Rozdział 17
Zarządzanie współbieżnością

Definicja i własności transakcji, zatwierdzanie i wycofywanie, punkty bezpieczeństwa, spójność, anomalie współbieżnego dostępu do danych, poziomy izolacji transakcji, blokady, zakleszczenie, odroczone ograniczenia integralnościowe

Plan ćwiczenia

- Koncepcja i cechy transakcji.
- Początek i koniec transakcji, punkty bezpieczeństwa transakcji.
- Spójność bazy danych, anomalie współbieżnego dostępu do danych.
- Poziomy izolacji transakcji.
- Blokady w bazie danych.
- Zakleszczenie.
- Opóźnione ograniczenia integralnościowe.

Dostęp współbieżny a dostęp spójny

- Współbieżny dostęp do danych
  - Wielu użytkowników chce jednocześnie uzyskać dostęp do tych samych danych – współbieżność sesji użytkowników
- Spójny dostęp do danych
  - Każdy użytkownik powinien „widzieć" i „pozostawiać" spójny stan bazy danych

Użytkownik A (przelew)	Użytkownik B (analiza)
SQL> update konta set stan = stan - 50 where numer = 111; 1 wiersz został zmodyfikowany.	
	SQL> select sum(stan) from konta; SUM(KONTA) ---------- 100323
SQL> update konta set stan = stan + 50 where numer = 222; 1 wiersz został zmodyfikowany.	

Spójność bazy danych

- Baza danych jest w stanie spójnym, jeśli:
  - stan bazy danych jest zgodny ze stanem reprezentowanego przez nią fragmentu rzeczywistości,
  - wszystkie ograniczenia integralnościowe w bazie danych są spełnione.
- Zagrożenia spójności:
  - awarie,
  - błędne działania użytkowników,
  - współbieżny dostęp do danych
- Konieczne zaimplementowanie w SZBD mechanizmów zarządzania współbieżnością

Zarządzanie współbieżnością w SZBD Oracle

Transakcje

+

Wielowersyjny model spójności

+

Blokady

Definicja i własności transakcji

Transakcja:

jest sekwencją (uporządkowanym zbiorem) logicznie powiązanych operacji na bazie danych, która przeprowadza bazę danych z jednego stanu spójnego w inny stan spójny.

Atomowość (ang. *atomicity*)
- **zbiór operacji wchodzących w skład transakcji jest niepodzielny; albo zostaną wykonane wszystkie operacje transakcji albo żadna**

Spójność (ang. *consistency*)
- **transakcja pozostawia bazę danych w stanie spójnym**

Izolacja (ang. *isolation*)
- **transakcje są od siebie logicznie odseparowane. Mogą wzajemnie oddziaływać na siebie w taki sposób jak gdyby były wykonywane sekwencyjnie**

Trwałość (ang. *durability*)
- **wyniki poprawnie zakończonych transakcji nie mogą zostać utracone, niezależnie od awarii systemu**

Rozpoczęcie transakcji

- **SZBD Oracle: rozpoczęcie nowej sesji, zakończenie poprzedniej transakcji**
- **Inne systemy: jawne zaznaczenie poleceniem** `BEGIN TRANSACTION`

Zakończenie transakcji (1)

Status zakończenia transakcji:
- **zatwierdzenie**
  - **powodzenie transakcji**
  - **zmiany, wprowadzone przez operacje w ramach transakcji, zostają trwale zapisane w bazie danych**
- **wycofanie**
  - **niepowodzenie (awaria) transakcji**
  - **wszystkie zmiany, wprowadzone przez operacje w ramach transakcji, zostają anulowane**

Zakończenie transakcji (2)

- **Jawne zakończenie transakcji – wykonanie poleceń:**
  - **zatwierdzenie** — **COMMIT [WORK];**
  - **wycofanie** — **ROLLBACK [WORK];**
- **Niejawne zakończenie transakcji:**
  - **wykonanie instrukcji DDL lub DCL – zatwierdzenie**
  - **normalne zakończenie sesji – zatwierdzenie**
  - **awaryjne zakończenie sesji – wycofanie**

