Funktionen und Trigger in MariaDB/MySQL

Funktionen in MariaDB/MySQL sind eine spezielle Art von (ab-)gespeichertem Programm, also ähnlich wie die Stored Procedures. . Der Unterschied zu letzteren besteht darin, dass SPs *nicht* explizit einen Rückgabewert liefern können, sondern nur über den „Umweg“ der OUTPUT-Parameter.  
Funktionen dagegen können einen expliziten RETURN-Wert liefern und können somit auch z.B. in weitere SELECT-Abfragen oder SPs eingebaut werden.

Delimiter

Wie man schon in den SPs gesehen hat, muss bei den Funktionen auf analoge Art und Weise der Default-Delimiter „;“ verändert werden, damit diverse SQL-Clients die Funktion nicht beim ersten Vorkommen eines „;“ abbrechen.  
Siehe dazu die Vorgehensweise im Kapitel zu den Stored Procedures. Dabei nicht vergessen: Nach dem Umstellen des Delimiters am Anfang muss man selbigen nach dem Funktionscode natürlich wieder auf den Standard zurücksetzen, damit die normale Funktionalität wieder hergestellt ist!

Erstellen einer Funktion in MariaDB

Zum Erstellen einer Funktion benutzt man die Syntax, ähnlich wie bei den SPs:

CREATE FUNCTION FunctionName (Parameter1, Parameter2, …)  
RETURNS Datentyp  
[NOT] DETERMINISTIC  
BEGIN

-\*SQL Statements\*-

END |

Zunächst gibt man einen Funktionsnamen und Parameter an. Diese Parameter sind alles „by default“ IN-Parameter; eine Unterscheidung zwischen IN/OUT/INOUT gibt es hier nicht!

Die nächste Zeile ist der entscheidende Unterschied zu SPs: Hier gibt man den Datentyp des Rückgabewertes an.

Anschließend muss man noch angeben, ob die Funktion deterministischer Art ist oder nicht. D.h.: Kommt bei jedem Aufruf einer Funktion (mit den gleichen Parametern etc.) das gleiche Ergebnis zurück, ist es eine deterministische Funktion; gleiche Eingaben ergeben gleiche Ausgabe.  
Nicht-Deterministische Funktionen haben vor Allem mit dem Aufruf von RANDOM()- oder Zeit-Funktionen zu tun!  
DETERMINISTIC ist hier der Default.

Nun gibt man zwischen BEGIN und END (m. verändertem Delimiter) den gewünschten Code ein, analog zu den SPs.  
Zu beachten dabei ist, dass es hier ein RETURN-Statement geben muss. Beim Erreichen dieses RETURN-Statements wird die Funktion danach sofort beendet.

Löschen einer Funktion

Wie immer funktioniert auch hier das Löschen einer abgespeicherten Funktion mit

DROP FUNCTION [IF EXISTS] FunctionName;

„Aufruf“ einer Funktion

Eine Funktion kann nun ganz einfach in z.B. eine Abfrage eingebaut werden:

SELECT Spalte1, Spalte2, Function1(Par1, Par2), Function2(Par3, Par4) FROM… ;

Weiteres zu Funktionen

Die Programmierkonstrukte und –Statements funktionieren analog zu denselben in den Stored Procedures. Ausnahmen hierbei sind nur z.B. das Wegfallen von IN/OUT/INOUT und das Hinzukommen eines RETURN-Statements (siehe oben!).

Auflisten von Stored Functions

Mit dem Statement

SHOW FUNCTION STATUS [LIKE ‚pattern‘ | WHERE bedingung];

Kann man sich eine Liste der Funktionen anzeigen lassen, die auf dem DB-Server liegen und auf die man Zugriff hat.

Trigger in MariaDB

Trigger dienen in SQL dazu, bei einem gewissen „Ereignis“ ein (ab-)gespeichertes Programm automatisch ablaufen zu lassen.

Mit „Ereignis“ ist hier gemeint:

* INSERT-
* UPDATE-
* DELETE-

Statements. Für z.B. SELECT-Abfragen steht die Trigger-Funktion nicht zur Verfügung.

Im SQL-Standard sind zwei Arten von Triggern vorgesehen: Zeilenweise Trigger und „Statement-level“ Trigger. Letztere werden nur einmal pro oben aufgelisteten Statements abgearbeitet; erstere machen dies für jede Zeile, die durch die obigen Statements betroffen sind.  
In MariaDB/MySQL gibt es nur die zeilenweisen Trigger!

Vorteile von Triggern:

* Ein weiterer Weg, um Datenkonsistenz oder –Integrität zu gewährleisten.
* Gute Möglichkeit, um Logging-Funktionen zu implementieren
* Möglichkeit, „Aufräumarbeiten“ automatisch zu erledigen

Nachteile:

* Wie oben erwähnt, stehen Trigger nur für INSERT/UPDATE/DELETE zur Verfügung
* Trigger können unter Umständen schwierig zu debuggen sein; speziell durch die Möglichkeit, in einem Trigger weitere Trigger auslösen zu können!
* Trigger können schnell zu hoher Serverlast beitragen.