Projekt Data-Mart-Erstellung in SQL

Phase 2: Erarbeitungs-/Reflexionsphase

Datum: 31.05.2025

Implementierung der Datenbank auf Grundlage des erstellten ER-Modells

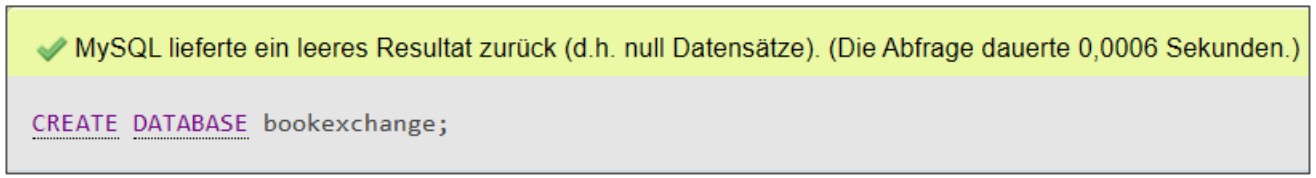
Im Folgenden wird die Erstellung der Datenbank mit den einzelnen Entitäten dargestellt. Dazu wird der CREATE Befehl für die Tabellenerstellung, als auch das INSERT Statement zum Einfügen der Daten als SQL-Ausdruck demonstriert.
Veranschaulicht wird dies mit dem Ergebnis der Befehle, wie es in der Datenbank zu finden ist.
Ergänzende SQL-Befehle und Erläuterungen sind je nach Notwendigkeit und Bedarf ebenfalls beschrieben.

bookexchange - Datenbank

Datenbank erstellen

```
CREATE DATABASE bookexchange;
```

Ergebnis



Country - Entität

Tabelle erstellen

```
CREATE TABLE Country (
    CountryID INT AUTO_INCREMENT,
    Name VARCHAR(100),
    PRIMARY KEY (CountryID)
);
```

Testdaten einfügen

```
INSERT INTO Country (Name) VALUES
    ('Deutschland')
;
```

Bei dieser Entität wurde auf 10 Einträge verzichtet, da aufgrund regionaler Nutzung die Ländervielfalt sehr eingeschränkt ist. Bei Ausweitung der Nutzung sind auch mehr Länder möglich.

Ergebnistabelle

CountryID	Name
1	Deutschland

City - Entität

Tabelle erstellen

```
CREATE TABLE City (
    CityID INT AUTO_INCREMENT,
    Postcode VARCHAR(10),
    Name VARCHAR(100),
    CountryID INT NOT NULL,
    PRIMARY KEY (CityID),
    FOREIGN KEY (CountryID) REFERENCES Country(CountryID)
    ON DELETE RESTRICT ON UPDATE CASCADE
);
```

Testdaten einfügen

```
INSERT INTO City (Postcode, Name, CountryID) VALUES
    (72250,'Freudenstadt',1),
    (72280,'Dornstetten',1),
    (72160,'Horb am Neckar',1),
    (72290,'Loßburg',1),
    (72293,'Glatten',1),
    (72275,'Alpirsbach',1),
    (72178,'Waldachtal',1),
    (72285,'Pfalzgrafenweiler',1),
    (72296,'Schopfloch',1),
    (72186,'Empfingen',1)
;
```

Ergebnistabelle

CityID	Postcode	Name	CountryID
1	72250	Freudenstadt	1
2	72280	Dornstetten	1
3	72160	Horb am Neckar	1
4	72290	Loßburg	1
5	72293	Glatten	1
6	72275	Alpirsbach	1
7	72178	Waldachtal	1
8	72285	Pfalzgrafenweiler	1
9	72296	Schopfloch	1
10	72186	Empfingen	1

Address - Entität

Tabelle erstellen

```
CREATE TABLE Address (
    AddressID INT AUTO_INCREMENT,
    Street VARCHAR(100),
    HouseNumber VARCHAR(5),
    CityID INT NOT NULL,
    PRIMARY KEY (AddressID),
    FOREIGN KEY (CityID) REFERENCES City(CityID)
    ON DELETE RESTRICT ON UPDATE CASCADE
);
```

Testdaten einfügen

```
INSERT INTO Address (Street, HouseNumber, CityID) VALUES
    ('Reichsstraße',40,1),
    ('Silberstraße',17,2),
    ('Mühlener Torweg',23,3),
    ('Oberer Schulweg',23,4),
    ('Lombacherstraße',51,5),
    ('Klosterplatz',2,6),
    ('Hauptstraße',35,7),
    ('Zeißstraße',6,8),
    ('Dornstetterstraße',21,9),
    ('Im Auchtert',5,10)
;
```

Ergebnistabelle

AddressID	Street	HouseNumber	CityID
1	Reichsstraße	40	1
2	Silberstraße	17	2
3	Mühlener Torweg	23	3
4	Oberer Schulweg	23	4
5	Lombacherstraße	51	5
6	Klosterplatz	2	6
7	Hauptstraße	35	7
8	Zeißstraße	6	8
9	Dornstetterstraße	21	9
10	Im Auchtert	5	10

Logindata - Entität

Tabelle erstellen

```
CREATE TABLE Logindata (
    LoginID INT AUTO_INCREMENT,
    EmailAddress VARCHAR(50),
    Passwort VARCHAR(300),
    PRIMARY KEY (LoginID)
);
```

Testdaten einfügen

```
INSERT INTO Logindata (EmailAddress, Passwort) VALUES
    ('alejandro.maier91@gmx.de',PASSWORD('123456')),
    ('f.alhassan87@web.de', PASSWORD('password')),
    ('yuki_tanaka22@t-online.de', PASSWORD('12345678')),
    ('liam.oconnor@gmx.de', PASSWORD('password1')),
    ('nora.flaig@web.de', PASSWORD('iloveyou')),
    ('levi.becker@freenet.de', PASSWORD('admin')),
    ('z.kowalska92@t-online.de', PASSWORD('Qwerty123')),
    ('elia.schlag@web.de', PASSWORD('Keyboard')),
    ('chenwei79@gmx.de', PASSWORD('Kalender2025')),
    ('hans.mueller@mail.de', PASSWORD('Rentier'))
;
```

Ergebnistabelle

LoginID	EmailAddress	Passwort
1	alejandro.maier91@gmx.de	*6BB4837EB74329105EE4568DDA7DC67ED2CA2AD9
2	f.alhassan87@web.de	*2470C0C06DEE42FD1618BB99005ADCA2EC9D1E19
3	yuki_tanaka22@t-online.de	*84AAC12F54AB666ECFC2A83C676908C8BBC381B1
4	liam.oconnor@gmx.de	*668425423DB5193AF921380129F465A6425216D0
5	nora.flaig@web.de	*CFBF459D9D6057BC2A85477A38327B96F06B1597
6	levi.becker@freenet.de	*4ACFE3202A5FF5CF467898FC58AAB1D615029441
7	z.kowalska92@t-online.de	*57B45B99D00B420E8C241804DE72C06746CBD3FD
8	elia.schlag@web.de	*BE71F5F97A7B78E55610E58F699A35E8F82783A0
9	chenwei79@gmx.de	*B29BC36FE811FEBD6370DF7902F59A24B8706E5F
10	hans.mueller@mail.de	*841425A6F8B904DA1722607D9EE9E59986B73A39

Die Einträge zum Attribut Passwort wurden mit der Funktion PASSWORD () verschlüsselt, sodass die Daten in der Tabellenansicht nicht ersichtlich sind.

Userdata - Entität

Tabelle erstellen

```
CREATE TABLE Userdata (
    UserID INT AUTO_INCREMENT,
    FirstName VARCHAR(100),
    LastName VARCHAR(100),
    MobileNumber VARCHAR(50),
    LoginID INT NOT NULL,
    AddressID INT NOT NULL,
    RatingReceiver DECIMAL(3,2),
    RatingLender DECIMAL(3,2),
    PRIMARY KEY (UserID),
    FOREIGN KEY (LoginID) REFERENCES Logindata(LoginID)
    ON DELETE RESTRICT ON UPDATE CASCADE,
    FOREIGN KEY (AddressID) REFERENCES Address(AddressID)
    ON DELETE RESTRICT ON UPDATE CASCADE
);
```

Entität wurde von User auf Userdata geändert, da User in phpMyAdmin ein SQL Begriff ist

Testdaten einfügen

```
INSERT INTO Userdata (FirstName, LastName,
MobileNumber, LoginID, AddressID) VALUES
    ('Alejandro', 'Maier', '+49 1571 2345678', 1, 1),
    ('Fatima', 'Al-Hassan', '+49 1522 98766543', 2, 2),
    ('Yuki', 'Tanaka', '+49 1573 4567890', 3, 3),
    ('Liam', 'O‘Connor', '+49 1521 6789123', 4, 4),
    ('Nora', 'Flaig', '+49 1579 2341567', 5, 5),
    ('Levi', 'Becker', '+49 1523 4567123', 6, 6),
    ('Zofia', 'Kowalska', '+49 1578 3456789', 7, 7),
    ('Elia', 'Schlag', '+49 1524 7891234', 8, 8),
    ('Chen', 'Wei', '+49 1575 9876123', 9, 9),
    ('Hans', 'Müller', '+49 1520 1234567', 10, 10)
;
```

Userdata - Entität

NULL bei Attributen RatingReceiver und RatingLender werden vorerst akzeptiert, da zu Beginn und auch während dem ersten Tausch keine Bewertung vorhanden ist

