

МІНІСТЕРСТВО ОСВІТИ Й НАУКИ УКРАЇНИ  
ОДЕСЬКИЙ НАЦІОНАЛЬНИЙ ПОЛІТЕХНІЧНИЙ УНІВЕРСИТЕТ  
Інститут комп'ютерних систем  
Кафедра інформаційних систем

Лабораторна робота № 10  
За дисципліною: "Операційні системи"  
**Тема: «Керування процесами-транзакціями в базах даних.  
Частина 2»**

Виконала:  
Студентка групи AI-205  
Алексеева Аліна  
Перевірили:  
Блажко О.А.  
Дрозд М.О.

Одеса 2021

**Мета роботи:** дослідити поведінку процесів-транзакцій в базах даних та засоби керування ними через механізм блокування з використанням сучасних систем керування базами даних.

## 2 Завдання

Для кожної транзакції підготуйте окремий термінал, в якому виконайте команду доступу до вашої БД з використанням утиліти `psql`.

### Завдання 1. Аналіз роботи багато версійного протоколу

В завданні 1 рішення попередньої лабораторної роботи було створено таблицю з декількома рядками.

Підготуйте чотири транзакції за прикладом з рисунку 2:

– T1 – отримання номеру транзакції, внесення нового рядка в таблицю та перегляд вмісту таблиці;

```
START TRANSACTION
alekseeva_alina=> select txid_current();
txid_current
-----
3161
(1 row)

alekseeva_alina=> insert into auto values (3,'Kia Rio',2015);
INSERT 0 1
alekseeva_alina=> select xmin,xmax,* from auto;
xmin | xmax | a_id | name | year
-----+-----+-----+-----+-----
2194 | 3100 | 2 | Audi A4 | 2018
3116 | 0 | 1 | BMW 5 | 2003
3161 | 0 | 3 | Kia Rio | 2015
(3 rows)

alekseeva_alina=> commit;
COMMIT
alekseeva_alina=>
```

3) T1 фіксує зміни;

4) T2 отримує значення

```
START TRANSACTION
xmin | xmax | a_id | name | year
-----+-----+-----+-----+-----
2194 | 3100 | 2 | Audi A4 | 2018
3116 | 0 | 1 | BMW 5 | 2003
(2 rows)

alekseeva_alina=> select xmin,xmax,* from auto;
xmin | xmax | a_id | name | year
-----+-----+-----+-----+-----
2194 | 3100 | 2 | Audi A4 | 2018
3116 | 0 | 1 | BMW 5 | 2003
(2 rows)

alekseeva_alina=> select xmin,xmax,* from auto;
xmin | xmax | a_id | name | year
-----+-----+-----+-----+-----
2194 | 3100 | 2 | Audi A4 | 2018
3116 | 0 | 1 | BMW 5 | 2003
3161 | 0 | 3 | Kia Rio | 2015
(3 rows)
```

alekseeva\_alina=>

– T2 – постійний перегляд вмісту таблиці

**Використовую команду:**

`select xmin,xmax,* from auto;`

– T3 – видалення рядку з наступною відміною цієї операції;

```
alekseeva_alina=> rollback;
ROLLBACK
alekseeva_alina=> start transaction;
START TRANSACTION
alekseeva_alina=> select xmin,xmax,* from auto;
xmin | xmax | a_id | name | year
-----+-----+-----+-----+-----
3116 | 0 | 1 | BMW 5 | 2003
3161 | 3170 | 3 | Kia Rio | 2015
3169 | 0 | 2 | Audi A4 | 2021
(3 rows)

alekseeva_alina=> delete from auto where a_id=1;
DELETE 1
alekseeva_alina=> rollback;
ROLLBACK
alekseeva_alina=>
```

```
alekseeva_alina=> select xmin,xmax,* from auto;
xmin | xmax | a_id | name | year
-----+-----+-----+-----+-----
3116 | 3171 | 1 | BMW 5 | 2003
3161 | 3170 | 3 | Kia Rio | 2015
3169 | 0 | 2 | Audi A4 | 2021
(3 rows)

alekseeva_alina=> select xmin,xmax,* from auto;
xmin | xmax | a_id | name | year
-----+-----+-----+-----+-----
3116 | 3171 | 1 | BMW 5 | 2003
3161 | 3170 | 3 | Kia Rio | 2015
3169 | 0 | 2 | Audi A4 | 2021
(3 rows)

alekseeva_alina=>
```