Transakcja DDL w SZBD Oracle

- **SZBD Oracle umieszcza każde polecenie DDL lub DCL (np. GRANT, REVOKE) w osobnej transakcji, automatycznie zatwierdzanej bezpośrednio po wykonaniu polecenia**

```
INSERT INTO zespoly
        VALUES (60, 'Finanse', 'Piotrowo 2');
UPDATE pracownicy
        SET id_zesp = 60 WHERE id_zesp = 20;

CREATE TABLE kadry
        (id number(6), nazwa varchar2(100));

DELETE pracownicy WHERE id_zesp = 60;
```

1. transakcja
2. transakcja
3. transakcja

Punkty bezpieczeństwa transakcji (1)

- **Pozwalają podzielić transakcję na etapy**
- **Operacje:**
  - **ustawienie punktu bezpieczeństwa:**

```
SAVEPOINT <etykieta>;
```

  - **usunięcie punktu bezpieczeństwa (brak wsparcia przez SZBD Oracle):**

```
RELEASE SAVEPOINT <etykieta>;
```

  - **cofnięcie transakcji do punktu bezpieczeństwa:**

```
ROLLBACK [WORK] TO [SAVEPOINT] <etykieta>;
```

- **Uwaga! Ustawienie punktu bezpieczeństwa z etykietą użytą przy definicji wcześniejszego punktu bezpieczeństwa powoduje usunięcie wcześniej zdefiniowanego punktu!**

Punkty bezpieczeństwa transakcji (2)

- **Wycofanie do punktu bezpieczeństwa:**
  - **transakcja pozostaje aktywna,**
  - **zmiany, wprowadzone przez operacje, zrealizowane między punktem bezpieczeństwa a poleceniem ROLLBACK są anulowane,**
  - **punkty bezpieczeństwa, ustawione pomiędzy punktem wskazanym a poleceniem ROLLBACK, zostają usunięte**
- **Przykładowe scenariusze:**

Sytuacja 1.

```
INSERT INTO...
SAVEPOINT S1;
DELETE...
SAVEPOINT S2;
UPDATE...
ROLLBACK;
```

Sytuacja 2.

```
INSERT INTO...
SAVEPOINT S1;
DELETE...
SAVEPOINT S2;
UPDATE...
ROLLBACK TO S1;
```

Sytuacja 3.

```
INSERT INTO...
SAVEPOINT S1;
DELETE...
SAVEPOINT S2;
UPDATE...
ROLLBACK TO S2;
```


Anomalie współbieżnego dostępu (1)

- **Brudny zapis (ang. *dirty write*)**
  - **zapis, wprowadzony przez transakcję, zostaje nadpisany przez zapis w innej transakcji**

T1	T2	placa_pod
		800
SQL> update pracownicy set placa_pod = 1000 where nazwisko='Kowalski';		1000
	SQL> update pracownicy set placa_pod = 2000 where nazwisko='Kowalski';	2000
SQL> rollback;		???

Anomalie współbieżnego dostępu (2)

- **Brudny odczyt** (ang. ***dirty read***)
  - **transakcja czyta dane zapisane przez inną, jeszcze nie zatwierdzoną transakcję**

T1	T2	placa_pod
...	...	1000
SQL> update pracownicy set placa_pod=placa_pod+100 where nazwisko='Kowalski';		1100
	SQL> select placa_pod from pracownicy where nazwisko='Kowalski';	1100
SQL> rollback;		1000

Anomalie współbieżnego dostępu (3)

- **Niepowtarzalny (rozmyty) odczyt** (ang. ***non-repeatable (fuzzy) read***)
  - **transakcja ponownie czyta wcześniej odczytane dane, które w międzyczasie zostały zmodyfikowane przez inną transakcję**

T1	T2	placa_pod
SQL> select placa_pod from pracownicy where id_prac = 200;		1200
	SQL> update pracownicy set placa_pod=placa_pod+100 where id_prac = 200; SQL> commit;	
SQL> select placa_pod from pracownicy where id_prac = 200;		1300

