Wyzwalacze

- Uruchamiane przez zajście określonego zdarzenia w bazie danych (na tabeli, perspektywie, schemacie lub całej bazie danych).
- Cele stosowania:
  - wymuszanie złożonych reguł biznesowych,
  - zaawansowane śledzenie działań użytkowników,
  - wymuszanie złożonych polityk bezpieczeństwa,
  - wypełnianie atrybutów tabeli wartościami domyślnymi,
  - modyfikacja złożonych perspektyw

Wyzwalacz związany jest przeważnie z tabelą, czasami z perspektywą

Może wykonywać inne procedury, ale także posiadać własne polecenia

Nie może zawierać i wywoływać procedur zawierających polecenia: COMMIT, ROLLBACK I SAVEPOINT

Wyzwalacze - tworzenie

CREATE [OR REPLACE] TRIGGER nazwa
<moment uruchomienia>
<zdarzenie uruchamiające>
[**WHEN** warunek]
[**FOR EACH ROW**]
[**DECLARE** <deklaracje stałych, zmiennych, kursorów>]
BEGIN
<ciało procedury wyzwalanej>
END;

Parametry wyzwalaczy

Zdarzenie uruchamiające:

– polecenie DML: INSERT, UPDATE, DELETE,

– polecenie DDL: CREATE, ALTER,

– zdarzenie w bazie danych zalogowanie/wylogowanie użytkownika, błąd, uruchomienie/zatrzymanie bazy danych

Parametry wyzwalaczy

Moment uruchomienia:

– **BEFORE -** wyzwalacze utworzone ze słowem BEFORE uruchamiają się przed wykonaniem określonego polecenia na wybranej tabeli

– **AFTER -** wyzwalacze utworzone ze słowem AFTER uruchamiają się po wykonaniu określonego polecenia na wybranej tabeli

– **INSTEAD OF** – powoduje, że polecenia wyzwalacza wykonywane są zamiast zdarzenia wyzwalającego

Warunek umieszczony po frazie **WHEN** oznacza uruchomienie wyzwalacza jedynie wtedy, gdy zostanie spełnione wyrażenie występujące w warunku

Parametry wyzwalaczy

- Wyzwalacze instrukcji DML ze słowami **BEFORE** oraz **AFTER** dotyczą instrukcji **INSERT, UPDATE i DELETE** odwołujących się do tabeli bazy danych
- Wyzwalacze utworzone z fazą **INSTEAD OF** mogą odwoływać się jedynie do perspektyw bazy
- Jeżeli wyzwalacz ma odwoływać się do operacji wykonywanej na tabeli, wtedy w jego definicji, w miejscu, gdzie znajduje się zdarzenie wyzwalające, należy umieścić frazę:

  ON nazwa_tabeli

- Jeżeli wyzwalacz ma dotyczyć konkretnego atrybutu tabeli, powinno się zaznaczyć pisząc:

  OF nazwa_atrybutu ON nazwa_tabeli

Parametry wyzwalaczy

Częstotliwość uruchamiania:

- wyzwalacz wierszowy - jednokrotnie dla każdego rekordu, przetworzonego przez polecenie
- wyzwalacz polecenia - jednokrotnie dla polecenia

Wyzwalacze - tworzenie

CREATE [OR REPLACE] TRIGGER nazwa
BEFORE | AFTER | INSTEAD OF
<zdarzenie uruchamiające>
[**WHEN** warunek]
[**FOR EACH ROW**]
[**DECLARE** <deklaracje stałych, zmiennych, kursorów>]
BEGIN
<ciało procedury wyzwalanej>
END;

Wyzwalacz polecenia

- Uruchamiany jednokrotnie dla polecenia
- Nie może bezpośrednio odwoływać się do atrybutów tabeli (perspektywy) wyzwalacza

Przykład:

wyzwalacz uruchamiany jednokrotnie po wykonaniu polecenia INSERT na tabeli KLIENCI

```
CREATE TRIGGER Zapisz
AFTER INSERT ON klienci
BEGIN
INSERT INTO log (data, tabela, operacja)
VALUES(sysdate, ‘klienci', 'INSERT');
END;
```

