

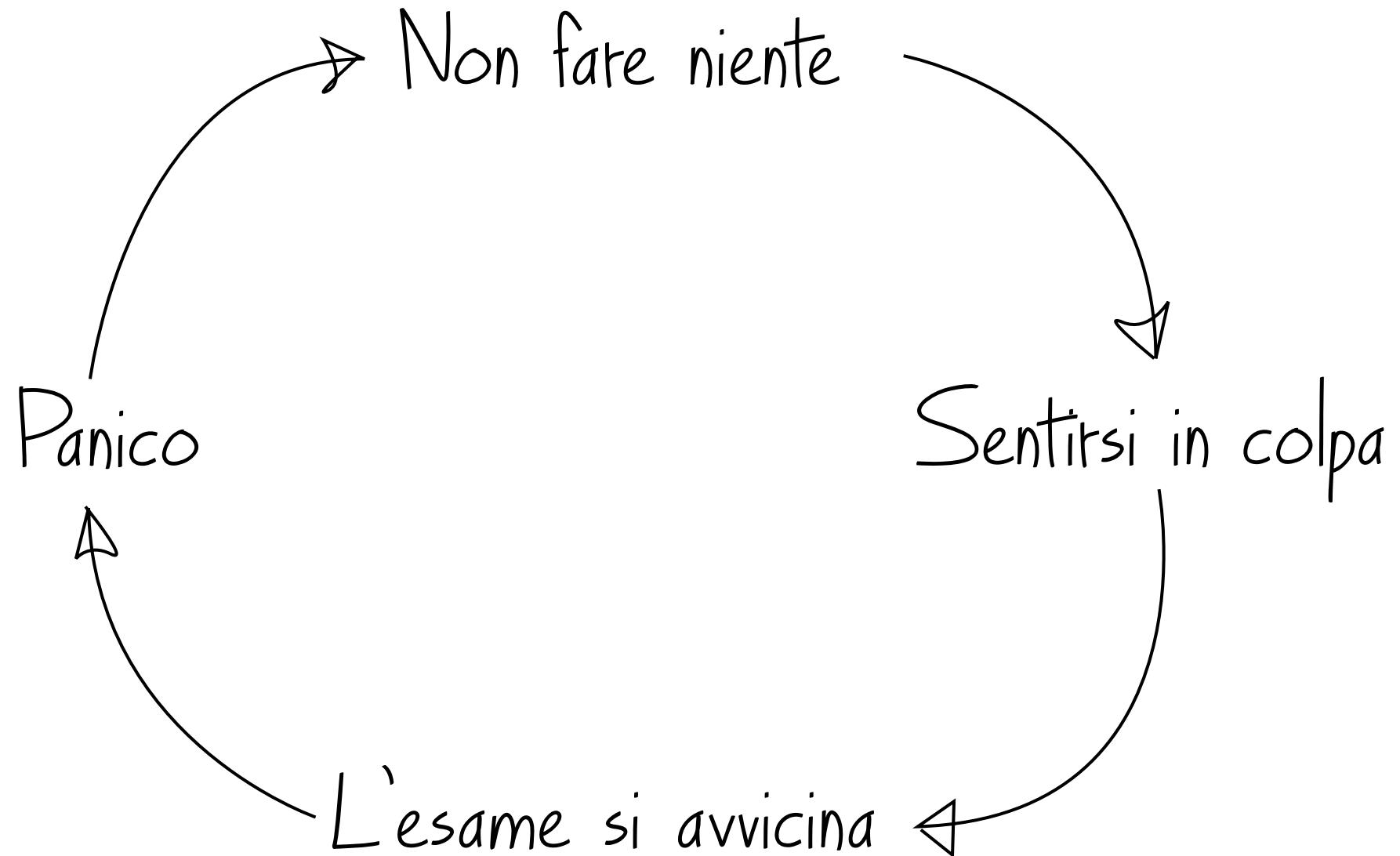
Basi di dati

Corso di laurea in Ingegneria Informatica
Scuola di Ingegneria – Università di Pisa

Oracle MySQL
A.A. 2017-2018

Ing. Francesco Pistoletti
Postdoctoral Researcher
Data Science and Engineering Lab
Dipartimento di Ingegneria dell'Informazione
francesco.pistoletti@iet.unipi.it

È Maggio...

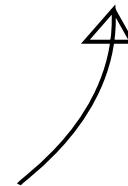


Data definition



Data definition

cioè il supporto per i dati



Segue la progettazione dei dati e permette di **creare le fondamenta** del sistema informativo



se i dati non sono definiti coerentemente con la realtà,
alcune operazioni potrebbero non essere fattibili

Data definition language (DDL)

Parte di linguaggio SQL per la **creazione/modifica/cancellazione** (non solo di tabelle) mediante i comandi **CREATE / ALTER / DROP**

è assistita dagli IDE

Creazione



Creazione: sintassi

[DROP TABLE IF EXISTS] NuovaTabella;

CREATE TABLE [IF NOT EXISTS] NuovaTabella(

 NomeAttributo1

 TipoAttributo1 **[NULL|NOT NULL]** **[DEFAULT** ValoreDefault1] **[AUTO_INCREMENT]**,

 :
 :

 NomeAttributoN

 TipoAttributoN **[NULL|NOT NULL]** **[DEFAULT** ValoreDefaultN] **[AUTO_INCREMENT]**,

PRIMARY KEY (AttributiSeparatiDaVirgola),

 /* vincoli aggiuntivi separati da virgola */

) **TYPE** InnoDB|MyISAM;

Vincoli aggiuntivi

Durante la creazione di una tabella, possono essere definiti i vincoli di chiave
primaria, candidata, e di integrità referenziale

vanno specificati all'interno della create (possono essere modificati)



Chiavi candidate

NON TRATTATO A LEZIONE

UNIQUE (AttributiSeparatiDaVirgola)



Possono esistere più chiavi candidate.
Ogni chiave candidata è espressa da un vincolo unique.

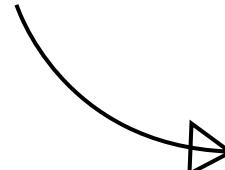
Esempio: l'omonimia

NON TRATTATO A LEZIONE

Creare una tabella Medico2 nella quale la chiave è Matricola. Inoltre, non sono accettati medici omonimi. Inserire i giusti vincoli di chiave.

```
CREATE TABLE Medico2 (
    Matricola VARCHAR(100) NOT NULL,
    Nome VARCHAR(100) NOT NULL,
    Cognome VARCHAR(100) NOT NULL,
    Specializzazione VARCHAR(100) NOT NULL,
    Citta VARCHAR(50) NOT NULL,
    Parcella DOUBLE NOT NULL,
    PRIMARY KEY (Matricola),
    UNIQUE (Nome, Cognome),
) ENGINE=InnoDB DEFAULT CHARSET=latin1
```

chiave candidata





Vincoli di integrità referenziale

NON TRATTATO A LEZIONE

È un "modo di gestire" le tabelle
che contempla la transazionalità

- le tabelle coinvolte devono essere **InnoDB**
- le tabelle coinvolte devono avere un **indice** sugli attributi coinvolti
- gli attributi oggetto **non** possono essere di tipo TEXT o BLOB

Vincoli di integrità referenziale

NON TRATTATO A LEZIONE

CONSTRAINT nome_vincolo_integrità

FOREIGN KEY (Attributo1,..., AttributoN)

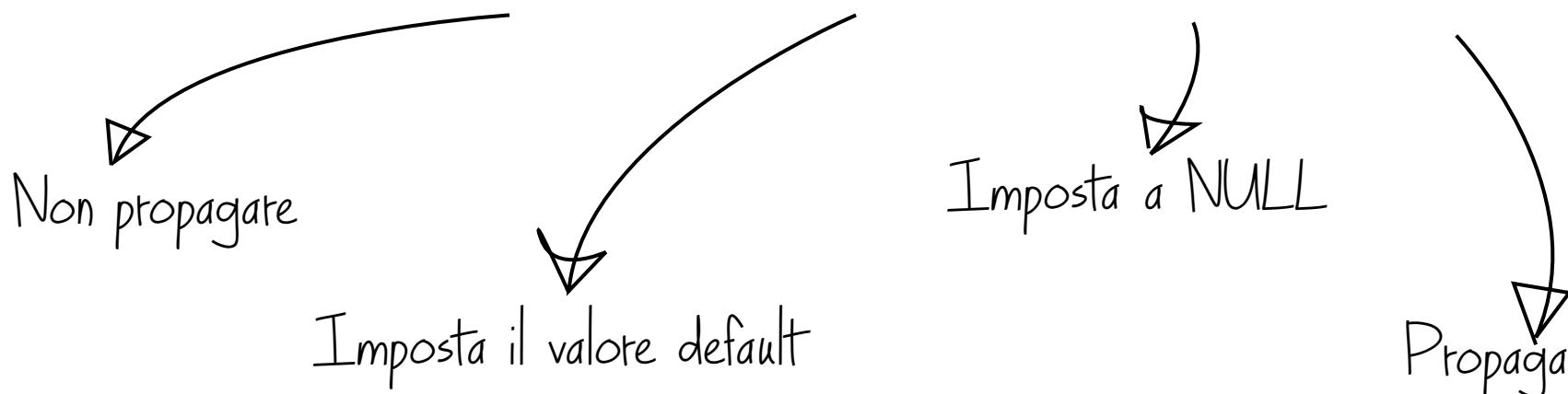
attributi vincolati di questa tabella

REFERENCES TabellaReferenced(Attributo1,...,AttributoN)

referenced

ON UPDATE NO ACTION / SET DEFAULT / SET NULL / CASCADE

ON DELETE NO ACTION / SET DEFAULT / SET NULL / CASCADE



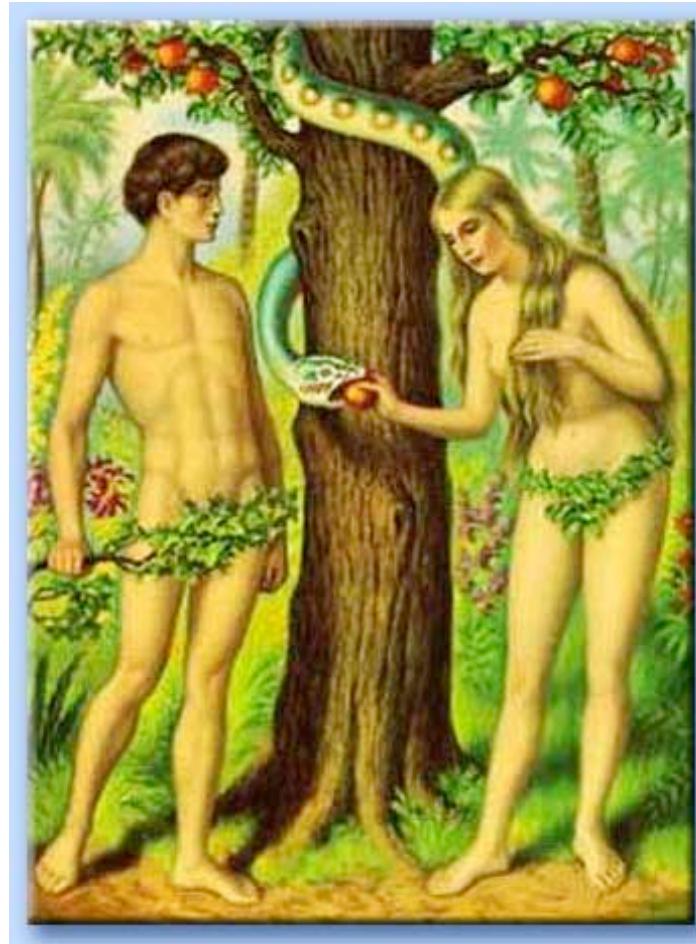
Esempio

NON TRATTATO A LEZIONE

Esprimere un vincolo di integrità referenziale per la tabella Visita che imponga l'esistenza (nella tabella Medico) del medico che ha effettuato una visita. Se una visita di un medico è eliminata o aggiornata non deve accadere nulla al record relativo nella tabella Medico.

```
CONSTRAINT FK_Visita_Medico FOREIGN KEY (Medico)
REFERENCES MEDICO(Matricola)
ON UPDATE NO ACTION
ON DELETE NO ACTION
```

