**Student Number:** 40214858

**Submission Date:** 02 December 2022

**Module Number:** SET11101

Database Systems  
Stage 2

Table of Contents

[Task D. Create tables. 4](#_Toc120643121)

[Task E. Insert statements. 5](#_Toc120643122)

[Task F: SQL statements. 11](#_Toc120643123)

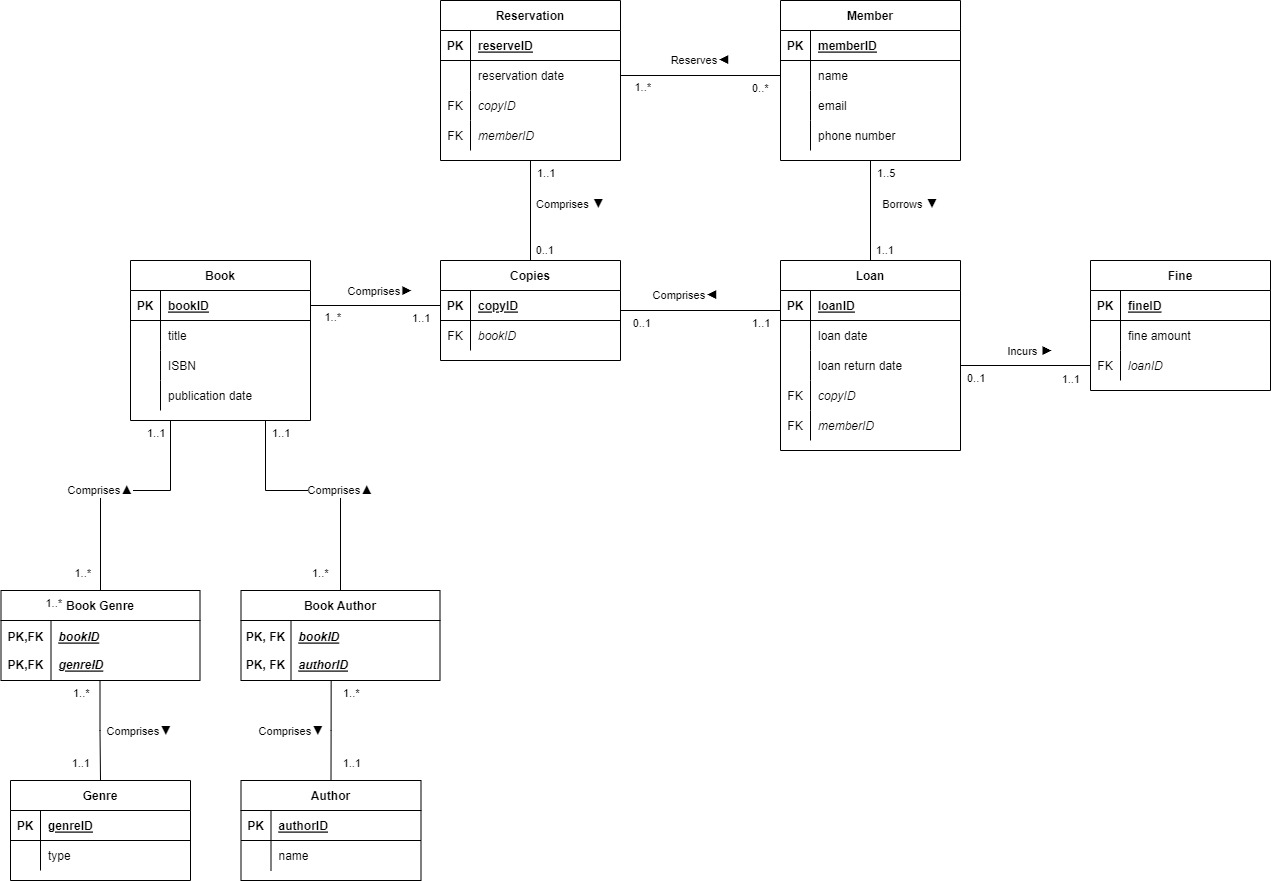
[Query 1. 11](#_Toc120643124)

[Query 2. 12](#_Toc120643125)

[Query 3 12](#_Toc120643126)

[Task G. Constraints and Business Rules for the given scenario. 14](#_Toc120643127)

The UML diagram was changed based on the feedback provided from coursework 1 and is displayed in Figure 1. An additional table was added that contains the copies of the books and linked to the loan, reservation and book. The Fine table was kept and used for the third query, but can be realistically removed and accounted for as a query.