UserID	FirstName	LastName	MobileNumber	LoginID	AddressID	RatingReceiver	RatingLender
1	Alejandro	Maier	+49 1571 2345678	1	1	*NULL*	*NULL*
2	Fatima	Al-Hassan	+49 1522 98766543	2	2	*NULL*	*NULL*
3	Yuki	Tanaka	+49 1573 4567890	3	3	*NULL*	*NULL*
4	Liam	O'Connor	+49 1521 6789123	4	4	*NULL*	*NULL*
5	Nora	Flaig	+49 1579 2341567	5	5	*NULL*	*NULL*
6	Levi	Becker	+49 1523 4567123	6	6	*NULL*	*NULL*
7	Zofia	Kowalska	+49 1578 3456789	7	7	*NULL*	*NULL*
8	Elia	Schlag	+49 1524 7891234	8	8	*NULL*	*NULL*
9	Chen	Wei	+49 1575 9876123	9	9	*NULL*	*NULL*
10	Hans	Müller	+49 1520 1234567	10	10	*NULL*	*NULL*

Ergebnistabelle

Favouritelist - Entität

Tabelle erstellen

```
CREATE TABLE Favouritelist (
    BookID INT NOT NULL,
    UserID INT NOT NULL,
    PRIMARY KEY(BookID,UserID),
    FOREIGN KEY (BookID) REFERENCES Book(BookID)
    ON DELETE RESTRICT ON UPDATE CASCADE,
    FOREIGN KEY (UserID) REFERENCES Userdata(UserID)
    ON DELETE RESTRICT ON UPDATE CASCADE
);
```

Testdaten einfügen

```
INSERT INTO Favouritelist (BookID, UserID) VALUES
    (12,1),
    (15,1),
    (3,5),
    (11,5),
    (16,5),
    (6,7),
    (1,9),
    (5,3),
    (9,10),
    (5,2)
;
```

Ergebnistabelle

BookID	UserID
1	9
3	5
5	2
5	3
6	7
9	10
11	5
12	1
15	1
16	5

Genre - Entität

Tabelle erstellen

```
CREATE TABLE Genre (
    GenreID INT AUTO_INCREMENT,
    Genre VARCHAR(100),
    PRIMARY KEY (GenreID)
);
```

Testdaten einfügen

```
INSERT INTO Genre (Genre) VALUES
    ('Biografie'),
    ('Englische Bücher'),
    ('Graphic Novel'),
    ('Horror Roman'),
    ('Kinder- und Jugendbücher'),
    ('Komödie'),
    ('Kriminalroman'),
    ('Psychothriller'),
    ('Roman'),
    ('Thriller')
;
```

Ergebnistabelle

GenreID	Genre
1	Biografie
2	Englische Bücher
3	Graphic Novel
4	Horror Roman
5	Kinder- und Jugendbücher
6	Komödie
7	Kriminalroman
8	Psychothriller
9	Roman
10	Thriller

Publisher - Entität

Tabelle erstellen

```
CREATE TABLE Publisher (
    PublisherID INT AUTO_INCREMENT,
    Name VARCHAR(100),
    PRIMARY KEY (PublisherID)
);
```

Testdaten einfügen

```
INSERT INTO Publisher (Name) VALUES
    ('Anaconda'),
    ('Aufbau TB'),
    ('Diogenes'),
    ('Droemer Taschenbuch'),
    ('DuMont Buchverlag'),
    ('Heyne'),
    ('Insel'),
    ('Knaur'),
    ('Laurence King Publishing'),
    ('List'),
    ('Lübbe'),
    ('Penguin Books Ltd'),
    ('Rowohlt Taschenbuch'),
    ('Thienemann Esslinger Verlag')
;
```

Ergebnistabelle

PublisherID	Name
1	Anaconda
2	Aufbau TB
3	Diogenes
4	Droemer Taschenbuch
5	DuMont Buchverlag
6	Heyne
7	Insel
8	Knaur
9	Laurence King Publishing
10	List
11	Lübbe
12	Penguin Books Ltd
13	Rowohlt Taschenbuch
14	Thienemann Esslinger Verlag

Language - Entität

Tabelle erstellen

```
CREATE TABLE Language (
    LanguageID INT AUTO_INCREMENT,
    Code VARCHAR(2),
    Name VARCHAR(100),
    PRIMARY KEY (LanguageID)
);
```

Testdaten einfügen

```
INSERT INTO Language (Code, Name) VALUES
    ('de', 'deutsch'),
    ('en', 'englisch')
;
```

Bei dieser Entität wurde auf 10 Einträge verzichtet, da aufgrund der regionalen Nutzung der Buchtausch App mehr als diese zwei Sprachen unrealistisch sind. Bei Ausweitung der Nutzung sind auch mehr Sprachen möglich.

Ergebnistabelle

LanguageID	Code	Name
1	de	deutsch
2	en	englisch

YearOfPublication - Entität

Tabelle erstellen

```
CREATE TABLE YearOfPublication (
    YearID INT AUTO_INCREMENT,
    Year INT,
    PRIMARY KEY (YearID)
);
```

Testdaten einfügen

```
INSERT INTO YearOfPublication (Year) VALUES
    (1998),
    (2009),
    (2014),
    (2015),
    (2017),
    (2019),
    (2020),
    (2021),
    (2022),
    (2023)
;
```

Ergebnistabelle

YearID	Year
1	1998
2	2009
3	2014
4	2015
5	2017
6	2019
7	2020
8	2021
9	2022
10	2023

BookCover - Entität

Tabelle erstellen

```
CREATE TABLE BookCover (
    CoverID INT AUTO_INCREMENT,
    Name VARCHAR(100),
    PRIMARY KEY (CoverID)
);
```

Entität wurde zusätzlich hinzugefügt, damit für die Nutzer eine bessere Suchfunktion nach Büchern besteht

Testdaten einfügen

```
INSERT INTO BookCover (Name) VALUES
    ('Gebundenes Buch: Hardcover'),
    ('Taschenbuch')
;
```

Bei dieser Entität wurde auf 10 Einträge verzichtet, da die Möglichkeiten für Einbände begrenzt sind

Ergebnistabelle

CoverID	Name
1	Gebundenes Buch: Hardcover
2	Taschenbuch

NumberOfPages - Entität

Tabelle erstellen

```
CREATE TABLE NumberOfPages (
    PageID INT AUTO_INCREMENT,
    Pages INT,
    PRIMARY KEY (PageID)
);
```

Entität wurde zusätzlich hinzugefügt, damit für die Nutzer eine bessere Suchfunktion nach Büchern besteht

Testdaten einfügen

```
INSERT INTO NumberOfPages (Pages) VALUES
    (96),
    (128),
    (208),
    (256),
    (304),
    (345),
    (400),
    (416),
    (432),
    (464)
;
```

Ergebnistabelle

PageID	Pages
1	96
2	128
3	208
4	256
5	304
6	345
7	400
8	416
9	432
10	464

Weight - Entität

Tabelle erstellen

```
CREATE TABLE Weight (
    WeightID INT AUTO_INCREMENT,
    WeightInGrams INT,
    PRIMARY KEY (WeightID)
);
```

Entität wurde zusätzlich hinzugefügt, damit für die Nutzer eine bessere Suchfunktion nach Büchern besteht

Testdaten einfügen

```
INSERT INTO Weight (WeightInGrams) VALUES
    (106),
    (206),
    (247),
    (257),
    (263),
    (312),
    (317),
    (320),
    (355),
    (372),
    (397),
    (398),
    (400),
    (415),
    (462),
    (553)
;
```

Ergebnistabelle

WeightID	WeightInGrams
1	106
2	206
3	247
4	257
5	263
6	312
7	317
8	320
9	355
10	372
11	397
12	398
13	400
14	415
15	462
16	553

Dimensions - Entität

Tabelle erstellen

```
CREATE TABLE Dimensions (
    DimensionsID INT AUTO_INCREMENT,
    LengthWidthHeightInCm VARCHAR(50),
    PRIMARY KEY (DimensionsID)
);
```

Entität wurde zusätzlich hinzugefügt, damit für die Nutzer eine bessere Suchfunktion nach Büchern besteht

Testdaten einfügen

```
INSERT INTO Dimensions
(LengthWidthHeightInCm) VALUES
    ('12,70/19,6/3,10'),
    ('18,00/11,20/1,20'),
    ('18,60/12,20/3,30'),
    ('18,70/12,30/2,50'),
    ('18,80/12,20/2,00'),
    ('18,80/13,80/1,70'),
    ('19,20/13,10/3,80'),
    ('19,30/11,30/2,70'),
    ('19,00/12,40/3,20'),
    ('19,00/12,50/3,50'),
    ('19,00/12,60/3,10'),
    ('20,50/13,40/4,50'),
    ('20,50/13,50/3,20'),
    ('20,50/13,50/4,30'),
    ('20,80/14,40/1,20'),
    ('21,40/13,50/2,90')
;
```

Ergebnistabelle

DimensionsID	LengthWidthHeightInCm
1	12,70/19,6/3,10
2	18,00/11,20/1,20
3	18,60/12,20/3,30
4	18,70/12,30/2,50
5	18,80/12,20/2,00
6	18,80/13,80/1,70
7	19,20/13,10/3,80
8	19,30/11,30/2,70
9	19,00/12,40/3,20
10	19,00/12,50/3,50
11	19,00/12,60/3,10
12	20,50/13,40/4,50
13	20,50/13,50/3,20
14	20,50/13,50/4,30
15	20,80/14,40/1,20
16	21,40/13,50/2,90