– T4 – зміна значення однієї з колонок рядка.

```
alekseeva_alina=> start transaction;
START TRANSACTION
alekseeva_alina=> select xmin,xmax,* from auto;
xmin | xmax | a_id | name | year
-----+-----+-----+-----+-----
3172 | 0 | 2 | Audi A4 | 2020
3116 | 3171 | 1 | BMW 5 | 2003
3161 | 3170 | 3 | Kia Rio | 2015
(3 rows)

alekseeva_alina=> update auto set year =2018 where a_id=2;
UPDATE 1
alekseeva_alina=> select xmin,xmax,* from auto;
xmin | xmax | a_id | name | year
-----+-----+-----+-----+-----
3173 | 0 | 2 | Audi A4 | 2018
3116 | 3171 | 1 | BMW 5 | 2003
3161 | 3170 | 3 | Kia Rio | 2015
(3 rows)

alekseeva_alina=> commit;
COMMIT
alekseeva_alina=> █
```

```
(3 rows)
alekseeva_alina=> select xmin,xmax,* from auto;
xmin | xmax | a_id | name | year
-----+-----+-----+-----+-----
3172 | 3173 | 2 | Audi A4 | 2020
3116 | 3171 | 1 | BMW 5 | 2003
3161 | 3170 | 3 | Kia Rio | 2015
(3 rows)

alekseeva_alina=> select xmin,xmax,* from auto;
xmin | xmax | a_id | name | year
-----+-----+-----+-----+-----
3173 | 0 | 2 | Audi A4 | 2018
3116 | 3171 | 1 | BMW 5 | 2003
3161 | 3170 | 3 | Kia Rio | 2015
(3 rows)

alekseeva_alina=> █
```

voyakovskij\_dmitro=> start transa

**В операцію читання рядка таблиці додайте системні колонки xmin, xmax. На кожному кроці виконання транзакції переглядайте значення колонок xmin, xmax та зробіть відповідні висновки.**

Виходить так, що з однієї транзакції ми можемо переглядати стан певного рядка при роботі іншої транзакції, до і після commit або rollback в залежності від операції значення xmax може бути різним.

## Завдання 2. Аналіз стану транзакцій на різних рівнях багаторівневого блокування

Виконайте послідовно в двох терміналах наступні комбінації блокувань таблиці: IX-IS, SIX-IX, SIX-IS. Надайте висновки про сумісність блокувань.

```
START TRANSACTION
alekseeva_alina=> lock table auto in row exclusive mode;
LOCK TABLE
alekseeva_alina=> █
```

alekseeva\_alina@vpsj3leQ:~

```
alekseeva_alina=> start transaction;
START TRANSACTION
alekseeva_alina=> lock table auto in row share mode;
LOCK TABLE
alekseeva_alina=> █
```

alekseeva\_alina@vpsj3leQ:~

relation	locktype	virtualtransaction	pid	mode	granted
16651	relation	3/95739	29749	RowExclusiveLock	t
16651	relation	2/906649	29655	RowShareLock	t
11673	relation	4/162721	30807	AccessShareLock	t

(3 rows)

```

alekseeva_alina=> start transaction;
START TRANSACTION
alekseeva_alina=> lock table auto in share row exclusive mode;
LOCK TABLE
alekseeva_alina=> 

```

```

alekseeva_alina@vpsj3leQ:~

```

```

ROLLBACK
alekseeva_alina=> start transaction;
START TRANSACTION
alekseeva_alina=> lock table auto in row exclusive mode;

```

```

alekseeva_alina@vpsj3leQ:~

```

relation	locktype	virtualtransaction	pid	mode	gran
11673	relation	4/162721	30807	AccessShareLock	t
16651	relation	3/95740	29749	ShareRowExclusiveLock	t
16651	relation	2/906650	29655	RowExclusiveLock	f

```

--More--

```

```

alekseeva_alina=> start transaction;
START TRANSACTION
alekseeva_alina=> lock table auto in share row exclusive mode;
LOCK TABLE
alekseeva_alina=> 

```

```

alekseeva_alina@vpsj3leQ:~

```

```

alekseeva_alina=> start transaction;
START TRANSACTION
alekseeva_alina=> lock table auto in row share mode;
LOCK TABLE
alekseeva_alina=> 

```

relation	locktype	virtualtransaction	pid	mode	granted
11673	relation	4/162721	30807	AccessShareLock	t
16651	relation	3/95742	29749	ShareRowExclusiveLock	t
16651	relation	2/906653	29655	RowShareLock	t

```

--More--

```

Для кожної комбінації блокувань перед завершенням 1-ї транзакції (яка розпочалася раніше) в додатковому терміналі через команду `psql` отримайте данні про стан транзакцій (таблиця `pg_locks`).