****

**Figure 1.** Amended Entity-Relationship UML diagram.

# Task D. Create tables.

The following section contains the tables required to implement the current database. The primary and foreign keys have been assigned with proper datatypes for each attribute.

**CREATE** **TABLE** **Member**(

memberID **INT** **PRIMARY** **KEY**,

**name** **CHAR**(150),

email **VARCHAR**(255),

phone**Number** **INT**

);

**CREATE** **TABLE** **Genre**(

genreID **VARCHAR**(150) **PRIMARY** **KEY**,

**Type** **VARCHAR**(150)

);

**CREATE** **TABLE** **Author**(

authorID **VARCHAR**(150) **PRIMARY** **KEY**,

**name** **VARCHAR**(150)

);

**CREATE** **TABLE** **Book**(

bookID **VARCHAR**(150) **PRIMARY** **KEY**,

**title** **VARCHAR**(150),

ISBN **CHAR**(17),

publicationDate **DATE**

);

**CREATE** **TABLE** BookGenre(

bookID **VARCHAR**(150),

genreID **VARCHAR**(150),

**PRIMARY** **KEY** (genreID, bookID),

**FOREIGN** **KEY** (genreID) **REFERENCES** Genre (genreID),

**FOREIGN** **KEY** (bookID) **REFERENCES** Book (bookID)

);

**CREATE** **TABLE** BookAuthor(

authorID **VARCHAR**(150),

bookID **VARCHAR**(150),

**PRIMARY** **KEY** (authorID, bookID),

**FOREIGN** **KEY** (authorID) **REFERENCES** Author (authorID),

**FOREIGN** **KEY** (bookID) **REFERENCES** Book (bookID)

);

**CREATE** **TABLE** **Copies**(

copyID **VARCHAR**(150) **PRIMARY** **KEY**,

bookID **VARCHAR**(150),

**FOREIGN** **KEY** (bookID) **REFERENCES** Book (bookID)

);

**CREATE** **TABLE** **Loan**(

loanID **VARCHAR**(150) **PRIMARY** **KEY**,

loanDate **DATE**,

loanReturnDate **DATE**,

copyID **VARCHAR**(150),

memberID **INT,**

**FOREIGN** **KEY** (memberID) **REFERENCES** Member (memberID),

**FOREIGN** **KEY** (copyID) **REFERENCES** Copies (copyID)

);

**CREATE** **TABLE** **Reservation**(

reserveID **VARCHAR**(150) **PRIMARY** **KEY**,

reserveDate **DATE**,

copyID **VARCHAR**(150),

memberID **INT,**

**FOREIGN** **KEY** (memberID) **REFERENCES** Member (memberID),

**FOREIGN** **KEY** (copyID) **REFERENCES** Copies (copyID)

);

**CREATE** **TABLE** **Fine**(

fineID **VARCHAR**(150) **PRIMARY** **KEY**,

fineAmount **DECIMAL(10,2),**

loanID **VARCHAR**(150),

**FOREIGN** **KEY** (loanID) **REFERENCES** Loan (loanID)

);

# Task E. Insert statements.

The following section contains insert data that can be copied and inserted to the database. This data was expanded to account for the new table of copies and adjusted to account for the queries required in the following section.

**INSERT** **INTO** **Member**

**VALUES** (20220001, 'John White', 'johnwhite@gmail.com', 755334455),

(20220002, 'John Brown', 'johnbrown@gmail.com', 732412332),

(20220003, 'John Green', 'johngreen@gmail.com', 788556543),

(20220004, 'John Black', 'johnblack@gmail.com', 723746776),

(20220005, 'John Red', 'johnred@gmail.com', 799880706),

(20220006, 'John Yellow', 'johnyellow@gmail.com', 732344568),

(20220007, 'John Purple', 'johnpurple@gmail.com', 753450677),

(20220008, 'John Blue', 'johnblue@gmail.com', 724345677),

(20220009, 'John Pink', 'johnpink@gmail.com', 766677888),

(20220010, 'John Grey', 'johngrey@gmail.com', 799900077),

(20220011, 'Jack White', 'jackwhite@gmail.com', 723493243),

(20220012, 'Jack Brown', 'jackbrown@gmail.com', 733344053),

(20220013, 'Jack Green', 'jackgreen@gmail.com', 711111112),

(20220014, 'Jack Black', 'jackblack@gmail.com', 755555756),

(20220015, 'Jack Red', 'jackred@gmail.com', 788888889),

(20220016, 'Jack Yellow', 'jackyellow@gmail.com', 799999000),

(20220017, 'Jack Purple', 'jackpurple@gmail.com', 744445555),

(20220018, 'Jack Blue', 'jackblue@gmail.com', 789899000),

(20220019, 'Jack Pink', 'jackpink@gmail.com', 799988899),

(20220020, 'Jack Grey', 'jackgrey@gmail.com', 733332224);