Book - Entität

Tabelle erstellen

```
CREATE TABLE Book (
    BookID INT AUTO_INCREMENT,
    Title VARCHAR(100),
    ISBN VARCHAR(20),
    PublisherID INT NOT NULL,
    GenreID INT NOT NULL,
    LanguageID INT NOT NULL,
    PageID INT NOT NULL,
    CoverID INT NOT NULL,
    WeightID INT NOT NULL,
    DimensionsID INT NOT NULL,
    YearID INT NOT NULL,
    PRIMARY KEY (BookID),
    FOREIGN KEY (PublisherID) REFERENCES Publisher(PublisherID)
    ON DELETE RESTRICT ON UPDATE CASCADE,
    FOREIGN KEY (GenreID) REFERENCES Genre(GenreID)
    ON DELETE RESTRICT ON UPDATE CASCADE,
    FOREIGN KEY (LanguageID) REFERENCES Language(LanguageID)
    ON DELETE RESTRICT ON UPDATE CASCADE,
    FOREIGN KEY (PageID) REFERENCES NumberOfPages(PageID)
    ON DELETE RESTRICT ON UPDATE CASCADE,
    FOREIGN KEY (CoverID) REFERENCES BookCover(CoverID)
    ON DELETE RESTRICT ON UPDATE CASCADE,
    FOREIGN KEY (WeightID) REFERENCES Weight(WeightID)
    ON DELETE RESTRICT ON UPDATE CASCADE,
    FOREIGN KEY (DimensionsID) REFERENCES      Dimensions(DimensionsID)
    ON DELETE RESTRICT ON UPDATE CASCADE,
    FOREIGN KEY (YearID) REFERENCES YearOfPublication(YearID)
    ON DELETE RESTRICT ON UPDATE CASCADE
);
```

Testdaten einfügen

```
INSERT INTO Book (Title, ISBN, PublisherID, GenreID, LanguageID, PageID, CoverID,
WeightID, DimensionsID, YearID) VALUES
   ('Die Physiker', '978-3-257-23047-5',3,6,1,1,2,1,2,1),
   ('1984', '978-3-7306-0976-7',1,9,1,7,1,12,7,8),
   ('Momo', '978-3-522-20210-7',14,5,1,5,2,10,16,3),
   ('Passagier23', '978-3-426-51017-9',8,8,1,9,2,14,9,4),
   ('AchtNacht', '978-3-426-52108-3',8,10,1,8,2,9,10,5),
   ('Flugangst7A', '978-3-426-51019-3',8,8,1,8,2,7,11,6),
   ('Alte Sorten', '978-3-8321-6530-7',5,9,1,4,2,4,4,7),
   ('Lebenssekunden', '978-3-426-30837-0',4,9,1,8,2,6,3,9),
   ('Der Junge von Angel Falls', '978-3-7466-3572-9',2,9,1,6,2,11,13,10),
   ('Jesus liebt mich', '978-3-499-24811-5',13,9,1,5,2,3,8,2),
   ('The Book of Cat Poems', '978-1-78627-944-6',9,2,2,2,1,8,6,8),
   ('Becoming', '978-0-241-98297-6',12,1,2,10,2,13,1,8),
   ('Rebellische Frauen - Women in Battle','978-3-458-68311-7', 7, 3, 1, 2, 2, 5,15,10),
   ('HEX', '978-3-453-31906-6',6,4,1,9,2,15,12,5),
   ('Der Donnerstagsmordclub', '978-3-471-36014-9',10,7,1,10,2,16,14,8),
   ('Café der Unsichtbaren', '978-3-8321-6674-8',5,9,1,3,2,2,5,10)
;
```

- Aufgrund zusätzlicher Entitäten, wurden die Attribute PageID, CoverID, WeightID und DimensionsID ergänzt
- Attribute Genre, Publisher, Language wurde von Booksummary zu Book geschoben, da Eigenschaften für Bücher allgemeingültig sind
- Anpassung des Attributs von Year auf YearID, damit es zu anderen Fremdschlüssel Attributen einheitlich ist
- Änderung des Attributs ISBN von VARCHAR(13) zu VARCHAR(20), da ISBN-Nummer Bindestriche enthält

Book - Entität

BookID	Title	ISBN	PublisherID	GenreID	LanguageID	PageID	CoverID	WeightID	DimensionsID	YearID
1	Die Physiker	978-3-257-23047-5	3	6	1	1	2	1	2	1
2	1984	978-3-7306-0976-7	1	9	1	7	1	12	7	8
3	Momo	978-3-522-20210-7	14	5	1	5	2	10	16	3
4	Passagier23	978-3-426-51017-9	8	8	1	9	2	14	9	4
5	AchtNacht	978-3-426-52108-3	8	10	1	8	2	9	10	5
6	Flugangst7A	978-3-426-51019-3	8	8	1	8	2	7	11	6
7	Alte Sorten	978-3-8321-6530-7	5	9	1	4	2	4	4	7
8	Lebenssekunden	978-3-426-30837-0	4	9	1	8	2	6	3	9
9	Der Junge von Angel Falls	978-3-7466-3572-9	2	9	1	6	2	11	13	10
10	Jesus liebt mich	978-3-499-24811-5	13	9	1	5	2	3	8	2
11	The Book of Cat Poems	978-1-78627-944-6	9	2	2	2	1	8	6	8
12	Becoming	978-0-241-98297-6	12	1	2	10	2	13	1	8
13	Rebellische Frauen - Women in Battle	978-3-458-68311-7	7	3	1	2	2	5	15	10
14	HEX	978-3-453-31906-6	6	4	1	9	2	15	12	5
15	Der Donnerstagsmordclub	978-3-471-36014-9	10	7	1	10	2	16	14	8
16	Café der Unsichtbaren	978-3-8321-6674-8	5	9	1	3	2	2	5	10

Ergebnistabelle

Author - Entität

Tabelle erstellen

```
CREATE TABLE Author (
    AuthorID INT AUTO_INCREMENT,
    FirstName VARCHAR(100),
    LastName VARCHAR(100),
    PRIMARY KEY (AuthorID)
);
```

Testdaten einfügen

```
INSERT INTO Author (FirstName, LastName) VALUES
    ('Ana', 'Sampson'),
    ('David', 'Safier'),
    ('Ewald', 'Arenz'),
    ('Friedrich', 'Dürrenmatt'),
    ('George', 'Orwell'),
    ('Judith', 'Kuckart'),
    ('Katharina', 'Fuchs'),
    ('Kristin', 'Hannah'),
    ('Marta', 'Breen'),
    ('Michael', 'Ende'),
    ('Michelle', 'Obama'),
    ('Richard', 'Osman'),
    ('Sebastian', 'Fitzek'),
    ('Thomas Olde', 'Heuvelt')
;
```

Ergebnistabelle

AuthorID	FirstName	LastName
1	Ana	Sampson
2	David	Safier
3	Ewald	Arenz
4	Friedrich	Dürrenmatt
5	George	Orwell
6	Judith	Kuckart
7	Katharina	Fuchs
8	Kristin	Hannah
9	Marta	Breen
10	Michael	Ende
11	Michelle	Obama
12	Richard	Osman
13	Sebastian	Fitzek
14	Thomas Olde	Heuvelt

BookAuthor - Entität

Tabelle erstellen

```
CREATE TABLE BookAuthor (
    BookID INT,
    AuthorID INT,
    PRIMARY KEY (BookID,AuthorID),
    FOREIGN KEY(BookID) REFERENCES Book(BookID)
   ON DELETE RESTRICT ON UPDATE CASCADE,
    FOREIGN KEY(AuthorID) REFERENCES Author(AuthorID)
    ON DELETE RESTRICT ON UPDATE CASCADE
);
```

Testdaten einfügen

```
INSERT INTO BookAuthor (BookID, AuthorID) VALUES
    (1,4),
    (2,5),
    (3,10),
    (4,13),
    (5,13),
    (6,13),
    (7,3),
    (8,7),
    (9,8),
    (10,2),
    (11,1),
    (12,11),
    (13,9),
    (14,14),
    (15,12),
    (16,6)
;
```

Ergebnistabelle

BookID	AuthorID
1	4
2	5
3	10
4	13
5	13
6	13
7	3
8	7
9	8
10	2
11	1
12	11
13	9
14	14
15	12
16	6

Status - Entität

Tabelle erstellen

```
CREATE TABLE Status (
    StatusID INT AUTO_INCREMENT,
    Name VARCHAR(100),
    PRIMARY KEY (StatusID)
);
```

Testdaten einfügen

```
INSERT INTO Status (Name) VALUES
    ('verfügbar'),
    ('reserviert'),
    ('zurzeit ausgeliehen'),
    ('bald wieder verfügbar'),
    ('nicht verfügbar')
;
```

Bei dieser Entität wurde auf 10 Einträge verzichtet, da eine weitere Unterteilung des Status keinen Mehrwert für die User darstellt.
Im Gegenteil: eine zu große Auswahl könnte die User verwirren

Ergebnistabelle

StatusID	Name
1	verfügbar
2	reserviert
3	zurzeit ausgeliehen
4	bald wieder verfügbar
5	nicht verfügbar

Bookquality - Entität

Tabelle erstellen

```
CREATE TABLE Bookquality (
    QualityID INT AUTO_INCREMENT,
    Name VARCHAR(50),
    Description VARCHAR(100),
    PRIMARY KEY (QualityID)
);
```

Testdaten einfügen

```
INSERT INTO Bookquality (Name, Description) VALUES
    ('neu', 'Noch nicht gelesen und evtl. mit Originalverpackung'),
    ('wie neu', 'Keine Gebrauchsspuren, Knicke, Verfärbungen,
    Einträge, wie ungelesen'),
    ('leichte Gebrauchsspuren', 'Leichte Knicke am Einband/bei
    Seiten, wenige Einträge, vergilbte Seiten, kleine Flecken'),
    ('deutliche Gebrauchsspuren', 'Knicke oder Flecken am
    Einband/bei Seiten, leichte Risse, mehrere Einträge'),
    ('stark abgenutzt', 'Viele Knicke, Flecken oder Risse, einzelne
    lose Seiten (aber vollständig), Wasserschaden')
;
```

Bei dieser Entität wurde auf 10 Einträge verzichtet, da eine weitere Unterteilung der Qualität zu detailliert ist und die Nutzer sich bei großer Auswahlmöglichkeit schwer entscheiden können.