śledzi operacje wstawiania rekordów do tabeli klienci

Wyzwalacz wierszowy

- Uruchamiany jednokrotnie dla każdego rekordu, przetworzonego przez polecenie
- Zawiera klauzulę **FOR EACH ROW**
- Nie może wykonywać zapytania ani żadnej operacji modyfikującej relację (perspektywę), na której założono wyzwalacz
- Może odwoływać się bezpośrednio do wartości atrybutów rekordu, dla którego został uruchomiony

Wyzwalacz wierszowy

Odwołanie do wartości atrybutów tabeli:

– :OLD.nazwa_atrybutu – sprzed wykonania polecenia

– :NEW.nazwa_atrybutu – po wykonaniu polecenia

	:OLD	:NEW
INSERT	wartość pusta	wartość wstawiona
UPDATE	wartość przed modyfikacją	wartość modyfikowana
DELETE	wartość z usuwanego rekordu	wartość pusta

Wyzwalacz wierszowy - przykład

- Uruchamiany dla każdego rekordu wstawianego przez polecenie INSERT do tabeli Samochody

```
CREATE TRIGGER WstawIdentyfikator
   BEFORE INSERT ON samochody
   FOR EACH ROW
BEGIN
   IF :NEW id_sam IS NULL THEN ...
   SELECT seq_samochody.nextval INTO :NEW id_sam
            FROM dual;
  END IF;
END;
```

Warunek uruchomienia wyzwalacza

- Postać WHEN (warunek logiczny)
- Przy przedrostkach NEW i OLD opuszczamy dwukropek

Przykład

```
CREATE OR REPLACE TRIGGER Wstawidentyfikator
   BEFORE INSERT ON Samochody
   FOR EACH ROW
   WHEN (NEW.id_sam IS NULL)
BEGIN
   SELECT seq_samochody.nextval INTO :NEW.id_sam
   FROM dual;
END;
```

Wyzwalacz dla wielu zdarzeń

- Uruchamiany przez kilka zdarzeń
- W ciele takiego wyzwalacza można selekcjonować kod, który ma być wykonany w przypadku wystąpienia określonego zdarzenia
- Używa się w tym celu zmiennych logicznych INSERTING, DELETING i UPDATING, które przyjmują wartość prawdy jeśli zdarzeniem uruchamiającym wyzwalacz jest odpowiednio zdarzenie INSERT, DELETE bądź UPDATE

```
CREATE OR REPLACE TRIGGER ZapamietajOperacje
  BEFORE INSERT OR UPDATE OR DELETE ON Samochody
  FOR EACH ROW
BEGIN
  IF INSERTING THEN ....
  ELSEIF DELETING THEN ...
  ELSEIF UPDATING (nazwa) THEN ...
  ENDIF;
END;
```

Wyzwalacz INSTEAD OF

- Definiowany tylko dla perspektyw
- Wykonywany zamiast polecenia, które uruchomiło wyzwalacz
- Stosowany najczęściej dla perspektyw złożonych celem zapewnienia ich modyfikowalności
- Nie można bezpośrednio odwoływać się do atrybutów perspektywy

Wyzwalacz INSTEAD OF

```
CREATE OR REPLACE VIEW LICZBA AS
   SELECT miejscowosc, COUNT(*) AS liczba_osob
FROM klienci
GROUP BY miejscowosc;
```

```
CREATE TRIGGER Wstaw
  INSTEAD OF INSERT ON klienci
  FOR EACH ROW
BEGIN
  INSERT INTO klienci(id_kli,miejscowosc)
  VALUES (seq_klienci.nextval,:NEW.miejscowosc);
END;
```

Wyzwalacze systemowe

Umożliwiają kontrolowanie następujących działań:

- Uruchamiania i zamykania serwera
- Błędów serwera
- Logowania się i wylogowywania przez użytkowników

CREATE [OR REPLACE] TRIGGER nazwa_wyzwalacza
BEFORE | AFTER zdarzenie_bazy_danych ON baza_danych | schemat
[DECLARE]
deklaracje;
BEGIN
wykonywane_instrukcje;
END;