**INSERT** **INTO** Genre

**VALUES** ('GE2000', 'Action and adventure'),

('GE2001', 'Alternate history'),

('GE2002', 'Anthology'),

('GE2003', 'Home and garden'),

('GE2004', 'Children'),

('GE2005', 'Classic'),

('GE2006', 'Comic book'),

('GE2007', 'Cookbook'),

('GE2008', 'Crime'),

('GE2009', 'Drama'),

('GE2010', 'Fairytale'),

('GE2011', 'Fantasy'),

('GE2012', 'Graphic novel'),

('GE2013', 'Historical fiction'),

('GE2014', 'Horror'),

('GE2015', 'Mystery'),

('GE2016', 'Paranormal romance'),

('GE2017', 'Picture book'),

('GE2018', 'Philosophy'),

('GE2019', 'Health/fitness');

**INSERT** **INTO** Author

**VALUES** ('AU1000', 'William Shakespeare'),

('AU1001', 'Agatha Christie'),

('AU1002', 'Barbara Cartland'),

('AU1003', 'Danielle Steel'),

('AU1004', 'Harold Robbins'),

('AU1005', 'Georges Simenon'),

('AU1006', 'Enid Blyton'),

('AU1007', 'Sidney Sheldon'),

('AU1008', 'Eiichiro Oda'),

('AU1009', 'J. K. Rowling'),

('AU1010', 'Gilbert Patten'),

('AU1011', 'Dr. Seuss'),

('AU1012', 'Akira Toriyama'),

('AU1013', 'Leo Tolstoy'),

('AU1014', 'Corín Tellado'),

('AU1015', 'Dean Koontz'),

('AU1016', 'Jackie Collins'),

('AU1017', 'Horatio Alger'),

('AU1018', 'Nora Roberts'),

('AU1019', 'R. L. Stine'),

('AU1020', 'Barbara Shakespeare');

