THEMENWOCHE 1 · LERNFELD 8

SQL und Datenbankaufbau

Befehlsgruppen verstehen, das Modell umsetzen und mit Daten arbeiten

Aufgabenpaket 4

Name	Klasse	Lerngruppe

Arbeitsorganisation

In Bereich 4.1 lernst du die SQL-Befehlsgruppen kennen und setzt dein Modell als Datenbank um. In Bereich 4.2 arbeitest du mit den Daten und prüfst die Ergebnisse. Nutze die 3NF und das Modell aus den Aufgabenpaketen 1 und 2 sowie deine lokale Umgebung aus Aufgabenpaket 3.

Arbeite vorrangig in der Konsole mit MySQL oder MariaDB. Verwende dein Projektmodell und die Datenbank dbverkauf. Baue sie schrittweise mit drei SQL-Skripten auf: 01_struktur.sql, 02_integritaet.sql und 03_testdaten.sql. Ab A4.2.2 führst du SQL-Anweisungen direkt in der Konsole aus und protokollierst sie samt Ergebnissen mit tee. Beginne jede Aufgabe mit der angegebenen Stelle im Nachschlagewerk.

Materialien

- LF8_TW1_1_Nachschlagewerk_Datenbanken_SQL.docx – besonders Abschnitte 4 und 5; konkrete Verweise stehen bei den Einzelaufgaben.
- BBK_TW_1_LF8.xlsx sowie Ergebnisse der Aufgabenpakete 1, 2 und 3.
- Für A4.2.5: die drei selbst erstellten SQL-Skripte, das relationale Tabellenmodell und die Soll-Tabelle einer anderen Gruppe.

Nachschlagwissen für Aufgabenpaket 4

- DDL verändert Strukturen, DML Datensätze; DQL liest Daten, DCL steuert Rechte und TCL Transaktionen.
- Datentypen und Schlüssel müssen zum Modell passen. Fremdschlüssel sichern die Beziehungen zwischen Tabellen.
- Prüfe vor UPDATE und DELETE mit SELECT und derselben WHERE-Bedingung, welche Zeilen betroffen sind.
- SQL-Muster sind Beispiele: Passe Tabellen- und Spaltennamen an dein Modell an und prüfe die Ergebnisse.

Abgabe und ePortfolio

Jede Person dokumentiert Aufgaben, Lösungen, den Lösungsweg und Herausforderungen im eigenen ePortfolio – auch bei Gruppenarbeit. Lege den Abschnitt „Aufgabenpaket 4 SQL und Datenbankaufbau“ an. Die Hinweise bei den Aufgaben sind Anregungen: Wähle geeignete Belege und ergänze eigene Ideen. Den Dokumentationsstandard findest du in Abschnitt 7 des Nachschlagewerks, die Nutzung von tee in Abschnitt 6.8.

4.1 · SQL verstehen und die Datenbank aufbauen

A4.1.1 · SQL Befehlsaufbau und Befehlsgruppen

Status	Sozialform
Pflichtaufgabe	Partnerarbeit

Ziel: Du erkennst, worauf sich ein SQL-Befehl auswirkt und wie eine Anweisung aufgebaut ist.

Nachschlagewerk: Abschnitt 4 „Grundlagen der Sprache SQL“, Tabelle der Befehlsgruppen und „Grundmuster für Datenbank und Tabelle“ (S. 5). Nutze das CREATE-Beispiel zum Untersuchen des Anweisungsaufbaus.

Arbeitsauftrag

1. Lies das folgende Nachschlagwissen. Erkläre in eigenen Worten den Unterschied zwischen Datenbankstruktur, gespeicherten Datensätzen und Zugriffsrechten.
2. Erstelle eine Übersicht mit DDL, DML, DQL, DCL und TCL. Ordne CREATE, ALTER, DROP, INSERT, UPDATE, DELETE, SELECT, GRANT, REVOKE, COMMIT und ROLLBACK zu. Ergänze Zweck und ein eigenes Anwendungsbeispiel je Gruppe.
3. Kennzeichne im untenstehenden CREATE-Beispiel Schlüsselwörter, Objektname, Definition in Klammern und Anweisungsende. Erläutere, warum Zeilenumbrüche und Einrückungen beim Lesen helfen. Führe das Beispiel noch nicht aus.

