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

Лабораторна робота №10

З дисципліни: «Операційні системи»

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

Виконав:
Студент групи АІ-202
Лукашак Д. О.

Перевірив:
Блажко О. А.

Одеса 2021

Мета роботи: дослідити поведінку процесів-транзакцій в базах даних та засоби керуванням 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 багато версійного протоколу

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

Підготуйте чотири транзакції за прикладом:

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_locs).

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 COMMITED. 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.

Виконання

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:

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.

Перший термінал:

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

lukashak_daniil=> START TRANSACTION;
START TRANSACTION
lukashak_daniil=> SELECT txid_current();
 txid_current
--------------
         2781
(1 row)

lukashak_daniil=> INSERT INTO teacher VALUES(3, 'Valuev', 'director');
INSERT 0 1
lukashak_daniil=> SELECT xmin, xmax, * from teacher;
 xmin | xmax | t_id |         name         |         post
------+------+------+----------------------+----------------------
 2259 |    0 |    2 | Lukashak             | ne docent
 2339 |    0 |    1 | Ivanovna             | docent
 2781 |    0 |    3 | Valuev               | director
(3 rows)

lukashak_daniil=> COMMIT;
COMMIT
```

Другий термінал:

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

lukashak_daniil=> START TRANSACTION;
START TRANSACTION
lukashak_daniil=> SELECT xmin, xmax, * from teacher;
 xmin | xmax | t_id |         name         |         post
------+------+------+----------------------+----------------------
 2259 |    0 |    2 | Lukashak             | ne docent
 2339 |    0 |    1 | Ivanovna             | docent
(2 rows)

lukashak_daniil=> SELECT xmin, xmax, * from teacher;
 xmin | xmax | t_id |         name         |         post
------+------+------+----------------------+----------------------
 2259 |    0 |    2 | Lukashak             | ne docent
 2339 |    0 |    1 | Ivanovna             | docent
 2781 |    0 |    3 | Valuev               | director
(3 rows)

lukashak_daniil=> SELECT xmin, xmax, * from teacher;
 xmin | xmax | t_id |         name         |         post
------+------+------+----------------------+----------------------
 2259 |    0 |    2 | Lukashak             | ne docent
 2339 |    0 |    1 | Ivanovna             | docent
 2781 | 2782 |    3 | Valuev               | director
(3 rows)

lukashak_daniil=> SELECT xmin, xmax, * from teacher;
 xmin | xmax | t_id |         name         |         post
------+------+------+----------------------+----------------------
 2259 |    0 |    2 | Lukashak             | ne docent
 2339 |    0 |    1 | Ivanovna             | docent
 2781 | 2782 |    3 | Valuev               | director
(3 rows)

lukashak_daniil=> SELECT xmin, xmax, * from teacher;
 xmin | xmax | t_id |         name         |         post
------+------+------+----------------------+----------------------
 2259 |    0 |    2 | Lukashak             | ne docent
 2339 |    0 |    1 | Ivanovna             | docent
 2781 | 2783 |    3 | Valuev               | director
(3 rows)

lukashak_daniil=> SELECT xmin, xmax, * from teacher;
 xmin | xmax | t_id |         name         |         post
------+------+------+----------------------+----------------------
 2259 |    0 |    2 | Lukashak             | ne docent
 2339 |    0 |    1 | Ivanovna             | docent
 2783 |    0 |    3 | NeValuev             | director
(3 rows)
```

Третій термінал:

