Transakcje w relacyjnych bazach danych

Tymon Tobolski 181037
Steve Olela 182928

Politechnika Wrocławska
2012

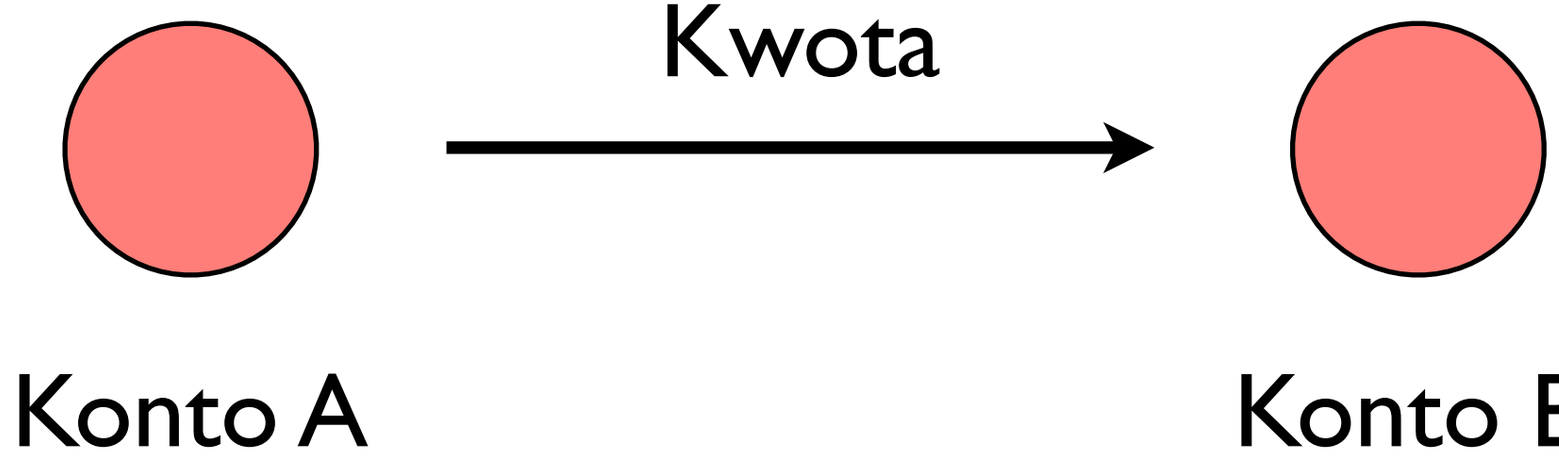
Plan prezentacji

- Wprowadzenie
- ACID
- Mechanizmy transakcji
- Niekorzystne zjawiska
- Poziomy izolacji
- Zakleszczenia