Anomalie współbieżnego dostępu (4)

- **Odczyt fantomu** (ang. ***phantom read***)
  - **transakcja ponownie czyta wcześniej odczytany zbiór danych, określony warunkiem wyszukiwania; zbiór w międzyczasie został zmodyfikowany przez operacje w innej transakcji**

T1	T2	
SQL> select nazwisko from pracownicy where etat='ASYSTENT';		HAPKE KOWALSKI
	SQL> insert into pracownicy (id_prac, nazwisko, etat) values(30,'NOWAK','ASYSTENT'); SQL> commit;	
SQL> select nazwisko from pracownicy where etat='ASYSTENT';		HAPKE KOWALSKI NOWAK

Poziom izolacji transakcji

- **Określa, w jaki sposób współbieżnie wykonywane transakcje mają na siebie wpływać**
- **Jest definiowany przez wskazanie anomalii, jakie eliminuje**
- **Niższy poziom – większy stopień współbieżności, ale większe prawdopodobieństwo wystąpienia anomalii**
- **Wyższy poziom – mniejsze prawdopodobieństwo anomalii kosztem ograniczenia współbieżności**

Poziomy izolacji transakcji ANSI SQL (1)

- **Poziomy izolacji:**
  - **odczyt niezatwierdzonych danych (ang. *read uncommitted*)**
  - **odczyt zatwierdzonych danych (ang. *read committed*)**
  - **powtarzalny odczyt (ang. *repeatable read*),**
  - **uszeregowalny (ang. *serializable*).**
- **Definiuje tylko cztery podstawowe anomalie, pomijając pozostałe**
- **Anomalia brudnego zapisu nie jest dopuszczalna na żadnym poziomie**

	Brudny odczyt	Rozmyty odczyt	Odczyt fantomu
READ UNCOMMITTED	możliwa	możliwa	możliwa
READ COMMITTED	nie występuje	możliwa	możliwa
REPEATABLE READ	nie występuje	nie występuje	możliwa
SERIALIZABLE	nie występuje	nie występuje	nie występuje

- **Zakłada, że stosowaną metodą synchronizacji transakcji jest blokowanie dwufazowe (2PL)**
  - Protokół 2PL gwarantuje poziom izolacji *repeatable read*. Uzyskanie pełnej uszeregowalności wymaga blokowania, za pomocą tej metody, całych tabel, zamiast pojedynczych wierszy.

Poziomy izolacji transakcji ANSI SQL (2)

- **Określenie poziomu izolacji dla transakcji**

SET TRANSACTION ISOLATION LEVEL {READ UNCOMMITTED | READ COMMITTED | REPEATABLE READ | SERIALIZABLE};

Poziomy izolacji transakcji w SZBD Oracle (1)

- **Dostępne jedynie dwa poziomy izolacji:**
  - **odczyt zatwierdzonych danych:**
    - poziom domyślny
    - odpowiedni dla baz danych z dużym prawdopodobieństwem konfliktu transakcji (np. równoczesny zapis tych samych danych)
    - gwarantuje wysoki stopień współbieżności
  - **uszeregowalny:**
    - odpowiedni dla dużych baz danych z krótkimi transakcjami modyfikującymi małe zbiory rekordów
    - odpowiedni dla baz danych z małym prawdopodobieństwem wystąpienia konfliktu transakcji
    - odpowiedni dla długich transakcji czytających dane
    - może zdegradować stopień współbieżności
    - Uwaga! Nie gwarantuje pełnej izolacji transakcji!