**INSERT** **INTO** Book

**VALUES** ('BO0001', 'A Christmas Carol', '123-456-789-0001', '2020-05-01'),

('BO0002', 'Animal Farm', '123-456-789-0002', '2020-05-02'),

('BO0003', 'Dracula', '123-456-789-0003', '2020-05-03'),

('BO0004', 'Fahrenheit 451', '123-456-789-0004', '2020-05-04'),

('BO0005', 'Jane Eyre', '123-456-789-0005', '2020-05-05'),

('BO0006', 'Little Women ', '123-456-789-0006', '2020-05-06'),

('BO0007', 'Moby Dick', '123-456-789-0007', '2020-05-07'),

('BO0008', 'Nineteen Eighty-Four', '123-456-789-0008', '2020-05-08'),

('BO0009', 'Of Mice and Men', '123-456-789-0009', '2020-05-09'),

('BO0010', 'One Hundred Years of Solitude', '123-456-789-0010', '2020-05-10'),

('BO0011', 'The Catcher in the Rye', '123-456-789-0011', '2020-05-11'),

('BO0012', 'The Great Gatsby', '123-456-789-0012', '2020-05-12'),

('BO0013', 'Persuasion', '123-456-789-0013', '2020-05-13'),

('BO0014', 'Pride & Prejudice', '123-456-789-0014', '2020-05-14'),

('BO0015', 'The Adventures of Tom Sawyer', '123-456-789-0015', '2020-05-15'),

('BO0016', 'The Phantom of the Opera', '123-456-789-0016', '2020-05-16'),

('BO0017', 'The Picture of Dorian Gray', '123-456-789-0017', '2020-05-17'),

('BO0018', 'The Three Musketeers', '123-456-789-0018', '2020-05-18'),

('BO0019', 'The War of the Worlds', '123-456-789-0019', '2020-05-19'),

('BO0020', 'The Wind in the Willows', '123-456-789-0020', '2020-05-20'),

('BO0021', 'A Christmas Carol', '123-456-789-0001', '2020-05-01'),

('BO0022', 'A Christmas Carol', '123-456-789-0001', '2020-05-01'),

('BO0023', 'Animal Farm', '123-456-789-0002', '2020-05-02'),

('BO0024', 'Animal Farm', '123-456-789-0002', '2020-05-02'),

('BO0025', 'Animal Farm', '123-456-789-0002', '2020-05-02'),

('BO0026', 'Animal Farm', '123-456-789-0002', '2020-05-02'),

('BO0027', 'Dracula', '123-456-789-0003', '2020-05-03'),

('BO0028', 'Fahrenheit 451', '123-456-789-0004', '2020-05-04'),

('BO0029', 'Fahrenheit 451', '123-456-789-0004', '2020-05-04'),

('BO0030', 'Fahrenheit 451', '123-456-789-0004', '2020-05-04');

**INSERT** **INTO** BookGenre

**VALUES** ('BO0001', 'GE2000'),

('BO0002', 'GE2001'),

('BO0003', 'GE2002'),

('BO0004', 'GE2003'),

('BO0005', 'GE2004'),

('BO0006', 'GE2005'),

('BO0007', 'GE2006'),

('BO0008', 'GE2007'),

('BO0009', 'GE2008'),

('BO0010', 'GE2009'),

('BO0011', 'GE2010'),

('BO0012', 'GE2011'),

('BO0013', 'GE2012'),

('BO0014', 'GE2013'),

('BO0015', 'GE2014'),

('BO0016', 'GE2015'),

('BO0017', 'GE2016'),

('BO0018', 'GE2017'),

('BO0019', 'GE2018'),

('BO0020', 'GE2019'),

('BO0021', 'GE2001'),

('BO0022', 'GE2002'),

('BO0023', 'GE2016'),

('BO0024', 'GE2017'),

('BO0025', 'GE2018'),

('BO0026', 'GE2000'),

('BO0027', 'GE2000'),

('BO0028', 'GE2000'),

('BO0029', 'GE2001'),

('BO0030', 'GE2001');

**INSERT** **INTO** BookAuthor

**VALUES** ('AU1000', 'BO0001'),

('AU1001', 'BO0002'),

('AU1002', 'BO0003'),

('AU1003', 'BO0004'),

('AU1004', 'BO0005'),

('AU1005', 'BO0006'),

('AU1006', 'BO0007'),

('AU1007', 'BO0008'),

('AU1008', 'BO0009'),

('AU1009', 'BO0010'),

('AU1010', 'BO0011'),

('AU1011', 'BO0012'),

('AU1012', 'BO0013'),

('AU1013', 'BO0014'),

('AU1014', 'BO0015'),

('AU1015', 'BO0016'),

('AU1016', 'BO0017'),

('AU1017', 'BO0018'),

('AU1018', 'BO0019'),

('AU1019', 'BO0020'),

('AU1020', 'BO0001');

**INSERT** **INTO** Copies

**VALUES** ('BO0004001', 'BO0004'),

('BO0004002', 'BO0004'),

('BO0004003', 'BO0004'),

('BO0004004', 'BO0004'),

('BO0001001', 'BO0001'),

('BO0001002', 'BO0001'),

('BO0001003', 'BO0001'),

('BO0002001', 'BO0002'),

('BO0002002', 'BO0002'),

('BO0002003', 'BO0002'),

('BO0002004', 'BO0002'),

('BO0002005', 'BO0002'),

('BO0003001', 'BO0003'),

('BO0003002', 'BO0003'),

('BO0005001', 'BO0005'),

('BO0006001', 'BO0006'),

('BO0007001', 'BO0007'),

('BO0008001', 'BO0008'),

('BO0009001', 'BO0009'),

('BO0010001', 'BO0010'),

('BO0011001', 'BO0011'),

('BO0012001', 'BO0012'),

('BO0013001', 'BO0013'),

('BO0014001', 'BO0014'),

('BO0015001', 'BO0015'),

('BO0016001', 'BO0016'),

('BO0017001', 'BO0017'),

('BO0018001', 'BO0018'),

('BO0019001', 'BO0019'),

('BO0020001', 'BO0020'),

('BO0021001', 'BO0021'),

('BO0022001', 'BO0022'),

('BO0023001', 'BO0023'),

('BO0024001', 'BO0024'),

('BO0025001', 'BO0025'),

('BO0026001', 'BO0026'),

('BO0027001', 'BO0027'),

('BO0028001', 'BO0028'),

('BO0029001', 'BO0029'),

('BO0030001', 'BO0030');