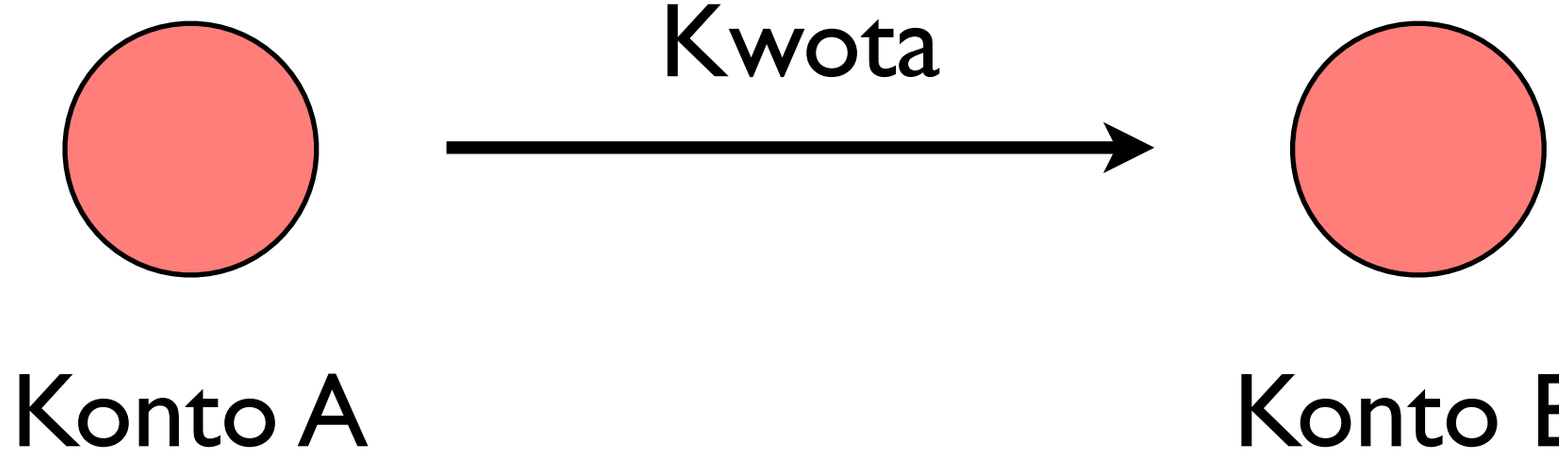
Wprowadzenie

Transakcja - zbiór operacji, które muszą być wykonane **w całości** lub wcale

Przykład - przelew



Przykład - przelew



- Wymaga 2 operacji w bazie danych
  - Odjęcie kwoty od konta A
  - Dodanie kwoty do konta B

Przykład - przelew

	Konto A	Konto B	Suma
Stan początkowy	100	100	200
KontoA -= 50	50	100	**150**
KontoB += 50	50	150	200

Przykład - przelew

- Obie operacje są nierozłączne
- Nie można dopuścic do sytuacji, w której tylko jedna zostanie wykonana
- Baza musi pozostać w którymś z **poprawnych** stanów

ACID

- A (atomicity) - atomowość
- C (consistency) - spójność
- I (isolation) - izolacja
- D (durability) - trwałość

Atomowość

- każda transakcja musi być wykonana albo w całości albo wcale
- jeśli jedna operacja się nie powiedzie to cała transakcja jest anulowana
- system odporny na awarie - brak istnienia “niedkończonych transakcji”

Spójność

- transakcja - przejście z jednego poprawnego stanu w drugi poprawny
- po zakończeniu transakcji dane muszą spełniać narzucone zasady

Izolacja

- zmiany dokonywane przez jedną transakcje są niewidoczne dla innej transakcji
- w praktyce stosuje sie mniej restrykcyjne **poziomy izolacji**

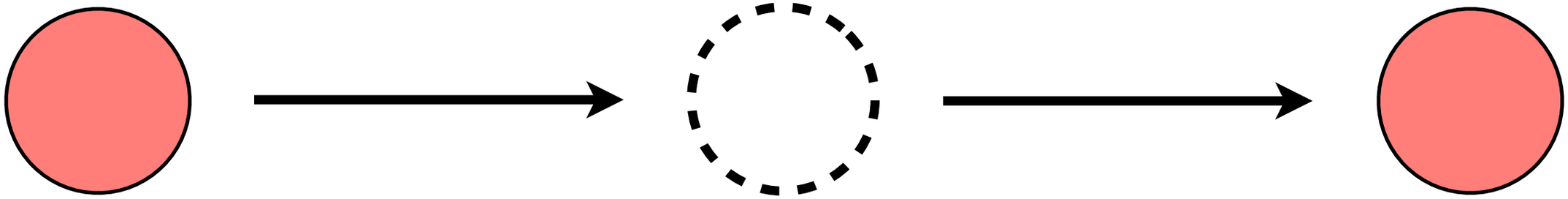
Trwałość

- po zatwierdzeniu transakcji zmiany zostaną zapisane (a transakcja zakończona)
- system jest odporny na awarie, czynniki zewnętrzne
- po awari można odtworzyć stan po ostatniej zatwierdzonej transakcji

Mechanizmy transakcji

- Logowanie transakcji
  - każdy etap transakcji jest logowany
  - możliwość odtworzenia stanu sprzed przerwanej transakcji (w wyniku błędu / ”ręcznie”)