Wyzwalacze systemowe

Wyzwalacze związane z logowaniem i wylogowywaniem umożliwiają śledzenie czasu trwania połączenia. Instrukcje związane z tymi wyzwalaczami znajdują się w pakiecie

USER _ CONNECTION

CONNECTING_TRIGGER

DISCONNECTING_TRIGGER

Wyzwalacze systemowe – śleczenie logowania i wylogowywania

```
CREATE OR REPLACE TRIGGER connecting_trigger
AFTER LOGON ON DATABASE
BEGIN
user_connection.connecting(sys.login_user);
END;
```

```
CREATE OR REPLACE TRIGGER disconnecting_trigger
BEFORE LOGOFF ON DATABASE
BEGIN
user_connection.disconnecting(sys.login_user);
END;
```

Tabeta connection_log

```
CREATE TABLE connection_log
(event_id   NUMBER(10),
event_user_name   VARCHAR2(30) CONSTRAINT log_event_nn1 NOT NULL,
event_type            VARCHAR2(14) CONSTRAINT log_event_nn2 NOT NULL,
event_date            DATE              CONSTRAINT log_event_nn3 NOT NULL,
CONSTRAINT connection_log_p1     PRIMARY KEY (event_id));
```

Specyfikacja i ciało pakietu user_connection

```
CREATE OR REPLACE PACKAGE  user_connection AS
PROCEDURE connecting       (user_name IN VARCHAR2);
PROCEDURE disconnecting  (user_name IN VARCHAR2);
END user_connection;
```

```
CREATE OR REPLACE PACKAGE BODY user_connection AS
PROCEDURE connecting (user_name IN VARCHAR2) IS
BEGIN
INSERT INTO connection_log(event_user_name, event_type, event_date)
VALUES (user_name,'CONNECT', SYSDATE);
END connecting;
PROCEDURE disconnecting (user_name IN VARCHAR2) IS
BEGIN
INSERT INTO connection_log(event_user_name, event_type, event_date)
VALUES (user_name,'DISCONNECT', SYSDATE);
END disconnecting;
END user_connection;
```

Ograniczenia

Istnieje kilka ograniczeń obowiązujących przy tworzeniu wyzwalaczy w bazie danych ORACLE

- Ciało wyzwalacza nie może być dłuższe niż 32760 bajtów
- Ciała wyzwalaczy niesystemowych nie mogą zawierać instrukcji DDL
- Nie można w nich umieszczać także poleceń z języka DCL, takich jak: ROLLBACK, SEVPOINT i COMMIT
- Wyzwalacze niesystemowe zadeklarowane jako autonomiczne mogą zawierać polecenia DCL
- Instrukcje SQL w transakcjach zdalnych - wywołanie zdalnej funkcji lub procedury z poziomu schematu w ciele wyzwalacza, może spowodować niedopasowanie znacznika czasu lub sygnatury

Usuwanie programów składowanych

Usunięcie procedury, funkcji lub wyzwalacza

DROP { PROCEDURE | FUNCTION | TRIGGER } nazwa**;**

Usunięcie pakietu

DROP PACKAGE [BODY] nazwa_pakietu**;**

Aktywne bazy danych

Funkcjonalność aktywnych bazy danych

Nowa funkcjonalność:

- monitorowanie zdarzeń,
- ewaluacja warunków logicznych,
- autonomiczne uruchamianie akcji,

umożliwia podejmowanie przez system bazy danych autonomicznej aktywności w obszarze zastrzeżonym dotąd dla aplikacji bazy danych.

Model ECA

Model: **E**vent (i) - **C**ondition (i, ii) - **A**ction (ii)

Definiowanie aktywnych reguł przez trzy elementy:

- wystąpienie zdarzenia
- weryfikacja warunku
- *odpalenie* akcji

Aktywne reguły są wykonywane w dwóch fazach:

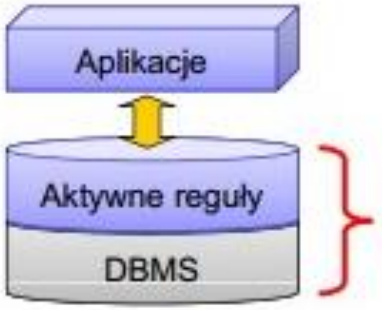
(i) - faza wystąpienie zdarzenia
(ii) - faza odpalenia akcji