Poziomy izolacji transakcji w SZBD Oracle (2)

- **Utracona modyfikacja (ang. *lost update*)**
  - **modyfikacja, wprowadzona przez transakcję, zostaje nadpisana przez modyfikację w innej transakcji, operującej na wartościach sprzed modyfikacji**
  - **występuje na poziomie READ COMMITED, eliminowana na poziomie SERIALIZABLE**

T1	T2	placa_pod
SQL> select placa_pod from pracownicy where nazwisko='Kowalski'; -- 1000, zwiększ o 100	SQL> select placa_pod from pracownicy where nazwisko='Kowalski'; -- 1000, zwiększ o 200	1000
SQL> update pracownicy set placa_pod=1100 where nazwisko='Kowalski';		1100
	SQL> update pracownicy set placa_pod=1200 where nazwisko='Kowalski';	1200
SQL> commit;		
	SQL> commit;	

Poziomy izolacji transakcji w SZBD Oracle (3)

- **Skrośny zapis** (ang. ***write skew***)
  - **współbieżna modyfikacja powiązanych danych**
  - **występuje zarówno na poziomie READ COMMITED jak i SERIALIZABLE**

T1	T2
Założenie: tab_x.wartosc (200) > tab_y.wartosc (150)	
SQL> select wartosc from tab_x; -- 200 SQL> select wartosc from tab_y; -- 150. OK. x-40 > y	
	SQL> select wartosc from tab_x; -- 200 SQL> select wartosc from tab_y; -- 150. OK. x > y+30
SQL> update tab_x set wartosc = 160; SQL> commit;	
	SQL> update tab_y set wartosc = 180; SQL> commit;
Stan końcowy: tab_x.wartosc (160) < tab_y.wartosc (180)	

Poziomy izolacji transakcji w SZBD Oracle (4)

- **Określenie poziomu izolacji transakcji:**
  - **dla pojedynczej transakcji:**

```
SET TRANSACTION ISOLATION LEVEL
    {READ COMMITTED | SERIALIZABLE};
```

  - Uwaga! Musi być pierwszym poleceniem w transakcji!

  - **Dla wszystkich transakcji w bieżącej sesji**

```
ALTER SESSION SET ISOLATION_LEVEL =
    {READ COMMITTED | SERIALIZABLE};
```


Spójność odczytu danych polecenia

- **Dane, czytane przez polecenie SQL:**
  - **pochodzą z tego samego punktu w czasie:**
    - ***read commited* – moment rozpoczęcia wykonania polecenia**
    - ***serializable* – moment rozpoczęcia transakcji**
  - **nie są zmienione przez inne, niezatwierdzone jeszcze transakcje**

T1	T2
SQL> select sum(placa_pod) from pracownicy;	
	SQL> update pracownicy set placa_pod = placa_pod + 200 where nazwisko = 'Kowalski'; 1 wiersz został zmodyfikowany. SQL> commit;
SUM(PLACA_POD) -------------- 3000	

Spójność odczytu danych transakcji (1)

- **Dane, czytane przez wszystkie polecenia SQL w ramach transakcji, pochodzą z tego samego punktu w czasie**
- **Dostępne tylko dla poziomu izolacji *serializable*.**

T1	T2
SQL> select sum(placa_pod) from pracownicy; -- wynik: 3000	
	SQL> update pracownicy set placa_pod = placa_pod + 200 where nazwisko = 'Kowalski'; 1 wiersz został zmodyfikowany. SQL> commit;
SQL> select sum(placa_pod) from pracownicy; -- wynik: 3000	
SQL> commit;	
SQL> select sum(placa_pod) from pracownicy; -- wynik: 3200	

Spójność odczytu danych transakcji (2)

- **Dostępne również po określeniu trybu dostępu transakcji na READ ONLY (domyślnie READ WRITE)**

```
SET TRANSACTION {READ WRITE | READ ONLY};
```


Wielowersyjny model spójności w SZBD Oracle (1)

- **SZBD może przechowywać jednocześnie wiele wersji danych**
- **Nowa wersja tworzona w momencie modyfikacji danych przez transakcję**
- **Wersje (wartości sprzed modyfikacji) składowane w specjalnych strukturach, segmentach wycofania**
- **Operacja odczytu jest kierowana do odpowiedniej wersji danych z odpowiedniego punktu w czasie (zależnego od poziomu izolacji, w jakim działa transakcja)**
- **Liczba wersji zależy od liczby transakcji oraz operacji operujących na tych samych danych**
- **Wersje danych wykorzystywane są również przy wycofywaniu transakcji**