Arbeitsergebnis

- Befehlsübersicht und kommentiertes Syntaxbeispiel

Gestufte Lösungshilfen

Hilfe 1 · DDL verändert Strukturen; DML verändert Datensätze; DQL liest Daten; DCL steuert Rechte; TCL steuert Transaktionen. SELECT wird in diesem Unterricht als DQL geführt.

Hilfe 2 · DDL: CREATE erstellt, ALTER verändert und DROP entfernt Strukturen. DML: INSERT fügt Zeilen ein, UPDATE ändert und DELETE löscht Zeilen. DCL: GRANT vergibt und REVOKE entzieht Rechte. TCL: COMMIT bestätigt und ROLLBACK verwirft Änderungen einer laufenden Transaktion. DCL lernst du praktisch in Aufgabenpaket 5 kennen.

Hilfe 3 · SQL-Anweisungen enthalten Schlüsselwörter und Objektnamen. Klammern gruppieren Definitionen, Kommas trennen Elemente und ein Semikolon beendet die Anweisung in der Konsole.

```
CREATE TABLE beispiel (
  id INT PRIMARY KEY,
  bezeichnung VARCHAR(80) NOT NULL
);
```

Hilfe 4 · SHOW zeigt Informationen, USE wählt eine Datenbank und DESCRIBE beschreibt eine Tabelle. Diese Hilfsbefehle verändern keine gespeicherten Datensätze.

Selbstkontrolle

- Jeder genannte SQL-Befehl ist einer Unterrichtskategorie zugeordnet.
- Du kannst unterscheiden, ob ein Befehl eine Struktur, einen Datensatz oder eine Berechtigung betrifft.

Was sollte in das ePortfolio übernommen werden

- Du könntest deine Befehlsübersicht und ein selbst kommentiertes Syntaxbeispiel aufnehmen.
- Ergänze bei Bedarf eine Merkhilfe zu Befehlsgruppen, die du zunächst verwechselt hast.

A4.1.2 · SQL-Skript für Datenbank und Tabellen erstellen

Status	Sozialform
Pflichtaufgabe	Partnerarbeit

Ziel: Du erstellst aus deinem Modell und den Daten in dritter Normalform ein SQL-Skript zum Anlegen der Datenbank und ihrer Tabellen mit Primärschlüsseln.

Nachschlagewerk: Abschnitt 4 „Grundmuster für Datenbank und Tabelle“ (S. 5) für CREATE und USE. Abschnitt 5, Entscheidungstabelle zu Datentypen, Zeichensatz und Fremdschlüsseln (S. 6), hilft bei den Attributdefinitionen.

Arbeitsauftrag

1. Vergleiche die bereinigte Arbeitsdatei aus Aufgabenpaket 1 mit dem relationalen Tabellenmodell aus Aufgabenpaket 2. Übernimm alle Tabellen, Attribute und Primärschlüssel. Lege auch die Spalten für spätere Fremdschlüssel an; die Fremdschlüsselregeln folgen in A4.1.4.
2. Bestimme für jedes Attribut Datentyp, Länge beziehungsweise Wertebereich, NULL-Regel und gegebenenfalls einen Standardwert. Begründe die Entscheidungen für Geldbeträge, Namen und Postleitzahlen.
3. Erstelle in einem Texteditor die Datei 01_struktur.sql mit CREATE DATABASE, USE und CREATE TABLE. Verwende utf8mb4, InnoDB, Primärschlüssel und die festgelegten Attributdefinitionen. Ordne referenzierte Tabellen vor abhängigen Tabellen an. Speichere als UTF-8 mit der Endung .sql; führe die Datei erst in A4.1.3 aus.

Arbeitsergebnis

- 01_struktur.sql mit Datenbank, Tabellen und Primärschlüsseln; SQL-Kommentare begründen ausgewählte Entscheidungen und benennen die späteren Fremdschlüssel.

Gestufte Lösungshilfen

Hilfe 1 · Eine SQL-Datei enthält ausführbare Anweisungen und SQL-Kommentare, keine Konsolenausgaben. Schreibe Kommentare beispielsweise als -- Begründung (mit Leerzeichen nach den beiden Bindestrichen). Speichere die Datei nicht versehentlich als .sql.txt.