Ergebnistabelle

QualityID	Name	Description
1	neu	Noch nicht gelesen und evtl. mit Originalverpackung
2	wie neu	Keine Gebrauchsspuren, Knicke, Verfärbungen, Einträge, wie ungelesen
3	leichte Gebrauchsspuren	Leichte Knicke am Einband/bei Seiten, wenige Einträge, vergilbte Seiten, kleine Flecken
4	deutliche Gebrauchsspuren	Knicke oder Flecken am Einband/bei Seiten, leichte Risse, mehrere Einträge
5	stark abgenutzt	Viele Knicke, Flecken oder Risse, einzelne lose Seiten (aber vollständig), Wasserschaden

Booksummary - Entität

Tabelle erstellen

```
CREATE TABLE Booksummary (
    SummaryID INT AUTO_INCREMENT,
    BookID INT NOT NULL,
    QualityID INT NOT NULL,
    UserID INT NOT NULL,
    StatusID INT NOT NULL,
    AvailableFrom DATE,
    AvailableUntil DATE,
    Mailing BOOLEAN,
    BookRating DECIMAL (3,2),
    PRIMARY KEY (SummaryID),
    FOREIGN KEY (BookID) REFERENCES  Book(BookID)
    ON DELETE RESTRICT ON UPDATE CASCADE,
    FOREIGN KEY (QualityID) REFERENCES Bookquality(QualityID)
    ON DELETE RESTRICT ON UPDATE CASCADE,
    FOREIGN KEY (UserID) REFERENCES  Userdata(UserID)
    ON DELETE RESTRICT ON UPDATE CASCADE,
    FOREIGN KEY (StatusID) REFERENCES Status(StatusID)
    ON DELETE RESTRICT ON UPDATE CASCADE
);
```

Testdaten einfügen

```
INSERT INTO Booksummary (BookID,QualityID, UserID, StatusID,
AvailableFrom, AvailableUntil, Mailing) VALUES
    (1,2,1,4,'2024-12-26','',TRUE),
    (2,2,4,1,'2024-12-23','2025-01-07',TRUE),
    (3,2,3,1,'2024-12-23','2025-01-02',FALSE),
    (4,3,2,3,'2025-01-08','2025-01-18',FALSE),
    (5,3,4,1,'2024-12-23','2025-01-03',TRUE),
    (6,3,5,4,'2024-12-26','2024-12-31',TRUE),
    (7,2,6,1,'2024-12-23','2024-12-29',FALSE),
    (8,4,6,1,'2024-12-23','2025-01-15',FALSE),
    (9,2,7,1,'2024-12-23','2025-01-11',FALSE),
    (10,3,5,2,'2025-01-03','',TRUE),
    (11,2,8,2,'2025-01-04','2025-01-31',TRUE),
    (12,2,5,3,'2025-01-16','2025-02-01',TRUE),
    (13,2,8,1,'2024-12-23','2025-01-28',TRUE),
    (14,4,10,5,'','',FALSE),
    (15,3,7,1,'2024-12-23','2025-01-03',FALSE),
    (16,4,9,3,'2025-01-05','2025-02-04',TRUE)
;
```

Zusätzlich hinzugefügtes Attribut BookRating, damit Buchbewertung für andere Nutzer einen Mehrwert bietet

Booksummary - Entität

⚠ Warning: #1265 Daten abgeschnitten für Feld 'AvailableUntil' in Zeile 1

⚠ Warning: #1265 Daten abgeschnitten für Feld 'AvailableUntil' in Zeile 10

⚠ Warning: #1265 Daten abgeschnitten für Feld 'AvailableFrom' in Zeile 14

⚠ Warning: #1265 Daten abgeschnitten für Feld 'AvailableUntil' in Zeile 14

Meldung wird absichtlich akzeptiert:
- Bei ID 1 und 10 ist kein weiterer Ausleihzeitraum hinterlegt, wodurch bei „AvailableUntil“ keine Daten gepflegt werden können
- Bei ID 14 sind aufgrund des Status 5 „nicht verfügbar“ ebenfalls keine Daten bei der Verfügbarkeit gepflegt

SummaryID	BookID	QualityID	UserID	StatusID	AvailableFrom	AvailableUntil	Mailing	BookRating
1	1	2	1	4	2024-12-26	0000-00-00	1	NULL
2	2	2	4	1	2024-12-23	2025-01-07	1	NULL
3	3	2	3	1	2024-12-23	2025-01-02	0	NULL
4	4	3	2	3	2025-01-08	2025-01-18	0	NULL
5	5	3	4	1	2024-12-23	2025-01-03	1	NULL
6	6	3	5	4	2024-12-26	2024-12-31	1	NULL
7	7	2	6	1	2024-12-23	2024-12-29	0	NULL
8	8	4	6	1	2024-12-23	2025-01-15	0	NULL
9	9	2	7	1	2024-12-23	2025-01-11	0	NULL
10	10	3	5	2	2025-01-03	0000-00-00	1	NULL
11	11	2	8	2	2025-01-04	2025-01-31	1	NULL
12	12	2	5	3	2025-01-16	2025-02-01	1	NULL
13	13	2	8	1	2024-12-23	2025-01-28	1	NULL
14	14	4	10	5	0000-00-00	0000-00-00	0	NULL
15	15	3	7	1	2024-12-23	2025-01-03	0	NULL
16	16	4	9	3	2025-01-05	2025-02-04	1	NULL

Ergebnistabelle

Borrowingperiod - Entität

Tabelle erstellen

```
CREATE TABLE Borrowingperiod (
    BorrowID INT AUTO_INCREMENT,
    SummaryID INT NOT NULL,
    UserID INT NOT NULL,
    BorrowingStart TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
    BorrowingEnd DATETIME,
    PRIMARY KEY (BorrowID),
    FOREIGN KEY (SummaryID) REFERENCES Booksummary(SummaryID)
    ON DELETE RESTRICT ON UPDATE CASCADE,
    FOREIGN KEY (UserID) REFERENCES Userdata(UserID)
    ON DELETE RESTRICT ON UPDATE CASCADE
);
```

Beim Attribut BorrowingEnd wurde der Datentyp auf DATETIME geändert, da in phpMyAdmin eine zweifache Anwendung von TIMESTAMP nicht möglich ist

Testdaten einfügen

```
INSERT INTO Borrowingperiod (SummaryID, UserID,
BorrowingStart, BorrowingEnd) VALUES
    (5,8,'2024-11-01 10:33:28','2024-11-28 12:03:14'),
    (2,1,'2024-11-02 15:18:36','2024-12-18 16:02:45'),
    (3,7,'2024-11-04 17:55:41','2024-11-28 11:27:31'),
    (6,6,'2024-11-11 11:52:31','2024-11-25 14:41:25'),
    (8,3,'2024-11-13 17:31:46','2024-12-02 14:12:53'),
    (7,9,'2024-11-15 19:07:15','2024-11-30 17:22:47'),
    (11,7,'2024-11-22 11:17:45','2024-12-12 18:39:02'),
    (6,8,'2024-11-29 17:50:33','2024-12-25 00:00:00'),
    (5,5,'2024-11-30 13:51:46','2024-12-19 16:37:09'),
    (8,9,'2024-12-04 09:24:38','2024-12-21 10:13:41'),
    (15,6,'2024-12-06 15:17:52','2024-12-19 17:49:21'),
    (16,3,'2024-12-07 14:16:55','2025-01-04 00:00:00'),
    (1,4,'2024-12-14 16:55:12','2024-12-25 00:00:00'),
    (12,7,'2024-12-16 18:58:02','2025-01-15 00:00:00'),
    (4,10,'2024-12-20 09:15:19','2025-01-07 00:00:00')
;
```

Borrowingperiod - Entität

BorrowID	SummaryID	UserID	BorrowingStart	BorrowingEnd
1	5	8	2024-11-01 10:33:28	2024-11-28 12:03:14
2	2	1	2024-11-02 15:18:36	2024-12-18 16:02:45
3	3	7	2024-11-04 17:55:41	2024-11-28 11:27:31
4	6	6	2024-11-11 11:52:31	2024-11-25 14:41:25
5	8	3	2024-11-13 17:31:46	2024-12-02 14:12:53
6	7	9	2024-11-15 19:07:15	2024-11-30 17:22:47
7	11	7	2024-11-22 11:17:45	2024-12-12 18:39:02
8	6	8	2024-11-29 17:50:33	2024-12-25 00:00:00
9	5	5	2024-11-30 13:51:46	2024-12-19 16:37:09
10	8	9	2024-12-04 09:24:38	2024-12-21 10:13:41
11	15	6	2024-12-06 15:17:52	2024-12-19 17:49:21
12	16	3	2024-12-07 14:16:55	2025-01-04 00:00:00
13	1	4	2024-12-14 16:55:12	2024-12-25 00:00:00
14	12	7	2024-12-16 18:58:02	2025-01-15 00:00:00
15	4	10	2024-12-20 09:15:19	2025-01-07 00:00:00

Ergebnistabelle

Bookrating - Entität

Tabelle erstellen

```
CREATE TABLE Bookrating (
    RatingID INT AUTO_INCREMENT,
    BorrowID INT NOT NULL,
    Quality INT NOT NULL,
    Content INT NOT NULL,
    Recommendation INT NOT NULL,
    AverageRate DECIMAL(3,2) GENERATED ALWAYS AS ((Quality + Content +
    Recommendation) / 3.0) STORED,
    FreeText VARCHAR(100),
    PRIMARY KEY (RatingID),
    FOREIGN KEY (BorrowID) REFERENCES Borrowingperiod(BorrowID)
    ON DELETE RESTRICT ON UPDATE CASCADE,
    CONSTRAINT chk_quality CHECK(Quality BETWEEN 1 AND 5),
    CONSTRAINT chk_content CHECK(Content BETWEEN 1 AND 5),
    CONSTRAINT chk_recommendation CHECK (Recommendation BETWEEN 1 AND 5)
);
```

Bei Attributen Quality, Content und Recommendation wird zusätzlich NOT NULL hinzugefügt, damit bei Teilung durch 3 der durchschnittlichen Bewertung, diese repräsentativ bleibt

Bei den Attributen Quality, Content und Recommendation wird die Werteauswahl auf 1-5 begrenzt, da es eine Sternebewertung abbildet

