Buchtauschapp

Erklärung des Datenbankdesign / Implementierungsverfahren / Testfälle

Entität - Standortfreigabe

```
CREATE TABLE buchtausch_app.Standortfreigabe (
freigabe_ID INT AUTO_INCREMENT PRIMARY KEY,
freigabe_beschreibung ENUM ('freigabe', 'abholfreigabe', 'keine freigabe'));
```

 $\textbf{INSERT INTO} \ buchtausch_app. Standort freigabe$

VALUES

- (1, 'freigabe'),
- (2, 'abholfreigabe'),
- (3, 'keine freigabe');

- Erstellung der Tabelle "Standortfreigabe" mit den entsprechenden Attributen und festlegen der Datentypen
 - Auto_Increment: Automatisches einfügen des Wertes (x+1), dadurch eindeutige ID eines Datensatzes
 - Primary Key: Festlegung des Primary Keys
 - ENUM: Beschränken der möglichen Eingabe auf entsprechende Werte
- Einfügen der Dummy-Daten



Entität – Zustand

```
CREATE TABLE buchtausch_app.Zustand (
zustand_ID INT AUTO_INCREMENT PRIMARY KEY,
zustand_beschreibung ENUM ('Neu', 'Gut', 'Schlecht')
);
```

INSERT INTO buchtausch_app.Zustand (zustand_ID, zustand_beschreibung) **Values**

- (1, 'Neu'),
- (2, 'Gut'),
- (3, 'Schlecht');

- Erstellung der Tabelle "Zustand" mit den entsprechenden Attributen und festlegen der Datentypen
 - Auto_Increment: Automatisches einfügen des Wertes (x+1), dadurch eindeutige ID eines Datensatzes
 - Primary Key: Festlegung des Primary Keys
 - ENUM: Beschränken der möglichen Eingabe auf entsprechende Werte
- Einfügen der Dummy-Daten



Entität – Versandoption

```
CREATE TABLE buchtausch_app.Versandoption (
versand_ID INT AUTO_INCREMENT PRIMARY KEY,
zustellung_beschreibung ENUM ('Versand', 'Selbstabholung')
);
```

INSERT INTO buchtausch_app.Versandoption (versand_ID , zustellung_beschreibung) **Values**

- (1, 'Versand'),
- (2, 'Selbstabholung');

- Erstellung der Tabelle "Zustand" mit den entsprechenden Attributen und festlegen der Datentypen
 - Auto_Increment: Automatisches einfügen des Wertes (x+1), dadurch eindeutige ID eines Datensatzes
 - Primary Key: Festlegung des Primary Keys
 - ENUM: Beschränken der möglichen Eingabe auf entsprechende Werte
- Einfügen der Dummy-Daten



Entität – Autor

CREATE TABLE buchtausch app. Autor (

```
autor_ID INT AUTO_INCREMENT PRIMARY KEY,
autor_nachname VARCHAR(50),
autor_name VARCHAR(50)
INSERT INTO Autor (autor_ID, autor_nachname, autor_name)
VALUES
(1, 'Kehlmann', 'Daniel'),
(2, 'Tellkamp', 'Uwe'),
(3, 'Stanišić', 'Saša'),
(4, 'Zeh', 'Juli'),
(5, 'Menasse', 'Robert'),
(6, 'Lüscher', 'Jonas'),
(7, 'Bierbichler', 'Josef'),
(8, 'Othmann', 'Ronya'),
(9, 'Erpenbeck', 'Jenny'),
(10, 'Leky', 'Mariana'),
(11, 'Bjerg', 'Bov'),
(12, 'Würger', 'Takis'),
(13, 'Rothmann', 'Ralf'),
(14, 'McCarthy', 'Cormac'),
(15, 'Rooney', 'Sally'),
(16, 'Paolini', 'Christopher');
```

- Erstellung der Tabelle "Zustand" mit den entsprechenden Attributen und festlegen der Datentypen
 - Auto_Increment: Automatisches einfügen des Wertes (x+1), dadurch eindeutige ID eines Datensatzes
 - Primary Key: Festlegung des Primary Keys
 - Varchar(50): Beschränken der möglichen Werte auf 50
- Einfügen der Dummy-Daten



Entität - Leihdauer

```
CREATE TABLE buchtausch_app.Leihdauer (
dauer_ID INT AUTO_INCREMENT PRIMARY KEY,
dauer_in_tagen INT CHECK (dauer_in_tagen=7 OR dauer_in_tagen=14 OR dauer_in_tagen=21
OR dauer_in_tagen=28 OR dauer_in_tagen=60) DEFAULT 28
);
```

INSERT INTO buchtausch_app.Leihdauer (dauer_ID, dauer_in_tagen)

Values

- (1, 7),
- (2, 14),
- (3, 21),
- (4, 28),
- (5, 60);

- Erstellung der Tabelle "Zustand" mit den entsprechenden Attributen und festlegen der Datentypen
 - Auto_Increment: Automatisches einfügen des Wertes (x+1), dadurch eindeutige ID eines Datensatzes
 - Primary Key: Festlegung des Primary Keys
 - Check: Beschränken der möglichen Eingabe auf entsprechende Werte (siehe Klammer). Default legt einen automatischen Wert fest, falls keine Eingabe erfolgt
- Einfügen der Dummy-Daten



Entität – Verlag

```
CREATE TABLE buchtausch_app.Verlag (
verlag_ID INT AUTO_INCREMENT PRIMARY KEY,
verlag_name VARCHAR(50)
);
```

```
INSERT INTO Verlag (verlag_ID, verlag_name)
VALUES
(1, 'Rowohlt Verlag'),
(2, 'Suhrkamp Verlag'),
(3, 'Luchterhand Literaturverlag'),
(4, 'C.H. Beck'),
(5, 'Hanser Berlin'),
(6, 'Knaus Verlag'),
(7, 'Dumont Buchverlag'),
(8, 'Claassen Verlag'),
(9, 'Hanser Verlag'),
(10, 'Alfred A. Knopf'),
(11, 'Faber & Faber');
```

