

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
ДЕРЖАВНИЙ УНІВЕРСИТЕТ “ОДЕСЬКА ПОЛІТЕХНІКА”
ІНСТИТУТ КОМП’ЮТЕРНИХ СИСТЕМ
КАФЕДРА ІНФОРМАЦІЙНИХ СИСТЕМ

Лабораторна робота №10
з дисципліни «Операційні Системи»
Тема
«Керування процесами-транзакціями в базах даних.
Частина 2 »

Виконала:
студентка групи АІ-203
Мягких. А.М.

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

Завдання для 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 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:

1) Транзакція 1 та частина транзакції 2

Ми бачимо, що транзакція 1 (номер 3109) додала до таблиці university рядок 3 (за колонкою xmin). Зміни можливо побачити після успішного завершення 1 транзакції.

```
myagkih_arina@vpsj3leQ:~  
Last login: Thu Jun 3 07:15:46 2021 from 185-206-36-57.broadband.tenet.odessa.ua  
[myagkih_arina@vpsj3leQ ~]$ psql  
psql (9.5.25)  
Type "help" for help.  
  
myagkih_arina=> select txid_current();  
txid_current  
-----  
(1 row)  
4088  
  
myagkih_arina=> INSERT INTO university VALUES (3, 'NOUNA', 2005);  
INSERT 0 1  
  
myagkih_arina=> SELECT xmin, xmax, * FROM university;  
xmin | xmax | u_id | name | year  
-----  
3870 | 3943 | 0 | DUOP | 2021  
3943 | 3944 | 1 | KPI | 2002  
4089 | 0 | 3 | NOUNA | 2005  
(3 rows)  
  
myagkih_arina=>
```

```
myagkih_arina@vpsj3leQ:~  
Last login: Thu Jun 3 07:26:32 2021 from 185-206-36-57.broadband.tenet.odessa.ua  
[myagkih_arina@vpsj3leQ ~]$ psql  
psql (9.5.25)  
Type "help" for help.  
  
myagkih_arina=> START TRANSACTION;  
START TRANSACTION  
  
myagkih_arina=> SELECT xmin, xmax, * FROM university;  
xmin | xmax | u_id | name | year  
-----  
3870 | 3943 | 0 | DUOP | 2021  
3943 | 3944 | 1 | KPI | 2002  
(2 rows)  
  
myagkih_arina=> SELECT xmin, xmax, * FROM university;  
xmin | xmax | u_id | name | year  
-----  
3870 | 3943 | 0 | DUOP | 2021  
3943 | 3944 | 1 | KPI | 2002  
4089 | 0 | 3 | NOUNA | 2005  
(3 rows)  
  
myagkih_arina=>
```

2) Транзакція 3 та частина транзакції 2

Бачимо, що в колонці xmin для u_id = 3, транзакція 3 (номер 3110) намагається видалити рядок. Транзакція 3 скасовується, але значення xmax залишається незмінним.


```
myagkih_arina@vpsj3leQ:~  
3943 | 3944 | 1 | KPI | 2002  
4089 | 0 | 3 | NOUNA | 2005  
(3 rows)  
  
myagkih_arina=> COMMIT;  
WARNING: there is no transaction in progress  
COMMIT  
myagkih_arina=> START TRANSACTION;  
myagkih_arina=> DELETE FROM university WHERE u_id = 3;  
DELETE 1  
myagkih_arina=> ROLLBACK;  
ROLLBACK  
myagkih_arina=> START TRANSACTION;  
myagkih_arina=> UPDATE university SET name = 'ONMU' WHERE u_id = 3;  
UPDATE 1  
myagkih_arina=> COMMIT;  
COMMIT  
myagkih_arina=> START TRANSACTION;  
myagkih_arina=> LOCK TABLE university IN ROW EXCLUSIVE MODE;  
LOCK TABLE  
myagkih_arina=> [ ]  
  
myagkih_arina@vpsj3leQ:~  
3870 | 0 | 2 | DUOP | 2021  
3943 | 3944 | 1 | KPI | 2002  
4089 | 0 | 3 | NOUNA | 2005  
(3 rows)  
  
myagkih_arina=> SELECT xmin, xmax, * FROM university;  
xmin | xmax | u_id | name | year  
-----  
3870 | 0 | 2 | DUOP | 2021  
3943 | 3944 | 1 | KPI | 2002  
4089 | 4090 | 3 | NOUNA | 2005  
(3 rows)  
  
myagkih_arina=> SELECT xmin, xmax, * FROM university;  
xmin | xmax | u_id | name | year  
-----  
3870 | 0 | 2 | DUOP | 2021  
3943 | 3944 | 1 | KPI | 2002  
4091 | 0 | 3 | ONMU | 2005  
(3 rows)  
  
myagkih_arina=> LOCK TABLE university IN ROW SHARE MODE;  
LOCK TABLE  
myagkih_arina=> [ ]  
  
myagkih_arina@vpsj3leQ:~  
myagkih_arina=> SELECT relation, locktype, virtualtransaction, pid, mode, granted FROM pg_locks WHERE locktype = 'relation';  
relation | locktype | virtualtransaction | pid | mode | granted  
-----  
11673 | relation | 4/508252 | 27986 | AccessShareLock | t  
16888 | relation | 3/281168 | 25212 | AccessShareLock | t  
16888 | relation | 3/281168 | 25212 | RowShareLock | t  
16888 | relation | 2/5448831 | 25201 | RowExclusiveLock | t  
(4 rows)  
  
myagkih_arina=> [ ]
```

