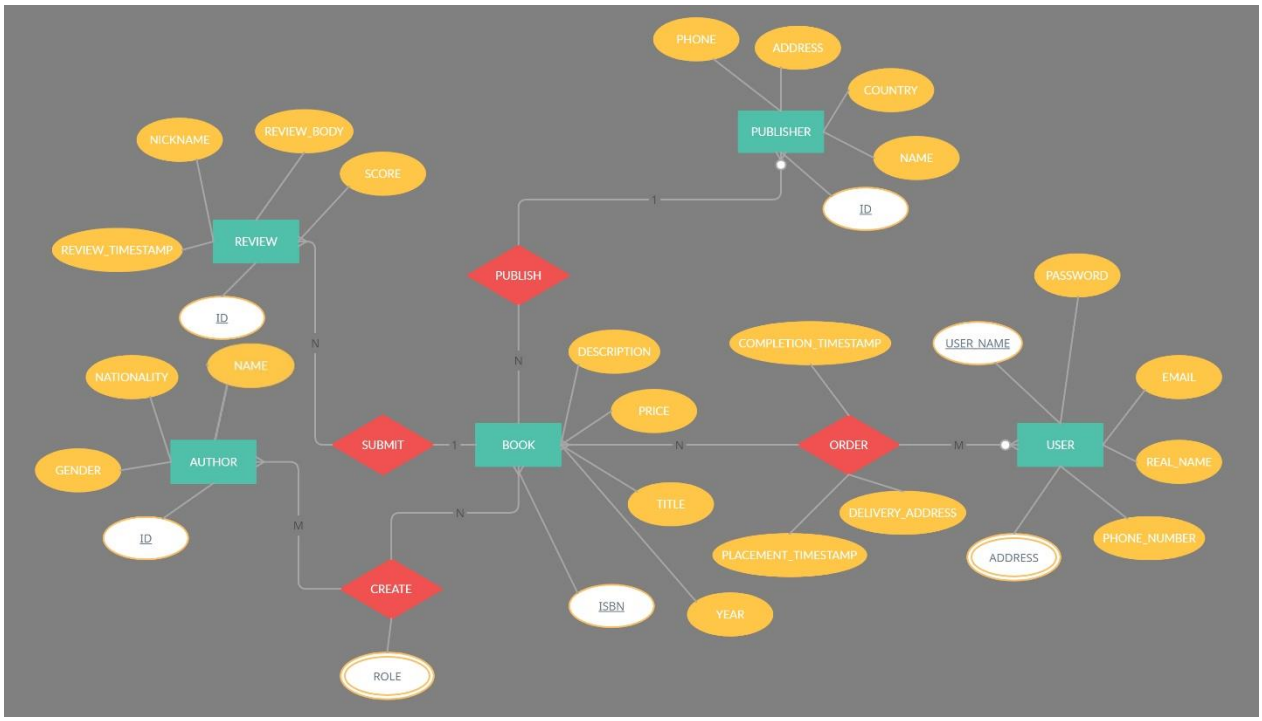
Relationship	Attribute	Comments
CREATE	ROLE (Multiple Values)	This attribute will describe all different roles undertaken by a specific author during the creation of a specific book
ORDER	COMPLETION_TIMESTAMP	Directly related with the order
ORDER	PLACEMENT_TIMESTAMP	Directly related with the order
ORDER	DELIVERY_ADDRESS	Directly related with the order (while each User can have multiple addresses)

Regarding the 'ROLE' attribute it's important to note the following. Based on the requirements, each author has a role in the creation of the book and multiple authors can have the same role. It's not explicitly mentioned that each author has **only one** role. In order to account for the possibility of an author participating for example in both 'Writing' and 'Illustrations', the initial thought was to handle the 'ROLE' attribute as a multi – value attribute. Given that an author having multiple roles does make sense logically, while the opposite is not defined clearly, we would rather be flexible than restrictive. However, after observing the data, it became clear that the cases of multiple roles for the same author were limited in our data set and that in those cases the role attribute included both roles of the author as a concatenated string (e.g. 'Writing, Illustrations' or 'Illustrations and Cover'), with no specific format. Consequently, the 'ROLE' attribute of the relationship 'CREATE' was handled as a single value attribute.