**INSERT** **INTO** Loan

**VALUES** ('LO-01092025BO0001','2022-09-01', '2025-09-14', 'BO0001001', 20220005),

('LO-03092025BO0002', '2022-09-03', '2025-09-13', 'BO0002001', 20220006),

('LO-15092025BO0001', '2022-09-15', '2025-09-18', 'BO0001001', 20220012),

('LO-14092025BO0002','2022-09-14', '2025-09-16', 'BO0002001', 20220013),

('LO-17092025BO0002', '2022-09-17', '2022-09-19', 'BO0002001', 20220014),

('LO-01102025BO0001', '2022-10-01', '2022-10-07', 'BO0001001', 20220001),

('LO-01102025BO0002', '2022-10-01', '2022-10-12', 'BO0002001', 20220002),

('LO-02092025BO0012','2022-10-02', '2022-10-07', 'BO0012001', 20220016),

('LO-05102025BO0006', '2022-10-06', '2022-10-08', 'BO0006001', 20220006),

('LO-06102025BO0007', '2022-10-07', '2022-10-10', 'BO0007001', 20220007),

('LO-07102025BO0008', '2022-10-08', '2022-10-11', 'BO0008001', 20220008),

('LO-07102025BO0009', '2022-10-09', '2022-10-12', 'BO0009001', 20220009),

('LO-08102025BO0012', '2022-10-12', '2022-10-15', 'BO0012001', 20220012),

('LO-12102025BO0015', '2022-10-15', '2022-10-18', 'BO0015001', 20220015),

('LO-15102025BO0018', '2022-10-18', '2022-10-18', 'BO0018001', 20220018);

**INSERT** **INTO** Loan (loanID, loan**Date**, copyID, memberID)

**VALUES** ('LO-01102025BO0003', '2022-11-25', 'BO0003001', 20220003),

('LO-03102022BO0003', '2022-11-26', 'BO0003002', 20220007),

('LO-03102025BO0004', '2022-10-04', 'BO0004001', 20220004),

('LO-04102025BO0005', '2022-10-05', 'BO0005001', 20220005),

('LO-07102025BO0010', '2022-10-10', 'BO0010001', 20220010),

('LO-07102025BO0011', '2022-10-11', 'BO0011001', 20220011),

('LO-10102025BO0013', '2022-10-13', 'BO0013001', 20220013),

('LO-11102025BO0014', '2022-10-14', 'BO0014001', 20220014),

('LO-13102025BO0016', '2022-10-16', 'BO0016001', 20220016),

('LO-14102025BO0017', '2022-10-17', 'BO0017001', 20220017),

('LO-15102025BO0019', '2022-10-19', 'BO0019001', 20220019),

('LO-15102025BO0020','2022-10-20', 'BO0020001', 20220020);

**INSERT** **INTO** Reservation

**VALUES** ('RE-BO000300120220001', '2022-11-26', 'BO0003001', 20220001),

('RE-BO000400120220002', '2022-10-05', 'BO0004001', 20220002),

('RE-BO001000120220001', '2022-10-11', 'BO0010001', 20220001),

('RE-BO001700120220007', '2022-10-18', 'BO0017001', 20220007),

('RE-BO001900120220007', '2022-10-20', 'BO0019001', 20220007);

**INSERT** **INTO** Fine

**VALUES** ('FI092025001', 6, 'LO-02092025BO0012'),

('FI102025001', 7.5, 'LO-01102025BO0003'),

('FI102025002', 6.5, 'LO-03102025BO0004'),

('FI102025003', 6, 'LO-04102025BO0005'),

('FI102025004', 4.5, 'LO-07102025BO0010'),

('FI102025005', 4.5, 'LO-07102025BO0011'),

('FI102025006', 3, 'LO-10102025BO0013'),

('FI102025007', 2.5, 'LO-11102025BO0014'),

('FI102025008', 1.5, 'LO-13102025BO0016'),

('FI102025009', 1, 'LO-14102025BO0017');

# Task F: SQL statements.

This section addresses the required queries and the statements that were constructed to implement them.