2) Блокування SIX-IX є несумісними, тому транзакція 2 чекає на кінець першої. У колонці granted стоїть f напроти IX.

```
myagkih_arina@vpsj3leQ:~  
Type "help" for help.  
myagkih_arina=> START TRANSACTION;  
START TRANSACTION  
myagkih_arina=> LOCK TABLE university IN SHARE ROW EXCLUSIVE MODE;  
LOCK TABLE  
myagkih_arina=> [ ]  
  
myagkih_arina@vpsj3leQ:~  
psql (9.5.25)  
Type "help" for help.  
myagkih_arina=> START TRANSACTION;  
START TRANSACTION  
myagkih_arina=> LOCK TABLE university IN ROW EXCLUSIVE MODE;  
LOCK TABLE  
myagkih_arina=> [ ]  
  
myagkih_arina@vpsj3leQ:~  
myagkih_arina=> SELECT relation, locktype, virtualtransaction, pid, mode, granted FROM pg_locks WHERE locktype = 'relation';  
relation | locktype | virtualtransaction | pid | mode | granted  
-----  
11673 | relation | 4/508282 | 29396 | AccessShareLock | t  
16888 | relation | 2/5448883 | 29316 | ShareRowExclusiveLock | t  
16888 | relation | 3/281220 | 29370 | RowExclusiveLock | f  
(3 rows)  
  
myagkih_arina=> [ ]
```

3) Блокування SIX-IS є сумісними, тому транзакції проходять без проблем. У колонці granted стоїть t

```
myagkih_arina@vpsj3leQ:~  
Type "help" for help.  
myagkih_arina=> START TRANSACTION;  
START TRANSACTION  
myagkih_arina=> LOCK TABLE university IN SHARE ROW EXCLUSIVE MODE;  
LOCK TABLE  
myagkih_arina=> [ ]  
  
myagkih_arina@vpsj3leQ:~  
Type "help" for help.  
myagkih_arina=> START TRANSACTION;  
START TRANSACTION  
myagkih_arina=> LOCK TABLE university IN ROW SHARE MODE;  
LOCK TABLE  
myagkih_arina=> [ ]  
  
myagkih_arina@vpsj3leQ:~  
myagkih_arina=> SELECT relation, locktype, virtualtransaction, pid, mode, granted FROM pg_locks WHERE locktype = 'relation';  
relation | locktype | virtualtransaction | pid | mode | granted  
-----  
11673 | relation | 4/508292 | 30038 | AccessShareLock | t  
16888 | relation | 2/5448895 | 30004 | ShareRowExclusiveLock | t  
16888 | relation | 3/281230 | 30021 | RowShareLock | t  
(3 rows)  
  
myagkih_arina=> [ ]
```

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

Підготувала транзакції, які було створено у завданні 3.1 рішення попередньої

лабораторної роботи, а саме, створила 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 Виконала роботу транзакцій при умові їх роботи на рівні ізоляції READ

COMMITTED.