Dziedziny zastosowania

Dziedziny zastosowania

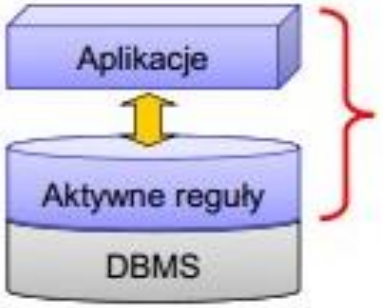
Systemowe – nowe funkcje DBMS

- weryfikacja złożonych więzów integralności
- utrzymywanie danych wywiedzionych
- zarządzanie rozproszonymi bazy danych
- zarządzanie przepływami pracy



Aplikacyjne – logika biznesowa w bazie danych

- zarządzanie procesami przemysłowymi
- systemy giełdowe
- bazy wiedzy



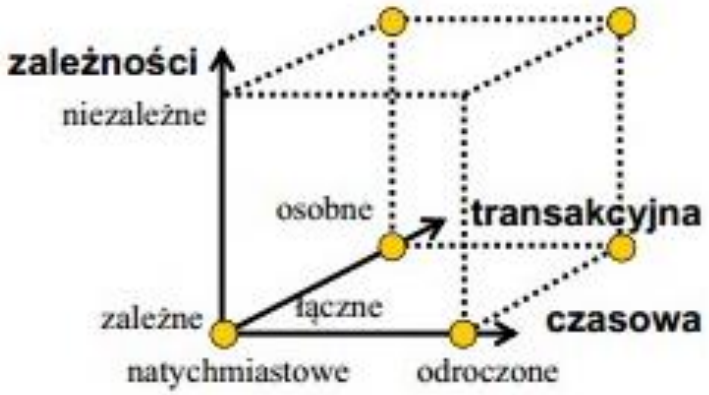
Własności aktywnych baz danych

- **Modele aktywności** – zależności czasowe i przyczynowo-skutkowe między zdarzeniami i akcjami
- **Zdarzenia elementarne** – zbiór typów zdarzeń, które mogą być podstawą definiowania aktywnych reguł
- **Operatory zdarzeniowe** – zbiór operatorów umożliwiających specyfikację złożonych wyrażeń zdarzeniowych
- **Kontekst definicji aktywnych reguł** – pojedyncza dana, zbiór danych, baza danych

Schematy aktywności

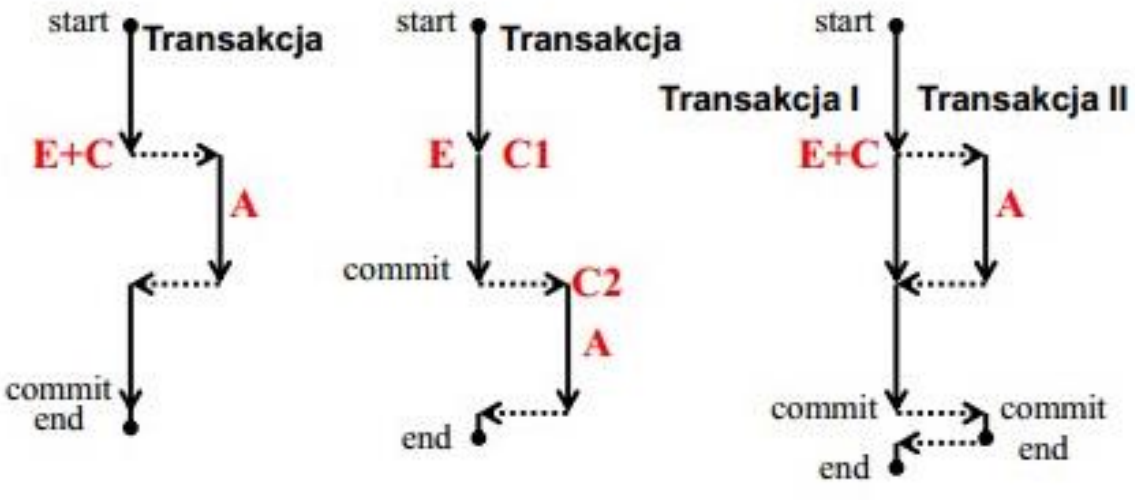
Można wyróżnić trzy relacje zachodzące między fazami wystąpienia zdarzenia i uruchomienia akcji:

