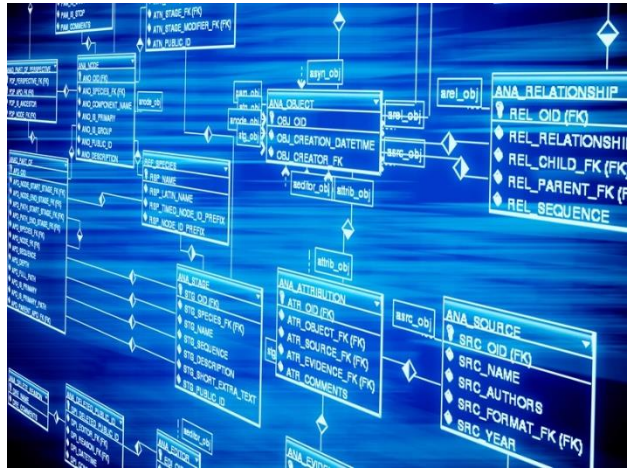
Politecnico di Milano - AA 2018-2019

Prof.ssa Sara Comai

Active Databases – Some Exercises



EXERCISE

Consider the following database:

EMPLOYEE(id, name, role, department)

TRIP(code, employee, destination, date, km, reg_plates)

CAR(reg_plates, model, cost_km)

DESTINATION(name, country)

and the following view:

TRIPS(employee, km_tot, total_cost)

Write two active rules that calculate the value of the view as a result of insertion of new trips:

a) **in incremental way**

b) **recalculating the entire view**

Incremental way

```
CREATE TRIGGER CalculateIncrementalView
AFTER INSERT ON Trip
FOR EACH ROW
BEGIN
    UPDATE Trips
    SET km_tot =  km_tot + new.km,
        total_cost = total_cost + new.km * ( SELECT cost_km
                                             FROM Car
                                             WHERE reg_plates = new.reg_plates )
    WHERE employee = new.employee
END
```

Recalculating the view

```
CREATE TRIGGER CalculateViewEntirely
AFTER INSERT ON Trip
FOR EACH STATEMENT
BEGIN
DELETE * FROM Trips;
INSERT INTO Trips
    SELECT employee, sum(km), sum(km*cost_km)
    FROM Trip T JOIN Car C ON T.reg_plates = C.reg_plates
    GROUP BY employee
END
```

EXERCISE

Consider the database about students' scholarships:

SCHOLARSHIP (StudentID, Points)

EXAM (StudentID, CourseID, Date, Mark)

COURSE (CourseID, Name, Year, NumberOfCredits)

The points of the scholarship of a student are obtained as the sum of the points of each passed exam, computed as (**mark*number of credits**). Identify all the operations that may affect this computation, and write a set of triggers to keep the total amount of points automatically updated.

EVENTS TO MONITOR?

Solution

```
CREATE TRIGGER POINTS1
AFTER INSERT ON EXAM
FOR EACH ROW
BEGIN
UPDATE SCHOLARSHIP
SET POINTS = POINTS + NEW.MARK * (SELECT NUMBEROFCREDITS
                                 FROM COURSE
                                 WHERE COURSEID=NEW.COURSEID)
WHERE STUDENTID=NEW.STUDENTID
END;
```

Solution

```
CREATE TRIGGER POINTS2
AFTER UPDATE OF MARK ON EXAM
FOR EACH ROW
BEGIN
UPDATE SCHOLARSHIP
SET POINTS= POINTS +(NEW.MARK-OLD.MARK)*
                    (SELECT NUMBEROFCREDITS
                    FROM EXAM JOIN COURSE ON COURSEID=COURSEID
                    WHERE COURSEID=NEW.COURSEID)
WHERE STUDENTID=NEW.STUDENTID;
END;
```