Bei dem Attribut AverageRate wird mithilfe GENERATED ALWAYS der Durchschnitt der Bewertungen berechnet und mit STORED gespeichert

Struktur der Tabelle

RatingID	BorrowID	Quality	Content	Recommendation	AverageRate	FreeText

Bookrating - Entität

Trigger Bookrating - INSERT

```
DELIMITER $$
CREATE TRIGGER Update_Book_AVGRatingBook_OnInsert
AFTER INSERT ON Bookrating
FOR EACH ROW
BEGIN
DECLARE summary_id INT;

SELECT BS.SummaryID INTO summary_id FROM
Booksummary BS INNER JOIN Borrowingperiod B ON
BS.SummaryID = B.SummaryID
WHERE B.BorrowID = NEW.BorrowID;

UPDATE Booksummary BS SET Bookrating = (SELECT
AVG(BR.AverageRate) FROM
Bookrating BR INNER JOIN Borrowingperiod B ON
BR.BorrowID = B.BorrowID INNER JOIN Booksummary BS
ON B.SummaryID = BS.SummaryID
WHERE BS.SummaryID = summary_id )
WHERE BS.SummaryID = summary_id;
END$$
```

Trigger Bookrating - UPDATE

```
DELIMITER $$
CREATE TRIGGER Update_Book_AVGRatingBook_OnUpdate
AFTER UPDATE ON Bookrating
FOR EACH ROW
BEGIN
DECLARE summary_id INT;

SELECT BS.SummaryID INTO summary_id FROM
Booksummary BS INNER JOIN Borrowingperiod B ON
BS.SummaryID = B.SummaryID
WHERE B.BorrowID = NEW.BorrowID;

UPDATE Booksummary BS SET Bookrating = ( SELECT
AVG(BR.AverageRate) FROM
Bookrating BR INNER JOIN Borrowingperiod B ON
BR.BorrowID = B.BorrowID INNER JOIN Booksummary BS
ON B.SummaryID = BS.SummaryID
WHERE BS.SummaryID = summary_id )
WHERE BS.SummaryID = summary_id;
END$$
```

Trigger wird benötigt, damit beim Einfügen von neuen Daten, Ändern und Löschen in der Entität Bookrating der Durchschnitt aller Bewertungen zu einem Buch aktualisiert und in Attribut BookRating in der Entität Booksummary eingetragen wird

DELIMITER ändert das Trennzeichen Semikolon ; ,sodass dieses nicht als Befehlsende angesehen wird. Somit wird der komplette Ausdruck als ein Befehl erkannt

Bookrating - Entität

Trigger Bookrating - DELETE

```
DELIMITER $$
CREATE TRIGGER Update_Book_AVGRatingBook_OnDelete
AFTER DELETE ON Bookrating
FOR EACH ROW
BEGIN
DECLARE summary_id INT;

SELECT BS.SummaryID INTO summary_id FROM
Booksummary BS INNER JOIN Borrowingperiod B ON
BS.SummaryID = B.SummaryID
WHERE B.BorrowID = OLD.BorrowID;

UPDATE Booksummary BS SET Bookrating = ( SELECT
AVG(BR.AverageRate) FROM
Bookrating BR INNER JOIN Borrowingperiod B ON
BR.BorrowID = B.BorrowID INNER JOIN Booksummary BS
ON B.SummaryID = BS.SummaryID
WHERE BS.SummaryID = summary_id )
WHERE BS.SummaryID = summary_id;
END$$
```

Testdaten einfügen

```
INSERT INTO bookrating (BorrowID, Quality, Content,
Recommendation, FreeText) VALUES
   (1, 3, 4, 5, 'Gutes Buch mit kleineren Mängeln'),
   (2, 4, 4, 5, 'Sehr lesenswert'),
   (3, 4, 3, 3, 'Durchschnittliche Lektüre'),
   (4, 3, 5, 4, 'Interessante Handlung, lässt einen nachdenklich
   zurück'),
   (5, 1, 5, 3, 'Inhaltlich schönes Buch, auch wenn es sehr
   mitgenommen aussieht'),
   (6, 4, 4, 5, 'Sehr gut, würde ich weiterempfehlen'),
   (7, 5, 4, 4, 'Tolles Buch!'),
   (9, 2, 5, 5, 'Ungewöhnliche Geschichte und nichts für schwache
   Nerven, aber spannend'),
   (10, 2, 4, 5, 'Erzählt die Vergangenheit mal anders ... gerne wieder'),
   (11, 3, 2, 2, 'Das Genre war leider nichts für mich, aber vielleicht für
   jemand anders :)')
;
```

Bookrating - Entität

RatingID	BorrowID	Quality	Content	Recommendation	AverageRate	FreeText
1	1	3	4	5	4.00	Gutes Buch mit kleineren Mängeln
2	2	4	4	5	4.33	Sehr lesenswert
3	3	4	3	3	3.33	Durchschnittliche Lektüre
4	4	3	5	4	4.00	Interessante Handlung, lässt einen nachdenklich zurück
5	5	1	5	3	3.00	Inhaltlich schönes Buch, auch wenn es sehr mitgenommen aussieht
6	6	4	4	5	4.33	Sehr gut, würde ich weiterempfehlen
7	7	5	4	4	4.33	Tolles Buch!
8	9	2	5	5	4.00	Ungewöhnliche Geschichte und nichts für schwache Nerven, aber spannend
9	10	2	4	5	3.67	Erzählt die Vergangenheit mal anders ... gerne wieder
10	11	3	2	2	2.33	Das Genre war leider nichts für mich, aber vielleicht für jemand anders :)

Ergebnistabelle

Meldung wird absichtlich akzeptiert: Datentyp von BookRating ist auf DECIMAL(3,2) begrenzt. Eine weitere Zulassung von Nachkommastellen, bietet für die Nutzer keinen Mehrwert

- Note: #1265 Daten abgeschnitten für Feld 'AverageRate' in Zeile 2
- Note: #1265 Daten abgeschnitten für Feld 'AverageRate' in Zeile 2
- Note: #1265 Daten abgeschnitten für Feld 'AverageRate' in Zeile 3
- Note: #1265 Daten abgeschnitten für Feld 'AverageRate' in Zeile 3
- Note: #1265 Daten abgeschnitten für Feld 'AverageRate' in Zeile 6
- Note: #1265 Daten abgeschnitten für Feld 'AverageRate' in Zeile 6
- Note: #1265 Daten abgeschnitten für Feld 'AverageRate' in Zeile 7
- Note: #1265 Daten abgeschnitten für Feld 'AverageRate' in Zeile 7
- Note: #1265 Daten abgeschnitten für Feld 'AverageRate' in Zeile 9
- Note: #1265 Daten abgeschnitten für Feld 'AverageRate' in Zeile 9
- Note: #1265 Daten abgeschnitten für Feld 'AverageRate' in Zeile 10
- Note: #1265 Daten abgeschnitten für Feld 'AverageRate' in Zeile 10

SummaryID	BookID	AuthorID	PublisherID	GenreID	LanguageID	QualityID	UserID	StatusID	AvailableFrom	AvailableUntil	Mailing	BookRating
1	1	4	3	6	1	2	1	4	2024-12-26	0000-00-00	1	*NULL*
2	2	5	1	9	1	2	4	1	2024-12-23	2025-01-07	1	4.33
3	3	10	14	5	1	2	3	1	2024-12-23	2025-01-02	0	3.33
4	4	13	8	8	1	3	2	3	2025-01-08	2025-01-18	0	*NULL*
5	5	13	8	10	1	3	4	1	2024-12-23	2025-01-03	1	4.00
6	6	13	8	8	1	3	5	4	2024-12-26	2024-12-31	1	4.00
7	7	3	5	9	1	2	6	1	2024-12-23	2024-12-29	0	4.33
8	8	7	4	9	1	4	6	1	2024-12-23	2025-01-15	0	3.34
9	9	8	2	9	1	2	7	1	2024-12-23	2025-01-11	0	*NULL*
10	10	2	13	9	1	3	5	2	2025-01-03	0000-00-00	1	*NULL*
11	11	1	9	2	2	2	8	2	2025-01-04	2025-01-31	1	4.33
12	12	11	12	1	2	2	5	3	2025-01-16	2025-02-01	1	*NULL*
13	13	9	7	3	1	2	8	1	2024-12-23	2025-01-28	1	*NULL*
14	14	14	6	4	1	4	10	5	0000-00-00	0000-00-00	0	*NULL*
15	15	12	10	7	1	3	7	1	2024-12-23	2025-01-03	0	2.33
16	16	6	5	9	1	4	9	3	2025-01-05	2025-02-04	1	*NULL*

Aktualisierte Entität Booksummary

UserratingLender - Entität

Tabelle erstellen

```
CREATE TABLE UserratingLender (
    RatingLenID INT AUTO_INCREMENT,
    BorrowID INT NOT NULL,
    Friendliness INT NOT NULL,
    Reliability INT NOT NULL,
    AccuracyOfTheBook INT NOT NULL,
    Recommendation INT NOT NULL,
    AverageRate DECIMAL(3,2) GENERATED ALWAYS AS ((Friendliness + Reliability +
    AccuracyOfTheBook + Recommendation) / 4.0) STORED,
    FreeText VARCHAR(100),
    PRIMARY KEY (RatingLenID),
    FOREIGN KEY (BorrowID) REFERENCES Borrowingperiod(BorrowID)
    ON DELETE RESTRICT ON UPDATE CASCADE,
    CONSTRAINT chklen_friendliness CHECK(Friendliness BETWEEN 1 AND 5),
    CONSTRAINT chklen_reliability CHECK(Reliability BETWEEN 1 AND 5),
    CONSTRAINT chklen_accuracyOfTheBook CHECK(AccuracyOfTheBook BETWEEN 1 AND 5),
    CONSTRAINT chklen_recommendation CHECK(Recommendation BETWEEN 1 AND 5)
);
```

Ursprüngliche Bezeichnung der Entität UserratingReceiver wurde auf UserratingLender umbenannt, da die Entität den Namen der User tragen soll, welche bewertet werden.

