

ДЕРЖАВНИЙ УНІВЕРСИТЕТ «ОДЕСЬКА ПОЛІТЕХНІКА»
ІНСТИТУТ КОМП'ЮТЕРНИХ СИСТЕМ
КАФЕДРА «ІНФОРМАЦІЙНИ СИСТЕМИ»

Лабораторна робота №10

з дисципліни

«Операційні системи»

Тема

«Керування процесами-транзакціями в базах даних. Частина 2»

Виконав:

Студент групи AI-202

Узун Михайло

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.

Завдання

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

Завдання 1. Аналіз роботи багатоверсійного 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;
- T2 – постійний перегляд вмісту таблиці
- T3 – видалення рядку з наступною відміною цієї операції;
- T4 – зміна значення однієї з колонок рядка.

В операцію читання рядка таблиці додайте системні колонки `xmin`, `xmax`.

На кожному кроці виконання транзакції переглядайте значення колонок `xmin`, `xmax` та зробіть відповідні висновки.

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. 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;
- операція редагування однієї із змінних таблиці в першому рядку;

- повторна операція читання першого рядку таблиці;
- операція фіксації всіх змін.

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.

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.

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.

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

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

3.2 Виконайте дві модифіковані транзакції.

Проаналізуйте реакцію СКБД на операцію UPDATE 2-ї транзакції (яка виконується пізніше) та яка призвела до тупику. Дайте свої висновки з урахуванням:

- ідентифікаторів процесів
- номерів транзакцій.

Хід роботи:

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;
- T2 – постійний перегляд вмісту таблиці
- T3 – видалення рядку з наступною відміною цієї операції;
- T4 – зміна значення однієї з колонок рядка.

В операцію читання рядка таблиці додайте системні колонки xmin, xmax.

На кожному кроці виконання транзакції переглядайте значення колонок xmin, xmax та зробіть відповідні висновки.

T1:

```
psql (9.5.25)
Type "help" for help.

uzun_mihajlo=> START TRANSACTION;
START TRANSACTION
uzun_mihajlo=> SELECT txid_current();
 txid_current
-----
          3313
(1 row)

uzun_mihajlo=> INSERT INTO auto VALUES(3, 'Ford', '2007');
INSERT 0 1
uzun_mihajlo=> SELECT xmin, xmax, * from auto;
 xmin | xmax | a_id |      name      | year
-----+-----+-----+-----+-----
 2470 |    0 |    2 | Mercedes      | 2010
 2479 | 2489 |    1 | Toyota        | 2004
 3313 |    0 |    3 | Ford          | 2007
(3 rows)

uzun_mihajlo=> COMMIT;
COMMIT
uzun_mihajlo=> █
```

T3:

```
[uzun_mihajlo@vpsj3IeQ ~]$ psql uzun_mihajlo
psql (9.5.25)
Type "help" for help.

uzun_mihajlo=> START TRANSACTION;
START TRANSACTION
uzun_mihajlo=> DELETE FROM auto WHERE a_id = 3;
DELETE 1
uzun_mihajlo=> ROLLBACK;
ROLLBACK
uzun_mihajlo=> █
```

T4:

```
[uzun_mihajlo@vpsj3IeQ ~]$ psql uzun_mihajlo
psql (9.5.25)
Type "help" for help.
```

```
uzun_mihajlo=> START TRANSACTION;
START TRANSACTION
uzun_mihajlo=> UPDATE auto SET name = 'Honda' WHERE a_id = 3;
UPDATE 1
uzun_mihajlo=> COMMIT;
COMMIT
uzun_mihajlo=> █
```

