Triggers and Active Databases
(supplemental material)

CSE 532, Theory of Database Systems
Stony Brook University

http://www.cs.stonybrook.edu/~cse532
About Triggers

- The basic structure of a trigger:
  - ON event IF precondition THEN action

- The difference between row-level and statement-level trigger granularities:
  - Row-level: Change of a single row is an event.
  - Statement-level: A statement that can change multiple rows is a single event.

- The trigger precondition:
  - An expression that evaluates to true or false based on database states, i.e., any condition allowed in the WHERE clause of SQL.
Quiz 10

- Create an AFTER trigger that updates the grade for a student transcript to “Pending” when a student swaps a class.

```
CREATE TRIGGER CrsChangeTrigger
AFTER UPDATE OF CrsCode, Semester ON Transcript
FOR EACH ROW
UPDATE Transcript SET Grade = 'Pending';
```
Example 1

- Consider a brokerage rm database with relations Holdings(AccountId, StockSymbol, Price, Quantity) and Balance(AccountId, Balance).
- Triggers for maintaining the correctness of the account balance when stock is bought (a tuple is added to Holdings or Quantity is incremented) or sold (a tuple is deleted from Holdings or Quantity is decremented).

CREATE TRIGGER UpdateBalanceRealTime
  AFTER INSERT, DELETE, UPDATE ON Holdings
  REFERENCING NEW AS N
  FOR EACH ROW
  UPDATE Balance
  SET Balance =
  (SELECT SUM(H.Price*H.Quantity) FROM Holdings H WHERE H.AccountId = N.AccountId)
Example 1

- If Holdings is updated only periodically (e.g., every day), then the trigger can work by erasing the old contents of Balance and then recomputing it from scratch:

```sql
CREATE TRIGGER UpdateBalanceAllAtOnce
AFTER INSERT, DELETE, UPDATE ON Holdings
FOR EACH STATEMENT
BEGIN
    DELETE FROM Balance;
    INSERT INTO Balance
    SELECT DISTINCT H.AccountId, SUM(H.Price*H.Quantity)
    FROM Holdings H
    GROUP BY H.AccountId
END
```
Example 2

- Consider the following schema: Student(Student, Status), Took(Student, Course), Course(Course, Credits, Type).
  - Status can be 'B' (beginner), 'CPR' (completed program requirements), and 'EG' (eligible to graduate).
  - Type can be 'C' (core course) or 'E' (elective course).
  1. Row-level trigger which monitors insertions into Took: when a 'CPR' student completes 130 credits, change the student's status to 'EG'.
  2. Another row-level trigger which monitors insertions into Took: when a beginner student completes all core ('C') courses plus 3 electives ('E'), change the status from 'B' to 'CPR'.

(c) Pearson and P. Fodor (CS Stony Brook)
Example 2

1. Row-level trigger which monitors insertions into Took: when a 'CPR' student completes 130 credits, change the student's status to 'EG'.

```sql
CREATE TRIGGER EligibleToGraduate AFTER INSERT ON Took
REFERENCING NEW AS N FOR EACH ROW
WHEN (
    130 <= ( SELECT SUM(C.Credits)
    FROM Took T, Course C, Student S
    WHERE T.Student = N.Student AND T.Course = C.Course
    AND T.Student = S.Student AND S.Status = 'CPR'
    )
)
UPDATE Student
SET Status = 'EG'
WHERE N.Student = Student
```
2. Another row-level trigger which monitors insertions into Took: when a beginner student completes all core ('C') courses plus 3 electives ('E'), change the status from 'B' to 'CPR'.

```
CREATE TRIGGER DoneWithProgram AFTER INSERT ON Took
FOR EACH ROW REFERENCING NEW AS N
WHEN (  
    EXISTS ( SELECT * FROM Student S -- Student has status 'B'
                WHERE N.Student = S.Student AND S.Status = 'B' )
    AND NOT EXISTS (---Division
                    (SELECT C.Course FROM Course C WHERE C.Type = 'C')
                    EXCEPT
                    (SELECT T.Course FROM Took T
                        WHERE T.Student = N.Student))
    AND 3 <= ( SELECT COUNT(T.Course) FROM Took T, Course C
                WHERE T.Student = N.Student AND T.Course = C.Course
                AND C.Type = 'E' )
)
    UPDATE Student SET Status = 'CPR' WHERE N.Student = Student
```