- Erstellung der Tabelle "Zustand" mit den entsprechenden Attributen und festlegen der Datentypen
 - Auto_Increment: Automatisches einfügen des Wertes (x+1), dadurch eindeutige ID eines Datensatzes
 - Primary Key: Festlegung des Primary Keys
 - Varchar(50): Beschränken der möglichen Werte auf 50
- Einfügen der Dummy-Daten



CREATE TABLE buchtausch app. Buecher (

```
buch_ID INT AUTO_INCREMENT PRIMARY KEY,

titel VARCHAR(100) NOT NULL,

genre VARCHAR (50) NOT NULL,

erscheinungsdatum DATE,

sprache CHAR(3)
);

ALTER TABLE buchtausch_app.Buecher

ADD COLUMN buch_status ENUM('verfuegbar', 'reserviert', 'ausgeliehen') DEFAULT

'verfuegbar',

ADD COLUMN FK_benutzer_ID INT,

ADD COLUMN FK_autor_ID INT,

ADD COLUMN Zustand_ID INT,

ADD COLUMN FK_leihdauer_ID INT,

ADD COLUMN FK_leihdauer_ID INT,

ADD COLUMN FK_versand_ID INT;
```

- Erstellung der Tabelle "Buecher" mit den entsprechenden Attributen und festlegen der Datentypen
 - Auto_Increment: Automatisches einfügen des Wertes (x+1), dadurch eindeutige ID eines Datensatzes
 - Primary Key: Festlegung des Primary Keys
 - Varchar(50): Beschränken der möglichen Werte auf 50
 - Not Null: Dieser Wert darf nicht leer (NULL) betragen
 - Date: Taggenaue Kalenderdaten
 - CHAR(3): Wert muss genau 3 Zeichen enthalten
- Nachträgliches hinzufügen der Fremdschlüsselspalten, um die Fremdschüsselverbindung zu den entsprechenden Tabellen herzustellen, sowie der Spalte status, welche die Information speichert, ob ein Buch aktuell reserviert, ausgeliehen oder verfügbar ist



-- Beziehung zur Tabelle Benutzer (benutzer_ID)
ALTER TABLE buchtausch_app.Buecher
ADD CONSTRAINT fk_benutzer
FOREIGN KEY (FK_benutzer_ID)
REFERENCES buchtausch_app.Benutzer(benutzer_ID)
ON DELETE CASCADE
ON UPDATE CASCADE;

-- Beziehung zur Tabelle Autor (autor_ID)
ALTER TABLE buchtausch_app.Buecher
ADD CONSTRAINT fk_autor
FOREIGN KEY (FK_autor_ID)
REFERENCES buchtausch_app.Autor(autor_ID)
ON DELETE CASCADE
ON UPDATE CASCADE;

-- Beziehung zur Tabelle Verlag (verlag_ID)

ALTER TABLE buchtausch_app.Buecher

ADD CONSTRAINT fk_verlag

FOREIGN KEY (FK_verlag_ID)

REFERENCES buchtausch_app.Verlag(verlag_ID)

ON DELETE CASCADE

ON UPDATE CASCADE;

--- Beziehung zur Tabelle Zustand (zustand_ID)

ALTER TABLE buchtausch_app.Buecher

ADD CONSTRAINT fk_zustand

FOREIGN KEY (zustand_ID)

REFERENCES buchtausch_app.Zustand(zustand_ID)

ON DELETE CASCADE

ON UPDATE CASCADE;

| Hier werden die Fremdschlüsselbeziehungen nachträglich zu der Entität Buecher hinzugefügt, um auf die Informationen zurückgreifen zu können: |
|--|
| Erstellen des Constraints fk_benutzer |
| Erstellen des Constraints fk_autor |
| |
| |
| |
| Erstellen des Constraints fk_verlag |
| |
| |
| Erstellen des Constraints fk_zustand |

-- Beziehung zur Tabelle Leihdauer (dauer_ID)

ALTER TABLE buchtausch_app.Buecher

ADD CONSTRAINT fk_leihdauer

FOREIGN KEY (FK_leihdauer_ID)

REFERENCES buchtausch_app.Leihdauer(dauer_ID)

ON DELETE CASCADE

ON UPDATE CASCADE;

-- Beziehung zur Tabelle Versandoption (versand_ID)

ALTER TABLE buchtausch_app.Buecher

ADD CONSTRAINT fk_versand

FOREIGN KEY (FK_versand_ID)

REFERENCES buchtausch_app.Versandoption(versand_ID)

ON DELETE CASCADE

ON UPDATE CASCADE;

• Erstellen des Constraints fk_leihdauer

• Erstellen des Constraints fk_versand



INSERT INTO buchtausch_app.Buecher (buch_ID, titel, FK_autor_ID, FK_verlag_ID, genre, erscheinungsdatum , sprache, FK_benutzer_ID, zustand_ID, FK_leihdauer_ID, FK_versand_ID) **VALUES**