Wielowersyjny model spójności w SZBD Oracle (2)

T1	T2	T3
SQL> set transaction read only;		
SQL> select A … -- wynik: 100		
		SQL> update … set A=200 SQL> commit;
	SQL> set transaction read only;	
SQL> select A … -- wynik: 100	SQL> select A … -- wynik: 200	SQL> select A … -- wynik: 200
		SQL> update … set A=300 SQL> commit;
SQL> select A … -- wynik: 100	SQL> select A … -- wynik: 200	SQL> select A … -- wynik: 300

Blokady w bazie danych

- **Blokada – przydzielenie zasobu do zadania.**
- **Rodzaje blokad:**
  - **wyłączne – zasób może być przydzielony tylko do jednego zadania,**
  - **współdzielone – zasób może zostać przydzielony jednocześnie wielu zadaniom.**
- **Zadanie, któremu przydzielany jest zasób – transakcja.**
- **Blokowany zasób:**
  - **atrybut,**
  - **rekord,**
  - **strona dyskowa,**
  - **relacja,**
  - **baza danych.**

- **Im mniejsze ziarno blokowania, tym większy stopień komplikacji algorytmu zarządzania współbieżnością – większe wykorzystanie zasobów systemowych.**

Blokady w SZBD Oracle

- **Rodzaje blokad:**
  - **blokady DML – zabezpieczają spójność danych obiektów bazodanowych**
  - **blokady DDL – zabezpieczają struktury obiektów bazodanowych**
  - **blokady systemowe – zabezpieczają wewnętrzne struktury bazy danych**
- **Blokowanie jest realizowane automatycznie bez udziału użytkownika:**
  - **możliwe ręczne zakładanie blokad DML, np. dla transakcji intensywnie przetwarzających dane (jedna blokada ręczna może zastąpić wiele blokad na rekordach)**

Blokady DML (1)

- **Zabezpieczają spójność danych**
- **Blokowane zasoby:**
  - **rekord – blokada wyłączna, oznaczenie: TX**
  - **relacja – blokada wyłączna lub współdzielona (zależnie od operacji), oznaczenie: TM**
- **Modyfikacja danych (UPDATE, INSERT, DELETE, SELECT FOR UPDATE) – założenie blokady wyłącznej na modyfikowanym rekordzie + założenie blokady na relacji:**
  - **blokada rekordu – nie pozwala innym transakcjom na równoległą modyfikację danych rekordu**
  - **blokada relacji – zapobiega np. współbieżnemu wykonaniu operacji modyfikacji struktury relacji**

Blokady DML (2)

- **Odczyt danych (SELECT) nie zakłada blokad:**
  - **odczyt danych nigdy nie jest blokowany przez modyfikację – odczyt kierowany jest do odpowiedniej wersji danych**
  - **odczyt nigdy nie blokuje modyfikacji – modyfikacja tworzy nową wersję danych.**
- **Czas utrzymania blokady:**
  - **do końca transakcji (zatwierdzenie lub wycofanie), w której blokada została założona (wymóg algorytmu 2PL)**
  - **wyjątek – polecenie ROLLBACK TO SAVEPOINT… zdejmuje blokady rekordów, założone przez operacje po wskazanym punkcie bezpieczeństwa**
    - blokady na zwolnionych zasobach mogą uzyskać tylko te transakcje, które dotąd na nie nie czekały, transakcje oczekujące są blokowane do zakończenia transakcji wykonującej ROLLBACK TO SAVEPOINT

Blokady DML (3)

- **Ręczne blokowanie rekordów:**

```
SELECT ... FROM ... WHERE ...
FOR UPDATE [OF <lista_atrybutów>] [NOWAIT];
```

