

<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

Mit Hilfe von MySQL wurde eine Datenbank für eine Buchaustausch-App entwickelt. Die Datenbank heißt buchange. Die App dient dem Zweck, Personen einer lokalen Gemeinschaft, die viele Bücher besitzen, aber nicht wegschmeißen wollen, den Tausch von Büchern zu ermöglichen. Eigene Bücher können zur Ausleihe angeboten werden und Bücher von anderen Anwendern ausgeliehen werden. Die Datenbank bildet dabei das zentrale Element zur strukturierten Speicherung und Verwaltung aller relevanten Informationen.

Zur Verwaltung der Daten sollte ein Datenbankmanagementsystem (DBMS) verwendet werden. Dies birgt folgende Vorteile:

- Effiziente Speicherung der Daten in einer konsistenten und übersichtlichen Form sowie Vermeidung von Datenredundanzen
- Dies ermöglicht die effiziente und einfache Abfrage von Informationen
- Beziehungen zwischen den Daten können modelliert werden
- Sicherstellung der Datenintegrität und Datenkonsistenz durch ausgewählte Datentypen, Bedingungen und Primär- und Fremdschlüsseldefinitionen
- Mehrbenutzerfähigkeit, sodass mehrere Anwender gleichzeitig arbeiten können

Das hier verwendete DBMS ist MySQL, ein relationales DBMS, das sich durch Zuverlässigkeit, Skalierbarkeit und eine breite Unterstützung von SQL-Abfragen auszeichnet. MySQL wurde ausgewählt, da es sich um eine kostenlose Open Source-Software mit einer großen Community handelt. Da die App nur für eine lokale Gemeinschaft bereitgestellt wird, ist nicht mit extrem großen Datenmengen zu rechnen. Es müssen darüber hinaus nur einfache analytische Auswertungen gemacht werden, um persönliche Anwendungsstatistiken aufzurufen. Hierfür reichen die Funktionalitäten von MySQL aus. Auch die Funktionalitäten, die aus fachlicher Sicht für die Entwicklung der Datenbank erforderlich sind, werden durch MySQL bereitgestellt, bspw. die Implementierung von Beziehungen über Fremdschlüssel und die Sicherstellung der referentiellen Integrität über Constraints und Trigger. Die große Community stellt sicher, dass bei Fragen Unterstützung verfügbar ist. Kombiniert mit der weiten Verbreitung der Software ist sichergestellt, dass auch andere Entwickler die Weiterentwicklung der Datenbank vornehmen können.

Die Datenbank besteht aus den unten genannten Tabellen, die mit Hilfe von Fremdschlüsseln miteinander in Beziehung gesetzt wurden. Mit Hilfe der Tabellen lassen sich die Daten, die in den zentralen Funktionen und Prozessen der App anfallen, strukturiert speichern. Für die Anzeige des aktuellen Standorts des Anwenders wurde keine Tabelle erstellt. Die Standortdaten des Anwenders wechseln ständig, sodass eine Speicherung der Informationen keinen Mehrwert bringt.

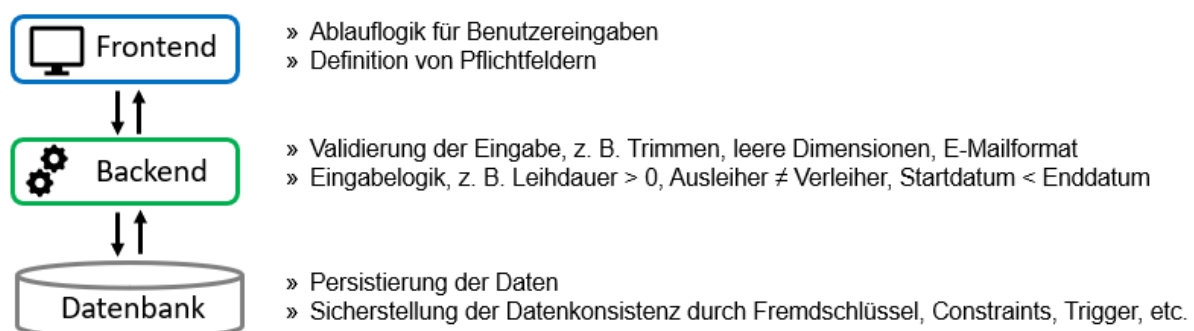
Tabellen	Zweck
Anwender	<ul style="list-style-type: none"><li>▪ Speichert Stammdaten zu registrierten Anwendern</li><li>▪ Wird gefüllt, wenn sich ein Anwender neu registriert</li></ul>
Adresse	<ul style="list-style-type: none"><li>▪ Speichert Adress-Stammdaten der Anwender</li><li>▪ Wird im Zuge der Registrierung durch den Anwender befüllt</li></ul>