- (1, 'Tyll', 1, 1, 'Historischer Roman', '2017-10-26', 'DE', 51, 2, 4, 1),
- (2, 'Der Turm', 2, 2, 'Gesellschaftsroman', '2008-09-01', 'DE', 60, 3, 5, 2),
- (3, 'Herkunft', 3, 3, 'Autobiografie', '2019-03-18', 'DE', 55, 2, 4, 1),
- (4, 'Unterleuten', 4, 3, 'Gesellschaftsroman', '2016-03-08', 'DE', 53, 2, 3, 1),
- (5, 'Die Hauptstadt', 5, 2, 'Gesellschaftsroman', '2017-09-11', 'DE', 52, 1, 5, 2),
- (6, 'Kraft', 6, 4, 'Gesellschaftsroman', '2017-02-13', 'DE', 58, 3, 3, 2),
- (7, 'Mittelreich', 7, 2, 'Familiensaga', '2011-08-25', 'DE', 59, 1, 4, 1),
- (8, 'Die Sommer', 8, 5, 'Roman', '2020-08-31', 'DE', 54, 1, 4, 1),
- (9, 'Gehen, ging, gegangen', 9, 6, 'Gesellschaftsroman', '2015-09-14', 'ENG', 55, 2, 5, 1),
- (10, 'Was man von hier aus sehen kann', 10, 7, 'Roman', '2017-07-21', 'DE', 55, 2, 3, 1),
- (11, 'Serpentinen', 11, 8, 'Roman', '2020-01-29', 'DE', 57, 3, 3, 1),
- (12. 'Stella', 12. 9. 'Roman', '2019-01-14', 'DE', 53. 1. 2. 2).
- (13, 'Im Frühling sterben', 13, 2, 'Historischer Roman', '2015-02-08', 'DE', 56, 2, 1, 2),
- (14, 'The Road', 14, 10, 'Dystopie', '2006-09-26', 'DE', 51, 1, 5, 1),
- (15, 'Normal People', 15, 11, 'Romanze', '2018-08-30', 'ENG', 53, 1, 3, 1),
- (16, 'Eragon: The Inheritance Cycle, Book 1', 16, 10, 'Fantasy', '2003-08-26', 'DE', 51, 2, 4, 1),
- (17, 'Eldest: The Inheritance Cycle, Book 2', 16, 10, 'Fantasy', '2005-08-23', 'DE', 51, 2, 4, 1),
- (18, 'Brisingr: The Inheritance Cycle, Book 3', 16, 10, 'Fantasy', '2008-09-20', 'DE', 51, 2, 4, 1);

• Einfügen der Dummy-Daten



```
CREATE EVENT update_book_status_daily
ON SCHEDULE EVERY 1 DAY
STARTS '2024-09-10 00:00:00'
DO
BEGIN
UPDATE buchtausch_app.Buecher
SET status = 'verfuegbar';
UPDATE buchtausch_app.Buecher
SET buch_status = 'reserviert'
WHERE buch_ID IN (
SELECT FK_buch_ID
FROM buchtausch_app.Reservierung
WHERE res_status = 'offen'
UPDATE buchtausch_app.Buecher
SET buch_status = 'ausgeliehen'
WHERE buch_ID IN (
SELECT FK_buch_ID
FROM buchtausch_app.Ausleihvorgang
WHERE leih status = 'aktiv'
);
END;
```

• Erstellen des automatischen Updates der Spalte "buch_status" • Aktualisiere den Status auf "reserviert", wenn es eine offene Reservierung gibt • . Aktualisiere den Status auf "ausgeliehen", wenn das Buch ausgeliehen ist



Entität – Reservierung

```
CREATE TABLE buchtausch app. Reservierung (
 reservierung ID INT AUTO_INCREMENT PRIMARY KEY,
 reservierungsdatum TIMESTAMP,
 res startdatum DATE,
 res enddatum DATE,
 res_status ENUM('offen', 'inaktiv') DEFAULT ('offen'),
 FK buch ID INT,
FOREIGN KEY (FK_buch_ID) REFERENCES buchtausch_app.Buecher (buch_ID)
 ON DELETE CASCADE ON UPDATE CASCADE
 SET GLOBAL event scheduler = ON;
 CREATE EVENT update_res_status_daily
 ON SCHEDULE EVERY 1 DAY
 STARTS CURRENT DATE() + INTERVAL 1 DAY
 DO
 UPDATE buchtausch app.Reservierung
 SET res status = 'inaktiv'
 WHERE res_enddatum < CURRENT_DATE();</pre>
 INSERT INTO buchtausch app.Reservierung (reservierung ID, reservierungsdatum,
 res_startdatum, res_enddatum, res_status, FK_buch_ID)
 Values
 (1, '2024-08-31', '2024-09-01', '2024-09-29', 'offen', 3),
 (2, '2024-08-25', '2024-08-26', '2024-09-02', inaktiv', 13),
 (3, '2024-03-01', '2024-03-05', '2024-04-02', 'inaktiv', 17);
```

- Erstellung der Tabelle "Reservierung" mit den entsprechenden Attributen und festlegen der Datentypen
 - Auto_Increment: Automatisches einfügen des Wertes (x+1), dadurch eindeutige ID eines Datensatzes
 - Primary Key: Festlegung des Primary Keys
 - ENUM: Beschränken der möglichen Eingabe auf entsprechende Werte
 - INT: Ganze Zahlen
 - Foreign Key … references: Hinzufügen der Fremdschlüsselbeziehung zu der Tabelle Buecher
 - Date: Taggenaue Kalenderdaten
 - Timestamp: Genaue Bestimmung des Zeitpunktes
- Automatisierung des Status (täglich), sobald das Enddatum der Reservierung in der Vergangenheit liegt, wird der Status auf "inaktiv" gesetzt

