Active Databases: Constraints (Assertions)  
Triggers

Douglas S. Kerr  
CIS 671  
Elmasri & Navathe, EN §23.1

Constraints (Assertions)  
• Integrity Constraints  
  (Basic Relational DB Theory)  
  – Domain Constraints  
  – Key constraints (Primary Keys)  
  – Referential Integrity (Foreign Keys)  
• General constraints (Assertions) (EN §8.6)  
  – E.g., Salary may not exceed $200,000.  
• Triggers (EN §23.1)

Triggers: The Problem - Examples from COMPANY Database

- Limit all salary increases to 50%.  
- Enforce policy that salaries may never decrease.  
- Maintain TotalSalary in DEPARTMENT relation as employees and their salaries change.  
- Inform a supervisor whenever a supervisee’s salary becomes larger than the supervisor’s.  
- All new hires for a given job code get the same starting salary, which is available in the STARTING_PAY table.

The Problem: Example from BANK Database  

- Branch Assets are maintained as \( \text{sum(Account.Balance)} \) for each branch.  
- Overdrafts do not produce a negative balance. Instead they are treated as a loan. The account balance is set to 0 and a loan is created for the amount of the overdraft.

The Problem: Example from TEMPERATURE Database

- Given table of temperatures \( \text{TEMPS} \), that is periodically updated, keep the table of extreme temperatures \( \text{EXTREMES} \) up to date.
  
Create the \( \text{EXTREMES} \) table and populate with all the cities in the \( \text{TEMPS} \) table, setting the other attributes to null.

What is Needed? The Event-Condition-Action Model (ECA Model)

- Rules (or triggers) with three components:  
  – Event triggering the rule, (insert, delete, update)  
    • E.g., an employee’s salary changes.  
  – Condition to determine if rule action should be executed.  
    • E.g., is new Temp in City higher than HighTemp for that City?  
  – Action to be taken.  
    • E.g., update the Department’s Total Salary.
What is Needed?
The Event-Condition-Action Model (ECA Model), continued

• Actions may apply before or after the triggering event is executed.
• An SQL statement may change several rows.
  – Apply action once per SQL statement.
  – Apply action for each row changed by SQL statement.

Availability

• Triggers included in SQL 1999 (SQL 3)
  – Not in earlier standards.
• Included much earlier in most products:
  – Oracle, Sybase, DB2
  – As a consequence syntax may differ from the standard.

The Problem:
Examples from COMPANY Database

1. COMPANY Database
   Limit all salary increases to 50%
   before trigger emp_salary_limit

```sql
create trigger emp_salary_limit
before update of EMPLOYEE
for each row
when (newSalary > 1.5 * oldSalary)
set newSalary = 1.5 * oldSalary;
```

```
Method depends on DBMS.
```

2. COMPANY Database
   Enforce policy that salaries may never decrease
   before trigger emp_salary_no_decrease

```sql
create trigger emp_salary_no_decrease
before update of EMPLOYEE
for each row
when (newSalary < oldSalary)
begin
  log the event;
  signal error condition;
end
```

```
```

5. COMPANY Database: All new hires for a given job code get the same starting salary, which is available in the STARTING_PAY table.
   before trigger emp_start_pay

```sql
create trigger emp_start_pay
before insert on EMPLOYEE
for each row
set Salary = (select StartPay
from STARTING_PAY
where JobCode = new.JobCode)
```

```
```

EMPLOYEE (Name, SSN, Salary, DNO, SupervisorSSN, JobCode)
DEPARTMENT (DNO, TotalSalary, ManagerSSN)
STARTING_PAY (JobCode, StartPay)
7. BANK Database: **Overdrafts** do not produce a negative balance. Instead they are treated as a **loan**. The account balance is set to 0 and a loan is created for the amount of the overdraft.

```sql
alter trigger after update on Account
begin
    insert into Loan values (new.AccountNo, new.BranchID, -new.Balance);
    insert into LoanCustomer (select CustID, AccountNo
        from AccountCustomer AC
        where new.AccountID = AC.AccountID);
    update Account
    set Balance = 0
    where Account.AccountNo = new.AccountNo;
end;
```