T2:

```
uzun_mihajlo=> START TRANSACTION;
START TRANSACTION
uzun_mihajlo=> SELECT xmin, xmax, * from auto;
  xmin | xmax | a_id |      name      | year
-----+-----+-----+-----+-----
  2470 |    0 |    2 | Mercedes       | 2010
  2479 | 2489 |    1 | Toyota         | 2004
(2 rows)
```

```
uzun_mihajlo=> SELECT xmin, xmax, * from auto;
  xmin | xmax | a_id |      name      | year
-----+-----+-----+-----+-----
  2470 |    0 |    2 | Mercedes       | 2010
  2479 | 2489 |    1 | Toyota         | 2004
(2 rows)
```

```
uzun_mihajlo=> SELECT xmin, xmax, * from auto;
  xmin | xmax | a_id |      name      | year
-----+-----+-----+-----+-----
  2470 |    0 |    2 | Mercedes       | 2010
  2479 | 2489 |    1 | Toyota         | 2004
  3313 |    0 |    3 | Ford           | 2007
(3 rows)
```

```
uzun_mihajlo=> SELECT xmin, xmax, * from auto;
  xmin | xmax | a_id |      name      | year
-----+-----+-----+-----+-----
  2470 |    0 |    2 | Mercedes       | 2010
  2479 | 2489 |    1 | Toyota         | 2004
  3313 | 3314 |    3 | Ford           | 2007
(3 rows)
```

```
uzun_mihajlo=> SELECT xmin, xmax, * from auto;
  xmin | xmax | a_id |      name      | year
-----+-----+-----+-----+-----
  2470 |    0 |    2 | Mercedes       | 2010
  2479 | 2489 |    1 | Toyota         | 2004
  3313 | 3315 |    3 | Ford           | 2007
(3 rows)
```

```
uzun_mihajlo=> SELECT xmin, xmax, * from auto;
  xmin | xmax | a_id |      name      | year
-----+-----+-----+-----+-----
  2470 |    0 |    2 | Mercedes       | 2010
  2479 | 2489 |    1 | Toyota         | 2004
  3315 |    0 |    3 | Honda          | 2007
(3 rows)
```

```
uzun_mihajlo=> COMMIT;
COMMIT
uzun_mihajlo=> █
```

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.

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`).

IX-IS

```
uzun_mihajlo@vpsj3IeQ:~  
[uzun_mihajlo@vpsj3IeQ ~]$ psql узун_мигајло  
psql (9.5.25)  
Type "help" for help.  
  
uzun_mihajlo=> START TRANSACTION;  
START TRANSACTION  
uzun_mihajlo=> lock table auto in row exclusive mode;  
LOCK TABLE  
uzun_mihajlo=>   
  
uzun_mihajlo@vpsj3IeQ:~  
[uzun_mihajlo@vpsj3IeQ ~]$ psql узун_мигајло  
psql (9.5.25)  
Type "help" for help.  
  
uzun_mihajlo=> START TRANSACTION;  
START TRANSACTION  
uzun_mihajlo=> lock table auto in row share mode;  
LOCK TABLE  
uzun_mihajlo=>   
  
uzun_mihajlo=> SELECT relation,locktype,virtualtransaction,pid,mode,granted FROM  
pg_locks WHERE locktype = 'relation';  
  relation | locktype | virtualtransaction | pid |      mode      | granted  
-----+-----+-----+-----+-----+-----  
    16735 | relation |    2/800631        | 8026 | RowExclusiveLock | t  
    16735 | relation |    3/87026         | 8047 | RowShareLock     | t  
    11673 | relation |    4/108550        | 8859 | AccessShareLock  | t  
(3 rows)  
  
uzun_mihajlo=>   

```

SIX-IX

```
uzun_mihajlo@vpsj3IeQ:~
```

```
[uzun_mihajlo@vpsj3IeQ ~]$ psql uzun_mihajlo
```

```
psql (9.5.25)
```

```
Type "help" for help.
```

```
uzun_mihajlo=> START TRANSACTION;
```

```
START TRANSACTION
```

```
uzun_mihajlo=> lock table auto in share row exclusive mode;
```

```
LOCK TABLE
```

```
uzun_mihajlo=> █
```

```
uzun_mihajlo@vpsj3IeQ:~
```

```
[uzun_mihajlo@vpsj3IeQ ~]$ psql uzun_mihajlo
```

```
psql (9.5.25)
```

```
Type "help" for help.
```

```
uzun_mihajlo=> START TRANSACTION;
```

```
START TRANSACTION
```

```
uzun_mihajlo=> lock table auto in row exclusive mode;
```

```
█
```

```
uzun_mihajlo=> SELECT relation,locktype,virtualtransaction,pid,mode,granted FROM  
pg_locks WHERE locktype = 'relation';
```

```
relation | locktype | virtualtransaction | pid | mode | granted
```

```
-----+-----+-----+-----+-----+-----
```

```
16687 | relation | 7/21581 | 10235 | RowExclusiveLock | t  
11673 | relation | 4/108551 | 8859 | AccessShareLock | t  
16687 | relation | 6/72638 | 9882 | AccessShareLock | t  
16687 | relation | 6/72638 | 9882 | RowExclusiveLock | t  
16735 | relation | 2/800653 | 10112 | ShareRowExclusiveLock | t  
16735 | relation | 3/87038 | 10133 | RowExclusiveLock | f
```

```
(6 rows)
```

```
uzun_mihajlo=> █
```