Einfügen der Dummy-Daten



Entität – Ausleihvorgang

```
CREATE TABLE buchtausch app. Ausleihvorgang (
 leih vorgang ID INT AUTO_INCREMENT PRIMARY KEY,
 FK_buch_ID INT,
 FK empfaenger INT,
 leih startdatum DATE,
 leih_enddatum DATE,
 leih status ENUM ('aktiv', 'abgeschlossen'),
FOREIGN KEY (FK_buch_ID) REFERENCES buchtausch_app.Buecher (buch_ID)
 ON DELETE CASCADE ON UPDATE CASCADE,
 FOREIGN KEY (FK_empfaenger) REFERENCES buchtausch_app.Benutzer (benutzer_ID)
 ON DELETE CASCADE ON UPDATE CASCADE
 INSERT INTO buchtausch_app.Ausleihvorgang (leih_vorgang_ID, FK_buch_ID, FK_empfaenger,
 leih startdatum, leih enddatum, leih status)
 VALUES
 (4, 14, 33, '2024-03-01', '2024-03-29', 'abgeschlossen'),
 (1, 2, 32, '2024-08-31', '2024-10-30', 'aktiv'),
 (2, 7, 33, '2024-08-31', '2024-09-28', 'aktiv'),
 (3, 11, 38, '2024-08-31', '2024-09-21', 'aktiv');
 CREATE EVENT update leih status daily
 ON SCHEDULE EVERY 1 DAY
 STARTS CURRENT_DATE() + INTERVAL 1 DAY
 DO
 UPDATE buchtausch app. Ausleihvorgang
 SET leih status = 'abgeschlossen'
 WHERE leih enddatum < CURRENT_DATE();</pre>
```

- Erstellung der Tabelle "Ausleihvorgang" mit den entsprechenden Attributen und festlegen der Datentypen
 - Auto_Increment: Automatisches einfügen des Wertes (x+1), dadurch eindeutige ID eines Datensatzes
 - Primary Key: Festlegung des Primary Keys
 - ENUM: Beschränken der möglichen Eingabe auf entsprechende Werte
 - INT: Ganze Zahlen
 - Foreign Key … references: Hinzufügen der Fremdschlüsselbeziehung zu der Tabelle Buecher und Benutzer
 - Date: Taggenaue Kalenderdaten
- Einfügen der Dummy-Daten

 Automatisierung des Status (täglich), sobald das Enddatum des Ausleihvorgangs in der Vergangenheit liegt, wird der Status auf "abgeschlossen" gesetzt

Entität - Nachricht

```
CREATE TABLE buchtausch app. Nachricht (
nachricht ID INT AUTO_INCREMENT PRIMARY KEY,
FK nachricht empfaenger INT,
FK nachricht sender INT,
nachricht inhalt TEXT,
gesendet am TIMESTAMP,
FOREIGN KEY (FK nachricht empfaenger) REFERENCES
buchtausch app.Benutzer(benutzer ID)
ON DELETE CASCADE ON UPDATE CASCADE,
FOREIGN KEY (FK_nachricht_sender) REFERENCES buchtausch_app.Benutzer(benutzer_ID)
 ON DELETE CASCADE ON UPDATE CASCADE
INSERT INTO buchtausch app. Nachricht (nachricht ID, FK nachricht empfaenger,
FK nachricht sender, nachricht inhalt, gesendet am)
 VALUES
(1, 51, 52, 'Hallo! Wie geht es dir?', '2024-08-30 10:15:00'),
(2, 53, 51, 'Mir geht es gut, danke! Wie läuft es bei dir?', '2024-08-30 10:20:00'),
(3, 54, 55, 'Kannst du das Buch morgen zurückgeben?', '2024-08-30 11:05:00'),
(4, 56, 57, 'Ja, ich bringe es gegen 14 Uhr vorbei.', '2024-08-30 11:10:00'),
(5, 58, 59, 'Vielen Dank für die schnelle Ausleihe!', '2024-08-30 12:30:00'),
(6, 60, 51, 'Kein Problem, jederzeit wieder.', '2024-08-30 13:00:00'),
(7, 52, 53, 'Hast du das Buch schon durchgelesen?', '2024-08-30 14:15:00'),
(8, 54, 56, 'Ja, es war großartig! Kann ich das nächste ausleihen?', '2024-08-30 15:00:00'),
(9, 57, 58, 'Natürlich! Wann hättest du Zeit?', '2024-08-30 15:20:00'),
(10, 59, 60, 'Ich kann am Wochenende vorbeikommen.', '2024-08-30 16:00:00'),
(11, 51, 54, 'Das klingt gut. Bis dann!', '2024-08-30 16:30:00'),
(12, 55, 56, 'Wollte nur nochmal danke sagen für die Empfehlung.', '2024-08-30 17:00:00'),
(13, 57, 59, 'Gern geschehen! Es ist wirklich ein tolles Buch.', '2024-08-30 17:30:00'),
(14, 58, 60, 'Hast du auch das neue Buch von diesem Autor gelesen?', '2024-08-30 18:00:00'),
(15, 60, 51, 'Ja, das werde ich als nächstes ausleihen.', '2024-08-30 18:15:00');
```

- Erstellung der Tabelle "Nachricht" mit den entsprechenden Attributen und festlegen der Datentypen
 - Auto_Increment: Automatisches einfügen des Wertes (x+1), dadurch eindeutige ID eines Datensatzes
 - Primary Key: Festlegung des Primary Keys
 - ENUM: Beschränken der möglichen Eingabe auf entsprechende Werte
 - INT: Ganze Zahlen
 - Foreign Key … references: Hinzufügen der Fremdschlüsselbeziehung zu der Tabelle Buecher
 - Text: Sehr lange Zeichenkette
 - Timestamp: Genaue Bestimmung des Zeitpunktes
- Einfügen der Dummy-Daten