### Завдання 3. Керування квазіпаралельним виконанням транзакцій на різних рівнях ізоляції транзакцій

Підготуйте транзакції, які було створено у завданні 3.1 рішення попередньої лабораторної роботи, а саме, створіть дві транзакції, кожна з яких повинна включати такі операції:

- операція читання першого рядку таблиці;

```
alekseeva_alina=> select * from auto where a_id=1;
```

a_id	name	year
1	BMW 5	2003

(1 row)

- операція редагування однієї із змінних таблиці в першому рядку;

```
alekseeva_alina=> update auto set year = 2004 where a_id=1;
UPDATE 1
alekseeva_alina=> select * from auto where a_id=1;
```

a_id	name	year
1	BMW 5	2004

(1 row)

- повторна операція читання першого рядку таблиці;

```
alekseeva_alina=> select * from auto where a_id=1;
```

a_id	name	year
1	BMW 5	2004

(1 row)

- операція фіксації всіх змін.  
commit;

1.1 Виконайте роботу транзакцій при умові їх роботи на рівні ізоляції READ COMMITTED. Проаналізуйте реакцію СКБД на операцію UPDATE 2-ї транзакції (яка виконується пізніше) та дайте свої висновки.

<pre>alekseeva_alina=&gt; start transaction; START TRANSACTION alekseeva_alina=&gt; set transaction isolation level read committed; SET alekseeva_alina=&gt; select * from auto where a_id=1;</pre> <table border="1"> <thead> <tr> <th>a_id</th> <th>name</th> <th>year</th> </tr> </thead> <tbody> <tr> <td>1</td> <td>BMW 5</td> <td>2003</td> </tr> </tbody> </table> <p>(1 row)</p> <pre>alekseeva_alina=&gt; update auto set year = 2003 where a_id=1; UPDATE 1 alekseeva_alina=&gt; select * from auto where a_id=1;</pre> <table border="1"> <thead> <tr> <th>a_id</th> <th>name</th> <th>year</th> </tr> </thead> <tbody> <tr> <td>1</td> <td>BMW 5</td> <td>2003</td> </tr> </tbody> </table> <p>(1 row)</p> <pre>alekseeva_alina=&gt; commit; COMMIT alekseeva_alina=&gt;</pre>	a_id	name	year	1	BMW 5	2003	a_id	name	year	1	BMW 5	2003	<pre>alekseeva_alina=&gt; start transaction; START TRANSACTION alekseeva_alina=&gt; set transaction isolation level read committed; SET alekseeva_alina=&gt; select * from auto where a_id=1;</pre> <table border="1"> <thead> <tr> <th>a_id</th> <th>name</th> <th>year</th> </tr> </thead> <tbody> <tr> <td>1</td> <td>BMW 5</td> <td>2003</td> </tr> </tbody> </table> <p>(1 row)</p> <pre>alekseeva_alina=&gt; update auto set year = 2004 where a_id=1; UPDATE 1 alekseeva_alina=&gt; select * from auto where a_id=1;</pre> <table border="1"> <thead> <tr> <th>a_id</th> <th>name</th> <th>year</th> </tr> </thead> <tbody> <tr> <td>1</td> <td>BMW 5</td> <td>2004</td> </tr> </tbody> </table> <p>(1 row)</p> <pre>alekseeva_alina=&gt; commit; COMMIT alekseeva_alina=&gt;</pre>	a_id	name	year	1	BMW 5	2003	a_id	name	year	1	BMW 5	2004
a_id	name	year																							
1	BMW 5	2003																							
a_id	name	year																							
1	BMW 5	2003																							
a_id	name	year																							
1	BMW 5	2003																							
a_id	name	year																							
1	BMW 5	2004																							

При спробі другої транзакції змінити значення змінної 2а транзакція йде в режим waiting, це пов'язано з тим, що транзакції вказується, що ця змінна зараз не доступна, але не йдеться про те, що вона не доступна через те, що



зазнала змін з боку першої транзакції, і ці зміни ще не набули чинності і, що ці зміни важливі для 1-ої транзакції. В даному випадку рівень ізоляції read committed ролі не грає, так як waiting пропадає після завершення першої транзакції.