Bei Attributen Friendliness, Reliability, AccuracyOfTheBook und Recommendation wird zusätzlich NOT NULL hinzugefügt, damit bei Teilung durch 4 der durchschnittlichen Bewertung, diese repräsentativ bleibt

Bei dem Attribut AverageRate wird mithilfe GENERATED ALWAYS der Durchschnitt der Bewertungen berechnet und mit STORED gespeichert

Bei den Attributen Friendliness, Reliability, AccuracyOfTheBook und Recommendation wird die Werteauswahl auf 1-5 begrenzt, da es eine Sternebewertung abbildet

Struktur der Tabelle

RatingLenID	BorrowID	Friendliness	Reliability	AccuracyOfTheBook	Recommendation	AverageRate	FreeText

UserratingLender - Entität

Trigger UserratingL - INSERT

```
DELIMITER $$
CREATE TRIGGER  Update_User_AVGRatingLender_OnInsert
AFTER INSERT ON UserratingLender
FOR EACH ROW
BEGIN
DECLARE user_id INT;

SELECT BS.UserID INTO user_id FROM
Booksummary BS INNER JOIN Borrowingperiod B ON
BS.SummaryID = B.SummaryID
WHERE B.BorrowID = NEW.BorrowID;

UPDATE Userdata U SET RatingLender = (SELECT
AVG(L.AverageRate)
FROM UserratingLender L INNER JOIN Borrowingperiod B ON
L.BorrowID = B.BorrowID INNER JOIN Booksummary BS ON
B.SummaryID = BS.SummaryID
WHERE BS.UserID = user_id )
WHERE U.UserID = user_id;
END$$
```

Trigger UserratingL - UPDATE

```
DELIMITER $$
CREATE TRIGGER  Update_User_AVGRatingLender_OnUpdate
AFTER UPDATE ON UserratingLender
FOR EACH ROW
BEGIN
DECLARE user_id INT;

SELECT BS.UserID INTO user_id FROM
Booksummary BS INNER JOIN Borrowingperiod B ON
BS.SummaryID = B.SummaryID
WHERE B.BorrowID = NEW.BorrowID;

UPDATE Userdata U SET RatingLender = (SELECT
AVG(L.AverageRate)
FROM UserratingLender L INNER JOIN Borrowingperiod B ON
L.BorrowID = B.BorrowID INNER JOIN Booksummary BS ON
B.SummaryID = BS.SummaryID
WHERE BS.UserID = user_id )
WHERE U.UserID = user_id;
END$$
```

Trigger wird benötigt, damit beim Einfügen von neuen Daten, Ändern und Löschen in der Entität UserratingLender der Durchschnitt aller Bewertungen zu einem Buch aktualisiert und in Attribut RatingLender in der Entität Userdata eingetragen wird

DELIMITER ändert das Trennzeichen Semikolon ; ,sodass dieses nicht als Befehlsende angesehen wird. Somit wird der komplette Ausdruck als ein Befehl erkannt

UserratingLender - Entität

Trigger UserratingL - DELETE

```
DELIMITER $$
CREATE TRIGGER  Update_User_AVGRatingLender_OnDelete
AFTER DELETE ON UserratingLender
FOR EACH ROW
BEGIN
DECLARE user_id INT;

SELECT BS.UserID INTO user_id FROM
Booksummary BS INNER JOIN Borrowingperiod B ON
BS.SummaryID = B.SummaryID
WHERE B.BorrowID = OLD.BorrowID;

UPDATE Userdata U SET RatingLender = (SELECT
AVG(L.AverageRate)
FROM UserratingLender L INNER JOIN Borrowingperiod B ON
L.BorrowID = B.BorrowID INNER JOIN Booksummary BS ON
B.SummaryID = BS.SummaryID
WHERE BS.UserID = user_id )
WHERE U.UserID = user_id;
END$$
```

Testdaten einfügen

```
INSERT INTO UserratingLender(BorrowID, Friendliness, Reliability,
AccuracyOfTheBook, Recommendation, FreeText) VALUES
    (1,5,4,4,5,'Sehr freundlicher Nutzer, alles lief problemlos.'),
    (2,3,4,4,3,'Zuverlässig, aber leider fand ich die Kommunikation
    etwas holprig.'),
    (3,3,3,5,4,'Das Buch entsprach der Beschreibung, Kommunikation
    war okay.'),
    (4,3,2,4,3,'Beim Abholen musste ich leider doch länger warten, bis
    das Buch zur Hand war.'),
    (5,4,2,4,4,'Er war freundlich, doch für ein Abholtermin musste ich
    etwas warten.'),
    (6,4,4,5,5,'Perfekter Austausch, sehr zuverlässig und freundlich!'),
    (7,2,3,4,3,'Leider konnte ich das Buch durch die Abwicklung mit der
    Userin nicht so genießen :('),
    (9,5,3,3,4,'Sehr freundlich, doch die Buchqualität könnte mal
    angepasst werden'),
    (10,5,5,5,5,'Alles lief hervorragend, ich werde zukünftig mehr bei
    ihm ausleihen'),
    (11,4,4,5,5,'Gerne wieder :)')
;
```

UserratingLender - Entität

RatingLenID	BorrowID	Friendliness	Reliability	AccuracyOfTheBook	Recommendation	AverageRate	FreeText
1	1	5	4	4	5	4.50	Sehr freundlicher Nutzer, alles lief problemlos.
2	2	3	4	4	3	3.50	Zuverlässig, aber leider fand ich die Kommunikation etwas holprig.
3	3	3	3	5	4	3.75	Das Buch entsprach der Beschreibung, Kommunikation war okay.
4	4	3	2	4	3	3.00	Beim Abholen musste ich leider doch länger warten, bis das Buch zur Hand war.
5	5	4	2	4	4	3.50	Er war freundlich, doch für ein Abholtermin musste ich etwas warten.
6	6	4	4	5	5	4.50	Perfekter Austausch, sehr zuverlässig und freundlich!
7	7	2	3	4	3	3.00	Leider konnte ich das Buch durch die Abwicklung mit der Userin nicht so genießen :(
8	9	5	3	3	4	3.75	Sehr freundlich, doch die Buchqualität könnte mal angepasst werden
9	10	5	5	5	5	5.00	Alles lief hervorragend, ich werde zukünftig mehr bei ihm ausleihen
10	11	4	4	5	5	4.50	Gerne wieder :)

Ergebnistabelle

UserID	FirstName	LastName	MobileNumber	LoginID	AddressID	RatingReceiver	RatingLender
1	Alejandro	Maier	+49 1571 2345678	1	1	*NULL*	*NULL*
2	Fatima	Al-Hassan	+49 1522 98766543	2	2	*NULL*	*NULL*
3	Yuki	Tanaka	+49 1573 4567890	3	3	*NULL*	3.75
4	Liam	O'Connor	+49 1521 6789123	4	4	*NULL*	3.92
5	Nora	Flaig	+49 1579 2341567	5	5	*NULL*	3.00
6	Levi	Becker	+49 1523 4567123	6	6	*NULL*	4.33
7	Zofia	Kowalska	+49 1578 3456789	7	7	*NULL*	4.50
8	Elia	Schlag	+49 1524 7891234	8	8	*NULL*	3.00
9	Chen	Wei	+49 1575 9876123	9	9	*NULL*	*NULL*
10	Hans	Müller	+49 1520 1234567	10	10	*NULL*	*NULL*

Aktualisierte Entität Userdata

UserratingReceiver - Entität

Tabelle erstellen

```
CREATE TABLE UserratingReceiver (
    RatingRecID INT AUTO_INCREMENT,
    BorrowID INT NOT NULL,
    Friendliness INT NOT NULL,
    Reliability INT NOT NULL,
    BookQualityAfterReturn INT NOT NULL,
    Recommendation INT NOT NULL,
    AverageRate DECIMAL(3,2) GENERATED ALWAYS AS ((Friendliness + Reliability +
    BookQualityAfterReturn + Recommendation) / 4.0) STORED,
    FreeText VARCHAR(100),
    PRIMARY KEY (RatingRecID),
    FOREIGN KEY (BorrowID) REFERENCES Borrowingperiod(BorrowID)
    ON DELETE RESTRICT ON UPDATE CASCADE,
    CONSTRAINT chkrec_friendliness CHECK(Friendliness BETWEEN 1 AND 5),
    CONSTRAINT chkrec_reliability CHECK(Reliability BETWEEN 1 AND 5),
    CONSTRAINT chkrec_bookqualityafterreturn CHECK(BookQualityAfterReturn BETWEEN 1 AND 5),
    CONSTRAINT chkrec_recommendation CHECK(Recommendation BETWEEN 1 AND 5)
);
```

Ursprüngliche Bezeichnung der Entität UserratingLender wurde auf UserratingReceiver umbenannt, da die Entität den Namen der User tragen soll, welche bewertet werden.

