TRIGGER

I trigger, come le asserzioni, sono innescati automaticamente a seguito di operazioni di aggiornamento. A differenza delle asserzioni, però, consentono di specificare le azioni da compiere sulla BD in corrispondenza di determinati eventi (**BD attive**).

Un trigger (o regola) si basa sul modello ECA (event-condition, action):

- Evento: operazione di aggiornamento che attiva la regola
- Condizione: determina se l'azione debba essere eseguita; è facoltativa
- Azione: sequenza di istruzioni SQL da eseguire quando l'evento si verifica e la condizione (se esiste) è verificata

I trigger vengono usati essenzialmente per mantenere l'integrità dei dati, calcolare dati derivati o gestire eccezioni, ecc.

Si consideri la seguente BD:

- Imp(matr, nome, stip, dip, superiore)
- Dip(cod, nome, stip-tot, direttore, IdAzi)
- Azienda(IdAzi, nome, città)

dove *superiore* e *direttore* sono chiavi secondarie definite sulla chiave primaria di Imp.

ATTRIBUTI CALCOLATI

- lo stipendio totale di un dipartimento deve essere aggiornato quando si inserisce un nuovo impiegato

Operazione che scatena il trigger: Insert(252, aldo, 25.000, d1, 333)

```
CREATE TRIGGER Stip-Tot1
AFTER INSERT ON Impiegato       % evento
FOR EACH ROW
WHEN (NEW.dip IS NOT NULL)  % condizione
      UPDATE Dip                          %azione
      SET stip-tot = stip-tot+NEW.stip
      WHERE Dip.cod = NEW.dip)
```

- lo stipendio totale di un dipartimento deve essere aggiornato quando un impiegato viene trasferito di dipartimento

Operazione che scatena il trigger:

```
Update Imp
Set dip = d2
Where matr=252

CREATE TRIGGER Stip-Tot2
AFTER UPDATE OF Dip ON Impiegato    % evento
FOR EACH ROW
      UPDATE Dip                          %azione
```

```
    SET stip-tot = stip-tot - OLD.stip
    WHERE Dip.cod = OLD.dip;
    UPDATE Dip                          %azione
    SET stip-tot = stip-tot + OLD.stip
    WHERE Dip.cod = NEW.dip; )
```

INTEGRITA' DEI DATI

Si possono gestire vincoli di integrità. La differenza con le asserzioni è che queste possono solo disfare un aggiornamento che ha violato un vincolo, mentre i trigger consentono una gestione "attiva" attraverso la definizione delle azioni che devono essere compiute.

- Esempio: il campo direttore di Dip deve essere messo a NULL quando la corrispondente tupla in Imp viene cancellata

Operazione che scatena il trigger: Delete Imp where matr=100

```
CREATE TRIGGER Integrità-Ref-Direttore
AFTER DELETE ON Impiegato                    % evento
FOR EACH ROW
    UPDATE Dip
    SET direttore=NULL
    WHERE direttore=OLD.matr
```

- Esempio. Creare una regola che riduca del 10% lo stipendio di *tutti* gli impiegati di un Dip quando la media dei salari dei dipententi di quel dip supera i 40.0000€

```
CREATE TRIGGER Rid-Salario
After Insert IMP
For each row
WHEN (Select avg(stip) From Imp Where dip = NEW.dip) > 40.000 //cond non a livello di riga
    Update Imp
    Set stip = stip – 0,1*stip
    Where dip = NEW.dip
```

NOTA: la condizione viene verificata dopo (AFTER) l'inserimento del nuovo impiegato.

TRIGGER CHE INNESCA UN ALTRO TRIGGER

- La cancellazione di una azienda comporta la cancellazione dei suoi dipartimenti

```
CREATE TRIGGER Integrità-Ref-Azienda
AFTER DELETE ON Azienda                      % evento
FOR EACH ROW
    DELETE Dip
    WHERE idAzi=OLD.IdAZI
```

La cancellazione di una dipartimento comporta la cancellazione dei suoi dipendenti

```
CREATE TRIGGER Integrità-Ref-Dip
AFTER DELETE ON Dip                          % evento
FOR EACH ROW
```

```
DELETE Imp
WHERE dip=OLD.cod
```

- ESEMPIO: trigger ricorsivo che cancella tutti i subordinati di un impiegato cancellato (integrità referenziale)

```
CREATE TRIGGER Cancella-Subordinati
AFTER  DELETE ON Imp
FOR EACH ROW
    DELETE Imp
    WHERE superiore = OLD.matr
```

La ricorsione non sempre è supportata dai DBMS; in alcuni casi è necessario impostare l'opzione RECURSIVE_TRIGGERS.

BEFORE e AFTER

1. I "before trigger" vengono usati per "condizionare" l'esito dell'operazione oppure per bloccarla segnalando errore
2. Gli "after trigger" servono a "reagire" alla modifica del DB mediante opportune azioni

- L'incremento massimo di stipendio è del 20%; se superiore, mantienilo comunque al 20%

```
CREATE TRIGGER LimiteStip
BEFORE UPDATE OF Stip ON Impiegato
FOR EACH ROW
WHEN (NEW.Stip > 1,2 * OLD.Stip)
      SET NEW.Stip = 1,2 * OLD.Stip
```

Il valore dello stipendio viene modificato prima che venga modificata la base di dati.
SET: serve a modificare uno o più campi di una nuova tupla (NEW)
NOTA: Una volta modificate le tuple di Impiegato, scatta il trigger Stip-Tot2 per l'aggiornamento dello stipendio totale di Dip.

Se si usa l'opzione AFTER, il trigger viene formulato come segue:

```
CREATE TRIGGER LimiteStip
AFTER UPDATE OF Stip ON Impiegato
FOR EACH ROW
WHEN (NEW.Stip > 1,2 * OLD.Stip)
      UPDATE Impiegato
      SET NEW.Stip = 1,2 * OLD.Stip
      WHERE matr = NEW.matr
```

In tal caso, la correzione viene fatta a posteriori, dopo che la BD è stata modificata.

- si ha un vincolo che impone di assegnare ad ogni nuovo impiegato lo stipendio minimo del suo dipartimento, a meno che non sia diversamente specificato:

Quando si usa il BEFORE, SET può essere solo sulle nuove tuple

```
CREATE TRIGGER ImpStipMin
BEFORE INSERT ON Impiegato
FOR EACH ROW
WHEN (New.Stip = NULL)
        SET New.Stip = (SELECT MIN(Stip) FROM Impiegato WHERE dip = New.dip),
```

Oppure (soluzione meno efficiente)

```
CREATE TRIGGER ImpStipMin
AFTER INSERT ON Impiegato
FOR EACH ROW
WHEN (New.Stip = NULL)
        UPDATE Imp
        SET New.Stip = (SELECT MIN(Stip) FROM Impiegato WHERE dip = New.dip),
        Where matr=new.matr
```

- rispristina il vecchio valore quando il nuovo stipendio è maggiore di quello del superiore

```
CREATE TRIGGER StipTroppoAlto
BEFORE UPDATE Impiegato on Stip
FOR EACH ROW
WHEN (New.Stip > (SELECT Stip FROM Impiegato WHERE matr = New.Sup) )
        SET Stip = Old.Stip
```