IS5102
Database Management Systems

Lecture 12: Advanced SQL

Alexander Konovalov

alexander.konovalov@st-andrews.ac.uk

(with thanks to Susmit Sarkar)

2021

This week: Advanced SQL

- Nested queries (subqueries)
- Join Expressions
- Authorisation
- Functions
- Triggers

Built-in Functions

Built-in functions include the following:

- Mathematical functions
- Date and time functions
- System functions
- . . . (**many** more)

Scalar SQL functions (SQLite)

We have seen some SQLite **aggregate** functions already: AVG, COUNT, MAX, MIN, SUM

For a full list and documentation, see https://sqlite.org/lang_aggfunc.html

SQLite also supports a number of built-in **scalar** functions. E.g. the following query:

```
SELECT RANDOM() AS 'RANDOM',
           ABS(-42) AS 'ABS',
     LENGTH('SQL') AS 'LENGTH';
```

produces

RANDOM	ABS	LENGTH
-44132881353127140	42	3

For a full list and documentation, see https://sqlite.org/lang_corefunc.html

Mathematical Functions (MySql)

MySql supports a range of mathematical functions

E.g. the output of the following query:

```
SELECT RAND() AS 'Random number',
       PI(),
       ROUND(PI(),4) AS 'Pi to four d.p.'
```

is

Random number	pi()	Pi to four d.p.
0.8797049102184409	3.141593	3.1416

The `round()` function rounds of a number to a specified number of places. The `ceil()` and `floor()` functions round off up/down to an integer value.

For a full list see:
https://dev.mysql.com/doc/refman/8.0/en/mathematical-functions.html

Date and Time Functions (SQLite example)

```
SELECT DATETIME('NOW') AS 'Current date/time',
       DATE('NOW') AS 'Date',
       TIME('NOW') AS 'Time',
       JULIANDAY('2021-12-06') - JULIANDAY('NOW')
                AS 'Days until exams';
```

For Date and Time functions available in SQLite, see:
https://sqlite.org/lang_datefunc.html

Date and Time Functions (MySQL example)

```
SELECT NOW()     AS 'Current date/time',
       CURTIME() AS 'Time',
       CURDATE() AS 'Date',
       DATEDIFF('2021-12-06', CURDATE())
                 AS 'Days until exams';
```

For Date and Time functions available in MySQL, see:
https://dev.mysql.com/doc/refman/8.0/en/date-and-time-functions.html

Triggers

- A **trigger** is a statement that is executed **automatically** by the system as a **side effect** of a modification to the database.
- An SQL trigger is executed each time an insert, update, or delete operation is performed.
- Triggers can be activated either **before** or **after** the event.

- For a trigger:
  - Specify the **conditions** under which the trigger is to be executed.
  - Specify the **actions** to be taken when the trigger executes.
  - This is referred to as the **Event-Condition-Action model**

Trigger Example

Enforcing naming conventions for courses depending on the department

```
CREATE TRIGGER validate_course
    BEFORE INSERT ON course
BEGIN
    SELECT
        CASE
            WHEN NEW.dept_id = 'CS' AND
                (NEW.course_id NOT LIKE 'CS____' AND
                 NEW.course_id NOT LIKE 'IS____') THEN
                RAISE (ABORT,'CS courses ID should start with CS or IS')
            WHEN NEW.dept_id = 'MATH' AND
                 NEW.course_id NOT LIKE 'MT____' THEN
                RAISE (ABORT,'MATH courses ID should start with MT')
        END;
END;
```

Trigger Example

Recording updates of student credits

```
CREATE TABLE credit_logs (
    id INTEGER PRIMARY KEY,
    stud_id TEXT,
    old_credits NUMERIC,
    new_credits NUMERIC,
    updated_at TEXT);

CREATE TRIGGER log_credits_update
    AFTER UPDATE ON student
    WHEN OLD.tot_cred <> NEW.tot_cred
BEGIN
    INSERT INTO credit_logs (stud_id, old_credits, new_credits, updated_at)
         VALUES (OLD.stud_id, OLD.tot_cred, NEW.tot_cred, DATETIME('NOW'));
END;
```

Triggering Events and Actions in MySQL

- Triggering event can be insert, delete or update

- In MySQL, values of attributes before and after an update can be referenced as follows:
  - `REFERENCING OLD ROW AS`: for deletes and updates
  - `REFERENCING NEW ROW AS`: for inserts and updates

Triggering Events and Actions - example in MySQL

Triggers can be activated before an event, which can serve as extra constraints.

E.g. convert blank grades to NULL.

```
CREATE TRIGGER setnull_trigger
    BEFORE UPDATE OF takes
    REFERENCING NEW ROW AS nrow
    FOR EACH ROW
    WHEN (nrow.grade = ' ')
BEGIN
    SET nrow.grade = NULL;
END;
```

When **Not** To Use Triggers

- Triggers were used earlier for tasks such as
  - Maintaining summary data (e.g., total salary of each department)
  - Replicating databases by recording changes to special relations (called change or delta relations) and having a separate process that applies the changes over to a replica
- There are **better** ways of doing these now:
  - Databases today provide built in materialized view facilities to maintain summary data
  - Databases provide built-in support for replication

Risks of Triggers

Risk of unintended execution of triggers, for example, when

- loading data from a backup copy
- replicating updates at a remote site

Trigger execution can be disabled before such actions.

Functions and Procedures

- SQL standard supports functions and procedures
  - Functions/procedures can be written in SQL itself, or in an external programming language.
  - Functions are particularly useful with specialized data types such as images and geometric objects.

    Example: functions to check if polygons overlap.
- SQL standard also supports a rich set of imperative constructs, including
  - Loops, if-then-else, assignment

SQL Functions & Procedures

- Can write own functions
- Stored in the database

- Functions mainly used where a value is returned
- Procedures may or may not return any value or may return more than one value

- Functions may be used in calculations and procedure in “business logic”

Defining SQL Function – MySQL Example

Define a function that, given the name of a department, returns the count of the number of instructors in that department.

```
DELIMITER //
CREATE FUNCTION dept_count (dept_id CHAR(5))
  RETURNS INTEGER
    BEGIN
      DECLARE d_count INTEGER;
       SELECT COUNT(*) INTO d_count
         FROM instructor
        WHERE instructor.dept_id = dept_id;
       RETURN d_count;
    END //
DELIMITER ;
```

Using defined SQL Function – MySQL Example

Find the department name and budget of all departments with less than 3 instructors.

```
SELECT dept_id, budget
  FROM department
 WHERE dept_count(dept_id) < 3
```

SQL Procedures in MySQL

The dept_count function could instead be written as procedure:

```
CREATE PROCEDURE
   dept_count_proc (IN  dept_id  VARCHAR(20),
                    OUT d_count  INTEGER)
BEGIN
  SELECT COUNT(*) INTO d_count
    FROM instructor
   WHERE instructor.dept_id = dept_count_proc.dept_id
END;
```

Procedures can be invoked either from an SQL procedure or from embedded SQL, using the CALL statement.

```
DECLARE d_count INTEGER;
CALL dept_count_proc('PHYS', d_count);
```

Consolidation

- Chapter 4 and 5, Database System Concepts, 6th Ed. Silberschatz, Korth & Sudarshan
- Chapter 7, Database Systems, Connolly & Begg

Useful URLs:
https://www.w3schools.com/sql/default.asp
https://sqlite.org/lang_createtrigger.html
https://www.sqlitetutorial.net/sqlite-trigger/
https://www.mysqltutorial.org/mysql-stored-procedure-tutorial.aspx
https://www.mysqltutorial.org/create-the-first-trigger-in-mysql.aspx