Hilfe 2 · Verwende INT für geeignete ganzzahlige IDs, VARCHAR für unterschiedlich lange Texte und DECIMAL mit ausreichender Genauigkeit für Geld. Postleitzahlen sind Textwerte; führende Nullen müssen erhalten bleiben.

Hilfe 3 · Die Spalten späterer Fremdschlüssel und die zugehörigen Primärschlüssel benötigen kompatible Datentypen. AUTO_INCREMENT erzeugt technische IDs; prüfe die fachliche Eindeutigkeit trotzdem.

Hilfe 4 · Grundmuster für dein Skript; ergänze alle Tabellen deines eigenen Modells:

```
CREATE DATABASE dbverkauf CHARACTER SET utf8mb4;
USE dbverkauf;
CREATE TABLE artikel (
  artikel_id INT PRIMARY KEY AUTO_INCREMENT,
  bezeichnung VARCHAR(80) NOT NULL,
  preis DECIMAL(10,2) NOT NULL
) ENGINE=InnoDB;
```

Selbstkontrolle

- Alle Tabellen und Attribute aus dem relationalen Tabellenmodell sind im SQL-Skript berücksichtigt.
- Primärschlüssel sind definiert. Spalten für spätere Fremdschlüssel sind vorhanden; die Fremdschlüsselregeln werden erst in A4.1.4 ergänzt.

Was sollte in das ePortfolio übernommen werden

- Als Beleg eignen sich Ausschnitte aus 01_struktur.sql mit deinen SQL-Kommentaren.
- An ausgewählten Attributen könntest du zeigen, wie du Datentypen oder Schlüssel begründet hast.

A4.1.3 · Datenbank und Tabellen anlegen

Status	Sozialform
Pflichtaufgabe	Partnerarbeit

Ziel: Du führst dein SQL-Skript in der Konsole aus und prüfst, ob die Datenbank und ihre Tabellen wie geplant angelegt wurden.

Nachschlagewerk: Abschnitt 4 „Grundmuster für Datenbank und Tabelle“ (S. 5) für CREATE und USE sowie Abschnitt 5 „Kontrollbefehle“ (S. 6) für die Strukturprüfung.

Arbeitsauftrag

1. Prüfe, ob dbverkauf bereits existiert. Verwende eine neue lokale Übungsdatenbank und überschreibe keine vorhandenen Arbeitsergebnisse. Falls nötig, wähle einen anderen Namen und verwende ihn konsequent.
2. Prüfe die Datei 01_struktur.sql aus A4.1.2: Datenbankname, Zeichensatz, Tabellen, Datentypen und Primärschlüssel müssen zu deinem Modell passen. Fremdschlüsselregeln folgen erst in A4.1.4.
3. Verbinde dich wie in Aufgabenpaket 3 über den MySQL-/MariaDB-Konsolenclient mit deinem lokalen Server. Führe 01_struktur.sql mit dem Clientbefehl source aus (siehe Hilfe 1). Prüfe die Rückmeldungen, bevor du weiterarbeitest.
4. Prüfe die Tabellen mit SHOW TABLES, DESCRIBE und SHOW CREATE TABLE. Vergleiche die gespeicherte Struktur mit deinem SQL-Skript und dem relationalen Tabellenmodell.

Arbeitsergebnis

- Ausgeführte und gegebenenfalls korrigierte 01_struktur.sql; Datenbank und Tabellen mit Primärschlüsseln sind angelegt.
- Strukturprotokoll mit SHOW CREATE TABLE für jede Tabelle

Gestufte Lösungshilfen

Hilfe 1 · Gib nach der Anmeldung im MySQL-/MariaDB-Konsolenclient folgende Zeile ein. Passe den Pfad an den Speicherort deiner Datei an; verwende möglichst einen Ordner ohne Leerzeichen:

```
source C:/LF8/01_struktur.sql
```

Hilfe 2 · source liest die lokale Datei und führt ihre SQL-Anweisungen aus. Es ist ein Clientbefehl, kein SQL-Befehl für den Server und kein Aufruf in PowerShell. Prüfe danach zum Beispiel:

```
USE dbverkauf;
SHOW TABLES;
DESCRIBE artikel;
SHOW CREATE TABLE artikel;
```

Hilfe 3 · Bei einem Fehler können vorherige Anweisungen bereits ausgeführt sein. Prüfe den vorhandenen Stand und korrigiere die Ursache, bevor du fortfährst. Zum erneuten Gesamtaufbau nutze eine Kopie des Skripts mit einem neuen Testdatenbanknamen in CREATE DATABASE und USE.