Modifica



Modifica



Modifica strutturale

Aggiunta di
una colonna

ALTER TABLE Tabella

ADD COLUMN NomeColonna TipoColonna



Rimozione di
una colonna

ALTER TABLE Tabella

DROP COLUMN NomeColonna



Esempio

Aggiungere un attributo DataInserimento nella tabella Paziente. Tale attributo conterrà la data in cui un paziente è stato inserito nel database della clinica.

```
ALTER TABLE Paziente  
    ADD COLUMN DataInserimento DATE;
```

se vogliamo l'attributo in una specifica posizione si può usare **FIRST** o **AFTER**

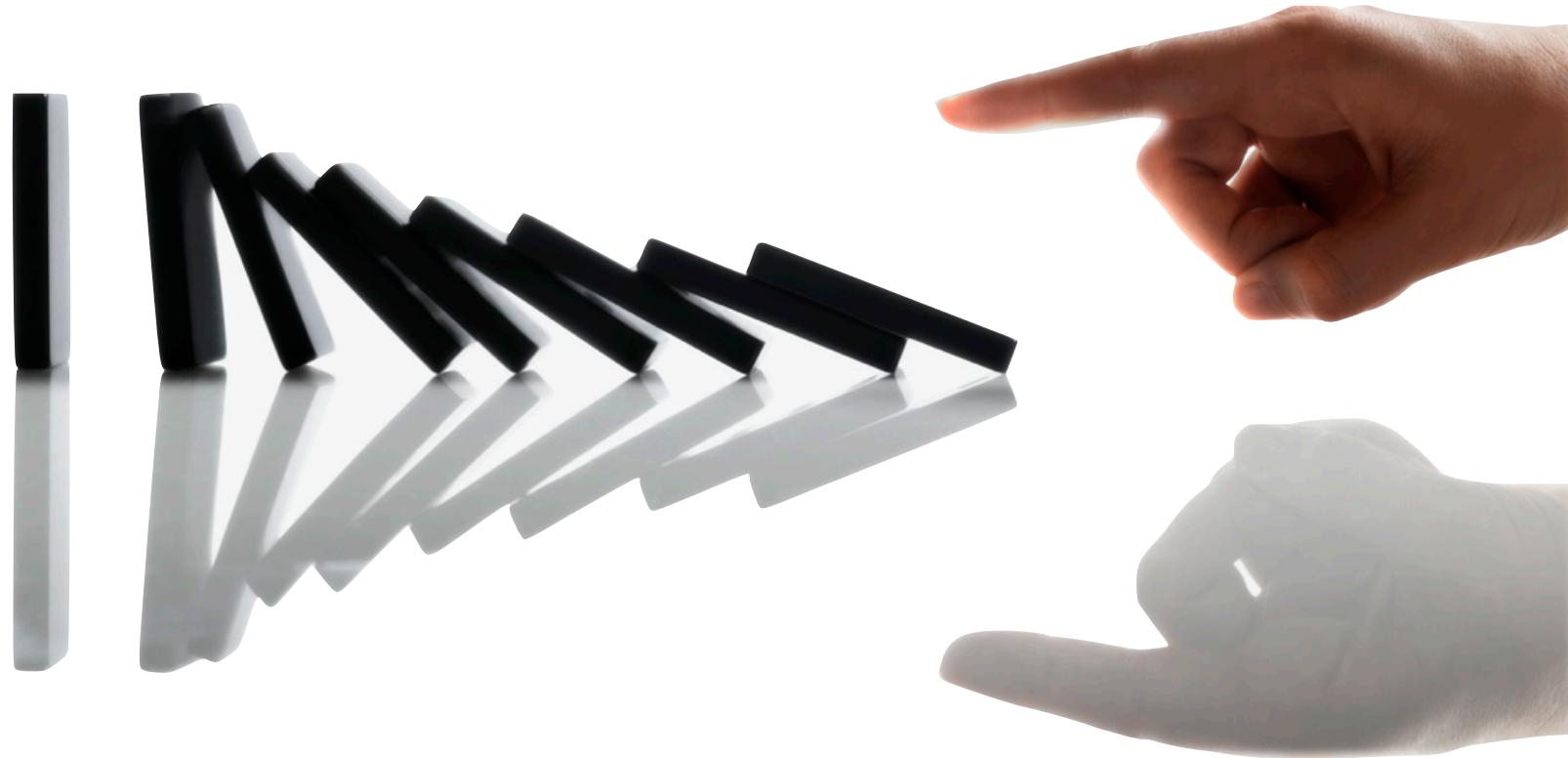
Modifica della chiave

Sostituire l'attuale chiave primaria della tabella Visita con un attributo ID automaticamente incrementato dal sistema.. Posizionare il nuovo attributo chiave come primo attributo del nuovo schema. Si mantenga il vincolo di unicità sulla vecchia chiave primaria.

ALTER TABLE Visita

```
ADD COLUMN IDVisita INT NOT NULL AUTO_INCREMENT FIRST,  
DROP PRIMARY KEY,  
ADD PRIMARY KEY (IDVisita),  
ADD UNIQUE(Medico, Paziente, Data);
```

Database attivi





Database attivi

eventi scatenanti (aggiornamenti o istanti temporali)

Sono dotati di una parte **reattiva** che permette loro di reagire ai cambiamenti mediante comportamenti specifici

particolari operazioni sui dati

Evento

Causa che scatena l'esecuzione di una azione (operazione) sui dati

→ inserimenti, cancellazioni, modifiche o timer



Condizione

Predicato booleano che viene valutato dopo l'evento e, se vero, provoca l'esecuzione dell'azione (è opzionale)

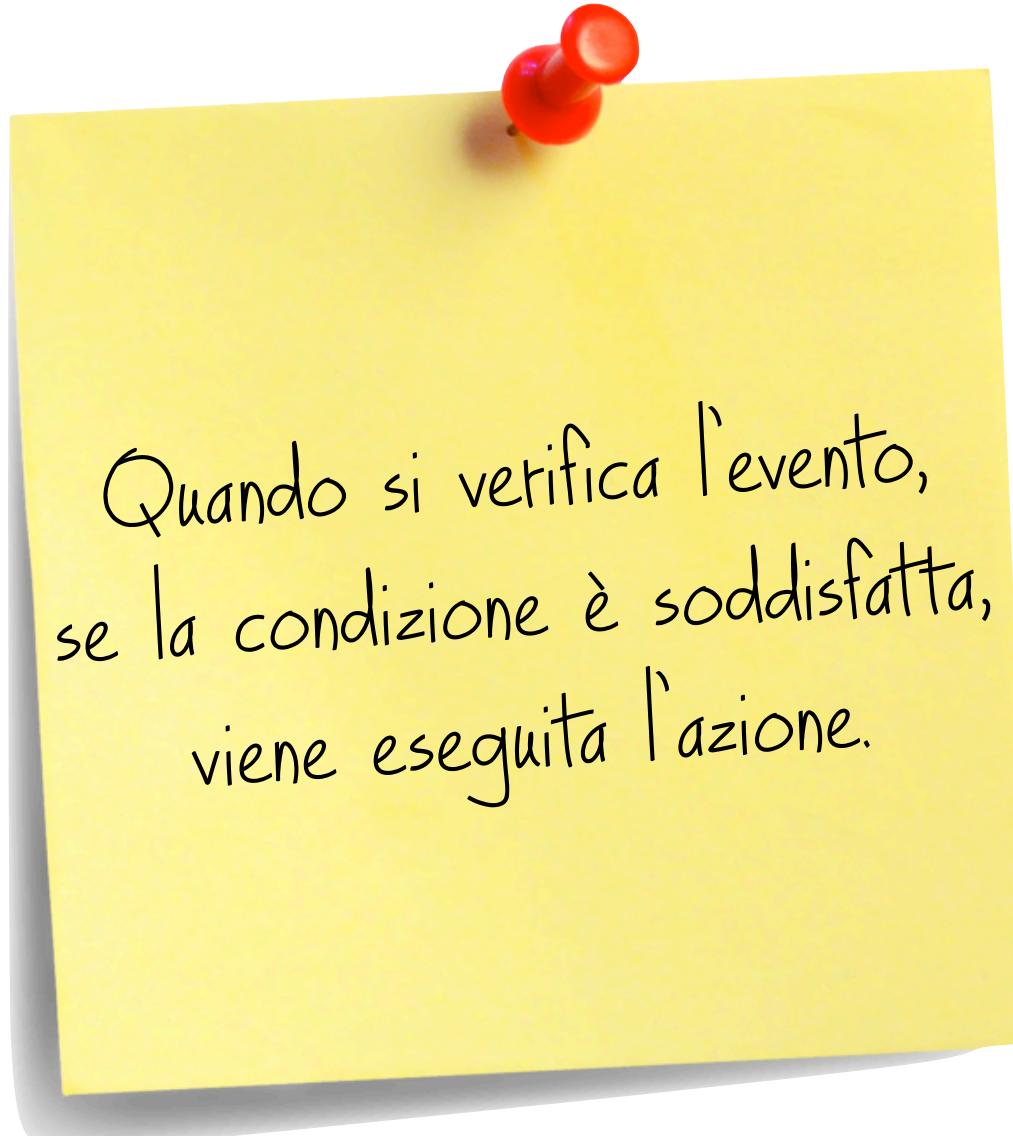


Azione

È formata dalle azioni che compie il trigger dopo lo scatto. Si specifica mediante il **linguaggio dichiarativo-procedurale MySQL**



Evento - Condizione - Azione



Trigger

Trigger

È un **insieme di istruzioni** che sono eseguite sui dati al verificarsi di un evento scatenante.

generalmente il codice è dichiarativo,
con alcune estensioni procedurali

inserimento, cancellazione, modifica

Esempio

Gestire un attributo ridondante nella tabella Paziente contenente la data nella quale un paziente è stato visitato l'ultima volta.

È ridondante perché
lo si può trovare nella tabella Visita.

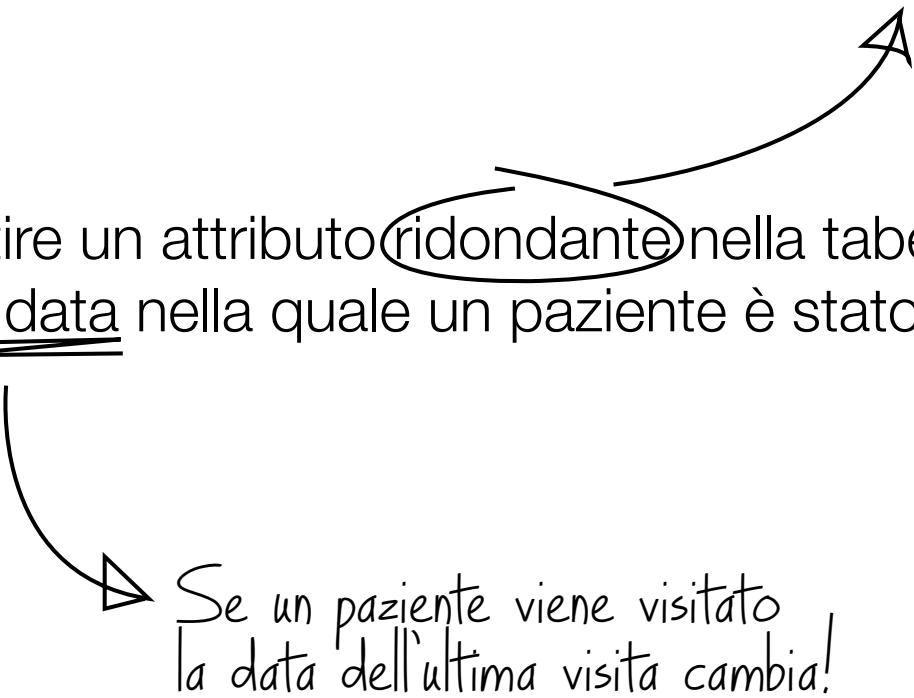


Tabella Paziente con ridondanza

<u>CodFiscale</u>	Cognome	Nome	UltimaVisita
BVEMDL68	Bove	Maddalena	2013-04-15
LPREDR75	Lepre	Edoardo	2012-11-07
CPRRNZ82	Capra	Renzo	2013-02-25
FRNLTA74	Fataona	Elettra	2013-03-30

Esempio: evento, condizione, azione

Gestire un attributo ridondante nella tabella Paziente contenente la data nella quale un paziente è stato visitato l'ultima volta.



