МІНЕСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
Одеський національний політехнічний університет
Інститут комп’ютерних систем
Кафедра інформаційних систем

Звіт
Лабораторної роботи № 10
З предмету «Операційні системи»

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

Виконала:
Студентка групи АІ-205
Шаповалова Вікторія

Перевірили:
Блажко О. А.
Дрозд М. О.

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.

Вимоги до оформлення протоколу виконання лабораторної роботи
Протокол у електронному вигляді повинен мати наступну структуру
1) титульній аркуш з назвою дисципліни, теми лабораторної роботи, групи та ПІБ виконавця;
2) аркуш із завданням до лабораторної роботи;
3) аркуші з результатами виконання пунктів завдань:
- пункт із завданням;
- скріншот частини екрану з командами (з кольорами тексту білий фон/чорний тон);
- скріншот частини екрану з результатом виконання команди;
4) аркуш з висновками:
- перелік завдань, які були найскладнішими.

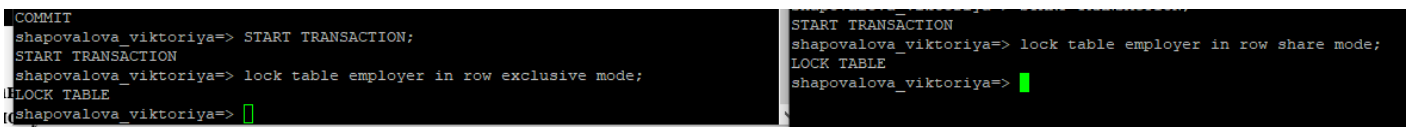
Завдання 1. Аналіз роботи багато версійного протоколу
Підготуйте чотири транзакції за прикладом з рисунку 2:
− T1 – отримання номеру транзакції, внесення нового рядка в таблицю та перегляд вмісту таблиці;
− T2 – постійний перегляд вмісту таблиці
− T3 – видалення рядку з наступною відміною цієї операції;
− 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.

```
shapovalova_viktoriya@vpsj3IeQ:~
login as: shapovalova_viktoriya
shapovalova_viktoriya@91.219.60.189's password:
Last login: Wed May 26 09:39:29 2021 from 46.250.25.175.pool.breezein.net
[shapovalova_viktoriya@vpsj3IeQ ~]$ psql
psql (9.5.25)
Type "help" for help.

shapovalova_viktoriya=> START TRANSACTION;
START TRANSACTION
shapovalova_viktoriya=> insert into employer values (1, 'Shapovalova', 300);
INSERT 0 1
shapovalova_viktoriya=> select xmin, xmax, * from employer;
 xmin | xmax | e_id |         name         | salary
------+------+------+----------------------+--------
 3940 |    0 |    2 | Sidorov              |    500
 3952 |    0 |    1 | Ivanov               |    200
 3952 |    0 |    1 | Ivanov               |    400
 3987 |    0 |    1 | Shapovalova          |    300
(4 rows)

shapovalova_viktoriya=> commit;
COMMIT
shapovalova_viktoriya=> START TRANSACTION;
START TRANSACTION
shapovalova_viktoriya=> delete from employer where e_id = 1;
DELETE 3
shapovalova_viktoriya=> select xmin, xmax, * from employer;
 xmin | xmax | e_id |         name         | salary
------+------+------+----------------------+--------
 3940 |    0 |    2 | Sidorov              |    500
(1 row)

shapovalova_viktoriya=> rollback;
ROLLBACK
shapovalova_viktoriya=> START TRANSACTION;
START TRANSACTION
shapovalova_viktoriya=> update employer set name = 'Ivanov' where e_id = 1;
UPDATE 3
shapovalova_viktoriya=> select xmin, xmax, * from employer;
 xmin | xmax | e_id |         name         | salary
------+------+------+----------------------+--------
 3940 |    0 |    2 | Sidorov              |    500
 3989 |    0 |    1 | Ivanov               |    200
 3989 |    0 |    1 | Ivanov               |    400
 3989 |    0 |    1 | Ivanov               |    300
(4 rows)

shapovalova_viktoriya=> commit;
COMMIT
shapovalova_viktoriya=>
```

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).