SIX-IS

```
uzun_mihajlo@vpsj3IeQ:~  
[uzun_mihajlo@vpsj3IeQ ~]$ psql uzun_mihajlo  
psql (9.5.25)  
Type "help" for help.  
  
uzun_mihajlo=> START TRANSACTION;  
START TRANSACTION  
uzun_mihajlo=> lock table auto in share row exclusive mode;  
LOCK TABLE  
uzun_mihajlo=> █
```

```
uzun_mihajlo@vpsj3IeQ:~  
[uzun_mihajlo@vpsj3IeQ ~]$ psql uzun_mihajlo  
psql (9.5.25)  
Type "help" for help.  
  
uzun_mihajlo=> START TRANSACTION;  
START TRANSACTION  
uzun_mihajlo=> lock table auto in row share mode;  
LOCK TABLE  
uzun_mihajlo=> █
```

```
uzun_mihajlo=> SELECT relation,locktype,virtualtransaction,pid,mode,granted FROM  
pg_locks WHERE locktype = 'relation';  
 relation | locktype | virtualtransaction | pid | mode | granted  
-----+-----+-----+-----+-----+-----  
---  
      16687 | relation | 7/21581 | 10235 | RowExclusiveLock | t  
      11673 | relation | 4/108552 | 8859 | AccessShareLock | t  
      16735 | relation | 3/87044 | 10718 | RowShareLock | t  
      16735 | relation | 2/800681 | 10713 | ShareRowExclusiveLock | t  
(4 rows)  
  
uzun_mihajlo=> █
```

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;
- операція редагування однієї із змінних таблиці в першому рядку;
- повторна операція читання першого рядку таблиці;
- операція фіксації всіх змін.

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.

```
uzun_mihajlo@vpsj3IeQ:~  
[uzun_mihajlo@vpsj3IeQ ~]$ psql uzun_mihajlo  
psql (9.5.25)  
Type "help" for help.  
  
uzun_mihajlo=> START TRANSACTION;  
START TRANSACTION  
uzun_mihajlo=> SET TRANSACTION ISOLATION LEVEL READ COMMITTED;  
SET  
uzun_mihajlo=> SELECT * FROM auto WHERE a_id = 1;  
 a_id |      name      | year  
-----+-----+-----  
    1 | Toyota         | 2004  
(1 row)  
  
uzun_mihajlo=> UPDATE auto SET name = 'Toyota2' WHERE a_id = 1;  
UPDATE 1  
uzun_mihajlo=> 
```

```
uzun_mihajlo@vpsj3IeQ:~  
[uzun_mihajlo@vpsj3IeQ ~]$ psql uzun_mihajlo  
psql (9.5.25)  
Type "help" for help.  
  
uzun_mihajlo=> START TRANSACTION;  
START TRANSACTION  
uzun_mihajlo=> SET TRANSACTION ISOLATION LEVEL READ COMMITTED;  
SET  
uzun_mihajlo=> SELECT * FROM auto WHERE a_id = 1;  
 a_id |      name      | year  
-----+-----+-----  
    1 | Toyota         | 2004  
(1 row)  
  
uzun_mihajlo=> UPDATE auto SET name = 'Toyota3' WHERE a_id = 1;  
 
```

Другий термінал переходить до стану очікування.

<pre>uzun_mihajlo=> SELECT * FROM auto WHERE a_id = 1; a_id name year -----+-----+----- 1 Toyota2 2004 (1 row) uzun_mihajlo=> COMMIT; COMMIT uzun_mihajlo=> </pre>	<pre>uzun_mihajlo=> SELECT * FROM auto WHERE a_id = 1; a_id name year -----+-----+----- 1 Toyota3 2004 (1 row) uzun_mihajlo=> COMMIT; COMMIT uzun_mihajlo=> </pre>
---	---

Другий термінал виходить зі стану очікування після COMMIT у першому.