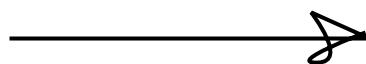
Evento



Inserimento di una nuova tupla
nella tabella Visita



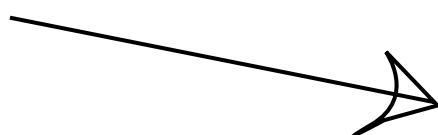
Condizione



Non c'è



Azione



Aggiornare l'attributo UltimaVisita nella
tabella Paziente con la data corrente

Sintassi MySQL per i trigger



```
DROP TRIGGER IF EXISTS nome_trigger;  
  
CREATE TRIGGER nome_trigger  
[BEFORE | AFTER] [INSERT | UPDATE | DELETE] ON TabellaTarget  
FOR EACH ROW blocco_istruzioni
```



Crea il trigger nome_trigger.
Prima (dopo) che una o più row siano (siano state) inserite o modificate nella TabellaTarget, o cancellate da essa, per ciascuna, esegui il blocco_istruzioni.

Before vs. After

per esempio, operazioni sugli attributi della row
che si sta inserendo o modificando



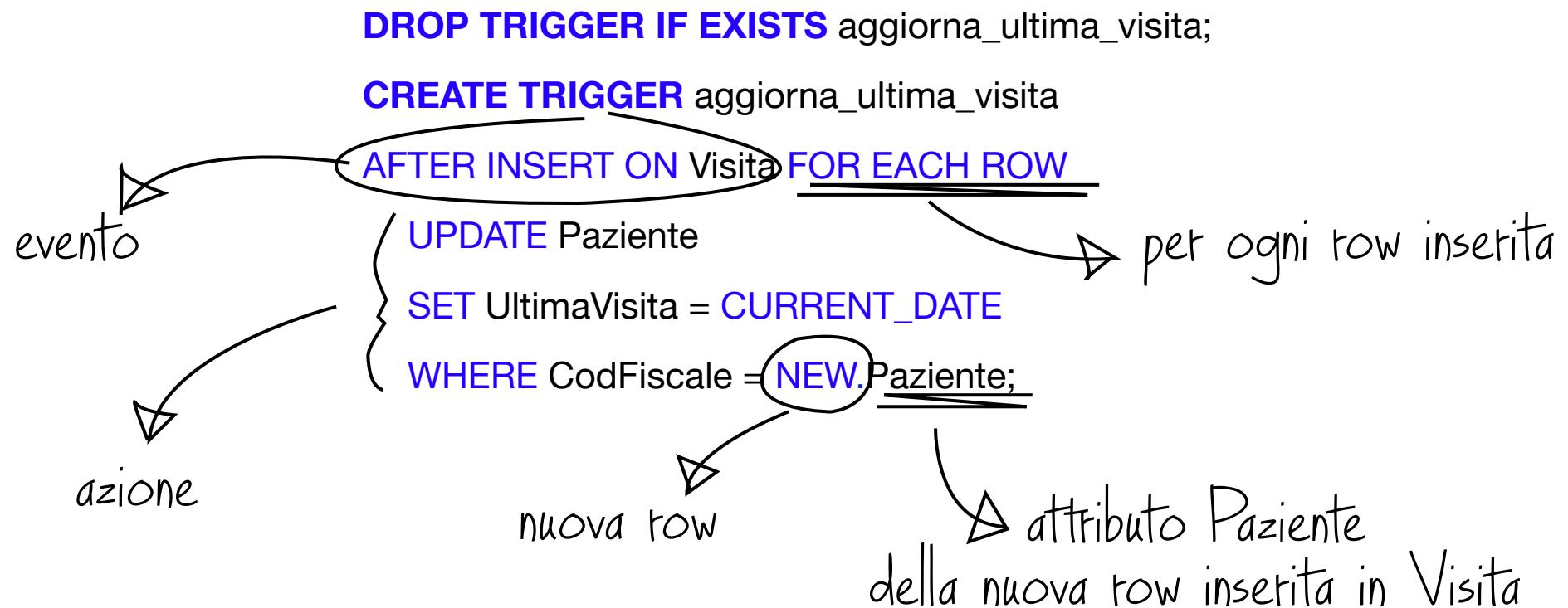
Con **BEFORE** si indica un'azione di **preprocessing**, mentre
AFTER denota un'azione di **a posteriori**, o comunque "collaterale"



una conseguenza, come ad esempio
una operazione su un'altra row o un'altra tabella

Trigger in MySQL

Gestire un attributo ridondante nella tabella Paziente contenente la data nella quale un paziente è stato visitato l'ultima volta.



Trigger multi-statement

ogni blocco è terminato da punto e virgola



L'azione svolta dal trigger è composta da **più blocchi di istruzioni** ed è specificata mediante una sintassi dichiarativo-procedurale

MySQL deve capire che dopo il punto e virgola inizia un nuovo blocco di istruzioni del trigger

Sintassi MySQL per trigger multi-statement

```
DROP TRIGGER IF EXISTS nome_trigger;  
DELIMITER $$  
CREATE TRIGGER nome_trigger  
[BEFORE | AFTER] [INSERT | UPDATE | DELETE] ON TabellaTarget  
FOR EACH ROW  
BEGIN  
Istruzione1;  
Istruzione2;  
...  
IstruzioneN;  
END$$  
DELIMITER ;
```

statement dichiarativo-procedurali
terminati da ;

L'azione finisce qui

Esempio

Scrivere un trigger che, ogni volta che viene inserita una nuova visita, se essa è mutuata, imposta l'attributo *Ticket* in base alle fasce di reddito annue

- ticket pari a euro 36.15 se reddito fra euro 0 ed euro 15,000
- ticket pari a euro 45.25 se reddito fra euro 15,000 ed euro 25,000
- ticket pari a 50.00 euro se reddito oltre 25,000 euro.

Se la visita non è mutuata, inserire NULL.

<u>Medico</u>	<u>Paziente</u>	<u>Data</u>	<u>Mutuata</u>	<u>Ticket</u>
AMRADV44	NTRLRL52	2013-05-14	TRUE	36.15
CLSCLL49	BVEMDL64	2013-05-18	FALSE	NULL
RSSMRO56	RCCEGS32	2013-05-18	FALSE	NULL
AMRADV44	MLLNIA68	2013-05-19	TRUE	45.25

Evento, condizione, azione

Scrivere un trigger che, ogni volta che viene inserita una nuova visita, se essa è mutuata, imposti l'attributo *Ticket* in base alle fasce di reddito annue

- ticket pari a euro 36.15 se reddito fra euro 0 ed euro 15,000
- ticket pari a euro 45.25 se reddito fra euro 15,000 ed euro 25,000
- ticket pari a 50.00 euro se reddito oltre 25,000 euro.

Se la visita non è mutuata, inserire NULL.

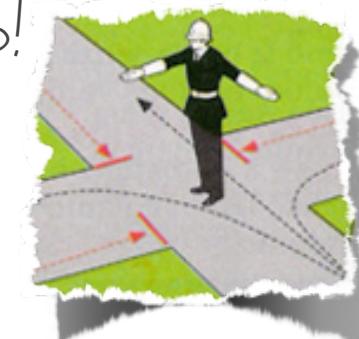


evento



azione

Occorre un controllo di flusso!



Dubbio amletico...

EVENTO → ogni volta che viene inserita una nuova visita



Evento



Evento

BEFORE INSERT ON Visita FOR EACH ROW

"prima di inserire una o più tuple in VISITA, per ognuna di esse..."

Calcolo del reddito annuo

	<u>Medico</u>	<u>Paziente</u>	<u>Data</u>	<u>Mutuata</u>	<u>Ticket</u>
NEW	018	NTRLRL64	2013-05-24	False	

```
SET @reddito_annuo = ( SELECT Reddito*12  
FROM Paziente  
WHERE CodFiscale = NEW.Paziente);
```

La variabile @reddito_annuo contiene il reddito annuo
del paziente del quale si deve inserire una visita.

Azione

[...] se essa è mutuata, impostare l'attributo *Ticket* in base alle fasce di reddito annue

- ticket pari a euro 36.15 se reddito fra euro 0 ed euro 15,000
- ticket pari a euro 45.25 se reddito fra euro 15,000 ed euro 25,000
- ticket pari a 50.00 euro se reddito oltre 25,000 euro.

Se la visita non è mutuata, inserire NULL.

```
IF NEW.Mutuata IS TRUE THEN
    IF @reddito_annuo BETWEEN 0 AND 14999 THEN
        SET NEW.Ticket = 36.15;
    ELSEIF @reddito_annuo BETWEEN 15000 AND 25000 THEN
        SET NEW.Ticket = 45.25;
    ELSE
        SET NEW.Ticket = 50.00;
    END IF;
ELSE
    SET NEW.Ticket = NULL;
END IF;
```

Trigger

```
1  DELIMITER $$  
2  CREATE TRIGGER ImpostaTicket  
3  BEFORE INSERT ON Visita  
4  FOR EACH ROW  
5  
6  BEGIN  
7      /* variabile contenente il reddito annuo del paziente */  
8      SET @redditoAnnuo = (SELECT Reddito*12  
9                          FROM Paziente  
10                         WHERE CodFiscale = NEW.Paziente  
11                         );  
12  
13      /* controllo della fascia di reddito e settaggio del ticket*/  
14      IF (NEW.Mutuata IS TRUE) THEN  
15          IF (@redditoAnnuo BETWEEN 0 AND 14999) THEN  
16              SET NEW.Ticket = 36.15;  
17          ELSEIF (@redditoAnnuo BETWEEN 15000 AND 25000) THEN  
18              SET NEW.Ticket = 45.25;  
19          ELSE  
20              SET NEW.Ticket = 50.00;  
21          END IF;  
22      ELSE  
23          SET NEW.Ticket = NULL;  
24      END IF;  
25  
26  END $$  
27  
28  DELIMITER ;
```

Attributi ridondanti



Sono attributi il cui valore è **ricavabile** da altri attributi di tabelle del database

ridondante significa "in più", cioè se ne può anche fare a meno, non si perde informazione

Esempio

Aggiungere un attributo ridondante alla tabella Paziente che contenga il numero di visite mutuate effettuate