Zmiana stanu bazy

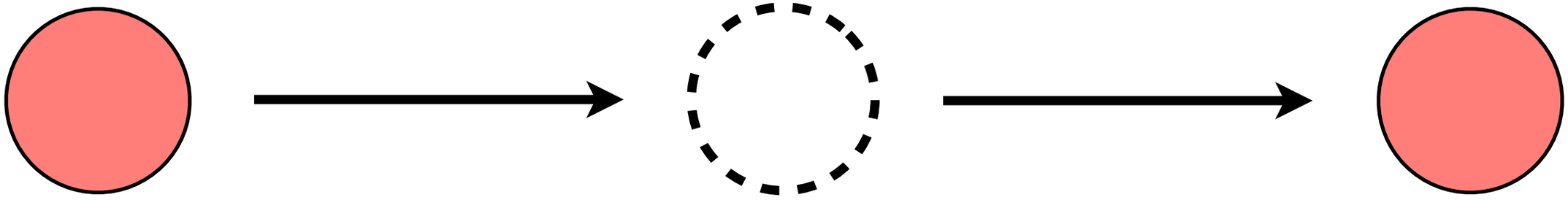


Stan A

Stan "przejściowy"

Stan B

Zmiana stanu bazy



Stan A

Stan "przejściowy"

Stan B

Początek transakcji

Operacje na bazie danych

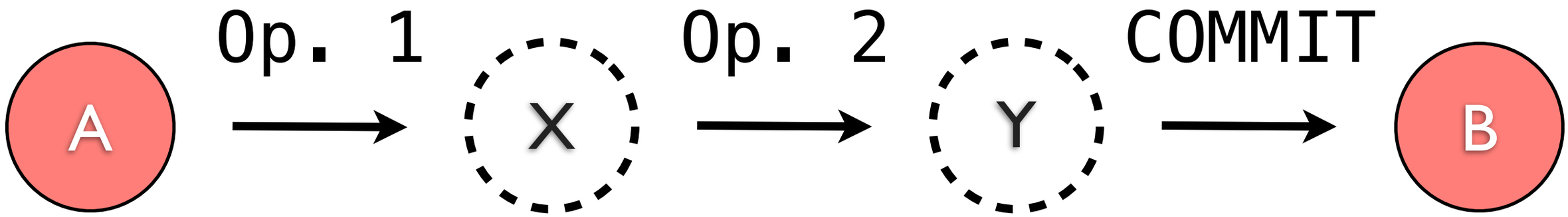
Koniec transakcji

Transakcje w SQL

		Komentarz
1	`BEGIN`	rozpoczęcie transakcji
2	`...`	operacje na bazie
...	`...`	
n-1	`...`	
n	`COMMIT`	zapisanie zmian