Entität - Benutzer

CREATE DATABASE buchtausch app;

```
CREATE TABLE buchtausch app.Benutzer
benutzer ID INT AUTO INCREMENT PRIMARY KEY,
nutzervorname VARCHAR(50) NOT NULL,
nutzernachname VARCHAR(50) NOT NULL,
geburtsdatum DATE,
email VARCHAR(50) NOT NULL,
telefonnummer VARCHAR(20)
ALTER TABLE buchtausch app.Benutzer
ADD COLUMN FK freigabe ID INT;
ALTER TABLE buchtausch app. Benutzer
ADD CONSTRAINT fk standortfreigabe
FOREIGN KEY (FK freigabe ID)
REFERENCES buchtausch app.Standortfreigabe(freigabe ID)
ON DELETE CASCADE
ON UPDATE CASCADE:
INSERT INTO buchtausch app.Benutzer (nutzervorname, nutzernachname, geburtsdatum,
email, telefonnummer, FK freigabe ID)
VALUES
('Max', 'Mustermann', '1990-01-15', 'max.mustermann@example.com', '015112345678', 1),
('Maria', 'Schmidt', '1985-05-24', 'maria.schmidt@example.com', '017612345678', 2),
('John', 'Doe', '1992-11-30', 'john.doe@example.com', '016312345678', 1),
('Anna', 'Müller', '1988-07-12', 'anna.mueller@example.com', '015512345678', 1),
('Peter', 'Pan', '1995-03-09', 'peter.pan@example.com', '017912345678', 1),
('Julia', 'Klein', '1993-09-17', 'julia.klein@example.com', '015812345678', 3),
('Stefan', 'Meyer', '1982-02-28', 'stefan.meyer@example.com', '016912345678', 1),
('Laura', 'Schneider', '1987-08-21', 'laura.schneider@example.com', '017512345678', 1),
('Thomas', 'Fischer', '1991-12-05', 'thomas.fischer@example.com', '015912345678',1),
('Nina', 'Krüger', '1994-06-15', 'nina.krueger@example.com', '016512345678', 2);
```

- Erstellung der Datenbank
- Erstellung der Tabelle "Benutzer" mit den entsprechenden Attributen und festlegen der Datentypen
 - Auto_Increment: Automatisches einfügen des Wertes (x+1), dadurch eindeutige ID eines Datensatzes
 - Primary Key: Festlegung des Primary Keys
 - Date: Taggenaue Kalenderdaten
 - Varchar(50): Beschränken der möglichen Werte auf 50
 - Not Null: Dieser Wert darf nicht leer (NULL) betragen
- Nachträgliches hinzufügen des Fremdschlüssels FK_freigabe_ID mit Bezug auf die Tabelle Standortfreigabe. Hier gibt es eine Änderung im Vergleich zum ER-Modell, denn dort ist die benutzer_ID als Fremdschlüssel in der Tabelle Standortfreigabe aufgeführt. Es macht allerdings mehr Sinn, über die Tabelle Benutzer auf die Information der Standortfreigabe zuzugreifen
- Einfügen der Dummy-Daten



Entität - Adresse

```
CREATE TABLE buchtausch app. Adresse (
adresse ID INT AUTO_INCREMENT PRIMARY KEY,
plz CHAR(5) NOT NULL,
ort VARCHAR(30) NOT NULL,
adresse VARCHAR(100) NOT NULL,
FK_benutzer_ID INT,
FOREIGN KEY (FK_benutzer_ID) REFERENCES buchtausch_app.Benutzer(benutzer_ID)
ON DELETE CASCADE ON UPDATE CASCADE
INSERT INTO buchtausch_app.Adresse (adresse_ID, plz, ort, adresse, FK_benutzer_ID) VALUES
(1, '40212', 'Düsseldorf', 'Königsallee 1', 51),
(2, '50667', 'Köln', 'Domstraße 5', 52),
(3, '45130', 'Essen', 'Rüttenscheider Straße 21', 53),
(4, '44137', 'Dortmund', 'Kampstraße 12', 54),
(5, '42103', 'Wuppertal', 'Alte Freiheit 6', 55),
(6, '48143', 'Münster', 'Prinzipalmarkt 8', 56),
(7, '41061', 'Mönchengladbach', 'Hindenburgstraße 33', 57),
(8, '47051', 'Duisburg', 'Königstraße 15', 58),
(9, '47798', 'Krefeld', 'Rheinstraße 25', 59),
(10, '45879', 'Gelsenkirchen', 'Bahnhofstraße 14', 60);
```

- Erstellung der Tabelle "Benutzer" mit den entsprechenden Attributen und festlegen der Datentypen
 - Auto_Increment: Automatisches einfügen des Wertes (x+1), dadurch eindeutige ID eines Datensatzes
 - Primary Key: Festlegung des Primary Keys
 - Date: Taggenaue Kalenderdaten
 - Varchar(n): Beschränken der möglichen Werte auf n Zeichen
 - INT: Ganze Zahlen
 - Not Null: Dieser Wert darf nicht leer (NULL) betragen
 - CHAR(5): Festlegen auf 5 Zeichen
 - Foreign Key ... references: Hinzufügen der Fremdschlüsselbeziehung zu der Tabelle Benutzer
- Einfügen der Dummy-Daten