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.

```
uzun_mihajlo@vpsj3IeQ:~  
[uzun_mihajlo@vpsj3IeQ ~]$ psql uzun_mihajlo  
psql (9.5.25)  
Type "help" for help.  
  
uzun_mihajlo=> START TRANSACTION;  
START TRANSACTION  
uzun_mihajlo=> SET TRANSACTION ISOLATION LEVEL REPEATABLE READ;  
SET  
uzun_mihajlo=> SELECT * FROM auto WHERE a_id = 1;  
 a_id |      name      | year  
-----+-----+-----  
    1 | Toyota3        | 2004  
(1 row)  
  
uzun_mihajlo=> UPDATE auto SET name = 'Toyota4' WHERE a_id = 1;  
UPDATE 1  
uzun_mihajlo=> 
```

```
uzun_mihajlo@vpsj3IeQ:~  
[uzun_mihajlo@vpsj3IeQ ~]$ psql uzun_mihajlo  
psql (9.5.25)  
Type "help" for help.  
  
uzun_mihajlo=> START TRANSACTION;  
START TRANSACTION  
uzun_mihajlo=> SET TRANSACTION ISOLATION LEVEL REPEATABLE READ;  
SET  
uzun_mihajlo=> SELECT * FROM auto WHERE a_id = 1;  
 a_id |      name      | year  
-----+-----+-----  
    1 | Toyota3        | 2004  
(1 row)  
  
uzun_mihajlo=> UPDATE auto SET name = 'Toyota5' WHERE a_id = 1;  
 
```

Другий термінал переходить до стану очікування.

```
uzun_mihajlo=> COMMIT; ERROR:  could not serialize access due to concurrent update  
COMMIT  
uzun_mihajlo=> 
```

Друга транзакція преривається після COMMIT у першому.

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.

```

uzun_mihajlo@vpsj3IeQ:~
[uzun_mihajlo@vpsj3IeQ ~]$ psql uzun_mihajlo
psql (9.5.25)
Type "help" for help.

uzun_mihajlo=> START TRANSACTION;
START TRANSACTION
uzun_mihajlo=> SET TRANSACTION ISOLATION LEVEL SERIALIZABLE;
SET
uzun_mihajlo=> SELECT * FROM auto WHERE a_id = 1;
 a_id |      name      | year
-----+-----+-----
    1 | Toyota4        | 2004
(1 row)

uzun_mihajlo=> UPDATE auto SET name = 'Toyota5' WHERE a_id = 1;
UPDATE 1
uzun_mihajlo=> █

uzun_mihajlo@vpsj3IeQ:~
[uzun_mihajlo@vpsj3IeQ ~]$ psql uzun_mihajlo
psql (9.5.25)
Type "help" for help.

uzun_mihajlo=> START TRANSACTION;
START TRANSACTION
uzun_mihajlo=> SET TRANSACTION ISOLATION LEVEL SERIALIZABLE;
SET
uzun_mihajlo=> SELECT * FROM auto WHERE a_id = 1;
 a_id |      name      | year
-----+-----+-----
    1 | Toyota4        | 2004
(1 row)

uzun_mihajlo=> UPDATE auto SET name = 'Toyota6' WHERE a_id = 1;
█

```

Другий термінал переходить до стану очікування.

```

uzun_mihajlo=> COMMIT; ERROR:  could not serialize access due to concurrent update
COMMIT
uzun_mihajlo=> █
uzun_mihajlo=> █

