

<b>Art der Arbeit</b>	Projekt – Finalisierungsphase
<b>Kursbezeichnung</b>	DLBDSPBDM01_D – Projekt: Data-Mart-Erstellung in SQL
<b>Studiengang</b>	UPS-DPSQLEA – SQL Entwicklung und Administration
<b>Datum</b>	29.01.2026
<b>Verfasserin</b>	Stefanie Ellersiek
<b>Matrikelnummer</b>	UPS10788065

## Abstract

Die vorliegende Arbeit beschäftigt sich mit der Konzeption und Umsetzung einer relationalen Datenbank für eine Buchtausch-App unter Verwendung von MySQL als Datenbankmanagementsystem. Ziel war es, ein strukturiertes System zu entwickeln, das die Verwaltung von Anwendern, Büchern, Autoren, Adressen, Ausleihen, Rückgaben, Anfragen sowie Bewertungen ermöglicht und dabei sowohl die Datenintegrität als auch die einfache Abfrage der Informationen gewährleistet.

Die Datenbank „buchange“ ist modular aufgebaut und umfasst insgesamt zehn Tabellen. Zunächst wurden die zentralen Entitäten Anwender mit allen anwenderbezogenen Informationen sowie Buch festgelegt. Im Zuge der Normalisierung der Tabellen wurden aus diesen beiden Tabellen weitere Entitäten entwickelt, um Redundanzen und Inkonsistenzen vorzubeugen: Adresse, Abholinfo, Autor. Anschließend wurden Prozesse identifiziert und die darin anfallenden Daten in eigene Entitäten gruppiert, um auch hier Redundanzen und die Gefahr von Inkonsistenzen zu minieren. Darüber hinaus ermöglicht die Gliederung in eigenen Tabellen eine übersichtliche Abbildung von Status je Prozess und Prozesshistorien. Anschließend wurden für alle Entitäten die jeweiligen Attribute mit ihren Datentypen und Constraints definiert, um Beziehungen zwischen den Entitäten abzubilden und die referentielle Integrität sicherzustellen.

Nachdem das konzeptionelle Datenmodell erstellt war, wurde mit der technischen Umsetzung begonnen. Diese erfolgte in MySQL, da das System sowohl die einfache Verwaltung relationaler Daten als auch die effiziente Abfrage von Daten unterstützt. Als Open Source-Technologie ist sie kostenlos und bietet durch ihre weite Verbreitung und große Community eine solide Entwicklungsumgebung. Für den begrenzten und einfachen Daten- und Funktionsumfang der App ist die Technologie völlig ausreichend.

Die Erstellung der Datenbank erfolgte schrittweise:

1. Tabellendefinitionen: Zunächst wurden alle Tabellen mit ihren Attributen, Datentypen, Standardwerten und NULL-Erlaubnissen angelegt. Dabei wurden auch die Primär- und Fremdschlüssel implementiert.
2. Performance-Optimierung: Es wurde eine Indexierung der Fremdschlüssel in allen Tabellen angelegt. Durch spätere Tests konnte gezeigt werden, dass dies bei großen Datenmengen zu Performance-Verbesserungen bei Datenabfragen führt.
3. Initiale Datenbefüllung der Tabellen mit Testdaten: Beispielhafte Datensätze für Anwender, Bücher, Autoren, Adressen, Ausleihen, Rückgaben, Anfragen und Bewertungen wurden eingefügt, um die Funktionalität und die definierten Beziehungen zu testen.
4. Tabellenaktualisierungen: In Abhängigkeit der Prozesse in der App müssen einzelne Attribute in Tabellen aktualisiert werden. Dafür wurden UPDATE-Statements definiert.

5. Transaktionen: Die Befüllung und Aktualisierung von Tabellen, die prozessual zusammenhängen wurden in Transaktionen gebündelt. Als Isolationslevel für die definierten Transaktionen wurde Read Committed gewählt, da das Risiko für Dateninkonsistenzen bei diesem Level aufgrund der gewählten Datenstruktur gering ist, jedoch Anwendungs-Vorteile generiert werden können. Mit Hilfe eines Tests konnte gezeigt werden, dass Datenabfragen nicht blockiert werden, sondern der letzte bestätigte Datenstand angezeigt wird.
6. Views: Für die Ermittlung einiger Datensätze wurden Views definiert, da die Speicherung der Daten redundant gewesen wäre oder Daten zur Laufzeit berechnet werden müssen.
7. Sicherstellung der Datenqualität: Es wurde ein Architekturbild definiert, dass die Verantwortlichkeiten der Datenbank im Zusammenspiel mit Frontend und Backend verortet. Anschließend wurde gemäß den definierten Verantwortlichkeiten folgende Maßnahmen ergriffen:
  1. Es wurden Trigger definiert, die prüfen, ob die Ausführung von Aktionen in der Datenbank erlaubt ist
  2. Durchführung von Tests inkl. Randfalltests
8. Select-Statements: Für die Anzeige von Nutzerstatistiken wurden beispielhaft einige SELECT-Statements definiert.
9. Metadaten: Um die Datenbank zu dokumentieren, wurden Statements für die Ermittlung von Metadaten definiert. Bspw. die Anzahl der Tabellen, die Anzahl der Datensätze pro Tabelle, die Größe der Datenbank und die Erzeugung eines Datenwörterbuches.