Selbstkontrolle

- Alle Tabellen sind angelegt; Datentypen und Primärschlüssel entsprechen deinem relationalen Tabellenmodell.
- Du kannst die Datenbankstruktur anhand deiner Datei in einer leeren Umgebung erneut aufbauen.

Was sollte in das ePortfolio übernommen werden

- Du könntest die SQL-Datei und ausgewählte Strukturabfragen mit ihren Ergebnissen aufnehmen.
- Beschreibe bei Bedarf eine Entscheidung oder Herausforderung beim Anlegen der Tabellen.

A4.1.4 · Fremdschlüssel und Integritätsregeln ergänzen

Status	Sozialform
Pflichtaufgabe	Partnerarbeit

Ziel: Du setzt die Beziehungen aus dem Modell als geprüfte Integritätsregeln um.

Nachschlagewerk: Abschnitt 5 „Fremdschlüssel definieren“ einschließlich der Tabelle zu RESTRICT, CASCADE und SET NULL (S. 6). Ein CASCADE-Beispiel bietet Z1.3 „Umsetzung als relationales Schema“ (S. 15).

Arbeitsauftrag

1. Erstelle eine separate Datei 02_integritaet.sql. Beginne mit USE dbverkauf; und ergänze die Fremdschlüssel mit ALTER TABLE. Vergib verständliche Namen für die Regeln, z. B. fk_bestellung_kunde für den Fremdschlüssel von bestellung zu kunde. Prüfe vorher die referenzierten Tabellen und Schlüssel.
2. Lege für jede Beziehung das Verhalten bei Änderungen und Löschungen fest: RESTRICT, CASCADE oder SET NULL. Ergänze diese Integritätsregeln im Skript und begründe deine Wahl fachlich.
3. Führe 02_integritaet.sql mit source aus und kontrolliere die gespeicherten Regeln mit SHOW CREATE TABLE. Notiere für jede Regel einen erlaubten und einen unzulässigen Testfall mit der erwarteten Reaktion des Servers.

Arbeitsergebnis

- 02_integritaet.sql mit ausgeführten Fremdschlüsseldefinitionen und eine Übersicht der Integritätsregeln
- Entscheidungstabelle zu ON UPDATE und ON DELETE mit geplanten Testfällen

Gestufte Lösungshilfen

Hilfe 1 · Mit ALTER TABLE ergänzt du eine vorhandene Tabelle. Nach USE dbverkauf; kann eine Regel mit Musternamen so aussehen:

```
ALTER TABLE kindtabelle
  ADD CONSTRAINT fk_kind_eltern
  FOREIGN KEY (eltern_id)
  REFERENCES elterntabelle (eltern_id)
  ON UPDATE CASCADE
  ON DELETE RESTRICT;
```

Hilfe 2 · RESTRICT verhindert das Löschen einer Elternzeile oder das Ändern ihres referenzierten Schlüsselwerts, solange Kindzeilen darauf verweisen. Andere Attribute der Elternzeile sind davon nicht betroffen. CASCADE überträgt Schlüsseländerung oder Löschung auf Kindzeilen. SET NULL setzt den Fremdschlüssel auf NULL und benötigt eine NULL-fähige Spalte.

Hilfe 3 · Führe das Skript mit source C:/LF8/02_integritaet.sql aus (Pfad anpassen). CASCADE darf keine fachlich wichtigen Daten unbeabsichtigt entfernen. Deaktiviere die Fremdschlüsselprüfung nicht, um Fehler zu umgehen. Der Regelname hilft, eine Fehlermeldung zuzuordnen.

Selbstkontrolle

- Jede Beziehung des Modells ist umgesetzt und in SHOW CREATE TABLE sichtbar.
- Die gewählten Aktionen passen zur fachlichen Bedeutung der Beziehung.

