Trigger exercises: Event calendar management

Given the following relational schema (primary keys are underlined):

EVENT(EvID, EventName, EventCategory, EventCost, EventDuration)
CALENDAR(EvID, Date, StartTime, Location)
CATEGORY_SUMMARY(EventCategory, Date, TotalNumberEvents, TotalEventCost)

You are requested to manage the planning of events in the city of Turin for the anniversary of the 150th anniversary of the unification of Italy (Italia 150).
Events belong to different categories (EventCategory attribute), such as exhibitions, debates, screenings, and are characterized by a cost of realization (EventCost attribute). Each event can be repeated several times on different dates. The CALENDAR table shows the planning of events on different days and places in the city. Write triggers to handle the following tasks.

(1) *Updating the table CATEGORY_SUMMARY*. The table CATEGORY_SUMMARY shows, for each category of event and for each date, the total number of events planned and the total cost for their realization. Write the trigger to propagate changes to the CATEGORY_SUMMARY table when a new calendar event is inserted (CALENDAR table insertion).

(2) *Integrity constraint on the maximum cost of the event*. The cost of an event in the film screening category (EventCategory attribute) cannot exceed 1500 euros. If a cost value greater than 1500 is entered in the EVENT table, the EventCost attribute must be assigned a value of 1500. Write the trigger for handling the integrity constraint.

(3) *Constraint on the maximum number of events per date*. No more than 10 events can be scheduled on any given date. Any changes to the table CALENDAR that cause the constraint violation should not be performed.

(1) *Updating the table CATEGORY_SUMMARY*

- **EVENT;** INSERT ON CALENDAR
- **TARGET TABLE (MUTATING TABLE):** CALENDAR
- **ACTION:** For each EVENT entered in CALENDAR
  - Read from EVENT the category and cost of the EVENT
  - Check to see if there is already a record for that EVENT category in table CATEGORY_SUMMARY with reference to the date on which the EVENT was scheduled
    - If the record already exists, the overall cost and the total number of events in the record in CATEGORY_SUMMARY is updated
    - If the record is not present, a new record is inserted for that category and date in CATEGORY_SUMMARY
- **GRANULARITY**: tuple (FOR EACH ROW)
- **EXECUTION MODE**: AFTER
- **CONDITION:** not present

```
CREATE OR REPLACE TRIGGER update_category_sommary
AFTER INSERT ON CALENDAR
FOR EACH ROW
DECLARE
        X CHAR(10);
        Y NUMBER;
        Z NUMBER;
BEGIN
--- for the EVENT added to the calendar  select the respective category and cost
SELECT EventCategory, EventCost into X,Y
FROM EVENT
WHERE CodE = :NEW.CodE;

--- Check if there is a row in the table of contents corresponding to the pair <EventCategory, Date>
SELECT COUNT(*) INTO Z
FROM CATEGORY_SUMMARY
WHERE EventCategory = X AND Date = :NEW.Date;

  IF (Z = 0) THEN
        --- insert the new pair <category, date> in CATEGORY_SUMMARY
        INSERT INTO CATEGORY_SUMMARY(EventCategory, Date, TotalNumberEvents,
        TotalEventCost) VALUES (X, :NEW.Date, 1, Y);
  ELSE
        --- update the values for the pair <category, date> already existing in CATEGORY_SUMMARY
        UPDATE CATEGORY_SUMMARY
        SET TotalEventCost = TotalEventCost + Y,
             TotalNumberEvents = TotalNumberEvents +1
         WHERE EventCategory = X AND Date = :NEW.Date;
  END IF; END;
```

(2) *Integrity constraint on the maximum cost of the event*.

- **EVENT:** INSERT ON EVENT or UPDATE of EventCost, EventCategory ON EVENT
- **TARGET TABLE (MUTATING TABLE):** EVENT
- **ACTION:** For each EVENT entered or modified in EVENT
    - If the EVENT is in the category Screening and the cost of the EVENT is more than 1500
        - Assign the value 1500 to the cost of the EVENT
- **GRANULARITY**: tuple (FOR EACH ROW)
- **EXECUTION MODE**: BEFORE
- **CONDITION:** the EVENT is in the category Screening and the cost of the EVENT is more than 1500

```
CREATE OR REPLACE TRIGGER maximum_event_cost
BEFORE INSERT OR UPDATE OF EventCost, EventCategory ON EVENT
FOR EACH ROW
WHEN ((NEW.EventCategory = 'Proiezione') AND (NEW.EventCost > 1500))
BEGIN
        :NEW.EventCost := 1500;
END;
```

(3) *Constraint on the maximum number of events per date.*

- **EVENT:** INSERT ON CALENDAR or UPDATE of Date ON CALENDAR
- **TARGET TABLE (MUTATING TABLE):** CALENDAR
- **ACTION:** For each EVENT entered or modified in CALENDAR
  - Verify whether there is at least one Date with more than 10 calendar events in the updated CALENDAR table

```
SELECT Date
FROM CALENDAR
GROUP BY Date
HAVING COUNT(*) > 10
```

    - If yes, the operation on the CALENDAR table must be canceled
- **GRANULARITY**: statement (defined ***not writing*** FOR EACH ROW)
- **EXECUTION MODE**: AFTER
- **CONDITION:** cannot be specified on statement triggers

```
CREATE TRIGGER verifica_numero_eventi
AFTER INSERT OR UPDATE of Date ON CALENDAR
DECLARE
        X NUMBER;
BEGIN

SELECT COUNT(*) INTO X
FROM CALENDAR
WHERE Date IN
                    (SELECT Date
                     FROM CALENDAR
                     GROUP BY Date
                     HAVING COUNT(*) > 10);
IF (X<>0) THEN
        raise_application_error (XXX, 'Too many events in one Date');
END IF;
END;
```