Реалізує безпеку від брудного читання. Транзакції бачать лише зафіксовані зміни (командою commit), тому update 2 транзакції не бачить змінених даних. Допоки транзакція 1 не завершиться, T2 перебуває в стані WAIT

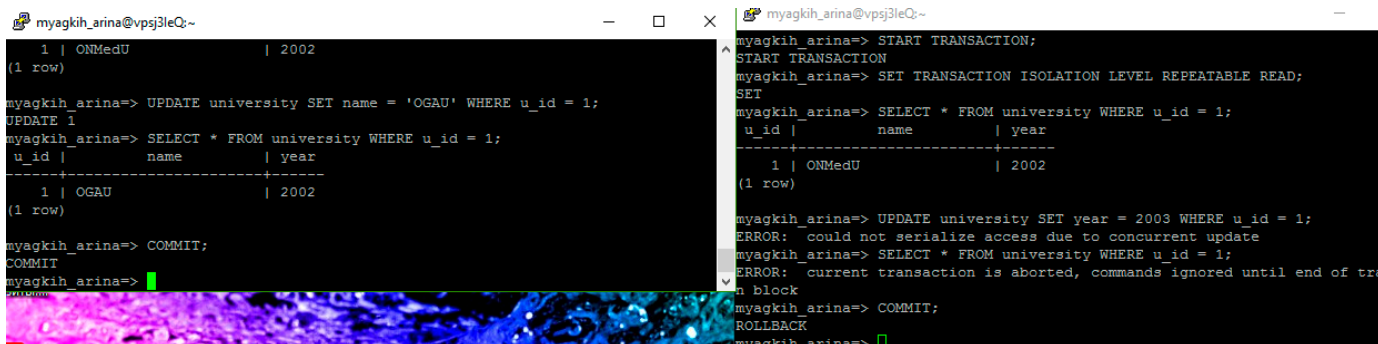
```
myagkih_arina@vpsj3leQ:~  
psql (9.5.25)  
Type "help" for help.  
  
myagkih_arina=> START TRANSACTION;  
START TRANSACTION  
myagkih_arina=> SET TRANSACTION ISOLATION LEVEL READ COMMITTED;  
SET
```

```
myagkih_arina@vpsj3leQ:~  
1 | KPI | 2002  
(1 row)  
  
myagkih_arina=> UPDATE university SET name = 'ONMedU' WHERE u_id = 1;  
UPDATE 1  
myagkih_arina=> SELECT * FROM university WHERE u_id = 1;  
 u_id | name | year  
-----+-----+-----  
1 | ONMedU | 2002  
(1 row)  
  
myagkih_arina=> COMMIT;  
COMMIT  
myagkih_arina=>   
  
myagkih_arina@vpsj3leQ:~  
myagkih_arina=> UPDATE university SET year = 1985 WHERE u_id = 1;  
UPDATE 1  
myagkih_arina=>   
myagkih_arina=> UPDATE university SET year = 1985 WHERE u_id = 1;  
UPDATE 1  
myagkih_arina=> SELECT * FROM university WHERE u_id = 1;  
 u_id | name | year  
-----+-----+-----  
1 | ONMedU | 1985  
(1 row)  
myagkih_arina=>
```

1.2 Повторила роботу транзакцій при умові їх роботи на рівні ізоляції REPEATABLE

READ.

