Triggers and Active Databases

CSE 532, Theory of Database Systems
Stony Brook University
http://www.cs.stonybrook.edu/~cse532
Trigger Overview

- Element of the database schema

- General form:
  \[ \text{ON} \ <\text{event}>\ \text{IF}\ <\text{condition}>\ \text{THEN}\ <\text{action}> \]

  - Event - request to execute database operation
  - Condition - predicate evaluated on database state
  - Action – execution of procedure that might involve database updates

- Example:
  \[ \text{ON}\ \text{updating maximum course enrollment}\ \text{IF}\ \text{number registered}\ >\ \text{new max enrollment limit}\ \text{THEN}\ \text{deregister students using LIFO policy} \]
Trigger Details

- **Activation** - Occurrence of the *event*
- **Consideration** - The point, after activation, when *condition* is evaluated
  - Immediate or deferred (when the transaction requests to commit)
  - *Condition* might refer to both the state before and the state after *event* occurs
Trigger Details

- **Execution** – point at which *action* occurs
  - With deferred consideration, execution is also deferred
  - With immediate consideration, execution can occur immediately after consideration or it can be deferred
    - If execution is immediate, execution can occur before, after, or instead of triggering event.
    - Before triggers adapt naturally to maintaining integrity constraints: violation results in rejection of event.
Trigger Details

- **Granularity**
  - *Row-level granularity*: change of a single row is an event (a single `UPDATE` statement might result in multiple events)
  - *Statement-level granularity*: events are statements (a single `UPDATE` statement that changes multiple rows is a single event).
Trigger Details

• **Multiple Triggers**
  • How should multiple triggers activated by a single event be handled?
    • Evaluate one condition at a time and if true immediately execute action or
    • Evaluate all conditions, then execute actions
  • The execution of an action can affect the truth of a subsequently evaluated condition so the choice is significant.
Triggers in SQL:1999

• **Events**: INSERT, DELETE, or UPDATE statements or changes to individual rows caused by these statements

• **Condition**: Anything that is allowed in a WHERE clause

• **Action**: An individual SQL statement or a program written in the language of Procedural Stored Modules (PSM) (which can contain embedded SQL statements)
Triggers in SQL:1999

- **Consideration**: *Immediate*
  - Condition can refer to both the state of the affected row or table before *and* after the event occurs

- **Execution**: *Immediate* — can be before or after the execution of the triggering event
  - Action of before trigger cannot modify the database

- **Granularity**: Both *row-level* and *statement-level*
CREATE TRIGGER Max_EnrollCheck
BEFORE INSERT ON Transcript
REFERENCING NEW AS N  --row to be added
FOR EACH ROW
WHEN
((SELECT COUNT(T.StudId) FROM Transcript T
  WHERE T.CrsCode = N.CrsCode
    AND T.Semester = N.Semester)
   >=
   (SELECT C.MaxEnroll FROM Course C
      WHERE C.CrsCode = N.CrsCode ))
ABORT TRANSACTION
After Trigger Example
(row granularity)

CREATE TRIGGER LimitSalaryRaise
AFTER UPDATE OF Salary ON Employee
REFERENCING OLD AS O
    NEW AS N
FOR EACH ROW
WHEN (N. Salary - O. Salary > 0.05 * O. Salary)
    UPDATE Employee -- action
    SET Salary = 1.05 * O. Salary
    WHERE Id = O. Id

Note: The action itself is a triggering event (but in this case a
chain reaction is not possible)
After Trigger Example
(statement granularity)

CREATE TRIGGER RecordNewAverage
AFTER UPDATE OF Salary ON Employee
FOR EACH STATEMENT
INSERT INTO Log
VALUES (CURRENT_DATE,
        SELECT AVG (Salary)
        FROM Employee)