Was sollte in das ePortfolio übernommen werden

- Eine Möglichkeit ist deine Entscheidungstabelle zu Fremdschlüsseln und den geplanten Testfällen.
- An einer ausgewählten Beziehung könntest du erläutern, warum deine Lösch- und Änderungsregeln fachlich passen.

4.2 · Mit Daten arbeiten und Ergebnisse prüfen

A4.2.1 · Eigene Testdaten einfügen

Status	Sozialform
Pflichtaufgabe	Partnerarbeit

Ziel: Du erstellst ein SQL-Skript mit Testdaten, führst es aus und prüfst die eingefügten Datensätze.

Nachschlagewerk: Abschnitt 4 „Grundmuster für Datenänderungen und Auswahl“, zunächst nur das INSERT-Beispiel (S. 5). Beachte die Einleitung von Abschnitt 5 zur Reihenfolge abhängiger Daten (S. 6).

Arbeitsauftrag

1. Plane zuerst benötigte Stammdaten und danach mindestens fünf eigene Artikel mit unterschiedlichen Bezeichnungen und Preisen. Berücksichtige alle weiteren Pflichtattribute und Fremdschlüssel deines Modells.
2. Erstelle 03_testdaten.sql mit USE dbverkauf; und INSERT-Anweisungen mit ausdrücklichen Spaltenlisten. Ordne referenzierte Datensätze vor abhängigen Datensätzen an. Kennzeichne deine Testdaten durch eindeutige IDs beziehungsweise erkennbare Testbezeichnungen.
3. Führe 03_testdaten.sql mit source aus (siehe Hilfe 3) und prüfe die Rückmeldungen. Halte die fünf Artikel zusätzlich als Soll-Tabelle fest und vergleiche sie mit SELECT * FROM artikel; mit den gespeicherten Werten. Sie sind die Grundlage der nächsten Aufgaben.

Arbeitsergebnis

- 03_testdaten.sql und Soll-Tabelle mit mindestens fünf Testartikeln sowie benötigten Stammdaten
- Ausführungsprotokoll der INSERT-Anweisungen

Gestufte Lösungshilfen

Hilfe 1 · INSERT benötigt eine vorhandene Tabelle. Spaltenliste und Werteliste müssen zueinander passen. Beispiel:

```
INSERT INTO artikel (bezeichnung, preis)
VALUES ('Testartikel A', 19.90),
       ('Testartikel B', 24.50);
```

Hilfe 2 · Texte stehen in einfachen Anführungszeichen; Dezimalzahlen verwenden den Punkt. Ein automatisch erzeugter Primärschlüssel kann aus der Spaltenliste entfallen. Bei manuell vergebenen IDs müssen alle Schlüssel eindeutig sein.

Hilfe 3 · Führe die Datei mit source C:/LF8/03_testdaten.sql aus (Pfad anpassen). Ein Kinddatensatz benötigt zuerst den passenden Elternsatz. Bewahre die IDs deiner Testdaten auf. Wiederhole INSERT nicht ungeprüft: Bereits eingefügte Zeilen können doppelt entstehen oder Schlüsselfehler auslösen.

Selbstkontrolle

- Alle INSERT-Anweisungen wurden ausgeführt und die Rückmeldungen sind nachvollziehbar.
- Die Soll-Tabelle enthält die tatsächlich eingegebenen Werte.

Was sollte in das ePortfolio übernommen werden

- Du könntest deine INSERT-Datei mit der Soll-Tabelle und passenden Ausführungsergebnissen verbinden.
- Ergänze bei Bedarf, wie du geeignete Testdaten ausgewählt und ihre Abhängigkeiten berücksichtigt hast.

A4.2.2 · Daten mit SELECT gezielt auswählen

Status	Sozialform
Pflichtaufgabe	Einzelarbeit

Ziel: Du beantwortest konkrete Fragen zu den zuvor eingefügten Daten.

Nachschlagewerk: Abschnitt 4, SELECT mit WHERE und ORDER BY (S. 5), sowie Abschnitt 6.8 „Konsolensitzung mit tee protokollieren“ (S. 11). Muster für LIKE, COUNT und AVG stehen in den Lösungshilfen dieser Aufgabe.