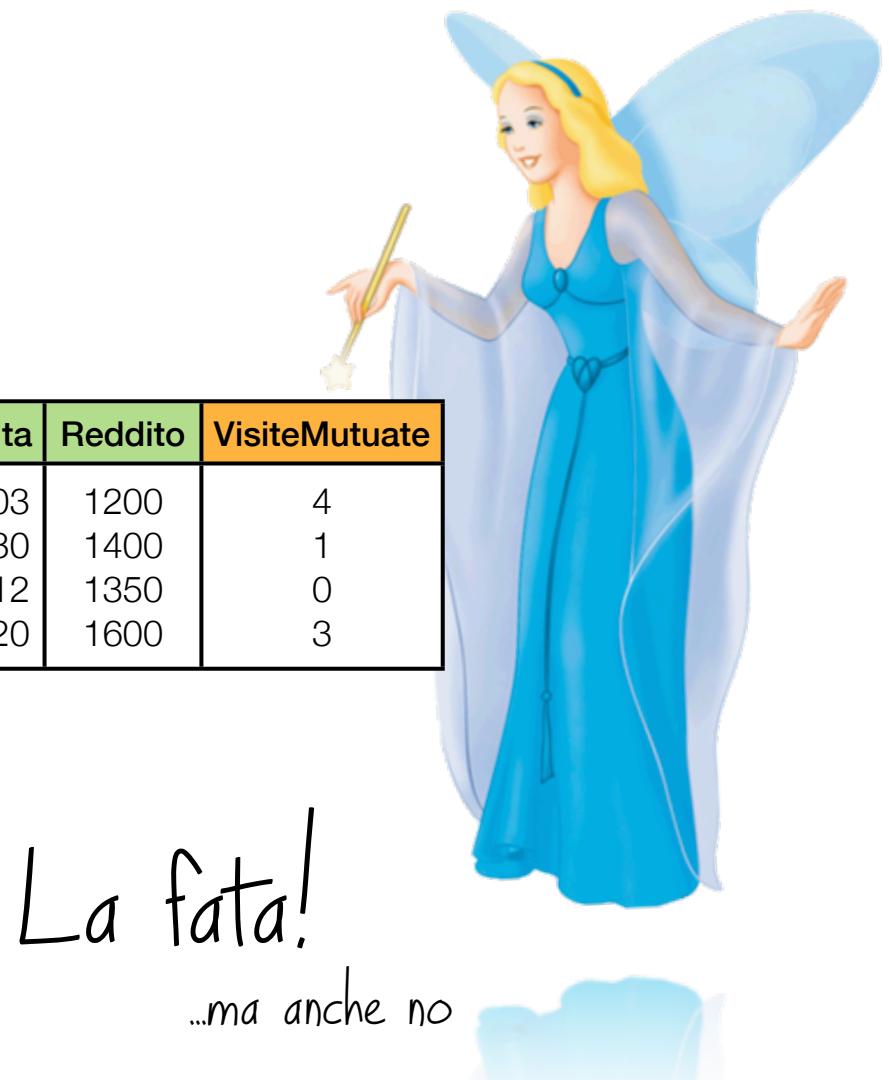
```
ALTER TABLE Paziente  
    ADD COLUMN VisiteMutuate INTEGER NOT NULL DEFAULT 0;  
  
UPDATE Paziente P  
SET VisiteMutuate = (  
    SELECT COUNT(*)  
    FROM Visita V  
    WHERE V.Paziente = P.CodFiscale  
    AND V.Mutuata IS TRUE  
);
```

è tutto finito?

Chi lo aggiorna l'attributo?



PAZIENTE							
CodFiscale	Cognome	Nome	Sesso	Citta	DataNascita	Reddito	VisiteMutuate
LPRNTA	Lepre	Antonella	F	Pisa	1958-05-03	1200	4
GTTRTA	Gatto	Rita	F	Firenze	1983-10-30	1400	1
MNZMBT	Manzi	Umberto	M	Pisa	1949-07-12	1350	0
CPRLND	Capra	Leonardo	M	Milano	1967-09-20	1600	3



Esempio di gestione di una ridondanza

Implementare il trigger che mantiene aggiornato l'attributo ridondante nella tabella Paziente, contenente il numero di vistite mutuate effettuate.

Soluzione

Implementare il trigger che mantiene aggiornato l'attributo ridondante nella tabella Paziente, contenente il numero di visite mutuate effettuate.

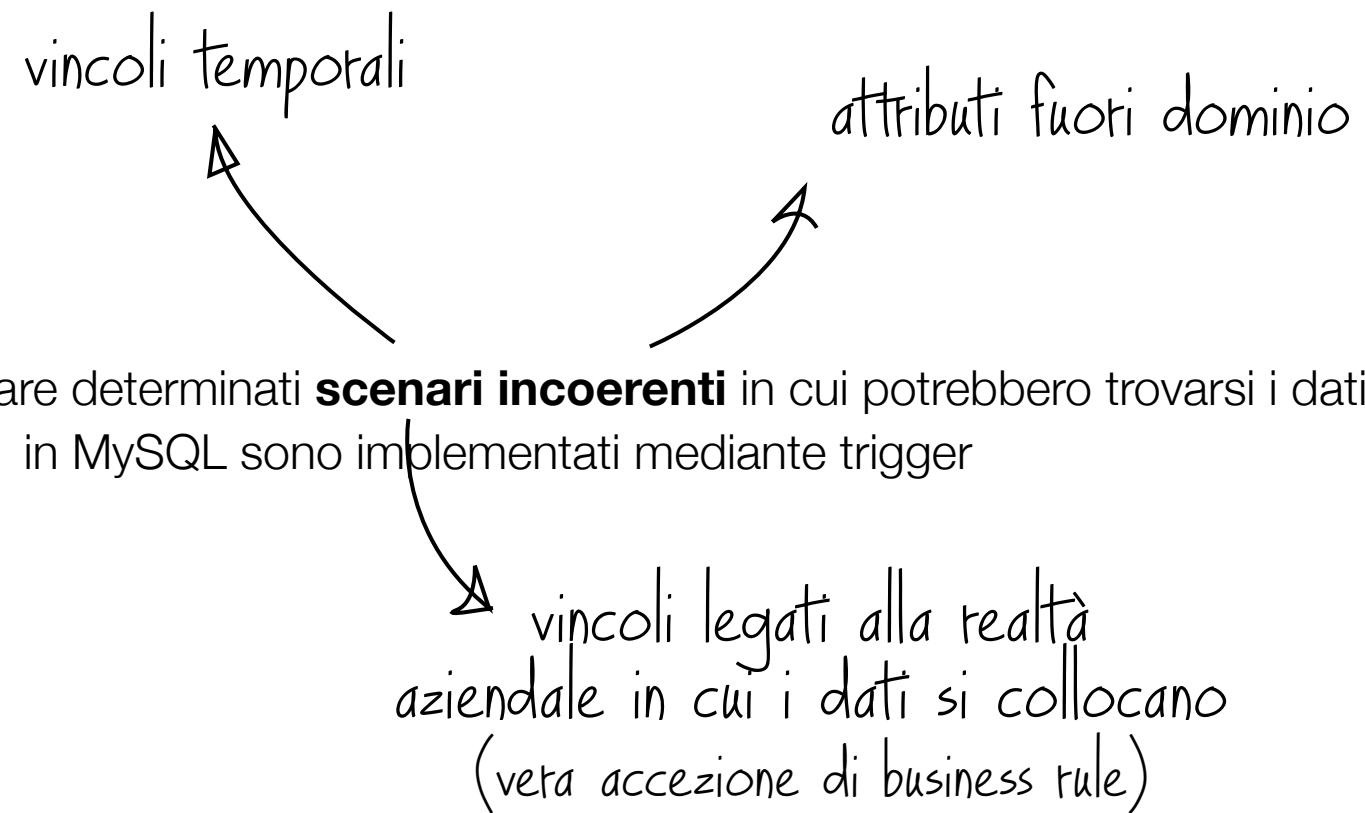
```
1  DELIMITER $$  
2  
3  CREATE TRIGGER AggiornaVisiteMutuate  
4  AFTER INSERT ON Visita  
5  FOR EACH ROW  
6  BEGIN  
7    IF NEW.Mutuata IS TRUE THEN  
8      UPDATE Paziente  
9      SET VisiteMutuate = VisiteMutuate + 1  
10     WHERE CodFiscale = NEW.Paziente;  
11    END IF;  
12  END $$  
13  
14  DELIMITER ;
```

Il trigger gestisce la ridondanza per sempre

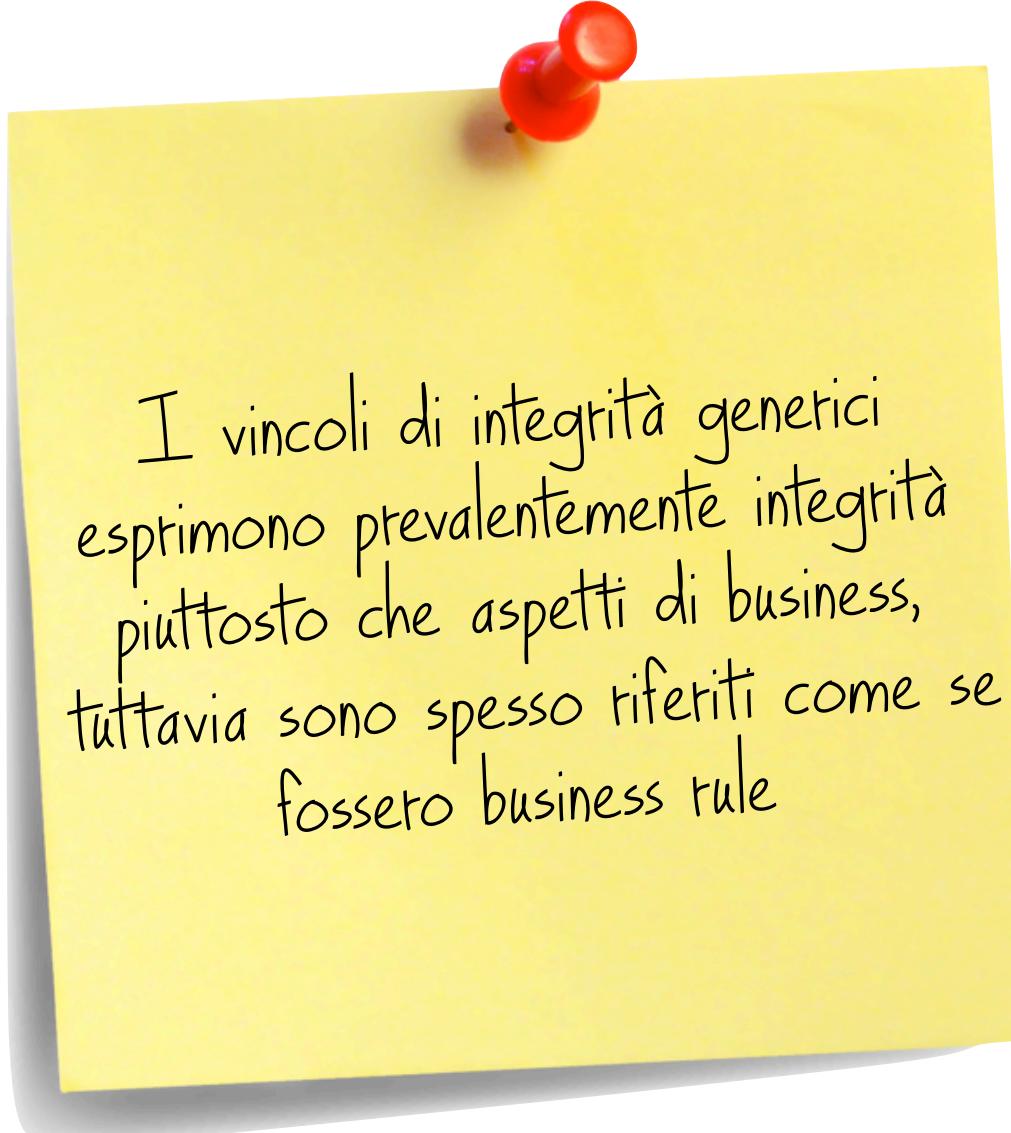
Business rule



Business rule (e vincoli di integrità generici)



Finezze...



Esempi di vincolo di integrità generico

Le matricole dei medici iniziano per M
e sono fatte da 4 cifre

Il codice fiscale deve essere un codice fiscale valido

Non si possono inserire visite antecedenti
alla data di inserimento di un medico nel database

Esempi di business rule

Il numero di visite per paziente
non può superare 5

Ogni mese, le visite non mutuate di un medico non
devono superare quelle mutuate.