- czasowa
- transakcyjna
- zależności



Przykłady schematów aktywności

a) natychmiastowe, zależne i łączne
b) odroczone, zależne i łączne
c) natychmiastowe, zależne, osobne



Przykłady schematów aktywności

- **Zmiany stanu bazy danych**

  Wstawienie, usunięcie, modyfikacja, odczyt, dostęp
- **Zmiany schematu bazy danych**

  Tworzenie, modyfikacja, usunięcie
- **Zdarzenia w systemie bazy danych**

  LOGIN, LOGOFF, STARTUP, SHUDOWN, ERROR
- **Zdarzenia związane ze zmianami stanu transakcji**

  Rozpoczęcie, punkt akceptacji, zatwierdzenie, wycofanie
- **Zdarzenia temporalne**

  punkt czasu, upływ czasu, periodycznie
- **Zdarzenia zgłaszane przez użytkownika**

Definiowanie aktywnych reguł - ORACLE

```
CREATE OR REPLACE TRIGGER zmodyfikuj_stan_osobowy
AFTER INSERT ON Pracownicy
FOR EACH ROW
BEGIN
  UPDATE Zespoły
  SET stanOsob = stanOsob + 1
  WHERE idZesp = :NEW.idZesp;
END;
```

Definiowanie aktywnych reguł - ORACLE

```
CREATE TRIGGER kontrola_plac
AFTER UPDATE OF pensja ON Pracownicy
FOR EACH ROW
WHEN (NEW.pensja < 1000)
BEGIN
  RAISE_APPLICATION_ERROR(
      -20000,'Poniżej płacy minimalnej');
END;
```

(0.9*1800 < 1000) → **FALSE**

idPrac	Nazwisko	pensja
100	Tarzan	1800
110	Kowalski	2500

fire

idPrac	Nazwisko	pensja
100	Tarzan	1620
110	Kowalski	2500

```
UPDATE Pracownicy
SET pensja = 0.9 * pensja
WHERE nazwisko = 'Tarzan'
```

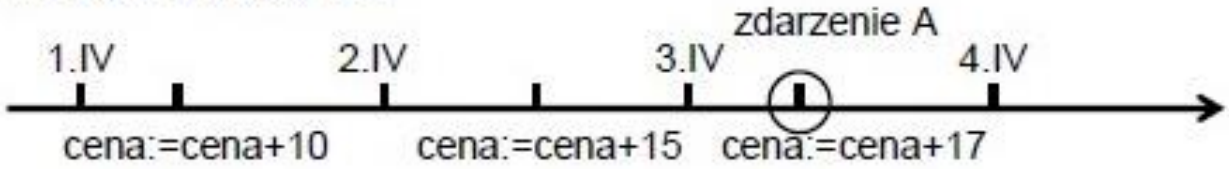
Definiowanie aktywnych reguł SQL SERVER

```
CREATE TRIGGER liczba_prac ON Pracownicy
AFTER DELETE AS
  UPDATE Zespoly
  SET liczba_prac = liczba_prac - (
    SELECT COUNT(*) FROM deleted
    WHERE deleted.id_zesp = Zespoly.id_zesp)
  WHERE id_zesp IN (
    SELECT DISTINCT id_zesp FROM deleted)
```

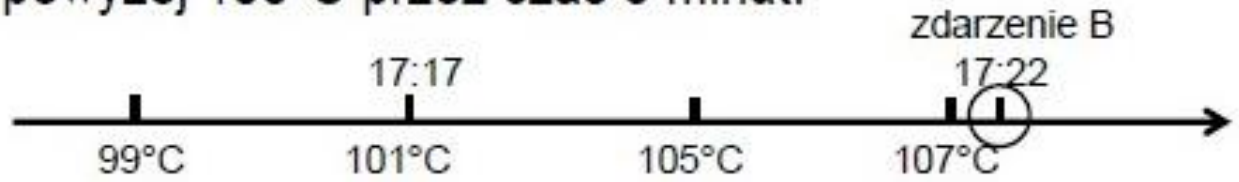
Zdarzenia złożone

Aktywne reguły wyzwalane zdarzeniami złożonymi:

- Zdarzenie A: trzy kolejne obniżki cen akcji giełdowych w czasie trzech dni



- Zdarzenie B: utrzymywanie się temperatury reaktora powyżej 100°C przez czas 5 minut.