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

lukashak_daniil=> START TRANSACTION;
START TRANSACTION
lukashak_daniil=> DELETE FROM teacher WHERE t_id = 3;
DELETE 1
lukashak_daniil=> SELECT xmin, xmax, * from teacher;
 xmin | xmax | t_id |         name         |         post
------+------+------+----------------------+----------------------
 2259 |    0 |    2 | Lukashak             | ne docent
 2339 |    0 |    1 | Ivanovna             | docent
(2 rows)

lukashak_daniil=> ROLLBACK;
ROLLBACK
```

Четвертий термінал:

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

lukashak_daniil=> START TRANSACTION;
START TRANSACTION
lukashak_daniil=> UPDATE teacher SET name = 'NeValuev' WHERE t_id = 3;
UPDATE 1
lukashak_daniil=> SELECT xmin, xmax, * from teacher;
 xmin | xmax | t_id |         name         |         post
------+------+------+----------------------+----------------------
 2259 |    0 |    2 | Lukashak             | ne docent
 2339 |    0 |    1 | Ivanovna             | docent
 2783 |    0 |    3 | NeValuev             | director
(3 rows)

lukashak_daniil=> COMMIT;
COMMIT
```

Як ми бачимо: транзакції не бачать зміни інших транзакцій, поки вони не завершаться, але ми можемо бачити id транзакцій, що встигли змінити рядки через стовпці xmin та xmax.

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_locs).

Перший термінал:

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

lukashak_daniil=> START TRANSACTION;
START TRANSACTION
lukashak_daniil=> LOCK TABLE teacher IN ROW EXCLUSIVE MODE;
LOCK TABLE
lukashak_daniil=> COMMIT;
COMMIT
lukashak_daniil=> START TRANSACTION;
START TRANSACTION
lukashak_daniil=> LOCK TABLE teacher IN SHARE ROW EXCLUSIVE MODE;
LOCK TABLE
lukashak_daniil=> COMMIT;
COMMIT
lukashak_daniil=> START TRANSACTION;
START TRANSACTION
lukashak_daniil=> LOCK TABLE teacher IN SHARE ROW EXCLUSIVE MODE;
LOCK TABLE
lukashak_daniil=>
```

Другий термінал:

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

lukashak_daniil=> START TRANSACTION;
START TRANSACTION
lukashak_daniil=> LOCK TABLE teacher IN ROW SHARE MODE;
LOCK TABLE
lukashak_daniil=> COMMIT;
COMMIT
lukashak_daniil=> START TRANSACTION;
START TRANSACTION
lukashak_daniil=> LOCK TABLE teacher IN ROW EXCLUSIVE MODE;
LOCK TABLE
lukashak_daniil=> COMMIT;
COMMIT
lukashak_daniil=> START TRANSACTION;
START TRANSACTION
lukashak_daniil=> LOCK TABLE teacher IN ROW SHARE MODE;
LOCK TABLE
lukashak_daniil=>
```

Третій термінал:

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

lukashak_daniil=> SELECT relation, locktype, virtualtransaction, pid, mode, granted FROM pg_locks WHERE locktype = 'relation';
 relation | locktype | virtualtransaction |  pid  |       mode       | granted
----------+----------+--------------------+-------+------------------+---------
    11673 | relation | 7/14147            | 29935 | AccessShareLock  | t
    16675 | relation | 6/63022            | 29807 | RowShareLock     | t
    16675 | relation | 2/718440           | 29764 | RowExclusiveLock | t
(3 rows)

lukashak_daniil=> SELECT relation, locktype, virtualtransaction, pid, mode, granted FROM pg_locks WHERE locktype = 'relation';
 relation | locktype | virtualtransaction |  pid  |         mode          | granted
----------+----------+--------------------+-------+-----------------------+---------
    11673 | relation | 7/14148            | 29935 | AccessShareLock       | t
    16675 | relation | 6/63023            | 29807 | RowExclusiveLock      | f
    16675 | relation | 2/718441           | 29764 | ShareRowExclusiveLock | t
(3 rows)