Non si accettano medici con parcella
superiore a una certa parcella massima

I pazienti devono essere maggiorenni

Esempio di business rule

Ogni mese, le visite non mutuate di un medico non devono superare quelle mutuate.

```
1  DROP TRIGGER IF EXISTS checkValiditaVisita;
2
3  DELIMITER $$ 
4  CREATE TRIGGER checkValiditaVisita BEFORE INSERT ON Visita
5  FOR EACH ROW
6  BEGIN
7      DECLARE visite_mutuate_mese INTEGER DEFAULT 0;
8      DECLARE visite_non_mutuate_mese INTEGER DEFAULT 0;
9
10     SET visite_mutuate_mese =
11         ( SELECT COUNT(*) + IF(NEW.Mutuata = 1, 1, 0)
12             FROM Visita V
13             WHERE V.Medico = NEW.Medico
14                 AND V.Mutuata = 1
15                 AND MONTH(`Data`) = MONTH(CURRENT_DATE)
16                 AND YEAR(`Data`) = YEAR(CURRENT_DATE)
17         );
18
19     SET visite_non_mutuate_mese =
20         ( SELECT COUNT(*) - visite_mutuate_mese + IF(NEW.Mutuata = 0, 1, 0)
21             FROM Visita V
22             WHERE V.Medico = NEW.Medico
23                 AND MONTH(`Data`) = MONTH(CURRENT_DATE)
24                 AND YEAR(`Data`) = YEAR(CURRENT_DATE)
25
26     ▼ IF visite_non_mutuate_mese >= visite_mutuate_mese THEN
27         SIGNAL SQLSTATE '45000'
28         SET MESSAGE_TEXT = 'Limite massimo visite mutuate superato';
29     ▲ END IF;
30
31 END $$ 
32 DELIMITER ;
```

Event



Event

Sono trigger la cui condizione di scatto è **dettata dal tempo**



possono anche essere ricorrenti



Esempio

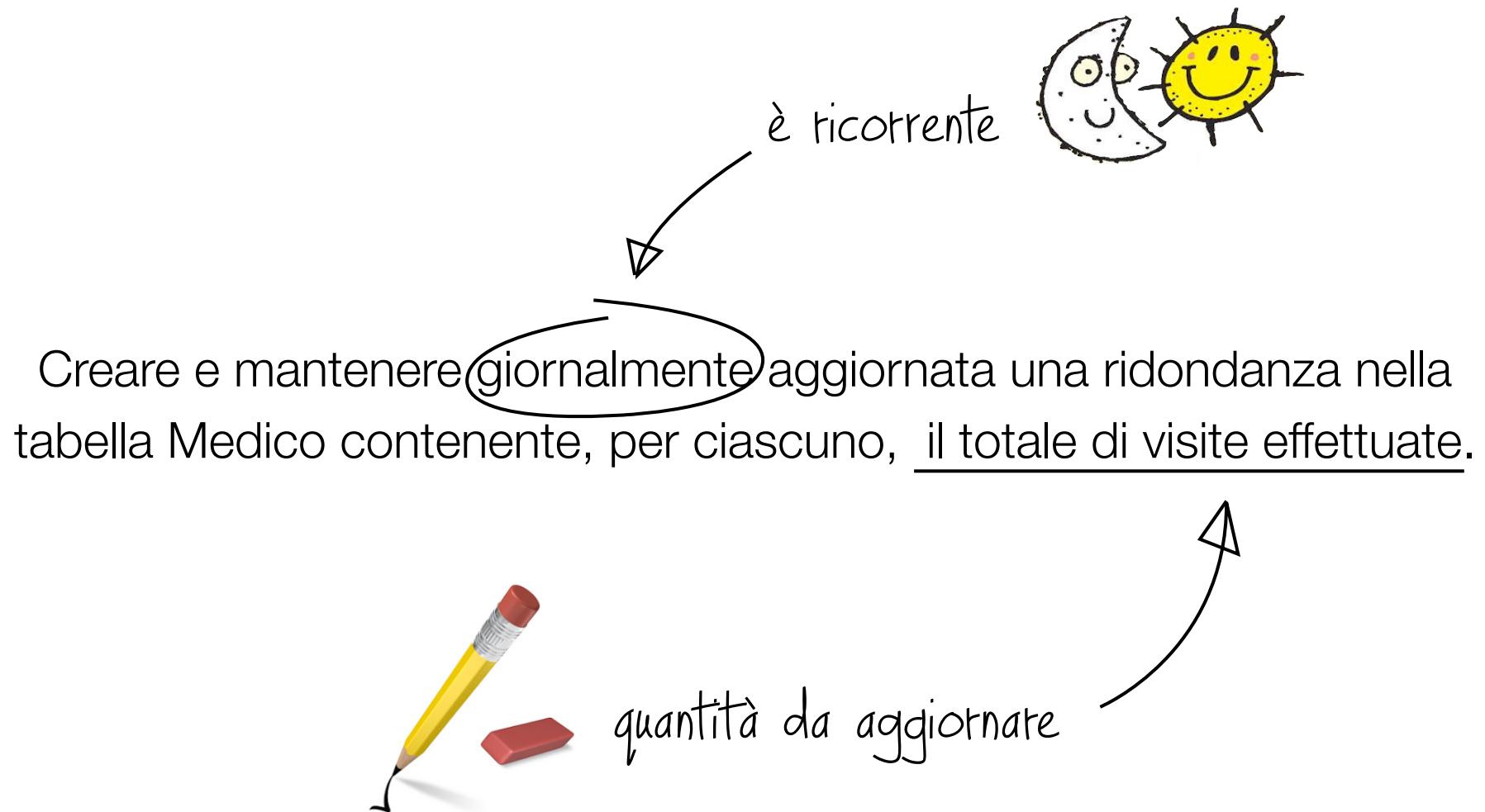


Tabella Medico con ridondanza

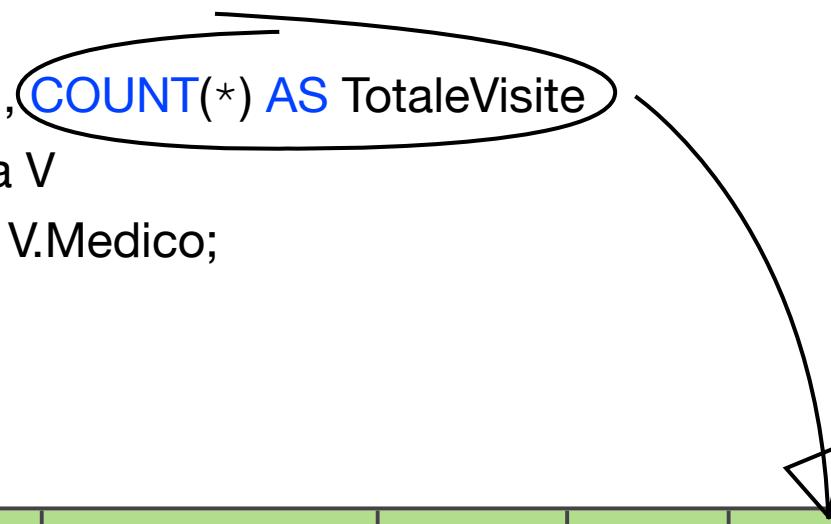
MEDICO

 Matricola	Cognome	Nome	Specializzazione	Parcella	Citta	TotaleVisite
18339	Verdi	Paolo	Ortopedia	150	Pisa	135
35512	Rossi	Marta	Ortopedia	120	Pisa	62
16220	Gialli	Rita	Cardiologia	135	Roma	97
29858	Neri	Rino	Dermatologia	110	Siena	211

è ridondante perché si può
ottenere dai dati!!!

Siamo sicuri che è una ridondanza???

```
SELECT V.* , COUNT(*) AS TotaleVisite  
FROM Visita V  
GROUP BY V.Medico;
```



Matricola	Cognome	Nome	Specializzazione	Parcella	Città	TotaleVisite
18339	Verdi	Paolo	Ortopedia	150	Pisa	135
35512	Rossi	Marta	Ortopedia	120	Pisa	62
16220	Gialli	Rita	Cardiologia	135	Roma	97
29858	Neri	Rino	Dermatologia	110	Siena	211

C'è da fidarsi del valore?

La ridondanza deve contenere una **copia conforme e aggiornata** di un'informazione già deducibile dai dati



non devo chiamare la maga!

Come faccio a fidarmi?



Aggiunta della ridondanza e assegnamento

```
ALTER TABLE Medico  
ADD COLUMN TotaleVisite INT DEFAULT 0 AFTER Citta;
```

```
UPDATE Medico  
SET TotaleVisite = (SELECT COUNT(*)  
                     FROM Visita V  
                     WHERE V.Medico = Matricola);
```



all'inserimento di una nuova visita, la ridondanza non è più aggiornata

Events do it better!



Eseguito ogni giorno, a partire dalla creazione.

```
CREATE EVENT AggiornaTotaliVisite  
ON SCHEDULE EVERY 1 DAY  
DO (aggiorna la tabella Medico  
incrementando il numero totale di visite  
ai soli medici che hanno visitato in data odierna  
);
```

Azione

Creare e mantenere giornalmente aggiornata una ridondanza nella tabella Medico contenente, per ciascuno, il totale di visite effettuate.

1. Trovate i medici che hanno visitato oggi
2. Incrementare TotaleVisite ai soli medici in #1

Event (giornaliero)

```
CREATE EVENT AggiornaTotaliVisite  
ON SCHEDULE EVERY 1 DAY  
DO  
    UPDATE Medico  
    SET TotaleVisite = TotaleVisite + (SELECT COUNT(*)  
        FROM Visita V  
        WHERE V.Medico = Matricola AND  
              V.Data = CURRENT_DATE);
```



Non oltre la mezzanotte!!!

Soluzione corretta

CREATE EVENT AggiornaTotaliVisite

ON SCHEDULE EVERY 1 DAY

periodicità

STARTS '2016-05-22 23:55:00'

prima esecuzione

DO

UPDATE Medico

SET TotaleVisite = TotaleVisite + (SELECT COUNT(*)

FROM Visita V

WHERE V.Medico = Matricola AND

V.Data = CURRENT_DATE);

Event: sintassi MySQL

ON SCHEDULE AT 'data_ora' + INTERVAL numero DAY | MONTH | YEAR |
SECOND | MINUTE | HOUR

[ON COMPLETION PRESERVE]