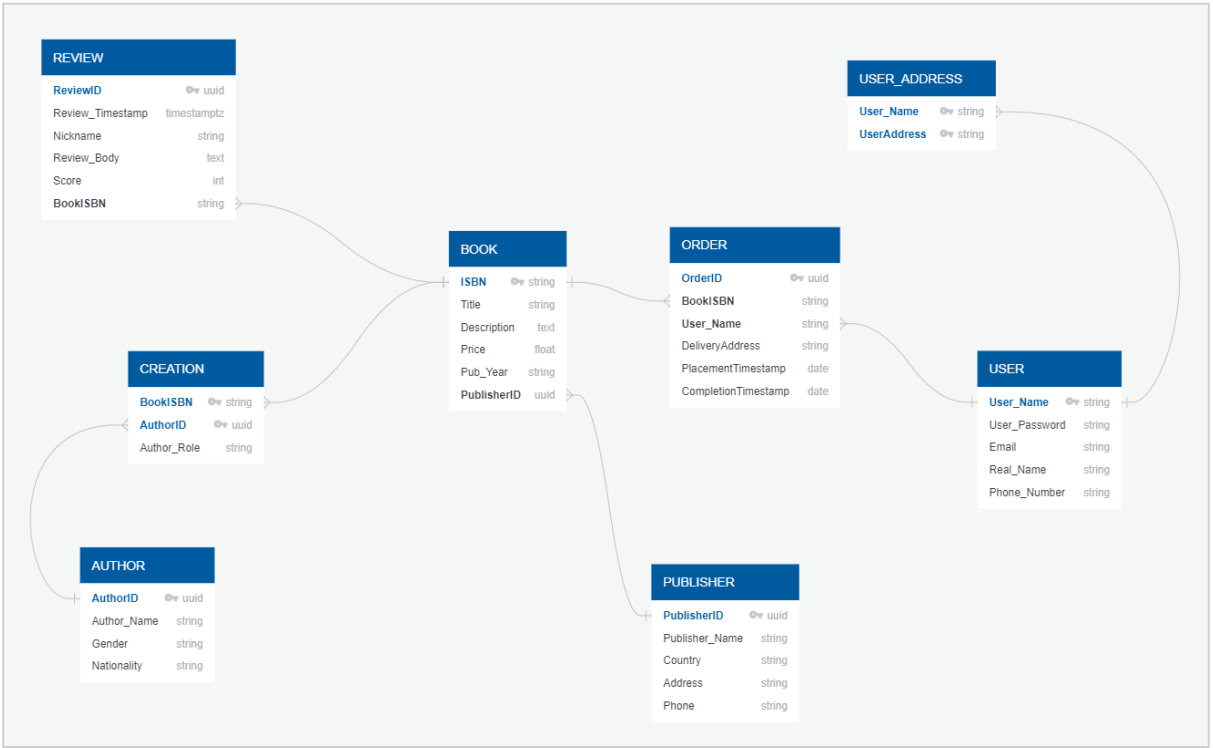
Moreover, we followed a mixed strategy to define the primary key for each entity set. In the case of 'BOOK', the 'ISBN' attribute was selected as a PK, while the 'USER_NAME' will be the primary key of USER'. In the cases of 'AUTHOR', 'REVIEW' and 'PUBLISHER' entity sets, no single attribute was fit to

be used as a PK. While it would be possible to use a combination of different attributes as PK in each case, this would complicate the connection of those entity sets with the central node, 'BOOK'. Therefore, we decided to introduce a new 'ID' attribute in each entity set to serve as PK.

With everything mentioned above in mind, the following ER-diagram was created:



The algorithm that was demonstrated in lecture 02 was used to transform the ER – diagram to the following relational schema:



The ‘USER_ADDRESS’ table was introduced by the algorithm to handle the multi – value attribute ‘UserAddress’. As mentioned earlier, “ROLE” was handled as a single value attribute and it was therefore included in the ‘CREATION’ table instead.

Moreover, there is a number of constraints that should be included in our schema based on requirements, intuition and practical needs for our database to be functional. These constraints, as well as the corresponding relations and attributes are described in the following table

Relation	Attribute	Data Type/Constraint
BOOK	ISBN	CHAR(10), PRIMARY KEY
	Title	VARCHAR(200), NOT NULL
	Description	TEXT
	Price	NUMERIC(5,2), NOT NULL
	Pub_Year	CHAR(4)
	PublisherID	UUID, FOREIGN KEY
PUBLISHER	PublisherID	UUID, PRIMARY KEY
	Publisher_Name	VARCHAR(100)
	Country	VARCHAR(30)
	Address	VARCHAR(60)
	Phone	VARCHAR(15)
REVIEW	ReviewID	UUID, PRIMARY KEY
	Review_Timestamp	TIMESTAMPTZ
	Nickname	VARCHAR(40)
	Review_Body	TEXT
	Score	INT, 1<Score<5
	BookISBN	CHAR(10), FOREIGN KEY
AUTHOR	AuthorID	UUID, PRIMARY KEY
	Author_Name	VARCHAR(100)
	Gender	CHAR(1), Is either ‘M’ or ‘F’
	Nationality	VARCHAR(15)
CREATION	AuthorID	UUID, FOREIGN KEY
	BookISBN	CHAR(10), FOREIGN KEY
	Author_Role	VARCHAR(100)
	(AuthorID, ISBN)	PRIMARY KEY
USER	User_Name	VARCHAR(30), PRIMARY KEY
	User_Password	VARCHAR(20), NOT NULL
	Email	VARCHAR(30), UNIQUE, NOT NULL
	Real_Name	VARCHAR(40), NOT NULL
	Phone_Number	VARCHAR(15), NOT NULL
USER_ADDRESS	User_Name	VARCHAR(30), FOREIGN KEY
	User_Address	VARCHAR(60), NOT NULL
	(User_Name, User_Address)	PRIMARY KEY
ORDER	Order_id	UUID, PRIMARY KEY
	BookISBN	CHAR(10), FOREIGN KEY
	User_Name	VARCHAR(30), FOREIGN KEY
	Delivery_Address	VARCHAR(60), NOT NULL
	Placement_Timestamp	TIMESTAMPTZ
	Completion_Timestamp	TIMESTAMPTZ, DEFAULT NULL

'NOT NULL' constraints:

Some of the attributes were declared to be 'NOT NULL', given that they are essential to the operation of the e-shop. For example, a customer should be able to see what book he/she is going to buy and what is the amount of money that it costs. It is, therefore, not acceptable for an e-shop to include a book in its database, without the book's ISBN, Title and Price. Moreover, we need to be able to identify our users and occasionally contact them in order to deliver our services. Therefore, the 'User_Name', 'User_Password', 'Email', 'Real_Name' and 'Phone_Number' attributes of the 'USER' relation cannot be NULL. Finally, the company should be able to deliver the books to the customer's residential address. This implies that both the 'User_Address' attribute of the 'User_Address' relation and the 'Delivery_Address' attribute of the 'ORDER' relation cannot be NULL.