SQL Triggers

A.K.A. Active Rules

DB2 : Syntax

<DB2-trigger> :: = CREATE TRIGGER <trigger-name>
{BEFORE | AFTER} <trigger-event>
ON <table-name>
[REFERENCING <references>]
FOR EACH {ROW | STATEMENT}
WHEN (<SQL-condition>)
<SQL-procedure-statements>

<trigger-event> :: = INSERT | DELETE | UPDATE
[ON <column-names>]

<reference> :: = OLD AS <old-value-tuple-name> |
NEW AS new-value-tuple-name |
OLDTABLE AS old-value-table-name |
NEWTABLE AS new-value-table-name

Sources

1. dev.mysql.com
2. www.mysqltutorial.org
3. www.digitalpropulsion.org
4. www.databasedesign-resource.com/ mysql-triggers.htm
5. MySQL Lectures Notes (by Lisa Ball)

Sample DB

```
CREATE TABLE NewsCategories
    ( catID int not null auto_increment,
    catName varchar(32),
    primary key(catID));

CREATE TABLE News
    ( newsID int not null auto_increment,
    catID int not null,
      title varchar(32) not null, txt blob,
     primary key(newsID));

CREATE TABLE NewsCount
    ( newsItemCount int );
```

Trigger Example

```
CREATE TRIGGER newsCounter
   AFTER INSERT ON News
   FOR EACH ROW
    BEGIN
       INSERT INTO NewsCount (newsItemCount)
       (SELECT count(*) FROM News);
     END;
```

Trigger Example

```
CREATE TRIGGER newsCategoryHandler
   AFTER DELETE ON NewsCategories
   FOR EACH ROW BEGIN
    DELETE FROM News
    WHERE catID=OLD.catID;
   END;
```

Trigger Example

```
CREATE TRIGGER newsCounter
  AFTER INSERT ON News
  FOR EACH ROW BEGIN
   DELETE FROM NewsCount;
   INSERT INTO NewsCount (newsItemCount)
    (SELECT count(*) FROM News);
  END;
```

Stored Procedures

- Block structured language similar to Oracle PL/SQL and IBM DB2 SQL
- Some folks recommend using MySQL query browser to aid creation, but can be done from command line
- Seeing what you have
  - SHOW PROCEDURE STATUS;
  - SHOW PROCEUDRE LIKE '%Test%';
  - SHOW CREATE PROCEDURE myproc;

Stored Procedures

- Sample DB

  -- create News table, be sure to be in a 'test' DB
  CREATE TABLE News
  (NewsID int auto_increment not null,
  Title varchar(32),
  primary key(NewsID))

Stored Procedures

```
DELIMITER $$

    DROP PROCEDURE IF EXISTS sprocTest $$
    CREATE PROCEDURE sprocTest (id int, title varchar(32))
    BEGIN
     -- INSERT NEW RECORD IF PREEXISTING RECORD DOESNT EXIST
     IF (id = 0) THEN
       SET id = null;
     END IF;

     IF (id IS NOT NULL) AND (EXISTS
       (SELECT * FROM News WHERE NewsID=id))
     THEN
       UPDATE News SET Title=title WHERE NewsID=id;
     ELSE
       INSERT INTO News (Title) VALUES (title);
     END IF;
    END $$

    DELIMITER ;

To call:

    CALL sprocTest(1,'Some News Title');  -- this will update recordID 1
```

Stored Procedures

- Using cursors
  - Let's us loop on each row returned from a query (the result set)
  - see design-resources link (also on next slide)

More Notes

- First:
  - Chapter 9 Slides 18-37 from Elmasri 5th edition
- Some sources for Java access
  - http://www.kitebird.com/articles/jdbc.html
  - http://www.cs.ucdavis.edu/~devanbu/teaching/160/docs/mysql_java.pdf
  - http://www.romow.com/computer-blog/how-to-use-mysql-with-java/
- Can also access mysql with Perl, PHP, Python, Ruby