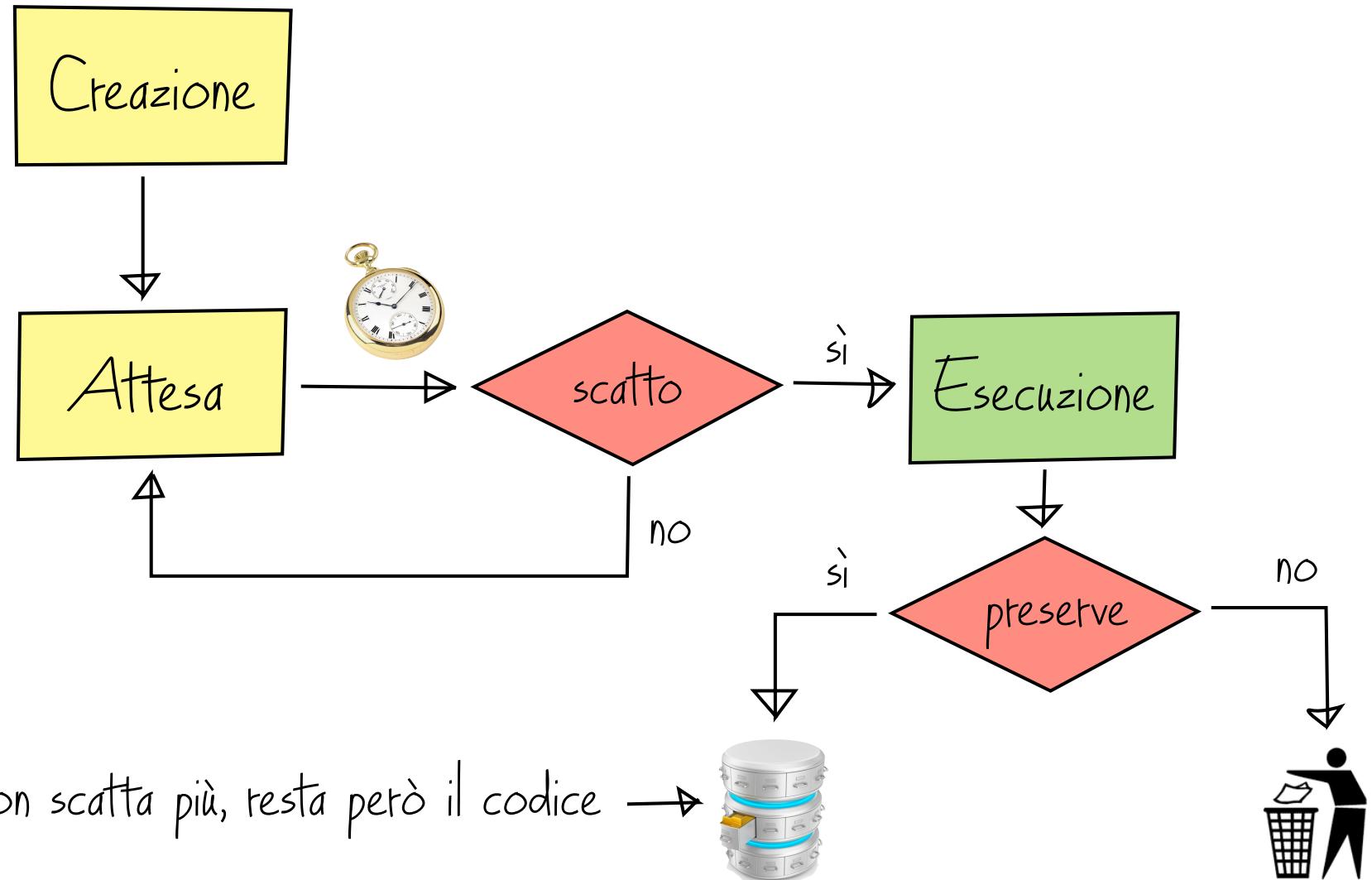


il risultato di questa espressione rappresenta
l'istante in cui il temporal trigger va in esecuzione

once upon
a time...



Event singolo scatto: ciclo di vita



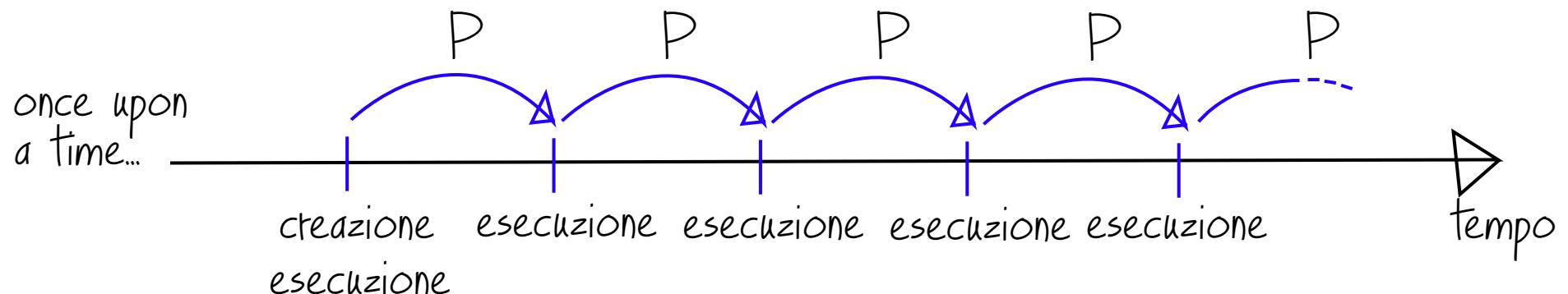
Dopo lo scatto...niente



Recurring event: sintassi MySQL

ON SCHEDULE **EVERY** numero DAY | MONTH | YEAR | SECOND | MINUTE | HOUR

periodicità P con cui avviene l'esecuzione



...it goes on, and on, and on



'cause we're all chained to the rhythm...

Varianti

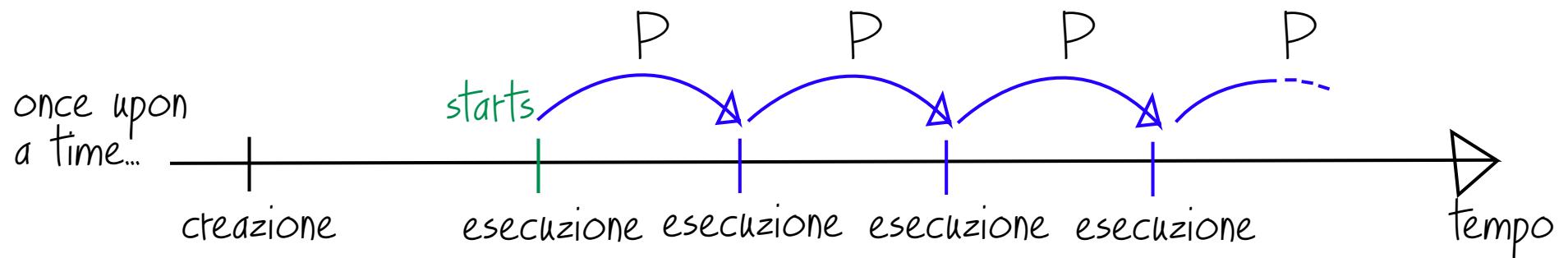
È possibile stabilire sia un istante di **inizio** che un istante di **fine**



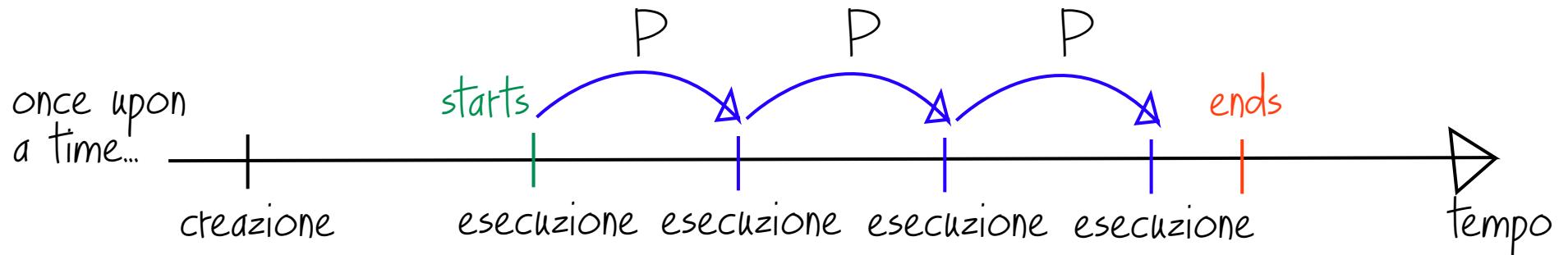
se non si specificano, il recurring event
scatta immediatamente, e poi segue la periodicità

Varianti starts e starts-ends

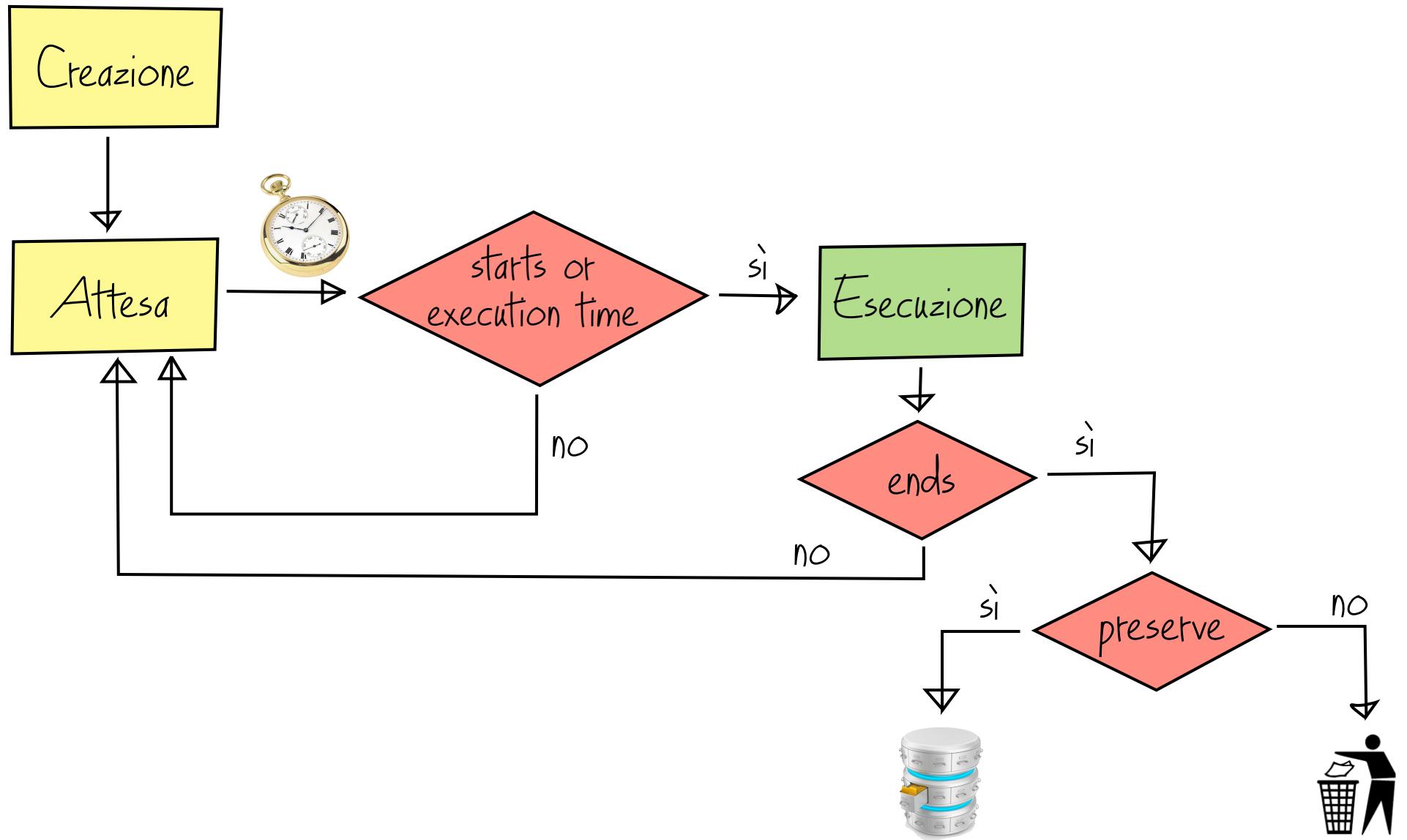
ON SCHEDULE EVERY numero DAY | MONTH | YEAR | SECOND | MINUTE | HOUR
STARTS 'data_ora'