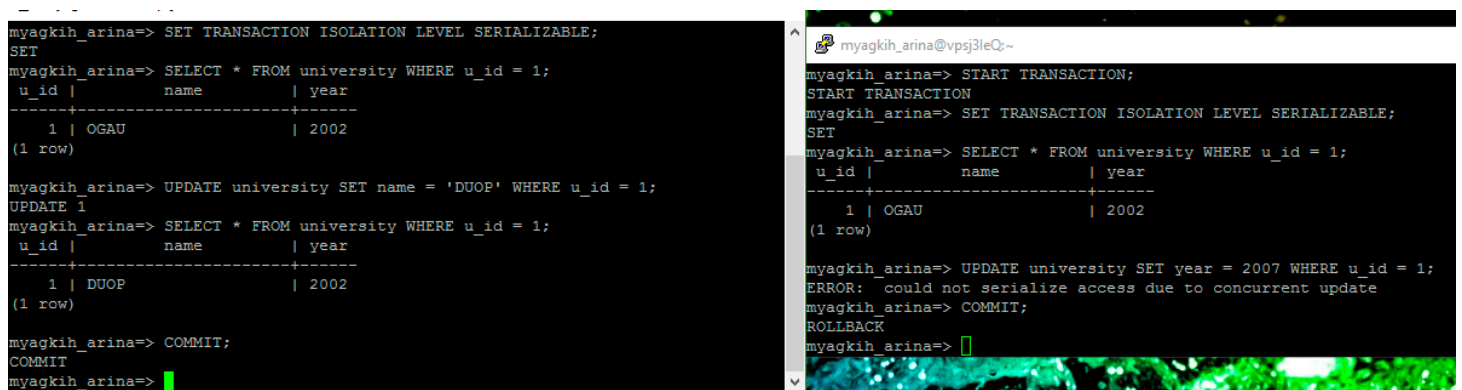
Реалізує безпеку від неповторного читання. Коли зміни першої транзакції були зафіксовані, то REPEATABLE READ відмінює другу транзакцію з помилкою ERROR: could not serialize access due to concurrent update, тому що REPEATABLE READ не може змінювати дані, змінені іншою транзакцією.



```
myagkih_arina@vpsj3leQ:~  
1 | ONMedU | 2002  
(1 row)  
myagkih_arina=> UPDATE university SET name = 'OGAU' WHERE u_id = 1;  
UPDATE 1  
myagkih_arina=> SELECT * FROM university WHERE u_id = 1;  
u_id | name | year  
-----  
1 | OGAU | 2002  
(1 row)  
myagkih_arina=> COMMIT;  
COMMIT  
myagkih_arina=>  
myagkih_arina@vpsj3leQ:~  
myagkih_arina=> START TRANSACTION;  
START TRANSACTION  
myagkih_arina=> SET TRANSACTION ISOLATION LEVEL REPEATABLE READ;  
SET  
myagkih_arina=> SELECT * FROM university WHERE u_id = 1;  
u_id | name | year  
-----  
1 | ONMedU | 2002  
(1 row)  
myagkih_arina=> UPDATE university SET year = 2003 WHERE u_id = 1;  
ERROR: could not serialize access due to concurrent update  
myagkih_arina=> SELECT * FROM university WHERE u_id = 1;  
ERROR: current transaction is aborted, commands ignored until end of transaction block  
myagkih_arina=> COMMIT;  
ROLLBACK  
myagkih_arina=>
```

1.3 Повторила роботу транзакцій при умові їх роботи на рівні ізоляції SERIALIZABLE.

Транзакції працюють так, ніби окрім них не існує. Так як 1 транзакція вже зафіксувала свої зміни, 2 завершилася без фіксації своїх змін. Рівні ізоляції Serializable заборонено виконувати паралельно зміни одних даних.



```
myagkih_arina=> SET TRANSACTION ISOLATION LEVEL SERIALIZABLE;  
SET  
myagkih_arina=> SELECT * FROM university WHERE u_id = 1;  
u_id | name | year  
-----  
1 | OGAU | 2002  
(1 row)  
myagkih_arina=> UPDATE university SET name = 'DUOP' WHERE u_id = 1;  
UPDATE 1  
myagkih_arina=> SELECT * FROM university WHERE u_id = 1;  
u_id | name | year  
-----  
1 | DUOP | 2002  
(1 row)  
myagkih_arina=> COMMIT;  
COMMIT  
myagkih_arina=>  
myagkih_arina@vpsj3leQ:~  
myagkih_arina=> START TRANSACTION;  
START TRANSACTION  
myagkih_arina=> SET TRANSACTION ISOLATION LEVEL SERIALIZABLE;  
SET  
myagkih_arina=> SELECT * FROM university WHERE u_id = 1;  
u_id | name | year  
-----  
1 | OGAU | 2002  
(1 row)  
myagkih_arina=> UPDATE university SET year = 2007 WHERE u_id = 1;  
ERROR: could not serialize access due to concurrent update  
myagkih_arina=> COMMIT;  
ROLLBACK  
myagkih_arina=>
```