Solution

```
CREATE TRIGGER POINTS3
AFTER UPDATE OF NUMBEROFCREDITS ON COURSE
FOR EACH ROW
BEGIN
UPDATE SCHOLARSHIP
SET POINTS=POINTS+(NEW.NUMBEROFCREDITS-OLD.NUMBEROFCREDITS)*
                    (SELECT MARK
                    FROM EXAM JOIN COURSE ON COURSEID=COURSEID
                    WHERE COURSEID=NEW.COURSEID)
WHERE STUDENTID=NEW.STUDENTID
```

Solution

DELETE ON EXAM? (Integrity constraints can be used)

DELETE ON COURSE? (Integrity constraints can be used)

```
create trigger newScholarship
AFTER insert ON scholarship
BEGIN
UPDATE scholarship
SET points  =   (SELECT sum(mark*numOfCredits)
                 FROM exam JOIN course
                 WHERE studentID=NEW.studentID)
WHERE studentID=NEW.studentID
END;
```

EXERCISE

A database supports a song contest, voted by a technical jury and using also a televote system. Specify a set of triggers for the automatic management of voting, taking into account the following rules. The system keeps track of the phone voting received in a certain time interval (managed by transactions): from the same telephone number only the first 10 votes can be accepted, which will contribute to the total number of televotes of the corresponding songs. In parallel, the jury expresses a score between 1 and 10 for each song: each vote will update the total number of jury votes and the total score of the corresponding song in table VOTE. The voting process ends when all the members of the jury have voted all the songs; at this point, the winning song (ties are allowed) is identified, by assigning a weight of 50% to the total number of televotes and the total score of the two voting systems. Assume that the VOTE table is initialized with all the songs and that all the scores are set to 0 and CompositeVote= NULL.

EXERCISE

JURYMEMBER (IDmember, Lastname, Name)

SONG (IDsong, Title, Author, Singer)

JURYVOTE (IDmember, IDsong, Score)

TELEVOTE (TelephoneNumber, Time, IDsong)

VOTE (IDsong, #Televotes, #JuryVotes, TotJuryScore, CompositeVote, Winner)

EXERCISE

A database supports a song contest, voted by a technical jury and using also a televote system. Specify a set of triggers for the automatic management of voting, taking into account the following rules. The system keeps track of the phone voting received in a certain time interval (managed by transactions): **from the same telephone number only the first 10 votes can be accepted**, which will contribute to the total number of televotes of the corresponding songs. In parallel, the jury expresses a score between 1 and 10 for each song: each vote will update the total number of jury votes and the total score of the corresponding song in table VOTE. The voting process ends when all the members of the jury have voted all the songs; at this point, the winning song (ties are allowed) is identified, by assigning a weight of 50% to the total number of televotes and the total score of the two voting systems. Assume that the VOTE table is initialized with all the songs and that all the scores are set to 0 and CompositeVote= NULL.

EXERCISE

A database supports a song contest, voted by a technical jury and using also a televote system. Specify a set of triggers for the automatic management of voting, taking into account the following rules. The system keeps track of the phone voting received in a certain time interval (managed by transactions): **from the same telephone number only the first 10 votes can be accepted**, **which will contribute to the total number of televotes of the corresponding songs**. In parallel, the jury expresses a score between 1 and 10 for each song: each vote will update the total number of jury votes and the total score of the corresponding song in table VOTE. The voting process ends when all the members of the jury have voted all the songs; at this point, the winning song (ties are allowed) is identified, by assigning a weight of 50% to the total number of televotes and the total score of the two voting systems. Assume that the VOTE table is initialized with all the songs and that all the scores are set to 0 and CompositeVote= NULL.

EXERCISE

A database supports a song contest, voted by a technical jury and using also a televote system. Specify a set of triggers for the automatic management of voting, taking into account the following rules. The system keeps track of the phone voting received in a certain time interval (managed by transactions): **from the same telephone number only the first 10 votes can be accepted**, **which will contribute to the total number of televotes of the corresponding songs**. In parallel, **the jury expresses a score between 1 and 10 for each song: each vote will update the total number of jury votes and the total score of the corresponding song in table VOTE.** The voting process ends when all the members of the jury have voted all the songs; at this point, the winning song (ties are allowed) is identified, by assigning a weight of 50% to the total number of televotes and the total score of the two voting systems. Assume that the VOTE table is initialized with all the songs and that all the scores are set to 0 and CompositeVote= NULL.