IX-IS



```
LOCK TABLE
shapovalova_viktoriya=> select relation,locktype,virtualtransaction,pid,mode,granted from pg_locks where locktype = 'relation';
 relation | locktype | virtualtransaction |  pid  |       mode       | granted
----------+----------+--------------------+-------+------------------+---------
    16906 | relation | 3/151905           |  1190 | RowShareLock     | t
    11673 | relation | 2/4016385          | 32050 | AccessShareLock  | t
    16906 | relation | 2/4016385          | 32050 | RowExclusiveLock | t
(3 rows)

shapovalova_viktoriya=>
```

SIX-IX

```
ROLLBACK
shapovalova_viktoriya=> START TRANSACTION;
START TRANSACTION
shapovalova_viktoriya=> lock table employer in share row exclusive mode;
LOCK TABLE
shapovalova_viktoriya=>
```

```
shapovalova_viktoriya=> START TRANSACTION;
START TRANSACTION
shapovalova_viktoriya=> lock table employer in row exclusive mode;
```

```
shapovalova_viktoriya=> select relation,locktype,virtualtransaction,pid,mode,gra
nted from pg_locks where locktype = 'relation';
 relation | locktype | virtualtransaction | pid  |         mode          | grant
ed
----------+----------+--------------------+------+-----------------------+------
---
    11673 | relation | 2/4016466          | 6650 | AccessShareLock       | t
    16906 | relation | 3/151948           | 6661 | RowExclusiveLock      | f
    16906 | relation | 2/4016466          | 6650 | ShareRowExclusiveLock | t
    16915 | relation | 4/464497           | 5919 | AccessExclusiveLock   | t
(4 rows)

shapovalova_viktoriya=>
[shapovalova_viktoriya@vpsj3IeQ ~]$ psql
```

SIX-IS

```
ROLLBACK
shapovalova_viktoriya=> START TRANSACTION;
START TRANSACTION
shapovalova_viktoriya=> lock table employer in share row exclusive mode;
LOCK TABLE
shapovalova_viktoriya=>
```

```
shapovalova_viktoriya=> START TRANSACTION;
START TRANSACTION
shapovalova_viktoriya=> lock table employer in row exclusive
```

```
shapovalova_viktoriya=> select relation,locktype,virtualtransaction,pid,mode,gra
nted from pg_locks where locktype = 'relation';
 relation | locktype | virtualtransaction | pid  |         mode          | grant
ed
----------+----------+--------------------+------+-----------------------+------
---
    11673 | relation | 2/4016467          | 6650 | AccessShareLock       | t
    16906 | relation | 3/151949           | 6661 | RowExclusiveLock      | f
    16906 | relation | 2/4016467          | 6650 | ShareRowExclusiveLock | t
    16915 | relation | 4/464497           | 5919 | AccessExclusiveLock   | t
(4 rows)

shapovalova_viktoriya=>
```

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).

```
shapovalova_viktoriya=> ABORT;
ROLLBACK
shapovalova_viktoriya=> START TRANSACTION;
START TRANSACTION
shapovalova_viktoriya=> SET TRANSACTION ISOLATION LEVEL READ COMMITTED;
SET
shapovalova_viktoriya=> SET TRANSACTION ISOLATION LEVEL READ COMMITTED;
SET
shapovalova_viktoriya=> select * from employer where e_id = 1;
 e_id |         name         | salary
------+----------------------+--------
    1 | Ivanov               |    200
    1 | Ivanov               |    400
    1 | Ivanov               |    300
(3 rows)

shapovalova_viktoriya=> update employer set name = 'Shapovalova' where e_id =1;
UPDATE 3
shapovalova_viktoriya=> commit;
COMMIT
shapovalova_viktoriya=>
```

```
shapovalova_viktoriya=> START TRANSACTION;
START TRANSACTION
shapovalova_viktoriya=> select * from employer where e_id = 1;
 e_id |         name         | salary
------+----------------------+--------
    1 | Ivanov               |    200
    1 | Ivanov               |    400
    1 | Ivanov               |    300
(3 rows)

shapovalova_viktoriya=> SET TRANSACTION ISOLATION LEVEL READ COMMITTED;
SET
shapovalova_viktoriya=> select * from employer where e_id = 1;
 e_id |         name         | salary
------+----------------------+--------
    1 | Ivanov               |    200
    1 | Ivanov               |    400
    1 | Ivanov               |    300
(3 rows)

shapovalova_viktoriya=> update employer set name = 'beer' where e_id =1;
UPDATE 3
shapovalova_viktoriya=> commit
shapovalova_viktoriya->
```