Bei Attributen Friendliness, Reliability, BookQualityAfterReturn und Recommendation wird zusätzlich NOT NULL hinzugefügt, damit bei Teilung durch 4 der durchschnittlichen Bewertung, diese repräsentativ bleibt

Bei dem Attribut AverageRate wird mithilfe GENERATED ALWAYS der Durchschnitt der Bewertungen berechnet und mit STORED gespeichert

Bei den Attributen Friendliness, Reliability, BookQualityAfterReturn und Recommendation wird die Werteauswahl auf 1-5 begrenzt, da es eine Sternebewertung abbildet

Struktur der Tabelle

UserratingReceiver - Entität

Trigger UserratingR - INSERT

```
DELIMITER $$
CREATE TRIGGER Update_User_AVGRatingReceiver_OnInsert
AFTER INSERT ON UserratingReceiver
FOR EACH ROW
BEGIN
DECLARE user_id INT;

SELECT B.UserID INTO user_id FROM
Borrowingperiod B
WHERE B.BorrowID = NEW.BorrowID;

UPDATE Userdata U SET RatingReceiver= (SELECT
AVG(R.AverageRate)
FROM UserratingReceiver R INNER JOIN Borrowingperiod B
ON R.BorrowID = B.BorrowID
WHERE B.UserID = user_id )
WHERE U.UserID = user_id;
END$$
```

Trigger UserratingR - UPDATE

```
DELIMITER $$
CREATE TRIGGER Update_User_AVGRatingReceiver_OnUpdate
AFTER UPDATE ON UserratingReceiver
FOR EACH ROW
BEGIN
DECLARE user_id INT;

SELECT B.UserID INTO user_id FROM
Borrowingperiod B
WHERE B.BorrowID = NEW.BorrowID;

UPDATE Userdata U SET RatingReceiver= (SELECT
AVG(R.AverageRate)
FROM UserratingReceiver R INNER JOIN Borrowingperiod B ON
R.BorrowID = B.BorrowID
WHERE B.UserID = user_id )
WHERE U.UserID = user_id;
END$$
```

Trigger wird benötigt, damit beim Einfügen von neuen Daten, Ändern und Löschen in der Entität UserratingReceiver der Durchschnitt aller Bewertungen zu einem Buch aktualisiert und in Attribut RatingReceiver in der Entität Userdata eingetragen wird

DELIMITER ändert das Trennzeichen Semikolon ; ,sodass dieses nicht als Befehlsende angesehen wird. Somit wird der komplette Ausdruck als ein Befehl erkannt

UserratingReceiver - Entität

Trigger UserratingR - DELETE

```
DELIMITER $$
CREATE TRIGGER Update_User_AVGRatingReceiver_OnDelete
AFTER DELETE ON UserratingReceiver
FOR EACH ROW
BEGIN
DECLARE user_id INT;

SELECT B.UserID INTO user_id FROM
Borrowingperiod B
WHERE B.BorrowID = OLD.BorrowID;

UPDATE Userdata U SET RatingReceiver= (SELECT
AVG(R.AverageRate)
FROM UserratingReceiver R INNER JOIN Borrowingperiod B ON
R.BorrowID = B.BorrowID
WHERE B.UserID = user_id )
WHERE U.UserID = user_id;
END$$
```

Testdaten einfügen

```
INSERT INTO UserratingReceiver (BorrowID, Friendliness, Reliability,
BookQualityAfterReturn, Recommendation, FreeText) VALUES
   (1,5,4,4,5, 'Problemlose und unkomplizierte Absprache, gerne
   wieder :).'),
   (2,4,3,5,4, 'Die Rückgabe wurde kurzfristig verlängert, eine frühere
   Info wäre besser gewesen.'),
   (3,3,3,5,4, 'Buch kam leicht verspätet zurück, die Kommunikation
   war okay.'),
   (4,3,2,5,3, 'Er hat das Buch viel zu spät zurückgegeben und war
   schlecht erreichbar.'),
   (5,4,5,5,5, 'Trotz Terminschwierigkeiten meinerseits eine geduldige
   und freundliche Person.'),
   (6,5,5,5,5,'Sehr freundlicher Nutzer, Buch kam pünktlich und in
   perfektem Zustand zurück.'),
   (7,2,2,2,1, 'Sie war unfreundlich und hat das Buch mit
   beschädigtem Einband zurückgegeben.'),
   (9,5,4,3,4, 'Sie war sehr freundlich und hat mich aktiv auf
   Qualitätsmängel zum Buch hingewiesen.'),
   (10,4,5,5,5, 'Super zuverlässig – Buch wurde wie vereinbart
   zurückgegeben, absolut problemlos.'),
   (11,5,5,3,3, 'Zuverlässiger Nutzer, aber die Seiten waren
   stellenweise geknickt.')
;
```

UserratingReceiver - Entität

RatingRecID	BorrowID	Friendliness	Reliability	BookQualityAfterReturn	Recommendation	AverageRate	FreeText
1	1	5	4	4	5	4.50	Problemlose und unkomplizierte Absprache, gerne wieder :).
2	2	4	3	5	4	4.00	Die Rückgabe wurde kurzfristig verlängert, eine frühere Info wäre besser gewesen.
3	3	3	3	5	4	3.75	Buch kam leicht verspätet zurück, die Kommunikation war okay.
4	4	3	2	5	3	3.25	Er hat das Buch viel zu spät zurückgegeben und war schlecht erreichbar.
5	5	4	5	5	5	4.75	Trotz Terminschwierigkeiten meinerseits eine geduldige und freundliche Person.
6	6	5	5	5	5	5.00	Sehr freundlicher Nutzer, Buch kam pünktlich und in perfektem Zustand zurück.
7	7	2	2	2	1	1.75	Sie war unfreundlich und hat das Buch mit beschädigtem Einband zurückgegeben.
8	9	5	4	3	4	4.00	Sie war sehr freundlich und hat mich aktiv auf Qualitätsmängel zum Buch hingewiesen.
9	10	4	5	5	5	4.75	Super zuverlässig – Buch wurde wie vereinbart zurückgegeben, absolut problemlos.
10	11	5	5	3	3	4.00	Zuverlässiger Nutzer, aber die Seiten waren stellenweise geknickt.

Ergebnistabelle

UserID	FirstName	LastName	MobileNumber	LoginID	AddressID	RatingReceiver	RatingLender
1	Alejandro	Maier	+49 1571 2345678	1	1	4.00	*NULL*
2	Fatima	Al-Hassan	+49 1522 98766543	2	2	*NULL*	*NULL*
3	Yuki	Tanaka	+49 1573 4567890	3	3	4.75	3.75
4	Liam	O'Connor	+49 1521 6789123	4	4	*NULL*	3.92
5	Nora	Flaig	+49 1579 2341567	5	5	4.00	3.00
6	Levi	Becker	+49 1523 4567123	6	6	3.63	4.33
7	Zofia	Kowalska	+49 1578 3456789	7	7	2.75	4.50
8	Elia	Schlag	+49 1524 7891234	8	8	4.50	3.00
9	Chen	Wei	+49 1575 9876123	9	9	4.88	*NULL*
10	Hans	Müller	+49 1520 1234567	10	10	*NULL*	*NULL*

Aktualisierte Entität Userdata

Testfall für die Datenbank bezüglich ER-Modells

Präambel:
Lisa Paulsen, ein sprichwörtlicher „Lesewurm“, hat sehr viele Bücher zuhause, welche sie jedoch nie alle gleichzeitig lesen kann. Sie hat letztens von einer Buchtauch App gehört, womit regional Bücher zum Tauschen eingestellt werden können. Super, dachte sie sich, dann kann sie Ihre Bücher einstellen und anderen eine Freude machen.

Im Folgenden wird der oben erläuterte Testfall anhand der erstellten Datenbank dargestellt. Zu jedem Vorgang erfolgt eine kurze Beschreibung und der daraus folgende SQL-Befehl. Das Ergebnis des Vorgangs wird mithilfe einer SELECT Anweisung gezeigt.

Testfall

Lisa lädt die Buchtauch App auf ihr Smartphone herunter und registriert sich. Dazu muss sie ihre E-Mail-Adresse eintragen und ein Passwort vergeben.

```
INSERT INTO logindata (EmailAddress,Passwort) VALUES ('l.paulsen@gmx.de', PASSWORD('Lesewurm'));
```

Nun muss sie ihr Profil vervollständigen, indem sie ihren Vornamen, Nachnamen, Mobil Nummer und Adresse einträgt.

```
START TRANSACTION;
INSERT INTO address (Street,HouseNumber,CityID) VALUES ('Stuttgarter Straße',4,1);
INSERT INTO userdata (FirstName,LastName,MobileNumber,LoginID,AddressID) VALUES ('Lisa','Paulsen','+49 1578 9891245',11,11);
COMMIT;
```

Ergebnis

```
SELECT UD.FirstName, UD.LastName, UD.MobileNumber, A.Street, A.HouseNumber, CT.Postcode, CT.Name AS City, CY.Name AS Country, LD.EmailAddress, LD.Passwort FROM userdata UD INNER JOIN logindata LD USING(LoginID) INNER JOIN address A USING(AddressID) INNER JOIN city CT USING(CityID) INNER JOIN country CY USING(CountryID) WHERE UD.UserID=11;
```

FirstName	LastName	MobileNumber	Street	HouseNumber	Postcode	City	Country	EmailAddress	Passwort
Lisa	Paulsen	+49 1578 9891245	Stuttgarter Straße	4	72250	Freudenstadt	Deutschland	l.paulsen@gmx.de	*68F01921DDBAA44A6AAE67EEF4BF2DA9FB488F2B

Testfall

Lisa ist nun erfolgreich registriert. Nun möchte sie ihr Lieblingsbuch zum Ausleihen einstellt. Dazu trägt sie die Buchdaten und weitere Daten ein.

```
START TRANSACTION;
INSERT INTO weight (WeightInGrams) VALUES (226);
INSERT INTO dimensions (LengthWidthHeightInCm) VALUES ('17,60/11,10/2,80');
INSERT INTO book (Title,ISBN,PublisherID,GenreID,LanguageID,PageID,CoverID,WeightID,DimensionsID,YearID) VALUES ('Before She Dissappeared','978-1-787-46439-1',12,2,2,8,2,17,17,8);
INSERT INTO author (FirstName, LastName) VALUES ('Lisa','Gardner‘);
INSERT INTO bookauthor (BookID,AuthorID) VALUES (17,15);
INSERT INTO booksummary (BookID,QualityID,UserID,StatusID,AvailableFrom,Mailing)  VALUES (17,3,11,1,'2024-12-25',false);
COMMIT;
```

```
SELECT B.Title, B.ISBN, NP.Pages, BC.Name AS Cover, W.WeightInGrams, D.LengthWidthHeightInCm, YP.Year AS YearOfPublication, A.FirstName, A.LastName, P.Name AS Publisher, G.Genre, L.Name AS Language, Q.Name AS Quality, S.Name AS Status, BS.AvailableFrom FROM book B INNER JOIN publisher P USING(PublisherID) INNER JOIN genre G USING (GenreID) INNER JOIN language L USING(LanguageID) INNER JOIN numberofpages NP USING(PageID) INNER JOIN bookcover BC USING(CoverID) INNER JOIN weight W USING(WeightID) INNER JOIN dimensions D USING(DimensionsID) INNER JOIN yearofpublication YP USING(YearID) INNER JOIN booksummary BS USING(BookID) INNER JOIN bookquality Q USING(QualityID) INNER JOIN status S USING(StatusID) INNER JOIN bookauthor BA USING(BookID) INNER JOIN author A USING(AuthorID) WHERE BS.UserID=11 AND B.BookID=17;
```