Transakcje w SQL

```
BEGIN
```

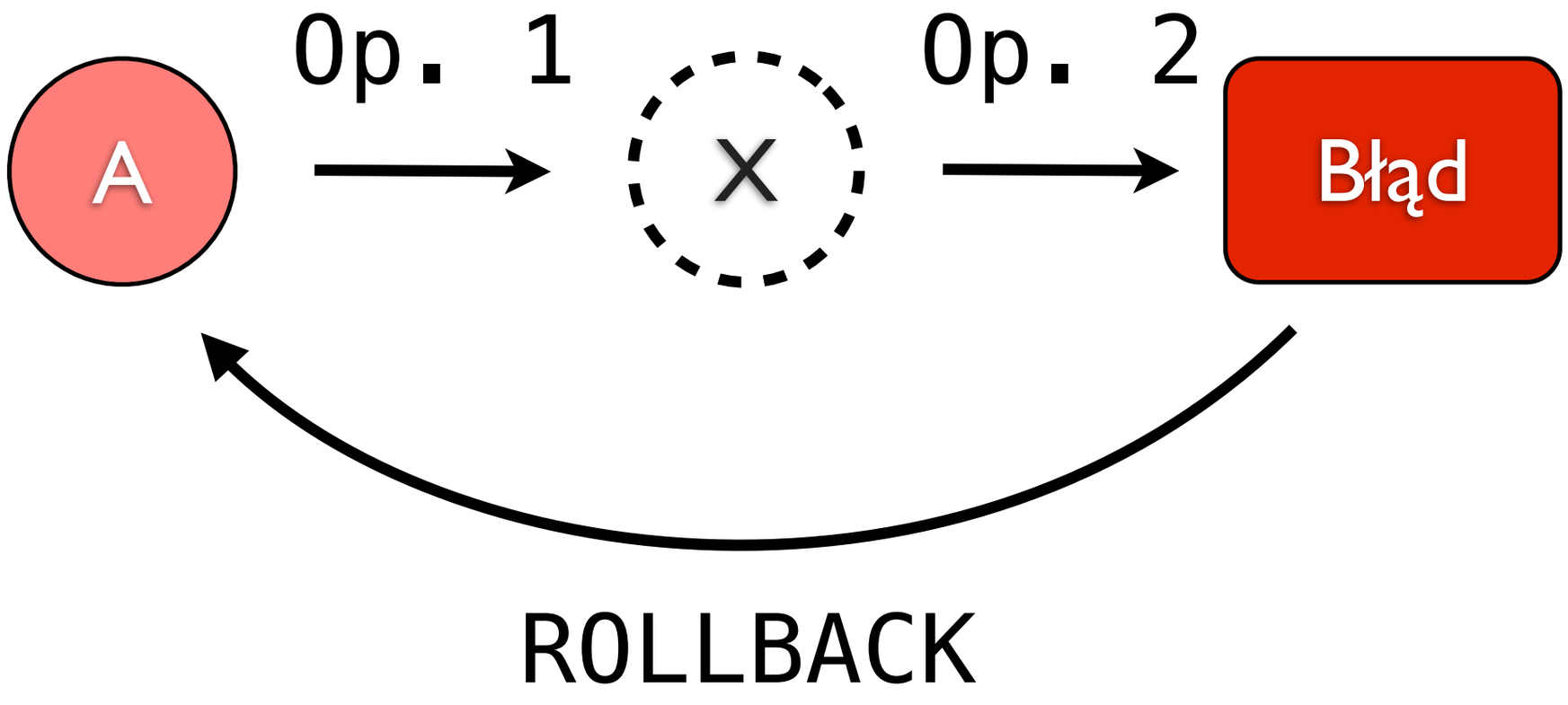


Transakcje w SQL

- COMMIT
  - trwały zapis zmian
  - zwolnienie blokad

Transakcje w SQL

```
BEGIN
```



Transakcje w SQL

- ROLLBACK
  - anulowanie wszystkich zmian
  - zwolnienie blokad

Transakcje w SQL

- w bazach SQL *każda* operacja jest "opakowana" w transakcje - AUTOCOMMIT

Transakcje w SQL

- w bazach SQL *każda* operacja jest "opakowana" w transakcje - AUTOCOMMIT

```
OPERACJA 1

OPERACJA 2
```

Interpretowane jako →

```
BEGIN
OPERACJA 1
COMMIT

BEGIN
OPERACJA 2
COMMIT
```

Transakcje w SQL

- w bazach SQL *każda* operacja jest "opakowana" w transakcje - AUTOCOMMIT

Dwie **niezależne** operacje!

```
BEGIN
OPERACJA 1
COMMIT
```

```
BEGIN
OPERACJA 2
COMMIT
```

Transakcje w SQL

```
BEGIN
OPERACJA 1
OPERACJA 2
COMMIT
```

1 atomowa operacja

```
BEGIN
OPERACJA 1
COMMIT

BEGIN
OPERACJA 2
COMMIT
```

2 atomowe operacje

Transakcje w SQL

`BEGIN`	rozpoczęcie transakcji
`...`	
`SAVEPOINT <P1>`	punkt kontrolny
`...`	
`SAVEPOINT <P2>`	punkt kontrolny
`...`	
`ROLLBACK TO <P1>`	zapisanie zmian

Niekorzystne zjawiska

Niekorzystne zjawiska powstające podczas współbieżnego wykonywania operacji na tych samych danych

- utracona zmiana
- brudny odczyt
- niepowtarzalny odczyt
- niewidzialne wiersze (fantomowe)

Utracona zmiana

t	Transakcja 1	Transakcja 2	x
t_0	x1 = Odczyt(x) // 20		20
t_1		x2 = Odczyt(x) // 20	20
t_2	x1' = Zmiana(x1) // 30		20
t_3		x2' = Zmiana(x1) // 10	20
t_4	x = Zapis(x1) // 30		30
t_5		**x = Zapis(x2')** // 10	10

Zmiana dokonana przez Transakcje1 została utracona

Utracona zmiana

t_0

```
SELECT * FROM "accounts"
WHERE "id" = 1 -- “Tom”
```

```
SELECT * FROM "accounts"
WHERE "id" = 1 -- “Tom”
```

Utracona zmiana

t_0

```
SELECT * FROM "accounts"
WHERE "id" = 1 -- “Tom”
```

```
UPDATE "accounts"
SET “name” = “John”
WHERE "id" = 1
```

```
SELECT * FROM "accounts"
WHERE "id" = 1 -- “Tom”
```

```
UPDATE "accounts"
SET “name” = “Mark”
WHERE "id" = 1
```