Entität – Buchbewertung

```
CREATE TABLE buchtausch app. Buchbewertung (
 buch bewertung ID INT AUTO_INCREMENT PRIMARY KEY,
 FK_zustand_ID INT,
 FK buch ID INT,
 buch bewertungstext TEXT,
 buch sternebewertung INT CHECK(buch sternebewertung >= 1 AND buch sternebewertung
 <= 5),
FOREIGN KEY (FK zustand ID) REFERENCES buchtausch app.Zustand (zustand ID)
 ON DELETE CASCADE ON UPDATE CASCADE,
 FOREIGN KEY (FK buch ID) REFERENCES buchtausch app. Buecher (buch ID)
 ON DELETE CASCADE ON UPDATE CASCADE
 INSERT INTO buchtausch app. Buchbewertung (buch bewertung ID, FK zustand ID,
 FK buch ID, buch bewertungstext, buch sternebewertung)
 VALUES
 (1, 2, 1, 'Ein faszinierender historischer Roman mit tiefgründigen Charakteren.', 5),
 (2, 3, 2, 'Ein anspruchsvoller Gesellschaftsroman, der zum Nachdenken anregt.', 4),
 (3, 2, 3, 'Eine bewegende Autobiografie, die sehr authentisch wirkt.', 4),
 (4, 2, 4, 'Gut geschrieben, aber das Thema hat mich nicht wirklich gepackt.', 3),
 (5, 1, 5, 'Ein Meisterwerk der Gesellschaftsliteratur, sehr empfehlenswert.', 5),
 (6, 3, 6, 'Ein solides Buch, aber es fehlt ein wenig an Spannung.', 3),
 (7, 2, 7, 'Eine tiefgründige Familiensaga, die sich über Generationen erstreckt.', 4),
 (8, 1, 8, 'Ein wunderschöner Roman über die Freuden und Herausforderungen des
 Sommers.', 5),
 (9, 2, 10, 'Ein emotionales Buch mit starken Charakteren.', 4),
 (10, 3, 14, 'Eine düstere und fesselnde Dystopie, die unter die Haut geht.', 5);
```

- Erstellung der Tabelle "Buchbewertung" mit den entsprechenden Attributen und festlegen der Datentypen
 - Auto_Increment: Automatisches einfügen des Wertes (x+1), dadurch eindeutige ID eines Datensatzes
 - Primary Key: Festlegung des Primary Keys
 - INT: Ganze Zahlen
 - Check: Beschränken der möglichen Eingabe auf entsprechende Werte (siehe Klammer)
 - Text: Lange Zeichenketten
 - Foreign Key … references: Hinzufügen der Fremdschlüsselbeziehung zu der Tabelle Buecher und Zustand
- Einfügen der Dummy-Daten



Entität -Nutzerbewertung

```
CREATE TABLE buchtausch app. Nutzerbewertung (
nutzer bewertungs ID INT AUTO_INCREMENT PRIMARY KEY,
FK bewertungs empfaenger INT,
FK bewertungs verfasser INT,
nutzer bewertungstext VARCHAR(150),
nutzer_sternebwertung INT CHECK(nutzer_sternebwertung >= 1 AND nutzer_sternebwertung <= 5),
FOREIGN KEY (FK bewertungs empfaenger) REFERENCES buchtausch app.Benutzer(benutzer ID)
ON DELETE CASCADE ON UPDATE CASCADE,
FOREIGN KEY (FK bewertungs verfasser) REFERENCES buchtausch app.Benutzer (benutzer ID)
 ON DELETE CASCADE ON UPDATE CASCADE
INSERT INTO buchtausch app.Nutzerbewertung (nutzer bewertungs ID, FK bewertungs empfaenger,
FK bewertungs verfasser, nutzer bewertungstext, nutzer sternebwertung)
VALUES
(6, 59, 58, 'Leider zu lange gewartet, sonst in Ordnung.', 2),
(7, 53, 55, 'Sehr freundlicher und zuverlässiger Nutzer!', 5),
(8, 52, 54, 'Schnelle Rückgabe, gerne wieder.', 5),
(9, 53, 56, 'Kommunikation war etwas schwierig, aber sonst okay.', 3),
(10, 54, 52, 'Buch wurde leider beschädigt zurückgegeben.', 2),
(11, 55, 60, 'Perfekt! Alles lief reibungslos.', 5),
(1, 52, 51, 'Sehr freundlicher Tauschpartner, alles lief reibungslos!', 5),
(2, 53, 56, 'Das Buch war leider beschädigt, aber die Kommunikation war gut.', 3),
(3, 54, 57, 'Super schnelle Lieferung und sehr nett! Gerne wieder.', 5),
(4, 55, 60, 'Alles in Ordnung, aber die Verpackung war etwas schlecht.', 4),
(5, 59, 58, 'Leider zu lange gewartet, sonst in Ordnung.', 2);
```

- Erstellung der Tabelle "Nutzerbewertung" mit den entsprechenden Attributen und festlegen der Datentypen
 - Auto_Increment: Automatisches einfügen des Wertes (x+1), dadurch eindeutige ID eines Datensatzes
 - Primary Key: Festlegung des Primary Keys
 - INT: Ganze Zahlen
 - Varchar(150): Festlegen auf 150 Zeichen
 - Foreign Key ... references: Hinzufügen der Fremdschlüsselbeziehung zu der Tabelle Benutzer
 - Check: Beschränken der möglichen Eingabe auf entsprechende Werte (siehe Klammer)
- Einfügen der Dummy-Daten



Testfall 1 -Registrierung

- Bei der Registrierung in der App wird ein neuer Nutzer mit den entsprechenden Daten hinzugefügt.
- Beispiele hierfür wurden bereits bei der Dokumentation der Tabelle "Nutzer" aufgezeigt, unten ein Einzelbeispiel mit entsprechendem Ergebnis aus der Datenbank:
- **INSERT INTO** buchtausch_app.Benutzer (nutzervorname, nutzernachname, geburtsdatum, email, telefonnummer, FK_freigabe_ID)

VALUES

('Max', 'Mustermann', '1990-01-15', 'max.mustermann@example.com', '015112345678', 1);

| - | 0 pg benutzer_ID ▼ | A-Z nutzervorname 🔻 | A-Z nutzernachname 🔻 | Ø geburtsdatum ▼ | A-Z email ▼ | A-Z telefonnummer ▼ | ୦-≘ FK_freigabe_ID ▼ |
|---|--------------------|---------------------|----------------------|------------------|----------------------------|---------------------|----------------------|
| | 51 | Max | Mustermann | 1990-01-15 | max.mustermann@example.com | 015112345678 | 1 🗹 |