Abholinfo	<ul style="list-style-type: none"> <li>▪ Speichert Stammdaten zu Abholinformationen je Anwender</li> <li>▪ Wird im Zuge der Registrierung durch den Anwender befüllt</li> </ul>
Autor	<ul style="list-style-type: none"> <li>▪ Speichert Autoren-Stammdaten</li> <li>▪ Wird gefüllt, wenn Buch erfasst wird, deren Autor noch nicht bekannt ist</li> </ul>
Buch	<ul style="list-style-type: none"> <li>▪ Speichert Stammdaten zu Büchern, die in der App ge- und verliehen werden</li> <li>▪ Wird gefüllt, wenn ein Anwender ein neues Buch zur Ausleihe anbietet</li> </ul>
Anfrage	<ul style="list-style-type: none"> <li>▪ Speichert eine Anfrage eines Anwenders für ein Buch</li> <li>▪ Wird gefüllt, sobald ein Anwender eine Anfrage für ein Buch stellt</li> <li>▪ Zeigt den Status der Anfrage: Wurde die Ausleihe bestätigt?</li> </ul>
Ausleihe	<ul style="list-style-type: none"> <li>▪ Speichert Informationen zu jedem Ausleihvorgang</li> <li>▪ Wird befüllt, sobald eine Anfrage bestätigt wurde</li> <li>▪ Zeigt den Status der Ausleihe an, Aktualisierung nach Rückgabe</li> </ul>
Rueckgabe	<ul style="list-style-type: none"> <li>▪ Speichert, wenn die Rückgabe eines Buches stattgefunden hat</li> <li>▪ Sowohl seitens des Ausleihenden als auch des Verleihers muss die Bestätigung zur Rückgabe vorliegen</li> </ul>
BuchBew	<ul style="list-style-type: none"> <li>▪ Speichert Bewertungen zu Büchern</li> <li>▪ Eine Bewertung zu einem Buch kann optional durch die Anwender erfolgen</li> </ul>
AnwBew	<ul style="list-style-type: none"> <li>▪ Speichert Bewertungen zu Anwendern.</li> <li>▪ Eine Anwender-Bewertung kann optional durch andere Anwender erfolgen</li> </ul>

Auf Basis der Tabellen wurden zusätzlich Views erstellt, die eine dynamische Ermittlung von Daten auf Basis der aktuell in den Tabellen gespeicherten Informationen, ermöglichen:

View	Zweck
ViewVerfuegbare Buecher	<ul style="list-style-type: none"> <li>▪ Ermittelt, welche Bücher nicht verliehen und Besitzer nicht gesperrt sind</li> <li>▪ Konsolidiert alle relevanten Informationen zu der Ausleihe aus verschiedenen Tabellen und ermöglicht die Anzeige auf einer Karte</li> </ul>
ViewFristueber	<ul style="list-style-type: none"> <li>▪ Ermittelt die Anzahl der Fristüberschreitungen</li> <li>▪ Ebenso wird ermittelt, ob der Anwender aktuell gesperrt ist, weil er ein fälliges Buch noch nicht zurückgegeben hat.</li> </ul>
ViewSperre	<ul style="list-style-type: none"> <li>▪ Ermittelt eine Sperre aufgrund einer aktuellen Fristüberschreitung oder der maximal erlaubten Fristüberschreitungen</li> <li>▪ Konsolidiert beide möglichen Sperren in einem weiteren Sperre-Attribut, damit eine einfache Auswertung ermöglicht wird</li> </ul>

Es wurde ein Architekturbild entwickelt, dass die Verantwortlichkeit der Datenbank in Abgrenzung zu Frontend und Backend verdeutlicht:



Zur Sicherstellung der Datenqualität auf Ebene der Datenbank wurden neben der Nutzung von klar definierten Datentypen, Fremdschlüsseln und Constraints folgende Trigger erstellt:

Trigger	Zweck
CheckAusleihe	<ul style="list-style-type: none"> <li>Prüft, ob in der Tabelle Ausleihe der Status „AusleiheAktiv“ FALSE beträgt, bevor ein Eintrag in der Tabelle AnwBew erlaubt wird</li> </ul>
CheckSperr	<ul style="list-style-type: none"> <li>Prüft, ob in der View Anwenderstatus die Sperr FALSE ist, bevor eine Anfrage zur Ausleihe erlaubt wird</li> </ul>

Die Definition von Transaktionen ermöglicht darüber hinaus die Sicherung der Datenkonsistenz, da alle SQL-Statements, die von einer Anwenderaktivität abhängen, gebündelt ausgeführt werden. Es wurde das Isolationslevel „Read Committed“ gewählt, um Overhead und Sperrkonflikte zu verringern und somit die Anwendung der App flüssiger zu gestalten.

Mit Hilfe verschiedener Testfälle wurden die Entwicklungen erfolgreich getestet. Ebenso wurde mit Randfalltests erfolgreich geprüft, dass die Maßnahmen zur Sicherung der Datenkonsistenz greifen.

Bei Bedarf können zusätzlich Select-Statements auf Basis der Tabellen erstellt werden, um weitere Analysen durchzuführen. Dies ist beispielweise für die Anzeige von Nutzungsstatistiken relevant.

Da die Performance der App wesentlich für die Anwenderakzeptanz ist, wurde darüber hinaus eine Indexierung von Fremdschlüsseln eingesetzt. Um zu messen, ob dies zu Verbesserungen führt, wurden für eine ausgewählte Tabelle Massendaten eingespielt. Mit Hilfe der in der Workbench integrierten Performance-Reports konnte gezeigt werden, dass insbesondere bei vielen Daten eine schnellere Durchführung von SELECT-Statements möglich ist. Dies ist relevant, da im Verlauf der Jahre mit wachsenden Datenmengen gerechnet werden kann.

Die auf diese Weise modellierte Datenbank besitzt folgende Metadaten:

- Die Anzahl der Tabellen beträgt 10, die Anzahl der Views beträgt 3.
- Je Tabelle existieren folgende Einträge:

	tabelle	zeilen
►	Anwender	100012
	Adresse	11
	Abholinfo	11
	Autor	12
	Buch	12
	Anfrage	1
	Ausleihe	17
	Rueckgabe	12
	BuchBew	11
	AnwBew	11

- Die Datenbank hat ein Volumen in Höhe von 8,95 Megabyte.