Arbeitsauftrag

1. Starte im verbundenen MySQL-/MariaDB-Konsolenclient das Protokoll mit tee C:/LF8/A4_2_2.txt (Pfad anpassen). Formuliere mindestens sechs Fragen an deine Testdaten. Führe zu jeder Frage eine SELECT-Abfrage direkt in der Konsole aus.
2. Zeige alle Spalten und anschließend nur ausgewählte Spalten. Filtere nach Preisgrenzen und nach Textwerten.
3. Sortiere mindestens eine Ausgabe aufsteigend und eine absteigend. Verwende mindestens einmal COUNT oder AVG mit einem geeigneten Alias.
4. Vergleiche jede Ausgabe mit deiner Soll-Tabelle. Kommentiere zwei Ergebnisausschnitte ausführlicher. Beende das Protokoll mit notee. Arbeite auch in den folgenden Aufgaben direkt in der Konsole und nutze je Aufgabe eine eigene tee-Protokolldatei.

Arbeitsergebnis

- Mindestens sechs SQL-Abfragen mit Fragen und Ergebnissen
- Zwei kommentierte Ergebnisausschnitte aus dem tee-Protokoll

Gestufte Lösungshilfen

Hilfe 1 · Grundaufbau: SELECT bestimmt die Ausgabespalten, FROM die Tabelle, WHERE die Auswahl und ORDER BY die Sortierung. WHERE und ORDER BY sind optional.

```
SELECT bezeichnung, preis
FROM artikel
WHERE preis >= 20.00
ORDER BY preis ASC;
```

Hilfe 2 · Weitere Muster zur Anpassung an deine eigenen Fragen:

```
SELECT * FROM artikel;
SELECT bezeichnung FROM artikel
WHERE bezeichnung LIKE 'Test%' ORDER BY bezeichnung DESC;
SELECT COUNT(*) AS anzahl FROM artikel;
SELECT AVG(preis) AS durchschnittspreis FROM artikel;
```

Hilfe 3 · ASC sortiert aufsteigend, DESC absteigend. LIKE prüft ein Textmuster; % steht dabei für beliebig viele Zeichen. AS vergibt eine lesbare Überschrift im Ergebnis.

Selbstkontrolle

- Jede Abfrage beantwortet die formulierte Frage.
- Preis- und Textfilter, beide Sortierrichtungen sowie eine Aggregation sind enthalten.
- Die Ergebnisse stimmen mit deinen Testdaten überein.

Was sollte in das ePortfolio übernommen werden

- Als Belege eignen sich eigene Fragen mit den zugehörigen SELECT-Anweisungen und Ergebnissen.
- Du könntest an ausgewählten Ausgaben zeigen, wie du die Ergebnisse geprüft und interpretiert hast.

A4.2.3 · Daten kontrolliert ändern und löschen

Status	Sozialform
Pflichtaufgabe	Partnerarbeit

Ziel: Du setzt UPDATE und DELETE gezielt ein und kannst deren Wirkung belegen.

Nachschlagewerk: Abschnitt 4 „Grundmuster für Datenänderungen und Auswahl“, UPDATE-Beispiel und anschließende Sicherheitsregel für UPDATE und DELETE (S. 5): zuerst SELECT mit derselben WHERE-Bedingung.

Arbeitsauftrag

1. Wähle einen eigenen Testartikel und ändere genau dessen Preis mit UPDATE.
2. Wähle einen selbst angelegten Testdatensatz und entferne ihn mit DELETE. Beachte dabei die vorhandenen Fremdschlüssel.
3. Führe unmittelbar vor jedem UPDATE und DELETE eine SELECT-Abfrage mit derselben WHERE-Bedingung aus. Führe nach der Änderung dieselbe Abfrage erneut aus und erkläre das Ergebnis.
4. Aktualisiere deine Soll-Tabelle. Zeige anhand der ursprünglich fünf Artikel, welcher Datensatz geändert und welcher gelöscht wurde.