- **Przykład:**

```
SELECT *
FROM pracownicy NATURAL JOIN zespoly
WHERE nazwa = 'ADMINISTRACJA' AND placa_pod > 1000
FOR UPDATE OF id_prac;
```

Blokady DML relacji

- Zakładane na relacjach
- Zapobiegają współbieżnej realizacji operacji DML i DDL
- Zakładane automatycznie (niektóre) lub ręcznie (wszystkie)
- Ręczne blokowanie relacji:

```
LOCK TABLE <nazwa_relacji>
IN <tryb_blokowania> MODE [NOWAIT];
```

  - przykład:

```
LOCK TABLE pracownicy IN EXCLUSIVE MODE NOWAIT;
```


Blokada relacji: ROW SHARE (RS)

- Alternatywna nazwa: Subshare Table Lock (SS)
- Zakładanie blokady:

```
LOCK TABLE <nazwa_relacji> IN [ROW SHARE|SHARE
UPDATE] MODE;
```

- Najmniej restrykcyjna
- Dopuszcza operacje modyfikacji wierszy tabeli przez inne transakcje, ale nie pozwala na założenie blokady wyłącznej na tabeli
- Przykład wykorzystania: zapobieganie zmianom struktury tabeli

Blokada relacji: EXCLUSIVE (X)

- Zakładanie blokady:

```
LOCK TABLE <nazwa_relacji> IN EXCLUSIVE MODE;
```

- Najbardziej restrykcyjna
  - Tylko jedna transakcja może uzyskać dla danej relacji taką blokadę
  - Żadna inna blokada nie jest kompatybilna – inne transakcje mogą tylko czytać
- Zakładana automatycznie np. w czasie wykonywania poleceń DDL
- Cel stosowania: utrzymanie zawartości relacji przez całą transakcję z zapewnieniem natychmiastowej możliwości zmiany danych relacji, serializacja zapisów do relacji

Blokada relacji: ROW EXCLUSIVE (RX)

- Alternatywna nazwa: Subexclusive Table Lock (SX)
- Zakładanie blokady:
  - niejawne (automatyczne)

```
INSERT INTO <nazwa_relacji> …
UPDATE <nazwa_relacji> …
DELETE FROM <nazwa_relacji> …
SELECT … FROM <nazwa_relacji> … FOR UPDATE; -- od 10g r2
```

  - jawne

```
LOCK TABLE <nazwa_relacji> IN ROW EXCLUSIVE MODE;
```

- Transakcja zmodyfikowała lub zamierza zmodyfikować wiersze
- Nie pozwala innym transakcjom na założenie blokady wyłącznej lub blokad uniemożliwiających modyfikację relacji

Blokada relacji: SHARE (S)

- **Zakładanie blokady:**

  LOCK TABLE <nazwa_relacji> **IN SHARE MODE**;

- **Uniemożliwia wykonywanie operacji modyfikacji na danych relacji**
- **Pozwala innym transakcjom na założenie takiej samej blokady**
- **Cel stosowania: utrzymanie zawartości relacji przez całą transakcję (bez zamiaru modyfikacji danych tej relacji w ramach transakcji)**

Blokada relacji: SHARE ROW EXCLUSIVE (SRX)

- **Alternatywna nazwa: Share-Subexclusive Table Lock (SSX)**
- **Zakładanie blokady:**

  LOCK TABLE <nazwa_relacji> **IN SHARE ROW EXCLUSIVE MODE**;

- **Podobna do blokady SHARE, ale może być założona tylko przez jedną transakcję**
- **Cel stosowania: utrzymanie zawartości relacji przez całą transakcję (z zamiarem modyfikacji danych tej relacji w ramach transakcji)**