Utracona zmiana

t_0

```
SELECT * FROM "accounts"
WHERE "id" = 1 -- “Tom”
```

t = kilka minut

```
UPDATE "accounts"
SET “name” = “John”
WHERE "id" = 1
```

```
SELECT * FROM "accounts"
WHERE "id" = 1 -- “Tom”
```

t = kilka minut

```
UPDATE "accounts"
SET “name” = “Mark”
WHERE "id" = 1
```

```
SELECT * FROM "accounts"
WHERE "id" = 1 -- “Mark”
```

Brudny odczyt

t	Transakcja 1	Transakcja 2	x
t_0	x1 = Odczyt(x) // 20		20
t_1	x1 = Zmiana() // 30		30
t_2		**x2 = Odczyt(x) // 30**	30
t_3	Rollback!		20
t_4		Commit!	20

Transakcja 2 operuje na nieistniejących danych zapisanych przez niezatwierdzona transakcje 1

Niepowtarzalny odczyt

t	Transakcja 1	Transakcja 2	x
t_0	x1 = Odczyt(x) // 20		20
t_1		x2 = Odczyt(x) // 20	20
t_2		x = Zmiana(x2) // 30	30
t_3		Commit!	30
t_4	**x1 = Odczyt(x) // 30**		30
t_5	Commit!		30

Transakcja 1 odczytuje dwa razy ten sam wiersz, ale z różnym rezultatem

Niewidzialne wiersze

t	Transakcja 1	Transakcja 2	x
t_0	Odczyt() // [A,B]		[A, B]
t_1		Insert(C)	[A, B, C]
t_2	**Odczyt() // [A,B,C]**		

Transakcja 1 widzi dane mimo że wcześniej ich nie było

Poziomy izolacji

- Poziom izolacji określa stopień niezależności wykonywania współbieżnych transakcji
- Określa dopuszczalność negatywnych zjawisk

Poziomy izolacji

- read uncommited
- read commited
- repeatable read
- serializable

Poziomy izolacji

Poziom	brudny odczyt	niepowtarzalny odczyt	niewidzialne wiersze
read uncommited	możliwy	możliwy	możliwe
read commited	ochrona	możliwy	możliwe
repeatable read	ochrona	ochrona	możliwe
serializable	ochrona	ochrona	ochrona

Read uncommited

- Transakcja może odczytywać niepotwierdzone dane, na których operują inne transakcje
- Używane z trybem READ ONLY
- Używane dla długich transakcji niewymagających pełnej poprawności

Read commited

- Gwarancja, że odczytane dane będą “pochodziły” tylko z zatwierdzonych transakcji
- Mimo to dane mogą sie zmienić podczas wykonywania transakcji
- Kompromis między wydajnością a izolacją

Repeatable read

- Gwarancja niezmienności danych przez cały czas trwania transakcji **- ale nie nowych rekordów**
- Obniżona wydajność
- Najczęściej stosowany przy wykonywaniu wielu zmian jednocześnie

Serializable

- Gwarancja niezmienności danych przez cały czas trwania transakcji
- Transakcje są wykonywane jedna po drugiej
- Najniższa wydajność
- Stosowany przy wymaganiu pełnej poprawności, głównie dla krótkich transakcji

Zakleszczenia

Zapewnienie odpowiedniego poziomu izolacji danych przy wielu współbieżnych transakcjach wymaga zastosowania mechanizmu blokad pojedynczych wierszy, grup wierszy, a nawet całych tabel

Zakleszczenia

- Zakleszczenie - sytuacja, w której conajmniej dwie transakcje czekają wzajemnie na zwolnienie zasobów, przez co żadna z nich nie może zakończyć działania

Zakleszczenia

t	Transakcja 1	Transakcja 2
t_0	Zablokuj(x)	
t_1		Zablokuj(y)
t_2	Zapisz(y)	
t_3	czekaj	Zapisz(x)
t_4	czekaj	czekaj
t_5	czekaj	czekaj