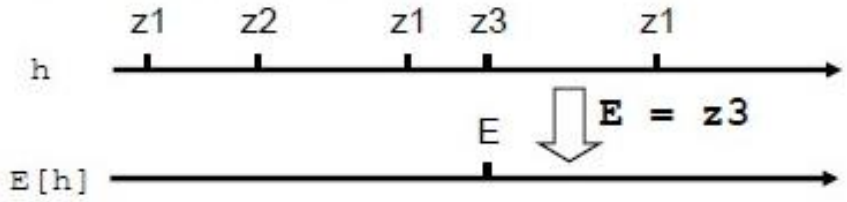


Wyrażenia zdarzeniowe

Ewaluacja zdarzeń złożonych wymaga utrzymywania i analizy historii zdarzeń.

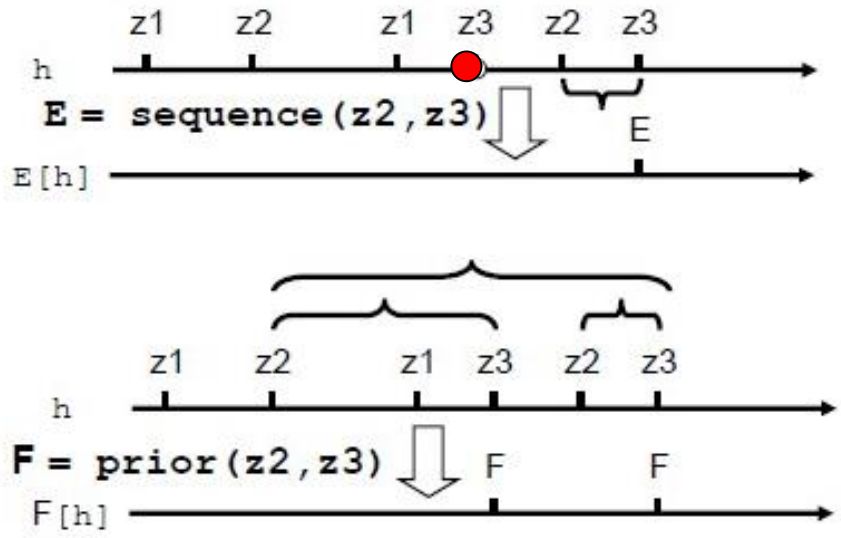
Historia zdarzeń jest skończonym zbiorem zdarzeń, w którym żadne dwa zdarzenia nie mogą mieć tego samego znacznika czasowego.

Wyrażenie zdarzeniowe *E* określone w historii zdarzeń *h* wyznacza podzbiór tych zdarzeń historii *h*, w których zdarzenie *E* jest spełnione.



