

## Esercizi trigger: Gestione calendario eventi

Si consideri il seguente schema relazionale (le chiavi primarie sono sottolineate):

EVENTO(CodE, NomeEvento, CategoriaEvento, CostoEvento, DurataEvento)

CALENDARIO\_EVENTI(CodE, Data, OraInizio, Luogo)

SOMMARIO\_CATEGORIA(CategoriaEvento, Data, NumeroTotaleEventi, CostoComplessivoEventi)

Si vuole gestire la pianificazione degli eventi nella città di Torino per l'anniversario dei 150 anni dell'unità d'Italia (Italia 150).

Gli eventi appartengono a diverse categorie (attributo CategoriaEvento), quali mostre, dibattiti, proiezioni, e sono caratterizzati da un costo di realizzazione (attributo CostoEvento). Ciascun evento può essere ripetuto più volte in date diverse. La tabella CALENDARIO\_EVENTI riporta la pianificazione degli eventi in diversi giorni e luoghi della città. Si scrivano i trigger per gestire le seguenti attività.

(1) *Aggiornamento della tabella SOMMARIO\_CATEGORIA.* La tabella SOMMARIO\_CATEGORIA riporta, per ogni categoria di evento e per ogni data, il numero complessivo di eventi previsti e il costo complessivo per la loro realizzazione.

Si scriva il trigger per propagare le modifiche alla tabella SOMMARIO\_CATEGORIA quando viene inserito un nuovo evento a calendario (inserimento nella tabella CALENDARIO\_EVENTI).

(2) *Vincolo di integrità sul costo massimo dell'evento.* Il costo di un evento della categoria proiezione cinematografica (attributo CategoriaEvento) non può essere superiore a 1500 euro. Se un valore di costo superiore a 1500 è inserito nella tabella EVENTO, all'attributo CostoEvento deve essere assegnato il valore 1500. Si scriva il trigger per la gestione del vincolo di integrità.

(3) *Vincolo sul numero massimo di eventi per data.* In ogni data non possono essere pianificati più di 10 eventi. Ogni modifica della tabella CALENDARIO\_EVENTI che causa la violazione del vincolo non deve essere eseguita.

(1) *Aggiornamento della tabella SOMMARIO\_CATEGORIA*

- **EVENTO;** INSERT ON CALENDARIO\_EVENTI
- **TABELLA TARGET (TABELLA MUTANTE):** CALENDARIO\_EVENTI
- **AZIONE:** Per ogni evento inserito in CALENDARIO\_EVENTI
  - Leggere da EVENTO la categoria e il costo dell'evento
  - Verificare se nella tabella SOMMARIO\_CATEGORIA è già presente un record per quella categoria di evento con riferimento alla data in cui l'evento è stato pianificato
    - Se è già presente il record, viene aggiornato il costo complessivo e il numero totale di eventi nel record in SOMMARIO\_CATEGORIA
    - Se non è presente il record, viene inserito un nuovo record per quella categoria e quella data in SOMMARIO\_CATEGORIA
- **GRANULARITA':** tupla (FOR EACH ROW)
- **MODO DI ESECUZIONE:** AFTER
- **CONDIZIONE:** non presente

```

CREATE OR REPLACE TRIGGER aggiorna_sommario_categoria
AFTER INSERT on CALENDARIO_EVENTI
FOR EACH ROW
DECLARE
    X CHAR(10);
    Y NUMBER;
    Z NUMBER;
BEGIN
--- per l'evento aggiunto a calendario seleziono le rispettive categoria e costo
SELECT CategoriaEvento, CostoEvento into X,Y
FROM EVENTO
WHERE CodE = :NEW.CodE;

--- verifico se nel sommario è presente una riga corrispondente alla coppia <categoria evento, data>
SELECT COUNT(*) INTO Z
FROM SOMMARIO_CATEGORIA
WHERE CategoriaEvento = X AND Data = :NEW.Data;

IF (Z = 0) THEN
    --- inserisco la nuova coppia <categoria, data> in SOMMARIO_CATEGORIA
    INSERT INTO SOMMARIO_CATEGORIA(CategoriaEvento, Data, NumeroTotaleEventi,
    CostoComplessivoEventi) VALUES (X, :NEW.Data, 1, Y);
ELSE
    --- aggiorno i valori per la coppia <categoria, data> già esistente in SOMMARIO_CATEGORIA
    UPDATE SOMMARIO_CATEGORIA
    SET CostoComplessivoEventi = CostoComplessivoEventi + Y,
        NumeroTotaleEventi = NumeroTotaleEventi +1
    WHERE CategoriaEvento = X AND Data = :NEW.Data;
END IF; END;

```

(2) *Vincolo di integrità sul costo massimo dell'evento.*

- **EVENTO:** INSERT ON EVENTO oppure UPDATE di CostoEvento, CategoriaEvento ON EVENTO
- **TABELLA TARGET (TABELLA MUTANTE):** CALENDARIO\_EVENTI
- **AZIONE:** Per ogni evento inserito in CALENDARIO\_EVENTI o per ogni evento nel record modificato
  - Se l'evento è della categoria "Proiezione" e il costo dell'evento è superiore a 1500
    - Assegnare il valore 1500 al costo dell'evento
- **GRANULARITA':** tupla (FOR EACH ROW)
- **MODO DI ESECUZIONE:** BEFORE
- **CONDIZIONE:** verificare se l'evento è della categoria "Proiezione" e il costo dell'evento è superiore a 1500

```
CREATE OR REPLACE TRIGGER massimo_costo_evento
BEFORE INSERT OR UPDATE OF CostoEvento, CategoriaEvento ON EVENTO
FOR EACH ROW
WHEN ((NEW.CategoriaEvento = 'Proiezione') AND (NEW.CostoEvento > 1500))
BEGIN
    :NEW.CostoEvento := 1500;
END;
```

(3) *Vincolo sul numero massimo di eventi per data.*

- **EVENTO;** INSERT ON CALENDARIO\_EVENTI oppure UPDATE di Data ON CALENDARIO\_EVENTI
- **TABELLA TARGET (TABELLA MUTANTE):** CALENDARIO\_EVENTI
- **AZIONE:**
  - Verifica se, nella tabella CALENDARIO\_EVENTI aggiornata, esiste almeno una data con più di 10 eventi in calendario
  - Date con più di 10 eventi in calendario

```
SELECT Data
FROM CALENDARIO_EVENTI
GROUP BY Data
HAVING COUNT(*) > 10
```

    - Se sì, l'operazione di modifica sulla tabella CALENDARIO\_EVENTI deve essere annullata
- **GRANULARITA':** istruzione (definita \*\*\*non scrivendo\*\*\* FOR EACH ROW)
- **MODO DI ESECUZIONE:** AFTER
- **CONDIZIONE:** non può essere specificata su trigger di istruzione

```
CREATE TRIGGER verifica_numero_eventi
AFTER INSERT OR UPDATE of Data ON CALENDARIO_EVENTI
DECLARE
    X NUMBER;
BEGIN

SELECT COUNT(*) INTO X
FROM CALENDARIO_EVENTI
WHERE Data IN
    (SELECT Data
    FROM CALENDARIO_EVENTI
    GROUP BY Data
    HAVING COUNT(*) > 10);

IF (X<>0) THEN
    raise_application_error (XXX, 'Troppi eventi in una data');
END IF;
END;
```