Ergebnis

Title	ISBN	Pages	Cover	WeightInGrams	LengthWidthHeightInCm	YearOfPublication	FirstName	LastName	Publisher	Genre	Language	Quality	Status	AvailableFrom
Before She Dissappeared	978-1-787-46439-1	416	Taschenbuch	226	17,60/11,10/2,80	2021	Lisa	Gardner	Penguin Books Ltd	Englische Bücher	englisch	leichte Gebrauchsspuren	verfügbar	2024-12-25

Testfall

Lisas Lieblingsbuch ist nun eingetragen und für die andern Nutzer sichtbar.
Liam O'Connor möchte bereits seit längerem ein englisches Buch lesen, jedoch ist er sich unsicher, ob er den Inhalt des Buches versteht und möchte daher auf ein Kauf verzichten. Daher geht er in die Buchtauschapp und sucht nach einem englischen Buch mit seinen gewünschten Kriterien.

```
SELECT B.Title, YP.Year AS YearOfPublication, NP.Pages, G.Genre FROM booksummary INNER JOIN book B USING(BookID) INNER JOIN yearofpublication YP USING(YearID) INNER JOIN numberofpages NP USING(PageID) INNER JOIN genre G USING(GenreID)
WHERE B.LanguageID=2 AND Booksummary.StatusID=1 AND YP.Year>2020;
```

Ergebnis

Title	YearOfPublication	Pages	Genre
Before She Dissappeared	2021	416	Englische Bücher

Testfall

Liam wird das Buch von Lisa angezeigt und er nimmt über die App Kontakt zu ihr auf. Sie freut sich sehr, dass sich jemand für ihr Lieblingsbuch interessiert und vereinbart mit ihm den 25.12.2024 14 Uhr, um das Buch abzuholen. Lisa ändert dabei den Status des Buches auf reserviert.

```
Update booksummary SET StatusID=2 WHERE SummaryID=17;
```

```
SELECT B.Title, S.Name AS Status FROM book B INNER JOIN booksummary BS USING(BookID) INNER JOIN status S
USING(StatusID) WHERE BS.SummaryID=17;
```

Title	Status
Before She Disappeared	reserviert

Ergebnis

Testfall

Liam kommt am 25.12.2024 um 14 Uhr zu Lisa nach Hause. Zu Beginn tauschen sie sich über das Buch aus und wickeln dann den Tausch ab. Dazu muss Lisa in der App den Tausch starten und Liams UserID eintragen. Liam wiederrum muss den Tausch in seiner App bestätigen. Das Rückgabedatum legen die beiden auch direkt fest.

```
START TRANSACTION;
INSERT INTO borrowingperiod (SummaryID, UserID, BorrowingStart, BorrowingEnd) VALUES (17,4,'2024-12-25 14:03:22','2025-12-30');
Update booksummary SET StatusID=3, AvailableFrom=DATE_ADD('2024-12-30',INTERVAL 1 DAY) WHERE SummaryID=17;
COMMIT;
```

Mit Liams Bestätigung in der App, wir der Ausleihbeginn mit dem aktuellen Zeitstempel versehen und der Status des Buches auf „ausgeliehen“ gesetzt. Der Wert bei BorrowingStart wird normalerweise durch den Default CURRENT_TIMESTAMP beim Hinzufügen festgelegt. Da es sich hierbei um einen Testfall handelt, wird der Wert manuell eingetragen.

AvailableFrom wird standardmäßig ausgehend vom Tag des Zurückgebens um ein Tag addiert, um einen Puffer zum nächsten Tausch zu erzeugen.

Ergebnis

```
SELECT B.Title, S.Name AS Status, BS.AvailableFrom, BP.BorrowingStart, BP.BorrowingEnd, UD.FirstName AS FirstNameReceiver, UD.LastName AS LastNameReceiver FROM book B INNER JOIN booksummary BS USING(BookID) INNER JOIN status S USING(StatusID) INNER JOIN borrowingperiod BP USING(SummaryID) INNER JOIN Userdata UD ON UD.UserID=BP.UserID WHERE BorrowID=16;
```

Title	Status	AvailableFrom	BorrowingStart	BorrowingEnd	FirstNameReceiver	LastNameReceiver
Before She Dissappeared	zurzeit ausgeliehen	2024-12-31	2024-12-25 14:03:22	2025-12-30 00:00:00	Liam	O‘Connor

Testfall

Liam hat das Buch bereits fertig gelesen und nimmt wieder Kontakt zu Lisa auf. Sie vereinbaren einen Termin am 29.12.2024 15 Uhr zur Rückgabe des Buches. Liam kommt bei Lisa vorbei, er gibt das Buch zurück und sie reden noch kurz über das Buch. Anschließend vermerkt Liam die Rückgabe in der App, was wiederrum von Lisa in der App bestätigt werden muss.

```
START TRANSACTION;
Update borrowingperiod SET BorrowingEnd='2024-12-29 15:04:45' WHERE BorrowID=16;
Update booksummary SET StatusID=1, AvailableFrom= DATE_ADD('2024-12-29',INTERVAL 1 DAY) WHERE SummaryID=17;
COMMIT;
```

Mit Lisas Bestätigung in der App, wir das Ausleihende mit dem aktuellen Zeitstempel versehen und der Status des Buches auf „verfügbar“ gesetzt. Der Wert bei BorrowingEnd wird normalerweise durch CURRENT_TIMESTAMP festgelegt. Da es sich hierbei um einen Testfall handelt, wird der Wert manuell eingetragen.

Ergebnis

```
SELECT B.Title, S.Name AS Status, BP.BorrowingEnd, BS.AvailableFrom FROM book B INNER JOIN booksummary BS
USING(BookID) INNER JOIN status S USING(StatusID) INNER JOIN borrowingperiod BP USING(SummaryID) WHERE BorrowID=16;
```

Title	Status	BorrowingEnd	AvailableFrom
Before She Dissappeared	verfügbar	2024-12-29 15:04:45	2024-12-30

Testfall

Nach der Rückgabe des Buches bekommen Liam und Lisa die Möglichkeit sich gegenseitig zu bewerten. Außerdem kann Liam noch das Buch bewerten. Beide nehmen sich die Zeit und geben ihre Bewertungen ab.

Liams Bewertung des Buches:
INSERT INTO bookrating (BorrowID,Quality,Content,Recommendation,FreeText) VALUES (16,4,5,5,'Mein erstes englisches Buch, habe es geliebt :)');

Liams Bewertung über Lisa:
INSERT INTO userratinglender (BorrowID,Friendliness,Reliability,AccuracyOfTheBook,Recommendation,FreeText) VALUES (16,4,5,5,5,'Terminvereinbarung lief super, das Haus war auch gut zu finden');

Lisas Bewertung über Liam:
INSERT INTO userratingreceiver (BorrowID,Friendliness,Reliability,BookQualityAfterReturn,Recommendation,FreeText) VALUES (16,5,5,5,5,'Super Erfahrung zu meinem ersten Tausch mit ihm gehabt');

Testfall

```
SELECT B.Title, BR.Quality, BR.Content, BR.Recommendation, BR.FreeText, BS.Bookrating FROM bookrating BR INNER JOIN
borrowingperiod BP USING (BorrowID) INNER JOIN booksummary BS USING (SummaryID) INNER JOIN book B USING(BookID)
WHERE BP.BorrowID=16;
```

Title	Quality	Content	Recommendation	FreeText	Bookrating
Before She Dissappeared	4	5	5	Mein erstes englisches Buch, habe es geliebt :)	4.67

```
SELECT UD.FirstName AS FirstNameLender, UD.LastName AS LastNameLender, UD.RatingLender, URL.Friendliness,
URL.Reliability, URL.AccuracyOfTheBook, URL.Recommendation, URL.FreeText FROM userratinglender URL INNER JOIN
borrowingperiod BP USING (BorrowID) INNER JOIN booksummary BS USING (SummaryID) INNER JOIN userdata UD ON
BS.UserID=UD.UserID WHERE BP.BorrowID=16;
```

FirstNameLender	LastNameLender	RatingLender	Friendliness	Reliability	AccuracyOfTheBook	Recommendation	FreeText
Lisa	Paulsen	4.75	4	5	5	5	Terminvereinbarung lief super, das Haus war auch gut zu finden

```
SELECT UD.FirstName AS FirstNameReceiver, UD.LastName AS LastNameReceiver, UD.RatingReceiver, URR.Friendliness,
URR.Reliability, URR.BookQualityAfterReturn, URR.Recommendation, URR.FreeText FROM userratingreceiver URR INNER JOIN
borrowingperiod BP USING (BorrowID) INNER JOIN booksummary BS USING (SummaryID) INNER JOIN userdata UD ON
BP.UserID=UD.UserID WHERE BP.BorrowID=16;
```

FirstNameReceiver	LastNameReceiver	RatingReceiver	Friendliness	Reliability	BookQualityAfterReturn	Recommendation	FreeText
Liam	O'Connor	5.00	5	5	5	5	Super Erfahrung zu meinem ersten Tausch mit ihm gehabt

Ergebnis