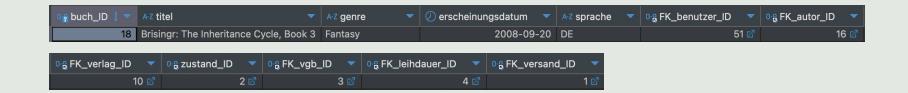


Testfall 2 –Speichern von Büchern

- Nach der Registrierung besteht die Möglichkeit, Bücher mit Eingabe aller relevanten zu speichern und zum Verleih bereitzustellen
- Beispiele hierfür wurden bereits bei der Dokumentation der Tabelle "Buecher" aufgezeigt, unten ein Einzelbeispiel mit entsprechendem Ergebnis aus der Datenbank:
- INSERT INTO buchtausch_app.Buecher (buch_ID, titel, FK_autor_ID, FK_verlag_ID, genre, erscheinungsdatum, sprache, FK_benutzer_ID, zustand_ID, FK_vgb_ID, FK_leihdauer_ID, FK_versand_ID)

VALUES

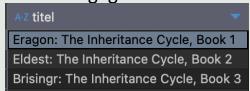
(18, 'Brisingr: The Inheritance Cycle, Book 3', 16, 10, 'Fantasy', '2008-09-20', 'DE', 51, 2, 3, 4, 1);





Testfall 3 –Büchersuche

- Jedem Nutzer ist es möglich, ein bestimmtes Buch auf Basis der eingegebenen Metadaten zum Buch zu suchen:
- Bsp. 1: Suche nach Genre
 - SELECT titel FROM Buecher WHERE genre = "Fantasy";



titel

Eragon: The Inheritance Cycle, Book 1

Eldest: The Inheritance Cycle, Book 2

Brisingr: The Inheritance Cycle, Book 3

- Bsp. 2: Suche nach Autor
 - **SELECT** Buecher.titel,

CONCAT (Autor.autor_name, ' ', Autor.autor_nachname) **AS** *autor_vollname* **FROM** buchtausch_app.Buecher

JOIN

buchtausch_app.Autor **ON** Buecher.FK_autor_ID = Autor.autor_ID

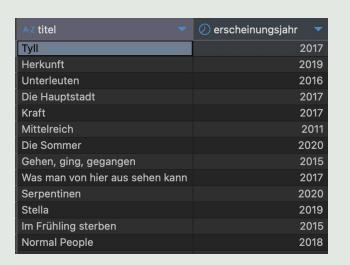
WHERE CONCAT(Autor.autor_name, '', Autor.autor_nachname) = 'Christopher Paolini';

- Bsp. 3: Suche nach Erscheinungsjahr
 - **SELECT** Buecher.titel,

YEAR(Buecher.erscheinungsdatum) AS erscheinungsjahr

FROM buchtausch app. Buecher

WHERE YEAR(Buecher.erscheinungsdatum) > 2010;





Testfall 4 –Standortfreigabe

- Bei Anmeldung ist es den Nutzern möglich anzugeben, inwiefern ihre Adresse genutzt werden kann.
 - Entweder ist der Standort vollständig freigegeben, gar nicht, oder wird nur für die Abholung freigegeben
- Bsp: Bei Anmeldung wird die vollständige Freigabe ausgewählt
 - INSERT INTO buchtausch_app.Benutzer (nutzervorname, nutzernachname, geburtsdatum, email, telefonnummer, FK_freigabe_ID)
 VALUE

 ('Max', 'Mustermann', '1990-01-15', 'max.mustermann@example.com', '015112345678', 1);
- Anbei die Informationen der bisher angemeldeten Nutzer:
 - SELECT

Benutzer.nutzervorname, Benutzer.nutzernachname, Standortfreigabe.freigabe_beschreibung **FROM** buchtausch_app.Benutzer

JOIN buchtausch app. Standortfreigabe ON Benutzer. FK freigabe ID = Standortfreigabe. freigabe ID;

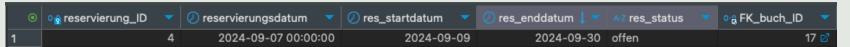
| • | A-Z nutzervorname 🔻 | A-Z nutzernachname 🔻 | A-Z freigabe_beschreibung 🔻 |
|----|---------------------|----------------------|-----------------------------|
| 1 | Max | Mustermann | freigabe |
| 2 | John | Doe | freigabe |
| 3 | Anna | Müller | freigabe |
| 4 | Peter | Pan | freigabe |
| 5 | Stefan | Meyer | freigabe |
| 6 | Laura | Schneider | freigabe |
| 7 | Thomas | Fischer | freigabe |
| 8 | Maria | Schmidt | abholfreigabe |
| 9 | Nina | Krüger | abholfreigabe |
| 10 | Julia | Klein | keine freigabe |



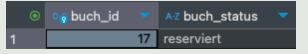
Testfall 5 –Reservierung von Bücher

- Wird eine Reservierung getätigt, wird ein neuer Datensatz der Entität "Reservierung" hinzugefügt, um die nötigen Informationen zu speichern
- Bsp.: Neue Reservierung wird getätigt:
 - INSERT INTO buchtausch_app.Reservierung (reservierung_ID, reservierungsdatum, res_startdatum, res_enddatum, res_status, FK_buch_ID) Values

(4, '2024-09-07', '2024-09-09', '2024-09-30', 'offen', 17);



- Entsprechend wird auch in der Tabelle Buecher der Status auf "Reserviert" gesetzt
 - SELECT buch_id, buch_status FROM Buecher WHERE buch_ID =17;





Testfall 6 –Leihen eines Buches

- Wird ein neuer Ausleihvorgang gestartet, wird ein neuer Datensatz der Entität "Ausleihvorgang" hinzugefügt, um die nötigen Informationen zu speichern
- Bsp.: Neuer Ausleihvorgang wird getätigt:
 - INSERT INTO buchtausch_app.Ausleihvorgang (leih_vorgang_ID, FK_buch_ID, FK_empfaenger, leih_startdatum, leih_enddatum, leih_status)