Query 1. **List all books in order of how often they have been checked out.**

The following statement selects the title by joining the book, loan and copies table to group them and display how often the book is checked out. Figure 2 displays the intended output.

**SELECT** Title, **COUNT**(\*) AS Total

**FROM** Book **JOIN** Copies **ON** Book.bookID = Copies.bookID

**JOIN** Loan **ON** Copies.copyID = Loan.copyID

**GROUP** **BY** Book.bookID

**ORDER** **BY** **COUNT**(\*) **DESC**;

Graphical user interface, application, table

Description automatically generated

**Figure 2.** Query 1 output.

Query 2. **Find the books where all copies are currently checked out, but where there are no reservations outstanding for the book.**

In the statement below the WITH syntax is used to create two temporary tables and match between them to extract the data required for the query. The output is displayed in Figure 3.

**WITH** TEMPTABLE1

**AS**

(**SELECT** Title, **COUNT**(\*) **AS** numOfCopies1

**FROM** Book **JOIN** Copies **ON** Book.bookID = Copies.bookID

**JOIN** Loan **ON** Copies.copyID = Loan.copyID

**left** **JOIN** Reservation **ON** Reservation.copyID = Copies.copyID

**WHERE** Loan.loanReturnDate **IS** **NULL** **AND** Reservation.copyID = Loan.copyID

**GROUP** **BY** Book.bookID),

TEMPTABLE2

**AS**

(**SELECT** Title, **COUNT**(\*) **AS** numOfCopies2

**FROM** Book **JOIN** Copies **ON** Book.bookID = Copies.bookID

**GROUP** **BY** Book.bookID)

**SELECT** TEMPTABLE2.Title

**FROM** TEMPTABLE1 **JOIN** TEMPTABLE2 **ON** TEMPTABLE1.Title = TEMPTABLE2.Title

**WHERE** TEMPTABLE2.numOfcopies2 > 1;

Graphical user interface, text, application

Description automatically generated

**Figure 3.** Query 2 output.

Query 3**. Find all borrowers who have outstanding fines totalling more than £5. Don’t include any fines that have already been paid.**

In the figure 4 we can see the output for the third query. This statement uses the current date to calculate the fine amount and display the fines over £5. This statement disregards the Fine table, meaning we could remove it from the database.

**Select** **NAME**, (**DATEDIFF**(**CURRENT\_DATE**, Loan.loanDate) -21) \* 0.5 **AS** Fine\_Amount

**From** **Member** **JOIN** Loan **ON** **Member**.memberID = Loan.memberID

**WHERE** Loan.loanReturnDate **IS** **NULL**

**HAVING** Fine\_Amount > 5

Table

Description automatically generated

**Figure 4.** Query 3(a) output.

The following statement uses the Fine table that calculates the fine amount, but this is hard coded in this database and doesn’t account for the current date. The output is shown in Figure 5.

**Select** **name**, fineAmount

**From** **Member** **JOIN** Loan **ON** **Member**.memberID = Loan.memberID

**JOIN** Fine **ON** Loan.loanID = Fine.loanID

**Where** fineAmount > 5 **AND** Loan.loanReturnDate **IS** **NULL**;

Table

Description automatically generated

**Figure 5.** Query 3(b) output.

# Task G. Constraints and Business Rules for the given scenario.

The given database for the scenario of a county library was planned and implemented as per the requirements of the coursework. The system life cycle was not addressed in this coursework and would need to be developed for the database to be implemented. No consideration of the process of maintaining it was discussed.

Further tables and entity relationship would need to be researched to implement publisher information and other forms of borrowing, such as online reservations.

Many of the constraints were addressed in the normalisation process in the previous coursework. Additionally, the primary and foreign keys defied in the attributes had to be accounted for when implementing the database. This meant that a certain table could not be created prior to the existence of another. For example due to the foreign key restraint in the Loan table, the Member and Copies table had to be implemented prior to Loan table. While implemented the database these constraints were accounted for and adjusted to allow the database to run and insert data.

Aspects of security, data protection and access where not addressed in the current database system. When members join their information needs to stored securely and follow data protection guidelines. The scenario did not consider people wanting to buy any books that can be further developed in the future.

Additionally, the database designed was implemented with mock data that was designed to produce the aforementioned queries. This could mean lead to problems with data quality in the relational model.

Furthermore, the database also did not account for other forms of books or media sources and would need to be adjusted in the future to develop a more robust and reliable database.