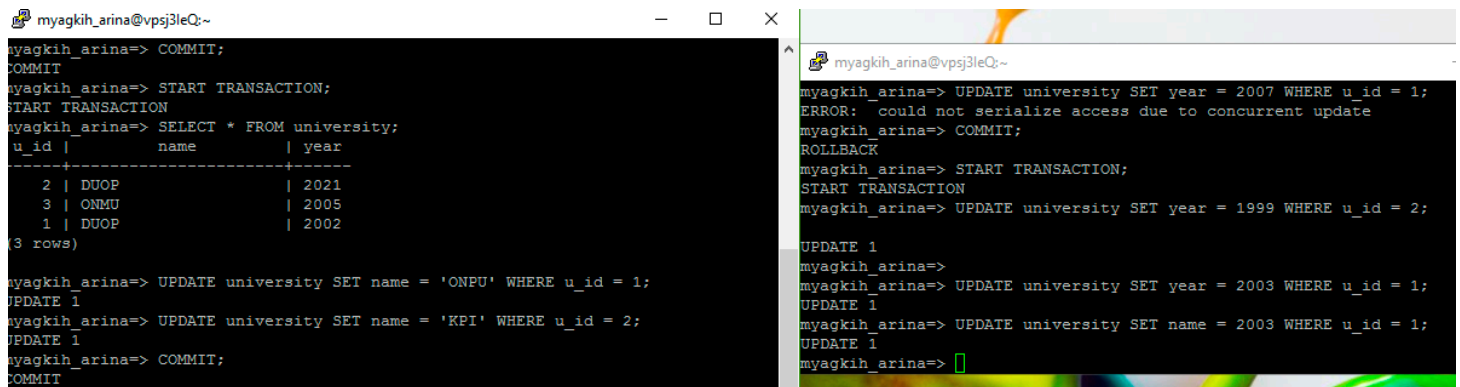
Завдання 4. Керування квазіпаралельним виконанням транзакцій при наявності тупикових ситуацій.

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

3.2 Виконала дві модифіковані транзакції.

Дві транзакції зависають і СКБД сама вирішує, яку з них видалити. Транзакція 1 заблокована та транзакція 2 заблокована (стан процесу Ss).

Кожна з них чекає на завершення попередньої, тому отримуємо цикл.



```
myagkih_arina@vpsj3leQ:~  
myagkih_arina=> COMMIT;  
COMMIT  
myagkih_arina=> START TRANSACTION;  
START TRANSACTION  
myagkih_arina=> SELECT * FROM university;  
u_id | name | year  
-----+-----+-----  
2 | DUOP | 2021  
3 | ONMU | 2005  
1 | DUOP | 2002  
(3 rows)  
myagkih_arina=> UPDATE university SET name = 'ONPU' WHERE u_id = 1;  
UPDATE 1  
myagkih_arina=> UPDATE university SET name = 'KPI' WHERE u_id = 2;  
UPDATE 1  
myagkih_arina=> COMMIT;  
COMMIT  
myagkih_arina@vpsj3leQ:~  
myagkih_arina=> UPDATE university SET year = 2007 WHERE u_id = 1;  
ERROR:  could not serialize access due to concurrent update  
myagkih_arina=> COMMIT;  
ROLLBACK  
myagkih_arina=> START TRANSACTION;  
START TRANSACTION  
myagkih_arina=> UPDATE university SET year = 1999 WHERE u_id = 2;  
UPDATE 1  
myagkih_arina=> UPDATE university SET year = 2003 WHERE u_id = 1;  
UPDATE 1  
myagkih_arina=> UPDATE university SET name = 2003 WHERE u_id = 1;  
UPDATE 1  
myagkih_arina=> 
```

Висновки: Під час виконання Лабораторної роботи № 10 мною було досліджено поведінку процесів-транзакцій в базах даних та засоби керування ними через механізм блокування з використанням сучасних систем керування базами даних.