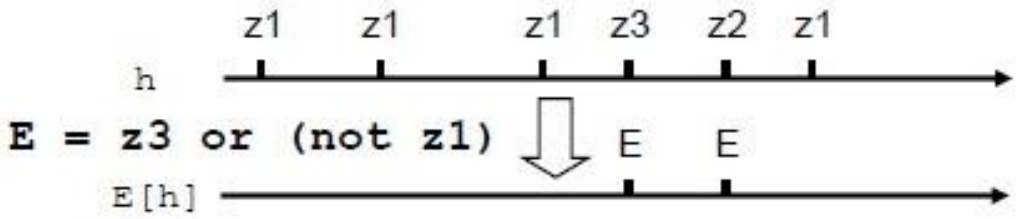
Operatory zdarzeniowe

Operatory następstwa: **sequence** i **prior**



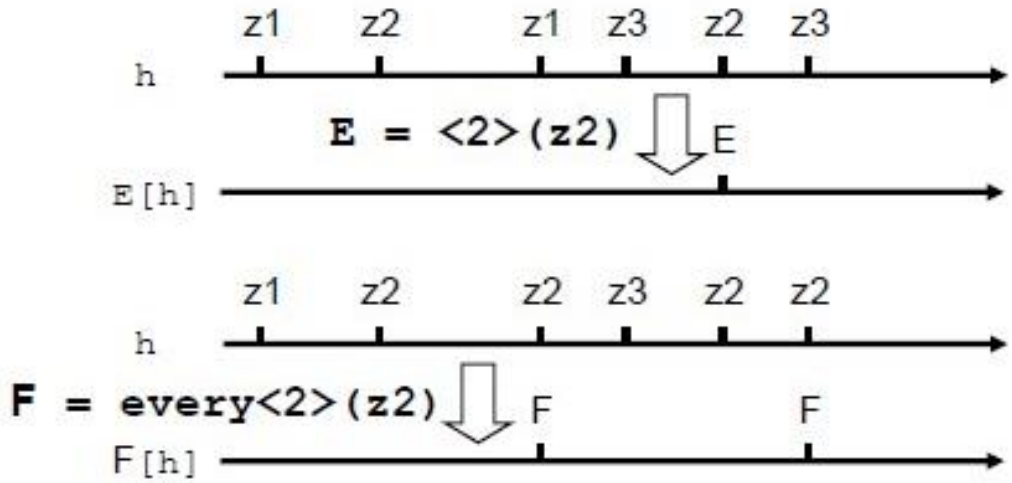
Operatory zdarzeniowe

Operatory logiczne: **or**, **and** i **not**



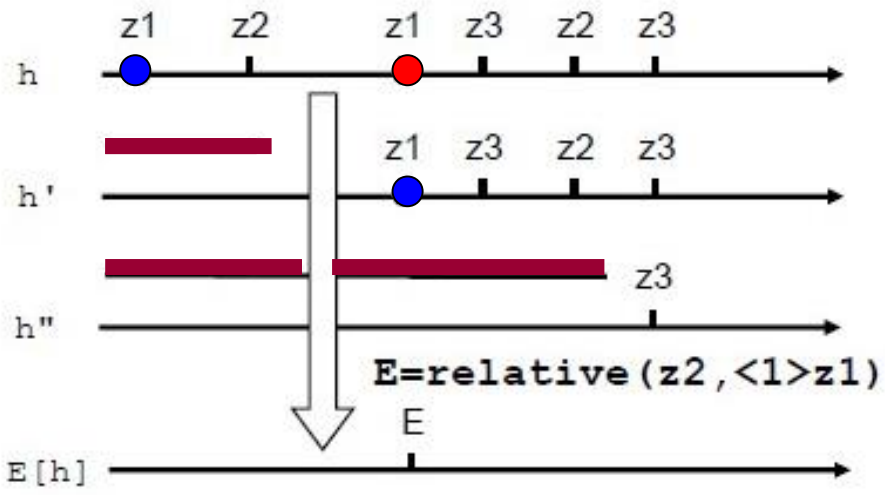
Operatory zdarzeniowe

Operatory wyliczeniowe: <n>E, every<n>E



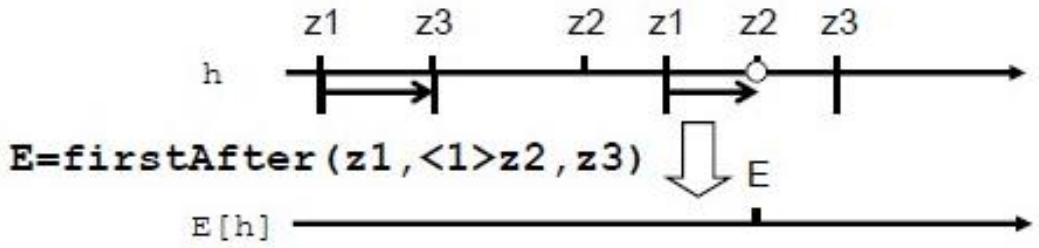
Operatory zdarzeniowe

Operator następstwa: **relative**



Operatory zdarzeniowe

Operator okna: **firstAfter**



Aktywna reguła ze zdarzeniem złożonym

Przełącz układ sieci jeżeli przepustowość przez 5 minut jest mniejsza od wartości progowej równej 10

```
#define PoniżejProgu AFTER UPDATE && przepustowość<10
#define PowyżejProgu AFTER UPDATE && przepustowość>10
class Łącze {
private:
      float przepustowość;
public:
      boolean Przełącz();
trigger:
      SpadekPrzepustowości():separate dependent
      firstAtfter( relative(
            ZmianaPowyżejProgu,
                  <1>(PoniżejProgu)),
            AFTER TIME (M=5), PowyżejProgu)
      ==> Przełącz ();};
```

Metodyka projektowania aktywnych reguł

- **własność stopu –** przetwarzanie akcji reguł uaktywnionych przez pojedyncze zdarzenie zostanie zakończone w skończonym czasie
- **determinizm stanu –** kolejność wykonania akcji reguł uaktywnionych tym samym zdarzeniem nie ma wpływu na końcowy stan bazy danych

Własności stopu zbioru reguł

```
CREATE TRIGGER zwiększ_rangę
AFTER UPDATE OF wysokość ON Premie
FOR EACH ROW
WHEN (NEW.wysokość - old.wysokość > 400)
BEGIN
      UPDATE Pracownicy SET ranga = ranga+1
