MySQL Equivalents

Zane Kansil
Loyola Marymount University
Database Systems

November 10, 2014

1. Triggers

   - Example:

```
-- change delimiter so that ‘;‘ is available for use
DELIMITER $$

CREATE
    TRIGGER ‘blog_after_update‘ AFTER UPDATE
    ON ‘blog‘
    FOR EACH ROW BEGIN

        IF NEW.deleted THEN
            SET @changetype = "DELETE";
        ELSE
            SET @changetype = "EDIT";
        END IF;

        INSERT INTO audit (blog_id, changetype) VALUES (NEW.id, @changetype);

    END$$

-- return delimiter to normal
DELIMITER ;
```

     source: “http://www.sitepoint.com/how-to-create-mysql-triggers/”, comments my own

   - For JohnDB:
     Every time a purchase becomes “shipped”, the database should decrement the amount of the physical goods left as stored in the “PHYSICAL_GOOD” table.

```
DELIMITER $$
```

```
CREATE
    TRIGGER ‘reduce_quantity_after_sale‘ AFTER UPDATE
    ON ‘PURCHASE‘
    FOR EACH ROW
    BEGIN
        IF NEW.sale_status = "shipped" THEN
            UPDATE (
                SELECT  *
                FROM       ‘PURCHASE‘      AS ‘p‘
                INNER JOIN ‘LINE_ITEM‘     AS ‘li‘
                ON          p.sale_id = li.sale_id
                INNER JOIN ‘GOOD‘          AS ‘g‘
                ON          li.g_sku = g.g_sku
                INNER JOIN ‘PHYSICAL_GOOD‘ AS ‘pg‘
                ON          g.g_sku = pg.g_sku
            )
            SET pg.quantity_available = pg.quantity_available - li.quantity
        END IF;
    END$$

DELIMITER ;
```

2. Assertions

   There do not appear to be assertions in MySQL. Suggestions on the internet point to using Triggers for assertion-like behaviour.

3. Check Constraints

   Check constraints “CHECK” statements are “parsed but ignored by all storage engines”. Suggestions on the internet point to using Triggers for assertion-like behaviour. In order to achieve this behaviour the user would specify “BEFORE INSERT” and “BEFORE UPDATE” triggers that would do the necessary checks and take specific action if the checks fail.

   - For JohnDB:

```
DELIMITER $$

CREATE TRIGGER non_us_country
BEFORE INSERT ON ‘PHYSICAL_CONSUMER‘
FOR EACH ROW
IF NOT (NEW.pc_country != "US") THEN
    -- some pre-defined procedure that raises an error
    CALL invalid_country_not_US;
```

```
END IF$$

CREATE TRIGGER non_us_country
BEFORE UPDATE ON ‘PHYSICAL_CONSUMER‘
FOR EACH ROW
IF NOT (NEW.pc_country != "US") THEN
    -- some pre-defined procedure that raises an error
    CALL invalid_country_not_US;
END IF$$

DELIMITER ;
```

source: “http://stackoverflow.com/questions/12438818/adding-a-check-constraint-to-a-table-after-it-is-full-of-data”

4. User-defined data types or domains

   User-defined data types and domains are not available in MySQL. They provide a way for the user to define a type that has an query type underlying it. For example, the `pc_phone` property in JohnDB could have the user defined type of `phone` which would be a type that aliases `CHAR(12)`. It is also possible to add checks to defined domains in the SQL dialects that support it, for example PostgreSQL.

5. Foreign key constraint qualifiers

   (a) on DELETE|UPDATE cascade

   Cascading is a style of updating the database in response to changes in other tables. It is useful when database tables have FK references that must be kept up to date or even deleted in response to changes in some parent table.

   - JohnDB example:
     When a purchase is deleted, all line items that refer to it should also be deleted. This prevents an error raised when deleting a row that is referenced by another table element. The result is an atomic operation that deletes an element and deleted whatever else should be deleted as well (same transaction). This is only available to dbms’s that support transactions, so not MySQL.

     ```
     ALTER TABLE ‘LINE_ITEM‘
     ADD FOREIGN KEY (sale_id)
     REFERENCES PURCHASE(sale_id)
     ON DELETE CASCADE;
     ```

   (b) on DELETE — UPDATE restrict

   This is the default behaviour of MySQL tables with foriegn keys. The deletion of parent rows is restricted if any referencing (has FK to) child rows exist.

```
-- No code example given for JohnDB, functionality is included in default MySQL beh
```

(c) on DELETE — UPDATE nullify

If the particular column is a FK and allows NULL values ("IS NULL"), then the column will be set to NULL in the event that the referenced row is deleted. I understand this to be a listener that will respond to referenced rows being deleted.

- JohnDB example:
  When a purchase is deleted, all line items that refer to it should also be deleted. This prevents an error raised when deleting a row that is referenced by another table element. The result is an atomic operation that deletes an element and deleted whatever else should be deleted as well (same transaction). This is only available to dbms's that support transactions, so not MySQL.

```
-- if a customer had a favorite song category I might implement it like
ALTER TABLE ‘CUSTOMER‘
ADD FOREIGN KEY (c_favorite_song)
REFERENCES SONG(m_id)
ON DELETE NULLIFY;
```