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).

```
shapovalova_viktoriya=> START TRANSACTION;
START TRANSACTION
shapovalova_viktoriya=> SET TRANSACTION ISOLATION LEVEL REPEATABLE READ;
SET
shapovalova_viktoriya=> select * from employer where e_id = 1;
 e_id |         name          | salary
------+-----------------------+--------
    1 | Shapovalova           |    200
    1 | Shapovalova           |    400
    1 | Shapovalova           |    300
(3 rows)

shapovalova_viktoriya=> update employer set name = 'beer' where e_id =1;
UPDATE 3
shapovalova_viktoriya=> select * from employer where e_id = 1;
 e_id |         name          | salary
------+-----------------------+--------
    1 | beer                  |    200
    1 | beer                  |    400
    1 | beer                  |    300
(3 rows)

shapovalova_viktoriya=> commit;
COMMIT
shapovalova_viktoriya=>
```

```
START TRANSACTION
shapovalova_viktoriya=> SET TRANSACTION ISOLATION LEVEL REPEATABLE READ;
SET
shapovalova_viktoriya=> select * from employer where e_id = 1;
 e_id |         name          | salary
------+-----------------------+--------
    1 | Shapovalova           |    200
    1 | Shapovalova           |    400
    1 | Shapovalova           |    300
(3 rows)

shapovalova_viktoriya=> select * from employer where e_id = 1;
 e_id |         name          | salary
------+-----------------------+--------
    1 | Shapovalova           |    200
    1 | Shapovalova           |    400
    1 | Shapovalova           |    300
(3 rows)

shapovalova_viktoriya=> select * from employer where e_id = 1;
 e_id |         name          | salary
------+-----------------------+--------
    1 | Shapovalova           |    200
    1 | Shapovalova           |    400
    1 | Shapovalova           |    300
(3 rows)

shapovalova_viktoriya=> select * from employer where e_id = 1;
 e_id |         name          | salary
------+-----------------------+--------
    1 | Shapovalova           |    200
    1 | Shapovalova           |    400
    1 | Shapovalova           |    300
(3 rows)

shapovalova_viktoriya=> update employer set name = 'beer' where e_id =1;
ERROR:  could not serialize access due to concurrent update
shapovalova_viktoriya=> commit;
ROLLBACK
shapovalova_viktoriya=>
```

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).

```
shapovalova_viktoriya=>  START TRANSACTION;
START TRANSACTION
shapovalova_viktoriya=> SET TRANSACTION ISOLATION LEVEL  SERIALIZABLE;
SET
shapovalova_viktoriya=> select * from employer where e_id = 1;
 e_id |         name          | salary
------+-----------------------+--------
    1 | beer                  |    200
    1 | beer                  |    400
    1 | beer                  |    300
(3 rows)

shapovalova_viktoriya=> update employer set name = 'Vikysya' where e_id =1;
UPDATE 3
shapovalova_viktoriya=> select * from employer where e_id = 1;
 e_id |         name          | salary
------+-----------------------+--------
    1 | Vikysya               |    200
    1 | Vikysya               |    400
    1 | Vikysya               |    300
(3 rows)

shapovalova_viktoriya=> commit;
COMMIT
shapovalova_viktoriya=>
```

```
ROLLBACK
shapovalova_viktoriya=> START TRANSACTION;
START TRANSACTION
shapovalova_viktoriya=> SET TRANSACTION ISOLATION LEVEL  SERIALIZABLE;
SET
shapovalova_viktoriya=> select * from employer where e_id = 1;
 e_id |         name         | salary
------+----------------------+--------
    1 | beer                 |    200
    1 | beer                 |    400
    1 | beer                 |    300
(3 rows)

shapovalova_viktoriya=> update employer set name = 'Ivasya' where e_id =1;
ERROR:  could not serialize access due to concurrent update
shapovalova_viktoriya=> commit;
ROLLBACK
shapovalova_viktoriya=>
```

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

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

```
shapovalova_viktoriya=> commit;
COMMIT
shapovalova_viktoriya=> START TRANSACTION;
START TRANSACTION
shapovalova_viktoriya=> update employer set name = 'Shapovalova' where e_id =1;
UPDATE 3
shapovalova_viktoriya=> update employer set name = 'Shapovalova' where e_id =2;
UPDATE 1
shapovalova_viktoriya=>
```

```
shapovalova_viktoriya=> update employer set name = 'Ivasya' where e_id =1;
ERROR:  could not serialize access due to concurrent update
shapovalova_viktoriya=> commit;
ROLLBACK
shapovalova_viktoriya=> START TRANSACTION;
START TRANSACTION
shapovalova_viktoriya=> update employer set name = 'Vikysya' where e_id =2;
UPDATE 1
shapovalova_viktoriya=> update employer set name = 'Vikysya' where e_id =1;
ERROR:  deadlock detected
DETAIL:  Process 6661 waits for ShareLock on transaction 3997; blocked by proce
s 6650.
Process 6650 waits for ShareLock on transaction 3998; blocked by process 6661.
HINT:  See server log for query details.
CONTEXT:  while updating tuple (0,29) in relation "employer"
shapovalova_viktoriya=>
```

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.

```
ps -u postgres -o pid,ppid,stat,cmd
```

```
[local] idle in transaction
[local] idle in transaction (aborted)
```

```
UPDATE 1
shapovalova_viktoriya=> select txid_current();
 txid_current
--------------
         4000
(1 row)

shapovalova_viktoriya=>
```

```
ERROR:  current transaction is aborted, commands ignored until end of transaction block
bogachik_egor=>
```

Висновок: в ході роботи ми дослідити поведінку процесів-транзакцій в базах даних та засоби керуванням ними через механізм блокування з використанням сучасних систем керування базами даних. Найскладнішими були перші завдання.