Blokady

Blokada założona: Polecenie	Typ blokady	Blokada zakładana: RS	RX	S	SRX	X
SELECT … FROM r	-	TAK	TAK	TAK	TAK	TAK
LOCK TABLE r IN **ROW SHARE** MODE	RS	TAK	TAK	TAK	TAK	NIE
INSERT INTO r …	RX	TAK	TAK	NIE	NIE	NIE
UPDATE r …	RX	TAK	TAK	NIE	NIE	NIE
DELETE FROM r …	RX	TAK	TAK	NIE	NIE	NIE
SELECT … FROM r … FOR UPDATE	RX	TAK	TAK	NIE	NIE	NIE
LOCK TABLE r IN **ROW EXCLUSIVE** MODE	RX	TAK	TAK	NIE	NIE	NIE
LOCK TABLE r IN **SHARE** MODE	S	TAK	NIE	TAK	NIE	NIE
LOCK TABLE r IN **SHARE ROW EXCLUSIVE** MODE	SRX	TAK	NIE	NIE	NIE	NIE
LOCK TABLE r IN **EXCLUSIVE** MODE	X	NIE	NIE	NIE	NIE	NIE

r – nazwa relacji TAK - oczekiwanie, jeżeli są konflikty na poziomie wierszy

Blokady DML – przykład

T1	T2
SQL> update pracownicy set placa_pod = placa_pod + 100 where nazwisko = 'Kowalski'; ***Zał. TX na rekordach nazwisko='Kowalski'*** ***Zał. TM (RX) na relacji PRACOWNICY***	
	SQL> update pracownicy set placa_pod = placa_pod + 200 where nazwisko = 'Nowak'; ***Zał. TX na rekordach nazwisko='Nowak'*** ***Zał. TM (RX) na relacji PRACOWNICY***
SQL> update pracownicy set etat = 'ASYSTENT' where nazwisko = 'Nowak'; ***oczekiwanie na zwolnienie TX na rekordach nazwisko='Nowak'*** ***zał. TX na rekordach nazwisko='Nowak'***	
	SQL> commit; ***Zwolnienie blokad TX i TM***
SQL> commit; ***Zwolnienie blokad TX i TM***	

Blokady DML a klucze obce

- Przy operacji, modyfikującej wartości klucza podstawowego w relacji nadrzędnej, zostaje (chwilowo) założona blokada S na relację podrzędną
- Blokada na relacji podrzędnej zostaje zdjęta zaraz po zakończeniu polecenia modyfikującego dane relacji nadrzędnej

T1
SQL> update zespoly set id_zesp = 80 where id_zesp = 70; ***Założenie TX na rekordach id_zesp=70*** ***Założenie TM (RX) na relacji ZESPOLY*** ***Założenie TM (S) na relacji PRACOWNICY***
1 wiersz został zmodyfikowany. ***Zdjęcie TM (S) na relacji PRACOWNICY***

- Zapobieganie blokadzie – założenie indeksu na kluczu obcym w relacji podrzędnej, blokowany jest indeks

Blokady DDL

- Blokady słownika danych (ang. *Data Dictionary Lock*)
- Zabezpieczają strukturę obiektów bazodanowych podczas operacji zmiany struktury (operacje DDL)
- Zakładane przez transakcję definiującą/modyfikującą obiekty (transakcję DDL)
- Zwalniane bezpośrednio po zakończeniu transakcji
- Typy blokad DDL:
  - wyłączne – np. dla poleceń ALTER TABLE, DROP TABLE
  - współdzielone – np. dla polecenia CREATE PROCEDURE, zablokowane blokadami współdzielonymi zostają relacje, użyte w kodzie procedury
  - parsingu – zakładane na obiekcie podczas wykonania polecenia SQL lub PL/SQL

Zakleszczenie w bazie danych

- Co najmniej dwie transakcje czekają wzajemnie na zwolnienie blokad, przez co żadna z nich nie może zakończyć działania
- SZBD Oracle wykrywa zakleszczenie i przerywa polecenie w tej transakcji, która wykryła zakleszczenie (obie transakcje pozostają aktywne!)