Arbeitsergebnis

- tee-Protokoll mit SQL-Anweisungen und den Ergebnissen vor und nach der Änderung beziehungsweise Löschung
- Aktualisierte Soll-Tabelle mit nachvollziehbarer Änderung und Löschung

Gestufte Lösungshilfen

Hilfe 1 · Prüfen, ändern, erneut prüfen. Ersetze die Beispiel-ID durch die ID deines Testartikels:

```
SELECT * FROM artikel WHERE artikel_id = 101;
UPDATE artikel SET preis = 21.50 WHERE artikel_id = 101;
SELECT * FROM artikel WHERE artikel_id = 101;
```

Hilfe 2 · DELETE entfernt Zeilen; die Tabelle bleibt bestehen. DROP TABLE würde hingegen die Tabelle entfernen.

```
SELECT * FROM artikel WHERE artikel_id = 102;
DELETE FROM artikel WHERE artikel_id = 102;
SELECT * FROM artikel WHERE artikel_id = 102;
```

Hilfe 3 · Ohne WHERE können UPDATE und DELETE alle Zeilen betreffen. Verwende für diese Übungen ausschließlich deine eigenen Testdatensätze.

Selbstkontrolle

- Die WHERE-Bedingung trifft genau den vorgesehenen Testdatensatz.
- Keine unbeabsichtigte Zeile wurde geändert oder gelöscht.
- Du kannst erklären, warum nicht jedes Benutzerkonto UPDATE oder DELETE ausführen dürfen sollte.

Was sollte in das ePortfolio übernommen werden

- Du könntest eine Änderung und eine Löschung mit den jeweiligen Vorher-Nachher-Ergebnissen erläutern.
- Auch deine aktualisierte Soll-Tabelle oder eine eigene Merkhilfe zum sicheren Einsatz von WHERE eignet sich.

A4.2.4 · Fehler analysieren und SQL verbessern

Status	Sozialform
Pflichtaufgabe	Teamarbeit

Ziel: Du unterscheidest Fehlermeldungen, Warnungen und die Bedienung der Konsole.

Nachschlagewerk: Abschnitt 5 „Kontrollbefehle“ (S. 6) zur Prüfung von Tabellenname und Struktur. Die konkreten Fälle zu Konsolenbedienung, SQL-Modus und Warnungen erläutern die Lösungshilfen dieser Aufgabe.

Arbeitsauftrag

1. Untersucht kontrolliert drei Fälle: einen falschen Tabellennamen, ein fehlendes Semikolon und einen ungültigen Wert für einen Datentyp. Nutzt dafür nur Testdaten.
2. Notiert jeweils Anweisung, Meldung oder beobachtetes Verhalten, Ursache und Korrektur. Ändert pro Versuch nur eine vermutete Ursache.
3. Ordnet die beabsichtigte Anweisung einer SQL-Befehlsgruppe zu. Führt sie nach der Korrektur erneut aus und dokumentiert die erfolgreiche Ausführung.

Arbeitsergebnis

- tee-Protokoll mit drei Fehlerfällen, Ursachen und korrigierten Anweisungen

Gestufte Lösungshilfen

Hilfe 1 · Bei einem falschen Tabellennamen helfen SHOW TABLES und ein Vergleich der Schreibweise. Prüft auch, ob mit USE die richtige Datenbank ausgewählt wurde.

Hilfe 2 · Ein fehlendes Semikolon erzeugt in der Konsole zunächst meist den Fortsetzungsprompt → beziehungsweise ->. Der Client wartet noch auf den Abschluss der Anweisung. Ergänzt das Semikolon; mit dem Clientkommando \c könnt ihr die noch nicht gesendete Eingabe verwerfen.

Hilfe 3 · Ein ungeeigneter Datentypwert kann je nach SQL-Modus einen Fehler oder eine Warnung auslösen. Prüft die Sitzung mit SELECT @@SESSION.sql_mode; und Warnungen direkt nach dem Test mit SHOW WARNINGS;. Ändert den SQL-Modus für diesen Test nicht.

Hilfe 4 · Wenn der Server einen Wert umgewandelt hat, kontrolliert den gespeicherten Datensatz mit SELECT und entfernt den betroffenen Testdatensatz anschließend gezielt.

Selbstkontrolle

- Ursache und Korrektur passen zur beobachteten Reaktion.
- Ein wartender Konsolenprompt wird nicht mit einer Serverfehlermeldung verwechselt.
- Die korrigierten Anweisungen funktionieren.