lukashak_daniil=> SELECT relation, locktype, virtualtransaction, pid, mode, granted FROM pg_locks WHERE locktype = 'relation';
 relation | locktype | virtualtransaction |  pid  |         mode          | granted
----------+----------+--------------------+-------+-----------------------+---------
    11673 | relation | 7/14149            | 29935 | AccessShareLock       | t
    16675 | relation | 6/63024            | 29807 | RowShareLock          | t
    16675 | relation | 2/718442           | 29764 | ShareRowExclusiveLock | t
(3 rows)
```

Серед цих трьох комбінацій, тільки друга визиває режим очікування: IX не сумісний з SIX, тому друга транзакція переходить у режим очікування, поки перша не завершиться. Це ми й бачимо у таблиці транзакцій третього терміналу

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 COMMITED. Проаналізуйте реакцію СКБД на операцію UPDATE 2-ї транзакції (яка виконується пізніше) та дайте свої висновки.

Перший термінал:

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

lukashak_daniil=> START TRANSACTION;
START TRANSACTION
lukashak_daniil=> SET TRANSACTION ISOLATION LEVEL READ COMMITTED;
SET
lukashak_daniil=> SELECT * FROM teacher WHERE t_id = 1;
 t_id |         name         |         post
------+----------------------+----------------------
    1 | Mikailow             | docent
(1 row)

lukashak_daniil=> UPDATE teacher SET name = 'Lakerniy' WHERE t_id = 1;
UPDATE 1
lukashak_daniil=> SELECT * FROM teacher WHERE t_id = 1;
 t_id |         name         |         post
------+----------------------+----------------------
    1 | Lakerniy             | docent
(1 row)

lukashak_daniil=> COMMIT;
COMMIT
```

Другий термінал:

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

lukashak_daniil=> START TRANSACTION;
START TRANSACTION
lukashak_daniil=> SET TRANSACTION ISOLATION LEVEL READ COMMITTED;
SET
lukashak_daniil=> SELECT * FROM teacher WHERE t_id = 1;
 t_id |         name         |         post
------+----------------------+----------------------
    1 | Mikailow             | docent
(1 row)

lukashak_daniil=> UPDATE teacher SET name = 'Lakernaya' WHERE t_id = 1;
UPDATE 1
lukashak_daniil=> SELECT * FROM teacher WHERE t_id = 1;
 t_id |         name         |         post
------+----------------------+----------------------
    1 | Lakernaya            | docent
(1 row)

lukashak_daniil=> COMMIT;
COMMIT
```

UPDATE 2-ї транзакції переводить її у режим очікування, поки не завершиться перша, а потім виконую функцію UPDATE, затираючи нове значення, що було присвоєно першою транзакцією.

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.

Перший термінал:

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

lukashak_daniil=> START TRANSACTION;
START TRANSACTION
lukashak_daniil=> SET TRANSACTION ISOLATION LEVEL REPEATABLE READ;
SET
lukashak_daniil=> SELECT * FROM teacher WHERE t_id = 1;
 t_id |         name         |         post
------+----------------------+----------------------
    1 | Lakernaya            | docent
(1 row)

lukashak_daniil=> UPDATE teacher SET name = 'Hepingore' WHERE t_id = 1;
UPDATE 1
lukashak_daniil=> SELECT * FROM teacher WHERE t_id = 1;
 t_id |         name         |         post
------+----------------------+----------------------
    1 | Hepingore            | docent
(1 row)

lukashak_daniil=> COMMIT;
COMMIT
```

Другий термінал:

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

lukashak_daniil=> START TRANSACTION;
START TRANSACTION
lukashak_daniil=> SET TRANSACTION ISOLATION LEVEL REPEATABLE READ;
SET
lukashak_daniil=> SELECT * FROM teacher WHERE t_id = 1;
 t_id |         name         |         post
------+----------------------+----------------------
    1 | Lakernaya            | docent
(1 row)