T1	T2
SQL> update pracownicy set placa_pod = 2000 where id_prac = 200; 1 wiersz został zmodyfikowany.	
	SQL> update zespoly set adres = 'Puławska 1' where nazwa = 'BADANIA'; 1 wiersz został zmodyfikowany.
SQL> update zespoly set nazwa = 'NOWE BADANIA' where nazwa = 'BADANIA';	
	SQL> delete pracownicy where id_prac = 200;

Wiadomości uzupełniające

Transakcja autonomiczna

- **Niezależna transakcja, wywoływana z innej transakcji, nazywanej transakcją główną**
- **Cechy:**
  - transakcja autonomiczna nie widzi zmian wprowadzonych przez jej transakcję główną, nie współdzieli również z nią żadnych blokad i zasobów,
  - zmiany, wprowadzone w ramach transakcji autonomicznej, są widoczne dla innych transakcji po jej zatwierdzeniu, niezależnie od tego, czy transakcja główna została zatwierdzona,
  - autonomiczna transakcja może uruchomić inną autonomiczną transakcję (stać się jej transakcją główną); nie ma ograniczenia poziomów wywołań transakcji autonomicznych.
- **Składnia:**

```
CREATE PROCEDURE | FUNCTION … IS
  PRAGMA AUTONOMOUS_TRANSACTION;
…
```


Przykład transakcji autonomicznej

```
DECLARE
  v_zmienna NUMBER := 1;
BEGIN
  /* początek transakcji głównej */
  INSERT INTO …
  UPDATE …
  v_zmienna := fun(111);
  SELECT …
  COMMIT;
  /* zakończenie transakcji głównej */
END;
```

```
CREATE FUNCTION fun(p_id NUMBER)
  RETURN NUMBER IS
  PRAGMA autonomous_transaction;
BEGIN
  /* zawieszenie transakcji głównej */
  /* początek transakcji autonomicznej */
  DELETE …
  UPDATE …
  COMMIT;
  /* zakończenie transakcji autonomicznej */
  /* wznowienie transakcji głównej */
END fun;
```


Odroczone ograniczenia integralnościowe (1)

- **Sprawdzane w momencie zatwierdzenia transakcji**
- **Definiowanie:**

```
[ CONSTRAINT nazwa_ograniczenia ] ograniczenie
DEFERRABLE
INITIALLY { IMMEDIATE | DEFERRED }
```

- **Określenie momentu sprawdzenia w bieżącej transakcji (tylko dla ograniczeń zdefiniowanych jako odroczone):**

```
SET CONSTRAINTS { <lista_ograniczeń> | ALL }
{ DEFERRED | IMMEDIATE };
```

```
ALTER TABLE nazwa_relacji MODIFY
{CONSTRAINT nazwa_ograniczenia | PRIMARY KEY |
UNIQUE(lista_kolumn)} INITIALLY {DEFERRED | IMMEDIATE};
```


Odroczone ograniczenia integralnościowe (2)

- **Przykład:**

```
CREATE TABLE osoby(
        id NUMBER(4) PRIMARY KEY
                DEFERRABLE INITIALLY DEFERRED,
        nazwisko VARCHAR2(100) NOT NULL);
INSERT INTO osoby VALUES(1, 'Nowacki');
INSERT INTO osoby VALUES(1, 'Kowalski');
SELECT * FROM osoby;
COMMIT;
```

Podsumowanie

- **Transakcja to sekwencja atomowych operacji w bazie danych. Własności transakcji tworzą czwórkę ACID.**
- **Zagrożeniem spójności bazy danych są m.in. współbieżne operacje, realizowane na danych w bazie.**
- **Poziomy izolacji transakcji określają, jakie anomalie mogą wystąpić przy współbieżnym dostępie do danych.**
- **Blokady są mechanizmem, wykorzystywanym do zabezpieczenia spójności danych i struktury obiektów.**

Bibliografia

- H. Berenson, P. Bernstein. J. Gray, j. Melton, E. O'Neal, P. O'Neal. A Critique of ANSI SQL Isolation Levels. SIGMOD'95
- Dokumentacja techniczna SZBD Oracle, wer. 11.2. http://oracle.com
- T. Koszlajda. Zarządzanie współbieżnością transakcji. PLOUG'2000.