Was sollte in das ePortfolio übernommen werden

- Eine Möglichkeit ist dein Fehlerprotokoll mit SQL-Anweisungen, beobachtetem Verhalten und Korrekturen.
- Du könntest einen besonders lehrreichen Fall auswählen und deinen Weg zur Ursache beschreiben.

A4.2.5 · SQL-Dateien einer anderen Gruppe prüfen

Status	Sozialform
Pflichtaufgabe	Teamarbeit

Ziel: Du prüfst die SQL-Skripte einer anderen Gruppe anhand ihres relationalen Tabellenmodells und begründest deine Bewertung.

Nachschlagewerk: Entscheidungstabelle in Abschnitt 5 „Datenbank und Tabellen aus dem Modell erzeugen“ (S. 6). Ergänze aus der Prüftabelle in Abschnitt 7 die Zeilen „Struktur“, „Daten“ und „Integrität“ (S. 12).

Arbeitsauftrag

1. Tauscht die drei SQL-Skripte 01_struktur.sql, 02_integritaet.sql und 03_testdaten.sql mit einer anderen Gruppe. Nehmt deren relationales Tabellenmodell und die zum INSERT-Skript passende Soll-Tabelle aus A4.2.1 hinzu. Kennzeichnet den Datenstand eindeutig.
2. Dokumentiert mindestens fünf Prüfpunkte anhand konkreter Stellen. Kennzeichnet jeweils: Anforderung erfüllt, begründeter Mangel oder noch zu klärende Frage. Unterschiedliche, fachlich passende Entwürfe sind keine Fehler.
3. Beschreibt bei tatsächlichen Mängeln die Auswirkung und eine begründete Korrektur. Haltet bei erfüllten Anforderungen fest, woran ihr dies erkennt. Arbeitet in Kopien und überschreibt die ursprünglichen Dateien nicht.
4. Falls eine Ausführung zur Prüfung nötig ist, nutzt Kopien in einer getrennten Testdatenbank. Passt zuvor sämtliche Datenbanknamen an und prüft mögliche DROP-Anweisungen. Führt die drei vorhandenen Skripte in der Reihenfolge 01, 02, 03 aus; protokolliert Prüfungsabfragen und Ergebnisse mit tee.

Arbeitsergebnis

- Prüfprotokoll mit mindestens fünf belegten Prüfpunkten und begründeter Bewertung; bei tatsächlichen Mängeln zusätzlich Auswirkungen und Korrekturvorschläge

Gestufte Lösungshilfen

Hilfe 1 · Prüft beispielsweise Datenbanknamen, Zeichensatz, Geldfelder, Postleitzahlen, Schlüssel und Schreibweisen. Bewertet anhand des zugehörigen relationalen Tabellenmodells und der fachlichen Anforderungen; eine andere passende Lösung ist kein Fehler.

Hilfe 2 · Achtung Datenstand: In A4.2.3 wurden ein Preis geändert und ein Artikel gelöscht. Die dort aktualisierte Soll-Tabelle passt deshalb nicht mehr zum ursprünglichen 03_testdaten.sql. Vergleicht das Skript mit dem Stand aus A4.2.1; falls dieser fehlt, rekonstruiert ihn aus den INSERT-Anweisungen. Spätere Änderungen sind kein Fehler im INSERT-Skript.

Hilfe 3 · Übernehmt SQL-Dateien nicht ungeprüft. Trennt belegte Mängel von offenen Fragen und vertretbaren Entwurfsentscheidungen. Auch ein erfüllter Prüfpunkt ist ein aussagekräftiges Prüfergebnis.

Selbstkontrolle

- Mindestens fünf Prüfpunkte sind anhand konkreter Stellen und nachvollziehbarer Maßstäbe bewertet.
- Die Originaldateien sind unverändert. Soll-Tabelle und geprüftes SQL-Skript beziehen sich auf denselben Datenstand.
- Tatsächliche Mängel sind von offenen Fragen und fachlich passenden Entwurfsvarianten unterschieden; vorgeschlagene Korrekturen passen zum geprüften Modell und zur dritten Normalform.

Was sollte in das ePortfolio übernommen werden

- Du könntest ausgewählte Prüfpunkte mit den betreffenden SQL-Stellen und deiner Bewertung darstellen.
- Ergänze bei Bedarf, was du aus dem Vergleich der Entwürfe für die eigene Datenbank gelernt hast.