lukashak_daniil=> UPDATE teacher SET name = 'Hepingora' WHERE t_id = 1;
ERROR:  could not serialize access due to concurrent update
lukashak_daniil=> SELECT * FROM teacher WHERE t_id = 1;
ERROR:  current transaction is aborted, commands ignored until end of transaction block
lukashak_daniil=> COMMIT;
ROLLBACK
```

UPDATE 2-ї транзакції переводить її у режим очікування, поки не завершиться перша, а потім визиває помилку через конкуруючий апдейт та завершує транзакцію.

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.

Перший термінал:

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

lukashak_daniil=> START TRANSACTION;
START TRANSACTION
lukashak_daniil=> SET TRANSACTION ISOLATION LEVEL SERIALIZABLE;
SET
lukashak_daniil=> SELECT * FROM teacher WHERE t_id = 1;
 t_id |         name         |         post
------+----------------------+----------------------
    1 | Hepingore            | docent
(1 row)

lukashak_daniil=> UPDATE teacher SET name = 'Lakerniy' WHERE t_id = 1;
UPDATE 1
lukashak_daniil=> SELECT * FROM teacher WHERE t_id = 1;
 t_id |         name         |         post
------+----------------------+----------------------
    1 | Lakerniy             | docent
(1 row)

lukashak_daniil=> COMMIT;
COMMIT
```

Другий термінал:

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

lukashak_daniil=> START TRANSACTION;
START TRANSACTION
lukashak_daniil=> SET TRANSACTION ISOLATION LEVEL SERIALIZABLE;
SET
lukashak_daniil=> SELECT * FROM teacher WHERE t_id = 1;
 t_id |         name         |         post
------+----------------------+----------------------
    1 | Hepingore            | docent
(1 row)

lukashak_daniil=> UPDATE teacher SET name = 'Lakernaya' WHERE t_id = 1;
ERROR:  could not serialize access due to concurrent update
lukashak_daniil=> SELECT * FROM teacher WHERE t_id = 1;
ERROR:  current transaction is aborted, commands ignored until end of transaction block
lukashak_daniil=> COMMIT;
ROLLBACK
```

UPDATE 2-ї транзакції переводить її у режим очікування, поки не завершиться перша, а потім визиває помилку через конкуруючий апдейт та завершує транзакцію.

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.

1.

Перший термінал:

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

lukashak_daniil=> START TRANSACTION;
START TRANSACTION
lukashak_daniil=> SET TRANSACTION ISOLATION LEVEL READ COMMITTED;
SET
lukashak_daniil=> SELECT * FROM teacher WHERE t_id = 1;
 t_id |         name         |         post
------+----------------------+----------------------
    1 | Lakerniy             | docent
(1 row)

lukashak_daniil=> UPDATE teacher SET name = 'Lakerniy' WHERE t_id = 1;
UPDATE 1
lukashak_daniil=> SELECT * FROM teacher WHERE t_id = 2;
 t_id |         name         |         post
------+----------------------+----------------------
    2 | Lukashak             | ne docent
(1 row)

lukashak_daniil=> UPDATE teacher SET name = 'Lakerniy' WHERE t_id = 2;
UPDATE 1
lukashak_daniil=> COMMIT;
COMMIT
```

Другий термінал:

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

lukashak_daniil=> START TRANSACTION;
START TRANSACTION
lukashak_daniil=> SET TRANSACTION ISOLATION LEVEL READ COMMITTED;
SET
lukashak_daniil=> SELECT * FROM teacher WHERE t_id = 2;
 t_id |         name         |         post
------+----------------------+----------------------
    2 | Lukashak             | ne docent
(1 row)

lukashak_daniil=> UPDATE teacher SET name = 'Lakerniy' WHERE t_id = 2;
UPDATE 1
lukashak_daniil=> SELECT * FROM teacher WHERE t_id = 1;
 t_id |         name         |         post
------+----------------------+----------------------
    1 | Lakerniy             | docent
(1 row)