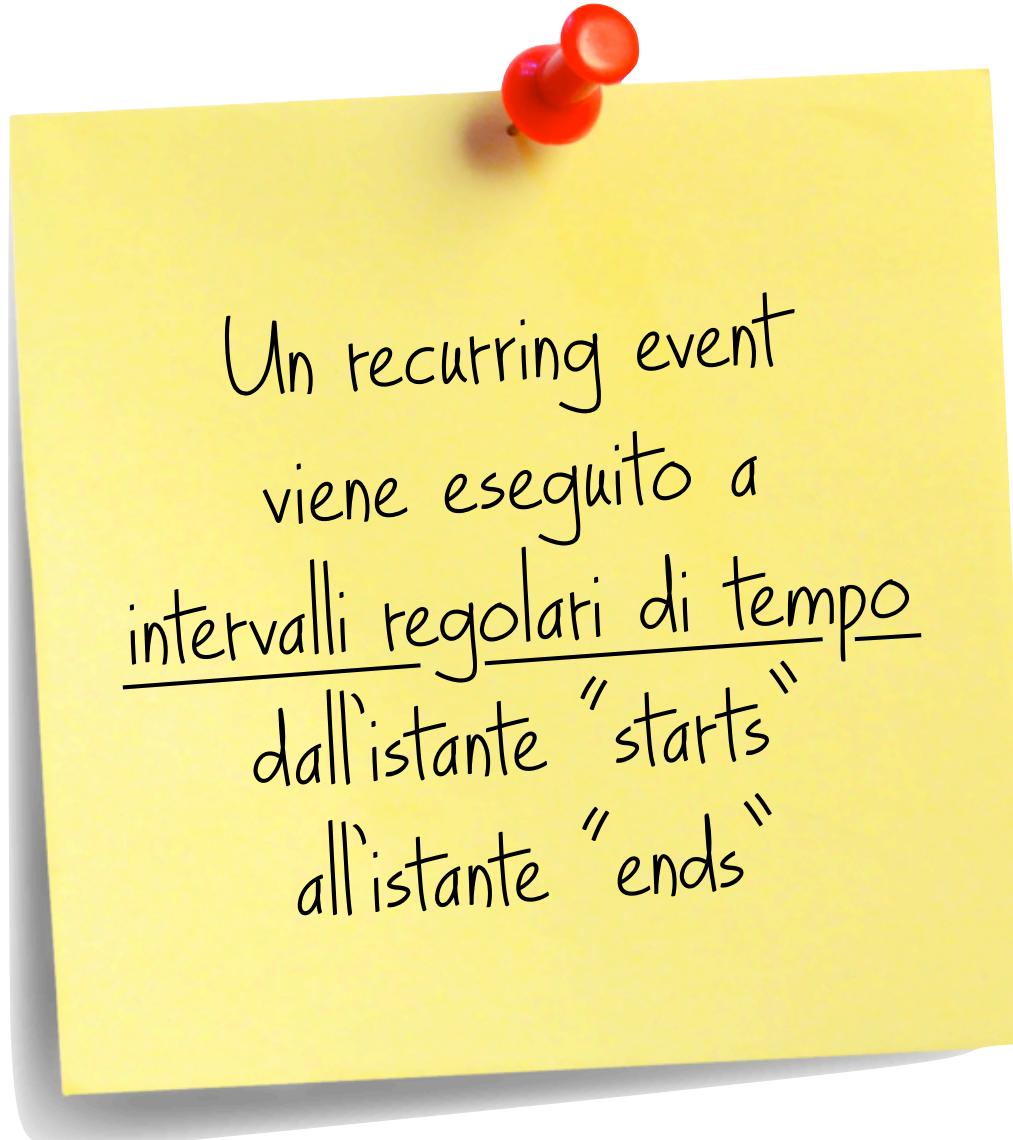
ON SCHEDULE EVERY numero DAY | MONTH | YEAR | SECOND | MINUTE | HOUR
STARTS 'data_ora' **ENDS** 'data_ora'



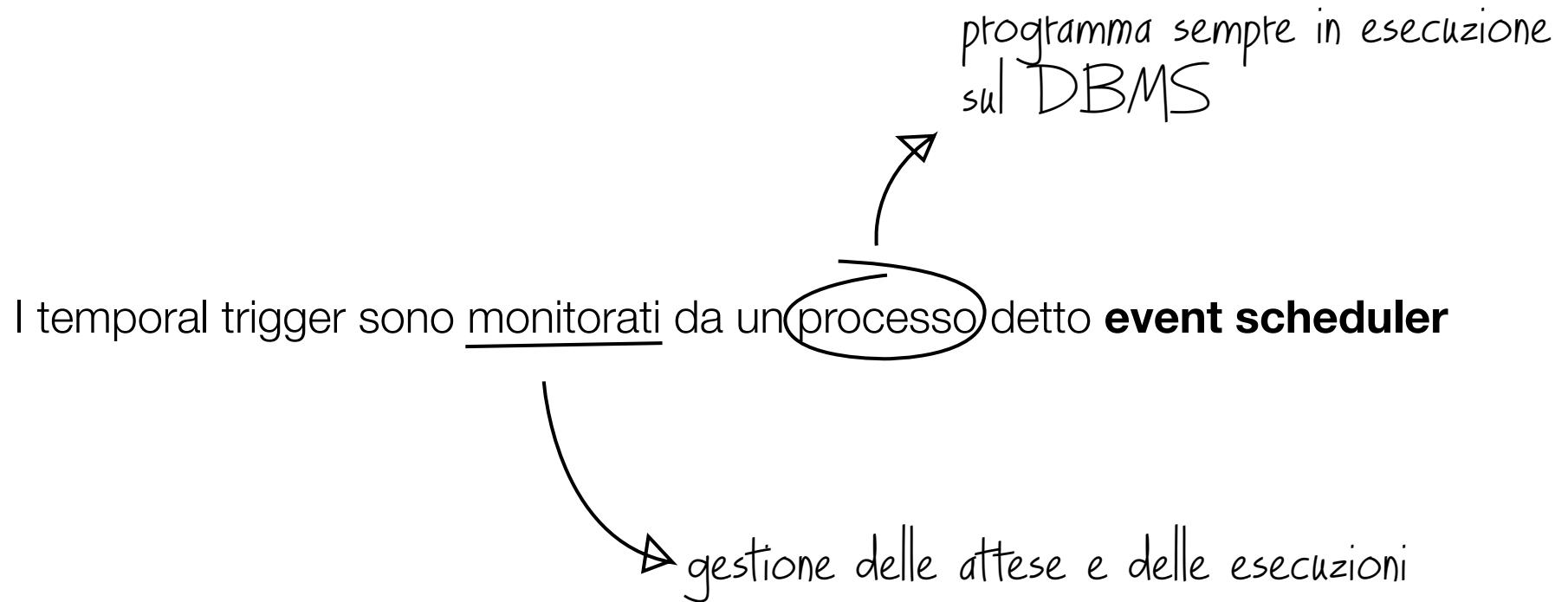
Recurring event: ciclo di vita



Dopo lo scatto... continua

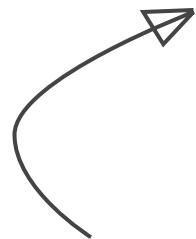


Schedulazione



Attivazione dello scheduler

SET GLOBAL event_scheduler = ON;



Settando questa variabile di sistema
parte la schedulazione degli event

Esercizio (realtà di progetto a.a. 2013-2014, non trattato nell'a.a. 2016-2017)

Siano date le tabelle Profilo e Post definite dai seguenti schemi:

Profilo (IDProfilo, Nome, Cognome, Rimossi)

Post (IDPost, Autore, DataInserimento, DataUltimaModifica)

Supporre che esista un trigger SalvaPost che alla rimozione di un post da parte dell'utente, lo salvi in una tabella PostRimossi, impostando la data di ultima modifica alla data di rimozione, incrementando, inoltre, l'attributo (ridondante) Rimossi nella tabella Profilo.

Scrivere un temporal trigger che, da domani mattina, ogni settimana alle ore 3:00 del mattino, archivi i post inseriti/ricevuti la settimana scorsa, in una tabella PostPrecedenti, per i soli utenti del social network che ne hanno pubblicati più di 100 nella stessa settimana. Modificare infine il trigger SalvaPost, cancellandolo e ricreandolo, per renderne il comportamento coerente con la nuova realtà.

Aggiornamento trigger SalvaPost



Evento

⇒ Rimozione post dalla tabella Post



Condizione

⇒ Rimozione effettuata dall'utente



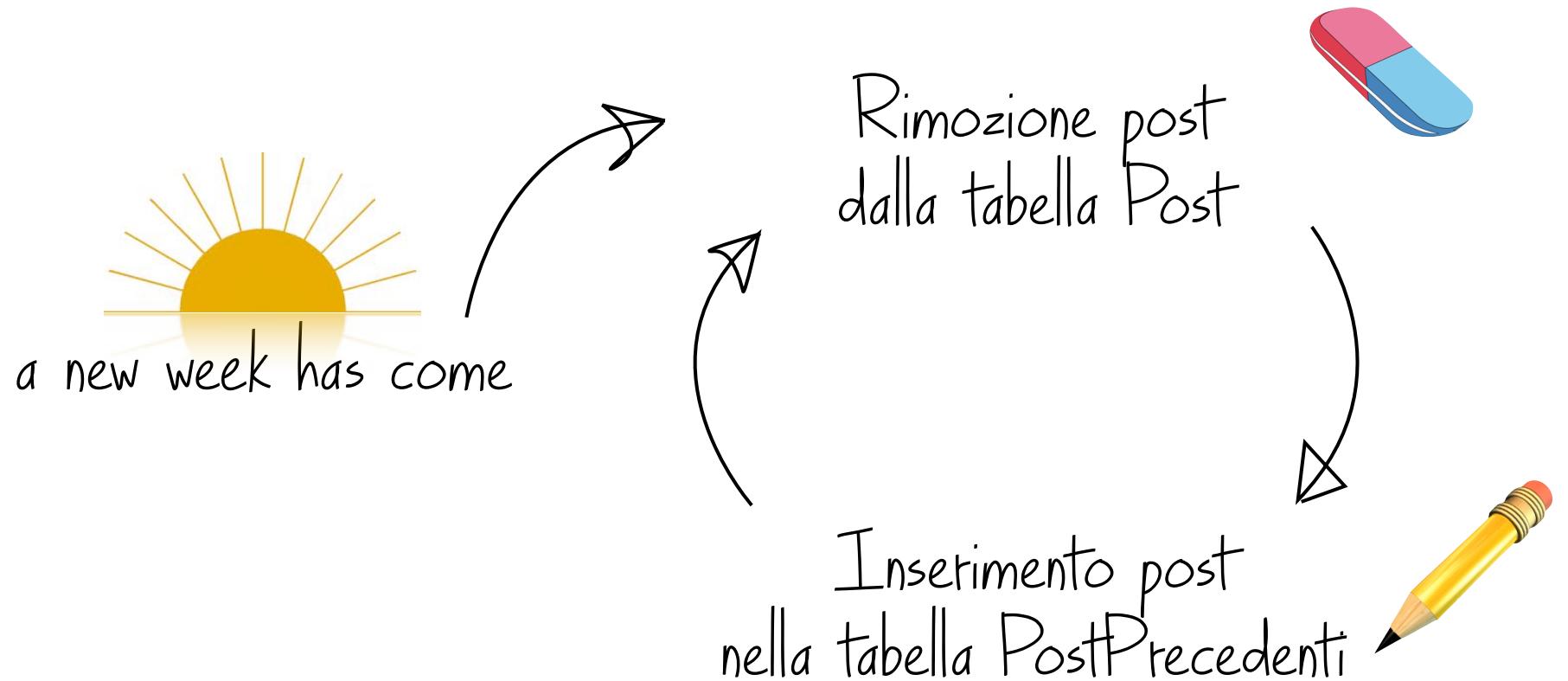
Azione

⇒ Spostare il post nella tabella
PostRimossi e incrementare
l'attributo Rimossi di Profilo

Trigger SalvaPost

```
1  DROP TRIGGER SalvaPost;
2
3  DELIMITER $$ 
4  CREATE TRIGGER SalvaPost AFTER DELETE ON Post
5  FOR EACH ROW
6  BEGIN
7      DECLARE spostato INT DEFAULT 0;
8
9      SELECT COUNT(*) INTO spostato
10     FROM PostPrecedenti PP
11    WHERE PP.IDPost = OLD.IDPost;
12
13     -- post rimosso dall'utente
14  IF spostato = 0 THEN
15      INSERT INTO PostRimossi
16      VALUES (OLD.IDPost, OLD.Autore, OLD.DataInserimento,
17              CURRENT_DATE);
18
19      UPDATE Profilo
20      SET Rimossi = Rimossi + 1;
21  END IF;
22 END $$
```