 VALUES

(5, 8, 57, '2024-09-07', '2024-11-06', 'aktiv');



- Entsprechend wird auch in der Tabelle Buecher der Status auf "ausgeliehen" gesetzt
 - **SELECT** buch_id, buch_status **FROM** Buecher **WHERE** buch_ID =8;





Testfall 7 –Bewertung von Buch & Nutzer

- Jeder Nutzer kann einen anderen Nutzer sowie Bücher bewerten.
- Bsp. Nutzerbewertung:
 - Auf Basis der eingegebenen Informationen wie einem kurzen Bewertungstext und einer Sternebewertung wird folgender Eintrag in die Datenbank eingefügt:
 - INSERT INTO buchtausch_app.Nutzerbewertung (nutzer_bewertungs_ID, FK_bewertungs_empfaenger, FK_bewertungs_verfasser, nutzer_bewertungstext, nutzer_sternebwertung)

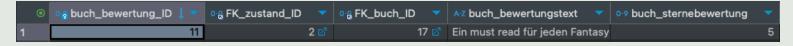
VALUES

(12, 51, 58, 'Es hat alles total reibungslos funktioniert.', 5);



- Bsp. Buchbewertung:
 - Auf Basis der eingegebenen Informationen wie einem kurzen Bewertungstext, dem Buchzustand und einer Sternebewertung wird folgender Eintrag in die Datenbank eingefügt:
 - INSERT INTO buchtausch_app.Buchbewertung (buch_bewertung_ID, FK_zustand_ID, FK_buch_ID, buch_bewertungstext, buch_sternebewertung)

 VALUES
 - (11, 2, 17, 'Ein must read für jeden Fantasy-Fan!', 5);





Testfall 8 -Senden von Nachrichten

- Um die Absprache zwischen den Nutzern zu vereinfachen, besteht die Möglichkeit, Nachrichten auszutauschen
- Diese werden wie folgt in der Tabelle Nachrichten gespeichert:
 - INSERT INTO buchtausch_app.Nachricht (nachricht_ID, FK_nachricht_empfaenger, FK_nachricht_sender, nachricht_inhalt, gesendet_am) VALUES

(16, 51, 52, 'Hallo! Ich würde gerne das Buch ausleihen.', '2024-09-07 10:15:00'),

(17, 51, 52, 'Klar, kein Thema!', '2024-09-07 10:20:00');

| • | ୍ରକୁ nachricht_ID ↓ ▼ | ಿಡ್ಡಿ FK_nachricht_empfaenger ▼ | ಿ-ಜ್ಞ FK_nachricht_sender ▼ | A-Z nachricht_inhalt | |
|---|-----------------------|---------------------------------|-----------------------------|--|---------------------|
| 1 | 17 | 51 ♂ | 52 ♂ | Klar, kein Thema! | 2024-09-07 10:20:00 |
| 2 | 16 | 51 ♂ | 52 ௴ | Hallo! Ich würde gerne das Buch ausleihen. | 2024-09-07 10:15:00 |



Ansicht aller aktuell verfügbaren Bücher

- Um den Nutzern einen Überblick über alle aktuell verfügbaren Bücher zu geben, wurde eine Ansicht erstellt, die genau diese Bücher zusammenfasst
- Hier ist somit schnell ersichtlich, welche Bücher gerade wo verfügbare sind. Dies fördert die Aktivität der Nutzer, da es die Übersichtlichkeit und Nutzerfreundlichkeit erleichtert

CREATE VIEW VerfuegbareBuecher AS

SELECT B.titel AS Buchtitel, A.autor_nachname AS Autor, AD.ort AS Ort

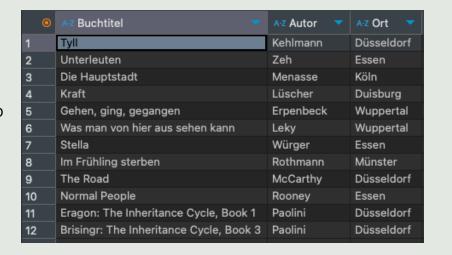
FROM buchtausch_app.Buecher B

JOIN buchtausch_app.Autor A ON B.FK_autor_ID = A.autor_ID

JOIN buchtausch_app.Benutzer U ON B.FK_benutzer_ID = U.benutzer_ID

JOIN buchtausch_app.Adresse AD ON U.benutzer_ID = AD.FK_benutzer_ID

WHERE B.buch status IN ('verfuegbar');





Zusammenfassung der Implementierung

- Änderungen zur Phase 1:
 - o Entfernen der Entität "Verfuegbarkeit"
 - Um die Performance der Datenbank zu verbessern. Es wären sonst weitere Fremdschlüsselbeziehungen zwischen den Entitäten Buecher, Reservierung, Ausleihvorgang und Verfuegbarkeit notwendig
 - So geschieht die Aktualisierung des Status der Bücher direkt in der Entität Buecher selbst, mithilfe der Fremdschlüsselbeziehungen zu den Entitäten Reservierung & Ausleihvorgang
- Vorgang für die Implementierung des Datenbankdesigns
 - o Zuerst wurden die Entitäten entsprechend dem ER-Modell erstellt, wie in den vorherigen Slides aufgeführt
 - o Anschließend wurden die Fremdschlüsselbeziehung erstellt. Auch dies ist in den vorherigen Slides ersichtlich
 - o Dann wurden die Dummy-Daten in die jeweiligen Tabellen eingefügt, um mit diesen die Testfälle entwickeln zu können
 - Die Testfälle sind ebenfalls in den vorherigen Slides mit dem entsprechenden SQL-Code und den Ergebnissen der Datenbank aufgeführt