**1.2 Повторіть роботу транзакцій при умові їх роботи на рівні ізоляції REPEATABLE READ. Проаналізуйте реакцію СКБД на операцію UPDATE 2-ї транзакції (яка виконується пізніше) та дайте свої висновки.**

```
START TRANSACTION
alekseeva_alina=> set transaction isolation level repeatable read;
SET
alekseeva_alina=> select * from auto where a_id=1;
a_id |      name      | year
-----+-----+-----
1 | BMW 5          | 2004
(1 row)

alekseeva_alina=> update auto set year = 2003 where a_id=1;
UPDATE 1
alekseeva_alina=> select * from auto where a_id=1;
a_id |      name      | year
-----+-----+-----
1 | BMW 5          | 2003
(1 row)

alekseeva_alina=> commit;
COMMIT
alekseeva_alina=> █
```

```
(1 row)
alekseeva_alina=> rollback;
ROLLBACK
alekseeva_alina=> start transaction;
START TRANSACTION
alekseeva_alina=> set transaction isolation level repeatable read;
SET
alekseeva_alina=> select * from auto where a_id=1;
a_id |      name      | year
-----+-----+-----
1 | BMW 5          | 2004
(1 row)

alekseeva_alina=> update auto set year = 2004 where a_id=1;
ERROR:  could not serialize access due to concurrent update
alekseeva_alina=> select * from auto where a_id=1;
ERROR:  current transaction is aborted, commands ignored until end
of transaction block
alekseeva_alina=> █
```

**1.3 Повторіть роботу транзакцій при умові їх роботи на рівні ізоляції SERIALIZABLE. Проаналізуйте реакцію СКБД на операцію UPDATE 2-ї транзакції (яка виконується пізніше) та дайте свої висновки.**

```
alekseeva_alina=> start transaction;
START TRANSACTION
alekseeva_alina=> set transaction isolation level serializable;
SET
alekseeva_alina=> select * from auto where a_id=1;
a_id |      name      | year
-----+-----+-----
1 | BMW 5          | 2003
(1 row)

alekseeva_alina=> update auto set year = 2005 where a_id=1;
UPDATE 1
alekseeva_alina=> select * from auto where a_id=1;
a_id |      name      | year
-----+-----+-----
1 | BMW 5          | 2005
(1 row)

alekseeva_alina=> commit;
COMMIT
```

```
(1 row)
alekseeva_alina=> rollback;
ROLLBACK
alekseeva_alina=> start transaction;
START TRANSACTION
alekseeva_alina=> set transaction isolation level serializable;
SET
alekseeva_alina=> select * from auto where a_id=1;
a_id |      name      | year
-----+-----+-----
1 | BMW 5          | 2003
(1 row)

alekseeva_alina=> update auto set year = 2006 where a_id=1;
ERROR:  could not serialize access due to concurrent update
alekseeva_alina=> select * from auto where a_id=1;
ERROR:  current transaction is aborted, commands ignored until end
of transaction block
```

**Завдання 4. Керування квазіпаралельним виконанням транзакцій при наявності тупикових ситуацій.**

**3.1 Виконайте модифікацію транзакцій так, щоб вони призводили до тупикової ситуації.**

```

alekseeva_alina=> start transaction;
START TRANSACTION
alekseeva_alina=> select * from auto;
a_id |      name      | year
-----+-----+-----
2 | Audi A4        | 2018
1 | BMW 5          | 2005
3 | Kia Rio        | 2015
(3 rows)

alekseeva_alina=> update auto set year=2020 where a_id=3;
UPDATE 1
alekseeva_alina=> update auto set year=2015 where a_id=2;
UPDATE 1
alekseeva_alina=> select * from auto;
a_id |      name      | year
-----+-----+-----
1 | BMW 5          | 2005
3 | Kia Rio        | 2020
2 | Audi A4        | 2015
(3 rows)

```

```

alekseeva_alina=> rollback;
ROLLBACK
alekseeva_alina=> start transaction;
START TRANSACTION
alekseeva_alina=> select * from auto;
a_id |      name      | year
-----+-----+-----
2 | Audi A4        | 2018
1 | BMW 5          | 2005
3 | Kia Rio        | 2015
(3 rows)

alekseeva_alina=> update auto set year=2010 where a_id=2;
UPDATE 1
alekseeva_alina=> update auto set year=2012 where a_id=3;
ERROR:  deadlock detected
DETAIL:  Process 29749 waits for ShareLock on transaction 3563; blocked by process 29749.
Process 29749 waits for ShareLock on transaction 3564; blocked by process 29749.
HINT:  See server log for query details.
CONTEXT:  while updating tuple (0,53) in relation "auto"

```

**Висновок:** в ході цієї лабораторної роботи ми дослідити поведінку процесів-транзакцій в базах даних та засоби керування ними через механізм блокування з використанням сучасних систем керування базами даних.