      WHERE nr = :NEW.nr_prac;
END;
```

```
CREATE TRIGGER zwiększ_premię
AFTER UPDATE OF ranga ON Pracownicy
FOR EACH ROW
BEGIN
  UPDATE Premie
  SET wysokość = wysokość + 500 * :NEW.ranga
  WHERE nr_prac = :NEW.nr;
END;
```

Własności stopu

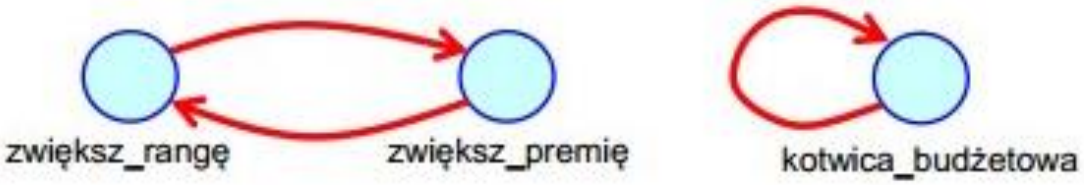
```
CREATE TRIGGER kotwica_budżetowa
AFTER INSERT OR UPDATE OF budżet ON Zespoły
DECLARE
  suma FLOAT;
BEGIN
  SELECT SUM(budżet) INTO suma FROM Zespoły;
  IF (suma > 100) THEN
    UPDATE Zespoły
    SET budżet = 0.9*budżet;
  END IF;
END;
```

Statystyczna walidacja własności stopu

Graf wyzwalania (GW)

Węzły grafu reprezentują zbiór aktywnych reguł. Dwa węzły grafu GW, reprezentujące reguły R1 i R2 są połączone krawędzią skierowaną od węzła R1 do R2, jeżeli kod akcji reguły R1 zawiera instrukcje manipulacji danymi na relacji, której modyfikacja jest zdarzeniem uaktywniającym regułę R2.

Brak cyklu w grafie wyzwalania oznacza, że analizowany zbiór reguł posiada własność stopu. Występowanie cyklu w grafie wyzwalania oznacza, że zbiór reguł może nie posiadać własności stopu.



Własności determinizmu stanu

```
CREATE TRIGGER zwiększ_premię_1
AFTER UPDATE OF wysokość ON Sprzedaż
FOR EACH ROW
WHEN (NEW.wysokość - OLD.wysokość > 100)
BEGIN
  UPDATE Pracownicy
  SET premia = premia + 1000
  WHERE nr = :NEW.nr_prac;
END;

CREATE TRIGGER zwiększ_premię_2
AFTER UPDATE OF wysokość ON Sprzedaż
FOR EACH ROW
WHEN (NEW.wysokość - OLD.wysokość > 500)
BEGIN
  UPDATE Pracownicy
  SET premia = 1.5*premia
  WHERE nr = :NEW.nr_prac;
END;
```