lukashak_daniil=> UPDATE teacher SET name = 'Lakerniy' WHERE t_id = 1;
ERROR:  deadlock detected
DETAIL:  Process 13252 waits for ShareLock on transaction 2817; blocked by process 13243.
Process 13243 waits for ShareLock on transaction 2818; blocked by process 13252.
HINT:  See server log for query details.
CONTEXT:  while updating tuple (0,49) in relation "teacher"
```

Виникає тупикова ситуація, так як два процеса очікують завершення один одного:

- транзакція 2817 процесу 13243 змінює перший рядок;
- транзакція 2818 процесу 13252 змінює другий рядок;
- транзакція 2817 процесу 13243 намагається змінити другий рядок, але він зайнятий другою транзакцією, тому вона переходить у режим очікування;
- транзакція 2818 процесу 13252 намагається змінити перший рядок, але він зайнятий першою транзакцією, тому вона переходить у режим очікування;
- обидві транзакції очікують завершення один одного, тому виникає тупик та система зупиняє виконання другої транзакції.

2.

Перший термінал:

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

lukashak_daniil=>
lukashak_daniil=> START TRANSACTION;
START TRANSACTION
lukashak_daniil=> SET TRANSACTION ISOLATION LEVEL REPEATABLE READ;
SET
lukashak_daniil=> SELECT * FROM teacher WHERE t_id = 1;
 t_id |         name         |         post
------+----------------------+----------------------
    1 | Lakerniy             | docent
(1 row)

lukashak_daniil=> UPDATE teacher SET name = 'Hepingore' WHERE t_id = 1;
UPDATE 1
lukashak_daniil=> SELECT * FROM teacher WHERE t_id = 2;
 t_id |         name         |         post
------+----------------------+----------------------
    2 | Lakerniy             | ne docent
(1 row)

lukashak_daniil=> UPDATE teacher SET name = 'Hepingore' WHERE t_id = 2;
UPDATE 1
lukashak_daniil=> ROLLBACK;
ROLLBACK
```

Другий термінал:

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

lukashak_daniil=>
lukashak_daniil=> START TRANSACTION;
START TRANSACTION
lukashak_daniil=> SET TRANSACTION ISOLATION LEVEL REPEATABLE READ;
SET
lukashak_daniil=> SELECT * FROM teacher WHERE t_id = 2;
 t_id |         name         |         post
------+----------------------+----------------------
    2 | Lakerniy             | ne docent
(1 row)

lukashak_daniil=> UPDATE teacher SET name = 'Hepingore' WHERE t_id = 2;
UPDATE 1
lukashak_daniil=> SELECT * FROM teacher WHERE t_id = 1;
 t_id |         name         |         post
------+----------------------+----------------------
    1 | Lakerniy             | docent
(1 row)

lukashak_daniil=> UPDATE teacher SET name = 'Hepingore' WHERE t_id = 1;
ERROR:  deadlock detected
DETAIL:  Process 13687 waits for ShareLock on transaction 2821; blocked by process 13683.
Process 13683 waits for ShareLock on transaction 2822; blocked by process 13687.
HINT:  See server log for query details.
CONTEXT:  while updating tuple (0,9) in relation "teacher"
```

Виникає тупикова ситуація, так як два процеса очікують завершення один одного:

- транзакція 2821 процесу 13683 змінює перший рядок;
- транзакція 2822 процесу 13687 змінює другий рядок;
- транзакція 2821 процесу 13683 намагається змінити другий рядок, але він зайнятий другою транзакцією, тому вона переходить у режим очікування;
- транзакція 2822 процесу 13687 намагається змінити перший рядок, але він зайнятий першою транзакцією, тому вона переходить у режим очікування;
- обидві транзакції очікують завершення один одного, тому виникає тупик та система зупиняє виконання другої транзакції.

Висновок: ми дослідили поведінку процесів-транзакцій в базах даних та засоби керуванням ними через механізм блокування з використанням сучасних систем керування базами даних. Всі завдання були досить легкими, якщо добре розібратися в теоретичній базі.