Funzionamento dell'event



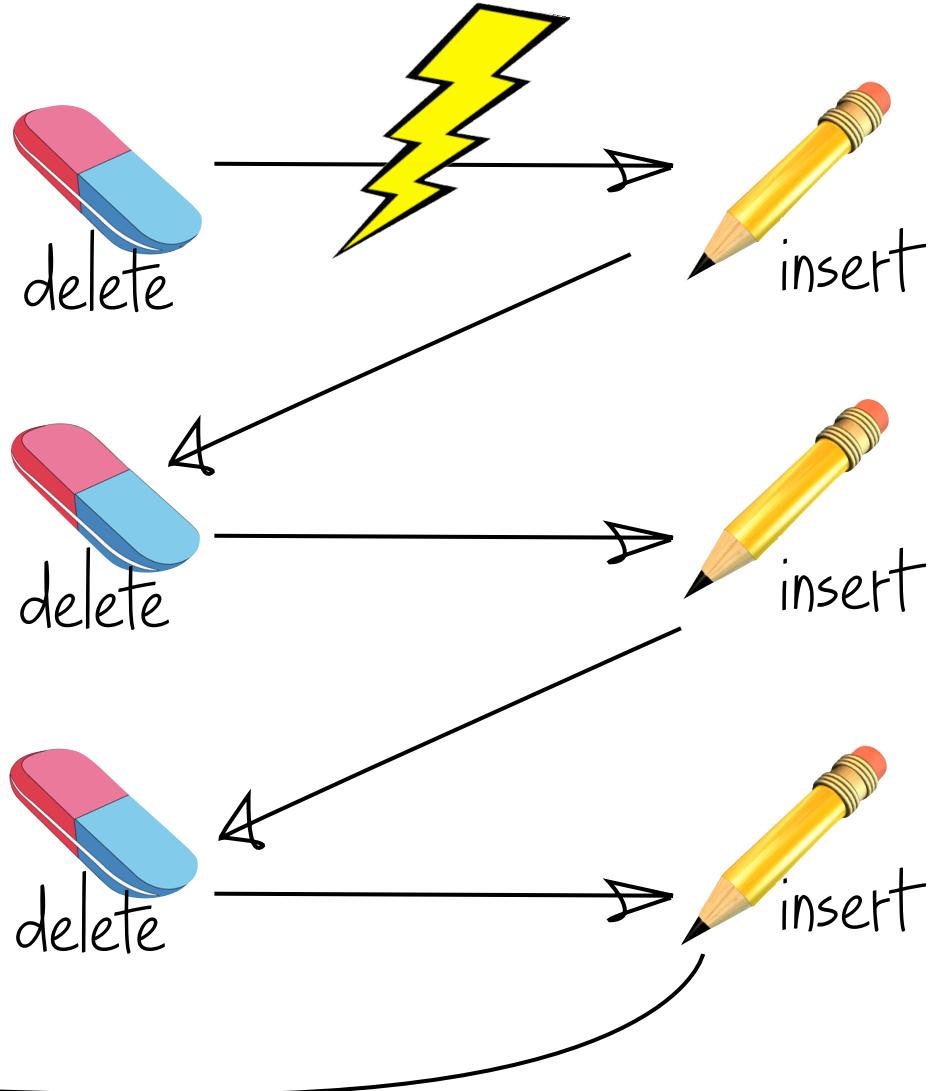
Event ArchiviaPost

```
24 CREATE EVENT ArchiviaPost
25 ON SCHEDULE EVERY 1 WEEK
26 STARTS STR_TO_DATE(DATE_FORMAT(CURRENT_DATE,'%Y%m%d 0300'),'%Y%m%d %H%i')
27     + INTERVAL 1 DAY
28 DO
29 BEGIN
30     -- creazione temporary table per i post da spostare
31     CREATE TEMPORARY TABLE IF NOT EXISTS _PostDaSpostare LIKE Post;
32     TRUNCATE TABLE _PostDaSpostare;
33     -- inserimento dei post da spostare nella temporary table
34     INSERT INTO _PostDaSpostare
35     SELECT *
36     FROM Post P
37     WHERE WEEKOFYEAR(P.DataInserimento) = WEEKOFYEAR(CURRENT_DATE) - 1
38     AND P.Autore IN(
39         SELECT P0.Autore
40         FROM Post P0
41         WHERE WEEKOFYEAR(P0.DataInserimento) = WEEKOFYEAR(CURRENT_DATE) - 1
42         GROUP BY P0.Autore
43         HAVING COUNT(*) > 100
44     );
45     -- rimozione dalla tabella Post dei post archiviati
46     DELETE FROM Post
47     WHERE IDPost IN (SELECT * FROM _PostDaSpostare);
48 END $$
```

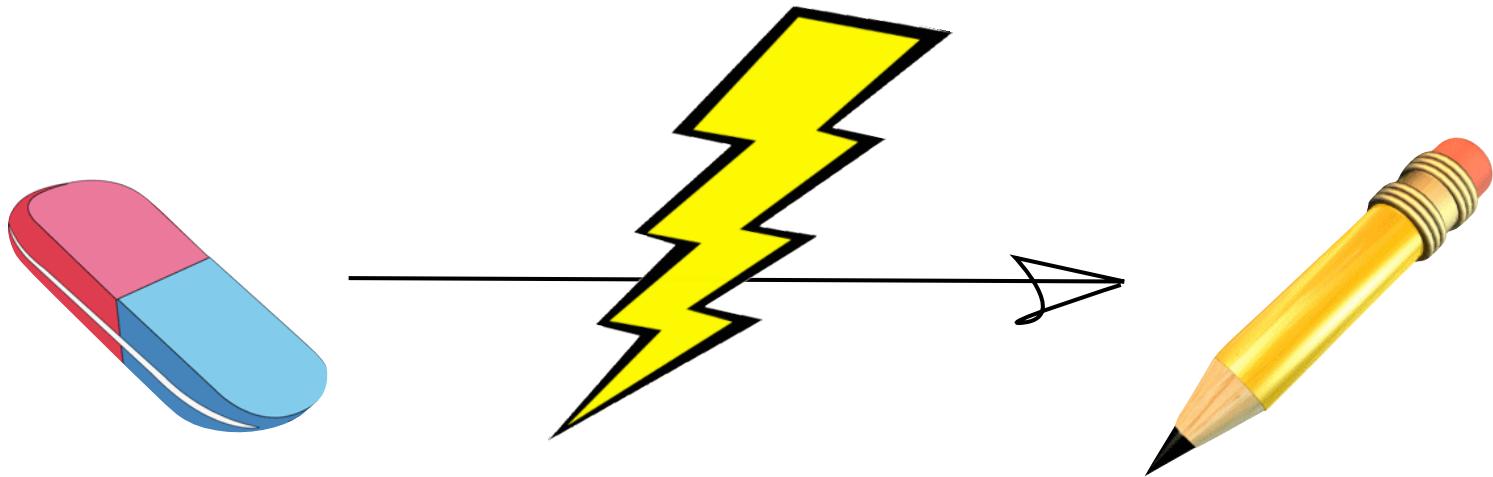
49 DELIMITER ;

Considerazioni sulla transazionalità

3 post da spostare



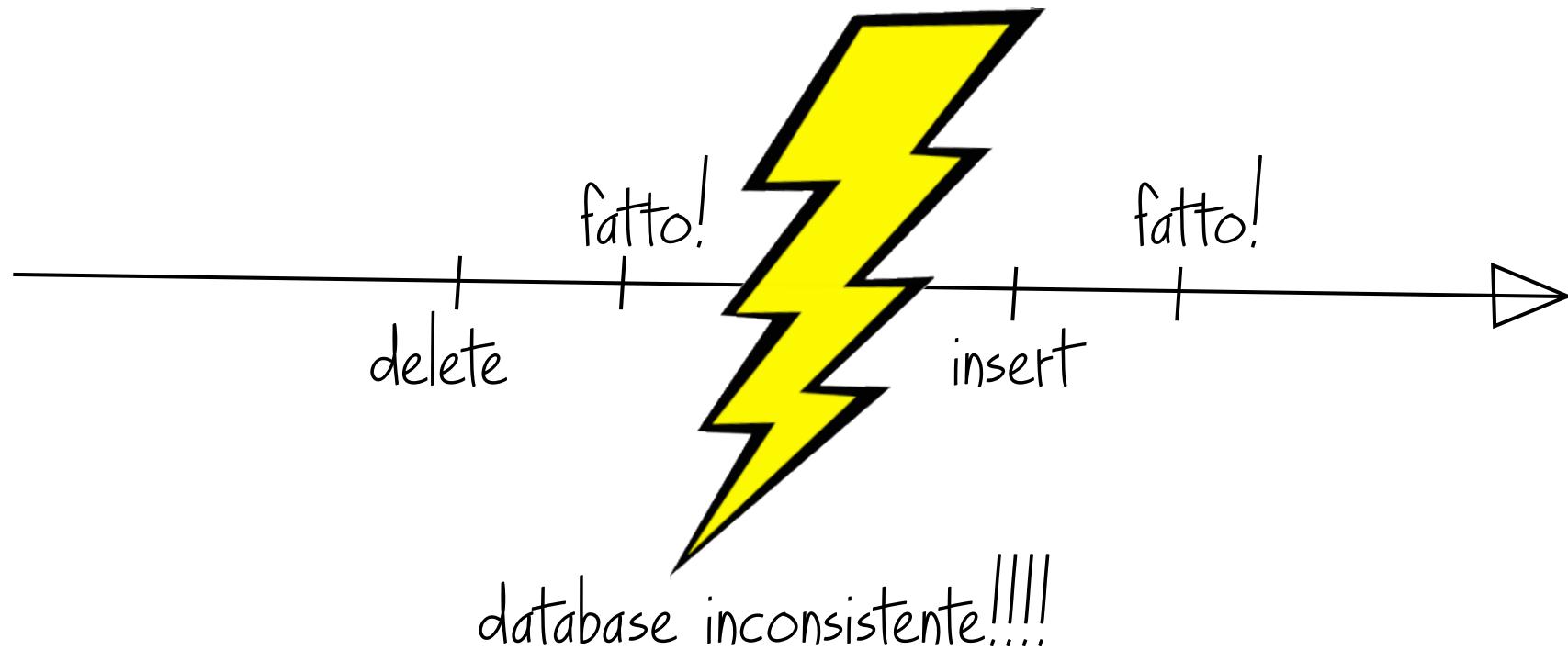
Oh shit!



delete from Post
where IDPost = 'xxx'

insert into PostRimossi
values ('xxx...')

Nel tempo



Trigger, event e transazionalità



Esercizi

Qui di seguito trovate gli esercizi relativi a **trigger** ed **event**.

1. Modificare la tabella ESORDIO aggiungendo un attributo EsordiPrecedenti contenente il numero di esordi precedenti di patologie dello stesso paziente relative allo stesso settore medico, curati con successo. L'attributo EsordiPrecenti deve essere aggiornato in modalità deferred, ogni cinque giorni, a partire dal 1° Luglio 2015, sempre alle ore 6:00 del mattino.
2. Scrivere un trigger che impedisca l'inserimento di due terapie consecutive per lo stesso paziente, caratterizzate dallo stesso farmaco, la più recente delle quali con una posologia superiore al doppio rispetto alla precedente.
3. Creare una business rule che permetta di inserire un nuovo farmaco *F* e le relative indicazioni, qualora non vi siano già più di due farmaci di cui almeno uno basato sullo stesso principio attivo, aventi, ciascuno, un'indicazione per una stessa patologia per la quale *F* è indicato. Supporre che per prima cosa sia inserito il farmaco, dopodiché siano inserite le varie indicazioni.

Esercizi (cont.)

4. Creare una business rule per impedire che un medico possa visitare mensilmente più di due volte lo stesso paziente, qualora all'atto delle due visite già effettuate in un dato mese dal medico sul paziente, quest'ultimo non fosse affatto da alcuna patologia.
5. Scrivere un event che sconti mensilmente del 2% i farmaci che sono stati assunti in meno del 10% delle terapie iniziate nel mese precedente. Nel caso in cui un farmaco venga scontato, mantenere nella tabella Farmaco anche il prezzo originario.
6. Scrivere un event che impedisca alle terapie in corso di protrarsi oltre un mese qualora, durante la terapia, il paziente abbia contratto al massimo tre patologie di cui almeno una a carico della parte del corpo oggetto della terapia. Spostare tali terapie in una tabella TERAPIEINTERROTTE avente lo schema (NomePaziente, CognomePaziente, Farmaco, DataInizio, DataInterruzione).