7. BANK Database: **Overdrafts**, continued

```sql
create trigger after update on Account
begin
    insert into Loan values (new.AccountID, new.BranchID, -new.Balance);
    insert into LoanCustomer (select CustID, AccountNo
        from AccountCustomer AC
        where new.AccountID = AC.AccountID);
    update Account
    set Balance = 0
    where Account.AccountNo = new.AccountNo;
end;
```

8. TEMPS Database: Create the **EXTREMES** table and populate with all the cities in the **TEMPS** table, setting the other attributes to null.

```sql
create trigger HighTempUpdate after update on TEMPS
begin
    for each row
    when new.Temp >
    ( select HighTemp
        from EXTREMES
        where City= new.City )
    or ( select HighTemp
        from EXTREMES
        where City = new.City ) is null
    update EXTREMES
    set HighTemp = new.Temp,
    HighDate = current date
    where City = new.City;
end;
```

8. TEMPS Database: continued. **Other Problems**

- Insert a new `City` into **TEMPS**.
  - Must also insert into **EXTREMES**.
  - Initial values of `HighTemp`, `HighDate`, `LowTemp`, `LowDate` must be set.
- Delete a `City` from **TEMPS**.
  - Leave `City` in **EXTREMES**.
  - Delete `City` from **EXTREMES**.
8. TEMPS Database: continued.
Insert a new City into TEMPS:
Insert City tuple into EXTREMES.
Set initial values of HighTemp, HighDate, LowTemp, LowDate.

```
create trigger NewCity
after insert of TEMPS
for each row
insert into EXTREMES(City, HighTemp, HighDate, LowTemp, LowDate)
values(new.City, new.Temp, Current Date, new.Temp, Current Date);
```

Active Database
CH 671
29

Problems with Use of Triggers

- How to guarantee set of triggers is consistent?
- **Recursion** is allowed.
  - How to guarantee termination?
- Tools are still needed to help address these problems.

Active Database
CH 671
21

9. COMPANY Database

For each employee, Span is the number of employees supervised.

```
create trigger EmpQuit
after delete on EMPLOYEE
for each row
update EMPLOYEE
set Span = Span - 1
where SSH = old.SupervisorSSN;
```

```
create trigger EmpTransfer
after update of SupervisorSSN on EMPLOYEE
for each row
begin
update EMPLOYEE
set Span = Span - 1
where SSH = old.SupervisorSSN;
end;
```

```
create trigger EmpHire
after insert of EMPLOYEE
for each row
update EMPLOYEE
set Span = Span + 1
where SSH = new.SupervisorSSN;
```

```
create trigger EmpTransfer
after update of SupervisorSSN on EMPLOYEE
for each row
begin
update EMPLOYEE
set Span = Span + 1
where SSH = new.SupervisorSSN;
end;
```

Active Database
CH 671
23
24
9. COMPANY Database
Propagate Span up the supervisor tree.

create trigger EmpPropagate 
after update of Span on 
EMPLOYEE 
for each row 
update EMPLOYEE 
set Span = Span + 
(new.Span - old.Span) 
where SSN = 
new.SupervisorSSN;

10. For each PERSON, record their mother, father and number of descendants.

create trigger NewMother 
after insert on PERSONS 
for each row 
update PERSONS 
set NumDescendants = NumDescendants + 1 
where Name = new.Mother;

create trigger NewFather 
after insert on PERSONS 
for each row 
update PERSONS 
set NumDescendants = NumDescendants + 1 
where Name = new.Father;

Then update the maternal and paternal ancestors.

create trigger MaternalAncestor 
after update of NumDescendants 
on PERSONS 
for each row 
update PERSONS 
set NumDescendants = NumDescendants + new.NumDescendants - old.NumDescendants 
where Name = new.Mother;

create trigger PaternalAncestor 
after update of NumDescendants 
on PERSONS 
for each row 
update PERSONS 
set NumDescendants = NumDescendants + new.NumDescendants - old.NumDescendants 
where Name = new.Father;

/* Similar. 
Just replace "Mother" with "Father." */

At nth level of family tree, how many triggers?