```

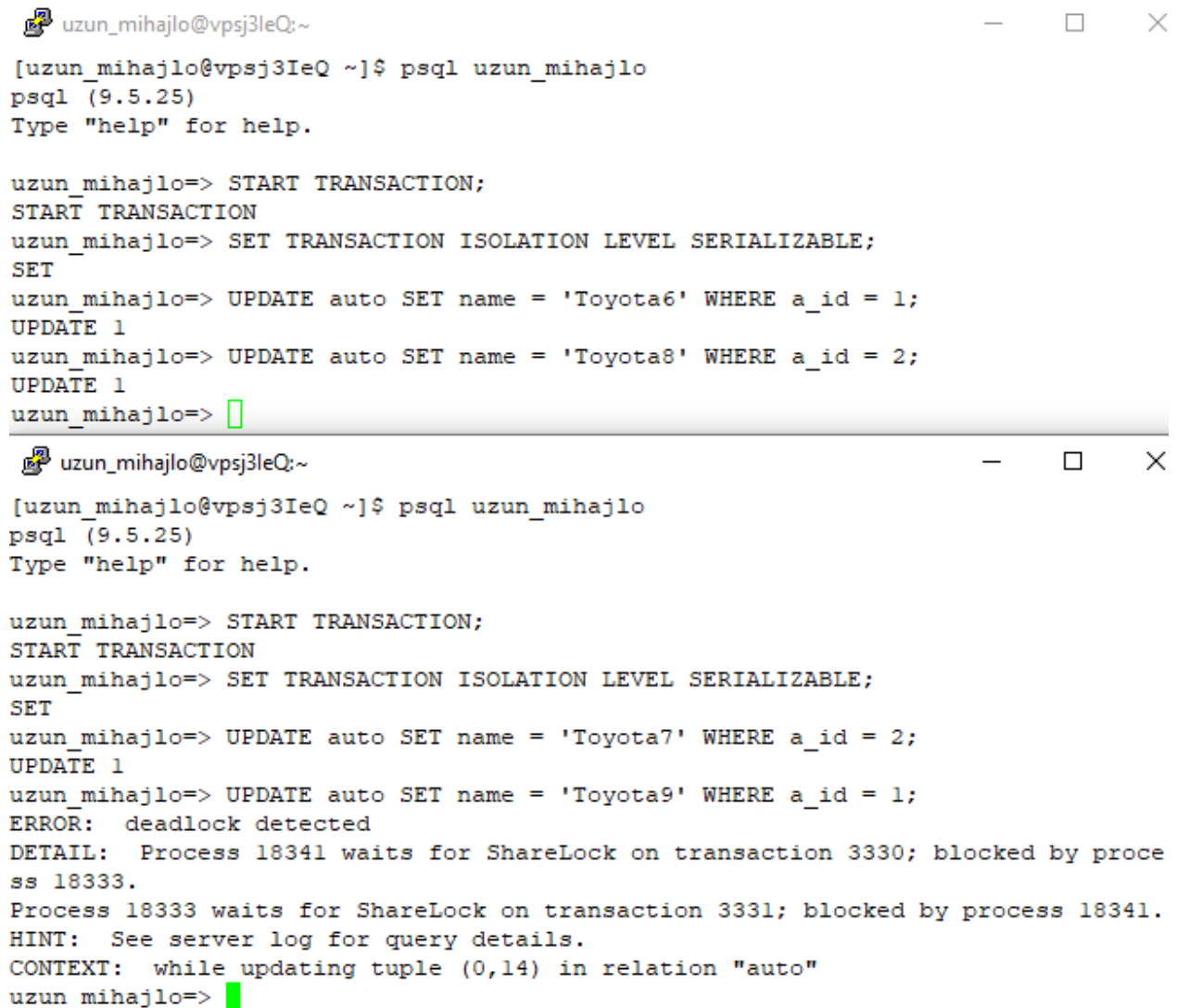
Завдання 4. Керування квазіпаралельним виконанням транзакцій при наявності тупикових ситуацій.

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

3.2 Виконайте дві модифіковані транзакції.

Проаналізуйте реакцію СКБД на операцію UPDATE 2-ї транзакції (яка виконується пізніше) та яка призвела до тупику. Дайте свої висновки з урахуванням:

- ідентифікаторів процесів
- номерів транзакцій.



```
uzun_mihajlo@vpsj3IeQ:~  
[uzun_mihajlo@vpsj3IeQ ~]$ psql узун_михайло  
psql (9.5.25)  
Type "help" for help.  
  
uzun_mihajlo=> START TRANSACTION;  
START TRANSACTION  
uzun_mihajlo=> SET TRANSACTION ISOLATION LEVEL SERIALIZABLE;  
SET  
uzun_mihajlo=> UPDATE auto SET name = 'Toyota6' WHERE a_id = 1;  
UPDATE 1  
uzun_mihajlo=> UPDATE auto SET name = 'Toyota8' WHERE a_id = 2;  
UPDATE 1  
uzun_mihajlo=> █  
  
uzun_mihajlo@vpsj3IeQ:~  
[uzun_mihajlo@vpsj3IeQ ~]$ psql узун_михайло  
psql (9.5.25)  
Type "help" for help.  
  
uzun_mihajlo=> START TRANSACTION;  
START TRANSACTION  
uzun_mihajlo=> SET TRANSACTION ISOLATION LEVEL SERIALIZABLE;  
SET  
uzun_mihajlo=> UPDATE auto SET name = 'Toyota7' WHERE a_id = 2;  
UPDATE 1  
uzun_mihajlo=> UPDATE auto SET name = 'Toyota9' WHERE a_id = 1;  
ERROR:  deadlock detected  
DETAIL:  Process 18341 waits for ShareLock on transaction 3330; blocked by process 18333.  
Process 18333 waits for ShareLock on transaction 3331; blocked by process 18341.  
HINT:  See server log for query details.  
CONTEXT:  while updating tuple (0,14) in relation "auto"  
uzun_mihajlo=> █
```

Транзакція 3330 чекає завершення процесу 18333, щоб виконати процес 18341. Транзакція 3331 чекає завершення процесу 18341, щоб виконати процес 18333.

Транзакції чекають завершення один одного – виникає тупик.

Висновок: була досліджена поведінка процесів-транзакцій в базах даних та засоби керування ними через механізм блокування з використанням сучасних систем керування базами даних.