Metody unikania konfliktów

- Zapobieganie zakleszczeń
- Wykrywanie zakleszczeń

Zapobieganie zakleszczeń

- **wait-die** - transakcja T_1 próbuje uzyskać blokadę na zasobie X, który jest zablokowany przez transakcje T_2
  - Jeżeli T_1 jest starsza od T_2 => T_1 czeka
  - Jeżeli T_1 jest młodsza od T_2 => T_1 jest anulowana i restartowana z tym samym znacznikiem czasu

Zapobieganie zakleszczeń

- **wound-die** - transakcja T_1 próbuje uzyskać blokadę na zasobie X, który jest zablokowany przez transakcje T_2
  - Jeżeli T_1 jest starsza od T_2 => T_2 jest anulowana i restartowana z tym samym znacznikiem czasu
  - Jeżeli T_1 jest młodsza od T_2 => T_1 czeka

Zapobieganie zakleszczeń

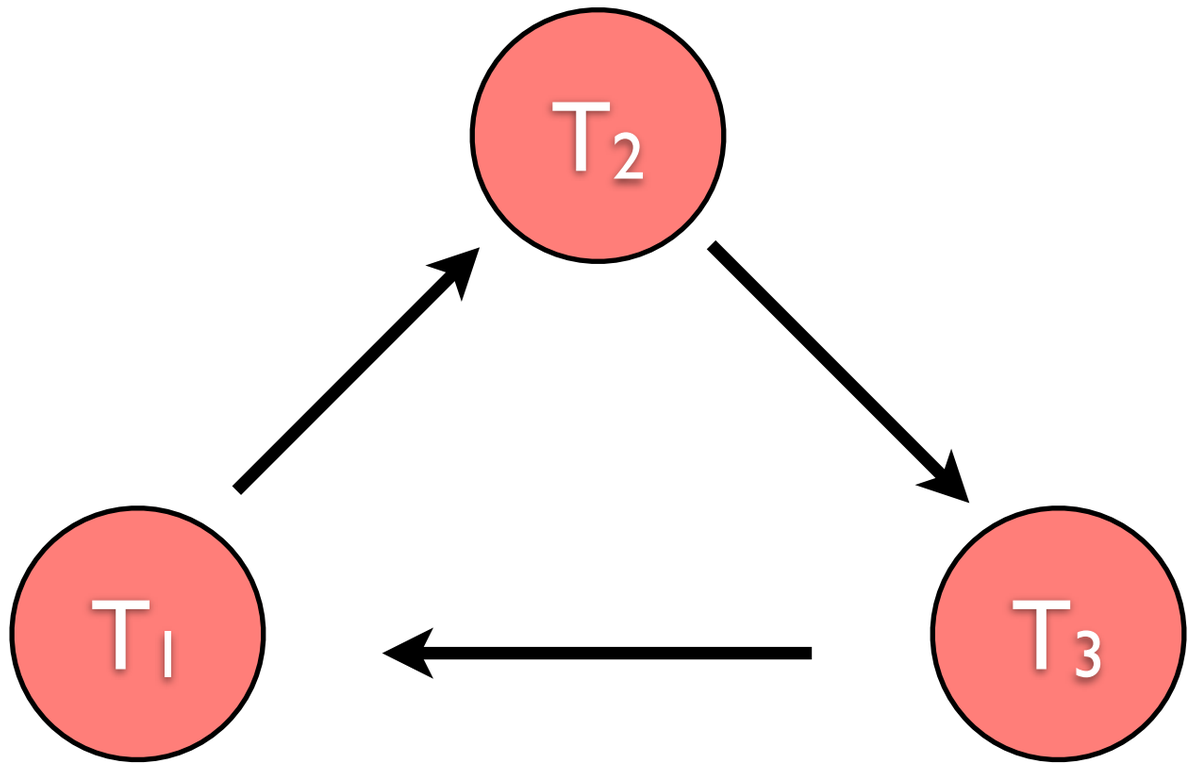
- **no-waiting** - transakcja T_1 próbuje uzyskać blokadę na zasobie X, który jest zablokowany przez transakcje T_2
  - transakcja T_1 jest anulowana i restartowana z pewnym opóźnieniem czasowym

Wykrywanie zakleszczeń

- Zakleszczenia są stosunkowo rzadkim zjawiskiem - bardziej opłaca się wykrywanie zakleszczeń i ich rozwiązywanie w momencie wystąpienia

Wykrywanie zakleszczeń

- Znalezienie cyklu w grafie oczekiwania



Wycofanie jednej transakcji i restart