DATENBANKEN & SQL

– Daten einfügen, ändern und löschen –

Kurs am RRZK der Universität zu Köln

Rüdiger Voigt, M.A.

22.07. – 26.07.2019

Übersicht

DATEN EINFÜGEN

INSERT INTO

Mit **INSERT INTO** fügen Sie Daten in eine Tabelle ein:

INSERT INTO exampleTable (var01int, var02char)
VALUES (12,'abc'), (13,'def'), (18,'ghi');

Wichtig:

- (var01int, var02char) ist die Liste der Spalten, welche Sie ändern möchten. Ersetzen Sie die Bezeichnungen durch die korrekten Spaltennamen.
- Der obige Befehl fügt drei Zeilen auf einmal ein!
- In jeder Klammer stehen genau so viele Werte wie zuvor Spalten benannt wurden und in der gleichen Reihenfolge.

INSERT: verkürzte Syntax

Hinweis: Es gibt die Möglichkeit die Auflistung der Spaltennamen im Befehl einzusparen. Das ist zumindest in Stored Procedures *nicht zu empfehlen*.
Ein Grund ist, dass dieser Befehl eine Änderung der Datenbankstruktur nicht übersteht.

Daten importieren I

Kleine Datensätze können Sie in der Regel mit phpMyAdmin oder ähnliche Oberflächen importieren.
Oftmals gibt es aber (gerade bei Webanwendungen) eine Begrenzung der Dateigröße und der Laufzeit von Skripten.
Dadurch wird der Import größerer Datenmengen über eine solche Schnittstelle unterbunden bzw. erschwert.

Daten importieren II

Eigentlich jedes DBMS bringt Werkzeuge mit um Daten zu importieren. Im Falle von MariaDB ist das mysqlimport[1].
Generell gilt:

- Bei sehr großen Datenmenge sollten Sie den Vorgang in mehrere Schritte aufteilen.
- Importieren Sie zunächst einen kleineren Teil der Daten und prüfen Sie ob die Datentypen passen und der Vorgang korrekt funktioniert.
- Sie müssen die Daten in ein für den Datenbank-Prozess zugängliches Verzeichnis speichern.

[1] https://www.ruediger-voigt.eu/big-dataset-csv-import-to-relational-databases.html

DATEN ÄNDERN ODER LÖSCHEN

UPDATE / Ändern von Daten I

Die Grundsyntax lautet:
UPDATE tablename **SET** col1={expr}, col2={expr}
WHERE where_condition;

Der Befehl UPDATE gefolgt vom Namen der Tabelle, auf welchen sich der Befehl bezieht. Dann die Anweisung SET gefolgt von einer durch Komma getrennten Liste von Anweisungen der Art column=wert. Der Wert kann hierbei entweder direkt übergeben, dabei berechnet oder aber durch eine eigene Anweisung (Sub-SELECT) erst noch ermittelt werden. Das gefolgt von einer WHERE Bedingung.

Eine passend gewählte WHERE-Bedingung erlaubt es ganze Gruppen von Einträgen auf einmal zu ändern!

Gezielt Ändern oder Löschen mit einem Unique Identifier

Um Daten gezielt zu ändern brauchen wir ein eindeutiges Identifikationsmerkmal / einen unique identifier!

Problemstellung: WHERE kombiniert mit einem unique identifier erlaubt es uns *gezielt* bestimmte Einträge zu löschen, oder zu ändern!

INSERT ...SELECT

Mittels des Konstrukts **INSERT ...SELECT** können Sie die gesamten Rückgabewerte einer SELECT Abfrage in eine Tabelle schreiben!

Legen Sie dazu eine Tabelle mit entsprechenden Feldern und passenden Data Types an. Die Syntax für das Speichern ist einfach:
INSERT INTO targetTab **SELECT** ‘exampleCol‘ **FROM** exampleTab;

Hinweis: bei einigen DBMS ist dies als **SELECT** ... **INTO** ... Befehl implementiert.

INSERT . . . SELECT: Verwendung

Dieses Konstrukt ist sehr hilfreich, wenn Sie eine Datenbank neu strukturieren möchten. Mit den richtigen Abfragen können Sie eine neue Tabellenstruktur erstellen und die vorhanden Tabellen dann löschen.

Falls die Struktur der Datenbanken jedoch in Ordnung ist und Sie dauerhaft eine bestimmte Zusammenstellung der Daten benötigen, verwenden Sie stattdessen eine View (behandeln wir im nächsten Themenblock). Ein View aktualisiert sich ständig, mit INSERT . . . SELECT erzeugen Sie eine Kopie Ihrer Daten in einem bestimmten Zustand.

DELETE FROM

Das Löschen von Zeilen funktioniert im Wesentlichen Analog zum Ändern von Inhalten:

- **Eine fehlende WHERE Condition bedeutet Sie löschen den gesamten Inhalt der Tabelle.**
- Mittels WHERE können Sie genau steuern was Sie löschen.

Der Grundbefehl lautet:
DELETE FROM table_name **WHERE** exampleColumn=some_value;

DROP ...

Hinweis

Mit DROP Statements löschen Sie komplette Datenbanken oder auch komplette Tabellen:
DROP DATABASE dbName;
DROP TABLE tableName;

Dafür müssen Sie dafür entsprechende Privileges haben!

Wenn Sie nur die Inhalte einer Tabelle löschen möchten, schauen Sie sich TRUNCATE TABLE an.

Falle fehlende oder immer wahre WHERE Bedingungen

Beachten Sie:

Eine fehlende WHERE Bedingung bewirkt, dass der UPDATE Befehl auf *alle* Zeilen der benannten Tabelle angewandt wird!

PhpMyAdmin hängt meist ein . . . WHERE 1 an generierte Queries. Diese Bedingung ist *immer* erfüllt und matcht somit genau so alle Zeilen.

SELECT vor einer Massenänderung

WARNUNG / Best Practice

In komplexe WHERE Bedingungen schleichen sich gelegentlich Fehler ein, die bewirken, dass diese Bedingung in unerwünschten Fällen oder gar immer wahr ist!

Bevor Sie ein UPDATE oder ein DELETE-Statement mit einer komplexen WHERE Bedingung ausführen, hängen Sie die WHERE Bedingung an ein einfaches SELECT-Statement und prüfen Sie, ob die gewünschten Daten zurückgeliefert werden.

Keine Rückfrage

WARNUNG

Wenn Sie in phpMyAdmin über das GUI einen DROP-, oder auch nur einen DELETE-Befehl auslösen, fragt Sie phpMyAdmin, ob Sie dies auch wirklich tun möchten.

Das ist eine Funktion von phpMyAdmin und nicht von SQL. **Der gleiche Befehl wird auf der Kommandozeile *augenblicklich* ausgeführt.**

Weiter mit dem Foliensatz “Übersicht gewinnen / komplexere Abfragen”.

Den kompletten Foliensatz finden Sie in aktueller Version unter:

`https://www.ruediger-voigt.eu/kurs-datenbanken-und-sql.html`