EXERCISE

A database supports a song contest, voted by a technical jury and using also a televote system. Specify a set of triggers for the automatic management of voting, taking into account the following rules. The system keeps track of the phone voting received in a certain time interval (managed by transactions): **from the same telephone number only the first 10 votes can be accepted**, **which will contribute to the total number of televotes of the corresponding songs**. In parallel, **the jury expresses a score between 1 and 10 for each song: each vote will update the total number of jury votes and the total score of the corresponding song in table VOTE.** **The voting process ends when all the members of the jury have voted all the songs; at this point, the winning song (ties are allowed) is identified**, by assigning a weight of 50% to the total number of televotes and the total score of the two voting systems. Assume that the VOTE table is initialized with all the songs and that all the scores are set to 0 and CompositeVote= NULL.

EXERCISE

A database supports a song contest, voted by a technical jury and using also a televote system. Specify a set of triggers for the automatic management of voting, taking into account the following rules. The system keeps track of the phone voting received in a certain time interval (managed by transactions): **from the same telephone number only the first 10 votes can be accepted**, **which will contribute to the total number of televotes of the corresponding songs**. In parallel, **the jury expresses a score between 1 and 10 for each song: each vote will update the total number of jury votes and the total score of the corresponding song in table VOTE.** **The voting process ends when all the members of the jury have voted all the songs; at this point, the winning song (ties are allowed) is identified**, **by assigning a weight of 50% to the total number of televotes and the total score of the two voting systems.** Assume that the VOTE table is initialized with all the songs and that all the scores are set to 0 and CompositeVote= NULL.

[...] from the same telephone number only the first 10 votes can be accepted [...]

```
CREATE TRIGGER CheckTelevotes
BEFORE INSERT ON Televote
FOR EACH ROW
WHEN ( SELECT COUNT(*)
        FROM Televote
        WHERE TelephoneNumber = new.TelephoneNumber ) > 10
ROLLBACK;
```

[...] which will contribute to the total number of televotes of the corresponding songs [...]

```
CREATE TRIGGER CountTelevote
AFTER INSERT ON Televote
FOR EACH ROW
BEGIN
    UPDATE Vote
    SET #Televotes = #Televotes + 1
    WHERE IDsong = new.IDsong
END
```

[...] jury expresses a score between 1 and 10 for each song: each vote will update the total number of jury votes [...]

```
CREATE TRIGGER CountJuryVotes
AFTER INSERT ON JuryVote
FOR EACH ROW
BEGIN
    UPDATE Vote
    SET #JuryVotes = #JuryVotes + 1,
          TotJuryScore = TotJuryScore + new.Score
    WHERE IDsong = new.IDsong
END
```

[...] by assigning a weight of 50% to the total number of televotes and the total score of the two voting systems [...]

```
CREATE TRIGGER ComputeCompositeVote
AFTER UPDATE OF #JuryVote ON Vote
FOR EACH ROW
WHEN #JuryVote = SELECT COUNT(*) FROM JuryMember
BEGIN
    UPDATE Vote
    SET CompositeVote = 0.5 * #Televotes + 0.5 * TotJuryScore
    WHERE IDsong = new.IDsong
END
```

[...] The voting process ends when all the members of the jury have voted all the songs; at this point, the winning song (ties are allowed) is identified [...]

```
CREATE TRIGGER ComputeWinner
AFTER UPDATE OF CompositeVote ON Vote
FOR EACH ROW
WHEN NOT EXISTS ( SELECT *  FROM Vote WHERE CompositeVote IS NULL)
BEGIN
    UPDATE Vote SET Winner = false;
    UPDATE Vote SET Winner = true
    WHERE CompositeVote = ( SELECT MAX(CompositeVote)
                            FROM Vote)
END
```